Panduan Pengguna

AWS Glue

AWS Glue: Panduan Pengguna

Table of Contents

Apa itu AWS Glue?

AWS Glueadalah layanan integrasi data tanpa server yang memudahkan pengguna analitik untuk menemukan, menyiapkan, memindahkan, dan mengintegrasikan data dari berbagai sumber. Anda dapat menggunakannya untuk analitik, machine learning, dan pengembangan aplikasi. Ini juga mencakup produktivitas tambahan dan perkakas operasi data untuk menulis, menjalankan pekerjaan, dan mengimplementasikan alur kerja bisnis.

DenganAWS Glue, Anda dapat menemukan dan terhubung ke lebih dari 70 sumber data yang beragam dan mengelola data Anda dalam katalog data terpusat. Anda dapat membuat, menjalankan, dan memantau pipeline ekstrak, mengubah, dan memuat (ETL) secara visual untuk memuat data ke dalam data lake Anda. Selain itu, Anda dapat segera mencari dan menanyakan data katalog menggunakan Amazon Athena, Amazon EMR, dan Amazon Redshift Spectrum.

AWS Gluemengkonsolidasikan kemampuan integrasi data utama ke dalam satu layanan. Ini termasuk penemuan data, ETL modern, pembersihan, transformasi, dan katalog terpusat. Ini juga tanpa server, yang berarti tidak ada infrastruktur untuk dikelola. Dengan dukungan fleksibel untuk semua beban kerja seperti ETL, ELT, dan streaming dalam satu layanan, AWS Glue mendukung pengguna di berbagai beban kerja dan jenis pengguna.

Juga, AWS Glue memudahkan untuk mengintegrasikan data di seluruh arsitektur Anda. Ini terintegrasi dengan layanan AWS analitik dan danau data Amazon S3. AWS Gluememiliki antarmuka integrasi dan alat penulisan pekerjaan yang mudah digunakan untuk semua pengguna, dari pengembang hingga pengguna bisnis, dengan solusi yang disesuaikan untuk beragam keahlian teknis.

Dengan kemampuan untuk menskalakan sesuai permintaan, AWS Glue membantu Anda fokus pada aktivitas bernilai tinggi yang memaksimalkan nilai data Anda. Ini menskalakan untuk ukuran data apa pun, dan mendukung semua tipe data dan varians skema. Untuk meningkatkan kelincahan dan mengoptimalkan biaya, AWS Glue menyediakan ketersediaan dan pay-as-you-go penagihan tinggi bawaan.

Untuk informasi harga, lihat Harga AWS Glue.

AWS Glue Studio

AWS Glue Studioadalah antarmuka grafis yang memudahkan untuk membuat, menjalankan, dan memantau pekerjaan integrasi data diAWS Glue. Anda dapat menyusun alur kerja transformasi data

secara visual dan menjalankannya dengan mulus di mesin ETL tanpa server berbasis Apache Spark. AWS Glue

DenganAWS Glue Studio, Anda dapat membuat dan mengelola pekerjaan yang mengumpulkan, mengubah, dan membersihkan data. Anda juga dapat menggunakan AWS Glue Studio untuk memecahkan masalah dan mengedit skrip pekerjaan.

Topik

Fitur AWS Glue

AWS Gluefitur terbagi dalam tiga kategori utama:

- Temukan dan atur data
- Mengubah, menyiapkan, dan membersihkan data untuk analisis
- Membangun dan memantau jaringan data

Temukan dan atur data

- Menyatukan dan mencari di beberapa penyimpanan data — Simpan, indeks, dan cari di berbagai sumber data dan sink dengan membuat katalog semua data Anda. AWS
- Temukan data secara otomatis — Gunakan AWS Glue crawler untuk secara otomatis menyimpulkan informasi skema dan mengintegrasikannya ke dalam file Anda. AWS Glue Data Catalog
- Kelola skema dan izin — Validasi dan kontrol akses ke database dan tabel Anda.
- Connect ke berbagai sumber data — Manfaatkan beberapa sumber data, baik di tempat maupun di tempat AWS, menggunakan AWS Glue koneksi untuk membangun data lake Anda.

Mengubah, menyiapkan, dan membersihkan data untuk analisis

- Transformasi data secara visual dengan antarmuka kanvas pekerjaan - Tentukan proses ETL Anda di editor pekerjaan visual dan buat kode secara otomatis untuk mengekstrak, mengubah, dan memuat data Anda.
- Bangun jaringan ETL yang kompleks dengan penjadwalan pekerjaan sederhana — Memanggil AWS Glue pekerjaan sesuai jadwal, sesuai permintaan, atau berdasarkan suatu acara.
- Bersihkan dan ubah data streaming dalam perjalanan - Aktifkan konsumsi data berkelanjutan, dan bersihkan serta ubah dalam perjalanan. Ini membuatnya tersedia untuk analisis dalam hitungan detik di penyimpanan data target Anda.
- Hapus duplikat dan bersihkan data dengan pembelajaran mesin bawaan — Bersihkan dan siapkan data Anda untuk dianalisis tanpa menjadi ahli pembelajaran mesin dengan menggunakan fitur ini. `FindMatches` Fitur ini menghapus duplikasi dan menemukan catatan yang tidak cocok satu sama lain.
- Notebook pekerjaan bawaan — notebook AWS Glue pekerjaan menyediakan notebook tanpa server dengan pengaturan minimal AWS Glue sehingga Anda dapat memulai dengan cepat.
- Edit, debug, dan uji kode ETL — Dengan sesi AWS Glue interaktif, Anda dapat mengeksplorasi dan menyiapkan data secara interaktif. Anda dapat menjelajahi, bereksperimen, dan memproses data secara interaktif menggunakan IDE atau notebook pilihan Anda.
- Menentukan, mendeteksi, dan memulihkan data sensitif — deteksi data AWS Glue sensitif memungkinkan Anda menentukan, mengidentifikasi, dan memproses data sensitif di pipeline data Anda dan di data lake Anda.

Membangun dan memantau jaringan data

- Secara otomatis menskalakan berdasarkan beban kerja — Menskalakan sumber daya secara dinamis ke atas dan ke bawah berdasarkan beban kerja. Ini menugaskan pekerja untuk pekerjaan hanya ketika dibutuhkan.
- Otomatiskan pekerjaan dengan pemicu berbasis peristiwa — Mulai crawler atau AWS Glue pekerjaan dengan pemicu berbasis peristiwa, dan rancang rantai pekerjaan dan perayap dependen.
- Jalankan dan pantau pekerjaan - AWS Glue Jalankan pekerjaan dengan mesin pilihan Anda, Spark atau Ray. Pantau mereka dengan alat pemantauan otomatis, wawasan AWS Glue pekerjaan, dan AWS CloudTrail. Tingkatkan pemantauan Anda terhadap pekerjaan yang didukung Spark dengan Apache Spark UI.
- Tentukan alur kerja untuk ETL dan aktivitas integrasi — Tentukan alur kerja untuk ETL dan aktivitas integrasi untuk beberapa crawler, pekerjaan, dan pemicu.

Belajar tentang inovasi di AWS Glue

Pelajari tentang inovasi terbaru AWS Glue dan dengarkan bagaimana pelanggan menggunakan AWS Glue untuk memungkinkan persiapan data swalayan di seluruh organisasi mereka.

Pelajari bagaimana skala pelanggan AWS Glue melampaui pengaturan tradisional dan cara mereka mengonfigurasi AWS Glue pemantauan dan kinerja pekerjaan.

Mulai menggunakan AWS Glue

Kami menyarankan Anda memulai dengan bagian berikut:

- Ikhtisar penggunaan AWS Glue
- AWS Gluekonsep
- Menyiapkan izin IAM untuk AWS Glue
- Memulai dengan AWS Glue Data Catalog
- Lowongan kerja Authoring di AWS Glue
- Memulai dengan sesi AWS Glue interaktif
- Orkestrasi di AWS Glue

Mengakses AWS Glue

Anda dapat membuat, melihat, dan mengelola AWS Glue pekerjaan Anda menggunakan antarmuka berikut:

- AWS Gluekonsol — Menyediakan antarmuka web bagi Anda untuk membuat, melihat, dan mengelola AWS Glue pekerjaan Anda. Untuk mengakses konsol tersebut, lihat AWS Glue.
- AWS Glue Studio— Menyediakan antarmuka grafis bagi Anda untuk membuat dan mengedit AWS Glue pekerjaan Anda secara visual. Untuk informasi lebih lanjut, lihat Apa itu AWS Glue Studio.
- AWS Gluebagian AWS CLI Referensi - Menyediakan AWS CLI perintah yang dapat Anda gunakanAWS Glue. Untuk informasi selengkapnya, lihat AWS CLI Referensi untuk AWS Glue.
- AWS GlueAPI - Menyediakan referensi API lengkap untuk pengembang. Untuk informasi selengkapnya, lihat AWS GlueAPI.

Layanan terkait

Pengguna AWS Glue juga menggunakan:

- AWS Lake Formation— Layanan yang merupakan lapisan otorisasi yang menyediakan kontrol akses berbutir halus ke sumber daya di. AWS Glue Data Catalog
- AWS Glue DataBrew— Alat persiapan data visual yang dapat Anda gunakan untuk membersihkan dan menormalkan data tanpa menulis kode apa pun.

AWS Glue: Cara kerjanya

AWS Glue menggunakan layanan AWS lainnya untuk mengatur tugas ETL (extract, transform, and load) Anda untuk membangun gudang data dan danau data dan menghasilkan pengaliran output. AWS Glue memanggil operasi API untuk mengubah data Anda, membuat log waktu aktif, menyimpan logika tugas Anda, dan membuat pemberitahuan untuk membantu Anda memantau tugas Anda berjalan. Konsol AWS Glue menghubungkan layanan ini ke dalam aplikasi terkelola, sehingga Anda dapat fokus pada menciptakan dan memantau tugas ETL Anda. Konsol melakukan operasi pengembangan administratif dan tugas atas nama Anda. Anda menyediakan kredensial dan properti lainnya bagi AWS Glue untuk mengakses sumber data Anda dan menulis ke target data Anda.

AWS Glue mengurus penyediaan dan mengelola sumber daya yang diperlukan untuk menjalankan beban kerja Anda. Anda tidak perlu membuat infrastruktur untuk alat ETL karena AWS Glue telah melakukannya untuk Anda. Ketika sumber daya diperlukan, untuk mengurangi waktu pemulaian, AWS Glue menggunakan sebuah instans dari kolam instans hangat untuk menjalankan beban kerja Anda.

Dengan AWS Glue, Anda membuat tugas menggunakan definisi tabel di Katalog Data Anda. Tugas terdiri dari skrip yang berisi logika pemrograman yang melakukan transformasi. Anda menggunakan pemicu untuk memulai tugas baik pada jadwal atau sebagai akibat dari peristiwa tertentu. Anda menentukan di mana data target Anda berada dan data sumber mana yang mengisi target Anda. Dengan input dari Anda, AWS Glue menghasilkan kode yang diperlukan untuk men-transformasi data Anda dari sumber ke target. Anda juga dapat menyediakan skrip di konsol atau API AWS Glue untuk memproses data Anda.

Sumber data dan tujuan

AWS Glueuntuk Spark memungkinkan Anda untuk membaca dan menulis data dari berbagai sistem dan database termasuk:

- Amazon S3
- Amazon DynamoDB
- Amazon Redshift
- Amazon Relational Database Service (Amazon RDS)
- Basis dapat diakses JDBC pihak ketiga
- MongoDB dan Amazon DocumentDB (dengan kompatibilitas MongoDB)
- Konektor pasar lainnya dan plugin Apache Spark

Aliran data

AWS Glueuntuk Spark dapat mengalirkan data dari sistem berikut:

- Amazon Kinesis Data Streams
- Apache Kafka

AWS Glue tersedia di beberapa Wilayah AWS. Untuk informasi selengkapnya, lihat AWSWilayah dan Titik Akhir di. Referensi Umum Amazon Web

Topik

Pekerjaan ETL tanpa server berjalan secara terpisah

AWS Gluemenjalankan pekerjaan ETL Anda di lingkungan tanpa server dengan mesin pilihan Anda, Spark atau Ray. AWS Gluemenjalankan pekerjaan ini pada sumber daya virtual yang disediakan dan dikelola di akun layanannya sendiri.

AWS Glue dirancang untuk melakukan hal berikut:

- Memisahkan data pelanggan.
- Melindungi data pelanggan saat transit dan saat tidak aktif.
- Mengakses data pelanggan hanya jika diperlukan dalam menanggapi permintaan pelanggan, menggunakan kredensial sementara, melakukan scope-down pada kredensial, atau dengan persetujuan pelanggan, pada IAM role di akun mereka.

Selama penyediaan tugas ETL, Anda menyediakan sumber data input dan target data output di virtual private cloud (VPC) Anda. Selain itu, Anda memberikan IAM role, VPC ID, subnet ID, dan grup keamanan yang diperlukan untuk mengakses sumber data dan target. Untuk setiap tuple (ID akun

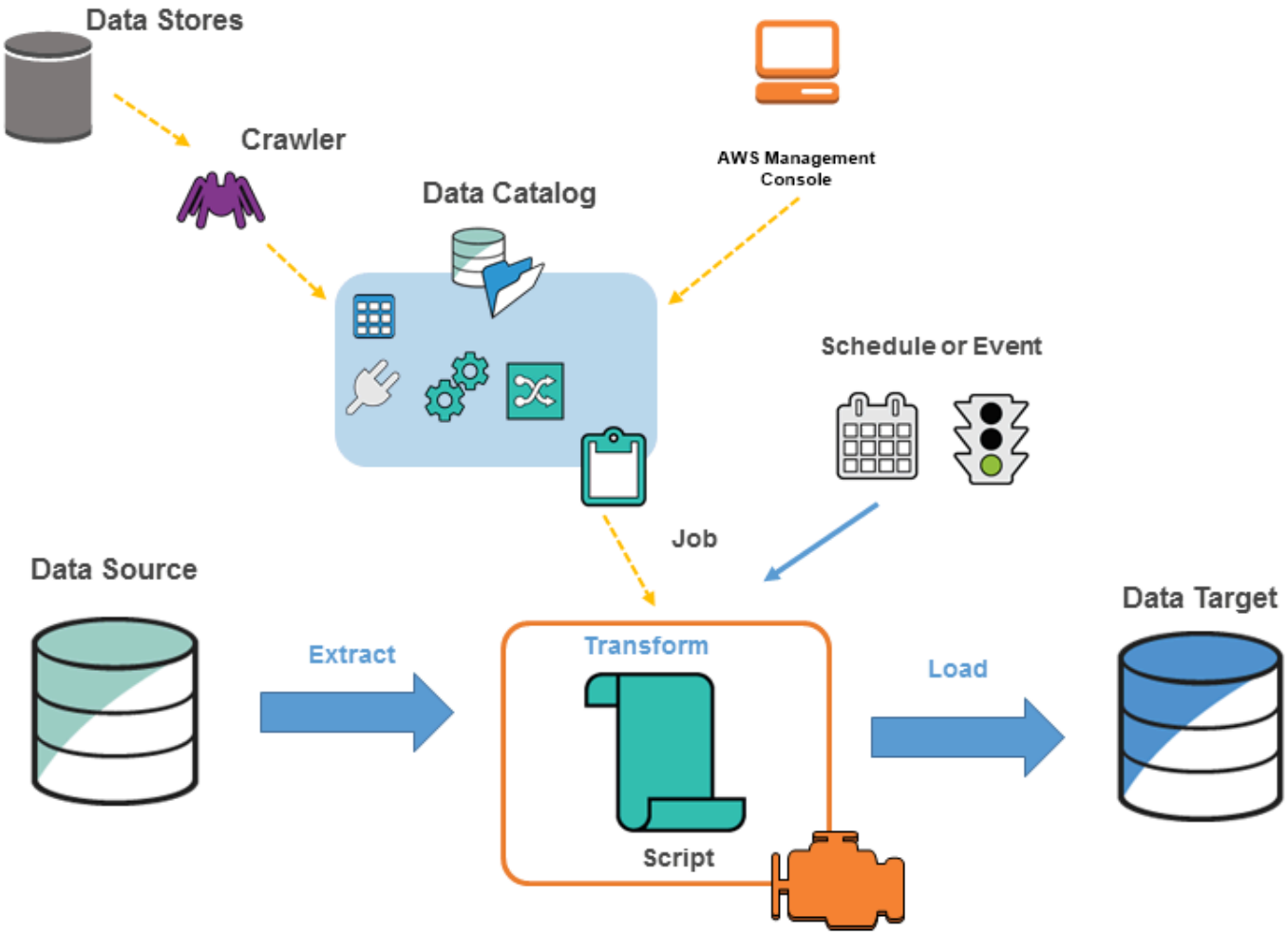
pelanggan, peran IAM, ID subnet, dan grup keamanan), AWS Glue menciptakan lingkungan baru yang terisolasi di tingkat jaringan dan manajemen dari semua lingkungan lain di dalam akun layanan. AWS Glue

AWS Glue membuat antarmuka jaringan elastis di subnet Anda dengan menggunakan alamat IP privat. Pekerjaan menggunakan antarmuka jaringan elastis ini untuk mengakses sumber data dan target data Anda. Lalu lintas masuk, keluar, dan di dalam lingkungan menjalankan pekerjaan diatur oleh VPC dan kebijakan jaringan Anda dengan satu pengecualian: Panggilan yang dilakukan AWS Glue ke perpustakaan dapat mem-proxy lalu lintas AWS Glue ke operasi API melalui VPC. AWS Glue Semua panggilan API AWS Glue dicatat; dengan demikian, pemilik data dapat meng-audit akses API dengan mengaktifkan AWS CloudTrail, yang mengirimkan log audit ke akun Anda.

AWS Gluelingkungan terkelola yang menjalankan pekerjaan ETL Anda dilindungi dengan praktik keamanan yang sama diikuti oleh AWS layanan lain. Untuk gambaran umum praktik dan tanggung jawab keamanan bersama, lihat laporan resmi Pengantar Proses Keamanan AWS.

Konsep AWS Glue

Diagram berikut menunjukkan arsitektur dari lingkungan AWS Glue.



Anda mendefinisikan tugas di AWS Glue untuk menyelesaikan tugas yang diperlukan untuk melakukan tugas extract, transform, and load (ETL) dari sumber data ke target data. Anda biasanya akan melakukan tindakan-tindakan berikut:

- Untuk sumber penyimpanan data, Anda menentukan crawler agar mengisi AWS Glue Data Catalog Anda dengan definisi tabel metadata. Anda mengarahkan crawler Anda di sebuah penyimpanan data, dan crawler menciptakan definisi tabel dalam Katalog Data. Untuk sumber streaming, Anda secara manual menentukan tabel Katalog Data dan menentukan properti aliran data.

  Selain definisi tabel, AWS Glue Data Catalog berisi metadata lain yang diperlukan untuk menentukan tugas ETL. Anda menggunakan metadata ini ketika Anda menentukan tugas untuk mengubah data Anda.

- AWS Glue dapat menghasilkan skrip untuk mengubah data Anda. Atau, Anda dapat memberikan skrip di konsol atau API AWS Glue.
- Anda dapat menjalankan tugas Anda sesuai permintaan, atau Anda dapat mengaturnya untuk memulai ketika pemicu yang ditentukan terjadi. Pemicu bisa menjadi jadwal berbasis waktu atau peristiwa.

  Saat tugas Anda berjalan, skrip mengekstrak data dari sumber data Anda, mengubah data, dan memasukkannya ke target data Anda. Skrip berjalan di lingkungan Apache Spark di AWS Glue.

> **Important**
>
> Tabel dan basis data di AWS Glue adalah obyek di AWS Glue Data Catalog. Mereka berisi metadata; mereka tidak berisi data dari penyimpanan data.

Data berbasis teks, seperti CSV, harus dikodekan dalam **`UTF-8`** untuk AWS Glue agar bisa berhasil diproses. Untuk informasi selengkapnya, lihat UTF-8 di Wikipedia.

Terminologi AWS Glue

AWS Gluebergantung pada interaksi beberapa komponen untuk membuat dan mengelola alur kerja ekstrak, transformasi, dan beban (ETL) Anda.

AWS Glue Data Catalog

Penyimpanan metadata persisten di AWS Glue. Ia berisi definisi tabel, definisi tugas, dan informasi kontrol lainnya untuk mengelola lingkungan AWS Glue Anda. Setiap akun AWS memiliki satu AWS Glue Data Catalog per wilayah.

Pengklasifikasi

Tentukan skema data Anda. AWS Glue menyediakan pengklasifikasi untuk jenis file umum, seperti CSV, JSON, AVRO, XML, dan lain-lain. Ia juga menyediakan pengklasifikasi untuk sistem pengelolaan basis data relasional umum dengan menggunakan koneksi JDBC. Anda dapat menulis pengklasifikasi Anda sendiri dengan menggunakan pola grok atau dengan menentukan tag baris dalam sebuah dokumen XML.

Koneksi

Sebuah objek Katalog Data yang berisi properti yang diperlukan untuk connect ke penyimpanan data tertentu.

Crawler

Program yang terhubung ke penyimpanan data (sumber atau target), berlangsung melalui daftar prioritas pengklasifikasi untuk menentukan skema untuk data Anda, dan kemudian membuat tabel metadata di AWS Glue Data Catalog.

Basis Data

Satu set definisi tabel Katalog Data terkait diatur ke dalam grup logis.

Penyimpanan data, sumber data, target data

Sebuah penyimpanan data adalah repositori untuk menyimpan data Anda secara terus-menerus. Contohnya meliputi bucket Amazon S3 dan basis data relasional. Sebuah sumber data adalah penyimpanan data yang digunakan sebagai masukan untuk proses atau transformasi. Sebuah target data adalah penyimpanan data yang padanya dituliskan proses atau transformasi.

Titik akhir pengembangan

Lingkungan yang dapat Anda gunakan untuk mengembangkan dan menguji skrip ETL AWS Glue Anda.

Bingkai Dinamis

Sebuah tabel terdistribusi yang mendukung data bersarang seperti struktur dan rangkaian string. Setiap catatan adalah swa-deskripsi, yang dirancang untuk fleksibilitas skema dengan data semi-terstruktur. Setiap catatan berisi data dan skema yang menggambarkan data tersebut. Anda dapat menggunakan frame dinamis dan Apache Spark DataFrames di skrip ETL Anda, dan mengonversinya. Bingkai dinamis menyediakan satu set transformasi lanjutan untuk pembersihan data dan ETL.

Tugas

Logika bisnis yang diperlukan untuk melakukan tugas ETL. Ia terdiri dari skrip transformasi, sumber data, dan target data. Eksekusi tugas dimulai oleh pemicu yang dapat dijadwalkan atau dipicu oleh peristiwa.

Dasbor performa tugas

AWS Gluemenyediakan dasbor run komprehensif untuk pekerjaan ETL Anda. Dasbor menampilkan informasi tentang eksekusi tugas dalam kerangka waktu tertentu.

Antarmuka notebook

Pengalaman notebook yang disempurnakan dengan penyiapan sekali klik untuk memudahkan penulisan pekerjaan dan eksplorasi data. Notebook dan koneksi dikonfigurasi secara otomatis untuk Anda. Anda dapat menggunakan antarmuka notebook berdasarkan Jupyter Notebook untuk mengembangkan, men-debug, dan menyebarkan skrip dan alur kerja secara interaktif menggunakan infrastruktur Apache Spark ETL tanpa server. AWS Glue Anda juga dapat melakukan kueri ad-hoc, analisis data, dan visualisasi (misalnya, tabel dan grafik) di lingkungan notebook.

Skrip

Kode yang mengekstrak data dari sumber, mengubahnya, dan memuatnya menjadi target. AWS Gluemenghasilkan PySpark atau skrip Scala.

Tabel

Definisi metadata yang mewakili data Anda. Baik data Anda berada dalam file Amazon Simple Storage Service (Amazon S3), atau tabel Amazon Relational Database Service (Amazon RDS), atau kumpulan data lainnya, tabel mendefinisikan skema data Anda. Sebuah tabel di AWS Glue Data Catalog terdiri dari nama-nama kolom, definisi tipe data, informasi partisi, dan metadata lainnya tentang set data dasar. Skema data Anda direpresentasikan dalam definisi tabel AWS Glue. Data aktual tetap di penyimpanan data aslinya, baik itu dalam file atau tabel basis data relasional. AWS Glue membuat katalog atas tabel file dan tabel basis data relasional Anda di AWS Glue Data Catalog. Mereka digunakan sebagai sumber dan target ketika Anda membuat tugas ETL.

Transformasi

Logika kode yang digunakan untuk memanipulasi data Anda ke dalam sebuah format yang berbeda.

Pemicu

Memulai tugas ETL. Pemicu dapat didefinisikan berdasarkan waktu yang dijadwalkan atau peristiwa.

Editor tugas visual

Editor pekerjaan visual adalah antarmuka grafis yang memudahkan untuk membuat, menjalankan, dan memantau pekerjaan ekstrak, transformasi, dan pemuatan (ETL) diAWS Glue. Anda dapat menyusun alur kerja transformasi data secara visual, menjalankannya dengan mulus di AWS Glue mesin ETL tanpa server berbasis Apache Spark, dan memeriksa skema dan hasil data di setiap langkah pekerjaan.

Pekerja

DenganAWS Glue, Anda hanya membayar waktu yang dibutuhkan pekerjaan ETL Anda untuk berjalan. Tidak ada sumber daya untuk dikelola, tidak ada biaya di muka, dan Anda tidak dikenakan biaya untuk waktu startup atau shutdown. Anda dikenakan tarif per jam berdasarkan jumlah Unit Pemrosesan Data (atau DPU) yang digunakan untuk menjalankan pekerjaan ETL Anda. Satu Data Processing Unit (DPU) juga disebut sebagai pekerja. AWS Gluedilengkapi dengan tiga jenis pekerja untuk membantu Anda memilih konfigurasi yang memenuhi latensi pekerjaan dan persyaratan biaya Anda. Pekerja datang dalam konfigurasi Standar, G.1X, G.2X, dan G.025X.

AWS Glue komponen

AWS Glue menyediakan konsol dan operasi API untuk mengatur dan mengelola beban kerja extract, transform, and load (ETL) Anda. Anda dapat menggunakan operasi API melalui beberapa SDK spesifik-bahasa dan AWS Command Line Interface (AWS CLI). Untuk informasi lebih lanjut tentang menggunakan AWS CLI, lihat Refensi Perintah AWS CLI.

AWS Glue menggunakan AWS Glue Data Catalog untuk menyimpan metadata tentang sumber data, transformasi, dan target. Katalog Data adalah pengganti drop-in untuk Apache Hive Metastore. AWS Glue Jobs system menyediakan infrastruktur terkelola untuk menentukan, penjadwalan, dan menjalankan operasi ETL pada data Anda. Untuk informasi selengkapnya tentang API AWS Glue, lihat AWS Glue API.

Konsol AWS Glue

Anda menggunakan konsol AWS Glue untuk menentukan dan mengatur alur kerja ETL Anda. Konsol memanggil beberapa operasi API di AWS Glue Data Catalog dan AWS Glue Jobs system untuk melakukan tugas berikut:

- Menentukan objek AWS Glue seperti tugas, tabel, crawler, dan koneksi.

- Menentukan jadwal kapan crawler berjalan.
- Menentukan peristiwa atau jadwal untuk pemicu tugas.
- Mencari dan mem-filter daftar objek AWS Glue.
- Mengedit skrip transformasi.

AWS Glue Data Catalog

AWS Glue Data CatalogIni adalah penyimpanan metadata teknis persisten Anda di Cloud. AWS

Setiap AWS akun memiliki satu AWS Glue Data Catalog per AWS Wilayah. Setiap Katalog Data adalah kumpulan tabel yang sangat skalabel yang disusun ke dalam database. Tabel adalah representasi metadata dari kumpulan data terstruktur atau semi-terstruktur yang disimpan dalam sumber seperti Amazon RDS, Apache Hadoop Distributed File System, Amazon Service, dan lain-lain. OpenSearch AWS Glue Data CatalogIni menyediakan repositori seragam di mana sistem yang berbeda dapat menyimpan dan menemukan metadata untuk melacak data dalam silo data. Anda kemudian dapat menggunakan metadata untuk menanyakan dan mengubah data tersebut secara konsisten di berbagai aplikasi.

Anda menggunakan Katalog Data bersama dengan AWS Identity and Access Management kebijakan dan Lake Formation untuk mengontrol akses ke tabel dan database. Dengan melakukan ini, Anda dapat mengizinkan grup yang berbeda di perusahaan Anda untuk mempublikasikan data dengan aman ke organisasi yang lebih luas sambil melindungi informasi sensitif dengan cara yang sangat terperinci.

Katalog Data, bersama dengan CloudTrail dan Lake Formation, juga memberi Anda kemampuan audit dan tata kelola yang komprehensif, dengan pelacakan perubahan skema dan kontrol akses data. Hal ini akan membantu memastikan bahwa data tidak dimodifikasi dengan tidak semestinya atau tidak dibagi dengan tidak disengaja.

Untuk informasi tentang mengamankan dan mengauditAWS Glue Data Catalog, lihat:

- AWS Lake FormationUntuk informasi lebih lanjut, lihat Apa ituAWS Lake Formation? di Panduan AWS Lake Formation Pengembang.
- CloudTrailUntuk informasi lebih lanjut, lihat Apa itu CloudTrail? dalam AWS CloudTrail User Guide.

Berikut ini adalah AWS layanan lain dan proyek sumber terbuka yang menggunakan: AWS Glue Data Catalog

- Amazon Athena — Untuk informasi selengkapnya, lihat Memahami Tabel, Basis Data, dan Katalog Data di Panduan Pengguna Amazon Athena.
- Amazon Redshift Spectrum — Untuk informasi selengkapnya, lihat Menggunakan Amazon Redshift Spectrum untuk Menanyakan Data Eksternal di Panduan Pengembang Basis Data Amazon Redshift.
- EMR Amazon — Untuk informasi selengkapnya, lihat Menggunakan Kebijakan Berbasis Sumber Daya untuk Akses AWS Glue Data Catalog EMR Amazon di Panduan Manajemen EMR Amazon.
- AWS Glue Data Catalogclient untuk Apache Hive metastore — Untuk informasi lebih lanjut tentang GitHub proyek ini, lihat AWS Glue Data CatalogKlien untuk Apache Hive Metastore.

AWS Gluecrawler dan pengklasifikasi

AWS Glue juga memungkinkan Anda mengatur crawler yang dapat memindai data di semua jenis repositori, mengklasifikasikannya, mengekstrak informasi skema darinya, dan menyimpan metadata secara otomatis di AWS Glue Data Catalog. Kemudian, AWS Glue Data Catalog dapat digunakan untuk memandu operasi ETL.

Untuk informasi tentang cara mengatur crawler dan pengklasifikasi, lihat Menggunakan crawler untuk mengisi Katalog Data . Untuk informasi tentang cara memprogram crawler dan pengklasifikasi menggunakan API AWS Glue, lihat API Crawler dan Pengklasifikasi.

AWS GlueOperasi ETL

Menggunakan metadata dalam Katalog Data, AWS Glue dapat secara otomatis menghasilkan Scala atau PySpark (API Python untuk Apache Spark) skrip dengan AWS Glue ekstensi yang dapat Anda gunakan dan modifikasi untuk melakukan berbagai operasi ETL. Sebagai contoh, Anda dapat mengekstrak, membersihkan, dan mengubah data mentah, dan kemudian menyimpan hasilnya dalam repositori yang berbeda, di mana data tersebut dapat di-kueri dan dianalisis. Skrip semacam itu mungkin akan mengubah file CSV menjadi bentuk relasional dan menyimpannya di Amazon Redshift.

Untuk informasi selengkapnya tentang cara menggunakan kemampuan ETL AWS Glue, lihat Pemrograman skrip Spark.

Streaming ETL di AWS Glue

AWS Glue memungkinkan Anda untuk melakukan operasi ETL pada streaming data dengan menggunakan tugas yang terus berjalan. Streaming ETL AWS Glue dibangun di atas mesin Apache Spark Structured Streaming, dan dapat menyerap pengaliran dari Amazon Kinesis Data Streams, Apache Kafka, dan Amazon Managed Streaming for Apache Kafka (Amazon MSK). Streaming ETL dapat membersihkan dan men-transformasi data streaming dan memuatnya ke Amazon S3 atau penyimpanan data JDBC. Menggunakan Streaming ETL di AWS Glue untuk memproses data peristiwa seperti pengaliran IoT, clickstream, dan log jaringan.

Jika Anda tahu skema sumber data streaming, Anda dapat menentukannya dalam tabel Katalog Data. Jika tidak, Anda dapat mengaktifkan deteksi skema dalam tugas ETL streaming. Tugas kemudian akan secara otomatis menentukan skema dari data yang masuk.

Tugas streaming ETL dapat menggunakan transformasi bawaan AWS Glue dan transformasi yang asli untuk Apache Spark Stuctured Streaming. Untuk informasi selengkapnya, lihat Operasi pada streaming DataFrames /Datasets di situs web Apache Spark.

Untuk informasi selengkapnya, lihat the section called “Lowongan kerja Streaming ETL”.

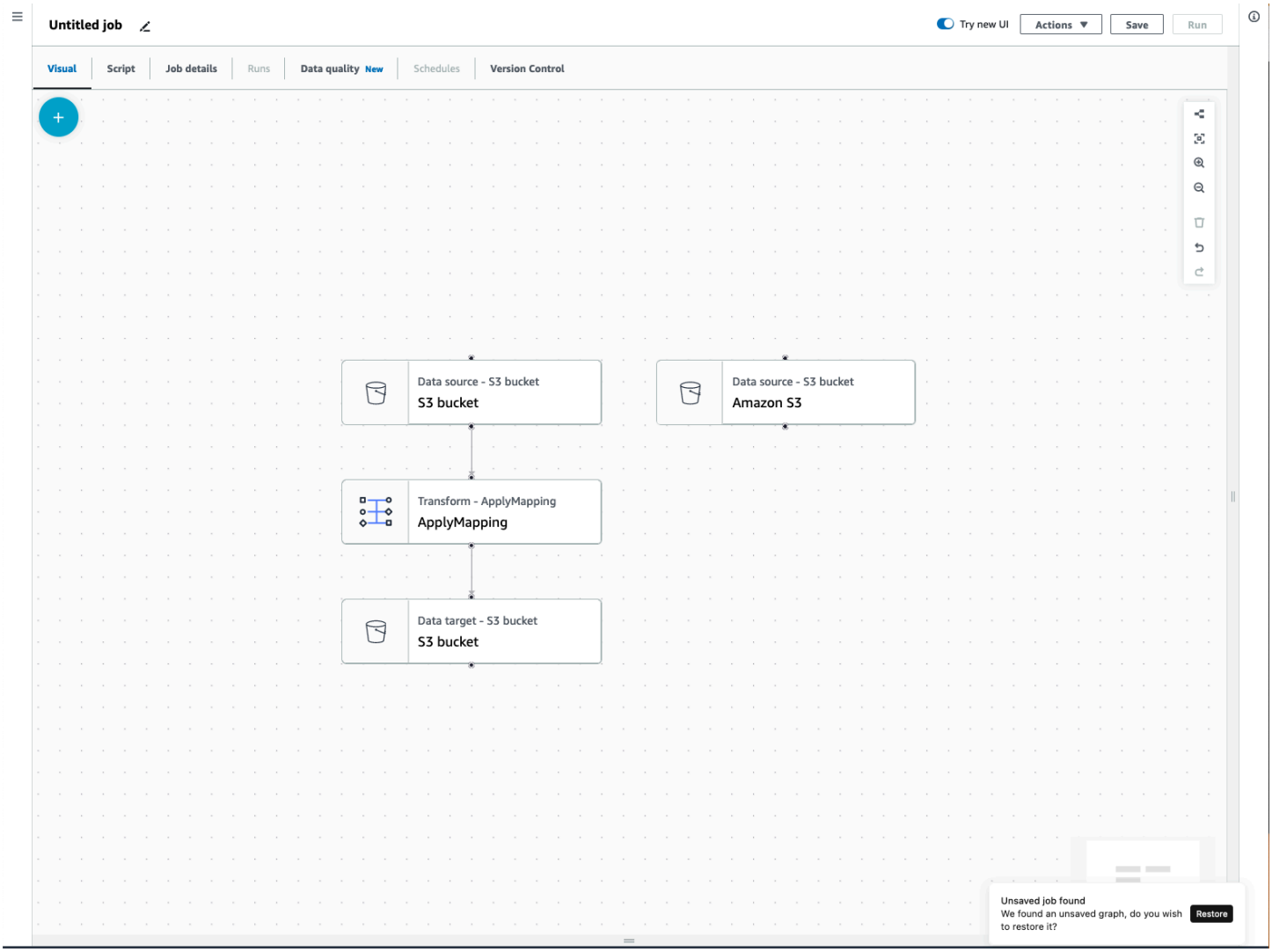
Sistem AWS Glue pekerjaan

AWS Glue Jobs system menyediakan infrastruktur terkelola untuk mengatur alur kerja ETL Anda. Anda dapat membuat tugas di AWS Glue yang mengotomatisasi skrip yang Anda gunakan untuk mengekstrak, mengubah, dan mentransfer data ke lokasi yang berbeda. Tugas dapat dijadwalkan dan dirangkai, atau mereka dapat dipicu oleh peristiwa seperti peristiwa datangnya data baru.

Untuk informasi selengkapnya tentang menggunakan AWS Glue Jobs system, lihat AWS Glue Pemantauan. Untuk informasi selengkapnya tentang pemrograman menggunakan API AWS Glue Jobs system, lihat API Tugas.

Komponen ETL visual

AWS Gluememungkinkan Anda membuat pekerjaan ETL melalui kanvas visual yang dapat Anda manipulasi.



Menu pekerjaan ETL

Opsi menu di bagian atas kanvas memungkinkan Anda mengakses berbagai tampilan dan detail konfigurasi tentang pekerjaan Anda.

- Visual - Kanvas editor pekerjaan Visual. Di sinilah Anda dapat menambahkan node untuk membuat pekerjaan.
- Script — Representasi skrip dari pekerjaan ETL Anda. AWS Gluemenghasilkan skrip berdasarkan representasi visual dari pekerjaan Anda. Anda juga dapat mengedit skrip Anda atau mengunduhnya.

> **Note**
>
> Jika Anda memilih untuk mengedit skrip, pengalaman penulisan pekerjaan secara permanen dikonversi ke mode skrip saja. Setelah itu, Anda tidak dapat menggunakan editor visual untuk mengedit pekerjaan lagi. Anda harus menambahkan semua sumber pekerjaan, transformasi, dan target, dan membuat semua perubahan yang Anda butuhkan dengan editor visual sebelum memilih untuk mengedit skrip.

- Rincian pekerjaan — Tab Job details memungkinkan Anda mengonfigurasi pekerjaan dengan menetapkan properti pekerjaan. Ada properti dasar, seperti nama dan deskripsi pekerjaan Anda, peran IAM, jenis pekerjaan, AWS Glue versi, bahasa, jenis pekerja, jumlah pekerja, bookmark pekerjaan, eksekusi fleksibel, jumlah pensiunan, dan batas waktu pekerjaan, dan ada properti lanjutan, seperti koneksi, perpustakaan, parameter pekerjaan, dan tag.
- Berjalan - Setelah pekerjaan Anda berjalan, tab ini dapat diakses untuk melihat pekerjaan Anda sebelumnya berjalan.
- Kualitas data — Kualitas data mengevaluasi dan memantau kualitas aset data Anda. Anda dapat mempelajari lebih lanjut tentang cara menggunakan kualitas data pada tab ini dan menambahkan transformasi kualitas data ke pekerjaan Anda.
- Jadwal — Pekerjaan yang telah Anda jadwalkan muncul di tab ini. Jika tidak ada jadwal yang dilampirkan pada pekerjaan ini, maka tab ini tidak dapat diakses.
- Kontrol versi — Anda dapat menggunakan Git dengan pekerjaan Anda dengan mengonfigurasi pekerjaan Anda ke repositori Git.

Panel ETL visual

Saat Anda bekerja di kanvas, beberapa panel tersedia untuk membantu Anda mengonfigurasi node, atau membantu Anda melihat pratinjau data dan melihat skema keluaran.

- Properties - Panel Properties muncul ketika Anda memilih node di kanvas Anda.
- Pratinjau data - Panel pratinjau data menyediakan pratinjau output data sehingga Anda dapat membuat keputusan sebelum menjalankan pekerjaan dan memeriksa output Anda.
- Skema keluaran — Tab skema Output memungkinkan Anda untuk melihat dan mengedit skema node transformasi Anda.

Mengubah ukuran panel

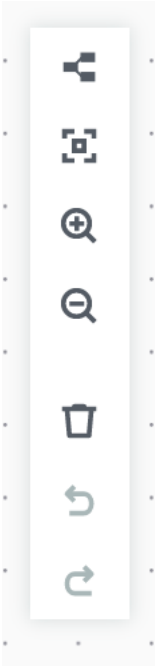
Anda dapat mengubah ukuran panel Properties di sisi kanan layar dan panel bawah yang berisi tab pratinjau Data dan skema Output dengan mengklik tepi panel dan menyeretnya ke kiri dan kanan atau ke atas dan ke bawah.

- Panel properti - Ubah ukuran panel properti dengan mengklik dan menyeret tepi kanvas di sisi kanan layar dan seret ke kiri untuk memperluas lebarnya. Secara default, panel diciutkan dan ketika sebuah node dipilih, panel properti terbuka ke ukuran defaultnya.
- Pratinjau data dan panel skema Output - Ubah ukuran panel bawah dengan mengklik dan menyeret tepi bawah kanvas di bagian bawah layar dan seret ke atas untuk memperluas ketinggiannya. Secara default, panel diciutkan dan ketika sebuah node dipilih, panel bawah terbuka ke ukuran defaultnya.

Kanvas Job

Anda dapat menambahkan, menghapus, dan memindahkan/menyusun ulang node langsung pada kanvas Visual ETL. Anggap saja sebagai ruang kerja Anda untuk membuat pekerjaan ETL yang berfungsi penuh yang dimulai dengan sumber data dan dapat diakhiri dengan target data.

Saat Anda bekerja dengan node di kanvas, Anda memiliki bilah alat yang dapat membantu Anda memperbesar dan memperkecil, menghapus node, membuat atau mengedit koneksi antar node, mengubah orientasi alur pekerjaan, dan membatalkan atau mengulang tindakan.



Bilah alat mengambang ditambatkan ke ukuran kanan atas kanvas dan berisi beberapa gambar yang melakukan tindakan:

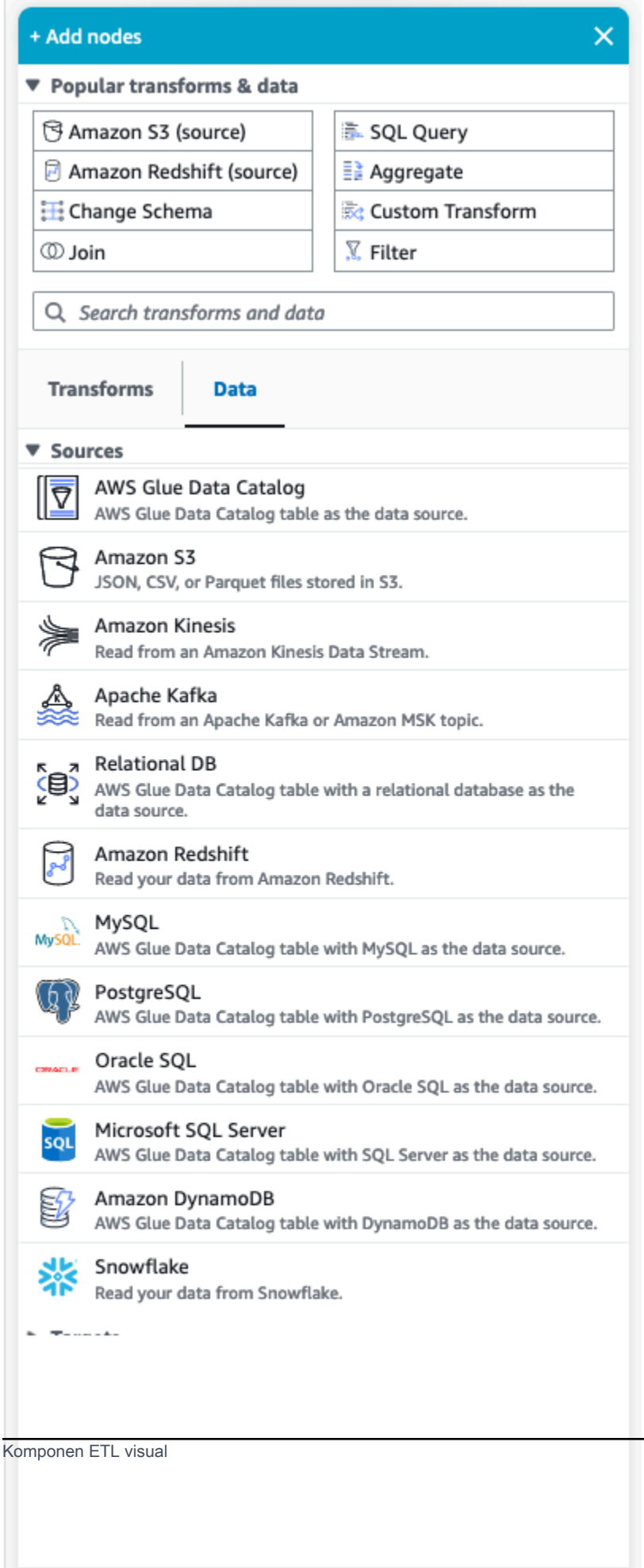
- Ikon tata letak - Ikon pertama di bilah alat adalah ikon tata letak. Secara default, arah pekerjaan visual adalah dari atas ke bawah.Ini mengatur ulang arah pekerjaan visual Anda dengan mengatur node secara horizontal dari kiri ke kanan. Mengklik ikon tata letak lagi mengubah arah kembali ke atas ke bawah.
- Ikon Recenter - Ikon recenter mengubah tampilan kanvas dengan memusatkannya. Anda dapat menggunakan ini dengan pekerjaan besar untuk kembali ke posisi tengah.
- Zoom in icon — Zoom in icon memperbesar ukuran node pada kanvas.
- Ikon zoom out - Ikon zoom out mengurangi ukuran node di kanvas.
- Ikon sampah - Ikon sampah menghapus simpul dari pekerjaan visual. Anda harus memilih node terlebih dahulu.
- Ikon batalkan - Ikon batalkan membalikkan tindakan terakhir yang diambil pada pekerjaan visual.
- Ikon Redo - Ikon redo mengulangi tindakan terakhir yang diambil pada pekerjaan visual.

Menggunakan peta mini

Panel sumber daya

Panel sumber daya berisi semua sumber data, mengubah tindakan, dan koneksi yang tersedia untuk Anda. Buka panel sumber daya di kanvas dengan mengklik ikon “+”. Ini akan membuka panel sumber daya.

Untuk menutup panel sumber daya, klik X di sudut kanan atas panel sumber daya. Ini akan menyembunyikan panel sampai Anda siap untuk membukanya lagi.


+ Add nodes
Popular transforms & data
Amazon S3 (source)
Amazon Redshift (source)
Change Schema
Join
SQL Query
Aggregate
Custom Transform
Filter
Search transforms and data
Transforms
Data
Sources
AWS Glue Data Catalog
AWS Glue Data Catalog table as the data source.
Amazon S3
JSON, CSV, or Parquet files stored in S3.
Amazon Kinesis
Read from an Amazon Kinesis Data Stream.
Apache Kafka
Read from an Apache Kafka or Amazon MSK topic.
Relational DB
AWS Glue Data Catalog table with a relational database as the data source.
Amazon Redshift
Read your data from Amazon Redshift.
MySQL
AWS Glue Data Catalog table with MySQL as the data source.
PostgreSQL
AWS Glue Data Catalog table with PostgreSQL as the data source.
Oracle SQL
AWS Glue Data Catalog table with Oracle SQL as the data source.
Microsoft SQL Server
AWS Glue Data Catalog table with SQL Server as the data source.
Amazon DynamoDB
AWS Glue Data Catalog table with DynamoDB as the data source.
Snowflake
Read your data from Snowflake.

Transformasi & data populer

Di bagian atas panel adalah kumpulan transformasi & data Populer. Node ini biasanya digunakan diAWS Glue. Pilih satu untuk menambahkannya ke kanvas. Anda juga dapat menyembunyikan Transformasi & data Populer dengan mengklik segitiga di sebelah judul Transformasi & data Populer.

Di bawah bagian Transformasi & data Populer, Anda dapat mencari transformasi dan node sumber data. Hasil muncul saat Anda mengetik. Semakin banyak huruf yang Anda tambahkan ke permintaan pencarian Anda, daftar hasil akan semakin kecil. Hasil pencarian diisi dari nama node dan/atau deskripsi. Pilih node untuk menambahkannya ke kanvas Anda.

Transformasi dan Data

Ada dua tab yang mengatur node menjadi Transforms dan Data.

Transformasi — Saat Anda memilih tab Transformasi, semua transformasi yang tersedia dapat dipilih. Pilih transformasi untuk menambahkannya ke kanvas. Anda juga dapat memilih Add Transform di bagian bawah daftar Transforms yang akan membuka halaman baru ke dokumentasi untuk membuat transformasi visual Kustom. Mengikuti langkah-langkah akan memungkinkan Anda untuk membuat transformasi Anda sendiri. Transformasi Anda kemudian akan muncul dalam daftar transformasi yang tersedia.

Data — Tab data berisi semua node untuk Sumber dan Target. Anda dapat menyembunyikan Sumber dan Target dengan mengklik segitiga di sebelah judul Sumber atau Target. Anda dapat menampilkan Sumber dan Target dengan mengklik segitiga lagi. Pilih sumber atau target node untuk menambahkannya ke kanvas. Anda juga dapat memilih Kelola Koneksi untuk menambahkan koneksi baru. Ini akan membuka halaman Konektor di konsol.

AWS Glueuntuk Spark dan AWS Glue untuk Ray

AWS GlueDi Apache Spark (AWS GlueETL), Anda dapat menggunakan untuk PySpark menulis kode Python untuk menangani data dalam skala besar. Spark adalah solusi yang akrab untuk masalah ini, tetapi insinyur data dengan latar belakang yang berfokus pada Python dapat menemukan transisi yang tidak intuitif. DataFrame Model Spark tidak mulus “Pythonic”, yang mencerminkan bahasa Scala dan runtime Java yang dibangun di atasnya.

DiAWS Glue, Anda dapat menggunakan pekerjaan shell Python untuk menjalankan integrasi data Python asli. Pekerjaan ini berjalan pada satu instans Amazon EC2 dan dibatasi oleh kapasitas instance itu. Ini membatasi throughput data yang dapat Anda proses, dan menjadi mahal untuk dipertahankan ketika berhadapan dengan big data.

AWS Glueuntuk Ray memungkinkan Anda untuk meningkatkan beban kerja Python tanpa investasi besar untuk mempelajari Spark. Anda dapat memanfaatkan skenario tertentu di mana Ray berkinerja lebih baik. Dengan menawarkan pilihan, Anda dapat menggunakan kekuatan Spark dan Ray.

AWS GlueETL dan AWS Glue untuk Ray berbeda di bawahnya, sehingga mereka mendukung fitur yang berbeda. Silakan periksa dokumentasi untuk menentukan fitur yang didukung.

Apa AWS Glue untuk Ray?

Ray adalah kerangka kerja komputasi terdistribusi open-source yang dapat Anda gunakan untuk meningkatkan beban kerja, dengan fokus pada Python. Untuk informasi lebih lanjut tentang Ray, lihat situs web Ray. AWS Glue Pekerjaan Ray dan sesi interaktif memungkinkan Anda menggunakan Ray di dalamnyaAWS Glue.

Anda dapat menggunakan Ray AWS Glue untuk menulis skrip Python untuk perhitungan yang akan berjalan secara paralel di beberapa mesin. Dalam pekerjaan Ray dan sesi interaktif, Anda dapat menggunakan pustaka Python yang sudah dikenal, seperti panda, untuk membuat alur kerja Anda mudah ditulis dan dijalankan. Untuk informasi selengkapnya tentang kumpulan data Ray, lihat Kumpulan Data Ray dalam dokumentasi Ray. Untuk informasi lebih lanjut tentang panda, lihat situs web Panda.

Ketika Anda menggunakan AWS Glue untuk Ray, Anda dapat menjalankan alur kerja panda Anda terhadap data besar pada skala perusahaan—dengan hanya beberapa baris kode. Anda dapat membuat pekerjaan Ray dari AWS Glue konsol atau AWS SDK. Anda juga dapat membuka sesi AWS Glue interaktif untuk menjalankan kode Anda di lingkungan Ray tanpa server. Pekerjaan visual AWS Glue Studio di belum didukung.

AWS Glueuntuk pekerjaan Ray memungkinkan Anda untuk menjalankan skrip pada jadwal atau dalam menanggapi acara dari Amazon EventBridge. Pekerjaan menyimpan informasi log dan statistik pemantauan CloudWatch yang memungkinkan Anda memahami kesehatan dan keandalan skrip Anda. Untuk informasi lebih lanjut tentang sistem AWS Glue pekerjaan, lihatthe section called “Bekerja dengan pekerjaan Ray”.

AWS Glueuntuk sesi interaktif Ray (pratinjau) memungkinkan Anda menjalankan cuplikan kode satu demi satu terhadap sumber daya yang disediakan yang sama. Anda dapat menggunakan ini untuk membuat prototipe dan mengembangkan skrip secara efisien, atau membangun aplikasi interaktif Anda sendiri. Anda dapat menggunakan sesi AWS Glue interaktif dari AWS Glue Studio Notebook di. AWS Management Console Untuk informasi selengkapnya, lihat Menggunakan Buku Catatan dengan AWS Glue Studio dan AWS Glue. Anda juga dapat menggunakannya melalui kernel Jupyter,

yang memungkinkan Anda menjalankan sesi interaktif dari alat pengeditan kode yang ada yang mendukung Notebook Jupyter, seperti VSCode. Untuk informasi selengkapnya, lihat the section called “AWS Glue untuk sesi interaktif Ray (pratinjau)”.

Ray mengotomatiskan pekerjaan penskalaan kode Python dengan mendistribusikan pemrosesan di sekelompok mesin yang dikonfigurasi ulang secara real time, berdasarkan beban. Hal ini dapat menyebabkan peningkatan kinerja per dolar untuk beban kerja tertentu. Dengan pekerjaan Ray, kami telah membuat penskalaan otomatis secara native ke dalam model AWS Glue pekerjaan, sehingga Anda dapat sepenuhnya memanfaatkan fitur ini. Pekerjaan Ray berjalan di AWS Graviton, yang mengarah ke kinerja harga keseluruhan yang lebih tinggi.

Selain penghematan biaya, Anda dapat menggunakan penskalaan otomatis asli untuk menjalankan beban kerja Ray tanpa menginvestasikan waktu ke pemeliharaan, penyetelan, dan administrasi klaster. Anda dapat menggunakan pustaka sumber terbuka yang sudah dikenal di luar kotak, seperti panda, dan AWS SDK untuk Pandas. Ini meningkatkan kecepatan iterasi saat Anda mengembangkan AWS Glue untuk Ray. Ketika Anda menggunakan AWS Glue untuk Ray, Anda akan dapat dengan cepat mengembangkan dan menjalankan beban kerja integrasi data yang hemat biaya.

Mengubah skema semi-terstruktur menjadi skema relasional dengan AWS Glue

Sudah umum kalau kita ingin mengkonversi data semi-terstruktur ke dalam tabel relasional. Secara konseptual, Anda meratakan skema hierarkis ke skema relasional. AWS Gluedapat melakukan konversi ini untuk Anda on-the-fly.

Data semi-terstruktur biasanya berisi mark-up untuk mengidentifikasi entitas dalam data. Ia dapat memiliki struktur data bersarang tanpa memiliki skema tetap. Untuk informasi selengkapnya tentang data semi-terstruktur, lihat Data semi-terstruktur di Wikipedia.

Data relasional direpresentasikan oleh tabel yang terdiri dari baris dan kolom. Hubungan antara tabel dapat direpresentasikan oleh hubungan kunci primer (PK) dengan kunci asing (FK). Untuk informasi selengkapnya, lihat Basis data relasional di Wikipedia.

AWS Glue menggunakan crawler untuk menyimpulkan skema untuk data semi-terstruktur. Kemudian mengubah data menjadi skema relasional dengan menggunakan tugas ETL (extract, transform, load). Misalnya, Anda mungkin ingin mengurai data JSON dari file sumber Amazon Simple Storage Service (Amazon S3) ke tabel Amazon Relational Database Service (Amazon RDS). Memahami

bagaimana AWS Glue menangani perbedaan antara skema dapat membantu Anda memahami proses transformasi.

Diagram ini menunjukkan bagaimana AWS Glue mengubah skema semi-terstruktur menjadi skema relasional.

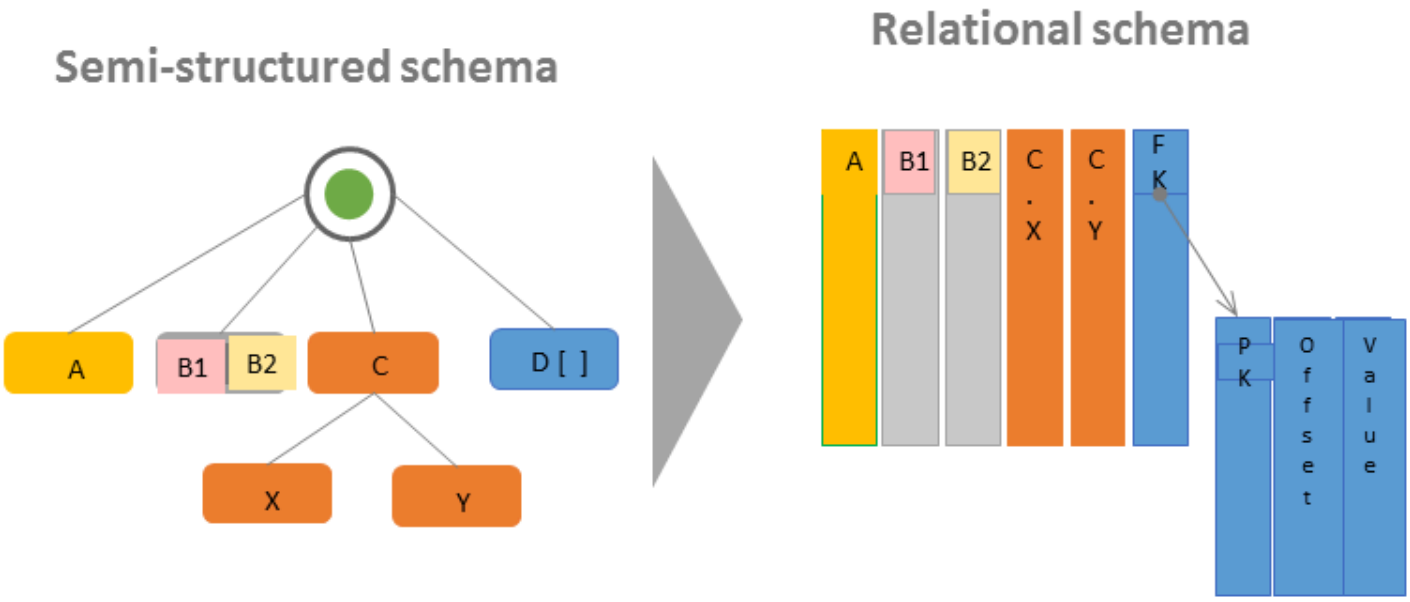


Diagram ini menggambarkan langkah-langkah berikut:

- Nilai tunggal `A` mengkonversi langsung ke kolom relasional.
- Pasangan nilai, `B1` dan `B2`, mengkonversi ke dua kolom relasional.
- Struktur `C`, dengan anak-anak `X` dan `Y`, mengkonversi ke dua kolom relasional.
- Array `D[]` mengkonversi ke kolom relasional dengan kunci asing (FK) yang mengarahkan ke tabel relasional lain. Bersama dengan kunci primer (PK), tabel relasional kedua memiliki kolom yang berisi offset dan nilai item dalam array tersebut.

AWS Sistem tipe Glue

AWS Glue menggunakan beberapa jenis sistem untuk menyediakan antarmuka serbaguna melalui sistem data yang menyimpan data dengan cara yang sangat berbeda. Dokumen ini membedakan sistem tipe AWS Glue dan standar data.

AWS Jenis Katalog Data Glue

Katalog Data adalah registri tabel dan bidang yang disimpan dalam berbagai sistem data, metastore. Ketika komponen AWS Glue, seperti AWS Glue crawler dan AWS Glue with Spark jobs, menulis ke Katalog Data, mereka melakukannya dengan sistem tipe internal untuk melacak jenis bidang. Nilai-nilai ini ditampilkan di kolom Tipe data dari skema tabel di AWS Glue Console. Sistem tipe ini didasarkan pada sistem tipe Apache Hive. Untuk informasi selengkapnya tentang sistem tipe Apache Hive, lihat Jenis di wiki Apache Hive. Untuk informasi lebih lanjut tentang jenis dan dukungan tertentu, contoh disediakan di AWS Glue Console, sebagai bagian dari Schema Builder.

Validasi, kompatibilitas, dan penggunaan lainnya

Katalog Data tidak memvalidasi jenis yang ditulis untuk mengetik bidang. Ketika komponen AWS Glue membaca dan menulis ke Katalog Data, mereka akan kompatibel satu sama lain. AWS Komponen Glue juga bertujuan untuk menjaga kompatibilitas tingkat tinggi dengan jenis Hive. Namun, komponen AWS Glue tidak menjamin kompatibilitas dengan semua jenis Hive. Ini memungkinkan interoperabilitas dengan alat seperti Athena DDL saat bekerja dengan tabel di Katalog Data.

Karena Katalog Data tidak memvalidasi tipe, layanan lain dapat menggunakan Katalog Data untuk melacak jenis menggunakan sistem yang secara ketat sesuai dengan sistem tipe Hive, atau sistem lainnya.

Jenis AWS Glue dengan skrip Spark

Saat skrip AWS Glue with Spark menafsirkan atau mengubah kumpulan data, kami menyediakan`DynamicFrame`, representasi dalam memori dari kumpulan data Anda seperti yang digunakan dalam skrip Anda. Tujuan dari A `DynamicFrame` mirip dengan Spark `DataFrame` — ini memodelkan dataset Anda sehingga Spark dapat menjadwalkan dan mengeksekusi transformasi pada data Anda. Kami menjamin bahwa jenis representasi saling `DynamicFrame` kompatibel dengan `DataFrame` dengan menyediakan `toDF` dan `fromDF` metode.

Jika informasi tipe dapat disimpulkan atau diberikan kepada a`DataFrame`, dapat disimpulkan atau diberikan kepada a`DynamicFrame`, kecuali didokumentasikan lain. Ketika kami menyediakan pembaca atau penulis yang dioptimalkan untuk format data tertentu, jika Spark dapat membaca atau menulis data Anda, pembaca dan penulis kami yang disediakan akan dapat, tunduk pada batasan yang didokumentasikan. Untuk informasi lebih lanjut tentang pembaca dan penulis, lihatthe section called “Opsi format data”.

Jenis Pilihan

`DynamicFrames`menyediakan mekanisme untuk memodelkan bidang dalam kumpulan data yang nilainya mungkin memiliki tipe yang tidak konsisten pada disk di seluruh baris. Misalnya, bidang dapat menyimpan nomor yang disimpan sebagai string di baris tertentu, dan bilangan bulat di baris lain. Mekanisme ini adalah tipe dalam memori yang disebut`Choice`. Kami menyediakan transformasi seperti `ResolveChoice` metode, untuk menyelesaikan kolom Choice menjadi tipe beton. AWS Glue ETL tidak akan menulis jenis Pilihan ke Katalog Data dalam operasi normal; Jenis pilihan hanya ada dalam konteks model DynamicFrame memori kumpulan data. Untuk contoh penggunaan jenis Pilihan, lihatthe section called “Sampel persiapan data”.

AWS Jenis Glue Crawler

Crawler bertujuan untuk menghasilkan skema yang konsisten dan dapat digunakan untuk kumpulan data Anda, lalu menyimpannya di Katalog Data untuk digunakan di komponen AWS Glue lainnya dan Athena. Crawler menangani jenis seperti yang dijelaskan di bagian sebelumnya pada Katalog Data,the section called “AWS Jenis Katalog Data Glue”. Untuk menghasilkan tipe yang dapat digunakan dalam skenario tipe “Pilihan”, di mana kolom berisi nilai dua atau lebih jenis, Crawler akan membuat `struct` tipe yang memodelkan tipe potensial.

Memulai dengan AWS Glue

Bagian berikut memberikan informasi tentang pengaturanAWS Glue. Tidak semua bagian pengaturan diperlukan untuk mulai menggunakanAWS Glue. Anda dapat menggunakan instruksi yang diperlukan untuk mengatur izin IAM, enkripsi, dan DNS (jika Anda menggunakan lingkungan VPC untuk mengakses penyimpanan data atau jika Anda menggunakan sesi interaktif).

Topik

Ikhtisar penggunaan AWS Glue

Dengan AWS Glue, Anda menyimpan metadata di AWS Glue Data Catalog. Anda menggunakan metadata ini untuk mengatur tugas ETL yang mengubah sumber data dan memuat gudang data atau danau data Anda. Langkah-langkah berikut menjelaskan alur kerja umum dan beberapa pilihan yang Anda buat saat bekerja dengan AWS Glue.

> **Note**
>
> Anda dapat menggunakan langkah-langkah berikut, atau Anda dapat membuat alur kerja yang secara otomatis melakukan langkah 1 hingga 3. Untuk informasi selengkapnya, lihat the section called “Melakukan aktivitas ETL yang kompleks menggunakan cetak biru dan alur kerja”.

1. Mengisi AWS Glue Data Catalog dengan definisi tabel.

   Di konsol, untuk penyimpanan data persisten, Anda dapat menambahkan crawler untuk mengisi AWS Glue Data Catalog. Anda dapat memulai penuntun Tambahkan crawler dari daftar tabel

atau daftar crawler. Anda memilih satu atau beberapa penyimpanan data untuk diakses oleh crawler Anda. Anda juga dapat membuat jadwal untuk menentukan seberapa sering Anda menjalankan crawler Anda. Untuk aliran data, Anda dapat secara manual membuat definisi tabel, dan menentukan properti pengaliran.

Opsional, Anda dapat memberikan pengklasifikasi kustom yang menyimpulkan skema data Anda. Anda dapat membuat pengklasifikasi kustom dengan menggunakan pola grok. Namun, AWS Glue menyediakan pengklasifikasi bawaan yang secara otomatis digunakan oleh crawler jika pengklasifikasi kustom tidak mengenali data Anda. Saat menentukan crawler, Anda tidak perlu memilih pengklasifikasi. Untuk informasi selengkapnya tentang cara mengklasifikasikan di AWS Glue, lihat Menambahkan pengklasifikasi ke crawler di AWS Glue.

Melakukan crawling pada beberapa jenis penyimpanan data yang memerlukan koneksi yang menyediakan autentikasi dan informasi lokasi. Jika diperlukan, Anda dapat membuat koneksi yang menyediakan informasi yang diperlukan ini di konsol AWS Glue.

Crawler membaca penyimpanan data Anda dan membuat definisi data dan tabel bernama di AWS Glue Data Catalog. Tabel ini diatur ke dalam basis data pilihan Anda. Anda juga dapat mengisi Katalog Data dengan tabel yang dibuat secara manual. Dengan metode ini, Anda menyediakan skema dan metadata lainnya untuk membuat tabel definisi dalam Katalog Data tersebut. Karena metode ini bisa sedikit membosankan dan rawan kesalahan, maka sebaiknya Anda biarkan crawler membuat definisi tabel.

Untuk informasi selengkapnya tentang cara mengisi AWS Glue Data Catalog dengan definisi tabel, lihat Membuat tabel.

2. Mendefinisikan tugas yang menggambarkan transformasi data dari sumber ke target.

   Umumnya, untuk membuat tugas, Anda harus membuat pilihan berikut:

   - Pilih tabel dari AWS Glue Data Catalog untuk menjadi sumber tugas. Tugas Anda menggunakan definisi tabel ini untuk mengakses sumber data Anda dan menafsirkan format data Anda.
   - Pilih tabel atau lokasi dari AWS Glue Data Catalog untuk menjadi target tugas. Tugas Anda menggunakan informasi ini untuk mengakses penyimpanan data Anda.
   - Katakan AWS Glue untuk menghasilkan skrip untuk mengubah sumber Anda menjadi target. AWS Gluemenghasilkan kode untuk memanggil transformasi bawaan untuk mengonversi data dari skema sumbernya ke format skema target. Transformasi ini melakukan operasi seperti salin data, mengubah nama kolom, dan mem-filter data untuk mengubah data yang diperlukan. Anda dapat memodifikasi skrip ini di konsol AWS Glue.

Untuk informasi selengkapnya tentang cara menentukan tugas di AWS Glue, lihat Membangun pekerjaan ETL visual dengan AWS Glue Studio.

3. Jalankan tugas Anda untuk melakukan transformasi pada data Anda.

   Anda dapat menjalankan tugas Anda sesuai permintaan, atau memulainya berdasarkan salah satu jenis pemicu ini:

   - Pemicu yang berbasis jadwal cron.
   - Pemicu yang berbasis peristiwa; misalnya, berhasil menyelesaikan tugas lain dapat memulai tugas AWS Glue.
   - Pemicu yang memulai tugas sesuai permintaan.

   Untuk informasi lebih lanjut tentang pemicu di AWS Glue, lihat Memulai pekerjaan dan crawler menggunakan pemicu.

4. Pantau crawler terjadwal dan tugas terpicu Anda.

   Gunakan konsol AWS Glue untuk melihat hal berikut ini:

   - Detail dan kesalahan eksekusi tugas.
   - Detail dan kesalahan eksekusi crawler.
   - Notifikasi tentang aktivitas AWS Glue

   Untuk informasi selengkapnya tentang cara memantau crawler dan tugas di AWS Glue, lihat AWS Glue Pemantauan.

Menyiapkan izin IAM untuk AWS Glue

Petunjuk dalam topik ini membantu Anda mengatur izin AWS Identity and Access Management (IAM) dengan cepat untuk. AWS Glue Anda akan menyelesaikan tugas-tugas berikut:

- Berikan identitas IAM Anda akses ke AWS Glue sumber daya.
- Buat peran layanan untuk menjalankan pekerjaan, mengakses data, dan menjalankan tugas Kualitas AWS Glue Data.

Untuk petunjuk terperinci yang dapat Anda gunakan untuk menyesuaikan izin IAM AWS Glue, lihat. Mengkonfigurasi izin IAM untuk AWS Glue

Untuk mengatur izin IAM untuk AWS Glue di AWS Management Console

1. Masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/.
2. Pilih Memulai.
3. Di bawah Siapkan akun Anda AWS Glue, pilih Siapkan izin IAM.
4. Pilih identitas IAM (peran atau pengguna) yang ingin Anda berikan AWS Glue izin. AWS Glue melampirkan kebijakan yang `AWSGlueConsoleFullAccess` dikelola pada identitas ini. Anda dapat melewati langkah ini jika Anda ingin mengatur izin ini secara manual atau hanya ingin menetapkan peran layanan default.
5. Pilih Selanjutnya.
6. Pilih tingkat akses Amazon S3 yang dibutuhkan peran dan pengguna Anda. Opsi yang Anda pilih dalam langkah ini diterapkan ke semua identitas yang Anda pilih.

   a. Di bawah Pilih lokasi S3, pilih lokasi Amazon S3 yang ingin Anda akses.

   b. Selanjutnya, pilih apakah identitas Anda harus memiliki akses Baca saja (disarankan) atau Baca dan tulis ke lokasi yang sebelumnya Anda pilih. AWS Glue menambahkan kebijakan izin ke identitas Anda berdasarkan kombinasi lokasi dan izin baca atau tulis yang Anda pilih.

      Tabel berikut menampilkan izin yang AWS Glue dilampirkan untuk akses Amazon S3.

Jika Anda memilih...	AWS Glue menempel...
Tidak ada perubahan	Tidak ada izin. AWS Glue tidak akan membuat perubahan apa pun pada izin identitas Anda.
Berikan akses ke lokasi Amazon S3 tertentu (hanya baca)	Kebijakan inline yang disematkan dalam identitas IAM yang Anda pilih. Untuk informasi selengkapnya, lihat Kebijakan sebaris di Panduan Pengguna IAM. AWS Glue menamai kebijakan menggunakan konvensi berikut:`AWSGlueConsole` *`<Role/User>`* `InlinePolicy-read-specific-access-` *`<UUID>`*.

Jika Anda memilih...	AWS Glue menempel...
	Misalnya: `AWSGlueConsoleRoleInlinePolicy-read-specific-access-123456780123` . Berikut ini adalah contoh kebijakan sebaris yang AWS Glue dilampirkan untuk memberikan akses hanya-baca ke lokasi Amazon S3 yang ditentukan. `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "s3:Get*", "s3:List*" ], "Resource": [ "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*" ] } ] }`

Jika Anda memilih...	AWS Glue menempel...
Berikan akses ke lokasi Amazon S3 tertentu (baca dan tulis)	Kebijakan inline yang disematkan dalam identitas IAM yang Anda pilih. Untuk informasi selengkapnya, lihat Kebijakan sebaris di Panduan Pengguna IAM. AWS Glue menamai kebijakan menggunakan konvensi berikut:`AWSGlueConsole` *`<Role/User>`* `InlinePolicy-read -and-write-specific-access-` *`<UUID>`*. Misalnya: `AWSGlueConsoleRole InlinePolicy-read-and-write-specific-access-123456780123 .` Berikut ini adalah contoh kebijakan sebaris yang AWS Glue dilampirkan untuk memberikan akses baca dan tulis ke lokasi Amazon S3 yang ditentukan. `{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "s3:Get*", "s3:List*", "s3:*Object*" ], "Resource": [ "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*", "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/*" ] } ]`

Jika Anda memilih...	AWS Glue menempel...
	`}`
Berikan akses penuh ke Amazon S3 (hanya baca)	Kebijakan IAM yang `AmazonS3ReadOnlyAccess` dikelola. Untuk mempelajari selengkapnya, lihat kebijakan AWS terkelola: ReadOnlyAccess AmazonS3.
Berikan akses penuh ke Amazon S3 (baca dan tulis)	Kebijakan IAM yang `AmazonS3FullAccess` dikelola. Untuk mempelajari selengkapnya, lihat kebijakan AWS terkelola: FullAccess AmazonS3.

7. Pilih Selanjutnya.
8. Pilih peran AWS Glue layanan default untuk akun Anda. Peran layanan adalah peran IAM yang AWS Glue digunakan untuk mengakses sumber daya di AWS layanan lain atas nama Anda. Untuk informasi selengkapnya, lihat Peran layanan untuk AWS Glue.

   - Bila Anda memilih peran AWS Glue layanan standar, AWS Glue buat peran IAM baru dalam Akun AWS nama Anda `AWSGlueServiceRole` dengan kebijakan terkelola berikut yang dilampirkan. Jika akun Anda sudah memiliki nama peran IAM`AWSGlueServiceRole`, AWS Glue lampirkan kebijakan ini ke peran yang ada.
     - AWSGlueServiceRole
     - AmazonS3 FullAccess
   - Saat Anda memilih peran IAM yang ada, AWS Glue tetapkan peran sebagai default, tetapi tidak menambahkan izin apa pun padanya. Pastikan Anda telah mengonfigurasi peran yang akan digunakan sebagai peran layanan AWS Glue. Untuk informasi selengkapnya, lihat Langkah 1: Buat kebijakan IAM untuk layanan AWS Glue dan Langkah 2: Buat peran IAM untuk AWS Glue.

9. Pilih Selanjutnya.
10. Terakhir, tinjau izin yang telah Anda pilih lalu pilih Terapkan perubahan. Saat Anda menerapkan perubahan, AWS Glue menambahkan izin IAM ke identitas yang Anda pilih. Anda dapat melihat atau memodifikasi izin baru di konsol IAM di https://console.aws.amazon.com/iam/.

Anda sekarang telah menyelesaikan pengaturan izin IAM minimum untuk. AWS Glue Dalam lingkungan produksi, kami menyarankan Anda membiasakan diri dengan Keamanan di AWS Glue dan Manajemen identitas dan akses untuk AWS Glue membantu Anda mengamankan AWS sumber daya untuk kasus penggunaan Anda.

Langkah selanjutnya

Sekarang setelah Anda memiliki izin IAM yang disiapkan, Anda dapat menjelajahi topik berikut untuk mulai menggunakan: AWS Glue

- Memulai dengan AWS Glue di AWS Skill Builder
- Memulai dengan AWS Glue Data Catalog

Pengaturan untuk AWS Glue Studio

Selesaikan tugas di bagian ini saat Anda menggunakan AWS Glue ETL visual untuk pertama kalinya:

Topik

- Tinjau izin IAM yang diperlukan untuk pengguna AWS Glue Studio
- Tinjau izin IAM yang diperlukan untuk pekerjaan ETL
- Menyiapkan izin IAM untuk AWS Glue Studio
- Konfigurasikan VPC untuk pekerjaan ETL Anda

Tinjau izin IAM yang diperlukan untuk pengguna AWS Glue Studio

Untuk menggunakannyaAWS Glue Studio, pengguna harus memiliki akses ke berbagai AWS sumber daya. Pengguna harus dapat melihat dan memilih bucket Amazon S3, kebijakan IAM dan IAM role, dan objek AWS Glue Data Catalog.

Izin layanan AWS Glue

AWS Glue Studiomenggunakan tindakan dan sumber daya AWS Glue layanan. Pengguna Anda memerlukan izin pada tindakan dan sumber daya ini untuk digunakan AWS Glue Studio secara efektif. Anda dapat memberi AWS Glue Studio pengguna kebijakan `AWSGlueConsoleFullAccess` terkelola, atau membuat kebijakan khusus dengan sekumpulan izin yang lebih kecil.

> **Important**
>
> Sesuai praktik keamanan terbaik, disarankan untuk membatasi akses dengan memperketat kebijakan untuk lebih membatasi akses ke bucket Amazon S3 dan grup log Amazon CloudWatch. Untuk contoh kebijakan Amazon S3, lihat Menulis Kebijakan IAM: Cara Memberikan Akses ke Bucket Amazon S3.

Membuat Kebijakan IAM Kustom untuk AWS Glue Studio

Anda dapat membuat kebijakan khusus dengan sekumpulan izin yang lebih kecil untukAWS Glue Studio. Kebijakan dapat memberikan izin untuk subset objek atau tindakan. Gunakan informasi berikut saat membuat kebijakan khusus.

Untuk menggunakan AWS Glue Studio API, sertakan `glue:UseGlueStudio` dalam kebijakan tindakan dalam izin IAM Anda. Menggunakan `glue:UseGlueStudio` akan memungkinkan Anda untuk mengakses semua AWS Glue Studio tindakan bahkan ketika lebih banyak tindakan ditambahkan ke API dari waktu ke waktu.

Tindakan grafik asiklik terarah (DAG)

- CreateDag
- UpdateDag
- GetDag
- DeleteDag

Aksi Job

- SaveJob
- GetJob
- CreateJob
- DeleteJob
- GetJobs
- UpdateJob

Job run Actions

- StartJobRun
- GetJobRuns
- BatchStopJobRun
- GetJobRun
- QueryJobRuns
- QueryJobs
- QueryJobRunsAggregated

Tindakan Skema

- GetSchema
- GetInferredSchema

Tindakan Database

- GetDatabases

Rencana Tindakan

- GetPlan

Tindakan Tabel

- SearchTables
- GetTables
- GetTable

Tindakan Koneksi

- CreateConnection
- DeleteConnection
- UpdateConnection
- GetConnections
- GetConnection

Tindakan Pemetaan

- GetMapping

Tindakan Proksi S3

- ListBuckets
- ListObjectsV2
- GetBucketLocation

Tindakan Konfigurasi Keamanan

- GetSecurityConfigurations

Tindakan Skrip

- CreateScript (berbeda dari API dengan nama yang sama diAWS Glue)

Mengakses API AWS Glue Studio

Untuk mengaksesAWS Glue Studio, tambahkan `glue:UseGlueStudio` daftar kebijakan tindakan di izin IAM.

Dalam contoh di bawah `glue:UseGlueStudio` ini, disertakan dalam kebijakan tindakan, tetapi AWS Glue Studio API tidak diidentifikasi secara individual. Itu karena ketika Anda menyertakan`glue:UseGlueStudio`, Anda secara otomatis diberikan akses ke API internal tanpa harus menentukan AWS Glue Studio API individual dalam izin IAM.

Dalam contoh, kebijakan tindakan tambahan yang terdaftar (misalnya,`glue:SearchTables`) bukanlah AWS Glue Studio API, jadi kebijakan tersebut harus disertakan dalam izin IAM sesuai kebutuhan. Anda mungkin juga ingin menyertakan tindakan Proxy Amazon S3 untuk menentukan tingkat akses Amazon S3 yang akan diberikan. Contoh kebijakan di bawah ini menyediakan akses untuk membukaAWS Glue Studio, membuat pekerjaan visual, dan menyimpan/menjalankannya jika peran IAM yang dipilih memiliki akses yang memadai.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "glue:UseGlueStudio",
                "iam:ListRoles",
                "iam:ListUsers",
                "iam:ListGroups",
                "iam:ListRolePolicies",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "glue:SearchTables",
                "glue:GetConnections",
                "glue:GetJobs",
                "glue:GetTables",
                "glue:BatchStopJobRun",
                "glue:GetSecurityConfigurations",
                "glue:DeleteJob",
                "glue:GetDatabases",
                "glue:CreateConnection",
                "glue:GetSchema",
                "glue:GetTable",
                "glue:GetMapping",
                "glue:CreateJob",
                "glue:DeleteConnection",
                "glue:CreateScript",
                "glue:UpdateConnection",
                "glue:GetConnection",
                "glue:StartJobRun",
                "glue:GetJobRun",
                "glue:UpdateJob",
                "glue:GetPlan",
                "glue:GetJobRuns",
                "glue:GetTags",
                "glue:GetJob",
                "glue:QueryJobRuns",
                "glue:QueryJobs",
                "glue:QueryJobRunsAggregated"
            ],
            "Resource": "*"
        },
        {
            "Action": [
                "iam:PassRole"
```

```
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:role/AWSGlueServiceRole*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "glue.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

Izin buku catatan dan pratinjau data

Pratinjau data dan notebook memungkinkan Anda untuk melihat sampel data Anda pada setiap tahap pekerjaan Anda (membaca, mengubah, menulis), tanpa harus menjalankan pekerjaan. Anda menentukan peran AWS Identity and Access Management (IAM) AWS Glue Studio untuk digunakan saat mengakses data. Peran IAM dimaksudkan untuk diasumsikan dan tidak memiliki kredensi jangka panjang standar seperti kata sandi atau kunci akses yang terkait dengannya. Sebaliknya, ketika AWS Glue Studio mengambil peran, IAM menyediakannya dengan kredensil keamanan sementara.

Untuk memastikan pratinjau data dan perintah notebook berfungsi dengan benar, gunakan peran yang memiliki nama yang dimulai dengan string`AWSGlueServiceRole`. Jika Anda memilih untuk menggunakan nama yang berbeda untuk peran Anda, Anda harus menambahkan `iam:passrole` izin dan mengonfigurasi kebijakan untuk peran di IAM. Untuk informasi selengkapnya, lihat Buat kebijakan IAM untuk peran yang tidak diberi nama "AWSGlueServiceRole*”.

> **Warning**
>
> Jika peran memberikan `iam:passrole` izin untuk buku catatan, dan Anda menerapkan rantai peran, pengguna dapat secara tidak sengaja mendapatkan akses ke buku catatan tersebut. Saat ini tidak ada audit yang diterapkan yang akan memungkinkan Anda untuk memantau pengguna mana yang telah diberikan akses ke notebook.

Jika Anda ingin menolak identitas IAM kemampuan untuk membuat sesi pratinjau data, lihat contoh the section called “Menolak identitas kemampuan untuk membuat sesi pratinjau data” berikut.

Izin Amazon CloudWatch

Anda dapat memantau AWS Glue Studio pekerjaan Anda menggunakanAmazon CloudWatch, yang mengumpulkan dan memproses data mentah dari AWS Glue menjadi metrik yang dapat dibaca. near-real-time Secara default, data AWS Glue metrik dikirim secara CloudWatch otomatis. Untuk informasi selengkapnya, lihat Apa itu Amazon CloudWatch? di Panduan CloudWatch Pengguna Amazon, dan AWS GlueMetrik di Panduan AWS Glue Pengembang.

Untuk mengakses CloudWatch dasbor, pengguna yang mengakses AWS Glue Studio memerlukan salah satu dari yang berikut:

- Kebijakan `AdministratorAccess`
- Kebijakan `CloudWatchFullAccess`
- Kebijakan kustom yang mencakup satu atau beberapa izin spesifik tersebut:
  - `cloudwatch:GetDashboard` dan `cloudwatch:ListDashboards` untuk melihat dasbor
  - `cloudwatch:PutDashboard` untuk membuat atau memodifikasi dasbor
  - `cloudwatch:DeleteDashboards` untuk menghapus dasbor

Untuk informasi selengkapnya tentang cara merubah izin bagi pengguna IAM yang menggunakan kebijakan, lihat Mengubah Izin untuk Pengguna IAM di Panduan Pengguna IAM.

Tinjau izin IAM yang diperlukan untuk pekerjaan ETL

Saat Anda membuat pekerjaan menggunakanAWS Glue Studio, pekerjaan mengasumsikan izin peran IAM yang Anda tentukan saat Anda membuatnya. IAM role ini harus memiliki izin untuk mengekstrak data dari sumber data Anda, menulis data ke target Anda, dan mengakses sumber daya AWS Glue.

Nama peran yang Anda buat untuk pekerjaan harus dimulai dengan string `AWSGlueServiceRole` agar dapat digunakan dengan benarAWS Glue Studio. Misalnya, Anda dapat memberi nama peran dengan `AWSGlueServiceRole-FlightDataJob`.

Izin sumber data dan target data

AWS Glue StudioPekerjaan harus memiliki akses ke Amazon S3 untuk sumber, target, skrip, dan direktori sementara apa pun yang Anda gunakan dalam pekerjaan Anda. Anda dapat membuat sebuah kebijakan untuk memberikan akses terperinci ke sumber daya Amazon S3 tertentu.

- Sumber data memerlukan izin `s3:ListBucket` dan `s3:GetObject`.
- Target data memerlukan izin `s3:ListBucket`, `s3:PutObject`, dan `s3:DeleteObject`.

Jika Anda memilih Amazon Redshift sebagai sumber data, maka Anda dapat memberikan sebuah peran untuk izin klaster. Tugas yang berjalan pada klaster Amazon Redshift mengeluarkan perintah yang mengakses Amazon S3 untuk penyimpanan sementara menggunakan kredensial sementara. Jika tugas Anda berjalan selama lebih dari satu jam, maka kredensial ini akan kedaluwarsa dan akan menyebabkan tugas gagal. Untuk menghindari masalah ini, Anda dapat menetapkan sebuah peran untuk klaster Amazon Redshift itu sendiri yang memberikan izin yang diperlukan untuk tugas tersebut dengan menggunakan kredensial sementara. Untuk informasi selengkapnya, lihat Memindahkan Data ke dan dari Amazon Redshift di Panduan Developer AWS Glue.

Jika tugas tersebut menggunakan sumber data atau target selain Amazon S3, maka Anda harus melampirkan izin yang diperlukan kepada IAM role yang digunakan oleh tugas tersebut untuk mengakses sumber dan target data ini. Untuk informasi selengkapnya, lihat Menyiapkan Lingkungan Anda untuk Mengakses Penyimpanan Data di Panduan Developer AWS Glue.

Jika Anda menggunakan konektor dan koneksi untuk penyimpanan data Anda, maka Anda memerlukan izin tambahan, seperti yang dijelaskan di the section called “Izin diperlukan untuk menggunakan konektor”.

Izin yang diperlukan untuk menghapus tugas

Di AWS Glue Studio Anda dapat memilih beberapa pekerjaan di konsol untuk dihapus. Untuk melakukan tindakan ini, Anda harus memiliki izin `glue:BatchDeleteJob`. Hal ini berbeda dari konsol AWS Glue, yang memerlukan izin `glue:DeleteJob` untuk menghapus tugas.

Izin AWS Key Management Service

Jika Anda berencana mengakses sumber Amazon S3 dan target yang menggunakan enkripsi sisi server dengan AWS Key Management Service (AWS KMS), lampirkan kebijakan ke AWS Glue Studio peran yang digunakan oleh pekerjaan yang memungkinkan pekerjaan mendekripsi data. Peran tugas membutuhkan izin `kms:ReEncrypt`, `kms:GenerateDataKey`, dan

`kms:DescribeKey`. Selain itu, peran tugas tersebut membutuhkan izin `kms:Decrypt` untuk mengunggah atau mengunduh objek Amazon S3 yang dienkripsi dengan sebuah kunci utama pelanggan (CMK) AWS KMS.

Tidak ada biaya tambahan untuk penggunaan AWS KMS CMKs. Untuk informasi selengkapnya, lihat Konsep AWS Key Management Service - Kunci Utama Pelanggan (CMK) dan Harga AWS Key Management Service di Panduan Developer AWS Key Management Service.

Izin diperlukan untuk menggunakan konektor

Jika Anda menggunakan Konektor Kustom AWS Glue dan koneksi untuk mengakses penyimpanan data, maka peran yang digunakan untuk menjalankan tugas ETL AWS Glue membutuhkan izin tambahan terlampir:

- Kebijakan terkelola AWS `AmazonEC2ContainerRegistryReadOnly` untuk mengakses konektor yang dibeli dari AWS Marketplace.
- Izin `glue:GetJob` dan `glue:GetJobs`.
- Izin AWS Secrets Manager untuk mengakses rahasia yang digunakan dengan koneksi. Lihat Contoh: Izin untuk mengambil nilai rahasia misalnya kebijakan IAM.

Jika eksekusi tugas ETL AWS Glue dalam VPC menjalankan Amazon VPC, maka VPC harus dikonfigurasi seperti yang dijelaskan dalam the section called “Konfigurasikan VPC untuk pekerjaan ETL Anda”.

Menyiapkan izin IAM untuk AWS Glue Studio

Anda dapat membuat peran dan menetapkan kebijakan untuk pengguna dan peran tugas dengan menggunakan pengguna administrator AWS.

Anda dapat menggunakan kebijakan AWSGlueConsoleFullAccessAWSterkelola untuk memberikan izin yang diperlukan untuk menggunakan AWS Glue Studio konsol.

Untuk membuat kebijakan Anda sendiri, ikuti langkah-langkah yang didokumentasikan di Membuat Kebijakan IAM untuk Layanan AWS Glue di Panduan Developer AWS Glue. Sertakan izin IAM yang dijelaskan sebelumnya di. Tinjau izin IAM yang diperlukan untuk pengguna AWS Glue Studio

Topik

- Lampirkan kebijakan ke AWS Glue Studio pengguna
- Buat kebijakan IAM untuk peran yang tidak diberi nama "AWSGlueServiceRole*”

Lampirkan kebijakan ke AWS Glue Studio pengguna

Setiap AWS pengguna yang masuk ke AWS Glue Studio konsol harus memiliki izin untuk mengakses sumber daya tertentu. Anda memberikan izin tersebut dengan menggunakan penetapan kebijakan IAM kepada pengguna.

Untuk melampirkan kebijakan AWSGlueConsoleFullAccessterkelola ke pengguna

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih kotak centang di sebelah AWSGlueConsoleFullAccess. Anda bisa memakai menu Filter dan kotak pencarian untuk mem-filter daftar kebijakan.
4. Pilih Tindakan kebijakan, lalu pilih Lampirkan.
5. Pilih pengguna untuk dilampiri kebijakan ini. Anda bisa menggunakan menu Filter dan kotak pencarian untuk mem-filter daftar entitas utama. Setelah memilih pengguna yang akan dilampiri kebijakan, pilih Lampirkan kebijakan.
6. Ulangi langkah sebelumnya untuk melampirkan kebijakan tambahan kepada pengguna, sesuai kebutuhan.

Buat kebijakan IAM untuk peran yang tidak diberi nama "AWSGlueServiceRole*”

Untuk mengonfigurasi kebijakan IAM untuk peran yang digunakan oleh AWS Glue Studio

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Tambahkan kebijakan IAM baru. Anda dapat menambahkan kebijakan yang sudah ada atau membuat kebijakan inline IAM baru. Untuk membuat kebijakan IAM:
    1. Pilih Kebijakan, lalu pilih Buat Kebijakan. Jika tombol Memulai muncul, pilih tombol tersebut, lalu pilih Buat Kebijakan.
    2. Di sebelah Buat Kebijakan Anda Sendiri, pilih Pilih.
    3. Untuk Nama Kebijakan, ketikkan nilai apa pun yang mudah Anda rujuk nanti. Secara opsional, ketik teks deskriptif dalam Deskripsi.
    4. Untuk Dokumen Kebijakan, ketik pernyataan kebijakan dengan format berikut, lalu pilih Buat Kebijakan:

3. Salin dan tempel blok berikut ke dalam kebijakan di bawah larik “Pernyataan”, ganti *my-interactive-session-role-prefix* dengan awalan untuk semua peran umum yang akan dikaitkan dengan izin untuk. AWS Glue

```
{
    "Action": [
        "iam:PassRole"
    ],
    "Effect": "Allow",
    "Resource": "arn:aws:iam::*:role/my-interactive-session-role-prefix*",
    "Condition": {
        "StringLike": {
            "iam:PassedToService": [
                "glue.amazonaws.com "
            ]
        }
    }
}
```

Berikut adalah contoh lengkap dengan array Versi dan Pernyataan yang disertakan dalam kebijakan

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
        "Action": [
            "iam:PassRole"
        ],
        "Effect": "Allow",
        "Resource": "arn:aws:iam::*:role/my-interactive-session-role-prefix*",
        "Condition": {
            "StringLike": {
                "iam:PassedToService": [
                    "glue.amazonaws.com "
                ]
            }
        }
    }
  ]
}
```

4. Untuk mengaktifkan kebijakan bagi pengguna, pilih Pengguna.
5. Pilih pengguna yang ingin Anda lampirkan kebijakan.

Konfigurasikan VPC untuk pekerjaan ETL Anda

Anda dapat menggunakan Amazon Virtual Private Cloud (Amazon VPC) untuk menentukan jaringan virtual di area Anda sendiri yang terisolasi secara logis di dalamAWS Cloud, yang dikenal sebagai virtual private cloud (VPC). Anda dapat meluncurkan sumber daya AWS, seperti instans, ke dalam VPC Anda. VPC Anda sangat menyerupai jaringan tradisional yang mungkin Anda operasikan di pusat data Anda sendiri, dengan memanfaatkan infrastruktur terukur dari AWS. Anda dapat mengonfigurasi VPC Anda; Anda dapat memilih baris alamat IP, membuat subnet, dan mengonfigurasi tabel rute, gateway jaringan, dan pengaturan keamanan. Anda dapat menghubungkan instans dalam VPC Anda ke internet. Anda dapat menghubungkan VPC Anda ke pusat data perusahaan Anda sendiri, membuat AWS Cloud perpanjangan pusat data Anda. Untuk melindungi sumber daya di setiap subnet, Anda dapat menggunakan beberapa lapisan keamanan, termasuk grup keamanan dan daftar kontrol akses jaringan. Untuk informasi selengkapnya, lihat Panduan Pengguna Amazon VPC.

Anda dapat mengkonfigurasi tugas ETL AWS Glue untuk dijalankan dalam sebuah VPC saat menggunakan konektor. Anda harus mengkonfigurasi VPC Anda seperti berikut, sesuai keperluan:

- Akses jaringan publik untuk menyimpan data tidak di AWS. Semua penyimpanan data yang diakses oleh tugas harus tersedia dari subnet VPC.
- Jika tugas Anda perlu mengakses sumber daya VPC dan internet publik, maka VPC perlu memiliki gateway network address translation (NAT) di dalam VPC tersebut.

  Untuk informasi selengkapnya, lihat Menyiapkan Lingkungan Anda untuk Mengakses Penyimpanan Data di Panduan Developer AWS Glue.

Memulai dengan notebook di AWS Glue Studio

Ketika Anda memulai buku catatanAWS Glue Studio, semua langkah konfigurasi dilakukan untuk Anda sehingga Anda dapat menjelajahi data Anda dan mulai mengembangkan skrip pekerjaan Anda setelah hanya beberapa detik.

Bagian berikut menjelaskan cara membuat peran dan memberikan izin yang sesuai untuk menggunakan buku catatan untuk pekerjaan ETL. AWS Glue Studio

Topik

Memberikan izin untuk peran IAM

Menyiapkan AWS Glue Studio adalah prasyarat untuk menggunakan notebook.

Untuk menggunakan buku catatanAWS Glue, peran Anda memerlukan yang berikut:

- Hubungan kepercayaan dengan AWS Glue untuk `sts:AssumeRole` tindakan dan, jika Anda ingin menandai maka`sts:TagSession`.
- Kebijakan IAM yang berisi semua operasi API untuk notebookAWS Glue, dan sesi interaktif.
- Kebijakan IAM untuk peran lulus karena peran tersebut harus dapat berpindah sendiri dari notebook ke sesi interaktif.

Misalnya, saat membuat peran baru, Anda dapat menambahkan kebijakan AWS terkelola standar seperti `AWSGlueConsoleFullAccessRole` peran tersebut, lalu menambahkan kebijakan baru untuk operasi buku catatan dan PassRole kebijakan IAM lainnya.

Tindakan yang diperlukan untuk hubungan kepercayaan dengan AWS Glue

Saat memulai sesi buku catatan, Anda harus menambahkan `sts:AssumeRole` ke hubungan kepercayaan dari peran yang diteruskan ke buku catatan. Jika sesi Anda menyertakan tag, Anda juga harus lulus `sts:TagSession` tindakan. Tanpa tindakan ini, sesi notebook tidak dapat dimulai.

Misalnya:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "glue.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Kebijakan yang berisi operasi API untuk notebook

Kebijakan contoh berikut menjelaskan izin AWS IAM yang diperlukan untuk buku catatan. Jika Anda membuat peran baru, buat kebijakan yang berisi hal-hal berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:StartNotebook",
                "glue:TerminateNotebook",
                "glue:GlueNotebookRefreshCredentials",
                "glue:DeregisterDataPreview",
                "glue:GetNotebookInstanceStatus",
                "glue:GlueNotebookAuthorize"
            ],
            "Resource": "*"
        }
    ]
}
```

Anda dapat menggunakan kebijakan IAM berikut untuk mengizinkan akses ke sumber daya tertentu:

- AwsGlueSessionUserRestrictedNotebookServicePeran: Menyediakan akses penuh ke semua AWS Glue sumber daya kecuali untuk sesi. Memungkinkan pengguna untuk membuat dan menggunakan hanya sesi notebook yang terkait dengan pengguna. Kebijakan ini juga mencakup izin lain yang diperlukan AWS Glue untuk mengelola AWS Glue sumber daya di AWS layanan lain.
- AwsGlueSessionUserRestrictedNotebookPolicy: Menyediakan izin yang memungkinkan pengguna untuk membuat dan menggunakan hanya sesi notebook yang terkait dengan pengguna. Kebijakan ini juga mencakup izin untuk secara eksplisit mengizinkan pengguna melewati peran sesi terbatasAWS Glue.

Kebijakan IAM untuk lulus peran

Saat Anda membuat buku catatan dengan peran, peran tersebut kemudian diteruskan ke sesi interaktif sehingga peran yang sama dapat digunakan di kedua tempat. Dengan demikian, `iam:PassRole` izin harus menjadi bagian dari kebijakan peran.

Buat kebijakan baru untuk peran Anda menggunakan contoh berikut. Ganti nomor akun dengan nomor Anda sendiri dan nama peran.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::090000000210:role/<role_name>"
        }
    ]
}
```

Menyiapkan profil AWS Glue penggunaan

Salah satu keuntungan utama menggunakan platform cloud adalah fleksibilitasnya. Namun, dengan kemudahan menciptakan sumber daya komputasi ini, muncul risiko peningkatan biaya cloud ketika dibiarkan tidak dikelola dan tanpa pagar pembatas. Akibatnya, admin perlu menyeimbangkan menghindari biaya infrastruktur yang tinggi sementara pada saat yang sama memungkinkan pengguna untuk bekerja tanpa gesekan yang tidak perlu.

Dengan profil AWS Glue penggunaan, admin dapat membuat profil yang berbeda untuk berbagai kelas pengguna dalam akun, seperti pengembang, penguji, dan tim produk. Setiap profil adalah seperangkat parameter unik yang dapat ditetapkan untuk berbagai jenis pengguna. Misalnya, pengembang mungkin membutuhkan lebih banyak pekerja dan dapat memiliki jumlah pekerja maksimum yang lebih tinggi sementara tim produk mungkin membutuhkan lebih sedikit pekerja dan batas waktu tunggu yang lebih rendah atau nilai batas waktu idle.

Contoh perilaku pekerjaan dan pekerjaan berjalan

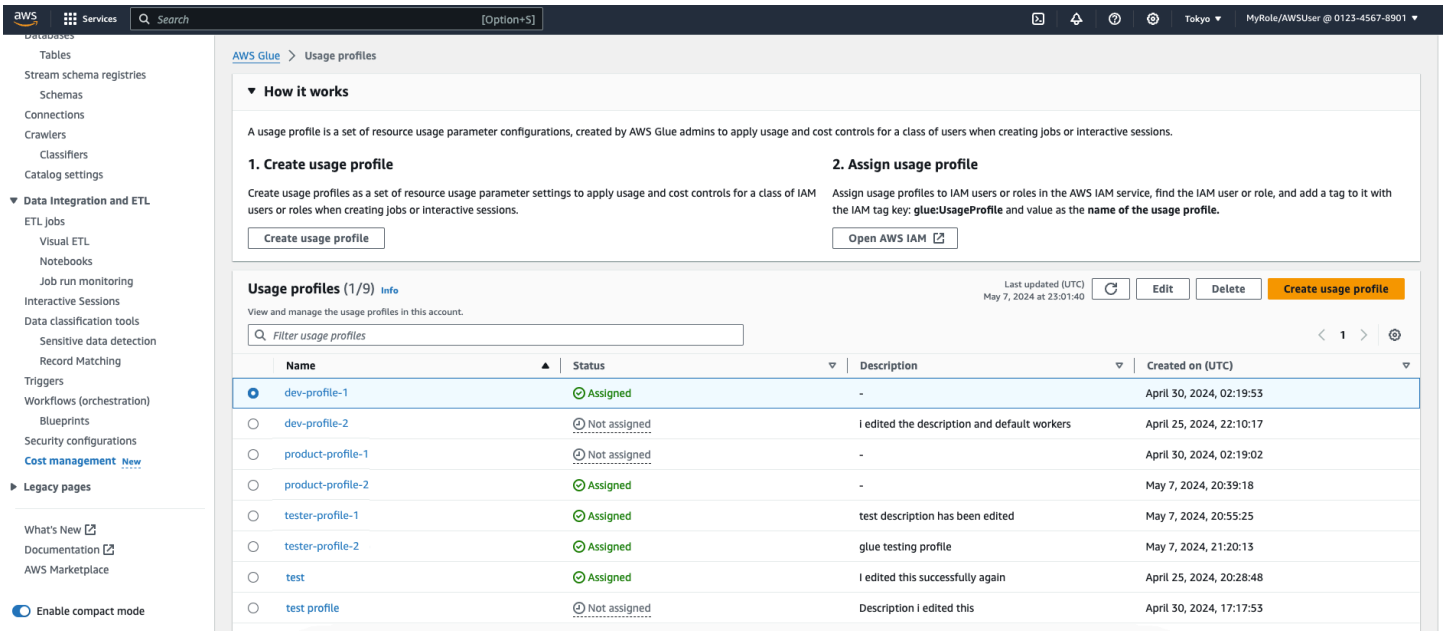
Misalkan pekerjaan dibuat oleh pengguna A dengan profil A. Pekerjaan disimpan dengan nilai parameter tertentu. Pengguna B dengan profil B akan mencoba menjalankan pekerjaan.

Ketika pengguna A menulis pekerjaan, jika dia tidak menetapkan jumlah pekerja tertentu, set default di profil pengguna A diterapkan dan disimpan dengan definisi pekerjaan.

Ketika pengguna B menjalankan pekerjaan, itu berjalan dengan nilai apa pun yang disimpan untuk itu. Jika profil pengguna B sendiri lebih ketat dan tidak diizinkan untuk berjalan dengan banyak pekerja, pekerjaan yang dijalankan akan gagal.

Profil penggunaan sebagai sumber daya

Profil AWS Glue penggunaan adalah sumber daya yang diidentifikasi oleh Amazon Resource Name (ARN). Semua kontrol IAM (Identity and Access Management) default berlaku, termasuk otorisasi berbasis tindakan dan berbasis sumber daya. Admin harus memperbarui kebijakan IAM pengguna yang membuat AWS Glue sumber daya, memberi mereka akses untuk menggunakan profil.



Topik

- Membuat dan mengelola profil penggunaan
- Profil penggunaan dan pekerjaan

Membuat dan mengelola profil penggunaan

Membuat profil AWS Glue penggunaan

Admin harus membuat profil penggunaan dan kemudian menetapkannya ke berbagai pengguna. Saat membuat profil penggunaan, Anda menentukan nilai default serta rentang nilai yang diizinkan

untuk berbagai parameter pekerjaan dan sesi. Anda harus mengkonfigurasi setidaknya satu parameter untuk pekerjaan atau sesi interaktif. Anda dapat menyesuaikan nilai default yang akan digunakan ketika nilai parameter tidak disediakan untuk pekerjaan tersebut, dan/atau mengatur batas rentang atau sekumpulan nilai yang diizinkan untuk validasi jika pengguna memberikan nilai parameter saat menggunakan profil ini.

Default adalah praktik terbaik yang ditetapkan oleh admin untuk membantu penulis pekerjaan. Saat pengguna membuat pekerjaan baru dan tidak menetapkan nilai batas waktu, batas waktu default profil penggunaan akan berlaku. Jika penulis tidak memiliki profil, maka default AWS Glue layanan akan berlaku dan disimpan dalam definisi pekerjaan. Saat runtime, AWS Glue memberlakukan batas yang ditetapkan dalam profil (min, max, pekerja yang diizinkan).

Setelah parameter dikonfigurasi, semua parameter lainnya adalah opsional. Parameter yang dapat disesuaikan untuk pekerjaan atau sesi interaktif adalah:

- Jumlah pekerja — membatasi jumlah pekerja untuk menghindari penggunaan sumber daya komputasi yang berlebihan. Anda dapat menetapkan nilai default, minimum, dan maksimum. Minimal adalah 1.
- Jenis pekerja — batasi jenis pekerja yang relevan untuk beban kerja Anda. Anda dapat menyetel tipe default dan mengizinkan tipe pekerja untuk profil pengguna.
- Timeout — tentukan waktu maksimum pekerjaan atau sesi interaktif dapat dijalankan dan mengkonsumsi sumber daya sebelum dihentikan. Siapkan nilai batas waktu untuk menghindari pekerjaan yang berjalan lama.

  Anda dapat mengatur nilai default, minimum, dan maksimum dalam hitungan menit. Minimal adalah 1 (menit). Meskipun waktu habis AWS Glue default adalah 2880 menit, Anda dapat mengatur nilai default apa pun di profil penggunaan.

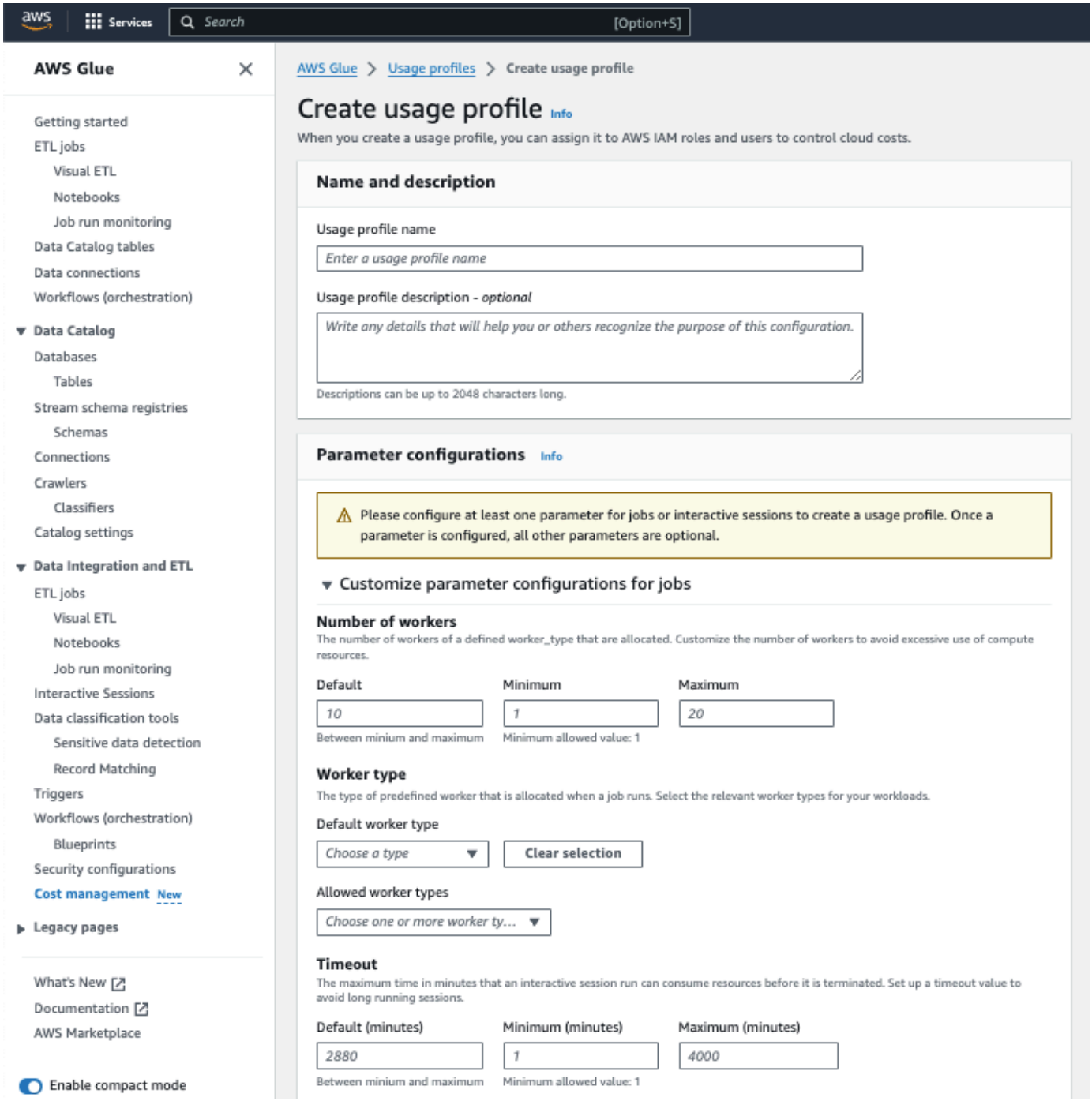
  Ini adalah praktik terbaik untuk menetapkan nilai untuk 'default'. Nilai ini akan digunakan untuk pembuatan pekerjaan atau sesi jika tidak ada nilai yang ditetapkan oleh pengguna.
- Batas waktu idle - tentukan jumlah menit sesi interaktif tidak aktif sebelum waktu habis setelah sel dijalankan. Tentukan batas waktu idle untuk sesi interaktif yang akan dihentikan setelah pekerjaan selesai. Rentang batas waktu idle harus dalam batas waktu tunggu.

  Anda dapat mengatur nilai default, minimum, dan maksimum dalam hitungan menit. Minimal adalah 1 (menit). Meskipun waktu habis AWS Glue default adalah 2880 menit, Anda dapat mengatur nilai default apa pun di profil penggunaan.

Ini adalah praktik terbaik untuk menetapkan nilai untuk 'default'. Nilai ini akan digunakan untuk pembuatan sesi jika tidak ada nilai yang ditetapkan oleh pengguna.

Untuk membuat profil AWS Glue penggunaan sebagai admin (konsol)

1. Di menu navigasi sebelah kiri, pilih Manajemen biaya.
2. Pilih Buat profil penggunaan.
3. Masukkan nama profil Penggunaan untuk profil penggunaan.
4. Masukkan deskripsi opsional yang akan membantu orang lain mengenali tujuan profil penggunaan.
5. Tentukan setidaknya satu parameter di profil. Bidang apa pun dalam formulir adalah parameter. Misalnya, batas waktu siaga sesi minimum.
6. Tentukan tag opsional apa pun yang berlaku untuk profil penggunaan.
7. Pilih Simpan.



Untuk membuat profil penggunaan (AWS CLI)

1. Masukkan perintah berikut.

```
aws glue create-usage-profile --name profile-name --configuration file://
config.json --tags list-of-tags
```

di mana config.json dapat menentukan nilai parameter untuk sesi interaktif (`SessionConfiguration`) dan pekerjaan (): `JobConfiguration`

```
//config.json (There is a separate blob for session/job configuration
{
    "SessionConfiguration": {
        "timeout": {
            "DefaultValue": "2880",
            "MinValue": "100",
            "MaxValue": "4000"
        },
        "idleTimeout": {
            "DefaultValue": "30",
            "MinValue": "10",
            "MaxValue": "4000"
        },
        "workerType": {
            "DefaultValue": "G.2X",
            "AllowedValues": [
                "G.2X",
                "G.4X",
                "G.8X"
            ]
        },
        "numberOfWorkers": {
            "DefaultValue": "10",
            "MinValue": "1",
            "MaxValue": "10"
        }
    },
    "JobConfiguration": {
        "timeout": {
            "DefaultValue": "2880",
            "MinValue": "100",
            "MaxValue": "4000"
        },
        "workerType": {
            "DefaultValue": "G.2X",
            "AllowedValues": [
                "G.2X",
                "G.4X",
                "G.8X"
```

```
            ]
        },
        "numberOfWorkers": {
            "DefaultValue": "10",
            "MinValue": "1",
            "MaxValue": "10"
        }
    }
}
```

2. Masukkan perintah berikut untuk melihat profil penggunaan yang dibuat:

```
aws glue get-usage-profile --name profile-name
```

Tanggapan:

```
{
    "ProfileName": "foo",
    "Configuration": {
        "SessionConfiguration": {
            "numberOfWorkers": {
                "DefaultValue": "10",
                "MinValue": "1",
                "MaxValue": "10"
            },
            "workerType": {
                "DefaultValue": "G.2X",
                "AllowedValues": [
                    "G.2X",
                    "G.4X",
                    "G.8X"
                ]
            },
            "timeout": {
                "DefaultValue": "2880",
                "MinValue": "100",
                "MaxValue": "4000"
            },
            "idleTimeout": {
                "DefaultValue": "30",
                "MinValue": "10",
                "MaxValue": "4000"
            }
```

```
        },
        "JobConfiguration": {
            "numberOfWorkers": {
                "DefaultValue": "10",
                "MinValue": "1",
                "MaxValue": "10"
            },
            "workerType": {
                "DefaultValue": "G.2X",
                "AllowedValues": [
                    "G.2X",
                    "G.4X",
                    "G.8X"
                ]
            },
            "timeout": {
                "DefaultValue": "2880",
                "MinValue": "100",
                "MaxValue": "4000"
            }
        }
    },
    "CreatedOn": "2024-01-19T23:15:24.542000+00:00"
}
```

Perintah CLI tambahan yang digunakan untuk mengelola profil penggunaan:

- lem aws list-usage-profiles
- *aws glue update-usage-profile --name profile-name --file konfigurasi: // config.json*
- *lem aws delete-usage-profile --nama profil-nama*

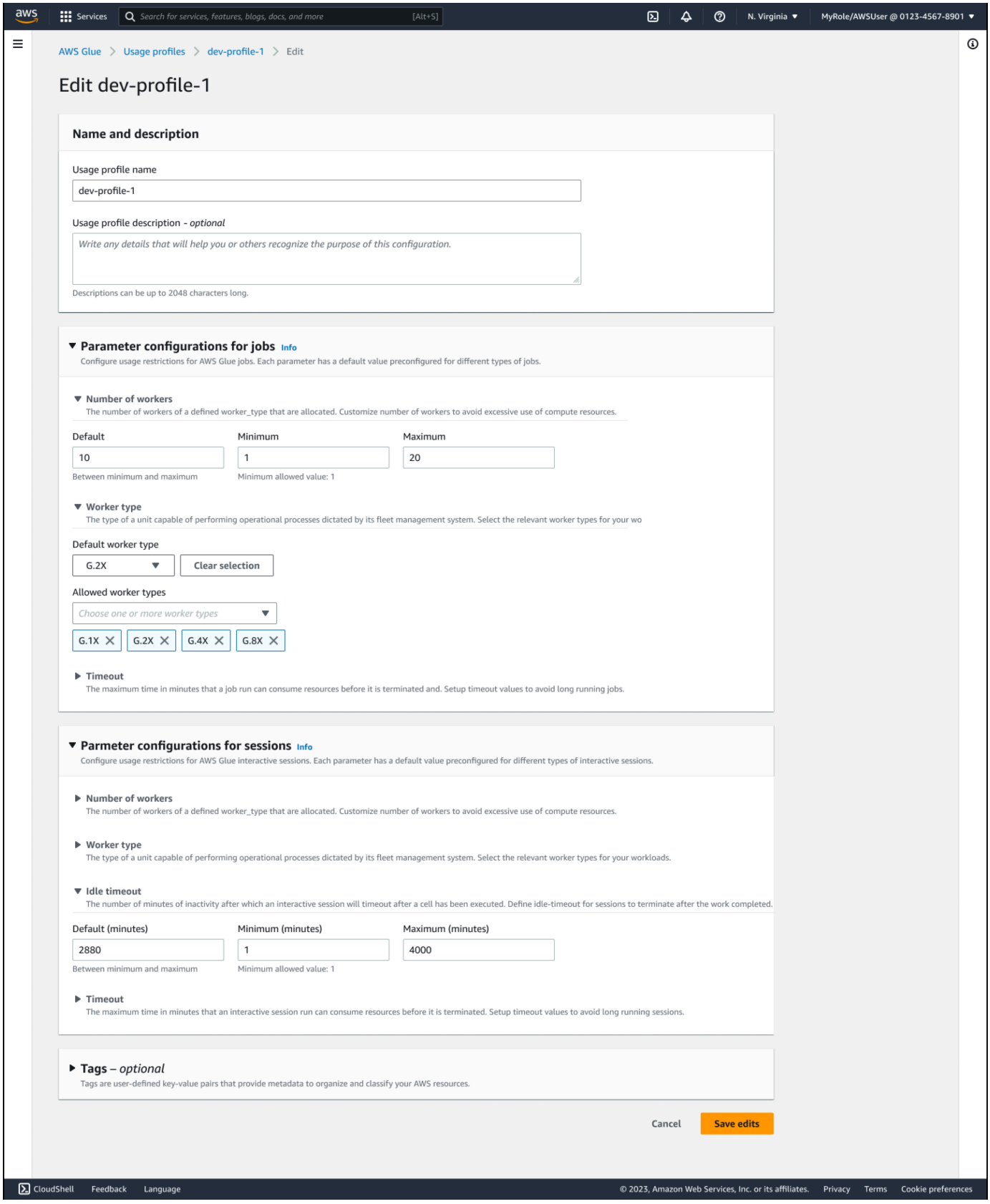
Mengedit profil penggunaan

Admin dapat mengedit profil penggunaan yang telah mereka buat, untuk mengubah nilai parameter profil untuk pekerjaan dan sesi interaktif.

Untuk mengedit profil penggunaan:

Untuk mengedit profil AWS Glue penggunaan sebagai admin (konsol)

1. Di menu navigasi sebelah kiri, pilih Manajemen biaya.
2. Pilih profil penggunaan yang memiliki izin untuk diedit dan pilih Edit.
3. Buat perubahan sesuai kebutuhan pada profil. Secara default, parameter yang sudah memiliki nilai diperluas.
4. Pilih Simpan Pengeditan.


Services
Search for services, features, blogs, docs, and more [Alt+S]
N. Virginia
MyRole/AWSUser @ 0123-4567-8901
AWS Glue > Usage profiles > dev-profile-1 > Edit
Edit dev-profile-1
Name and description
Usage profile name
dev-profile-1
Usage profile description - optional
Write any details that will help you or others recognize the purpose of this configuration.
Descriptions can be up to 2048 characters long.
Parameter configurations for jobs Info
Configure usage restrictions for AWS Glue jobs. Each parameter has a default value preconfigured for different types of jobs.
Number of workers
The number of workers of a defined worker_type that are allocated. Customize number of workers to avoid excessive use of compute resources.
Default
10
Between minimum and maximum
Minimum
1
Minimum allowed value: 1
Maximum
20
Worker type
The type of a unit capable of performing operational processes dictated by its fleet management system. Select the relevant worker types for your wo
Default worker type
G.2X
Clear selection
Allowed worker types
Choose one or more worker types
G.1X
G.2X
G.4X
G.8X
Timeout
The maximum time in minutes that a job run can consume resources before it is terminated and. Setup timeout values to avoid long running jobs.
Parmeter configurations for sessions Info
Configure usage restrictions for AWS Glue interactive sessions. Each parameter has a default value preconfigured for different types of interactive sessions.
Number of workers
The number of workers of a defined worker_type that are allocated. Customize number of workers to avoid excessive use of compute resources.
Worker type
The type of a unit capable of performing operational processes dictated by its fleet management system. Select the relevant worker types for your workloads.
Idle timeout
The number of minutes of inactivity after which an interactive session will timeout after a cell has been executed. Define idle-timeout for sessions to terminate after the work completed.
Default (minutes)
2880
Between minimum and maximum
Minimum (minutes)
1
Minimum allowed value: 1
Maximum (minutes)
4000
Timeout
The maximum time in minutes that an interactive session run can consume resources before it is terminated. Setup timeout values to avoid long running sessions.
Tags – optional
Tags are user-defined key-value pairs that provide metadata to organize and classify your AWS resources.
Cancel
Save edits
CloudShell
Feedback
Language
© 2023, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Untuk mengedit profil penggunaan (AWS CLI)

- Masukkan perintah berikut. Sintaks `--configuration` file yang sama digunakan seperti yang ditunjukkan di atas dalam perintah create.

  ```
  aws glue update-usage-profile --name profile-name --configuration file://
  config.json
  ```

  di mana config.json mendefinisikan nilai parameter untuk sesi interaktif (`SessionConfiguration`) dan pekerjaan (): `JobConfiguration`

Menetapkan profil penggunaan

Kolom Status pemanfaatan di halaman Profil penggunaan menunjukkan apakah profil penggunaan ditetapkan ke pengguna. Mengarahkan kursor ke status menunjukkan entitas IAM yang ditetapkan.

Admin dapat menetapkan profil AWS Glue penggunaan untuk pengguna/peran yang membuat sumber daya. AWS Glue Menetapkan profil adalah kombinasi dari dua tindakan:

- Memperbarui tag pengguna/peran IAM dengan kunci, `glue:UsageProfile` lalu
- Memperbarui kebijakan IAM pengguna/peran.

Bagi pengguna yang menggunakan AWS Glue Studio untuk membuat pekerjaan/sesi interaktif, admin menandai peran berikut:

- Untuk pembatasan pekerjaan, admin menandai peran konsol yang masuk
- Untuk pembatasan sesi interaktif, admin menandai peran yang diberikan pengguna saat mereka membuat buku catatan

Berikut ini adalah contoh kebijakan yang perlu diperbarui admin pada pengguna/peran IAM yang membuat sumber daya: AWS Glue

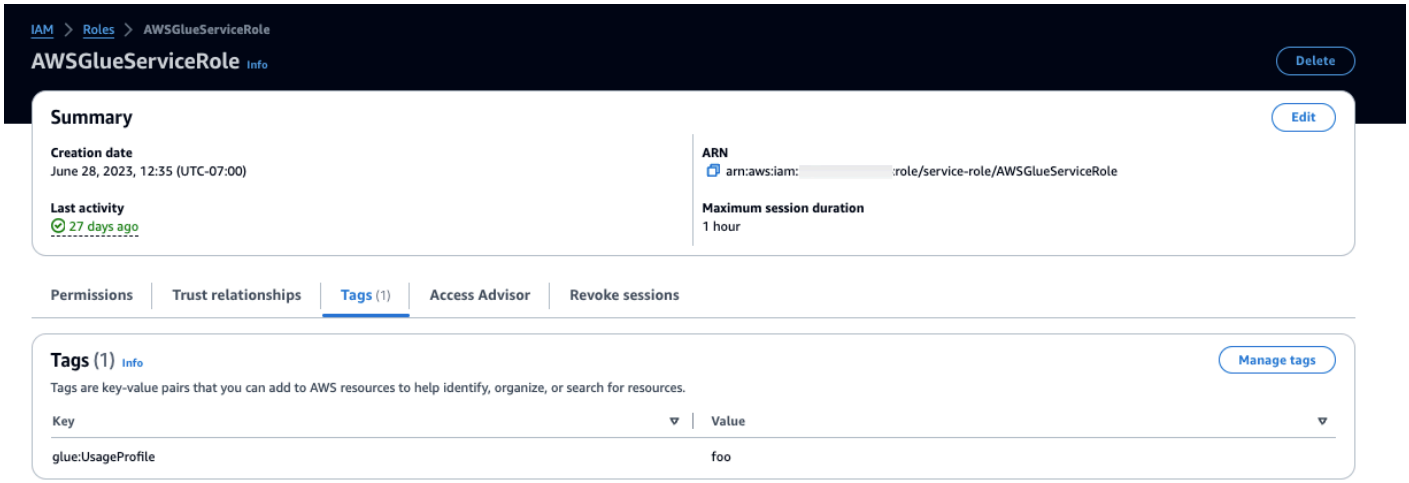
```
{
    "Effect": "Allow",
    "Action": [
        "glue:GetUsageProfile"
    ],
    "Resource": [
```

```
        "arn:aws:glue:us-east-1:123456789012:usageProfile/foo"
    ]
}
```

AWS Glue memvalidasi permintaan job, job run, dan session berdasarkan nilai yang ditentukan dalam profil AWS Glue penggunaan dan memunculkan pengecualian jika permintaan tersebut tidak diizinkan. Untuk API sinkron, kesalahan akan dilemparkan ke pengguna. Untuk jalur asinkron, menjalankan pekerjaan yang gagal dibuat dengan pesan kesalahan bahwa parameter input berada di luar rentang yang diizinkan untuk profil pengguna/peran yang ditetapkan.

Untuk menetapkan profil penggunaan ke pengguna/peran:

1. Buka konsol IAM (Identity and Access Management).
2. Di navigasi kiri, pilih Pengguna atau Peran.
3. Pilih pengguna atau peran.
4. Pilih tab Tanda.
5. Pilih Tambahkan tag baru
6. Tambahkan tag dengan Kunci `glue:UsageProfile` dan Nilai nama profil penggunaan Anda.
7. Pilih Save changes (Simpan perubahan)



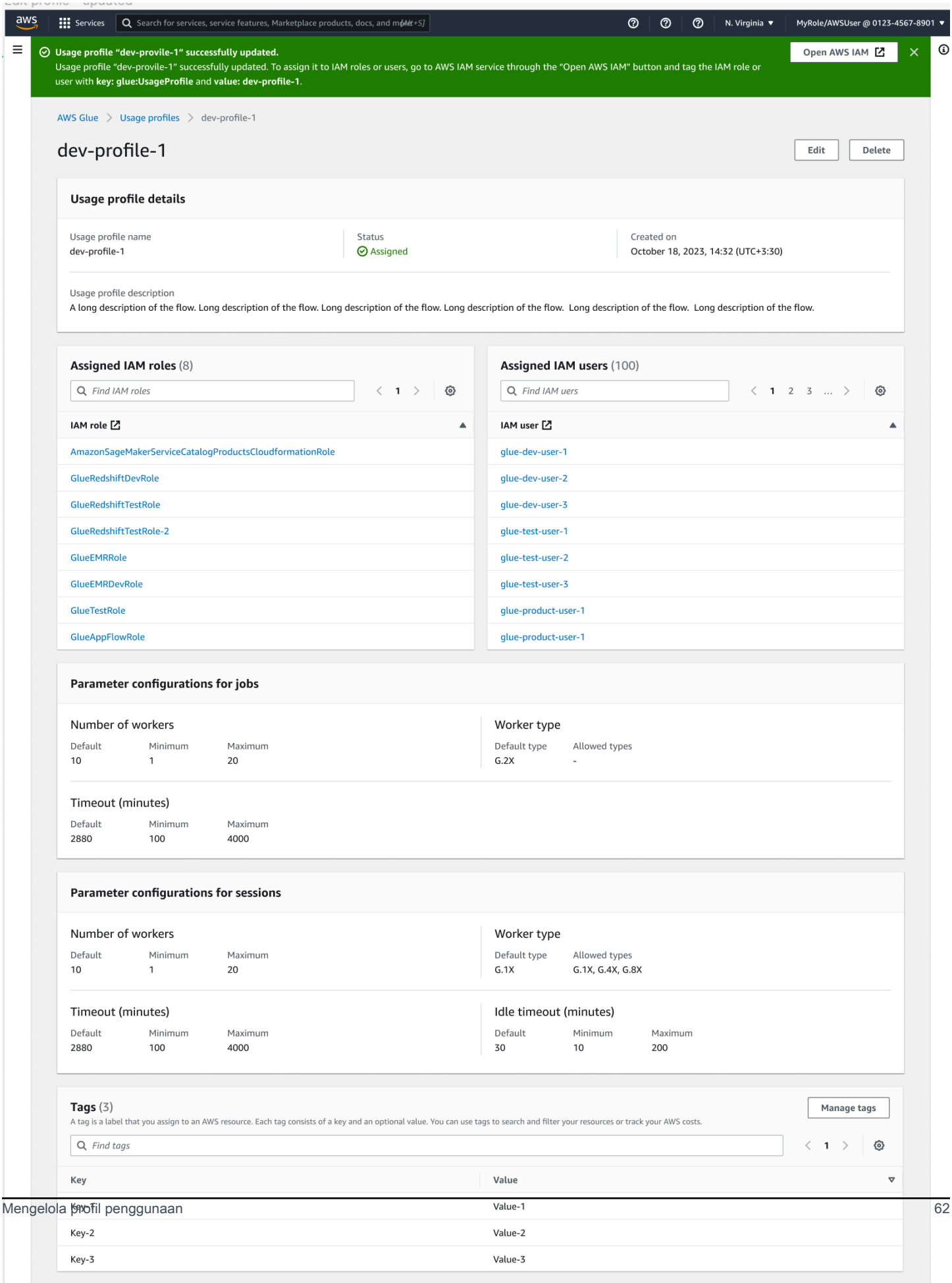
Melihat profil penggunaan yang Anda tetapkan

Pengguna dapat melihat profil penggunaan yang ditetapkan dan menggunakannya saat melakukan panggilan API untuk membuat sumber daya AWS Glue pekerjaan dan sesi, atau memulai pekerjaan.

Izin profil disediakan dalam kebijakan IAM. Selama kebijakan penelepon memiliki `glue:UsageProfile` izin, pengguna dapat melihat profil. Jika tidak, Anda akan mendapatkan kesalahan akses ditolak.

Untuk melihat profil penggunaan yang ditetapkan:

1. Di menu navigasi sebelah kiri, pilih Manajemen biaya.
2. Pilih profil penggunaan yang memiliki izin untuk dilihat.


Services
Search for services, service features, Marketplace products, docs, and more [Alt+S]
N. Virginia
MyRole/AWSUser @ 0123-4567-8901
Usage profile "dev-provile-1" successfully updated.
Usage profile "dev-provile-1" successfully updated. To assign it to IAM roles or users, go to AWS IAM service through the "Open AWS IAM" button and tag the IAM role or user with key: glue:UsageProfile and value: dev-profile-1.
Open AWS IAM
AWS Glue > Usage profiles > dev-profile-1
dev-profile-1
Edit
Delete
Usage profile details
Usage profile name
dev-profile-1
Status
Assigned
Created on
October 18, 2023, 14:32 (UTC+3:30)
Usage profile description
A long description of the flow. Long description of the flow. Long description of the flow. Long description of the flow. Long description of the flow. Long description of the flow.
Assigned IAM roles (8)
Find IAM roles
IAM role
AmazonSageMakerServiceCatalogProductsCloudformationRole
GlueRedshiftDevRole
GlueRedshiftTestRole
GlueRedshiftTestRole-2
GlueEMRRole
GlueEMRDevRole
GlueTestRole
GlueAppFlowRole
Assigned IAM users (100)
Find IAM uers
IAM user
glue-dev-user-1
glue-dev-user-2
glue-dev-user-3
glue-test-user-1
glue-test-user-2
glue-test-user-3
glue-product-user-1
glue-product-user-1
Parameter configurations for jobs
Number of workers
Default 10
Minimum 1
Maximum 20
Worker type
Default type G.2X
Allowed types -
Timeout (minutes)
Default 2880
Minimum 100
Maximum 4000
Parameter configurations for sessions
Number of workers
Default 10
Minimum 1
Maximum 20
Worker type
Default type G.1X
Allowed types G.1X, G.4X, G.8X
Timeout (minutes)
Default 2880
Minimum 100
Maximum 4000
Idle timeout (minutes)
Default 30
Minimum 10
Maximum 200
Tags (3)
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.
Manage tags
Find tags
Key
Value
Key-1
Value-1
Key-2
Value-2
Key-3
Value-3

Profil penggunaan dan pekerjaan

Menulis pekerjaan dengan profil penggunaan

Saat menulis pekerjaan, batasan dan default yang ditetapkan dalam profil penggunaan Anda akan berlaku. Profil Anda akan ditugaskan ke pekerjaan setelah disimpan.

Menjalankan pekerjaan dengan profil penggunaan

Saat Anda memulai pekerjaan, AWS Glue terapkan batasan yang ditetapkan dalam profil penelepon Anda. Jika tidak ada penelepon langsung, Glue kemudian akan menerapkan batasan dari profil yang ditetapkan ke pekerjaan oleh penulisnya.

> **Note**
>
> Ketika pekerjaan dijalankan sesuai jadwal (berdasarkan AWS Glue alur kerja atau AWS Glue pemicu), profil yang ditetapkan untuk pekerjaan yang akan diterapkan penulis.
> Ketika pekerjaan dijalankan oleh layanan eksternal (Step Functions, MWAA) atau `StartJobRun` API, batas profil pemanggil akan diberlakukan.

Untuk AWS Glue alur kerja atau AWS Glue pemicu: pekerjaan yang sudah ada sebelumnya perlu diperbarui untuk menyimpan nama profil baru sehingga batas profil (min, maks, dan pekerja yang diizinkan) akan diberlakukan saat runtime untuk proses terjadwal.

Melihat profil penggunaan yang ditetapkan untuk pekerjaan

Untuk melihat profil yang ditetapkan ke pekerjaan Anda (yang akan digunakan saat runtime dengan AWS Glue alur kerja atau AWS Glue pemicu terjadwal), Anda dapat melihat tab Detail pekerjaan. Anda juga dapat melihat profil yang digunakan dalam proses sebelumnya di tab rincian pekerjaan berjalan.

Memperbarui atau menghapus profil penggunaan yang dilampirkan ke pekerjaan

Profil yang ditetapkan untuk pekerjaan diubah setelah pembaruan. Jika penulis tidak diberi profil penggunaan, profil apa pun yang sebelumnya dilampirkan ke pekerjaan akan dihapus darinya.

Memulai dengan AWS Glue Data Catalog

AWS Glue Data CatalogIni adalah toko metadata teknis Anda yang persisten. Ini adalah layanan terkelola yang dapat Anda gunakan untuk menyimpan, membubuhi keterangan, dan berbagi metadata di Cloud. AWS Untuk informasi selengkapnya, lihat AWS Glue Data Catalog.

AWS GlueKonsol dan beberapa antarmuka pengguna baru-baru ini diperbarui.

Ikhtisar

Anda dapat menggunakan tutorial ini untuk membuat Katalog AWS Glue Data pertama Anda, yang menggunakan bucket Amazon S3 sebagai sumber data Anda.

Dalam tutorial ini, Anda akan melakukan hal berikut menggunakan AWS Glue konsol:

1. Buat database
2. Buat tabel
3. Gunakan bucket Amazon S3 sebagai sumber data

Setelah menyelesaikan langkah-langkah ini, Anda akan berhasil menggunakan bucket Amazon S3 sebagai sumber data untuk mengisi Katalog Data. AWS Glue

Langkah 1: Buat database

Untuk memulai, masuk ke AWS Management Console dan buka AWS Gluekonsol.

Untuk membuat database menggunakan AWS Glue konsol:

1. Di AWS Glue konsol, pilih Database di bawah Katalog data dari menu sebelah kiri.
2. Pilih Add database (Tambahkan basis data).
3. Di halaman Buat database, masukkan nama untuk database. Di bagian Lokasi - opsional, atur lokasi URI untuk digunakan oleh klien Katalog Data. Jika Anda tidak mengetahui hal ini, Anda dapat melanjutkan dengan membuat database.
4. (Opsional). Masukkan deskripsi untuk database.
5. Pilih Buat basis data.

Selamat, Anda baru saja menyiapkan database pertama Anda menggunakan AWS Glue konsol. Database baru Anda akan muncul dalam daftar database yang tersedia. Anda dapat mengedit database dengan memilih nama database dari dasbor Database.

Langkah selanjutnya

Cara lain untuk membuat database:

Anda baru saja membuat database menggunakan AWS Glue konsol, tetapi ada cara lain untuk membuat database:

- Anda dapat menggunakan crawler untuk membuat database dan tabel untuk Anda secara otomatis. Untuk menyiapkan database menggunakan crawler, lihat Bekerja dengan Crawler di Konsol. AWS Glue
- Anda dapat menggunakan AWS CloudFormation template. Lihat Membuat AWS Glue Sumber Daya Menggunakan AWS Glue Data Catalog Template.
- Anda juga dapat membuat database menggunakan operasi API AWS Glue Database.

  Untuk membuat database menggunakan `create` operasi, struktur permintaan dengan memasukkan parameter `DatabaseInput` (wajib).

  Sebagai contoh:

  Berikut ini adalah contoh bagaimana Anda dapat menggunakan CLI, Boto3, atau DDL untuk menentukan tabel berdasarkan file flights_data.csv yang sama dari bucket S3 yang Anda gunakan dalam tutorial.

  CLI

```
aws glue create-database --database-input "{\"Name\":\"clidb\"}"
```

  Boto3

```
glueClient = boto3.client('glue')

response = glueClient.create_database(
    DatabaseInput={
        'Name': 'boto3db'
    }
```

```
)
```

Untuk informasi selengkapnya tentang tipe, struktur, dan operasi data API Database, lihat API Database.

Langkah selanjutnya

Di bagian berikutnya, Anda akan membuat tabel dan menambahkan tabel itu ke database Anda.

Anda juga dapat menjelajahi pengaturan dan izin untuk Katalog Data Anda. Lihat Bekerja dengan Pengaturan Katalog Data di AWS Glue Konsol.

Langkah 2. Buat tabel

Pada langkah ini, Anda membuat tabel menggunakan AWS Glue konsol.

1. Di AWS Glue konsol, pilih Tabel di menu sebelah kiri.
2. Pilih Tambahkan tabel.
3. Atur properti tabel Anda dengan memasukkan nama untuk tabel Anda di detail Tabel.
4. Di bagian Database, pilih database yang Anda buat di Langkah 1 dari menu drop-down.
5. Di bagian Tambahkan penyimpanan data, S3 akan dipilih secara default sebagai jenis sumber.
6. Untuk Data terletak di, pilih Jalur yang ditentukan di akun lain.
7. Salin dan tempel jalur untuk bidang Include path input:

   ```
   s3://crawler-public-us-west-2/flight/2016/csv/
   ```
8. Di bagian Format data, untuk Klasifikasi, pilih CSV. Dan untuk Delimiter, pilih koma (,). Pilih Berikutnya.
9. Anda diminta untuk mendefinisikan skema. Sebuah skema mendefinisikan struktur dan format catatan data. Pilih Tambahkan kolom. (Untuk informasi selengkapnya, lihat Daftar skema).
10. Tentukan properti kolom:

    a. Masukkan nama kolom.

    b. Untuk tipe Kolom, 'string' sudah dipilih secara default.

    c. Untuk nomor Kolom, '1' sudah dipilih secara default.

    d. Pilih Tambahkan.

11. Anda diminta untuk menambahkan indeks partisi. Ini bersifat opsional. Untuk melewati langkah ini, pilih Berikutnya.
12. Ringkasan properti tabel ditampilkan. Jika semuanya terlihat seperti yang diharapkan, pilih Buat. Jika tidak, pilih Kembali dan lakukan pengeditan sesuai kebutuhan.

Selamat, Anda telah berhasil membuat tabel secara manual dan mengaitkannya ke database. Tabel yang baru Anda buat akan muncul di dasbor Tabel. Dari dasbor, Anda dapat memodifikasi dan mengelola semua tabel Anda.

Untuk informasi selengkapnya, lihat Bekerja dengan Tabel di AWS Glue Konsol.

Langkah selanjutnya

Langkah selanjutnya

Sekarang setelah Katalog Data diisi, Anda dapat mulai menulis pekerjaan di. AWS Glue Lihat Membangun pekerjaan ETL visual dengan AWS Glue Studio.

Selain menggunakan konsol, ada cara lain untuk menentukan tabel di Katalog Data termasuk:

- Membuat dan menjalankan crawler
- Menambahkan pengklasifikasi ke crawler di AWS Glue
- Menggunakan API AWS Glue Tabel
- Menggunakan AWS Glue Data Catalog template
- Migrasi metastore Apache Hive
- Menggunakan AWS CLI, Boto3, atau bahasa definisi data (DDL)

  Berikut ini adalah contoh bagaimana Anda dapat menggunakan CLI, Boto3, atau DDL untuk menentukan tabel berdasarkan file flights_data.csv yang sama dari bucket S3 yang Anda gunakan dalam tutorial.

  Lihat dokumentasi tentang cara menyusun AWS CLI perintah. Contoh CLI berisi sintaks JSON untuk nilai 'aws glue create-table --table-input'.

  CLI

```
{
        "Name": "flights_data_cli",
        "StorageDescriptor": {
```

```
            "Columns": [
                {
                    "Name": "year",
                    "Type": "bigint"
                },
                {
                    "Name": "quarter",
                    "Type": "bigint"
                }
            ],
            "Location": "s3://crawler-public-us-west-2/flight/2016/csv",
            "InputFormat": "org.apache.hadoop.mapred.TextInputFormat",
            "OutputFormat":
"org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat",
            "Compressed": false,
            "NumberOfBuckets": -1,
            "SerdeInfo": {
                "SerializationLibrary":
"org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe",
                "Parameters": {
                    "field.delim": ",",
                    "serialization.format": ","
                }
            }
        },
        "PartitionKeys": [
            {
                "Name": "mon",
                "Type": "string"
            }
        ],
        "TableType": "EXTERNAL_TABLE",
        "Parameters": {
            "EXTERNAL": "TRUE",
            "classification": "csv",
            "columnsOrdered": "true",
            "compressionType": "none",
            "delimiter": ",",
            "skip.header.line.count": "1",
            "typeOfData": "file"
        }
    }
```

Boto3

```
import boto3

glue_client = boto3.client("glue")

response = glue_client.create_table(
    DatabaseName='sampledb',
    TableInput={
        'Name': 'flights_data_manual',
    'StorageDescriptor': {
      'Columns': [{
        'Name': 'year',
        'Type': 'bigint'
      }, {
        'Name': 'quarter',
        'Type': 'bigint'
      }],
      'Location': 's3://crawler-public-us-west-2/flight/2016/csv',
      'InputFormat': 'org.apache.hadoop.mapred.TextInputFormat',
      'OutputFormat':
 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat',
      'Compressed': False,
      'NumberOfBuckets': -1,
      'SerdeInfo': {
        'SerializationLibrary':
 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe',
        'Parameters': {
          'field.delim': ',',
          'serialization.format': ','
        }
      },
    },
    'PartitionKeys': [{
      'Name': 'mon',
      'Type': 'string'
    }],
    'TableType': 'EXTERNAL_TABLE',
    'Parameters': {
      'EXTERNAL': 'TRUE',
      'classification': 'csv',
      'columnsOrdered': 'true',
      'compressionType': 'none',
```

```
      'delimiter': ',',
      'skip.header.line.count': '1',
      'typeOfData': 'file'
    }
    }
)
```

DDL

```
CREATE EXTERNAL TABLE `sampledb`.`flights_data` (
  `year` bigint,
  `quarter` bigint)
PARTITIONED BY (
  `mon` string)
ROW FORMAT DELIMITED
  FIELDS TERMINATED BY ','
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  's3://crawler-public-us-west-2/flight/2016/csv/'
TBLPROPERTIES (
  'classification'='csv',
  'columnsOrdered'='true',
  'compressionType'='none',
  'delimiter'=',',
  'skip.header.line.count'='1',
  'typeOfData'='file')
```

Menyiapkan akses jaringan ke penyimpanan data

Untuk menjalankan tugas extract, transform, and load (ETL) Anda, AWS Glue harus dapat mengakses penyimpanan data Anda. Jika tugas tidak perlu dijalankan di subnet virtual private cloud (VPC) Anda—misalnya, mengubah data dari Amazon S3 ke Amazon S3—maka tidak ada konfigurasi tambahan yang diperlukan.

Jika tugas perlu berjalan di VPC subnet—misalnya, mengubah data dari penyimpanan data JDBC di subnet privat—maka AWS Glue akan menyiapkan antarmuka jaringan elastis yang memungkinkan

tugas Anda connect dengan aman ke sumber daya lain dalam VPC Anda. Setiap antarmuka jaringan elastis ditetapkan untuknya alamat IP privat dari rentang alamat IP dalam subnet yang Anda tentukan. Tidak ada alamat IP publik yang ditetapkan. Grup keamanan yang ditentukan dalam koneksi AWS Glue digunakan pada setiap antara muka jaringan elastis. Untuk informasi selengkapnya, lihat Menyiapkan Amazon VPC untuk koneksi JDBC ke penyimpanan data Amazon RDS AWS Glue.

Semua penyimpanan data JDBC yang diakses oleh tugas harus tersedia dari subnet VPC. Untuk mengakses Amazon S3 dari dalam VPC Anda, diperlukan VPC endpoint. Jika tugas Anda perlu mengakses sumber daya VPC dan internet publik, maka VPC perlu memiliki gateway Network Address Translation (NAT) di dalam VPC tersebut.

Sebuah tugas atau pengembangan titik akhir hanya dapat mengakses satu VPC (dan subnet) pada suatu waktu. Jika Anda perlu mengakses penyimpanan data di VPC yang berbeda, Anda memiliki opsi-opsi berikut:

- Gunakan peering VPC untuk mengakses penyimpanan data. Untuk lebih lanjut tentang peering VPC, lihat Dasar-dasar Peering VPC
- Gunakan bucket Amazon S3 sebagai lokasi penyimpanan perantara. Membagi tugas menjadi dua tugas, dengan output Amazon S3 dari tugas 1 sebagai masukan untuk tugas 2.

Untuk detail tentang cara menyambung ke penyimpanan data Amazon Redshift menggunakan Amazon VPC, lihat. the section called “Konfigurasikan Redshift”

Untuk detail tentang cara menghubungkan ke penyimpanan data Amazon RDS menggunakan Amazon VPC, lihat. the section called “Menyiapkan Amazon VPC untuk terhubung ke penyimpanan data Amazon RDS”

Setelah aturan yang diperlukan ditetapkan di Amazon VPC, Anda membuat koneksi AWS Glue dengan properti yang diperlukan untuk terhubung ke penyimpanan data Anda. Untuk informasi selengkapnya tentang koneksi, lihatMenghubungkan ke data.

> **Note**
>
> Pastikan Anda mengatur lingkungan DNS Anda untuk AWS Glue. Untuk informasi selengkapnya, lihat Menyiapkan DNS di VPC.

Topik

Menyiapkan VPC untuk terhubung ke PyPI untuk AWS Glue

Indeks Paket Python (PyPI) adalah repositori perangkat lunak untuk bahasa pemrograman Python. Topik ini membahas detail yang diperlukan untuk mendukung penggunaan paket yang diinstal pip (seperti yang ditentukan oleh pembuat sesi menggunakan `--additional-python-modules` bendera).

Menggunakan sesi AWS Glue interaktif dengan konektor menghasilkan penggunaan jaringan VPC melalui subnet yang ditentukan untuk konektor. Akibatnya AWS layanan dan tujuan jaringan lainnya tidak tersedia kecuali Anda mengatur konfigurasi khusus.

Resolusi untuk masalah ini meliputi:

- Gunakan gateway internet yang dapat dijangkau oleh sesi Anda.
- Siapkan dan gunakan bucket S3 dengan PyPI/repo sederhana yang berisi penutupan transitif dependensi set paket.
- Penggunaan CodeArtifact repositori yang mencerminkan PyPI dan dilampirkan ke VPC Anda.

Menyiapkan gateway internet

Aspek teknis dirinci dalam kasus penggunaan gateway NAT tetapi perhatikan persyaratan ini untuk digunakan`--additional-python-modules`. Secara khusus, `--additional-python-modules` memerlukan akses ke pypi.org yang ditentukan oleh konfigurasi VPC Anda. Perhatikan persyaratan berikut:

1. Persyaratan menginstal modul python tambahan melalui pip install untuk sesi pengguna. Jika sesi menggunakan konektor, konfigurasi Anda mungkin terpengaruh.
2. Ketika konektor sedang digunakan`--additional-python-modules`, ketika sesi dimulai, subnet yang terkait dengan konektor `PhysicalConnectionRequirements` harus menyediakan jalur jaringan untuk mencapai pypi.org.
3. Anda harus menentukan apakah konfigurasi Anda benar atau tidak.

Menyiapkan bucket Amazon S3 untuk meng-host PyPI/repo sederhana yang ditargetkan

Contoh ini menyiapkan mirror PyPI di Amazon S3 untuk satu set paket dan dependensinya.

Untuk mengatur cermin PyPI untuk satu set paket:

```
# pip download all the dependencies
pip download -d s3pypi --only-binary :all: plotly gglplot
pip download -d s3pypi --platform manylinux_2_17_x86_64 --only-binary :all: psycopg2-
binary
# create and upload the pypi/simple index and wheel files to the s3 bucket
s3pypi -b test-domain-name --put-root-index -v s3pypi/*
```

Jika Anda sudah memiliki repositori artefak yang ada, itu akan memiliki URL indeks untuk penggunaan pip yang dapat Anda berikan sebagai pengganti URL contoh untuk bucket Amazon S3 seperti di atas.

Untuk menggunakan url indeks khusus, dengan beberapa contoh paket:

```
%%configure
{
    "--additional-python-modules": "psycopg2_binary==2.9.5",
    "python-modules-installer-option": "--no-cache-dir --verbose --index-url https://
test-domain-name.s3.amazonaws.com/ --trusted-host test-domain-name.s3.amazonaws.com"
}
```

Menyiapkan CodeArtifact cermin pypi yang terpasang pada VPC Anda

Untuk mengatur cermin:

1. Buat repositori di wilayah yang sama dengan subnet yang digunakan oleh konektor.

   Pilih `Public upstream repositories` dan pilih`pypi-store`.
2. Berikan akses ke repositori dari VPC untuk subnet.
3. Tentukan yang benar `--index-url` menggunakan`python-modules-installer-option`.

   ```
   %%configure
   {
       "--additional-python-modules": "psycopg2_binary==2.9.5",
   ```

```
    "python-modules-installer-option": "--no-cache-dir --verbose --index-url https://
test-domain-name.s3.amazonaws.com/ --trusted-host test-domain-name.s3.amazonaws.com"
}
```

Untuk informasi selengkapnya, lihat Menggunakan CodeArtifact dari VPC.

Menyiapkan DNS di VPC

Sistem Nama Domain (DNS) adalah standar dimana nama yang digunakan di internet diubah ke alamat IP yang sesuai. Nama host DNS unik secara menjadi nama sebuah komputer dan terdiri dari nama host dan nama domain. Server DNS mengubah nama host DNS ke alamat IP yang sesuai.

Untuk menyiapkan DNS di VPC Anda, pastikan bahwa nama host DNS dan resolusi DNS keduanya diaktifkan di VPC Anda. Atribut jaringan VPC `enableDnsHostnames` dan `enableDnsSupport` harus diatur ke `true`. Untuk melihat dan mengubah atribut ini, pergi ke konsol VPC di https://console.aws.amazon.com/vpc/.

Untuk informasi selengkapnya, lihat Menggunakan DNS dengan VPC Anda. Juga, Anda dapat menggunakan AWS CLI dan memanggil modify-vpc-attributeperintah untuk mengkonfigurasi atribut jaringan VPC.

> **Note**
>
> Jika Anda menggunakan Route 53, konfirmasi bahwa konfigurasi Anda tidak menimpa atribut jaringan DNS.

Menyiapkan enkripsi di AWS Glue

Alur kerja contoh berikut menyoroti opsi untuk mengkonfigurasi ketika Anda menggunakan enkripsi dengan AWS Glue. Contoh ini menunjukkan penggunaan kunci AWS Key Management Service (AWS KMS) spesifik, namun Anda dapat memilih pengaturan lain berdasarkan kebutuhan khusus Anda. Alur kerja ini hanya menyoroti opsi yang berkaitan dengan enkripsi saat menyiapkan AWS Glue.

1. Jika pengguna konsol AWS Glue tidak menggunakan kebijakan izin yang mengizinkan semua operasi API AWS Glue (misalnya, `"glue:*"`), maka konfirmasikan bahwa tindakan berikut diizinkan:

- "glue:GetDataCatalogEncryptionSettings"
- "glue:PutDataCatalogEncryptionSettings"
- "glue:CreateSecurityConfiguration"
- "glue:GetSecurityConfiguration"
- "glue:GetSecurityConfigurations"
- "glue:DeleteSecurityConfiguration"

2. Setiap klien yang mengakses atau menulis ke katalog terenkripsi—yaitu, pengguna konsol, crawler, tugas, atau titik akhir pengembangan—memerlukan izin berikut.

```
{
 "Version": "2012-10-17",
  "Statement": {
     "Effect": "Allow",
     "Action": [
           "kms:GenerateDataKey",
           "kms:Decrypt",
           "kms:Encrypt"
      ],
     "Resource": "<key-arns-used-for-data-catalog>"
   }
}
```

3. Setiap pengguna atau peran yang mengakses kata sandi koneksi terenkripsi memerlukan izin berikut.

```
{
 "Version": "2012-10-17",
  "Statement": {
     "Effect": "Allow",
     "Action": [
           "kms:Decrypt"
          ],
     "Resource": "<key-arns-used-for-password-encryption>"
          }
}
```

4. Peran dari setiap tugas extract, transform, and load (ETL) yang menulis data terenkripsi ke Amazon S3 membutuhkan izin berikut.

```
{
 "Version": "2012-10-17",
  "Statement": {
     "Effect": "Allow",
     "Action": [
           "kms:Decrypt",
           "kms:Encrypt",
           "kms:GenerateDataKey"
      ],
     "Resource": "<key-arns-used-for-s3>"
   }
}
```

5. Setiap pekerjaan ETL atau crawler yang menulis Log CloudWatch Amazon terenkripsi memerlukan izin berikut dalam kebijakan kunci dan IAM.

   Dalam kebijakan utama (bukan kebijakan IAM):

```
{
  "Effect": "Allow",
  "Principal": {
   "Service": "logs.region.amazonaws.com"
  },
  "Action": [
   "kms:Encrypt*",
   "kms:Decrypt*",
   "kms:ReEncrypt*",
   "kms:GenerateDataKey*",
   "kms:Describe*"
  ],
  "Resource": "<arn of key used for ETL/crawler cloudwatch encryption>"
 }
```

   Untuk informasi selengkapnya tentang kebijakan kunci, lihat Menggunakan Kebijakan Kunci di AWS KMS dalam Panduan Developer AWS Key Management Service.

   Dalam kebijakan IAM lampirkan `logs:AssociateKmsKey` izin:

```
{
  "Effect": "Allow",
  "Principal": {
```

```
    "Service": "logs.region.amazonaws.com"
   },
   "Action": [
    "logs:AssociateKmsKey"
   ],
   "Resource": "<arn of key used for ETL/crawler cloudwatch encryption>"
  }
```

6. Setiap tugas ETL yang menggunakan bookmark tugas terenkripsi memerlukan izin berikut.

```
{
 "Version": "2012-10-17",
  "Statement": {
     "Effect": "Allow",
     "Action": [
           "kms:Decrypt",
           "kms:Encrypt"
      ],
     "Resource": "<key-arns-used-for-job-bookmark-encryption>"
    }
}
```

7. Pada konsol AWS Glue, pilih Pengaturan dalam panel navigasi.

   a. Pada halaman Pengaturan katalog data, enkripsi Katalog Data Anda dengan memilih Enkripsi metadata. Opsi ini mengenkripsi semua objek dalam Katalog Data dengan kunci AWS KMS yang Anda pilih.

   b. Untuk kunci AWS KMS, pilih aws/glue. Anda juga dapat memilih AWS KMS kunci yang Anda buat.

> **Important**
>
> AWS Glue hanya mendukung kunci master utama pelanggan simetris (CMC). Kunci AWS KMS hanya menampilkan kunci simetris saja. Namun, jika Anda memilih Pilih ARN kunci AWS KMS, konsol memungkinkan Anda memasukkan ARN untuk setiap jenis kunci. Pastikan bahwa Anda hanya memasukkan ARN untuk kunci simetris.

Ketika enkripsi diaktifkan, klien yang mengakses Katalog Data harus memiliki izin AWS KMS.

8. Di panel navigasi, pilih Konfigurasi keamanan. Konfigurasi keamanan adalah seperangkat properti keamanan yang dapat digunakan untuk mengkonfigurasi proses AWS Glue. Lalu pilih Tambahkan konfigurasi keamanan. Dalam konfigurasi, pilih salah satu opsi berikut ini:

   a. Pilih Enkripsi S3. Untuk Mode enkripsi, pilih SSE-KMS. Untuk kunci AWS KMS, pilih aws/s3 (memastikan bahwa pengguna memiliki izin untuk menggunakan kunci ini). Hal ini memungkinkan data yang ditulis oleh tugas ke Amazon S3 untuk menggunakan kunci AWS KMS AWS Glue terkelola AWS.

   b. Pilih enkripsi CloudWatch log, dan pilih CMK. (Pastikan bahwa pengguna memiliki izin untuk menggunakan kunci ini). Untuk informasi selengkapnya, lihat Mengenkripsi Data Log di CloudWatch Log Menggunakan AWS KMS dalam Panduan AWS Key Management Service Pengembang.

   > **Important**
   >
   > AWS Glue hanya mendukung kunci master utama pelanggan simetris (CMC). Kunci AWS KMS hanya menampilkan kunci simetris saja. Namun, jika Anda memilih Pilih ARN kunci AWS KMS, konsol memungkinkan Anda memasukkan ARN untuk setiap jenis kunci. Pastikan bahwa Anda hanya memasukkan ARN untuk kunci simetris.

   c. Pilih Properti lanjutan, dan pilih Enkripsi bookmark tugas. Untuk kunci AWS KMS, pilih aws/glue (memastikan bahwa pengguna memiliki izin untuk menggunakan kunci ini). Hal ini memungkinkan enkripsi bookmark tugas ditulis ke Amazon S3 dengan kunci AWS KMS AWS Glue.

9. Di panel navigasi, pilih Koneksi.

   a. Pilih Tambahkan koneksi untuk membuat koneksi ke penyimpanan data Java Database Connectivity (JDBC) yang merupakan target dari tugas ETL Anda.

   b. Untuk menerapkan itu, enkripsi Secure Sockets Layer (SSL) digunakan, pilih Wajibkan koneksi SSL, dan uji koneksi Anda.

10. Di panel navigasi, pilih Tugas.

    a. Pilih Tambahkan tugas untuk membuat tugas yang mengubah data.

    b. Dalam definisi tugas, pilih konfigurasi keamanan yang Anda buat.

11. Pada konsol AWS Glue, jalankan tugas Anda sesuai permintaan. Verifikasi bahwa data Amazon S3 apa pun yang ditulis oleh pekerjaan, CloudWatch Log yang ditulis oleh pekerjaan, dan bookmark pekerjaan semuanya dienkripsi.

Menyiapkan jaringan untuk pengembangan AWS Glue

Untuk menjalankan skrip ekstrak, transformasi, dan muat (ETL)AWS Glue, Anda dapat mengembangkan dan menguji skrip Anda menggunakan titik akhir pengembangan. Titik akhir pengembangan tidak didukung untuk digunakan dengan tugas AWS Glue versi 2.0. Untuk versi 2.0 dan yang lebih baru, metode pengembangan yang disukai adalah menggunakan Jupyter Notebook dengan salah satu kernel. AWS Glue Untuk informasi selengkapnya, lihat the section called “Memulai dengan sesi AWS Glue interaktif”.

Menyiapkan jaringan Anda untuk titik akhir pengembangan

Ketika Anda mengatur titik akhir, Anda menentukan virtual private cloud (VPC), subnet, dan grup keamanan.

> **Note**
>
> Pastikan Anda mengatur lingkungan DNS Anda untuk AWS Glue. Untuk informasi selengkapnya, lihat Menyiapkan DNS di VPC.

Untuk mengaktifkan AWS Glue agar mengakses sumber daya yang diperlukan, tambahkan baris dalam tabel rute subnet Anda untuk meng-associate daftar prefiks untuk Amazon S3 ke VPC endpoint. ID daftar prefiks diperlukan untuk membuat aturan grup keamanan keluar yang memungkinkan lalu lintas dari VPC untuk mengakses layanan AWS melalui sebuah VPC endpoint. Untuk memudahkan menyambung ke server notebook yang dikaitkan dengan titik akhir pengembangan ini, dari mesin lokal Anda, tambahkan baris ke tabel rute untuk menambahkan ID gateway internet. Untuk informasi lebih lanjut, lihat VPC endpoint. Memperbarui tabel rute subnet agar menjadi mirip dengan Daftar Tabel berikut:

Tujuan	Target		
10.0.0.0/16	lokal		
pl-id untuk Amazon S3	vpce-id		
0.0.0.0/0	igw-xxxx		

Untuk mengaktifkan AWS Glue untuk berkomunikasi antara komponen-komponennya, tentukan grup keamanan dengan aturan inbound self-referencing untuk semua port TCP. Dengan membuat aturan self-referencing, Anda dapat membatasi sumber ke grup keamanan yang sama di VPC, dan ia tidak terbuka untuk semua jaringan. Grup keamanan default untuk VPC Anda mungkin sudah memiliki aturan self-referencing inbound untuk semua lalu lintas.

Untuk menyiapkan grup keamanan

1. Masuk ke AWS Management Console dan buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
2. Di panel navigasi sebelah kiri, pilih Grup Keamanan.
3. Pilih grup keamanan yang sudah ada dari daftar, atau Buat Grup Keamanan untuk digunakan dengan titik akhir pengembangan.
4. Di panel grup keamanan, arahkan ke tab Inbound.
5. Tambahkan aturan self-referencing untuk mengizinkan komponen AWS Glue untuk berkomunikasi. Secara khusus, tambahkan atau konfirmasi bahwa ada aturan Jenis `All TCP`, Protokol adalah `TCP`, Rentang Port mencakup semua port, dan yang Sumber adalah nama grup keamanan yang sama seperti ID Grup.

   Aturan inbound terlihat serupa dengan ini:

Tipe	Protokol	Rentang Port	Sumber
Semua TCP	TCP	0–65535	*kelompok keamanan*

   Berikut ini adalah contoh aturan self-referencing inbound:

6. Menambahkan aturan untuk lalu lintas outbound juga. Buka lalu lintas keluar ke semua port, atau membuat aturan self-referencing di mana Jenis `All TCP`, Protokol adalah `TCP`, Rentang Port mencakup semua port, dan yang Sumber adalah nama grup keamanan yang sama seperti ID Grup.

   Aturan outbound terlihat mirip dengan salah satu aturan ini:

Tipe	Protokol	Rentang Port	Tujuan
Semua TCP	TCP	0–65535	*kelompok keamanan*
Semua Lalu Lintas	SEMUA	SEMUA	0.0.0.0/0

Menyiapkan Amazon EC2 untuk server notebook

Dengan titik akhir pengembangan, Anda dapat membuat server notebook untuk menguji skrip ETL Anda dengan notebook Jupyter. Untuk mengaktifkan komunikasi ke notebook Anda, tentukan grup keamanan dengan aturan masuk untuk HTTPS (port 443) dan SSH (port 22). Pastikan bahwa sumber aturan adalah 0.0.0.0/0 atau alamat IP dari mesin yang terhubung ke notebook.

Untuk menyiapkan grup keamanan

1. Masuk ke AWS Management Console dan buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
2. Di panel navigasi sebelah kiri, pilih Grup Keamanan.
3. Pilih grup keamanan yang sudah ada dari daftar, atau Buat Grup Keamanan untuk digunakan dengan server notebook. Grup keamanan yang dikaitkan dengan titik akhir pengembangan Anda juga digunakan untuk menciptakan server notebook Anda.
4. Di panel grup keamanan, arahkan ke tab Inbound.
5. Tambahkan aturan inbound yang mirip dengan ini:

Tipe	Protokol	Rentang Port	Sumber
SSH	TCP	22	0.0.0.0/0
HTTPS	TCP	443	0.0.0.0/0

Berikut ini adalah contoh aturan inbound untuk grup keamanan:

Security Group: sg-19e1b768

Description | **Inbound** | Outbound | Tags

Edit

Type	Protocol	Port Range	Source
SSH	TCP	22	0.0.0.0/0
HTTPS	TCP	443	0.0.0.0/0

Penemuan dan katalogisasi data di AWS Glue

AWS Glue Data Catalog Ini adalah repositori terpusat yang menyimpan metadata tentang kumpulan data organisasi Anda. Ini bertindak sebagai indeks lokasi, skema, dan metrik runtime dari sumber data Anda. Metadata disimpan dalam tabel metadata, di mana setiap tabel mewakili penyimpanan data tunggal.

Anda dapat mengisi Katalog Data menggunakan crawler, yang secara otomatis memindai sumber data Anda dan mengekstrak metadata. Crawler dapat terhubung ke sumber data yang internal (AWS berbasis) dan eksternal ke AWS.

Untuk informasi selengkapnya tentang sumber data yang didukung, lihat Penyimpanan data mana yang dapat saya jelajahi?

Anda juga dapat membuat tabel dalam Katalog Data secara manual dengan mendefinisikan struktur tabel, skema, dan struktur partisi sesuai dengan kebutuhan spesifik Anda.

Untuk informasi selengkapnya tentang membuat tabel metadata secara manual, lihat. Mendefinisikan metadata secara manual

Anda dapat menggunakan informasi dalam Katalog Data untuk membuat dan memantau pekerjaan ETL Anda. Katalog Data terintegrasi dengan layanan AWS analitik lainnya, memberikan tampilan terpadu sumber data sehingga lebih mudah untuk mengelola dan menganalisis data.

- Amazon Athena — Simpan dan kueri metadata tabel di Katalog Data untuk data Amazon S3 menggunakan SQL.
- AWS Lake Formation — Mendefinisikan dan mengelola kebijakan akses data berbutir halus dan akses data audit secara terpusat.
- Amazon EMR — Akses sumber data yang ditentukan dalam Katalog Data untuk pemrosesan data besar.
- Amazon SageMaker — Membangun, melatih, dan menerapkan model pembelajaran mesin dengan cepat dan percaya diri.

Fitur utama dari Katalog Data

Berikut ini adalah aspek-aspek kunci dari Katalog Data.

Repositori metadata

Katalog Data bertindak sebagai repositori metadata pusat, menyimpan informasi tentang lokasi, skema, dan properti sumber data Anda. Metadata ini disusun ke dalam database dan tabel, mirip dengan katalog database relasional tradisional.

Dapat ditemukan data otomatis

Perayap AWS Glue s dapat secara otomatis menemukan dan membuat katalog sumber data baru atau yang diperbarui, mengurangi overhead manajemen metadata manual dan memastikan bahwa Katalog Data Anda tetap ada. up-to-date Dengan membuat katalog sumber data Anda, Katalog Data memudahkan pengguna dan aplikasi untuk menemukan dan memahami aset data yang tersedia dalam organisasi Anda, mempromosikan penggunaan kembali dan kolaborasi data.

Katalog Data mendukung berbagai sumber data, termasuk Amazon S3, Amazon RDS, Amazon Redshift, Apache Hive, dan banyak lagi. Secara otomatis dapat menyimpulkan dan menyimpan metadata dari sumber-sumber ini menggunakan s. Perayap AWS Glue

Untuk informasi lebih lanjut lihat,Menggunakan crawler untuk mengisi Katalog Data .

Manajemen skema

Katalog Data secara otomatis menangkap dan mengelola skema sumber data Anda, termasuk inferensi skema, evolusi, dan pembuatan versi. Anda dapat memperbarui skema dan partisi Anda di Katalog Data menggunakan pekerjaan AWS Glue ETL.

Optimalisasi tabel

Untuk kinerja pembacaan yang lebih baik oleh layanan AWS analitik seperti Amazon Athena dan Amazon EMR, dan pekerjaan AWS Glue ETL, Katalog Data menyediakan pemadatan terkelola (proses yang memadatkan objek Amazon S3 kecil menjadi objek yang lebih besar) untuk tabel Gunung Es di Katalog Data. Anda dapat menggunakan AWS Glue konsol, AWS Lake Formation konsol AWS CLI, atau AWS API untuk mengaktifkan atau menonaktifkan pemadatan untuk tabel Iceberg individual yang ada di Katalog Data.

Untuk informasi selengkapnya, lihat Mengoptimalkan tabel Iceberg.

Statistik kolom

Anda dapat menghitung statistik tingkat kolom untuk tabel Katalog Data dalam format data seperti Parket, ORC, JSON, ION, CSV, dan XMLtanpa menyiapkan pipeline data tambahan. Statistik kolom membantu Anda memahami profil data dengan mendapatkan wawasan tentang nilai dalam

kolom. Katalog Data mendukung pembuatan statistik untuk nilai kolom seperti nilai minimum, nilai maksimum, nilai nol total, nilai total yang berbeda, panjang rata-rata nilai, dan total kemunculan nilai sebenarnya.

Untuk informasi selengkapnya, lihat Mengoptimalkan kinerja kueri menggunakan statistik kolom.

Silsilah data

Katalog Data menyimpan catatan transformasi dan operasi yang dilakukan pada data Anda, memberikan informasi garis keturunan data. Informasi silsilah ini berharga untuk audit, kepatuhan, dan pemahaman asal data.

Integrasi dengan AWS layanan lain

Katalog Data terintegrasi dengan mulus dengan AWS layanan lain, seperti, Amazon Athena AWS Lake Formation, Amazon Redshift Spectrum, dan Amazon EMR. Integrasi ini memungkinkan Anda untuk menanyakan dan menganalisis data di berbagai penyimpanan data menggunakan satu lapisan metadata yang konsisten.

Keamanan dan kontrol akses

AWS Glue terintegrasi dengan AWS Lake Formation untuk mendukung kontrol akses berbutir halus untuk sumber daya Katalog Data, memungkinkan Anda mengelola izin dan mengamankan akses ke aset data berdasarkan kebijakan dan persyaratan organisasi Anda. AWS Glue terintegrasi dengan AWS Key Management Service (AWS KMS) untuk mengenkripsi metadata yang disimpan dalam Katalog Data.

Topik

Mengisi Katalog AWS Glue Data

Anda dapat mengisi AWS Glue Data Catalog menggunakan metode berikut:

- Perayap AWS Glue — An Perayap AWS Glue dapat secara otomatis menemukan dan membuat katalog sumber data seperti database, data lake, dan streaming data. Crawler adalah metode yang paling umum dan direkomendasikan untuk mengisi Katalog Data karena mereka dapat secara otomatis menemukan dan menyimpulkan metadata untuk berbagai sumber data.
- Menambahkan metadata secara manual — Anda dapat menentukan database, tabel, dan detail koneksi secara manual dan menambahkannya ke Katalog Data menggunakan konsol, AWS Glue konsol Lake Formation AWS CLI, atau API. AWS Glue Entri manual berguna saat Anda ingin membuat katalog sumber data yang tidak dapat dirayapi.
- Mengintegrasikan dengan AWS layanan lain — Anda dapat mengisi Katalog Data dengan metadata dari layanan seperti dan Amazon AWS Lake Formation Athena. Layanan ini dapat menemukan dan mendaftarkan sumber data di Katalog Data.
- Mengisi dari repositori metadata yang ada — Jika Anda memiliki penyimpanan metadata yang ada seperti Apache Hive Metastore, Anda dapat menggunakannya untuk mengimpor metadata tersebut ke dalam Katalog Data. AWS Glue Untuk informasi selengkapnya, lihat Migrasi antara Metastore Hive dan on. AWS Glue Data Catalog GitHub

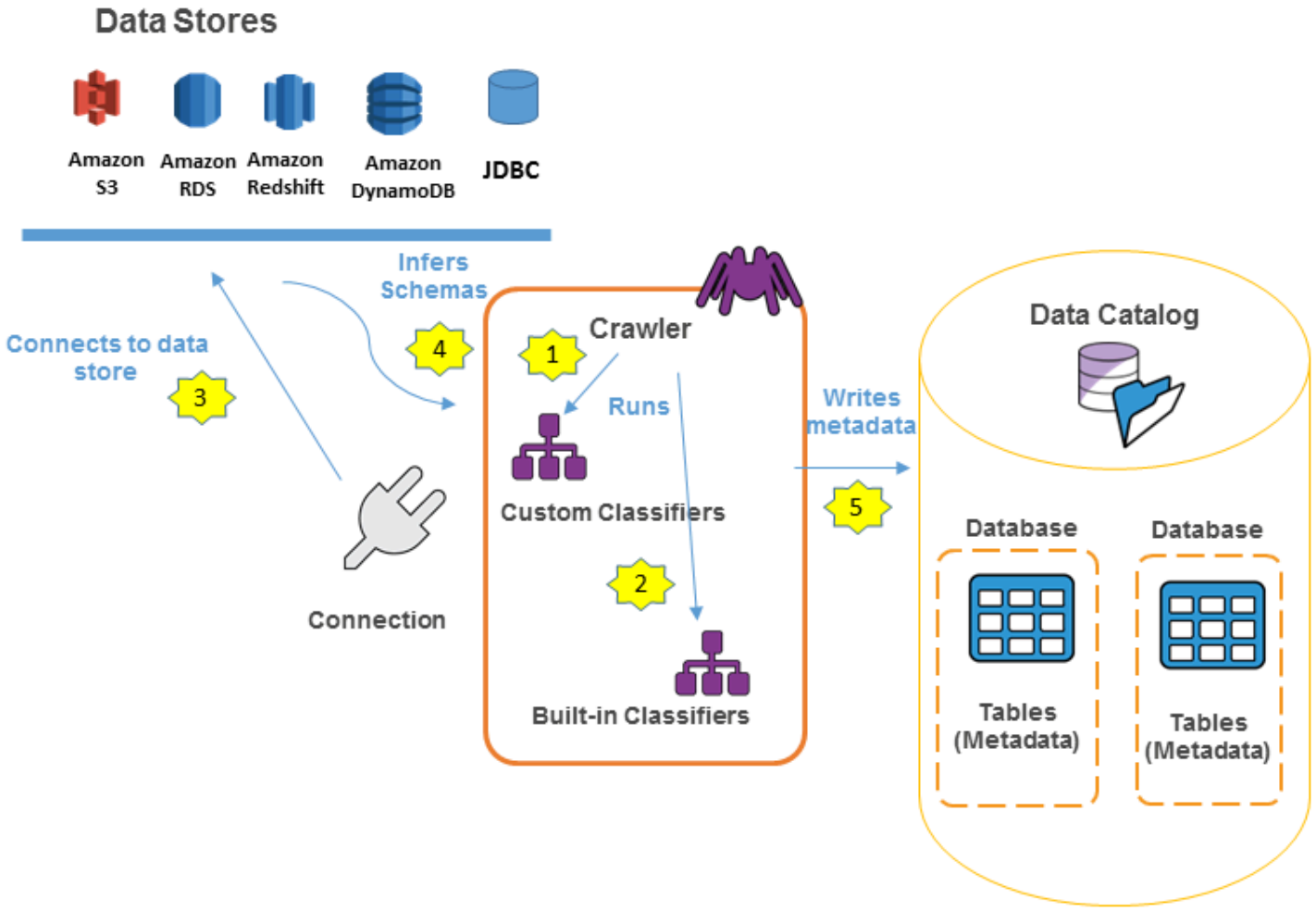
Topik

Menggunakan crawler untuk mengisi Katalog Data

Anda dapat menggunakan sebuah Perayap AWS Glue untuk mengisi AWS Glue Data Catalog dengan database dan tabel. Ini adalah metode utama yang digunakan oleh sebagian besar AWS Glue pengguna. Sebuah crawler dapat melakukan perayapan pada beberapa penyimpanan data dalam satu kali eksekusi. Setelah selesai, crawler tersebut membuat atau memperbarui satu atau beberapa tabel dalam Katalog Data Anda. Tugas extract, transform, and load (ETL) yang Anda tentukan di AWS Glue menggunakan tabel Katalog Data ini sebagai sumber dan target. Tugas ETL membaca dari dan menulis ke penyimpanan data yang ditentukan dalam sumber dan target tabel Katalog Data.

Alur kerja

Diagram alur kerja berikut menunjukkan bagaimana crawler AWS Glue berinteraksi dengan penyimpanan data dan elemen lain untuk mengisi Katalog Data.



Berikut ini adalah alur kerja umum bagaimana crawler mengisi AWS Glue Data Catalog:

1. Sebuah crawler menjalankan setiap pengklasifikasi kustom yang Anda pilih untuk menyimpulkan format dan skema data Anda. Anda menyediakan kode untuk pengklasifikasi kustom, dan mereka berjalan dengan urutan yang Anda tentukan.

   Pengklasifikasi kustom pertama yang berhasil mengenali struktur data Anda digunakan untuk membuat sebuah skema. Pengklasifikasi kustom yang ada di bagian bawah daftar dilewati.

2. Jika tidak ada pengklasifikasi kustom yang cocok dengan skema data Anda, maka pengklasifikasi bawaan mencoba mengenali skema data Anda. Contoh pengklasifikasi bawaan adalah pengklasifikasi yang mengenali JSON.
3. Crawler terhubung ke penyimpanan data. Beberapa penyimpanan data memerlukan properti koneksi untuk akses crawler.
4. Skema kesimpulan dibuat untuk data Anda.
5. Crawler menulis metadata ke Katalog Data. Definisi tabel berisi metadata tentang data yang ada di penyimpanan data Anda. Tabel tersebut ditulis ke sebuah basis data, yang merupakan kontainer dari tabel dalam Katalog Data. Atribut sebuah tabel termasuk klasifikasi, yang merupakan label yang dibuat oleh pengklasifikasi yang telah menyimpulkan skema tabel.

Topik

Cara kerja crawler

Saat sebuah crawler berjalan, crawler tersebut melakukan tindakan berikut untuk mengambil data dari penyimpanan data:

- Mengklasifikasikan data untuk menentukan format, skema, dan properti terkait data mentah — Anda dapat mengkonfigurasi hasil klasifikasi dengan membuat sebuah pengklasifikasi kustom.
- Mengelompokkan data ke dalam tabel atau partisi — Data dikelompokkan dalam grup berdasarkan heuristik crawler.

- Menulis metadata ke Katalog Data — Anda dapat mengkonfigurasi bagaimana crawler menambahkan, memperbarui, dan menghapus tabel dan partisi.

Saat menentukan crawler, Anda memilih satu atau beberapa pengklasifikasi yang mengevaluasi format data Anda untuk menyimpulkan sebuah skema. Ketika crawler tersebut berjalan, pengklasifikasi pertama dalam daftar Anda, agar berhasil mengenali penyimpanan data Anda, digunakan untuk membuat sebuah skema untuk tabel Anda. Anda dapat menggunakan pengklasifikasi bawaan atau menggunakan pengklasifikasi Anda sendiri. Anda menentukan pengklasifikasi kustom Anda dalam operasi terpisah, sebelum Anda menentukan crawler. AWS Glue menyediakan pengklasifikasi bawaan untuk menyimpulkan skema dari file umum dengan format yang mencakup JSON, CSV, dan Apache Avro. Untuk daftar terkini dari pengklasifikasi bawaan di AWS Glue, lihat Pengklasifikasi bawaan di AWS Glue.

Tabel metadata yang dibuat crawler terkandung dalam sebuah basis data ketika Anda menentukan sebuah crawler. Jika crawler Anda tidak menentukan sebuah basis data, maka tabel Anda ditempatkan di basis data default. Selain itu, setiap tabel memiliki sebuah kolom klasifikasi yang diisi oleh pengklasifikasi yang pertama kali berhasil mengenali penyimpanan data.

Jika file yang di-crawl dikompresi, maka crawler harus mengunduhnya untuk memprosesnya. Ketika sebuah crawler berjalan, crawler tersebut mengambil data dari file untuk menentukan format dan jenis kompresi mereka dan menulis properti ini ke dalam Katalog Data. Beberapa format file (misalnya, Apache Parket) memungkinkan Anda untuk meng-kompresi bagian dari file seperti yang tertulis. Untuk file ini, data yang terkompresi adalah komponen internal file, dan AWS Glue tidak mengisi properti `compressionType` ketika menulis tabel ke dalam Katalog Data. Sebaliknya, jika seluruh file dikompresi oleh sebuah algoritme kompresi (misalnya, gzip), maka properti `compressionType` diisi ketika tabel ditulis ke dalam Katalog Data.

Crawler menghasilkan nama-nama untuk tabel yang dibuatnya. Nama-nama tabel yang disimpan dalam AWS Glue Data Catalog mengikuti aturan ini:

- Hanya karakter alfanumerik dan garis bawah (_) yang boleh digunakan.
- Prefiks kustom tidak boleh lebih dari 64 karakter.
- Panjang nama maksimum tidak dapat lebih panjang dari 128 karakter. Crawler memotong nama yang dihasilkan agar sesuai dengan batasan.
- Jika ada nama tabel duplikat yang ditemukan, maka crawler menambahkan akhiran string hash ke nama tersebut.

Jika crawler Anda berjalan lebih dari sekali, mungkin berdasarkan jadwal, maka crawler tersebut akan mencari file atau tabel baru atau yang diubah di penyimpanan data Anda. Output dari crawler termasuk tabel baru dan partisi yang ditemukan sejak eksekusi sebelumnya.

Penyimpanan data mana yang dapat saya jelajahi?

Crawler dapat melakukan perayapan pada penyimpanan data berbasis file dan berbasis tabel berikut.

Jenis akses yang digunakan oleh crawler	Penyimpanan data
Klien asli	• Amazon Simple Storage Service (Amazon S3) • Amazon DynamoDB • Danau Delta 2.0.x • Gunung Es Apache 1.5 • Apache Hudi 0,14
JDBC	Amazon Redshift Kepingan salju Dalam Amazon Relational Database Service (Amazon RDS) atau eksternal ke Amazon RDS: • Amazon Aurora • MariaDB • Microsoft SQL Server • MySQL • Oracle • PostgreSQL
Klien MongoDB	• MongoDB • MongoDB Atlas • Amazon DocumentDB (dengan kompatibilitas MongoDB)

> **Note**
>
> Saat ini AWS Glue tidak mendukung crawler untuk aliran data.

Untuk penyimpanan data JDBC, MongoDB, MongoDB Atlas, dan Amazon DocumentDB (dengan kompatibilitas MongoDB), Anda harus menentukan koneksi yang dapat digunakan crawler untuk terhubung ke penyimpanan data. AWS Glue Untuk Amazon S3, Anda dapat menentukan sebuah koneksi tipe Jaringan. Sebuah koneksi adalah objek Katalog Data yang menyimpan informasi koneksi, seperti kredensial, URL, informasi Amazon Virtual Private Cloud, dan banyak lagi. Untuk informasi selengkapnya, lihat Menghubungkan ke data.

Berikut ini adalah versi driver yang didukung oleh crawler:

Produk	Driver yang didukung crawler
PostgreSQL	42.2.1
Amazon Aurora	Sama seperti driver crawler asli
MariaDB	8.0.13
Microsoft SQL Server	6.1.0
MySQL	8.0.13
Oracle	11.2.2
Amazon Redshift	4.1
Kepingan salju	3.13.20
MongoDB	4.7.2
MongoDB Atlas	4.7.2

Berikut ini adalah catatan tentang berbagai penyimpanan data.

Amazon S3

Anda dapat memilih untuk melakukan perayapan pada sebuah path di akun Anda atau di akun lain. Jika semua file Amazon S3 dalam folder memiliki skema yang sama, maka crawler menciptakan sebuah tabel. Selain itu, jika objek Amazon S3 dipartisi, maka hanya akan ada satu tabel metadata yang dibuat dan informasi partisi ditambahkan ke Katalog Data untuk tabel tersebut.

Amazon S3 dan Amazon DynamoDB

Crawler menggunakan peran AWS Identity and Access Management (IAM) untuk izin mengakses penyimpanan data Anda. Peran yang Anda berikan ke crawler harus memiliki izin untuk mengakses path Amazon S3 dan tabel Amazon DynamoDB yang di-crawl.

Amazon DynamoDB

Saat menentukan crawler menggunakan konsol AWS Glue, Anda menentukan satu tabel DynamoDB. Jika Anda menggunakan API AWS Glue, Anda dapat menentukan sebuah daftar tabel. Anda dapat memilih untuk melakukan perayapan hanya pada sampel kecil data untuk mengurangi waktu aktif crawler.

Danau Delta

Untuk setiap penyimpanan data Delta Lake, Anda menentukan cara membuat tabel Delta:

- Buat tabel Native: Izinkan integrasi dengan mesin kueri yang mendukung kueri log transaksi Delta secara langsung. Untuk informasi lebih lanjut, lihat Menanyakan tabel Danau Delta.
- Buat tabel Symlink: Buat `_symlink_manifest` folder dengan file manifes yang dipartisi oleh tombol partisi, berdasarkan parameter konfigurasi yang ditentukan.

Gunung es

Untuk setiap penyimpanan data Iceberg, Anda menentukan jalur Amazon S3 yang berisi metadata untuk tabel Iceberg Anda. Jika crawler menemukan metadata tabel Iceberg, ia mendaftarkannya di Katalog Data. Anda dapat mengatur jadwal untuk crawler untuk menjaga tabel diperbarui.

Anda dapat menentukan parameter ini untuk penyimpanan data:

- Pengecualian: Memungkinkan Anda melewati folder tertentu.
- Kedalaman Traversal Maksimum: Menetapkan batas kedalaman yang dapat dirayapi crawler di bucket Amazon S3 Anda. Kedalaman traversal maksimum default adalah 10 dan kedalaman maksimum yang dapat Anda atur adalah 20.

Hudi

Untuk setiap penyimpanan data Hudi, Anda menentukan jalur Amazon S3 yang berisi metadata untuk tabel Hudi Anda. Jika crawler menemukan metadata tabel Hudi, ia mendaftarkannya di Katalog Data. Anda dapat mengatur jadwal untuk crawler untuk menjaga tabel diperbarui.

Anda dapat menentukan parameter ini untuk penyimpanan data:

- Pengecualian: Memungkinkan Anda melewati folder tertentu.
- Kedalaman Traversal Maksimum: Menetapkan batas kedalaman yang dapat dirayapi crawler di bucket Amazon S3 Anda. Kedalaman traversal maksimum default adalah 10 dan kedalaman maksimum yang dapat Anda atur adalah 20.

> **Note**
>
> Kolom stempel waktu dengan `millis` tipe logis akan ditafsirkan sebagai`bigint`, karena ketidakcocokan dengan Hudi 0.13.1 dan tipe stempel waktu. Resolusi dapat diberikan dalam rilis Hudi mendatang.

Tabel Hudi dikategorikan sebagai berikut, dengan implikasi spesifik untuk masing-masing:

- Copy on Write (CoW): Data disimpan dalam format kolom (Parket), dan setiap pembaruan membuat versi file baru selama penulisan.
- Merge on Read (MoR): Data disimpan menggunakan kombinasi format kolumnar (Parket) dan berbasis baris (Avro). Pembaruan dicatat ke file delta berbasis baris dan dipadatkan sesuai kebutuhan untuk membuat file kolumnar versi baru.

Dengan set data CoW, setiap kali ada pembaruan ke catatan, file yang berisi catatan ditulis ulang dengan nilai yang diperbarui. Dengan set data MoR, setiap kali ada pembaruan, Hudi hanya menulis baris untuk catatan yang berubah. MoR lebih cocok untuk beban kerja tulis atau perubahan berat dengan lebih sedikit pembacaan. CoW lebih cocok untuk beban kerja pembacaan berat pada data yang jarang berubah.

Hudi menyediakan tiga tipe kueri untuk mengakses data:

- Kueri snapshot: Kueri yang melihat snapshot terbaru dari tabel sebagai tindakan komit atau pemadatan yang diberikan. Untuk tabel MoR, kueri snapshot memapar status terbaru tabel dengan menggabungkan file dasar dan delta potongan file terbaru pada pada saat kueri.

- Kueri tambahan: Kueri hanya melihat data baru yang ditulis ke tabel, karena komit/pemadatan yang diberikan. Ini secara efektif menyediakan pengaliran perubahan untuk mengaktifkan data pipeline tambahan.
- Baca kueri yang dioptimalkan: Untuk tabel MoR, kueri melihat data terbaru yang dipadatkan. Untuk tabel CoW, kueri melihat data terbaru yang dikomit.

Untuk tabel Copy-On-Write, crawler membuat satu tabel di Katalog Data dengan serde. ReadOptimized `org.apache.hudi.hadoop.HoodieParquetInputFormat`

Untuk tabel Merge-On-Read, crawler membuat dua tabel di Katalog Data untuk lokasi tabel yang sama:

- Sebuah tabel dengan akhiran `_ro` yang menggunakan ReadOptimized `org.apache.hudi.hadoop.HoodieParquetInputFormat` serde.
- Tabel dengan akhiran `_rt` yang menggunakan RealTime Serde yang memungkinkan kueri Snapshot:. `org.apache.hudi.hadoop.realtime.HoodieParquetRealtimeInputFormat`

MongoDB dan Amazon DocumentDB (dengan kompatibilitas MongoDB)

Mendukung MongoDB versi 3.2 dan versi setelahnya. Anda dapat memilih untuk melakukan perayapan hanya pada sampel kecil data untuk mengurangi waktu aktif crawler.

Basis data relasional

Autentikasi dengan nama pengguna dan kata sandi basis data. Tergantung pada jenis mesin basis data, Anda dapat memilih objek yang di-crawl, seperti basis data, skema, dan tabel.

Kepingan salju

Crawler Snowflake JDBC mendukung crawling Table, External Table, View, dan Materialized View. Definisi Tampilan Terwujud tidak akan diisi.

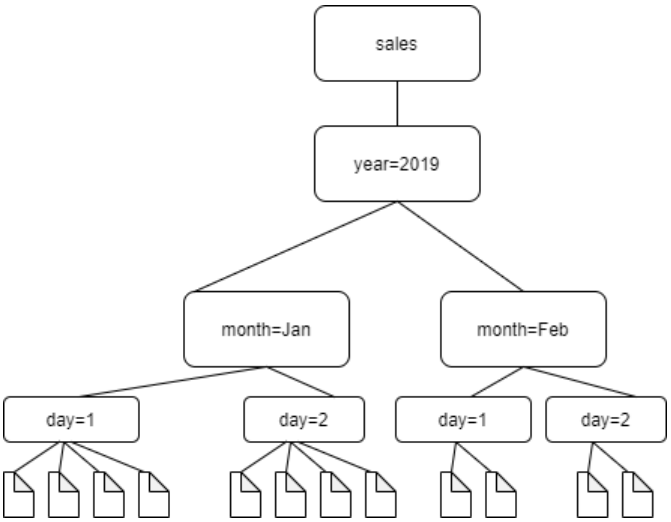
Untuk tabel eksternal Snowflake, crawler hanya akan merangkak jika menunjuk ke lokasi Amazon S3. Selain skema tabel, crawler juga akan merayapi lokasi Amazon S3, format file, dan output sebagai parameter tabel di tabel Katalog Data. Perhatikan bahwa informasi partisi dari tabel eksternal yang dipartisi tidak diisi.

ETL saat ini tidak didukung untuk tabel Katalog Data yang dibuat menggunakan crawler Snowflake.

Bagaimana crawler menentukan kapan harus membuat partisi?

Saat sebuah crawler AWS Glue memindai Amazon S3 dan mendeteksi beberapa folder dalam sebuah bucket, ia menentukan akar tabel dalam struktur folder dan folder mana yang merupakan partisi dari sebuah tabel. Nama tabel didasarkan pada prefiks Amazon S3 atau nama folder. Anda memberikan Termasuk path yang mengarahkan ke tingkat folder yang akan di-crawl. Ketika sebagian besar skema pada tingkat folder serupa, crawler tersebut menciptakan partisi dari sebuah tabel bukan tabel terpisah. Untuk mempengaruhi crawler agar membuat tabel terpisah, tambahkan folder akar dari setiap tabel sebagai penyimpanan data terpisah ketika Anda menentukan crawler.

Sebagai contoh, pertimbangkan struktur folder Amazon S3 berikut.



Path ke folder empat tingkat terendah adalah sebagai berikut:

```
S3://sales/year=2019/month=Jan/day=1
S3://sales/year=2019/month=Jan/day=2
S3://sales/year=2019/month=Feb/day=1
S3://sales/year=2019/month=Feb/day=2
```

Asumsikan bahwa target crawler ditetapkan pada `Sales`, dan bahwa semua file dalam folder `day=`*n* mempunyai format yang sama (sebagai contoh, JSON, tidak dienkripsi), dan mempunyai skema yang sama atau sangat serupa. Crawler tersebut akan membuat sebuah tabel tunggal dengan empat partisi, dengan kunci partisi `year`, `month`, dan `day`.

Pada contoh selanjutnya, pertimbangkan struktur Amazon S3 berikut:

```
s3://bucket01/folder1/table1/partition1/file.txt
s3://bucket01/folder1/table1/partition2/file.txt
s3://bucket01/folder1/table1/partition3/file.txt
s3://bucket01/folder1/table2/partition4/file.txt
s3://bucket01/folder1/table2/partition5/file.txt
```

Jika skema untuk file di bawah `table1` dan `table2` skemanya serupa, dan penyimpanan data tunggal didefinisikan dalam crawler dengan Termasuk path `s3://bucket01/folder1/`, maka crawler membuat sebuah tabel tunggal dengan dua kolom kunci partisi. Kolom kunci partisi pertama berisi `table1` dan `table2`, dan kolom kunci partisi kedua berisi `partition1` hingga `partition3` untuk partisi `table1` dan `partition4` dan `partition5` untuk partisi `table2`. Untuk membuat dua tabel terpisah, tentukan crawler dengan dua penyimpanan data. Dalam contoh ini, tentukan Termasuk path yang pertama sebagai `s3://bucket01/folder1/table1/` dan yang kedua sebagai `s3://bucket01/folder1/table2`.

> **Note**
>
> Di Amazon Athena, setiap tabel sesuai dengan prefiks Amazon S3 dengan semua objek di dalamnya. Jika objek memiliki skema yang berbeda, maka Athena tidak akan mengenali objek yang berbeda dalam prefiks yang sama sebagai tabel terpisah. Hal ini dapat terjadi jika crawler membuat beberapa tabel dari prefiks Amazon S3 yang sama. Hal ini mungkin menyebabkan kueri di Athena mengembalikan hasil nol. Bagi Athena, untuk dapat mengenali dan meng-kueri tabel dengan benar, buatlah crawler dengan Termasuk path terpisah untuk setiap skema tabel yang berbeda dalam struktur folder Amazon S3. Untuk informasi selengkapnya, lihat Praktik Terbaik Saat Menggunakan Athena dengan AWS Glue dan Artikel Pusat Pengetahuan AWS ini.

Prasyarat perayap

Crawler mengasumsikan izin peran AWS Identity and Access Management (IAM) yang Anda tentukan saat Anda mendefinisikannya. IAM role ini harus memiliki izin untuk mengekstrak data dari penyimpanan data Anda dan menuliskannya ke Katalog Data. Konsol AWS Glue hanya mencantumkan IAM role yang telah melampirkan kebijakan kepercayaan saja untuk layanan prinsipal utama AWS Glue. Dari konsol tersebut, Anda juga dapat membuat IAM role dengan kebijakan IAM untuk mengakses penyimpanan data Amazon S3 yang diakses oleh crawler. Untuk informasi lebih lanjut tentang menyediakan peran untuk AWS Glue, lihat Kebijakan berbasis identitas untuk Glue AWS.

> **Note**
>
> Saat merayapi penyimpanan data Delta Lake, Anda harus memiliki izin Baca/Tulis ke lokasi Amazon S3.

Untuk crawler Anda, Anda dapat membuat sebuah peran dan melampirkan kebijakan berikut:

- Kebijakan `AWSGlueServiceRole` AWS terkelola, yang memberikan izin yang diperlukan pada Katalog Data
- Sebuah kebijakan inline yang memberikan izin pada sumber data.
- Kebijakan inline yang memberikan `iam:PassRole` izin pada peran tersebut.

Pendekatan yang lebih cepat adalah membiarkan penuntun crawler konsol AWS Glue membuat peran untuk Anda. Peran yang dibuatnya khusus untuk crawler, dan menyertakan kebijakan `AWSGlueServiceRole` AWS terkelola ditambah kebijakan sebaris yang diperlukan untuk sumber data yang ditentukan.

Jika Anda menentukan peran yang ada untuk sebuah crawler, pastikan bahwa crawler tersebut menyertakan kebijakan `AWSGlueServiceRole` atau kebijakan yang setara (atau versi lingkup diperkecil dari kebijakan ini), ditambah kebijakan inline yang diperlukan. Sebagai contoh, untuk penyimpanan data Amazon S3, kebijakan inline minimal harus berupa kebijakan berikut:

```
{
   "Version": "2012-10-17",
    "Statement": [
        {
          "Effect": "Allow",
          "Action": [
              "s3:GetObject",
              "s3:PutObject"
          ],
          "Resource": [
              "arn:aws:s3:::bucket/object*"
          ]
        }
    ]
}
```

Untuk penyimpanan data Amazon DynamoDB, kebijakan minimalnya harus berupa kebijakan berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:DescribeTable",
        "dynamodb:Scan"
      ],
      "Resource": [
        "arn:aws:dynamodb:region:account-id:table/table-name*"
      ]
    }
  ]
}
```

Selain itu, jika crawler membaca AWS Key Management Service (AWS KMS) data Amazon S3 yang dienkripsi, maka peran IAM harus memiliki izin dekripsi pada kunci. AWS KMS Untuk informasi selengkapnya, lihat Langkah 2: Buat peran IAM untuk AWS Glue.

Mengkonfigurasi crawler

Crawler mengakses penyimpanan data Anda, mengekstrak metadata, dan membuat definisi tabel di AWS Glue Data Catalog. Panel Crawler di AWS Glue konsol mencantumkan semua crawler yang Anda buat. Daftar ini menampilkan status dan metrik dari eksekusi crawler terakhir Anda.

> **Note**
>
> Jika Anda memilih untuk membawa versi driver JDBC Anda sendiri, AWS Glue crawler akan menggunakan sumber daya dalam AWS Glue pekerjaan dan bucket Amazon S3 untuk memastikan driver yang Anda berikan dijalankan di lingkungan Anda. Penggunaan sumber daya tambahan akan tercermin di akun Anda. Selain itu, menyediakan driver JDBC Anda sendiri tidak berarti bahwa crawler dapat memanfaatkan semua fitur pengemudi. Driver terbatas pada properti yang dijelaskan dalam Menambahkan AWS Glue koneksi.

Untuk mengkonfigurasi crawler

1. Masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/. Pilih Crawler di panel navigasi.
2. Pilih Create crawler, dan ikuti petunjuk di Add crawler wizard. Wizard akan memandu Anda langkah-langkah yang diperlukan untuk membuat crawler. Jika Anda ingin menambahkan calssifier khusus untuk menentukan skema, lihat. Menambahkan pengklasifikasi ke crawler di AWS Glue

Langkah 1: Mengatur properti crawler

Masukkan nama untuk crawler dan deskripsi Anda (opsional). Secara opsional, Anda dapat menandai crawler dengan sebuah Kunci tag dan Nilai tag opsional. Setelah dibuat, kunci tag bersifat baca-saja. Gunakan tag ke sumber daya Anda untuk membantu mengatur dan mengidentifikasi sumber daya tersebut. Untuk informasi selengkapnya, lihat AWS tag di AWS Glue.

Nama

Nama dapat berisi huruf (A-Z), angka (0-9), tanda hubung (-), atau garis bawah (_), dan dapat mencapai 255 karakter.

Deskripsi

Deskripsi dapat mencapai 2048 karakter.

Tanda

Gunakan tag untuk mengatur dan mengidentifikasi sumber daya Anda. Untuk informasi selengkapnya, lihat berikut ini:

- AWS tag di AWS Glue

Langkah 2: Pilih sumber data dan pengklasifikasi

Konfigurasi sumber data

Pilih opsi yang sesuai untuk Apakah data Anda sudah dipetakan ke AWS Glue tabel? pilih 'Belum lagi' atau 'Ya'. Secara default, 'Belum lagi' dipilih.

Crawler dapat mengakses penyimpanan data secara langsung sebagai sumber perayapan, atau crawler dapat menggunakan tabel yang ada di Katalog Data sebagai sumbernya. Jika

crawler tersebut menggunakan tabel katalog yang ada, maka crawler melakukan perayapan pada penyimpanan data yang ditentukan oleh tabel katalog tersebut.

- Belum: Pilih satu atau beberapa sumber data yang akan dirayapi. Sebuah crawler dapat melakukan perayapan pada beberapa penyimpanan data dari berbagai jenis (Amazon S3, JDBC, dan sebagainya).

  Anda dapat mengkonfigurasi hanya satu penyimpanan data dalam satu waktu. Setelah Anda telah memberikan informasi koneksi dan termasuk path dan mengecualikan pola, Anda kemudian memiliki pilihan untuk menambahkan penyimpanan data yang lain.

- Ya: Pilih tabel yang ada dari Katalog AWS Glue Data Anda. Tabel katalog menentukan penyimpanan data yang akan di-crawl. Crawler hanya dapat melakukan perayapan tabel katalog dalam satu kali eksekusi; tidak dapat dicampur jenis sumber lainnya.

  Alasan umum untuk menentukan tabel katalog sebagai sumber adalah ketika Anda membuat tabel secara manual (karena Anda sudah tahu struktur penyimpanan data-nya) dan Anda ingin sebuah crawler selalu memperbarui tabel, termasuk menambahkan partisi baru. Untuk diskusi tentang alasan yang lain, lihat Memperbarui tabel Katalog Data yang dibuat secara manual menggunakan crawler.

  Bila Anda menentukan tabel yang ada sebagai jenis sumber crawler, maka kondisi berikut berlaku:

  - Nama basis data bersifat opsional.
  - Hanya tabel katalog yang menentukan penyimpanan data Amazon S3 atau Amazon DynamoDB saja yang diizinkan.
  - Tidak ada tabel katalog baru yang dibuat saat crawler berjalan. Tabel yang ada diperbarui sesuai kebutuhan, termasuk melakukan penambahan partisi baru.
  - Objek dihapus yang ditemukan di penyimpanan data akan diabaikan; tidak ada tabel katalog yang dihapus. Sebaliknya, crawler tersebut menulis pesan log. (`SchemaChangePolicy.DeleteBehavior=LOG`)
  - Opsi konfigurasi crawler untuk membuat satu skema tunggal untuk setiap path Amazon S3 diaktifkan secara default dan tidak dapat dinonaktifkan. (`TableGroupingPolicy=CombineCompatibleSchemas`) Untuk informasi lebih lanjut, lihatCara membuat skema tunggal untuk setiap Amazon S3 termasuk jalur.
  - Anda tidak dapat mencampur tabel katalog sebagai sebuah sumber dengan jenis sumber lainnya (misalnya Amazon S3 atau Amazon DynamoDB).

Sumber data

Pilih atau tambahkan daftar sumber data yang akan dipindai oleh crawler.

(Opsional) Jika Anda memilih JDBC sebagai sumber data, Anda dapat menggunakan driver JDBC Anda sendiri saat menentukan akses Koneksi tempat info driver disimpan.

Termasuk path

Saat mengevaluasi apa yang harus disertakan atau dikecualikan dalam perayapan, crawler akan memulai dengan mengevaluasi penyertaan path yang diperlukan. Untuk Amazon S3, MongoDB, MongoDB Atlas, Amazon DocumentDB (dengan kompatibilitas MongoDB), dan penyimpanan data relasional, Anda harus menentukan jalur sertakan.

Untuk penyimpanan data Amazon S3

Pilih apakah akan menentukan jalur di akun ini atau di akun lain, lalu telusuri untuk memilih jalur Amazon S3.

Untuk penyimpanan data Amazon S3, sertakan sintaksis path `bucket-name/folder-name/file-name.ext`. Untuk melakukan perayapan pada semua objek dalam sebuah bucket, Anda harus menentukan hanya nama bucket dalam penyertaan path. Pola pengecualian relatif terhadap penyertaan path

Untuk penyimpanan data Delta Lake

Tentukan satu atau beberapa jalur Amazon S3 ke tabel Delta sebagai s3://bucket/prefix/object.

Untuk penyimpanan data Iceberg atau Hudi

Tentukan satu atau beberapa jalur Amazon S3 yang berisi folder dengan metadata tabel Iceberg atau Hudi sebagai awalan s3://bucket/.

Untuk penyimpanan data Hudi, folder Hudi mungkin terletak di folder anak dari folder root. Crawler akan memindai semua folder di bawah jalur untuk folder Hudi.

Untuk penyimpanan data JDBC

Masukkan *<database>*/*<schema>*/*<table>* atau *<database>*/*<table>*, tergantung pada produk basis data. Basis Data Oracle dan MySQL tidak mendukung skema dalam path. Anda dapat mengganti karakter persen (%) untuk *<schema>* atau *<table>*. Sebagai contoh, untuk basis data Oracle dengan pengenal sistem (SID) dari `orcl`, masukkan `orcl/%` untuk mengimpor semua tabel yang diberi nama oleh pengguna dalam koneksi yang ia miliki aksesnya.

> **Important**
>
> Bidang ini peka huruf besar dan kecil.

Untuk penyimpanan data MongoDB, MongoDB Atlas, atau Amazon DocumentDB

Masukkan *database*/*collection*.

Untuk MongoDB, MongoDB Atlas, dan Amazon DocumentDB (dengan kompatibilitas MongoDB), sintaksnya adalah. `database/collection`

Untuk penyimpanan data JDBC, sintaksisnya adalah `database-name/schema-name/table-name` atau `database-name/table-name`, salah satunya. Sintaksis tergantung pada apakah mesin basis data mendukung skema dalam sebuah basis data. Misalnya, untuk mesin basis data seperti MySQL atau Oracle, jangan menentukan `schema-name` di penyertaan path Anda. Anda dapat mengganti tanda persen (`%`) untuk skema atau tabel dalam penyertaan path untuk mewakili semua skema atau semua tabel dalam sebuah basis data. Anda tidak dapat mengganti tanda persen (`%`) untuk basis data dalam penyertaan path.

Kedalaman transversal maksimum (hanya untuk penyimpanan data Iceberg atau Hudi)

Menentukan kedalaman maksimum jalur Amazon S3 yang dapat dilalui crawler untuk menemukan folder metadata Iceberg atau Hudi di jalur Amazon S3 Anda. Tujuan dari parameter ini adalah untuk membatasi waktu berjalan crawler. Nilai default adalah 10 dan maksimum adalah 20.

Mengecualikan pola

Hal ini memungkinkan Anda untuk mengecualikan file atau tabel tertentu dari perayapan. Pengecualian path relatif terhadap penyertaan path. Misalnya, untuk mengecualikan tabel di penyimpanan data JDBC Anda, ketik nama tabel dalam pengecualian path.

Sebuah crawler menghubungkan ke sebuah penyimpanan data JDBC menggunakan koneksi AWS Glue yang berisi sebuah string koneksi URI JDBC. Crawler hanya memiliki akses ke objek yang ada di mesin basis data saja menggunakan nama pengguna dan kata sandi JDBC dalam koneksi AWS Glue. Crawler hanya dapat membuat tabel yang dapat diakses melalui koneksi JDBC. Setelah crawler mengakses mesin basis data dengan URI JDBC, penyertaan path digunakan untuk menentukan tabel di mesin basis data yang dibuat dalam Katalog Data. Sebagai contoh, dengan MySQL, jika Anda menentukan penyertaan path `MyDatabase/%`, maka semua tabel dalam `MyDatabase` dibuat dalam Katalog Data. Saat mengakses Amazon Redshift, jika Anda menentukan penyertaan path `MyDatabase/%`, maka semua tabel dalam semua

skema untuk basis data `MyDatabase` akan dibuat dalam Katalog Data. Jika Anda menentukan penyertaan path `MyDatabase/MySchema/%`, maka semua tabel dalam basis data `MyDatabase` dan skema `MySchema` dibuat.

Setelah Anda menentukan penyertaan path, maka Anda kemudian dapat mengecualikan objek dari perayapan di mana penyertaan path Anda akan dinyatakan termasuk dengan menentukan satu atau beberapa gaya pola pengecualian `glob` dengan gaya Unix. Pola ini diterapkan ke penyertaan path Anda untuk menentukan objek yang dikecualikan. Pola-pola ini juga disimpan sebagai properti tabel yang dibuat oleh crawler. AWS Glue PySpark ekstensi, seperti`create_dynamic_frame.from_catalog`, membaca properti tabel dan mengecualikan objek yang ditentukan oleh pola pengecualian.

AWS Glue mendukung jenis-jenis pola `glob` berikut dalam pola pengecualian.

Pola pengecualian	Deskripsi
`*.csv`	Cocok dengan path Amazon S3 yang mewakili nama objek dalam folder saat ini yang diakhiri dengan `.csv`
`*.*`	Cocok dengan semua nama objek yang berisi sebuah titik
`*.{csv,avro}`	Cocok nama objek yang diakhiri dengan `.csv` atau `.avro`
`foo.?`	Cocok nama objek yang dimulai dengan `foo.` yang diikuti oleh ekstensi karakter tunggal
`myfolder/*`	Cocok dengan objek dalam satu tingkat subfolder dari `myfolder`, seperti `/myfolder/mysource`
`myfolder/*/*`	Cocok dengan objek dalam dua tingkat subfolder dari `myfolder`, seperti `/myfolder/mysource/data`
`myfolder/**`	Cocok dengan objek dalam semua subfolder dari `myfolder`, seperti `/myfolder/`

Pola pengecualian	Deskripsi
	`mysource/mydata` dan `/myfolder/mysource/data`
`myfolder**`	Cocok dengan subfolder `myfolder` serta file di bawah `myfolder`, seperti `/myfolder` dan `/myfolder/mydata.txt`
`Market*`	Cocok dengan tabel dalam sebuah basis data JDBC dengan nama-nama yang dimulai dengan `Market`, seperti `Market_us` dan `Market_fr`

AWS Glue menafsirkan pola pengecualian `glob` sebagai berikut:

- Karakter garis miring (`/`) adalah pembatas untuk memisahkan kunci Amazon S3 ke dalam sebuah hierarki folder.
- Karakter tanda bintang (`*`) cocok dengan karakter nol atau karakter lain dari komponen nama tanpa melintasi batas folder.
- Tanda bintang ganda (`**`) cocok dengan karakter nol karakter lainnya yang melintasi folder atau skema batas.
- Karakter tanda tanya (`?`) cocok persis dengan satu karakter dari komponen nama.
- Karakter garis miring terbalik (`\`) digunakan untuk meng-escape karakter yang jika tidak dapat diartikan sebagai karakter khusus. Ekspresi `\\` cocok dengan backslash tunggal, dan `\{` cocok dengan kurung kiri.
- Kurung `[ ]` membuat ekspresi kurung yang cocok dengan karakter tunggal dari komponen nama dari satu set karakter. Misalnya, `[abc]` cocok dengan `a`, `b`, atau `c`. Tanda hubung (`-`) dapat digunakan untuk menentukan rentang, sehingga `[a-z]` menentukan rentang yang cocok dari `a` hingga `z` (inklusif). Bentuk-bentuk ini dapat dicampur, jadi `[abce-g]` cocok dengan `a`, `b`, `c`, `e`, `f`, atau `g`. Jika karakter setelah kurung (`[`) adalah tanda seru (`!`), maka ekspresi kurung dinegasikan. Misalnya, `[!a-c]` cocok dengan karakter apapun kecuali `a`, `b`, atau `c`.

  Dalam sebuah ekspresi kurung, karakter `*`, `?`, dan `\` cocok dengannya sendiri. Karakter tanda hubung (`-`) cocok dengan dirinya sendiri jika ia merupakan karakter pertama dalam kurung, atau jika ia adalah karakter pertama setelah `!` ketika Anda melakukan negasi.

- Kurung (`{ }`) melampirkan sebuah grup subpola, di grup tersebut cocok jika setiap subpola dalam grup cocok. Karakter koma (`,`) digunakan untuk memisahkan subpola. Grup tidak dapat disarangkan.
- Karakter titik atau titik didepan dalam nama file diperlakukan sebagai karakter normal dalam operasi pencocokan. Misalnya, pola pengecualian `*` cocok dengan nama file `.hidden`.

Example Amazon S3 mengecualikan pola

Setiap pola pengecualian dievaluasi terhadap pola penyertaan. Sebagai contoh, asumsikan bahwa Anda memiliki struktur direktori Amazon S3 berikut:

```
/mybucket/myfolder/
   departments/
      finance.json
      market-us.json
      market-emea.json
      market-ap.json
   employees/
      hr.json
      john.csv
      jane.csv
      juan.txt
```

Mengingat penyertaan path `s3://mybucket/myfolder/`, berikut ini adalah beberapa hasil sampel untuk pola pengecualian:

Pola pengecualian	Hasil
`departments/**`	Mengecualikan semua file dan folder di bawah `departments` dan mencakup folder `employees` dan file-nya
`departments/market*`	Mengecualikan `market-us.json` , `market-emea.json` , dan `market-ap.json`
`**.csv`	Mengecualikan semua objek di bawah `myfolder` yang memiliki nama yang diakhiri dengan `.csv`

Pola pengecualian	Hasil
`employees/*.csv`	Mengecualikan semua file `.csv` dalam folder `employees`

Example Tidak termasuk subset partisi Amazon S3

Misalkan data Anda dipartisi berdasarkan hari, sehingga setiap hari dalam setahun berada dalam partisi Amazon S3 yang terpisah. Untuk Januari 2015, ada 31 partisi. Sekarang, untuk melakukan perayapan pada data hanya untuk minggu pertama bulan Januari, Anda harus mengecualikan semua partisi kecuali hari 1 sampai 7:

```
2015/01/{[!0],0[8-9]}**, 2015/0[2-9]/**, 2015/1[0-2]/**
```

Lihatlah bagian-bagian dari pola glob ini. Bagian pertama, `2015/01/{[!0],0[8-9]}**`, mengecualikan semua hari yang tidak dimulai dengan "0" selain hari 08 dan hari 09 dari bulan 01 di tahun 2015. Perhatikan bahwa "**" digunakan sebagai akhiran untuk pola angka hari dan melintasi batas folder ke folder tingkat bawah. Jika "*" digunakan, maka tingkat folder yang lebih rendah tidak dikecualikan.

Bagian kedua, `2015/0[2-9]/**`, mengecualikan hari dalam bulan 02 hingga 09, pada tahun 2015.

Bagian ketiga, `2015/1[0-2]/**`, mengecualikan hari dalam bulan 10, 11, dan 12, pada tahun 2015.

Example JDBC mengecualikan pola

Misalkan Anda melakukan perayapan pada basis data JDBC dengan struktur skema berikut:

```
MyDatabase/MySchema/
   HR_us
   HR_fr
   Employees_Table
   Finance
   Market_US_Table
   Market_EMEA_Table
```

```
Market_AP_Table
```

Mengingat penyertaan path `MyDatabase/MySchema/%`, berikut ini adalah beberapa hasil sampel untuk pola pengecualian:

Pola pengecualian	Hasil
`HR*`	Mengecualikan tabel dengan nama yang dimulai dengan `HR`
`Market_*`	Mengecualikan tabel dengan nama yang dimulai dengan `Market_`
`**_Table`	Mengecualikan semua tabel dengan nama yang diakhiri dengan `_Table`

Parameter sumber crawler tambahan

Setiap jenis sumber memerlukan seperangkat parameter tambahan yang berbeda. Berikut ini bukan daftar lengkap:

Koneksi

Pilih atau tambahkan koneksi AWS Glue. Untuk informasi tentang koneksi, lihat Menghubungkan ke data.

Metadata tambahan - opsional (untuk penyimpanan data JDBC)

Pilih properti metadata tambahan untuk crawler untuk dirayapi.

- Komentar: Merayapi komentar tingkat tabel dan tingkat kolom terkait.
- Jenis mentah: Pertahankan tipe data mentah dari kolom tabel dalam metadata tambahan. Sebagai perilaku default, crawler menerjemahkan tipe data mentah ke tipe yang kompatibel dengan HIVE.

Nama Kelas Driver JDBC - opsional (untuk penyimpanan data JDBC)

Ketik nama kelas driver JDBC kustom agar crawler dapat terhubung ke sumber data:

- Postgres: org.PostgreSQL.Driver
- MySQL: com.mysql.jdbc.driver, com.mysql.cj.jdbc.driver

- Redshift: com.amazon.redshift.jdbc.driver, com.amazon.redshift.jdbc42.driver
- Oracle: oracle.jdbc.driver. OracleDriver
- SQL Server: com.Microsoft.SqlServer.jdbc.sql ServerDriver

JDBC Driver S3 Path - opsional (untuk penyimpanan data JDBC)

Pilih jalur Amazon S3 yang ada ke file. `.jar` Di sinilah `.jar` file akan disimpan saat menggunakan driver JDBC khusus agar crawler dapat terhubung ke sumber data.

Aktifkan pengambilan sampel data (hanya untuk penyimpanan data Amazon DynamoDB, MongoDB, MongoDB Atlas, dan Amazon DocumentDB)

Pilih apakah akan melakukan perayapan pada sampel data saja. Jika tidak dipilih, maka seluruh tabel akan di-crawl. Memindai semua catatan dapat memakan waktu lama ketika tabel tersebut bukan merupakan tabel throughput tinggi.

Buat tabel untuk kueri (hanya untuk penyimpanan data Delta Lake)

Pilih bagaimana Anda ingin membuat tabel Delta Lake:

- Buat tabel Native: Izinkan integrasi dengan mesin kueri yang mendukung kueri log transaksi Delta secara langsung.
- Buat tabel Symlink: Buat folder manifes symlink dengan file manifes yang dipartisi oleh kunci partisi, berdasarkan parameter konfigurasi yang ditentukan.

Tingkat pemindaian - opsional (hanya untuk penyimpanan data DynamoDB)

Tentukan persentase tabel DynamoDB Membaca Unit Kapasitas yang akan digunakan oleh crawler. Unit kapasitas baca adalah istilah yang didefinisikan oleh DynamoDB, dan merupakan nilai numerik yang bertindak sebagai tingkat pembatar untuk jumlah baca yang dapat dilakukan pada tabel tersebut per detik. Masukkan nilai antara 0,1 dan 1,5. Jika tidak ditentukan, maka nilai default 0,5% akan digunakan untuk tabel sediaan dan 1/4 kapasitas dikonfigurasi maksimum untuk tabel sesuai permintaan. Perhatikan bahwa hanya mode kapasitas yang disediakan yang harus digunakan dengan AWS Glue crawler.

> **Note**
>
> Untuk penyimpanan data DynamoDB, atur mode kapasitas yang disediakan untuk memproses pembacaan dan penulisan di tabel Anda. AWS Glue Crawler tidak boleh digunakan dengan mode kapasitas sesuai permintaan.

Koneksi jaringan - opsional (hanya untuk penyimpanan data Amazon S3)

Secara opsional sertakan koneksi Jaringan untuk digunakan dengan target Amazon S3 ini. Perhatikan bahwa setiap crawler terbatas pada satu koneksi Jaringan sehingga target Amazon S3 lainnya juga akan menggunakan koneksi yang sama (atau tidak ada, jika dibiarkan kosong).

Untuk informasi tentang koneksi, lihat Menghubungkan ke data.

Sampel hanya sebagian file dan ukuran Sampel (hanya untuk penyimpanan data Amazon S3)

Tentukan jumlah file di setiap folder daun yang akan di-crawl saat melakukan perayapan pada file sampel dalam set data. Ketika fitur ini diaktifkan, alih-alih melakukan perayapan pada semua file dalam set data ini, crawler akan secara acak memilih beberapa file di setiap folder daun untuk di-crawl.

Crawler sampling paling cocok untuk pelanggan yang memiliki pengetahuan sebelumnya tentang format data mereka dan tahu bahwa skema pada folder mereka tidak berubah. Dengan mengaktifkan fitur ini akan secara signifikan mengurangi waktu aktif crawler.

Nilai yang valid adalah bilangan bulat antara 1 dan 249. Jika tidak ditentukan, maka semua file di-crawl.

Perayap selanjutnya berjalan

Bidang ini adalah bidang global yang memengaruhi semua sumber data Amazon S3.

- Merayapi semua sub-folder: Merayapi semua folder lagi dengan setiap perayapan berikutnya.
- Hanya crawl sub-folder baru: Hanya folder Amazon S3 yang ditambahkan sejak crawl terakhir yang akan dirayapi. Jika skema kompatibel, partisi baru akan ditambahkan ke tabel yang ada. Untuk informasi selengkapnya, lihat the section called “Perayapan tambahan untuk menambahkan partisi baru”.
- Perayapan berdasarkan peristiwa: Andalkan peristiwa Amazon S3 untuk mengontrol folder apa yang akan dirayapi. Untuk informasi selengkapnya, lihat the section called “Mempercepat crawl menggunakan notifikasi acara Amazon S3”.

Pengklasifikasi khusus - opsional

Tentukan pengklasifikasi kustom sebelum mendefinisikan crawler. Pengklasifikasi memeriksa apakah file tertentu dalam format yang dapat ditangani oleh crawler. Jika ya, maka pengklasifikasi menciptakan sebuah skema dalam bentuk objek `StructType` yang cocok dengan format data tersebut.

Untuk informasi selengkapnya, lihat Menambahkan pengklasifikasi ke crawler di AWS Glue.

Langkah 3: Konfigurasikan pengaturan keamanan

Peran IAM

Crawler mengambil peran ini. Itu harus memiliki izin yang mirip dengan kebijakan AWS `AWSGlueServiceRole` terkelola. Untuk sumber Amazon S3 dan DynamoDB, ia juga harus memiliki izin untuk mengakses penyimpanan data. Jika crawler membaca data Amazon S3 yang dienkripsi AWS Key Management Service dengan AWS KMS(), maka peran tersebut harus memiliki izin dekripsi pada kunci. AWS KMS

Untuk penyimpanan data Amazon S3, izin tambahan yang dilampirkan pada peran akan mirip dengan berikut ini:

```
{
   "Version": "2012-10-17",
    "Statement": [
        {
          "Effect": "Allow",
          "Action": [
              "s3:GetObject",
              "s3:PutObject"
          ],
          "Resource": [
              "arn:aws:s3:::bucket/object*"
          ]
        }
    ]
}
```

Untuk penyimpanan data Amazon DynamoDB, izin tambahan yang dilampirkan pada peran akan mirip dengan berikut ini:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:DescribeTable",
        "dynamodb:Scan"
      ],
      "Resource": [
```

```
        "arn:aws:dynamodb:region:account-id:table/table-name*"
      ]
    }
  ]
}
```

Untuk menambahkan driver JDBC Anda sendiri, izin tambahan perlu ditambahkan.

- Berikan izin untuk tindakan pekerjaan berikut:`CreateJob`,,`DeleteJob`, `GetJobGetJobRun`,`StartJobRun`.
- Berikan izin untuk tindakan Amazon S3`s3:DeleteObjects`:`s3:GetObject`,,,`s3:ListBucket`. `s3:PutObject`

> **Note**
>
> Tidak `s3:ListBucket` diperlukan jika kebijakan bucket Amazon S3 dinonaktifkan.

- Berikan akses utama layanan ke ember/folder dalam kebijakan Amazon S3.

Contoh kebijakan Amazon S3:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::bucket-name/driver-parent-folder/driver.jar",
                "arn:aws:s3:::bucket-name"
            ]
        }
    ]
}
```

AWS Gluemembuat folder berikut (`_crawler`dan `_glue_job_crawler` pada tingkat yang sama dengan driver JDBC di bucket Amazon S3 Anda. Misalnya, jika jalur driver`<s3-path/driver_folder/driver.jar>`, maka folder berikut akan dibuat jika belum ada:

- <s3-path/driver_folder/_crawler>
- <s3-path/driver_folder/_glue_job_crawler>

Opsional, Anda dapat menambahkan konfigurasi keamanan untuk sebuah crawler untuk menentukan opsi enkripsi secara at-rest.

Untuk informasi selengkapnya, lihat Langkah 2: Buat peran IAM untuk AWS Glue dan Manajemen identitas dan akses untuk AWS Glue.

Konfigurasi Lake Formation - opsional

Izinkan crawler menggunakan kredensil Lake Formation untuk merayapi sumber data.

Memeriksa kredenal Gunakan Lake Formation untuk merayapi sumber data S3 akan memungkinkan crawler menggunakan kredenal Lake Formation untuk merayapi sumber data. Jika sumber data milik akun lain, Anda harus memberikan ID akun terdaftar. Jika tidak, crawler hanya akan merayapi sumber data yang terkait dengan akun tersebut. Hanya berlaku untuk sumber data Amazon S3 dan Katalog Data.

Konfigurasi keamanan - opsional

Pengaturan termasuk konfigurasi keamanan. Untuk informasi selengkapnya, lihat berikut ini:

- Mengenkripsi data yang ditulis oleh AWS Glue

> **Note**
>
> Setelah konfigurasi keamanan diatur pada crawler, Anda dapat mengubahnya, tetapi Anda tidak dapat menghapusnya. Untuk menurunkan tingkat keamanan pada crawler, setel fitur keamanan secara eksplisit ke `DISABLED` dalam konfigurasi Anda, atau buat crawler baru.

Langkah 4: Atur output dan penjadwalan

Konfigurasi keluaran

Pilihannya meliputi bagaimana crawler harus menangani perubahan skema yang terdeteksi, objek yang dihapus dalam penyimpanan data, dan banyak lagi. Untuk informasi selengkapnya, lihat Menyesuaikan perilaku crawler

Jadwal crawler

Anda dapat menjalankan crawler sesuai permintaan atau menentukan jadwal berbasis waktu untuk crawler dan pekerjaan Anda di. AWS Glue Penentuan jadwal ini menggunakan sintaksis cron yang mirip UNIX. Untuk informasi selengkapnya, lihat Menjadwalkan sebuah Crawler AWS Glue.

Langkah 5: Tinjau dan buat

Tinjau pengaturan crawler yang Anda konfigurasikan, dan buat crawler.

Menambahkan pengklasifikasi ke crawler di AWS Glue

Sebuah pengklasifikasi membaca data di sebuah penyimpanan data. Jika ia mengakui format data, maka ia membuat skema. Pengklasifikasi juga mengembalikan nomor kepastian untuk menunjukkan seberapa pasti pengakuan format itu.

AWS Glue menyediakan serangkaian pengklasifikasi bawaan, tetapi Anda juga dapat membuat pengklasifikasi kustom. AWS Glue memanggil pengklasifikasi kustom terlebih dahulu, dalam urutan yang Anda tentukan dalam definisi crawler Anda. Tergantung pada hasil yang dikembalikan dari pengklasifikasi kustom, AWS Glue mungkin juga akan memanggil pengklasifikasi bawaan. Jika pengklasifikasi mengembalikan `certainty=1.0` selama pemrosesan, maka itu menunjukkan bahwa pengklasifikasi 100 persen yakin bahwa ia dapat membuat skema yang benar. AWS Glue kemudian menggunakan output dari pengklasifikasi itu.

Jika tidak, pengklasifikasi mengembalikan `certainty=1.0`, AWS Glue menggunakan output dari pengklasifikasi yang memiliki kepastian tertinggi. Jika tidak, pengklasifikasi mengembalikan kepastian lebih besar dari `0.0`, AWS Glue mengembalikan string klasifikasi default `UNKNOWN`.

Kapan saya menggunakan classifier?

Anda menggunakan pengklasifikasi ketika Anda melakukan crawling pada penyimpanan data untuk menentukan tabel metadata di AWS Glue Data Catalog. Anda dapat mengatur crawler dengan satu

set pengklasifikasi yang telah diurutkan. Ketika crawler memanggil pengklasifikasi, pengklasifikasi tersebut akan menentukan apakah data diakui. Jika pengklasifikasi tidak dapat mengenali data atau tidak 100 persen yakin, maka crawler memanggil pengklasifikasi berikutnya dalam daftar untuk menentukan apakah ia dapat mengenali data tersebut.

Untuk informasi lebih lanjut tentang membuat distribusi menggunakan konsol AWS Glue, lihat Bekerja dengan pengklasifikasi di konsol AWS Glue.

Pengklasifikasi khusus

Output dari pengklasifikasi termasuk string yang menunjukkan klasifikasi atau format file (misalnya, `json`) dan skema dari file tersebut. Untuk pengklasifikasi kustom, Anda menentukan logika untuk membuat skema berdasarkan jenis pengklasifikasi. Jenis pengklasifikasi termasuk menentukan skema berdasarkan pola grok, tag XML, dan path JSON.

Jika Anda mengubah sebuah definisi pengklasifikasi, maka data yang sebelumnya di-crawling menggunakan pengklasifikasi tersebut tidak akan direklasifikasi. Crawler mempertahankan jejak data yang telah di-crawling sebelumnya. Data baru diklasifikasikan dengan pengklasifikasi yang diperbarui, yang dapat menghasilkan skema diperbarui. Jika skema data Anda telah berkembang, perbarui pengklasifikasi untuk memperhitungkan perubahan skema apa pun saat crawler Anda berjalan. Untuk mengklasifikasi ulang data untuk mengoreksi pengklasifikasi yang salah, buat sebuah crawler baru dengan pengklasifikasi yang sudah diperbarui.

Untuk informasi selengkapnya tentang membuat pengklasifikasi di AWS Glue, lihat Menulis pengklasifikasi kustom.

> **Note**
>
> Jika format data Anda dikenali oleh salah satu pengklasifikasi bawaan, maka Anda tidak perlu membuat pengklasifikasi kustom.

Pengklasifikasi bawaan di AWS Glue

AWS Glue menyediakan pengklasifikasi bawaan untuk berbagai format, termasuk JSON, CSV, log web, dan banyak sistem basis data.

Jika AWS Glue tidak menemukan pengklasifikasi kustom yang sesuai dengan format input data dengan 100 persen kepastian, maka ia akan memanggil pengklasifikasi bawaan dalam urutan yang

ditunjukkan dalam tabel berikut. Pengklasifikasi bawaan mengembalikan hasil untuk menunjukkan apakah format cocok (`certainty=1.0`) atau tidak cocok (`certainty=0.0`). Pengklasifikasi pertama yang memiliki `certainty=1.0` menyediakan string klasifikasi dan skema untuk tabel metadata dalam Katalog Data Anda.

Jenis pengklasifikasi	String klasifikasi	Catatan
Apache Avro	`avro`	Membaca skema pada awal file untuk menentukan formatnya.
Apache ORC	`orc`	Membaca metadata file untuk menentukan formatnya.
Apache Parquet	`parquet`	Membaca skema pada akhir file untuk menentukan formatnya.
JSON	`json`	Membaca awal file untuk menentukan formatnya.
JSON biner	`bson`	Membaca awal file untuk menentukan formatnya.
XML	`xml`	Membaca awal file untuk menentukan format. AWS Glue menentukan skema tabel berdasarkan tag XML dalam dokumen. Untuk informasi tentang cara membuat pengklasifikasi XML kustom untuk menentukan baris dalam dokumen, lihat Menulis XHTML pengklasifikasi kustom.
Amazon Ion	`ion`	Membaca awal file untuk menentukan formatnya.
Log Apache gabungan	`combined_apache`	Menentukan format log melalui pola grok.
Apache log	`apache`	Menentukan format log melalui pola grok.
Log kernel Linux	`linux_kernel`	Menentukan format log melalui pola grok.
Log Microsoft	`microsoft_log`	Menentukan format log melalui pola grok.

Jenis pengklasifikasi	String klasifikasi	Catatan
Log Ruby	`ruby_logger`	Membaca awal file untuk menentukan formatnya.
Log Squid 3.x	`squid`	Membaca awal file untuk menentukan formatnya.
Log monitor Redis	`redismonlog`	Membaca awal file untuk menentukan formatnya.
Log Redis	`redislog`	Membaca awal file untuk menentukan formatnya.
CSV	`csv`	Cek untuk pembatas berikut: koma (,), pipa (|), tab (\ t), titik koma (;), dan Ctrl-A (\ u0001). Ctrl-A adalah karakter kontrol Unicode untuk `Start Of Heading`.
Amazon Redshift	`redshift`	Menggunakan koneksi JDBC untuk mengimpor metadata.
MySQL	`mysql`	Menggunakan koneksi JDBC untuk mengimpor metadata.
PostgreSQL	`postgresql`	Menggunakan koneksi JDBC untuk mengimpor metadata.
Basis data Oracle	`oracle`	Menggunakan koneksi JDBC untuk mengimpor metadata.
Microsoft SQL Server	`sqlserver`	Menggunakan koneksi JDBC untuk mengimpor metadata.
Amazon DynamoDB	`dynamodb`	Membaca data dari tabel DynamoDB.

File dalam format terkompresi berikut dapat diklasifikasikan:

- ZIP (didukung untuk arsip yang hanya berisi satu file). Perhatikan bahwa Zip tidak didukung dengan baik dalam layanan lain (karena arsip).
- BZIP

- GZIP
- LZ4
- Snappy (didukung untuk format Snappy asli standar dan Hadoop)

Pengklasifikasi CSV bawaan

Pengklasifikasi CSV bawaan menguraikan konten file CSV untuk menentukan skema untuk sebuah tabel AWS Glue. Pengklasifikasi ini memeriksa pembatas berikut:

- Koma (,)
- Pipa (|)
- Tab (\t)
- Titik koma (;)
- Ctrl-A (\u0001)

  Ctrl-A adalah karakter kontrol Unicode untuk `Start Of Heading`.

Untuk bisa diklasifikasikan sebagai CSV, skema tabel harus memiliki setidaknya dua kolom dan dua baris data. Pengklasifikasi CSV menggunakan sejumlah heuristik untuk menentukan apakah header ditemukan dalam file tertentu. Jika pengklasifikasi tidak dapat menentukan header dari baris pertama data, maka header kolom ditampilkan sebagai `col1`, `col2`, `col3`, dan sebagainya. Pengklasifikasi CSV bawaan menentukan apakah akan menyimpulkan header dengan menilai karakteristik file berikut:

- Setiap kolom di sebuah header potensial mem-parsing sebagai tipe data STRING.
- Kecuali untuk kolom terakhir, setiap kolom di header potensial memiliki konten yang kurang dari 150 karakter. Untuk memungkinkan untuk pembatas dibelakang, kolom terakhir dapat dibiarkan kosong di seluruh file.
- Setiap kolom di header potensial harus memenuhi persyaratan `regex` AWS Glue untuk nama kolom.
- Baris header harus cukup berbeda dari baris data. Untuk menentukan ini, satu atau lebih dari baris tersebut harus mengurai selain jenis STRING. Jika semua kolom adalah tipe STRING, maka baris pertama data tidak cukup berbeda dari baris berikutnya yang akan digunakan sebagai header.

> **Note**
>
> Jika pengklasifikasi CSV bawaan tidak membuat tabel AWS Glue seperti yang Anda inginkan, maka Anda mungkin dapat menggunakan salah satu alternatif berikut:
>
> - Mengubah nama kolom dalam Katalog Data, mengatur `SchemaChangePolicy` ke LOG, dan mengatur konfigurasi partisi output ke `InheritFromTable` untuk eksekusi crawler di masa depan.
> - Membuat sebuah pengklasifikasi grok kustom untuk mengurai data dan menetapkan kolom yang Anda inginkan.
> - Pengklasifikasi CSV bawaan membuat tabel yang me-referensi `LazySimpleSerDe` sebagai perpustakaan serialisasi, yang merupakan pilihan yang baik untuk inferensi tipe. Namun, jika data CSV berisi string yang dikutip, edit definisi tabel dan ubah pustaka menjadi SerDe . `OpenCSVSerDe` Sesuaikan setiap jenis yang disimpulkan menjadi STRING, atur `SchemaChangePolicy` ke LOG, dan atur konfigurasi partisi output menjadi `InheritFromTable` untuk eksekusi crawler di masa depan. Untuk informasi selengkapnya tentang SerDe perpustakaan, lihat SerDe Referensi di Panduan Pengguna Amazon Athena.

Menulis pengklasifikasi kustom

Anda dapat menyediakan pengklasifikasi kustom untuk mengklasifikasikan data Anda di AWS Glue. Anda dapat membuat pengklasifikasi kustom menggunakan pola grok, tag XHTML, JavaScript Object Notation (JSON), atau nilai yang dipisahkan koma (CSV). Sebuah crawler AWS Glue memanggil sebuah pengklasifikasi kustom. Jika pengklasifikasi tersebut mengakui data, maka ia akan mengembalikan klasifikasi dan skema data kepada crawler. Anda mungkin perlu menentukan pengklasifikasi kustom jika data Anda tidak cocok dengan pengklasifikasi bawaan, atau jika Anda ingin menyesuaikan tabel yang dibuat oleh crawler tersebut.

Untuk informasi lebih lanjut tentang membuat distribusi menggunakan konsol AWS Glue, lihat Bekerja dengan pengklasifikasi di konsol AWS Glue.

AWS Glue menjalankan pengklasifikasi tersuai sebelum pengklasifikasi bawaan, dalam susunan yang anda tentukan. Ketika sebuah crawler menemukan pengklasifikasi yang cocok dengan data, string klasifikasi dan skema digunakan dalam definisi tabel yang ditulis ke AWS Glue Data Catalog.

Topik

Menulis pengklasifikasi kustom grok

Grok adalah alat yang digunakan untuk mengurai data tekstual mengingat pola yang cocok. Pola grok adalah seperangkat ekspresi reguler (regex) yang diberi nama yang digunakan untuk mencocokkan data satu baris pada suatu waktu. AWS Glue menggunakan pola grok untuk menyimpulkan skema data Anda. Ketika pola grok cocok dengan data Anda, AWS Glue akan menggunakan pola tersebut untuk menentukan struktur data Anda dan memetakan ke dalam bidang.

AWS Glue menyediakan banyak pola bawaan, atau Anda dapat menentukannya sendiri. Anda dapat membuat pola grok dengan menggunakan pola bawaan dan pola kustom dalam definisi pengklasifikasi kustom Anda. Anda dapat menyesuaikan pola grok untuk mengklasifikasikan format file teks kustom.

> **Note**
>
> Pengklasifikasi kustom grok AWS Glue menggunakan pustaka serialisasi `GrokSerDe` untuk tabel yang dibuat di AWS Glue Data Catalog. Jika Anda menggunakan AWS Glue Data Catalog dengan Amazon Athena, Amazon EMR, atau Redshift Spectrum, periksa dokumentasi tentang layanan tersebut untuk informasi tentang support `GrokSerDe`. Saat ini, Anda mungkin mengalami masalah dalam meng-kueri tabel yang dibuat dengan `GrokSerDe` dari Amazon EMR dan Redshift Spectrum.

Berikut ini adalah sintaksis dasar untuk komponen pola grok:

```
%{PATTERN:field-name}
```

Data yang cocok dengan nama `PATTERN` dipetakan ke kolom `field-name` dalam skema, dengan tipe data default `string`. Opsional, tipe data untuk bidang dapat diberikan ke `byte`, `boolean`, `double`, `short`, `int`, `long`, atau `float` dalam skema yang dihasilkan.

```
%{PATTERN:field-name:data-type}
```

Misalnya, untuk memberikan bidang `num` ke tipe data `int`, Anda dapat menggunakan pola ini:

```
%{NUMBER:num:int}
```

Pola dapat terdiri dari pola lainnya. Misalnya, Anda dapat memiliki sebuah pola untuk stempel waktu `SYSLOG` yang ditentukan berdasarkan pola untuk bulan, tanggal, dan waktu (misalnya, `Feb 1 06:25:43`). Untuk data ini, Anda dapat menentukan pola berikut:

```
SYSLOGTIMESTAMP %{MONTH} +%{MONTHDAY} %{TIME}
```

> **Note**
>
> Pola grok dapat memproses hanya satu baris pada satu waktu. Pola multi-baris tidak didukung. Selain itu, jeda baris dalam pola tidak didukung.

Nilai pengklasifikasi kustom di AWS Glue

Ketika Anda menentukan pengklasifikasi grok, Anda menyediakan nilai-nilai berikut untuk AWS Glue untuk membuat pengklasifikasi kustom.

Nama

Nama pengklasifikasi.

Klasifikasi

String teks yang ditulis untuk menjelaskan format data yang diklasifikasikan; misalnya, `special-logs`.

Pola Grok

Serangkaian pola yang diterapkan ke penyimpanan data untuk menentukan apakah ada kecocokan. Pola-pola ini berasal dari pola bawaan AWS Glue dan pola kustom apa pun yang Anda tetapkan.

Berikut ini adalah contoh pola grok:

```
%{TIMESTAMP_ISO8601:timestamp} \[%{MESSAGEPREFIX:message_prefix}\]
 %{CRAWLERLOGLEVEL:loglevel} : %{GREEDYDATA:message}
```

Ketika data cocok dengan `TIMESTAMP_ISO8601`, kolom skema `timestamp` dibuat. Perilaku ini mirip untuk pola lain yang diberi nama dalam contoh.

Pola kustom

Pola kustom opsional yang Anda tetapkan. Pola-pola ini direferensikan oleh pola grok yang mengklasifikasikan data Anda. Anda dapat me-referensi pola kustom ini dalam pola grok yang diterapkan ke data Anda. Setiap pola komponen kustom harus ada pada baris terpisah. Sinstaksis ekspresi reguler (regex) digunakan untuk menentukan pola.

Berikut ini adalah contoh penggunaan pola kustom:

```
CRAWLERLOGLEVEL (BENCHMARK|ERROR|WARN|INFO|TRACE)
MESSAGEPREFIX .*-.*-.*-.*-.*
```

Pola bernama kustom yang pertama, `CRAWLERLOGLEVEL`, adalah kecocokan ketika data cocok dengan salah satu string yang disebutkan satu per satu. Pola kustom kedua, `MESSAGEPREFIX`, mencoba mencocokkan string prefiks pesan.

AWS Glue menyimpan jejak waktu pembuatan, waktu update terakhir, dan versi pengklasifikasi Anda.

AWS Gluepola bawaan

AWS Glue menyediakan banyak pola umum yang dapat Anda gunakan untuk membangun sebuah pengklasifikasi kustom. Anda menambahkan sebuah pola bernama ke `grok pattern` dalam sebuah definisi pengklasifikasi.

Daftar berikut terdiri dari sebuah baris untuk setiap pola. Di setiap baris, nama pola diikuti definisinya. Sinstaksis ekspresi reguler (regex) digunakan untuk menentukan pola.

```
#<noloc>&GLU;</noloc> Built-in patterns
 USERNAME [a-zA-Z0-9._-]+
 USER %{USERNAME:UNWANTED}
 INT (?:[+-]?(?:[0-9]+))
 BASE10NUM (?<![0-9.+-])(?>[+-]?(?:(?:[0-9]+(?:\.[0-9]+)?)|(?:\.[0-9]+)))
 NUMBER (?:%{BASE10NUM:UNWANTED})
 BASE16NUM (?<![0-9A-Fa-f])(?:[+-]?(?:0x)?(?:[0-9A-Fa-f]+))
 BASE16FLOAT \b(?<![0-9A-Fa-f.])(?:[+-]?(?:0x)?(?:(?:[0-9A-Fa-f]+(?:\.[0-9A-Fa-f]*)?)|
(?:\.[0-9A-Fa-f]+)))\b
 BOOLEAN (?i)(true|false)
```

```
 POSINT \b(?:[1-9][0-9]*)\b
 NONNEGINT \b(?:[0-9]+)\b
 WORD \b\w+\b
 NOTSPACE \S+
 SPACE \s*
 DATA .*?
 GREEDYDATA .*
 #QUOTEDSTRING (?:(?<!\\)(?:"(?:\\.|[^\\"])*"|(?:'(?:\\.|[^\\'])*')|(?:`(?:\\.|[^\
\`])*`)))
 QUOTEDSTRING (?>(?<!\\)(?>"(?>\\.|[^\\"]+)+"|""|(?>'(?>\\.|[^\\']+)+')|''|(?>`(?>\\.|
[^\\`]+)+`)|``))
 UUID [A-Fa-f0-9]{8}-(?:[A-Fa-f0-9]{4}-){3}[A-Fa-f0-9]{12}

 # Networking
 MAC (?:%{CISCOMAC:UNWANTED}|%{WINDOWSMAC:UNWANTED}|%{COMMONMAC:UNWANTED})
 CISCOMAC (?:(?:[A-Fa-f0-9]{4}\.){2}[A-Fa-f0-9]{4})
 WINDOWSMAC (?:(?:[A-Fa-f0-9]{2}-){5}[A-Fa-f0-9]{2})
 COMMONMAC (?:(?:[A-Fa-f0-9]{2}:){5}[A-Fa-f0-9]{2})
 IPV6 ((([0-9A-Fa-f]{1,4}:){7}([0-9A-Fa-f]{1,4}|:))|(([0-9A-Fa-f]{1,4}:){6}(:[0-9A-
Fa-f]{1,4}|((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
{3})|:))|(([0-9A-Fa-f]{1,4}:){5}(((:[0-9A-Fa-f]{1,4}){1,2})|:((25[0-5]|2[0-4]\d|1\d
\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3})|:))|(([0-9A-Fa-f]{1,4}:){4}(((:
[0-9A-Fa-f]{1,4}){1,3})|((:[0-9A-Fa-f]{1,4})?:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.
(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){3}(((:[0-9A-Fa-f]
{1,4}){1,4})|((:[0-9A-Fa-f]{1,4}){0,2}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|
2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){2}(((:[0-9A-Fa-f]{1,4}){1,5})|
((:[0-9A-Fa-f]{1,4}){0,3}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d
\d|[1-9]?\d)){3}))|:))|(([0-9A-Fa-f]{1,4}:){1}(((:[0-9A-Fa-f]{1,4}){1,6})|((:[0-9A-Fa-
f]{1,4}){0,4}:((25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d))
{3}))|:))|(:(((:[0-9A-Fa-f]{1,4}){1,7})|((:[0-9A-Fa-f]{1,4}){0,5}:((25[0-5]|2[0-4]\d|
1\d\d|[1-9]?\d)(\.(25[0-5]|2[0-4]\d|1\d\d|[1-9]?\d)){3}))|:)))(%.+)?
 IPV4 (?<![0-9])(?:(?:25[0-5]|2[0-4][0-9]|[0-1]?[0-9]{1,2})[.](?:25[0-5]|2[0-4][0-9]|
[0-1]?[0-9]{1,2})[.](?:25[0-5]|2[0-4][0-9]|[0-1]?[0-9]{1,2})[.](?:25[0-5]|2[0-4][0-9]|
[0-1]?[0-9]{1,2}))(?![0-9])
 IP (?:%{IPV6:UNWANTED}|%{IPV4:UNWANTED})
 HOSTNAME \b(?:[0-9A-Za-z][0-9A-Za-z-_]{0,62})(?:\.(?:[0-9A-Za-z][0-9A-Za-z-_]
{0,62}))*(\.?|\b)
 HOST %{HOSTNAME:UNWANTED}
 IPORHOST (?:%{HOSTNAME:UNWANTED}|%{IP:UNWANTED})
 HOSTPORT (?:%{IPORHOST}:%{POSINT:PORT})

 # paths
 PATH (?:%{UNIXPATH}|%{WINPATH})
 UNIXPATH (?>/(?>[\w_%!$@:.,~-]+|\\.)*)+
```

```
 #UNIXPATH (?<![\w\/])(?:/[^\/\s?*]*)+
 TTY (?:/dev/(pts|tty([pq])?)(\w+)?/?(?:[0-9]+))
 WINPATH (?>[A-Za-z]+:|\\)(?:\\[^\\?*]*)+
 URIPROTO [A-Za-z]+(\+[A-Za-z+]+)?
 URIHOST %{IPORHOST}(?::%{POSINT:port})?
 # uripath comes loosely from RFC1738, but mostly from what Firefox
 # doesn't turn into %XX
 URIPATH (?:/[A-Za-z0-9$.+!*'(){},~:;=@#%_\-]*)+
 #URIPARAM \?(?:[A-Za-z0-9]+(?:=(?:[^&]*))?(?:&(?:[A-Za-z0-9]+(?:=(?:[^&]*))?)?)*)?
 URIPARAM \?[A-Za-z0-9$.+!*'|(){},~@#%&/=:;_?\-\[\]]*
 URIPATHPARAM %{URIPATH}(?:%{URIPARAM})?
 URI %{URIPROTO}://(?:%{USER}(?::[^@]*)?@)?(?:%{URIHOST})?(?:%{URIPATHPARAM})?

 # Months: January, Feb, 3, 03, 12, December
 MONTH \b(?:Jan(?:uary)?|Feb(?:ruary)?|Mar(?:ch)?|Apr(?:il)?|May|Jun(?:e)?|Jul(?:y)?|
Aug(?:ust)?|Sep(?:tember)?|Oct(?:ober)?|Nov(?:ember)?|Dec(?:ember)?)\b
 MONTHNUM (?:0?[1-9]|1[0-2])
 MONTHNUM2 (?:0[1-9]|1[0-2])
 MONTHDAY (?:(?:0[1-9])|(?:[12][0-9])|(?:3[01])|[1-9])

 # Days: Monday, Tue, Thu, etc...
 DAY (?:Mon(?:day)?|Tue(?:sday)?|Wed(?:nesday)?|Thu(?:rsday)?|Fri(?:day)?|
Sat(?:urday)?|Sun(?:day)?)

 # Years?
 YEAR (?>\d\d){1,2}
 # Time: HH:MM:SS
 #TIME \d{2}:\d{2}(?::\d{2}(?:\.\d+)?)?
 # TIME %{POSINT<24}:%{POSINT<60}(?::%{POSINT<60}(?:\.%{POSINT})?)?
 HOUR (?:2[0123]|[01]?[0-9])
 MINUTE (?:[0-5][0-9])
 # '60' is a leap second in most time standards and thus is valid.
 SECOND (?:(?:[0-5]?[0-9]|60)(?:[:.,][0-9]+)?)
 TIME (?!<[0-9])%{HOUR}:%{MINUTE}(?::%{SECOND})(?![0-9])
 # datestamp is YYYY/MM/DD-HH:MM:SS.UUUU (or something like it)
 DATE_US %{MONTHNUM}[/-]%{MONTHDAY}[/-]%{YEAR}
 DATE_EU %{MONTHDAY}[./-]%{MONTHNUM}[./-]%{YEAR}
 DATESTAMP_US %{DATE_US}[- ]%{TIME}
 DATESTAMP_EU %{DATE_EU}[- ]%{TIME}
 ISO8601_TIMEZONE (?:Z|[+-]%{HOUR}(?::?%{MINUTE}))
 ISO8601_SECOND (?:%{SECOND}|60)
 TIMESTAMP_ISO8601 %{YEAR}-%{MONTHNUM}-%{MONTHDAY}[T ]%{HOUR}:?%{MINUTE}(?::?
%{SECOND})?%{ISO8601_TIMEZONE}?
 TZ (?:[PMCE][SD]T|UTC)
```

```
 DATESTAMP_RFC822 %{DAY} %{MONTH} %{MONTHDAY} %{YEAR} %{TIME} %{TZ}
 DATESTAMP_RFC2822 %{DAY}, %{MONTHDAY} %{MONTH} %{YEAR} %{TIME} %{ISO8601_TIMEZONE}
 DATESTAMP_OTHER %{DAY} %{MONTH} %{MONTHDAY} %{TIME} %{TZ} %{YEAR}
 DATESTAMP_EVENTLOG %{YEAR}%{MONTHNUM2}%{MONTHDAY}%{HOUR}%{MINUTE}%{SECOND}
 CISCOTIMESTAMP %{MONTH} %{MONTHDAY} %{TIME}

 # Syslog Dates: Month Day HH:MM:SS
 SYSLOGTIMESTAMP %{MONTH} +%{MONTHDAY} %{TIME}
 PROG (?:[\w._/%-]+)
 SYSLOGPROG %{PROG:program}(?:\[%{POSINT:pid}\])?
 SYSLOGHOST %{IPORHOST}
 SYSLOGFACILITY <%{NONNEGINT:facility}.%{NONNEGINT:priority}>
 HTTPDATE %{MONTHDAY}/%{MONTH}/%{YEAR}:%{TIME} %{INT}

 # Shortcuts
 QS %{QUOTEDSTRING:UNWANTED}

 # Log formats
 SYSLOGBASE %{SYSLOGTIMESTAMP:timestamp} (?:%{SYSLOGFACILITY} )?%{SYSLOGHOST:logsource}
 %{SYSLOGPROG}:

 MESSAGESLOG %{SYSLOGBASE} %{DATA}

 COMMONAPACHELOG %{IPORHOST:clientip} %{USER:ident} %{USER:auth}
 \[%{HTTPDATE:timestamp}\] "(?:%{WORD:verb} %{NOTSPACE:request}(?: HTTP/
%{NUMBER:httpversion})?|%{DATA:rawrequest})" %{NUMBER:response} (?:%{Bytes:bytes=
%{NUMBER}|-})
 COMBINEDAPACHELOG %{COMMONAPACHELOG} %{QS:referrer} %{QS:agent}
 COMMONAPACHELOG_DATATYPED %{IPORHOST:clientip} %{USER:ident;boolean} %{USER:auth}
 \[%{HTTPDATE:timestamp;date;dd/MMM/yyyy:HH:mm:ss Z}\] "(?:%{WORD:verb;string}
 %{NOTSPACE:request}(?: HTTP/%{NUMBER:httpversion;float})?|%{DATA:rawrequest})"
 %{NUMBER:response;int} (?:%{NUMBER:bytes;long}|-)


 # Log Levels
 LOGLEVEL ([A|a]lert|ALERT|[T|t]race|TRACE|[D|d]ebug|DEBUG|[N|n]otice|NOTICE|[I|i]nfo|
INFO|[W|w]arn?(?:ing)?|WARN?(?:ING)?|[E|e]rr?(?:or)?|ERR?(?:OR)?|[C|c]rit?(?:ical)?|
CRIT?(?:ICAL)?|[F|f]atal|FATAL|[S|s]evere|SEVERE|EMERG(?:ENCY)?|[Ee]merg(?:ency)?)
```

Menulis XHTML pengklasifikasi kustom

XML mendefinisikan struktur sebuah dokumen dengan penggunaan tag dalam file. Dengan pengklasifikasi kustom XML, Anda dapat menentukan nama tag yang digunakan untuk menentukan sebuah baris.

Nilai pengklasifikasi kustom di AWS Glue

Ketika Anda menentukan pengklasifikasi XML, Anda menyediakan nilai-nilai berikut untuk AWS Glue untuk membuat pengklasifikasi. Bidang klasifikasi dari pengklasifikasi ini diatur ke `xml`.

Nama

Nama pengklasifikasi.

Tag baris

Nama tag XML yang mendefinisikan baris tabel dalam dokumen XML, tanpa kurung sudut `< >`. Nama harus sesuai dengan aturan tag dalam XML.

> **Note**
>
> Elemen yang berisi data baris tidak dapat menjadi elemen kosong menutup-sendiri. Misalnya, elemen kosong ini tidak di-parsing oleh AWS Glue:
>
> ```
> <row att1=”xx” att2=”yy” />
> ```
>
> Elemen kosong dapat ditulis sebagai berikut:
>
> ```
> <row att1=”xx” att2=”yy”> </row>
> ```

AWS Glue menyimpan jejak waktu pembuatan, waktu update terakhir, dan versi pengklasifikasi Anda.

Sebagai contoh, anggap bahwa Anda memiliki file XML berikut. Untuk membuat tabel AWS Glue yang hanya berisi kolom untuk penulis dan judul, buat sebuah pengklasifikasi di konsol AWS Glue

dengan Tag baris sebagai `AnyCompany`. Kemudian tambahkan dan jalankan sebuah crawler yang menggunakan pengklasifikasi kustom ini.

```
<?xml version="1.0"?>
<catalog>
   <book id="bk101">
     <AnyCompany>
       <author>Rivera, Martha</author>
       <title>AnyCompany Developer Guide</title>
     </AnyCompany>
   </book>
   <book id="bk102">
     <AnyCompany>
       <author>Stiles, John</author>
       <title>Style Guide for AnyCompany</title>
     </AnyCompany>
   </book>
</catalog>
```

Menulis pengklasifikasi kustom JSON

JSON adalah sebuah format saling-mengganti-data. Ia mendefinisikan struktur data dengan pasangan nama-nilai atau daftar nilai-nilai yang diurutkan. Dengan pengklasifikasi kustom JSON, Anda dapat menentukan path JSON untuk sebuah struktur data yang digunakan untuk menentukan skema untuk tabel Anda.

Nilai pengklasifikasi kustom di AWS Glue

Ketika Anda menentukan pengklasifikasi JSON, Anda menyediakan nilai-nilai berikut untuk AWS Glue untuk membuat pengklasifikasi. Bidang klasifikasi dari pengklasifikasi ini diatur ke `json`.

Nama

Nama pengklasifikasi.

Path JSON

Sebuah path JSON yang mengarahkan ke sebuah objek yang digunakan untuk mendefinisikan skema tabel. Path JSON dapat ditulis dalam notasi dot atau notasi kurung. Operator berikut ini didukung:

Deskripsi
Elemen akar dari sebuah objek JSON. Ia mengawali semua ekspresi path
Karakter wildcard. Sediakan di mana saja nama atau numerik yang diperlukan di path JSON.
Anak dinotasi-titik. Menentukan bidang anak dalam sebuah objek JSON.
Anak dinotasi-kurung. Menentukan bidang anak dalam sebuah objek JSON. Hanya satu bidang anak saja yang dapat ditentukan.
Indeks array. Menentukan nilai array berdasarkan indeks.

AWS Glue menyimpan jejak waktu pembuatan, waktu update terakhir, dan versi pengklasifikasi Anda.

Example Menggunakan pengklasifikasi JSON untuk menarik catatan dari array

Misalkan data JSON Anda adalah sebuah array catatan. Sebagai contoh, beberapa baris pertama dari file Anda mungkin terlihat seperti berikut ini:

```
[
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ak",
    "name": "Alaska"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:1",
    "name": "Alabama's 1st congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:2",
    "name": "Alabama's 2nd congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:3",
```

```
    "name": "Alabama's 3rd congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:4",
    "name": "Alabama's 4th congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:5",
    "name": "Alabama's 5th congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:6",
    "name": "Alabama's 6th congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:al\/cd:7",
    "name": "Alabama's 7th congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ar\/cd:1",
    "name": "Arkansas's 1st congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ar\/cd:2",
    "name": "Arkansas's 2nd congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ar\/cd:3",
    "name": "Arkansas's 3rd congressional district"
  },
  {
    "type": "constituency",
    "id": "ocd-division\/country:us\/state:ar\/cd:4",
    "name": "Arkansas's 4th congressional district"
  }
]
```

Saat Anda menjalankan sebuah crawler menggunakan pengklasifikasi JSON bawaan, seluruh file digunakan untuk menentukan skema. Karena Anda tidak menentukan path JSON, crawler memperlakukan data sebagai satu objek, yaitu, hanya sebuah array. Sebagai contoh, skema tersebut mungkin terlihat seperti berikut ini:

```
root
|-- record: array
```

Namun demikian, untuk membuat skema yang didasarkan pada setiap catatan dalam array JSON, buatlah sebuah pengklasifikasi JSON kustom dan tentukan path JSON sebagai `$[*]`. Bila Anda menentukan path JSON ini, pengklasifikasi akan mengambil data dari semua 12 catatan dalam array untuk menentukan skema. Skema yang dihasilkan berisi bidang terpisah untuk setiap objek, mirip dengan contoh berikut ini:

```
root
|-- type: string
|-- id: string
|-- name: string
```

Example Menggunakan pengklasifikasi JSON untuk memeriksa hanya bagian dari file

Misalkan data JSON Anda mengikuti pola file contoh JSON `s3://awsglue-datasets/examples/us-legislators/all/areas.json` yang diambil dari http://everypolitician.org/. Objek contoh dalam file JSON terlihat seperti berikut ini:

```
{
  "type": "constituency",
  "id": "ocd-division\/country:us\/state:ak",
  "name": "Alaska"
}
{
  "type": "constituency",
  "identifiers": [
    {
      "scheme": "dmoz",
      "identifier": "Regional\/North_America\/United_States\/Alaska\/"
    },
    {
```

```
      "scheme": "freebase",
      "identifier": "\/m\/0hjy"
    },
    {
      "scheme": "fips",
      "identifier": "US02"
    },
    {
      "scheme": "quora",
      "identifier": "Alaska-state"
    },
    {
      "scheme": "britannica",
      "identifier": "place\/Alaska"
    },
    {
      "scheme": "wikidata",
      "identifier": "Q797"
    }
  ],
  "other_names": [
    {
      "lang": "en",
      "note": "multilingual",
      "name": "Alaska"
    },
    {
      "lang": "fr",
      "note": "multilingual",
      "name": "Alaska"
    },
    {
      "lang": "nov",
      "note": "multilingual",
      "name": "Alaska"
    }
  ],
  "id": "ocd-division\/country:us\/state:ak",
  "name": "Alaska"
}
```

Saat Anda menjalankan sebuah crawler menggunakan pengklasifikasi JSON bawaan, seluruh file digunakan untuk membuat skema. Anda mungkin mendapatkan skema seperti ini:

```
root
|-- type: string
|-- id: string
|-- name: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
```

Namun demikian, untuk membuat sebuah skema hanya menggunakan objek "`id`", buatlah sebuah pengklasifikasi JSON kustom dan tentukan path JSON sebagai `$.id`. Kemudian skema tersebut didasarkan pada bidang "`id`" saja:

```
root
|-- record: string
```

Beberapa baris pertama data yang diekstraksi dengan skema ini akan terlihat seperti ini:

```
{"record": "ocd-division/country:us/state:ak"}
{"record": "ocd-division/country:us/state:al/cd:1"}
{"record": "ocd-division/country:us/state:al/cd:2"}
{"record": "ocd-division/country:us/state:al/cd:3"}
{"record": "ocd-division/country:us/state:al/cd:4"}
{"record": "ocd-division/country:us/state:al/cd:5"}
{"record": "ocd-division/country:us/state:al/cd:6"}
{"record": "ocd-division/country:us/state:al/cd:7"}
{"record": "ocd-division/country:us/state:ar/cd:1"}
{"record": "ocd-division/country:us/state:ar/cd:2"}
{"record": "ocd-division/country:us/state:ar/cd:3"}
{"record": "ocd-division/country:us/state:ar/cd:4"}
{"record": "ocd-division/country:us/state:as"}
```

```
{"record": "ocd-division/country:us/state:az/cd:1"}
{"record": "ocd-division/country:us/state:az/cd:2"}
{"record": "ocd-division/country:us/state:az/cd:3"}
{"record": "ocd-division/country:us/state:az/cd:4"}
{"record": "ocd-division/country:us/state:az/cd:5"}
{"record": "ocd-division/country:us/state:az/cd:6"}
{"record": "ocd-division/country:us/state:az/cd:7"}
```

Untuk menciptakan sebuah skema berdasarkan objek yang benar-benar di-nest, seperti "`identifier`," dalam file JSON, Anda dapat membuat pengklasifikasi JSON kustom dan menentukan path JSON sebagai `$.identifiers[*].identifier`. Meskipun skema ini mirip dengan contoh sebelumnya, tetapi ia didasarkan pada objek yang berbeda dalam file JSON tersebut.

Skema ini terlihat seperti berikut:

```
root
|-- record: string
```

Mencantumkan beberapa baris pertama data dari tabel menunjukkan bahwa skema didasarkan pada data dalam objek "`identifier`":

```
{"record": "Regional/North_America/United_States/Alaska/"}
{"record": "/m/0hjy"}
{"record": "US02"}
{"record": "5879092"}
{"record": "4001016-8"}
{"record": "destination/alaska"}
{"record": "1116270"}
{"record": "139487266"}
{"record": "n79018447"}
{"record": "01490999-8dec-4129-8254-eef6e80fadc3"}
{"record": "Alaska-state"}
{"record": "place/Alaska"}
{"record": "Q797"}
{"record": "Regional/North_America/United_States/Alabama/"}
{"record": "/m/0gyh"}
{"record": "US01"}
{"record": "4829764"}
{"record": "4084839-5"}
{"record": "161950"}
```

```
{"record": "131885589"}
```

Untuk menciptakan sebuah tabel berdasarkan objek yang benar-benar di-nest lainnya, seperti bidang "name" dalam array "other_names" dalam file JSON, Anda dapat membuat pengklasifikasi JSON kustom dan menentukan path JSON sebagai `$.other_names[*].name`. Meskipun skema ini mirip dengan contoh sebelumnya, tetapi ia didasarkan pada objek yang berbeda dalam file JSON tersebut. Skema ini terlihat seperti berikut:

```
root
|-- record: string
```

Mencantumkan beberapa baris pertama data dalam tabel menunjukkan bahwa ia didasarkan pada data dalam objek "name" dalam array "other_names":

```
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Аляска"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "######"}
{"record": "######"}
{"record": "######"}
{"record": "Alaska"}
{"record": "Alyaska"}
{"record": "Alaska"}
{"record": "Alaska"}
{"record": "Штат Аляска"}
{"record": "Аляска"}
{"record": "Alaska"}
{"record": "#######"}
```

Menulis pengklasifikasi khusus CSV

Pengklasifikasi CSV khusus memungkinkan Anda menentukan tipe data untuk setiap kolom di bidang pengklasifikasi csv khusus. Anda dapat menentukan tipe data setiap kolom yang dipisahkan dengan

koma. Dengan menentukan tipe data, Anda dapat mengganti tipe data yang disimpulkan crawler dan memastikan data akan diklasifikasikan dengan tepat.

Anda dapat mengatur SerDe untuk memproses CSV di pengklasifikasi, yang akan diterapkan di Katalog Data.

Saat membuat pengklasifikasi kustom, Anda juga dapat menggunakan kembali classifer untuk crawler yang berbeda.

- Untuk file csv dengan hanya header (tanpa data), file-file ini akan diklasifikasikan sebagai UNKNOWN karena tidak cukup informasi yang disediakan. Jika Anda menentukan bahwa CSV 'Memiliki judul' di opsi judul Kolom, dan memberikan tipe data, kami dapat mengklasifikasikan file-file ini dengan benar.

Anda dapat menggunakan pengklasifikasi CSV kustom untuk menyimpulkan skema dari berbagai jenis data CSV. Atribut kustom yang dapat Anda berikan untuk pengklasifikasi Anda termasuk pembatas, opsi CSV, SerDe opsi tentang header, dan apakah akan melakukan validasi tertentu pada data.

Nilai pengklasifikasi kustom di AWS Glue

Ketika Anda menentukan pengklasifikasi CSV, Anda menyediakan nilai-nilai berikut untuk AWS Glue untuk membuat pengklasifikasi. Bidang klasifikasi dari pengklasifikasi ini diatur ke `csv`.

Nama pengklasifikasi

Nama pengklasifikasi.

CSV Serde

Menetapkan SerDe untuk memproses CSV di classifier, yang akan diterapkan dalam Katalog Data. Pilihannya adalah Open CSV SerDe, Lazy Simple SerDe, dan None. Anda dapat menentukan nilai None saat Anda ingin crawler melakukan deteksi.

Kolom pembatas

Sebuah simbol kustom untuk menunjukkan apa yang memisahkan masing-masing entri kolom pada baris. Berikan karakter unicode. Jika Anda tidak dapat mengetik pembatas Anda, Anda dapat menyalin dan menempelkannya. Ini berfungsi untuk karakter yang dapat dicetak, termasuk karakter yang tidak didukung oleh sistem Anda (biasanya ditampilkan sebagai □).

Simbol kutipan

Sebuah simbol kustom untuk menunjukkan apa yang menggabungkan konten ke dalam satu nilai kolom tunggal. Harus berbeda dari pembatas kolom. Berikan karakter unicode. Jika Anda tidak dapat mengetik pembatas Anda, Anda dapat menyalin dan menempelkannya. Ini berfungsi untuk karakter yang dapat dicetak, termasuk karakter yang tidak didukung oleh sistem Anda (biasanya ditampilkan sebagai □).

Judul kolom

Menunjukkan perilaku untuk bagaimana mendeteksi judul kolom dalam file CSV. Jika file CSV kustom Anda memiliki judul kolom, masukkan daftar pembatas koma judul kolom.

Pilihan pemrosesan: Izinkan file dengan kolom tunggal

Memungkinkan pemrosesan file yang hanya berisi satu kolom saja.

Pilihan pemrosesan: Buang spasi kosong sebelum mengidentifikasi nilai kolom

Menentukan apakah akan memotong nilai sebelum mengidentifikasi jenis nilai kolom.

Tipe data khusus - opsional

Masukkan tipe data khusus yang dipisahkan dengan koma. Menentukan tipe data kustom dalam file CSV. Jenis data khusus harus berupa tipe data yang didukung. Jenis data yang didukung adalah: “BINARY”, “BOOLEAN”, “DATE”, “DECIMAL”, “DOUBLE”, “FLOAT”, “INT”, “LONG”, “SHORT”, “STRING”, “TIMESTAMP”. Tipe data yang tidak didukung akan menampilkan kesalahan.

Bekerja dengan pengklasifikasi di konsol AWS Glue

Sebuah pengklasifikasi menentukan skema dari data Anda. Anda dapat menulis sebuah pengklasifikasi kustom dan mengarahkannya dari AWS Glue.

Melihat pengklasifikasi

Untuk melihat daftar semua pengklasifikasi yang telah Anda buat, buka konsol AWS Glue di https://console.aws.amazon.com/glue/, dan pilih tab Pengklasifikasi.

Daftar tersebut menampilkan properti berikut tentang masing-masing pengklasifikasi:

- Pengklasifikasi — Nama pengklasifikasi. Saat Anda menciptakan pengklasifikasi, Anda harus menyediakan nama untuknya.
- Klasifikasi — Jenis klasifikasi tabel yang disimpulkan oleh pengklasifikasi ini.

- Terakhir diperbarui — Terakhir kali pengklasifikasi ini diperbarui.

Mengelola pengklasifikasi

Dari daftar Pengklasifikasi di konsol AWS Glue tersebut, Anda dapat menambahkan, mengedit, dan menghapus pengklasifikasi. Untuk melihat detail lebih lanjut untuk pengklasifikasi, pilih nama pengklasifikasi dalam daftar itu. Detail mencakup informasi yang Anda tetapkan saat Anda membuat pengklasifikasi.

Membuat pengklasifikasi

Untuk menambahkan sebuah pengklasifikasi di konsol AWS Glue, pilih Tambahkan pengklasifikasi. Ketika Anda menentukan sebuah pengklasifikasi, Anda menyediakan nilai-nilai untuk hal berikut:

- Nama pengklasifikasi — Berikan nama yang unik untuk pengklasifikasi Anda.
- Jenis pengklasifikasi — Jenis klasifikasi tabel yang disimpulkan oleh pengklasifikasi ini.
- Terakhir diperbarui — Terakhir kali pengklasifikasi ini diperbarui.

Nama pengklasifikasi

Berikan nama yang unik untuk pengklasifikasi Anda.

Jenis pengklasifikasi

Pilih jenis pengklasifikasi yang akan dibuat.

Tergantung pada jenis pengklasifikasi yang Anda pilih, konfigurasikan properti berikut untuk pengklasifikasi Anda:

Grok

- Klasifikasi

  Deskripsikan format atau jenis data yang diklasifikasikan atau berikan label kustom.

- Pola grok

  Hal ini digunakan untuk mengurai data Anda ke dalam skema terstruktur. Pola grok terdiri dari pola bernama yang mendeskripsikan format penyimpanan data Anda. Anda menulis pola grok ini dengan menggunakan pola bawaan bernama yang disediakan oleh AWS Glue dan pola

kustom yang Anda tulis dan sertakan dalam bidang Pola kustom. Meskipun hasil debugger grok mungkin tidak cocok dengan hasil dari AWS Glue secara persis, kami sarankan bahwa Anda mencoba pola Anda dengan menggunakan beberapa data sampel dengan debugger grok. Anda dapat menemukan debugger grok di web. Pola bawaan bernama disediakan oleh AWS Glue pada umumnya kompatibel dengan pola grok yang tersedia di web.

Membangun pola grok Anda dengan menambahkan pola bernama dan memeriksa hasil Anda dalam debugger secara berulang-ulang. Kegiatan ini akan memberikan Anda keyakinan bahwa ketika crawler AWS Glue menjalankan pola grok Anda, data Anda dapat di-parsing.

- Pola kustom

  Untuk pengklasifikasi grok, ini adalah blok bangunan opsional untuk Pola Grok yang Anda tulis. Ketika pola bawaan tidak dapat mengurai data Anda, maka Anda mungkin perlu menulis sebuah pola kustom. Pola kustom ini didefinisikan dalam bidang ini dan direferensikan dalam di bidang Pola Grok. Masing-masing pola kustom didefinisikan pada baris terpisah. Sama seperti pola bawaan, pola itu terdiri dari definisi pola bernama yang menggunakan sintaksis ekspresi reguler (regex).

  Sebagai contoh, berikut ini memiliki nama `MESSAGEPREFIX` yang diikuti dengan definisi ekspresi reguler untuk diterapkan ke data Anda untuk menentukan apakah itu mengikuti pola atau tidak.

  ```
  MESSAGEPREFIX .*-.*-.*-.*-.*
  ```

XML

- Tag baris

  Untuk pengklasifikasi XML, hal ini adalah nama tag XML yang mendefinisikan baris tabel dalam dokumen XML. Ketik nama tanpa kurung sudut < >. Nama harus sesuai dengan aturan tag dalam XML.

  Untuk informasi selengkapnya, lihat Menulis XHTML pengklasifikasi kustom.

JSON

- Jalur JSON

Untuk pengklasifikasi JSON, hal ini adalah path JSON ke objek, array, atau nilai yang mendefinisikan baris tabel yang sedang dibuat. Ketik nama, baik sintaksis JSON titik atau kurung dengan menggunakan operator yang didukung AWS Glue.

Untuk informasi lebih lanjut, lihat daftar operator di Menulis pengklasifikasi kustom JSON.

CSV

- Pembatas kolom

  Sebuah simbol atau karakter tunggal untuk menunjukkan apa yang memisahkan masing-masing entri kolom pada baris. Pilih pembatas dari daftar tersebut, atau pilih `Other` untuk memasukkan pembatas kustom.

- Simbol kutipan

  Sebuah karakter atau simbol tunggal untuk menunjukkan apa yang menggabungkan konten ke dalam satu nilai kolom tunggal. Harus berbeda dari pembatas kolom. Pilih simbol kutipan dari daftar tersebut, atau pilih `Other` untuk memasukkan karakter kutipan kustom.

- Judul kolom

  Menunjukkan perilaku untuk bagaimana mendeteksi judul kolom dalam file CSV. Anda dapat memilih `Has headings`, `No headings`, atau `Detect headings`. Jika file CSV kustom Anda memiliki judul kolom, masukkan daftar pembatas koma judul kolom.

- Izinkan file dengan kolom tunggal

  Untuk bisa diklasifikasikan sebagai CSV, data harus memiliki setidaknya dua kolom dan dua baris data. Gunakan opsi ini untuk mengizinkan pengolahan file yang berisi hanya satu kolom.

- Potong spasi sebelum mengidentifikasi nilai kolom

  Opsi ini menentukan apakah akan memotong nilai sebelum mengidentifikasi jenis nilai kolom.

- Jenis data khusus

  (Opsional) - Masukkan tipe data khusus dalam daftar yang dibatasi koma. Jenis data yang didukung adalah: “BINARY”, “BOOLEAN”, “DATE”, “DECIMAL”, “DOUBLE”, “FLOAT”, “INT”, “LONG”, “SHORT”, “STRING”, “TIMESTAMP”.

- CSV Serde

(Opsional) - A SerDe untuk memproses CSV di classifier, yang akan diterapkan dalam Katalog Data. Pilih dari`Open CSV SerDe`,`Lazy Simple SerDe`, atau`None`. Anda dapat menentukan `None` nilai saat Anda ingin crawler melakukan deteksi.

Untuk informasi selengkapnya, lihat Menulis pengklasifikasi kustom.

Menjadwalkan sebuah Crawler AWS Glue

Anda dapat menjalankan sebuah crawler AWS Glue atas permintaan atau berdasarkan jadwal reguler. Jadwal crawler dapat dinyatakan dalam format Cron. Untuk informasi selengkapnya, lihat Cron di Wikipedia.

Bila Anda membuat sebuah crawler berdasarkan jadwal, Anda dapat menentukan batasan tertentu, seperti frekuensi eksekusi crawler, pada hari dalam seminggu crawler tersebut berjalan, dan pada jam berapa. Batasan-batasan tersebut ini didasarkan pada cron. Saat menyiapkan jadwal untuk sebuah crawler, Anda harus mempertimbangkan fitur dan keterbatasan cron. Misalnya, jika Anda memilih untuk menjalankan crawler pada hari ke 31 setiap bulan, ingatlah bahwa ada bulan yang tidak terdiri dari 31 hari.

Crawl untuk setiap crawler hanya berlaku hingga 12 bulan

Untuk informasi selengkapnya tentang penggunaan cron untuk menjadwalkan Tugas dan crawler, lihat Jadwal berbasis waktu untuk pekerjaan dan crawler.

Melihat hasil dan detail crawler

Setelah crawler berjalan dengan sukses, crawler tersebut akan menciptakan definisi tabel dalam Katalog Data. Pilih Tabel di panel navigasi untuk melihat tabel yang dibuat oleh crawler Anda dalam basis data yang Anda tentukan.

Anda dapat melihat informasi yang terkait dengan crawler itu sendiri dengan langkah sebagai berikut:

- Halaman Crawler di AWS Glue konsol menampilkan properti berikut untuk crawler:

Properti	Deskripsi
Nama	Bila Anda membuat sebuah crawler, Anda harus memberinya nama yang unik.

Properti	Deskripsi
Status	Sebuah crawler dapat berada dalam keadaan siap, mulai, berhenti, dijadwalkan, atau jadwal berhenti. Sebuah crawler yang berjalan, progresnya dari mulai hingga berhenti. Anda dapat melanjutkan atau menjeda jadwal yang dilampirkan ke sebuah crawler.
Jadwal	Anda dapat memilih untuk menjalankan crawler Anda sesuai permintaan atau memilih frekuensi dengan jadwal. Untuk informasi selengkapnya tentang penjadwalan crawler, lihat Menjadwalkan sebuah Crawler.
Lari terakhir	Tanggal dan waktu terakhir kali crawler dijalankan.
Log	Tautkan ke log yang tersedia dari eksekusi crawler terakhir.
Tabel berubah dari proses terakhir	Jumlah tabel di AWS Glue Data Catalog yang diperbarui oleh proses crawler terbaru.

- Untuk melihat riwayat crawler, pilih Crawler di panel navigasi untuk melihat crawler yang Anda buat. Pilih crawler dari daftar crawler yang tersedia. Anda dapat melihat properti crawler dan melihat riwayat crawler di tab Crawler running.

  Tab Crawler run menampilkan informasi tentang setiap kali crawler berjalan, termasuk Waktu mulai (UTC), Waktu akhir (UTC), Durasi, Status, jam DPU, dan perubahan Tabel.

  Tab Crawler run hanya menampilkan crawl yang telah terjadi sejak tanggal peluncuran fitur riwayat crawler, dan hanya mempertahankan perayapan hingga 12 bulan. Perayapan yang lebih tua tidak akan dikembalikan.
- Untuk melihat informasi tambahan, pilih tab di halaman detail crawler. Setiap tab akan menampilkan informasi yang terkait dengan crawler.
  - Jadwal: Setiap jadwal yang dibuat untuk crawler akan terlihat di sini.
  - Sumber data: Semua sumber data yang dipindai oleh crawler akan terlihat di sini.

- Pengklasifikasi: Semua pengklasifikasi yang ditetapkan ke crawler akan terlihat di sini.
- Tag: Setiap tag yang dibuat dan ditetapkan ke AWS sumber daya akan terlihat di sini.

Parameter diatur pada tabel Katalog Data oleh crawler

Properti tabel ini diatur oleh AWS Glue crawler. Kami mengharapkan pengguna untuk mengkonsumsi `classification` dan `compressionType` properti. Properti lain, termasuk perkiraan ukuran tabel, digunakan untuk perhitungan internal, dan kami tidak menjamin keakuratan atau penerapannya pada kasus penggunaan pelanggan. Mengubah parameter ini dapat mengubah perilaku crawler, kami tidak mendukung alur kerja ini.

Kunci properti	Nilai properti
`UPDATED_BY_CRAWLER`	Nama crawler yang melakukan pembaruan.
`connectionName`	Nama koneksi dalam Katalog Data untuk crawler yang digunakan untuk menghubungkan ke penyimpanan data.
`recordCount`	Perkirakan jumlah catatan dalam tabel, berdasarkan ukuran file dan header.
`skip.header.line.count`	Baris dilewati untuk melewati header. Ditetapkan pada tabel yang diklasifikasikan sebagai CSV.
`CrawlerSchemaSerializerVersion`	Untuk penggunaan internal
`classification`	Format data, disimpulkan oleh crawler. Untuk informasi selengkapnya tentang format data yang didukung oleh AWS Glue crawler, lihatthe section called “Pengklasifikasi bawaan di AWS Glue”.
`CrawlerSchemaDeserializerVersion`	Untuk penggunaan internal
`sizeKey`	Ukuran gabungan file dalam tabel dirayapi.

Kunci properti	Nilai properti
`averageRecordSize`	Ukuran rata-rata baris dalam tabel, dalam byte.
`compressionType`	Jenis kompresi yang digunakan pada data dalam tabel. Untuk informasi selengkapnya tentang jenis kompresi yang didukung oleh AWS Glue crawler, lihatthe section called “Pengklasifikasi bawaan di AWS Glue”.
`typeOfData`	`file`, `table` atau`view`.
`objectCount`	Jumlah objek di bawah jalur Amazon S3 untuk tabel.

Properti tabel tambahan ini diatur oleh AWS Glue crawler untuk penyimpanan data Snowflake.

Kunci properti	Nilai properti
`aws:RawTableLastAltered`	Merekam stempel waktu terakhir yang diubah dari tabel Snowflake.
`ViewOriginalText`	Lihat pernyataan SQL.
`ViewExpandedText`	Lihat pernyataan SQL yang dikodekan dalam format Base64.
`ExternalTable:S3Location`	Lokasi Amazon S3 dari tabel eksternal Snowflake.
`ExternalTable:FileFormat`	Format file Amazon S3 dari tabel eksternal Snowflake.

Properti tabel tambahan ini ditetapkan oleh AWS Glue crawler untuk penyimpanan data tipe JDBC seperti Amazon Redshift, Microsoft SQL Server, MySQL, PostgreSQL, dan Oracle.

Kunci properti	Nilai properti
`aws:RawType`	Saat crawler menyimpan data dalam Katalog Data, perayap menerjemahkan tipe data ke tipe yang kompatibel dengan HIVE, yang berkali-kali menyebabkan informasi pada tipe data asli hilang. Crawler mengeluarkan `aws:RawType` parameter untuk menyediakan tipe data tingkat asli.
`aws:RawColumnComment`	Jika komentar dikaitkan dengan kolom dalam database, crawler mengeluarkan komentar yang sesuai dalam tabel katalog. String komentar dipotong menjadi 255 byte. Komentar tidak didukung untuk Microsoft SQL Server.
`aws:RawTableComment`	Jika komentar dikaitkan dengan tabel dalam database, crawler mengeluarkan komentar yang sesuai dalam tabel katalog. String komentar dipotong menjadi 255 byte. Komentar tidak didukung untuk Microsoft SQL Server.

Menyesuaikan perilaku crawler

Saat sebuah crawler berjalan, crawler tersebut mungkin mengalami perubahan pada penyimpanan data Anda yang menghasilkan sebuah skema atau partisi yang berbeda dari perayapan sebelumnya. Anda dapat menggunakan AWS Management Console atau AWS Glue API untuk mengonfigurasi cara crawler memproses jenis perubahan tertentu.

Topik

- Perayapan tambahan untuk menambahkan partisi baru
- Mengatur opsi konfigurasi crawler indeks partisi
- Mempercepat crawl menggunakan notifikasi acara Amazon S3
- Bagaimana mencegah crawler mengubah skema yang ada
- Cara membuat skema tunggal untuk setiap Amazon S3 termasuk jalur
- Cara menentukan lokasi tabel dan tingkat partisi
- Cara menentukan jumlah maksimum tabel yang diizinkan untuk dibuat oleh crawler
- Cara menentukan opsi konfigurasi untuk penyimpanan data Delta Lake

- Cara mengonfigurasi crawler untuk menggunakan kredenal Lake Formation

Console

Bila Anda menentukan sebuah crawler menggunakan konsol AWS Glue, Anda memiliki beberapa pilihan untuk mengkonfigurasi perilaku crawler Anda. Untuk informasi tentang cara menggunakan konsol AWS Glue untuk menambahkan sebuah crawler, lihat Mengkonfigurasi crawler.

Ketika sebuah crawler berjalan terhadap penyimpanan data yang di-crawl sebelumnya, mungkin ia menemukan bahwa skema telah berubah atau bahwa beberapa objek dalam penyimpanan data telah dihapus. Crawler tersebut mencatat log perubahan skema. Tergantung pada jenis sumber crawler, tabel dan partisi baru dapat dibuat terlepas dari kebijakan perubahan skemanya.

Untuk menentukan apa yang dilakukan crawler ketika crawler tersebut menemukan perubahan dalam skema, Anda dapat memilih salah satu tindakan berikut pada konsol:

- Memperbarui definisi tabel dalam Katalog Data — Menambahkan kolom baru, menghapus kolom yang hilang, dan mengubah definisi kolom yang ada di AWS Glue Data Catalog. Menghapus metadata yang tidak diatur oleh crawler tersebut. Ini adalah pengaturan default.
- Menambahkan kolom baru saja — Untuk tabel yang memetakan ke penyimpanan data Amazon S3, menambahkan kolom baru seperti yang ditemukan, tetapi tidak menghapus atau mengubah jenis kolom yang ada di Katalog Data. Pilih opsi ini bila kolom saat ini dalam Katalog Data benar dan Anda tidak ingin crawler menghapus atau mengubah jenis kolom yang ada. Jika atribut tabel Amazon S3 mendasar berubah, seperti klasifikasi, jenis kompresi, atau CSV pembatas, tandai tabel sebagai tidak lagi digunakan. Mempertahankan format input dan format output seperti yang ada dalam Katalog Data. Perbarui SerDe parameter hanya jika parameternya adalah salah satu yang diatur oleh crawler. Untuk semua penyimpanan data lainnya, ubah definisi kolom yang ada.
- Mengabaikan perubahan dan tidak memperbarui tabel di Katalog Data — Hanya tabel dan partisi baru dibuat.

  Ini adalah pengaturan default untuk perayapan tambahan.

Sebuah crawler juga dapat menemukan partisi baru atau yang diubah. Secara default, partisi baru ditambahkan dan partisi yang ada diperbarui jika mereka berubah. Selain itu, Anda dapat mengatur opsi konfigurasi crawler menjadi Perbarui semua partisi baru dan yang sudah ada dengan metadata dari tabel pada konsol AWS Glue. Ketika opsi ini disetel, partisi mewarisi

properti metadata—seperti klasifikasi, format input, format keluaran, informasi, dan skema—dari tabel induknya. SerDe Setiap perubahan pada properti ini dalam sebuah tabel disebarkan ke partisi-partisinya. Bila opsi konfigurasi ini diatur pada crawler yang ada, maka partisi yang ada akan diperbarui untuk mencocokkan properti tabel induk saat eksekusi crawler berikutnya.

Untuk menentukan apa yang dilakukan crawler saat menemukan sebuah objek yang dihapus dalam penyimpanan data, pilih salah satu tindakan berikut:

- Hapus tabel dan partisi dari Katalog Data
- Abaikan perubahan dan jangan perbarui tabel di Katalog Data

  Ini adalah pengaturan default untuk perayapan tambahan.
- Tandai tabel sebagai tidak lagi digunakan dalam Katalog Data — Ini adalah pengaturan default.

AWS CLI

```
aws glue create-crawler \
--name "your-crawler-name" \
--role "your-iam-role-arn" \
--database-name "your-database-name" \
--targets 'S3Targets=[{Path="s3://your-bucket-name/path-to-data"}]' \
--configuration '{"Version": 1.0, "CrawlerOutput": {"Partitions":
 {"AddOrUpdateBehavior": "InheritFromTable"}, "Tables": {"AddOrUpdateBehavior":
 "MergeNewColumns"}}}'
```

API

Saat Anda menentukan crawler menggunakan AWS Glue API, Anda dapat memilih dari beberapa bidang untuk mengonfigurasi crawler Anda. `SchemaChangePolicy` di API crawler menentukan apa yang dilakukan crawler saat menemukan terjadinya perubahan skema atau objek yang dihapus. Crawler mencatat skema perubahan saat berjalan.

Contoh kode python yang menunjukkan opsi konfigurasi crawler

```
import boto3
import json

# Initialize a boto3 client for AWS Glue
glue_client = boto3.client('glue', region_name='us-east-1')  # Replace 'us-east-1'
 with your desired AWS region
```

```
# Define the crawler configuration
crawler_configuration = {
    "Version": 1.0,
    "CrawlerOutput": {
        "Partitions": {
            "AddOrUpdateBehavior": "InheritFromTable"
        },
        "Tables": {
            "AddOrUpdateBehavior": "MergeNewColumns"
        }
    }
}

configuration_json = json.dumps(crawler_configuration)
# Create the crawler with the specified configuration
response = glue_client.create_crawler(
    Name='your-crawler-name',  # Replace with your desired crawler name
    Role='crawler-test-role',  # Replace with the ARN of your IAM role for Glue
    DatabaseName='default',  # Replace with your target Glue database name
    Targets={
        'S3Targets': [
            {
                'Path': "s3://your-bucket-name/path/",  # Replace with your S3 path
 to the data
            },
        ],
        # Include other target types like 'JdbcTargets' if needed
    },
    Configuration=configuration_json,
    # Include other parameters like Schedule, Classifiers, TablePrefix,
 SchemaChangePolicy, etc., as needed
)

print(response)a
```

Ketika sebuah crawler berjalan, tabel baru dan partisi selalu dibuat terlepas dari kebijakan perubahan skema. Anda dapat memilih salah satu tindakan berikut di bidang `UpdateBehavior` dalam struktur `SchemaChangePolicy` untuk menentukan apa yang harus dilakukan crawler tidak ketika crawler tersebut menemukan skema tabel yang berubah:

- `UPDATE_IN_DATABASE` — Memperbarui tabel di AWS Glue Data Catalog. Menambahkan kolom baru, menghapus kolom yang hilang, dan mengubah definisi kolom yang sudah ada. Menghapus metadata yang tidak diatur oleh crawler tersebut.
- `LOG` — Mengabaikan perubahan, dan tidak memperbarui tabel di Katalog Data.

  Ini adalah pengaturan default untuk perayapan tambahan.

Anda juga dapat mengganti struktur `SchemaChangePolicy` menggunakan sebuah objek JSON yang disediakan dalam bidang `Configuration` API crawler. Objek JSON ini dapat berisi pasangan nilai kunci untuk mengatur kebijakan untuk tidak memperbarui kolom yang ada dan hanya menambahkan kolom baru. Sebagai contoh, berikan objek JSON berikut sebagai string:

```
{
   "Version": 1.0,
   "CrawlerOutput": {
      "Tables": { "AddOrUpdateBehavior": "MergeNewColumns" }
   }
}
```

Opsi ini sesuai dengan opsi Tambahkan kolom baru saja pada konsol AWS Glue. Ia menggantikan struktur `SchemaChangePolicy` untuk tabel yang dihasilkan dari perayapan penyimpanan data Amazon S3 saja. Pilih opsi ini jika Anda ingin mempertahankan metadata yang ada di Katalog Data (sumber kebenaran). Kolom baru ditambahkan saat mereka ditemui, termasuk tipe data bersarang. Tapi kolom yang ada tidak dihapus, dan jenisnya tidak berubah. Jika atribut tabel Amazon S3 berubah secara signifikan, tandai tabel sebagai tidak lagi digunakan, dan mencatat log peringatan bahwa atribut yang tidak kompatibel perlu diubah. Opsi ini tidak berlaku untuk crawler inkremental.

Ketika sebuah crawler berjalan terhadap penyimpanan data yang di-crawl sebelumnya, crawler mungkin menemukan partisi baru atau partisi yang diubah. Secara default, partisi baru ditambahkan dan partisi yang ada diperbarui jika mereka berubah. Selain itu, Anda dapat mengatur opsi konfigurasi crawler ke `InheritFromTable` (sesuai dengan opsi Perbarui semua partisi baru dan yang sudah ada dengan metadata dari tabel pada konsol AWS Glue). Ketika opsi ini diatur, partisi mewarisi properti metadata dari tabel induknya, seperti klasifikasi, format input, format output, SerDe informasi, dan skema. Setiap perubahan properti pada tabel induk disebarkan ke partisi-partisinya.

Bila opsi konfigurasi ini diatur pada crawler yang ada, maka partisi yang ada akan diperbarui untuk mencocokkan properti tabel induk saat eksekusi crawler berikutnya. Perilaku ini diatur di bidang `Configuration` API crawler. Sebagai contoh, berikan objek JSON berikut sebagai string:

```
{
   "Version": 1.0,
   "CrawlerOutput": {
      "Partitions": { "AddOrUpdateBehavior": "InheritFromTable" }
   }
}
```

Bidang `Configuration` API crawler dapat mengatur beberapa pilihan konfigurasi. Sebagai contoh, untuk mengkonfigurasi output crawler baik untuk partisi ataupun tabel, Anda dapat memberikan sebuah representasi string dari objek JSON berikut:

```
{
    "Version": 1.0,
    "CrawlerOutput": {
       "Partitions": { "AddOrUpdateBehavior": "InheritFromTable" },
       "Tables": {"AddOrUpdateBehavior": "MergeNewColumns" }
    }
}
```

Anda dapat memilih salah satu tindakan berikut untuk menentukan apa yang dilakukan crawler saat menemukan objek yang dihapus dalam penyimpanan data. Bidang `DeleteBehavior` dalam struktur `SchemaChangePolicy` di API crawler menetapkan perilaku crawler ketika crawler menemukan objek yang dihapus.

- `DELETE_FROM_DATABASE` — Menghapus tabel dan partisi dari Katalog Data.
- `LOG` — Mengabaikan perubahannya. Jangan memperbarui Katalog Data. Tulis pesan log sebagai gantinya.
- `DEPRECATE_IN_DATABASE` — Tandai tabel sebagai tidak lagi digunakan dalam Katalog Data. Ini adalah pengaturan default.

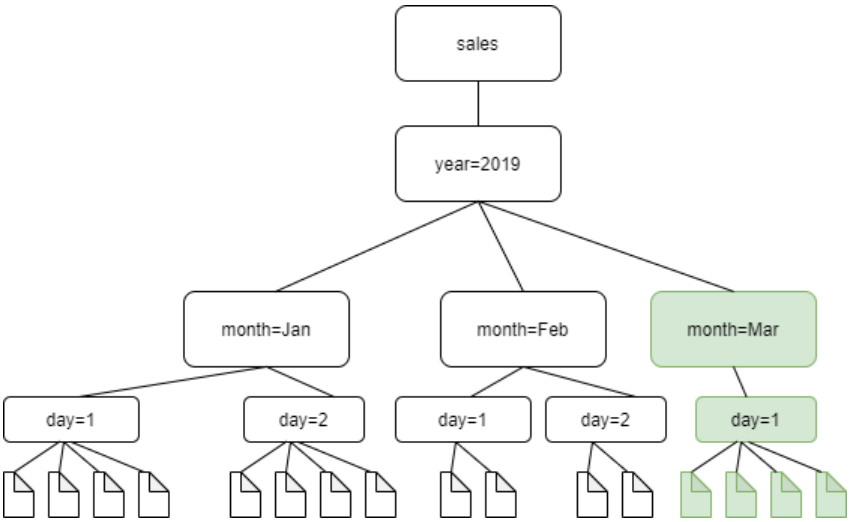
Perayapan tambahan untuk menambahkan partisi baru

Crawler menyediakan opsi untuk menambahkan partisi baru yang menghasilkan crawl yang lebih cepat untuk kumpulan data tambahan dengan skema tabel stabil. Kasus penggunaan yang umum

adalah untuk crawler terjadwal, di mana selama setiap perayapan, partisi baru ditambahkan. Ketika opsi ini diaktifkan, pertama-tama akan menjalankan crawl lengkap pada kumpulan data target untuk memungkinkan crawler merekam skema awal dan struktur partisi. Selama rawl ulang, partisi baru akan ditambahkan ke tabel yang ada hanya jika skema kompatibel. Tidak ada perubahan skema yang dibuat dan tidak ada tabel baru yang akan ditambahkan ke Katalog Data setelah crawl pertama dijalankan.

Anda dapat menggunakan opsi ini saat mengatur sumber data Amazon S3. Anda dapat mengatur `RecrawlPolicy` with `RecrawlBehavior` sebagai “Crawl_New_Folders” di `CreateCrawler` API atau Perayap berikutnya berjalan sebagai Crawl sub-folder baru hanya di konsol.

Melanjutkan dengan contoh di the section called “Bagaimana crawler menentukan kapan harus membuat partisi?”, diagram berikut menunjukkan bahwa file untuk bulan Maret telah ditambahkan.



Jika Anda menetapkan `RecrawlBehavior` sebagai opsi “Crawl_New_Folders”, hanya folder baru, yang dirayapi. `month=Mar`

Catatan dan batasan

Bila opsi ini diaktifkan, Anda tidak dapat mengubah penyimpanan data target Amazon S3 saat mengedit crawler. Opsi ini memengaruhi pengaturan konfigurasi crawler tertentu. Bila diaktifkan, tindakan ini akan memaksa perilaku pembaruan dan menghapus perilaku crawler ke `LOG`. Ini artinya bahwa:

- Jika menemukan objek di mana skema tidak kompatibel, crawler tidak akan menambahkan objek dalam Katalog Data, dan menambahkan detail ini sebagai log di Log. CloudWatch
- Ini tidak akan memperbarui objek yang dihapus di Katalog Data.

Untuk informasi selengkapnya, lihat the section called “Menyesuaikan perilaku crawler”.

Mengatur opsi konfigurasi crawler indeks partisi

Katalog Data mendukung indeks partisi untuk menyediakan pencarian yang efisien untuk partisi tertentu. Untuk informasi selengkapnya, lihat Bekerja dengan indeks partisi di AWS Glue. AWS Glue Crawler membuat indeks partisi untuk target Amazon S3 dan Delta Lake dengan tuli.

Saat Anda mendefinisikan cralwer, opsi untuk Membuat indeks partisi secara otomatis diaktifkan secara default di bawah Opsi lanjutan pada halaman Set output dan penjadwalan.

Untuk menonaktifkan opsi ini, Anda dapat membatalkan pilihan kotak centang Buat indeks partisi secara otomatis di konsol. Anda juga dapat menonaktifkan opsi ini dengan menggunakan crawler API, atur `CreatePartitionIndex` di. `Configuration` Nilai default-nya adalah betul.

Catatan penggunaan untuk indeks partisi

- Tabel yang dibuat oleh crawler tidak memiliki variabel secara `partition_filtering.enabled` default. Untuk informasi selengkapnya, lihat pengindeksan dan pemfilteran AWS Glue partisi.
- Membuat indeks partisi untuk partisi terenkripsi tidak didukung.

Mempercepat crawl menggunakan notifikasi acara Amazon S3

Alih-alih mencantumkan objek dari target Amazon S3 atau Katalog Data, Anda dapat mengonfigurasi crawler untuk menggunakan peristiwa Amazon S3 untuk menemukan perubahan apa pun. Fitur ini meningkatkan waktu rawl ulang dengan menggunakan peristiwa Amazon S3 untuk mengidentifikasi perubahan antara dua crawl dengan mencantumkan semua file dari subfolder yang memicu peristiwa alih-alih mencantumkan target Amazon S3 atau Katalog Data lengkap.

Crawl pertama mencantumkan semua objek Amazon S3 dari target. Setelah crawl pertama berhasil, Anda dapat memilih untuk meng-rawl ulang secara manual atau pada jadwal yang ditetapkan. Crawler hanya akan mencantumkan objek dari peristiwa tersebut alih-alih mencantumkan semua objek.

Keuntungan pindah ke crawler berbasis acara Amazon S3 adalah:

- Recrawl lebih cepat karena daftar semua objek dari target tidak diperlukan, alih-alih daftar folder tertentu dilakukan di mana objek ditambahkan atau dihapus.
- Pengurangan biaya crawl keseluruhan karena daftar folder tertentu dilakukan di mana objek ditambahkan atau dihapus.

Crawl peristiwa Amazon S3 berjalan dengan menggunakan peristiwa Amazon S3 dari antrean SQS berdasarkan jadwal crawler. Tidak akan ada biaya jika tidak ada acara dalam antrian. Acara Amazon S3 dapat dikonfigurasi untuk langsung masuk ke antrian SQS atau dalam kasus di mana beberapa konsumen memerlukan acara yang sama, kombinasi SNS dan SQS. Untuk informasi selengkapnya, lihat the section called “Menyiapkan akun Anda untuk pemberitahuan acara Amazon S3”.

Setelah membuat dan mengonfigurasi crawler dalam mode peristiwa, crawl pertama berjalan dalam mode daftar dengan melakukan daftar lengkap target Amazon S3 atau Katalog Data. Log berikut mengonfirmasi pengoperasian crawl dengan menggunakan peristiwa Amazon S3 setelah perayapan pertama yang berhasil: “Perayapan berjalan dengan menggunakan peristiwa Amazon S3.”

Setelah membuat crawl peristiwa Amazon S3 dan memperbarui properti crawler yang dapat memengaruhi perayapan, crawl beroperasi dalam mode daftar dan log berikut ditambahkan: “Perayapan tidak berjalan dalam mode peristiwa S3”.

> **Note**
>
> Jumlah maksimum pesan yang akan dikonsumsi adalah 10.000 pesan per crawl.

Target katalog

Ketika targetnya adalah Katalog Data, crawler memperbarui tabel yang ada di Katalog Data dengan perubahan (misalnya, partisi tambahan dalam tabel).

Topik

- Menyiapkan akun Anda untuk pemberitahuan acara Amazon S3
- Menggunakan enkripsi dengan perayap peristiwa Amazon S3

Menyiapkan akun Anda untuk pemberitahuan acara Amazon S3

Bagian ini menjelaskan cara mengatur akun Anda untuk pemberitahuan peristiwa Amazon S3, dan memberikan instruksi untuk melakukannya menggunakan skrip, atau konsol. AWS Glue

Prasyarat

Selesaikan tugas pengaturan berikut. Perhatikan nilai dalam tanda kurung merujuk pengaturan yang dapat dikonfigurasi dari skrip.

1. Buat bucket Amazon S3 ()`s3_bucket_name`.

2. Identifikasi target crawler (`folder_name`, seperti “test1") yang merupakan jalur di bucket yang diidentifikasi.
3. Siapkan nama crawler () `crawler_name`
4. Siapkan nama Topik SNS (`sns_topic_name`) yang bisa sama dengan nama crawler.
5. Siapkan AWS Wilayah tempat crawler dijalankan dan bucket S3 ada ()`region`.
6. Secara opsional siapkan alamat email jika email digunakan untuk mendapatkan acara `subscribing_email` Amazon S3 ().

Anda juga dapat menggunakan CloudFormation tumpukan untuk membuat sumber daya Anda. Selesaikan langkah-langkah berikut:

1. Luncurkan CloudFormation tumpukan Anda di AS Timur (Virginia N.):
2. Di bawah Parameter, masukkan nama untuk bucket Amazon S3 Anda (sertakan nomor akun Anda).
3. Pilih `I acknowledge that AWS CloudFormation might create IAM resources with custom names`.
4. Pilih `Create stack`.

Pembatasan:

- Hanya satu target yang didukung oleh crawler, baik untuk target Amazon S3 atau Katalog Data.
- SQS pada VPC pribadi tidak didukung.
- Pengambilan sampel Amazon S3 tidak didukung.
- Target crawler harus berupa folder untuk target Amazon S3, atau satu atau AWS Glue beberapa tabel Katalog Data untuk target Katalog Data.
- Wildcard jalur 'semuanya' tidak didukung: s3: //%
- Untuk target Katalog Data, semua tabel katalog harus mengarah ke bucket Amazon S3 yang sama untuk mode acara Amazon S3.
- Untuk target Katalog Data, tabel katalog tidak boleh mengarah ke lokasi Amazon S3 dalam format Delta Lake (berisi folder _symlink, atau memeriksa tabel katalog). `InputFormat`

Untuk menggunakan crawler berbasis peristiwa Amazon S3, Anda harus mengaktifkan pemberitahuan peristiwa di bucket S3 dengan peristiwa yang difilter dari awalan yang sama dengan

target S3 dan penyimpanan di SQS. Anda dapat mengatur SQS dan pemberitahuan acara melalui konsol dengan mengikuti langkah-langkah di Walkthrough: Mengonfigurasi bucket untuk notifikasi atau menggunakan. the section called “Skrip untuk menghasilkan SQS dan mengonfigurasi peristiwa Amazon S3 dari target”

Kebijakan SQS

Tambahkan kebijakan SQS berikut yang harus dilampirkan ke peran yang digunakan oleh crawler.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "sqs:DeleteMessage",
                "sqs:GetQueueUrl",
                "sqs:ListDeadLetterSourceQueues",
                "sqs:ReceiveMessage",
                "sqs:GetQueueAttributes",
                "sqs:ListQueueTags",
                "sqs:SetQueueAttributes",
                "sqs:PurgeQueue"
            ],
            "Resource": "arn:aws:sqs:{region}:{accountID}:cfn-sqs-queue"
        }
    ]
}
```

Skrip untuk menghasilkan SQS dan mengonfigurasi peristiwa Amazon S3 dari target

Setelah memastikan prasyarat terpenuhi, Anda dapat menjalankan skrip Python berikut untuk membuat SQS. Ganti pengaturan yang Dapat Dikonfigurasi dengan nama yang disiapkan dari prasyarat.

> **Note**
>
> Setelah menjalankan skrip, masuk ke konsol SQS untuk menemukan ARN dari SQS yang dibuat.

Amazon SQS menetapkan batas waktu visibilitas, periode waktu di mana Amazon SQS mencegah konsumen lain menerima dan memproses pesan. Setel batas waktu visibilitas kira-kira sama dengan waktu proses crawl.

```
#!venv/bin/python
import boto3
import botocore

#---------Start : READ ME FIRST ----------------------#
# 1. Purpose of this script is to create the SQS, SNS and enable S3 bucket
 notification.
#     The following are the operations performed by the scripts:
#      a. Enable S3 bucket notification to trigger 's3:ObjectCreated:' and
 's3:ObjectRemoved:' events.
#      b. Create SNS topic for fan out.
#      c. Create SQS queue for saving events which will be consumed by the crawler.
#          SQS Event Queue ARN will be used to create the crawler after running the
 script.
# 2. This script does not create the crawler.
# 3. SNS topic is created to support FAN out of S3 events. If S3 event is also used by
 another
#    purpose, SNS topic created by the script can be used.
# 1. Creation of bucket is an optional step.
#    To create a bucket set create_bucket variable to true.
# 2. The purpose of crawler_name is to easily locate the SQS/SNS.
#     crawler_name is used to create SQS and SNS with the same name as crawler.
# 3. 'folder_name' is the target of crawl inside the specified bucket 's3_bucket_name'
#
#---------End : READ ME FIRST -----------------------#


#--------------------------------#
# Start : Configurable settings  #
#--------------------------------#

#Create
region = 'us-west-2'
s3_bucket_name = 's3eventtestuswest2'
folder_name = "test"
crawler_name = "test33S3Event"
sns_topic_name = crawler_name
sqs_queue_name = sns_topic_name
create_bucket = False
```

```
#------------------------------#
# End : Configurable settings   #
#------------------------------#

# Define aws clients
dev = boto3.session.Session(profile_name='myprofile')
boto3.setup_default_session(profile_name='myprofile')
s3 = boto3.resource('s3', region_name=region)
sns = boto3.client('sns', region_name=region)
sqs = boto3.client('sqs', region_name=region)
client = boto3.client("sts")
account_id = client.get_caller_identity()["Account"]
queue_arn = ""


def print_error(e):
    print(e.message + ' RequestId: ' + e.response['ResponseMetadata']['RequestId'])

def create_s3_bucket(bucket_name, client):
    bucket = client.Bucket(bucket_name)
    try:
        if not create_bucket:
            return True
        response = bucket.create(
            ACL='private',
            CreateBucketConfiguration={
                'LocationConstraint': region
            },
        )
        return True
    except botocore.exceptions.ClientError as e:
        print_error(e)
        if 'BucketAlreadyOwnedByYou' in e.message:  # we own this bucket so continue
            print('We own the bucket already. Lets continue...')
            return True
    return False

def create_s3_bucket_folder(bucket_name, client, directory_name):
    s3.put_object(Bucket=bucket_name, Key=(directory_name + '/'))

def set_s3_notification_sns(bucket_name, client, topic_arn):
    bucket_notification = client.BucketNotification(bucket_name)
    try:
```

```
        response = bucket_notification.put(
            NotificationConfiguration={
                'TopicConfigurations': [
                    {
                        'Id' : crawler_name,
                        'TopicArn': topic_arn,
                        'Events': [
                            's3:ObjectCreated:*',
                            's3:ObjectRemoved:*',

                        ],
                        'Filter' :  {'Key': {'FilterRules': [{'Name': 'prefix',
 'Value': folder_name}]}}
                    },
                ]
            }
        )
        return True
    except botocore.exceptions.ClientError as e:
        print_error(e)
    return False


def create_sns_topic(topic_name, client):
    try:
        response = client.create_topic(
            Name=topic_name
        )
        return response['TopicArn']
    except botocore.exceptions.ClientError as e:
        print_error(e)
    return None

def set_sns_topic_policy(topic_arn, client, bucket_name):
    try:
        response = client.set_topic_attributes(
            TopicArn=topic_arn,
            AttributeName='Policy',
            AttributeValue='''{
              "Version": "2008-10-17",
              "Id": "s3-publish-to-sns",
              "Statement": [{
                  "Effect": "Allow",
```

```
                    "Principal": { "AWS" : "*" },
                    "Action": [ "SNS:Publish" ],
                    "Resource": "%s",
                    "Condition": {
                      "StringEquals": {
                          "AWS:SourceAccount": "%s"
                        },
                        "ArnLike": {
                            "aws:SourceArn": "arn:aws:s3:*:*:%s"
                        }
                    }
                }]
  }''' % (topic_arn, account_id, bucket_name)
        )
        return True
    except botocore.exceptions.ClientError as e:
        print_error(e)

    return False


def subscribe_to_sns_topic(topic_arn, client, protocol, endpoint):
    try:
        response = client.subscribe(
            TopicArn=topic_arn,
            Protocol=protocol,
            Endpoint=endpoint
        )
        return response['SubscriptionArn']
    except botocore.exceptions.ClientError as e:
        print_error(e)
    return None


def create_sqs_queue(queue_name, client):
    try:
        response = client.create_queue(
            QueueName=queue_name,
        )
        return response['QueueUrl']
    except botocore.exceptions.ClientError as e:
        print_error(e)
    return None
```

```
def get_sqs_queue_arn(queue_url, client):
    try:
        response = client.get_queue_attributes(
            QueueUrl=queue_url,
            AttributeNames=[
                'QueueArn',
            ]
        )
        return response['Attributes']['QueueArn']
    except botocore.exceptions.ClientError as e:
        print_error(e)
    return None

def set_sqs_policy(queue_url, queue_arn, client, topic_arn):
    try:
        response = client.set_queue_attributes(
            QueueUrl=queue_url,
            Attributes={
                'Policy': '''{
                  "Version": "2012-10-17",
                  "Id": "AllowSNSPublish",
                  "Statement": [
                    {
                      "Sid": "AllowSNSPublish01",
                      "Effect": "Allow",
                      "Principal": "*",
                      "Action": "SQS:SendMessage",
                      "Resource": "%s",
                      "Condition": {
                        "ArnEquals": {
                          "aws:SourceArn": "%s"
                        }
                      }
                    }
                  ]
}''' % (queue_arn, topic_arn)
            }
        )
        return True
    except botocore.exceptions.ClientError as e:
        print_error(e)
    return False
```

```
if __name__ == "__main__":
    print('Creating S3 bucket %s.' % s3_bucket_name)
    if create_s3_bucket(s3_bucket_name, s3):
        print('\nCreating SNS topic %s.' % sns_topic_name)
        topic_arn = create_sns_topic(sns_topic_name, sns)
        if topic_arn:
            print('SNS topic created successfully: %s' % topic_arn)

            print('Creating SQS queue %s' % sqs_queue_name)
            queue_url = create_sqs_queue(sqs_queue_name, sqs)
            if queue_url is not None:
                print('Subscribing sqs queue with sns.')
                queue_arn = get_sqs_queue_arn(queue_url, sqs)
                if queue_arn is not None:
                    if set_sqs_policy(queue_url, queue_arn, sqs, topic_arn):
                        print('Successfully configured queue policy.')
                        subscription_arn = subscribe_to_sns_topic(topic_arn, sns,
 'sqs', queue_arn)
                        if subscription_arn is not None:
                            if 'pending confirmation' in subscription_arn:
                                print('Please confirm SNS subscription by visiting the
 subscribe URL.')
                            else:
                                print('Successfully subscribed SQS queue: ' +
 queue_arn)
                        else:
                            print('Failed to subscribe SNS')
                    else:
                        print('Failed to set queue policy.')
                else:
                    print("Failed to get queue arn for %s" % queue_url)
            # ------------ End subscriptions to SNS topic -----------------

            print('\nSetting topic policy to allow s3 bucket %s to publish.' %
 s3_bucket_name)
            if set_sns_topic_policy(topic_arn, sns, s3_bucket_name):
                print('SNS topic policy added successfully.')
                if set_s3_notification_sns(s3_bucket_name, s3, topic_arn):
                    print('Successfully configured event for S3 bucket %s' %
 s3_bucket_name)
                    print('Create S3 Event Crawler using SQS ARN %s' % queue_arn)
                else:
                    print('Failed to configure S3 bucket notification.')
```

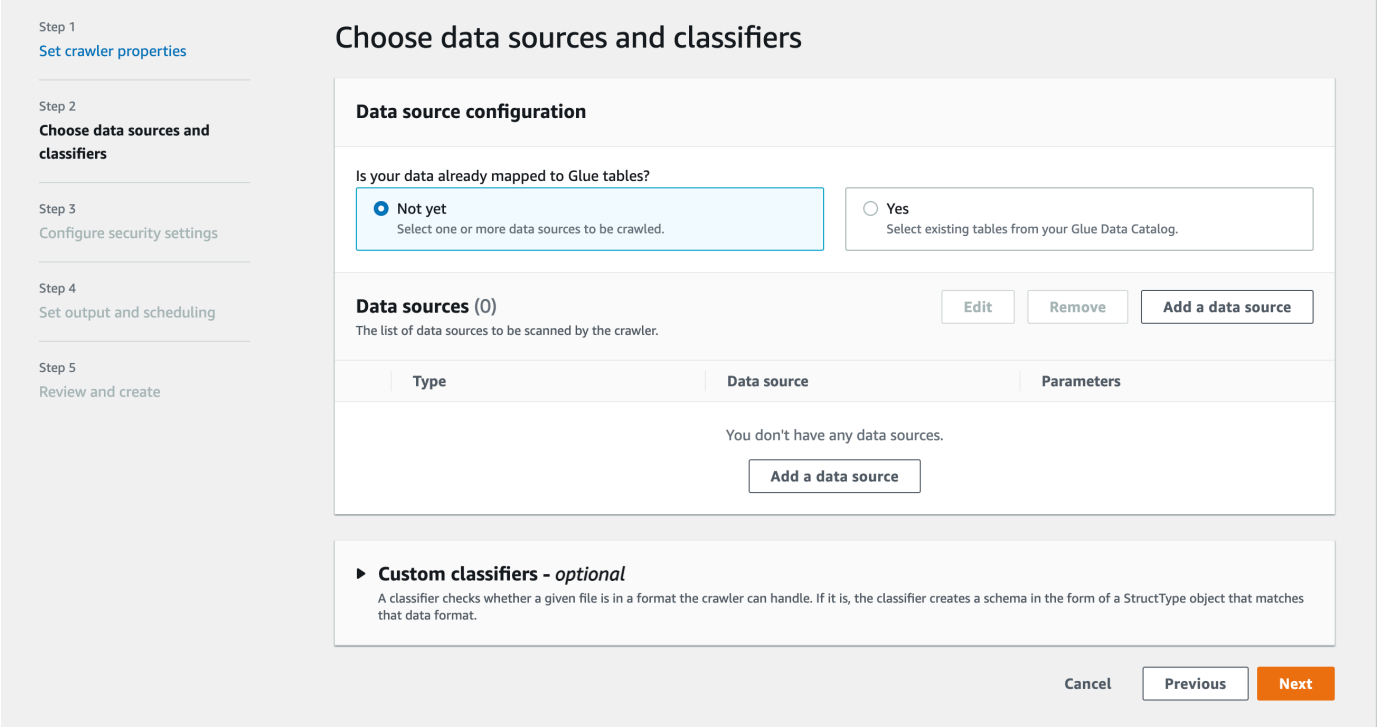
```
            else:
                print('Failed to add SNS topic policy.')
        else:
            print('Failed to create SNS topic.')
```

Menyiapkan crawler untuk notifikasi peristiwa Amazon S3 menggunakan konsol (target Amazon S3)

Untuk menyiapkan crawler untuk notifikasi peristiwa Amazon S3 menggunakan konsol untuk AWS Glue target Amazon S3:

1. Tetapkan properti crawler Anda. Untuk informasi selengkapnya, lihat Mengatur Opsi Konfigurasi Crawler di AWS Glue konsol.
2. Di bagian Konfigurasi sumber data, Anda ditanya Apakah data Anda sudah dipetakan ke AWS Glue tabel?

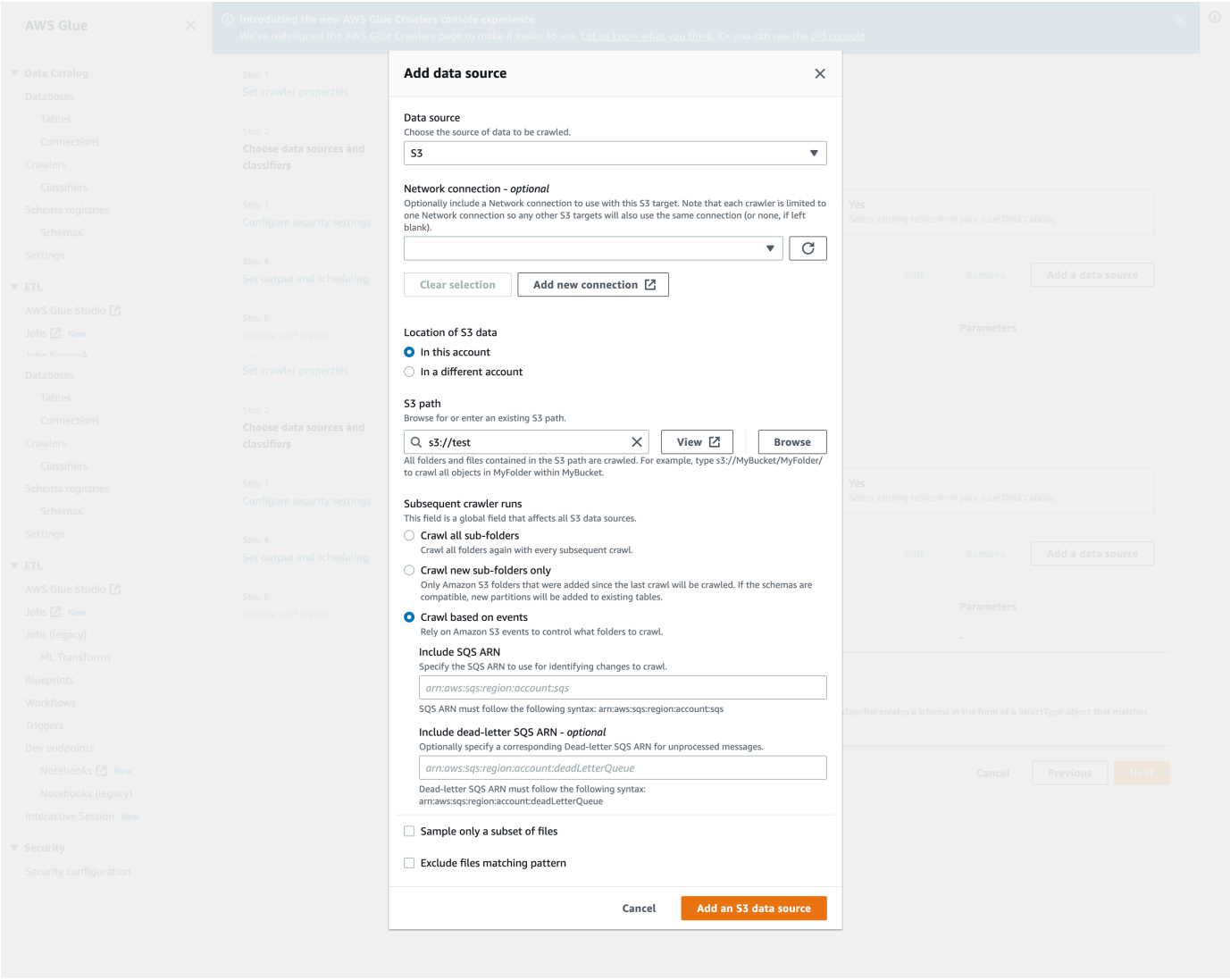
   Secara default Belum dipilih. Biarkan ini sebagai default karena Anda menggunakan sumber data Amazon S3 dan data belum dipetakan ke tabel. AWS Glue
3. Di bagian Sumber data, pilih Tambahkan sumber data.



4. Dalam modal Tambah sumber data, konfigurasikan sumber data Amazon S3:

   - Sumber data: Secara default, Amazon S3 dipilih.
   - Koneksi jaringan (Opsional): Pilih Tambahkan koneksi baru.

- Lokasi data Amazon S3: Secara default, Di akun ini dipilih.
- Jalur Amazon S3: Tentukan jalur Amazon S3 tempat folder dan file dirayapi.
- Perayap berikutnya berjalan: Pilih Crawl berdasarkan peristiwa untuk menggunakan notifikasi peristiwa Amazon S3 untuk crawler Anda.
- Sertakan SQS ARN: Tentukan parameter penyimpanan data termasuk SQS ARN yang valid. (Misalnya,`arn:aws:sqs:region:account:sqs`).
- Sertakan SQS ARN huruf mati (Opsional): Tentukan SQS ARN surat mati Amazon yang valid. (Misalnya,`arn:aws:sqs:region:account:deadLetterQueue`).
- Pilih Tambahkan sumber data Amazon S3.



Menyiapkan crawler untuk notifikasi peristiwa Amazon S3 menggunakan AWS CLI

Berikut ini adalah contoh AWS CLI panggilan Amazon S3 untuk membuat antrian SQS dan pemberitahuan acara penyiapan di bucket target Amazon S3.

```
S3 Event AWS CLI
aws sqs create-queue --queue-name MyQueue --attributes file://create-queue.json
create-queue.json
```
```
{
    "Policy": {
        "Version": "2012-10-17",
        "Id": "example-ID",
        "Statement": [
            {
                "Sid": "example-statement-ID",
                "Effect": "Allow",
                "Principal": {
                    "Service": "s3.amazonaws.com"
                },
                "Action": [
                    "SQS:SendMessage"
                ],
                "Resource": "SQS-queue-ARN",
                "Condition": {
                    "ArnLike": {
                        "aws:SourceArn": "arn:aws:s3:*:*:awsexamplebucket1"
                    },
                    "StringEquals": {
                        "aws:SourceAccount": "bucket-owner-account-id"
                    }
                }
            }
        ]
    }
}
```
```
aws s3api put-bucket-notification-configuration --bucket customer-data-pdx --
notification-configuration file://s3-event-config.json
s3-event-config.json
```
```
{
    "QueueConfigurations": [
```

```
        {
          "Id": "s3event-sqs-queue",
          "QueueArn": "arn:aws:sqs:{region}:{account}:queuename",
          "Events": [
                "s3:ObjectCreated:*",
                "s3:ObjectRemoved:*"
          ],
          "Filter": {
              "Key": {
                  "FilterRules": [
                      {
                          "Name": "Prefix",
                          "Value": "/json"
                      }
                  ]
              }
          }
        }
    ]
}
```

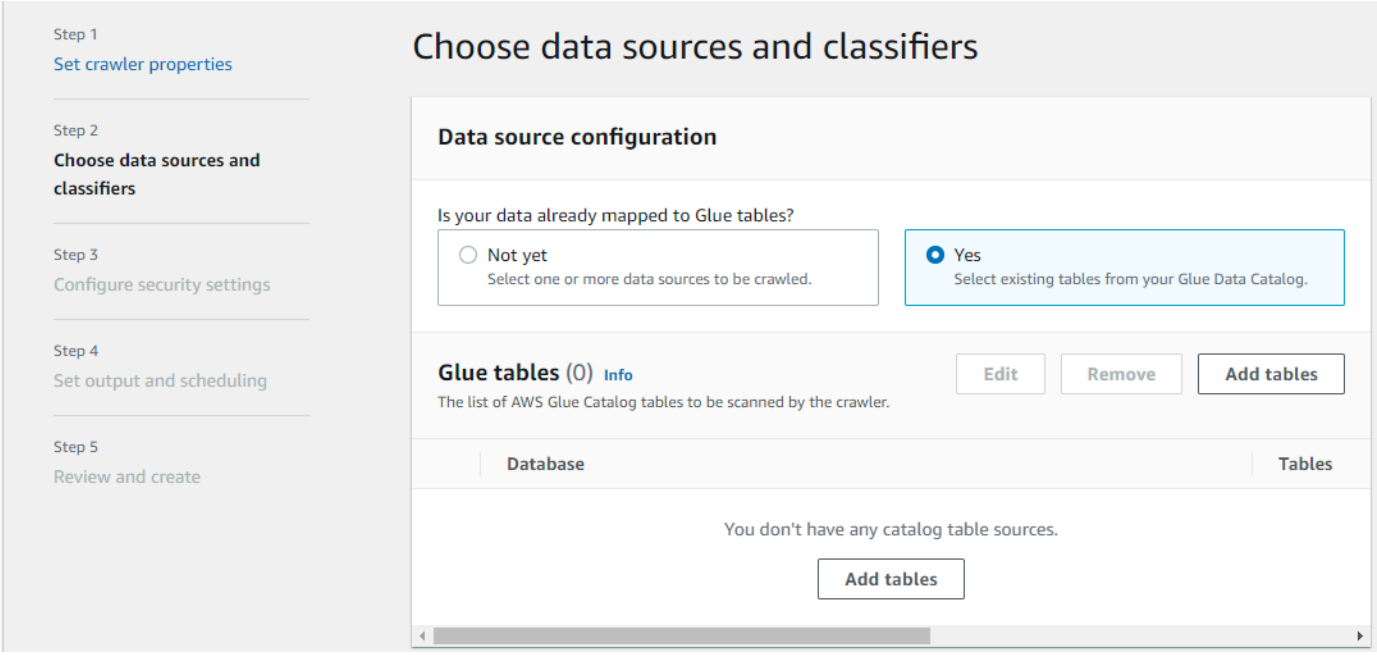
Create Crawler:

Menyiapkan crawler untuk notifikasi peristiwa Amazon S3 menggunakan konsol (Target Katalog Data)

Jika Anda memiliki target katalog, siapkan crawler untuk notifikasi peristiwa Amazon S3 menggunakan AWS Glue konsol:

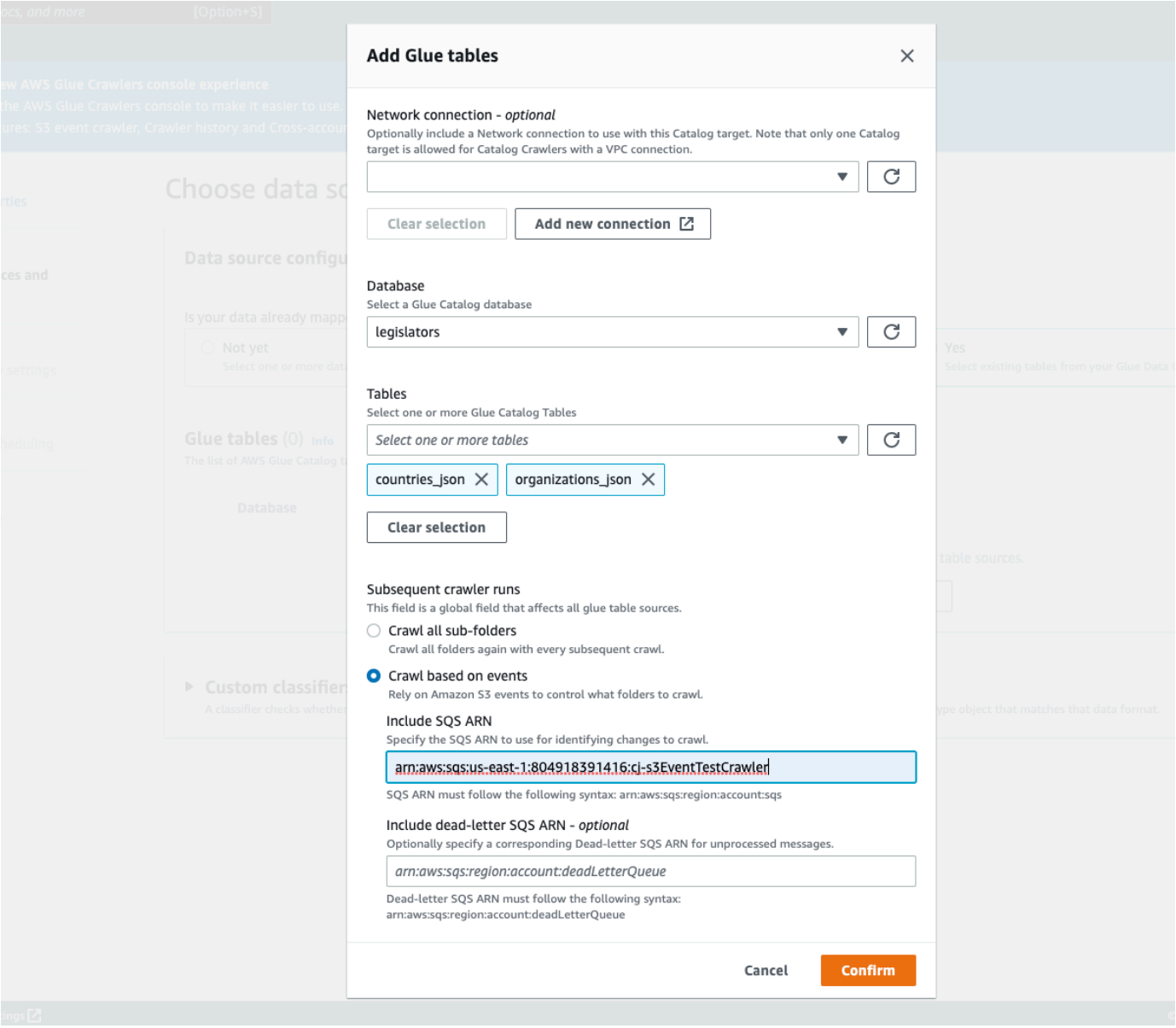
1. Tetapkan properti crawler Anda. Untuk informasi selengkapnya, lihat Mengatur Opsi Konfigurasi Crawler di AWS Glue konsol.
2. Di bagian Konfigurasi sumber data, Anda ditanya Apakah data Anda sudah dipetakan ke AWS Glue tabel?

   Pilih Ya untuk memilih tabel yang ada dari Katalog Data Anda sebagai sumber data Anda.
3. Di bagian Glue tables, pilih Add tables.



4. Dalam modal Tambahkan tabel, konfigurasikan database dan tabel:

   - Koneksi jaringan (Opsional): Pilih Tambahkan koneksi baru.
   - Database: Pilih database di Katalog Data.
   - Tabel: Pilih satu atau beberapa tabel dari database tersebut di Katalog Data.
   - Perayap berikutnya berjalan: Pilih Crawl berdasarkan peristiwa untuk menggunakan notifikasi peristiwa Amazon S3 untuk crawler Anda.
   - Sertakan SQS ARN: Tentukan parameter penyimpanan data termasuk SQS ARN yang valid. (Misalnya,`arn:aws:sqs:region:account:sqs`).
   - Sertakan SQS ARN huruf mati (Opsional): Tentukan SQS ARN surat mati Amazon yang valid. (Misalnya,`arn:aws:sqs:region:account:deadLetterQueue`).
   - Pilih Konfirmasi.



Menggunakan enkripsi dengan perayap peristiwa Amazon S3

Bagian ini menjelaskan penggunaan enkripsi pada SQS saja atau pada SQS dan Amazon S3.

Topik

- Mengaktifkan enkripsi hanya pada SQS
- Mengaktifkan enkripsi pada SQS dan Amazon S3
- Pertanyaan yang Sering Diajukan

Mengaktifkan enkripsi hanya pada SQS

Amazon SQS menyediakan enkripsi dalam perjalanan secara default. Untuk menambahkan Enkripsi Sisi Server (SSE) opsional ke antrian Anda, Anda dapat melampirkan kunci master pelanggan (CMK) di panel edit. Ini berarti SQS mengenkripsi semua data pelanggan saat istirahat di server SQS.

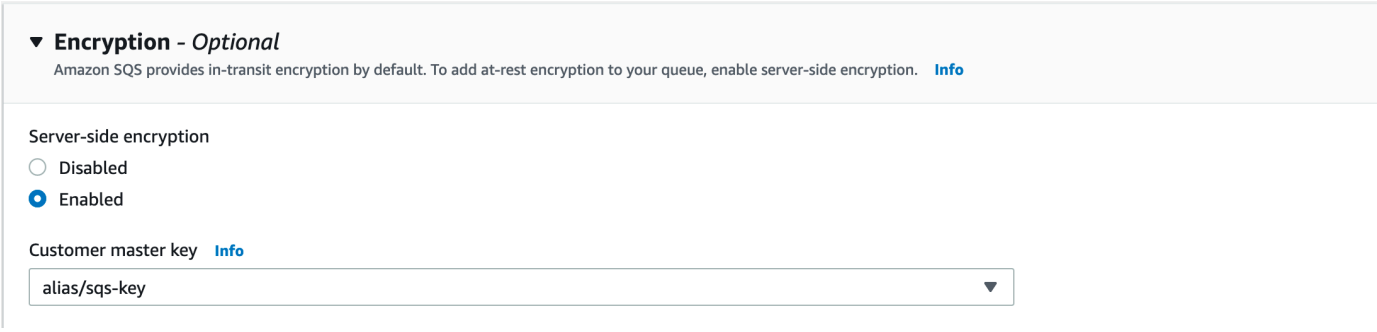
Buat Customer Master Key (CMK)

1. Pilih Key Management Service (KMS) > Customer Managed Keys > Create key.
2. Ikuti langkah-langkah untuk menambahkan alias dan deskripsi Anda sendiri.
3. Tambahkan peran IAM masing-masing yang Anda inginkan untuk dapat menggunakan kunci ini.
4. Dalam kebijakan utama, tambahkan pernyataan lain ke daftar “Pernyataan” sehingga kebijakan kunci kustom Anda memberikan izin penggunaan kunci yang memadai kepada Amazon SNS.

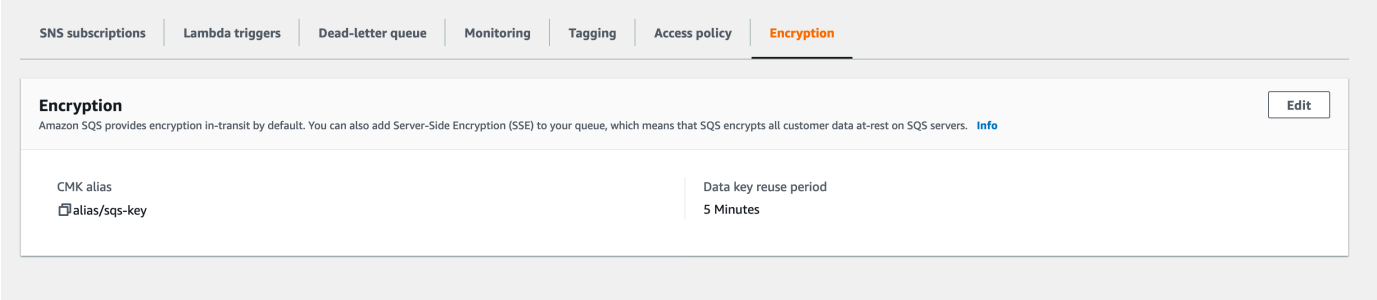
```
"Statement": [
       {
           "Effect": "Allow",
           "Principal": {
               "Service": "sns.amazonaws.com"
           },
           "Action": [
               "kms:GenerateDataKey",
               "kms:Decrypt"
           ],
           "Resource": "*"
       }
]
```

Aktifkan Server-Side Encryption (SSE) pada antrian Anda

1. Pilih Amazon SQS > Antrian > sqs_queue_name > tab Enkripsi.
2. Pilih Edit, dan gulir ke bawah ke drop down Encryption.
3. Pilih Diaktifkan untuk menambahkan SSE.
4. Pilih CMK yang Anda buat sebelumnya, dan bukan kunci default dengan nama`alias/aws/sqs`.



Setelah menambahkan ini, tab Enkripsi Anda diperbarui dengan kunci yang Anda tambahkan.



> **Note**
>
> Amazon SQS secara otomatis menghapus pesan yang telah berada dalam antrian selama lebih dari periode penyimpanan pesan maksimum. Periode penyimpanan pesan default adalah 4 hari. Untuk menghindari peristiwa yang hilang, ubah SQS MessageRetentionPeriodhingga maksimal 14 hari.

Mengaktifkan enkripsi pada SQS dan Amazon S3

Aktifkan Enkripsi Sisi Server (SSE) di SQS

1. Ikuti langkah-langkahnya di the section called “Mengaktifkan enkripsi hanya pada SQS”.
2. Pada langkah terakhir penyiapan CMK, berikan izin penggunaan kunci yang cukup kepada Amazon S3.

   Tempelkan yang berikut ini ke daftar “Pernyataan”:

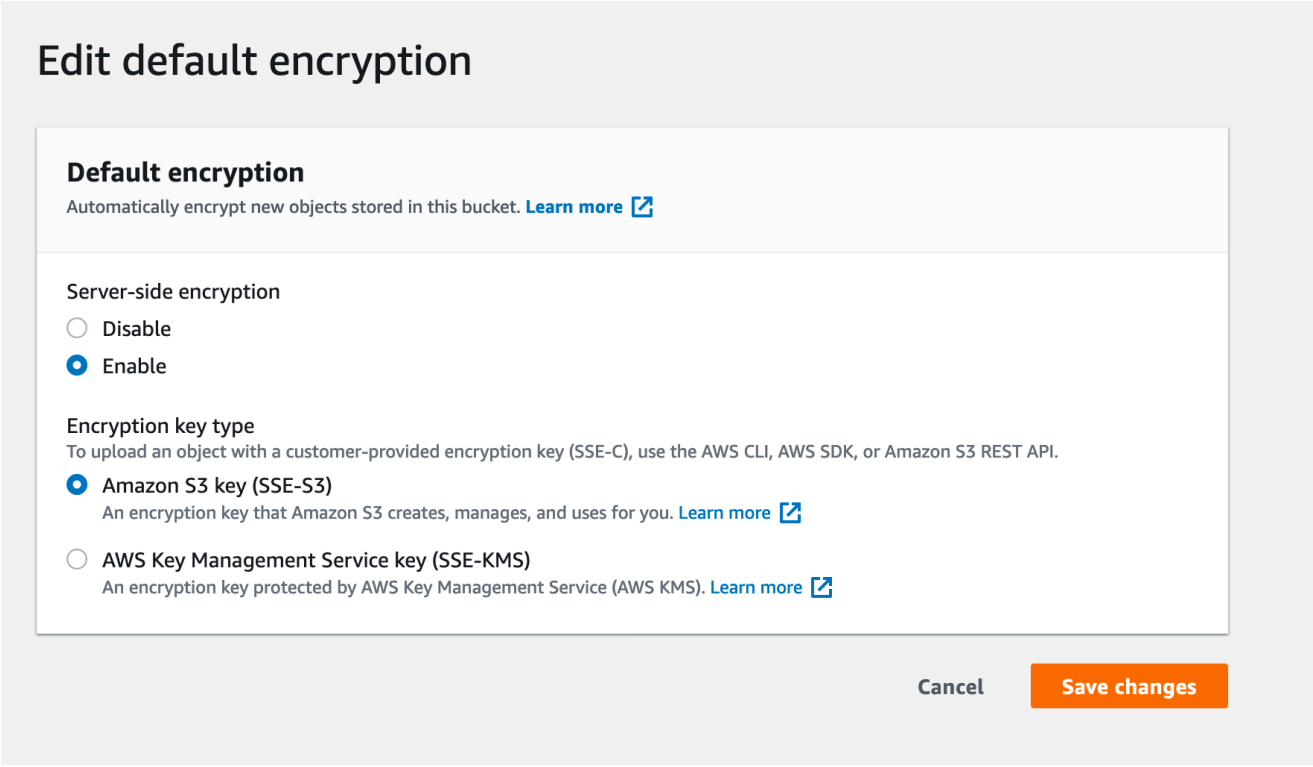
```
"Statement": [
      {
          "Effect": "Allow",
```

```
            "Principal": {
                "Service": "s3.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey",
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
]
```

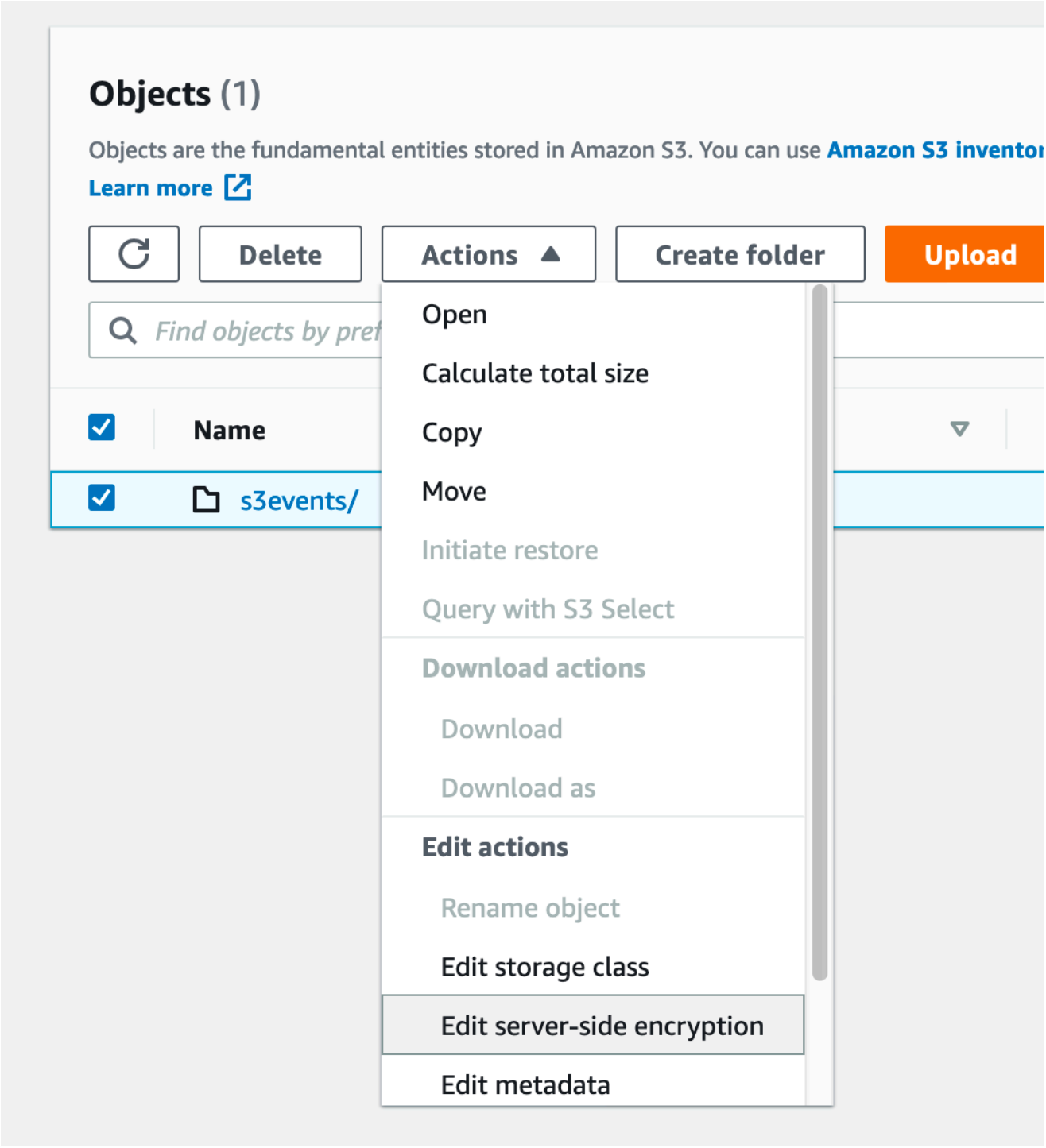
Aktifkan Enkripsi Sisi Server (SSE) di bucket Amazon S3

1. Ikuti langkah-langkahnya di the section called “Mengaktifkan enkripsi hanya pada SQS”.
2. Lakukan salah satu dari berikut:
   - Untuk mengaktifkan SSE untuk seluruh bucket S3 Anda, navigasikan ke tab Properties di bucket target Anda.

     Di sini Anda dapat mengaktifkan SSE dan memilih jenis enkripsi yang ingin Anda gunakan. Amazon S3 menyediakan kunci enkripsi yang dibuat, dikelola, dan digunakan Amazon S3 untuk Anda, atau Anda juga dapat memilih kunci dari KMS.



- Untuk mengaktifkan SSE pada folder tertentu, klik kotak centang di samping folder target Anda dan pilih Edit enkripsi sisi server di bawah drop-down Tindakan.



Pertanyaan yang Sering Diajukan

Mengapa pesan yang saya terbitkan ke topik Amazon SNS saya tidak dikirim ke antrean Amazon SQS berlangganan saya yang mengaktifkan enkripsi sisi server (SSE)?

Periksa kembali apakah antrian Amazon SQS Anda menggunakan:

1. Customer Master Key (CMK) yang dikelola oleh pelanggan. Bukan yang default yang disediakan oleh SQS.
2. CMK Anda dari (1) menyertakan kebijakan kunci khusus yang memberikan izin penggunaan kunci yang memadai kepada Amazon SNS.

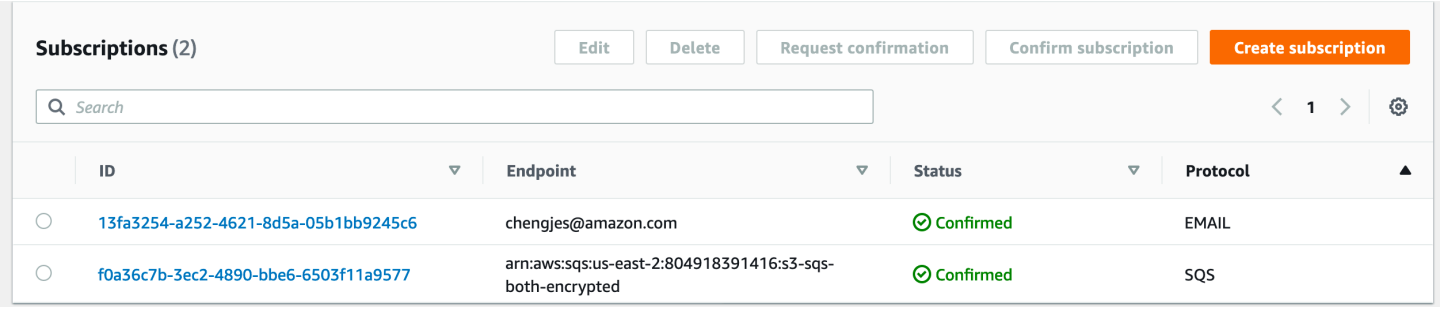
Untuk informasi lebih lanjut, lihat artikel ini di pusat pengetahuan.

Saya telah berlangganan notifikasi email, tetapi saya tidak menerima pembaruan email apa pun saat saya mengedit ember Amazon S3 saya.

Pastikan Anda telah mengonfirmasi alamat email Anda dengan mengklik tautan “Konfirmasi Berlangganan” di email Anda. Anda dapat memverifikasi status konfirmasi Anda dengan memeriksa tabel Langganan di bawah topik SNS Anda.

Pilih Amazon SNS > Topik > > `sns_topic_name` Tabel langganan.

Jika Anda mengikuti skrip prasyarat kami, Anda akan menemukan bahwa naskah sama dengan naskah prasyarat Anda. `sns_topic_name` `sqs_queue_name` Itu terlihat serupa dengan yang berikut ini:



Hanya beberapa folder yang saya tambahkan yang muncul di tabel saya setelah mengaktifkan enkripsi sisi server pada antrian SQS saya. Mengapa saya melewatkan beberapa parket?

Jika perubahan bucket Amazon S3 dilakukan sebelum mengaktifkan SSE pada antrean SQS Anda, mereka mungkin tidak diambil oleh crawler. Untuk memastikan bahwa Anda telah merayapi semua pembaruan ke bucket S3 Anda, jalankan crawler lagi dalam mode daftar (“Crawl All Folder”). Pilihan lainnya adalah memulai yang baru dengan membuat crawler baru dengan acara S3 diaktifkan.

Bagaimana mencegah crawler mengubah skema yang ada

Jika Anda tidak ingin crawler menimpa pembaruan yang Anda buat ke bidang yang ada dalam definisi tabel Amazon S3, maka Anda harus memilih opsi di konsol untuk Tambahkan kolom baru

saja atau tetapkan opsi konfigurasi `MergeNewColumns`. Hal ini berlaku untuk tabel dan partisi, kecuali `Partitions.AddOrUpdateBehavior` diganti ke `InheritFromTable`.

Jika Anda tidak ingin skema tabel berubah sama sekali saat sebuah crawler berjalan, tetapkan kebijakan perubahan skema ke `LOG`. Anda juga dapat mengatur opsi konfigurasi yang menetapkan skema partisi untuk mewarisi dari tabel tersebut.

Jika Anda mengkonfigurasi crawler di konsol, Anda dapat memilih tindakan-tindakan berikut:

- Abaikan perubahan dan jangan perbarui tabel di Katalog Data
- Perbarui semua partisi baru dan yang sudah ada dengan metadata dari tabel

Saat Anda mengkonfigurasi crawler tersebut menggunakan API, atur parameter berikut:

- Atur bidang `UpdateBehavior` dalam struktur `SchemaChangePolicy` ke `LOG`.
- Atur bidang `Configuration` dengan representasi string dari objek JSON berikut dalam API crawler; sebagai contoh:

```
{
   "Version": 1.0,
   "CrawlerOutput": {
      "Partitions": { "AddOrUpdateBehavior": "InheritFromTable" }
   }
}
```

Cara membuat skema tunggal untuk setiap Amazon S3 termasuk jalur

Secara default, ketika sebuah crawler mendefinisikan tabel untuk data yang disimpan di Amazon S3, ia mempertimbangkan kompatibilitas data dan kemiripan skemanya. Faktor kompatibilitas data yang dianggap termasuk apakah data mempunyai format yang sama (misalnya, JSON), mempunyai jenis kompresi yang sama (misalnya, GZIP), struktur path Amazon S3, dan atribut data lainnya. Skema kesamaan menjadi sebuah ukuran seberapa dekat keserupaan antara skema objek Amazon S3 yang terpisah.

Anda dapat mengkonfigurasi sebuah crawler `CombineCompatibleSchemas` ke dalam definisi tabel umum bila memungkinkan. Dengan pilihan ini, crawler tersebut masih mempertimbangkan kompatibilitas data, namun mengabaikan kesamaan skema spesifik saat mengevaluasi objek Amazon S3 di penyertaan path yang ditentukan.

Jika Anda mengkonfigurasi crawler tersebut di konsol, untuk menggabungkan skema, pilih opsi crawler Buat skema tunggal untuk setiap path S3.

Saat Anda mengkonfigurasi crawler tersebut menggunakan API, atur opsi konfigurasi berikut:

- Atur bidang `Configuration` dengan representasi string dari objek JSON berikut dalam API crawler; sebagai contoh:

```
{
   "Version": 1.0,
   "Grouping": {
      "TableGroupingPolicy": "CombineCompatibleSchemas" }
}
```

Untuk membantu mengilustrasikan opsi ini, anggaplah Anda menentukan sebuah crawler dengan penyertaan path `s3://bucket/table1/`. Ketika crawler tersebut berjalan, ia menemukan dua file JSON dengan karakteristik sebagai berikut:

- File 1 — `S3://bucket/table1/year=2017/data1.json`
- Isi file — `{“A”: 1, “B”: 2}`
- Skema — `A:int, B:int`

- File 2 — `S3://bucket/table1/year=2018/data2.json`
- Isi file — `{“C”: 3, “D”: 4}`
- Skema — `C: int, D: int`

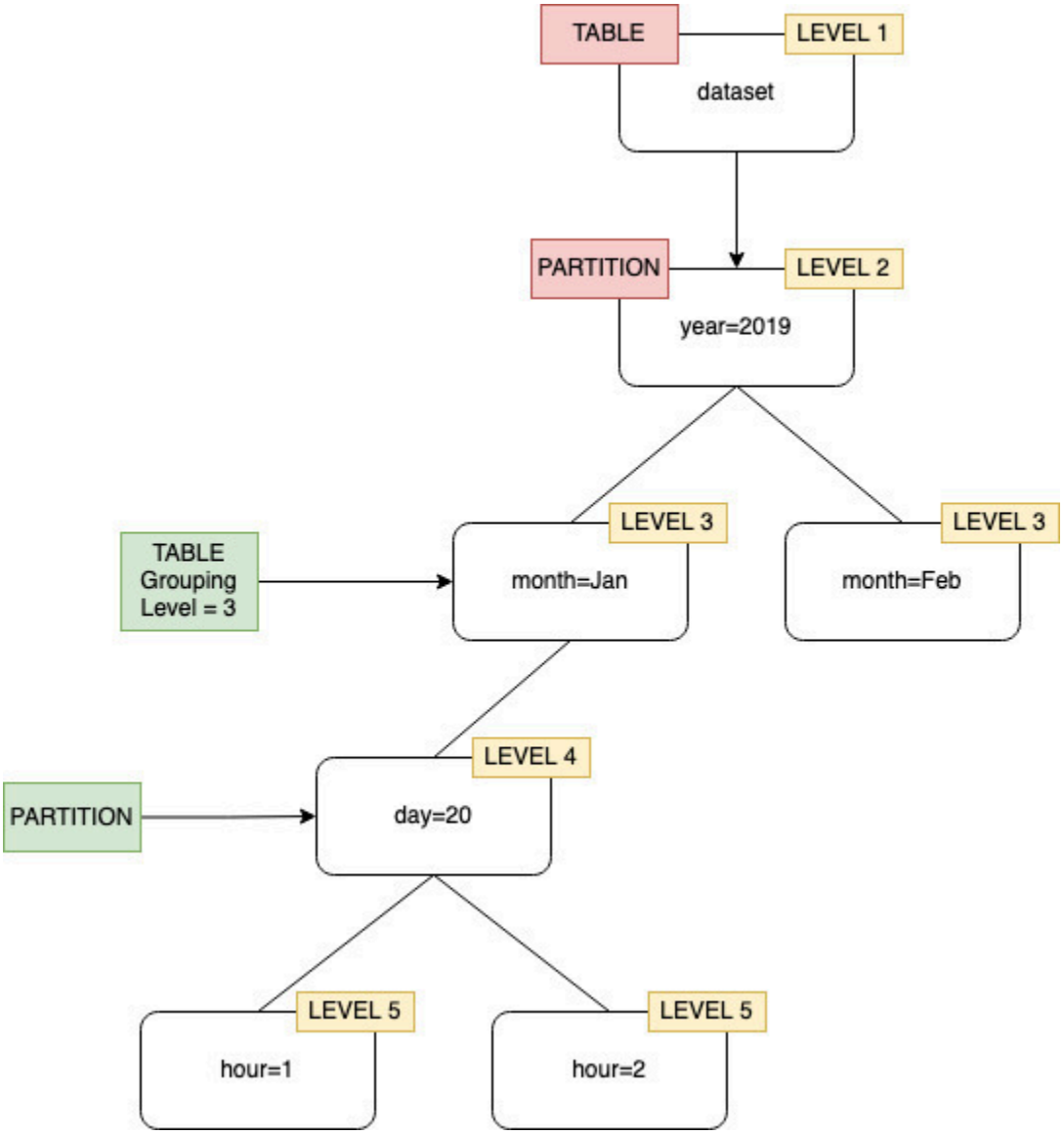
Secara default, crawler menciptakan dua tabel, bernama `year_2017` dan `year_2018` karena skema tidak cukup mirip. Namun demikian, jika pilihan Buat skema tunggal untuk setiap path S3 dipilih, dan jika data kompatibel, maka crawler akan membuat satu tabel. Tabel ini memiliki skema `A:int,B:int,C:int,D:int` dan `partitionKey year:string`.

Cara menentukan lokasi tabel dan tingkat partisi

Secara default, ketika sebuah crawler mendefinisikan tabel untuk data yang disimpan di Amazon S3, maka crawler tersebut akan mencoba menggabungkan skema bersama-sama dan membuat tabel tingkat atas (`year=2019`). Dalam beberapa kasus, Anda mungkin mengharapkan sebuah crawler

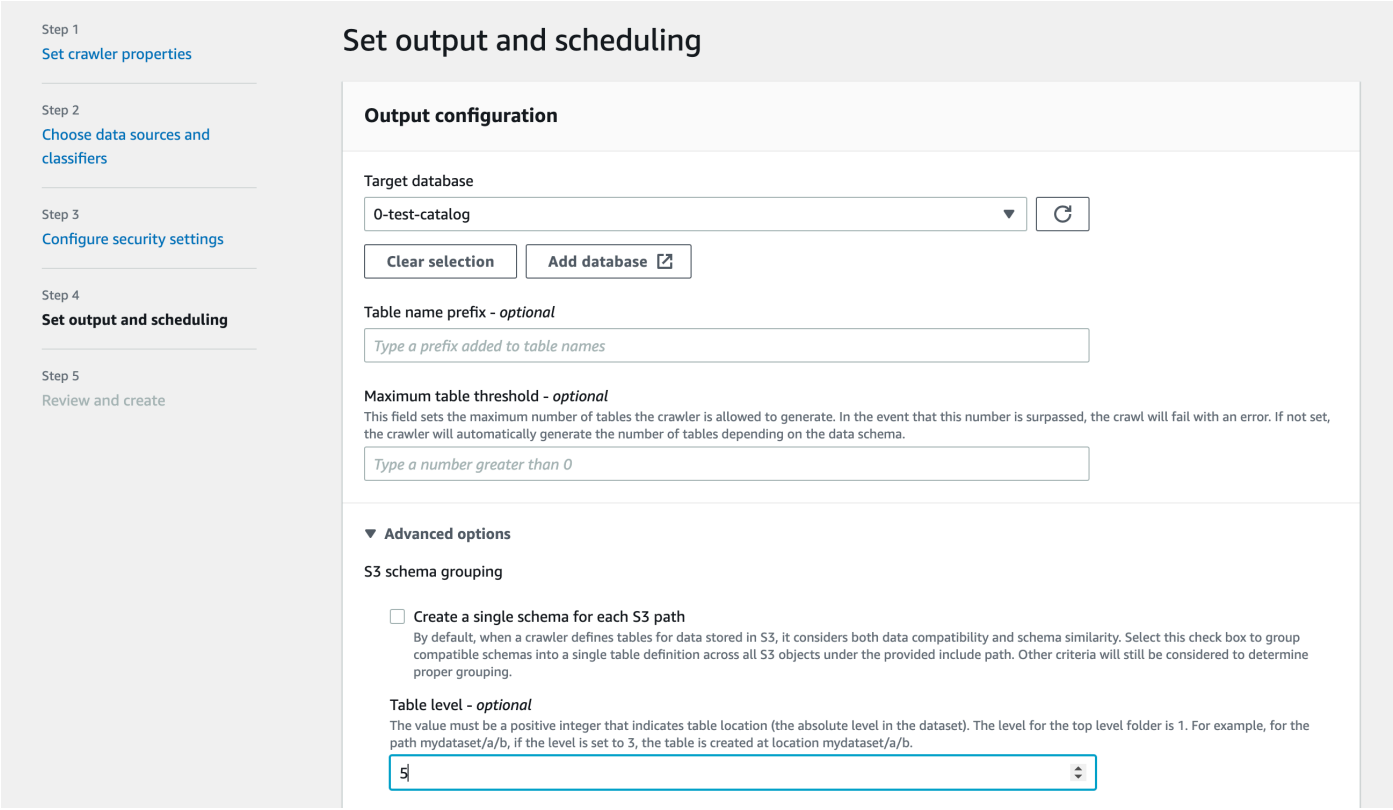
membuat tabel untuk folder `month=Jan` namun crawler membuat partisi karena folder saudara (`month=Mar`) digabung ke dalam tabel yang sama.

Opsi crawler tingkat tabel memberikan fleksibilitas untuk memberitahu crawler di mana tabel berada, dan bagaimana Anda ingin membuat partisi. Bila Anda menentukan sebuah Tingkat tabel, tabel tersebut dibuat pada tingkat absolut dari bucket Amazon S3.



Saat mengkonfigurasi crawler di konsol, Anda dapat menentukan nilai untuk opsi crawler Tingkat tabel. Nilai-nya harus bilangan bulat positif yang menunjukkan lokasi tabel (tingkat absolut dalam set data). Tingkat untuk folder tingkat atas adalah 1. Misalnya, untuk path `mydataset/year/month/day/hour`, jika tingkat diatur ke 3, maka tabel dibuat di lokasi `mydataset/year/month`.

Console



API

Saat Anda mengonfigurasi crawler menggunakan API, atur `Configuration` bidang dengan representasi string dari objek JSON berikut; misalnya:

```
configuration = jsonencode(

{
   "Version": 1.0,
   "Grouping": {
            TableLevelConfiguration = 2
        }
})
```

CloudFormation

Dalam contoh ini, Anda mengatur opsi Table level yang tersedia di konsol dalam CloudFormation template Anda:

```
"Configuration": "{
    \"Version\":1.0,
```

```
    \"Grouping\":{\"TableLevelConfiguration\":2}
}"
```

Cara menentukan jumlah maksimum tabel yang diizinkan untuk dibuat oleh crawler

Anda dapat secara opsional menentukan jumlah maksimum tabel yang diizinkan untuk dibuat oleh crawler dengan menentukan melalui konsol AWS Glue atau CLI. `TableThreshold` Jika tabel yang terdeteksi oleh crawler selama perayapan lebih besar dari nilai input ini, crawl gagal dan tidak ada data yang ditulis ke Katalog Data.

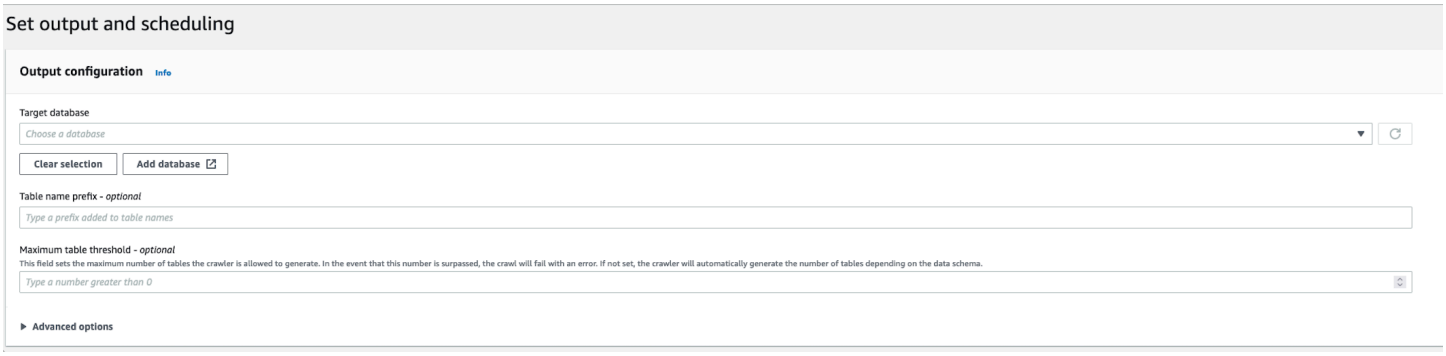
Parameter ini berguna ketika tabel yang akan dideteksi dan dibuat oleh crawler jauh lebih besar dari yang Anda harapkan. Mungkin ada beberapa alasan untuk ini, seperti:

- Saat menggunakan AWS Glue pekerjaan untuk mengisi lokasi Amazon S3 Anda, Anda dapat berakhir dengan file kosong pada tingkat yang sama dengan folder. Dalam kasus seperti itu ketika Anda menjalankan crawler di lokasi Amazon S3 ini, crawler membuat beberapa tabel karena file dan folder hadir pada tingkat yang sama.
- Jika Anda tidak mengonfigurasi, `"TableGroupingPolicy": "CombineCompatibleSchemas"` Anda mungkin berakhir dengan lebih banyak tabel dari yang diharapkan.

Anda menentukan `TableThreshold` sebagai nilai integer lebih besar dari 0. Nilai ini dikonfigurasi berdasarkan per crawler. Artinya, untuk setiap crawl nilai ini dipertimbangkan. Misalnya: crawler memiliki `TableThreshold` nilai yang ditetapkan sebagai 5. Di setiap crawl AWS Glue membandingkan jumlah tabel yang terdeteksi dengan nilai ambang tabel ini (5) dan jika jumlah tabel yang terdeteksi kurang dari 5, AWS Glue tulis tabel ke Katalog Data dan jika tidak, crawl gagal tanpa menulis ke Katalog Data.

Konsol

Untuk mengatur `TableThreshold` menggunakan AWS konsol:



CLI

Untuk mengatur `TableThreshold` menggunakan AWS CLI:

```
"{"Version":1.0,
"CrawlerOutput":
{"Tables":{"AddOrUpdateBehavior":"MergeNewColumns",
"TableThreshold":5}}}";
```

Pesan galat dicatat untuk membantu Anda mengidentifikasi jalur tabel dan membersihkan data Anda. Contoh log di akun Anda jika crawler gagal karena jumlah tabel lebih besar dari nilai ambang tabel yang disediakan:

```
Table Threshold value = 28, Tables detected - 29
```

Di CloudWatch, kami mencatat semua lokasi tabel yang terdeteksi sebagai pesan INFO. Kesalahan dicatat sebagai alasan kegagalan.

```
ERROR com.amazonaws.services.glue.customerLogs.CustomerLogService - CustomerLogService
 received CustomerFacingException with message
The number of tables detected by crawler: 29 is greater than the table threshold value
 provided: 28. Failing crawler without writing to Data Catalog.
com.amazonaws.services.glue.exceptions.CustomerFacingInternalException: The number of
 tables detected by crawler: 29 is greater than the table threshold value provided:
 28.
Failing crawler without writing to Data Catalog.
```

Cara menentukan opsi konfigurasi untuk penyimpanan data Delta Lake

Saat Anda mengonfigurasi crawler untuk penyimpanan data Delta Lake, Anda menentukan parameter konfigurasi ini:

Koneksi

Secara opsional pilih atau tambahkan koneksi Jaringan untuk digunakan dengan target Amazon S3 ini. Untuk informasi tentang koneksi, lihat Menghubungkan ke data.

Buat tabel untuk kueri

Pilih bagaimana Anda ingin membuat tabel Delta Lake:

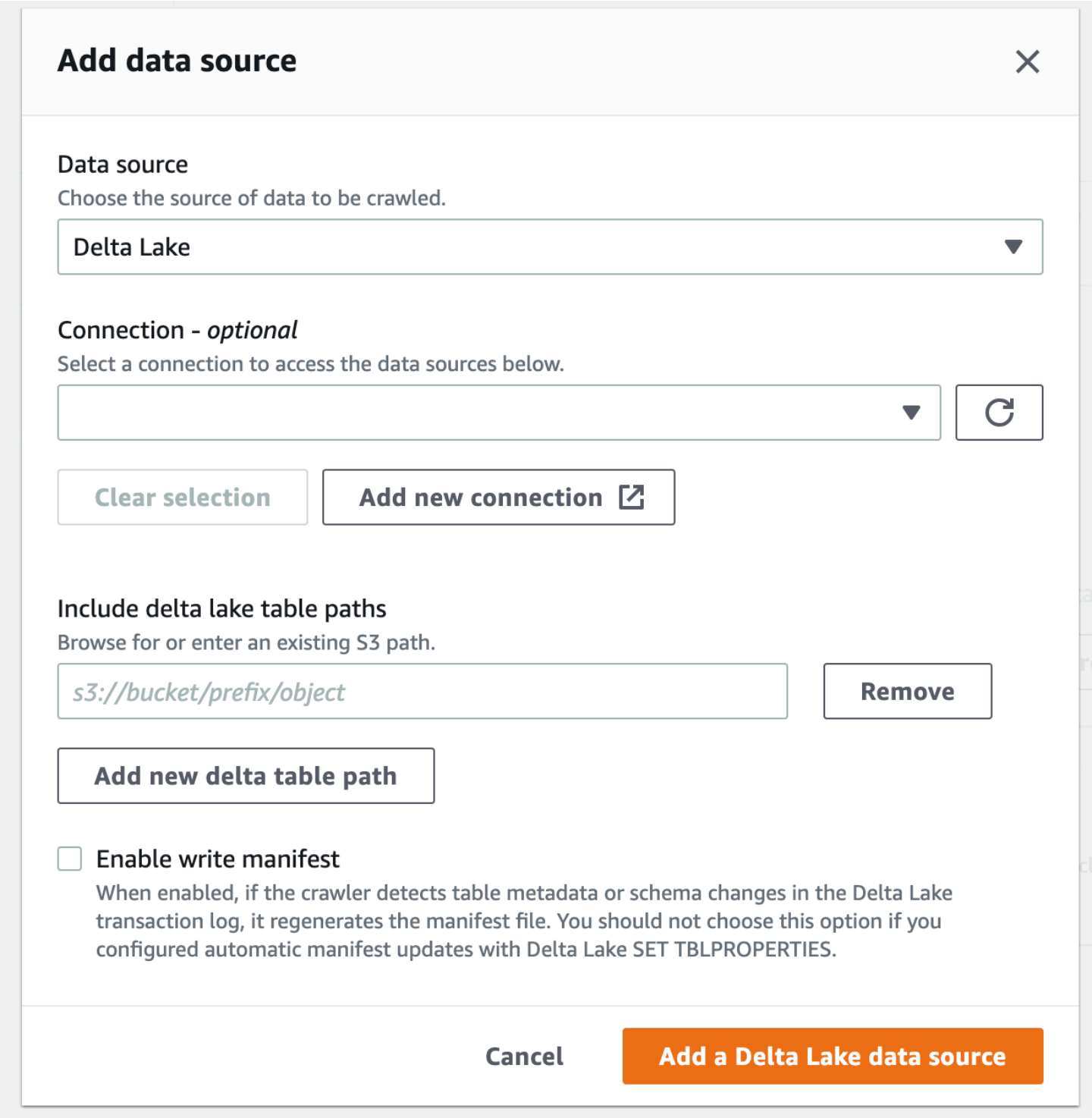
- Buat tabel Native: Izinkan integrasi dengan mesin kueri yang mendukung kueri log transaksi Delta secara langsung.
- Buat tabel Symlink: Buat folder manifes symlink dengan file manifes yang dipartisi oleh kunci partisi, berdasarkan parameter konfigurasi yang ditentukan.

Aktifkan manifes tulis (hanya dapat dikonfigurasi yang telah Anda pilih untuk Membuat tabel Symlink untuk sumber Delta Lake

Pilih apakah akan mendeteksi metadata tabel atau perubahan skema di log transaksi Delta Lake; itu meregenerasi file manifes. Anda tidak boleh memilih opsi ini jika Anda mengonfigurasi pembaruan manifes otomatis dengan Delta Lake`SET TBLPROPERTIES`.

Sertakan jalur meja danau delta

Tentukan satu atau beberapa jalur Amazon S3 ke tabel Delta sebagai s3://bucket/prefix/object.



Cara mengonfigurasi crawler untuk menggunakan kredenal Lake Formation

Anda dapat mengonfigurasi crawler untuk menggunakan AWS Lake Formation kredensional untuk mengakses penyimpanan data Amazon S3 atau tabel Katalog Data dengan lokasi Amazon S3 yang mendasarinya dalam hal yang sama atau lainnya. Akun AWS Akun AWS Anda dapat mengonfigurasi

tabel Katalog Data yang ada sebagai target crawler, jika crawler dan tabel Katalog Data berada di akun yang sama. Saat ini, hanya satu target katalog dengan satu tabel katalog yang diizinkan saat menggunakan tabel Katalog Data sebagai target cralwer.

> **Note**
>
> Saat Anda mendefinisikan tabel Katalog Data sebagai target crawler, pastikan lokasi dasar tabel Katalog Data adalah lokasi Amazon S3. Crawler yang menggunakan kredenal Lake Formation hanya mendukung target Katalog Data dengan lokasi Amazon S3 yang mendasarinya.

Penyiapan diperlukan saat crawler dan lokasi Amazon S3 terdaftar atau tabel Katalog Data berada di akun yang sama (perayapan dalam akun)

Untuk mengizinkan crawler mengakses penyimpanan data atau tabel Katalog Data dengan menggunakan kredenal Lake Formation, Anda perlu mendaftarkan lokasi data dengan Lake Formation. Selain itu, peran IAM crawler harus memiliki izin untuk membaca data dari tujuan tempat bucket Amazon S3 terdaftar.

Anda dapat menyelesaikan langkah-langkah konfigurasi berikut menggunakan AWS Management Console or AWS Command Line Interface (AWS CLI).

AWS Management Console

1. Sebelum mengonfigurasi crawler untuk mengakses sumber crawler, daftarkan lokasi data penyimpanan data atau Katalog Data dengan Lake Formation. Di konsol Lake Formation (https://console.aws.amazon.com/lakeformation/), daftarkan lokasi Amazon S3 sebagai lokasi root danau data Anda di Akun AWS tempat crawler ditentukan. Untuk informasi selengkapnya, lihat Mendaftarkan lokasi Amazon S3.
2. Berikan izin lokasi Data ke peran IAM yang digunakan untuk menjalankan crawler sehingga crawler dapat membaca data dari tujuan di Lake Formation. Untuk informasi selengkapnya, lihat Memberikan izin lokasi data (akun yang sama).
3. Berikan izin akses peran crawler (`Create`) ke database, yang ditetapkan sebagai database keluaran. Untuk informasi selengkapnya, lihat Memberikan izin database menggunakan konsol Lake Formation dan metode sumber daya bernama.
4. Di konsol IAM (https://console.aws.amazon.com/iam/), buat peran IAM untuk crawler. Tambahkan `lakeformation:GetDataAccess` kebijakan ke peran.

5. Di AWS Glue konsol (https://console.aws.amazon.com/glue/), saat mengonfigurasi crawler, pilih opsi Gunakan kredenal Lake Formation untuk merayapi sumber data Amazon S3.

> **Note**
>
> Bidang accountID bersifat opsional untuk crawling dalam akun.

AWS CLI

```
aws glue --profile demo create-crawler --debug --cli-input-json '{
    "Name": "prod-test-crawler",
    "Role": "arn:aws:iam::111122223333:role/service-role/AWSGlueServiceRole-prod-
test-run-role",
    "DatabaseName": "prod-run-db",
    "Description": "",
    "Targets": {
    "S3Targets":[
                {
                 "Path": "s3://crawl-testbucket"
                }
                ]
                },
   "SchemaChangePolicy": {
      "UpdateBehavior": "LOG",
      "DeleteBehavior": "LOG"
  },
  "RecrawlPolicy": {
    "RecrawlBehavior": "CRAWL_EVERYTHING"
  },
  "LineageConfiguration": {
    "CrawlerLineageSettings": "DISABLE"
  },
  "LakeFormationConfiguration": {
    "UseLakeFormationCredentials": true,
    "AccountId": "111122223333"
  },
  "Configuration": {
           "Version": 1.0,
           "CrawlerOutput": {
             "Partitions": { "AddOrUpdateBehavior": "InheritFromTable" },
             "Tables": {"AddOrUpdateBehavior": "MergeNewColumns" }
```

```
            },
            "Grouping": { "TableGroupingPolicy": "CombineCompatibleSchemas" }
          },
  "CrawlerSecurityConfiguration": "",
  "Tags": {
    "KeyName": ""
  }
}'
```

Penyiapan diperlukan saat crawler dan lokasi Amazon S3 terdaftar berada di akun yang berbeda (crawling lintas akun)

Untuk mengizinkan crawler mengakses penyimpanan data di akun lain menggunakan kredensil Lake Formation, Anda harus terlebih dahulu mendaftarkan lokasi data Amazon S3 dengan Lake Formation. Kemudian, Anda memberikan izin lokasi data ke akun crawler dengan mengambil langkah-langkah berikut.

Anda dapat menyelesaikan langkah-langkah berikut menggunakan AWS Management Console atau AWS CLI.

AWS Management Console

1. Di akun tempat lokasi Amazon S3 terdaftar (akun B):
   a. Daftarkan jalur Amazon S3 dengan Lake Formation. Untuk informasi selengkapnya, lihat Mendaftarkan lokasi Amazon S3.
   b. Berikan izin lokasi Data ke akun (akun A) tempat crawler akan dijalankan. Untuk informasi selengkapnya, lihat Memberikan izin lokasi data.
   c. Buat database kosong di Lake Formation dengan lokasi yang mendasarinya sebagai lokasi target Amazon S3. Untuk informasi selengkapnya, lihat Membuat database.
   d. Berikan akun A (akun tempat crawler akan dijalankan) akses ke database yang Anda buat pada langkah sebelumnya. Untuk informasi selengkapnya, lihat Memberikan izin database.
2. Di akun tempat crawler dibuat dan akan dijalankan (akun A):
   a. Menggunakan AWS RAM konsol, terima database yang dibagikan dari akun eksternal (akun B). Untuk informasi selengkapnya, lihat Menerima undangan berbagi sumber daya dari AWS Resource Access Manager.
   b. Buat peran IAM untuk crawler. Tambahkan `lakeformation:GetDataAccess` kebijakan ke peran.

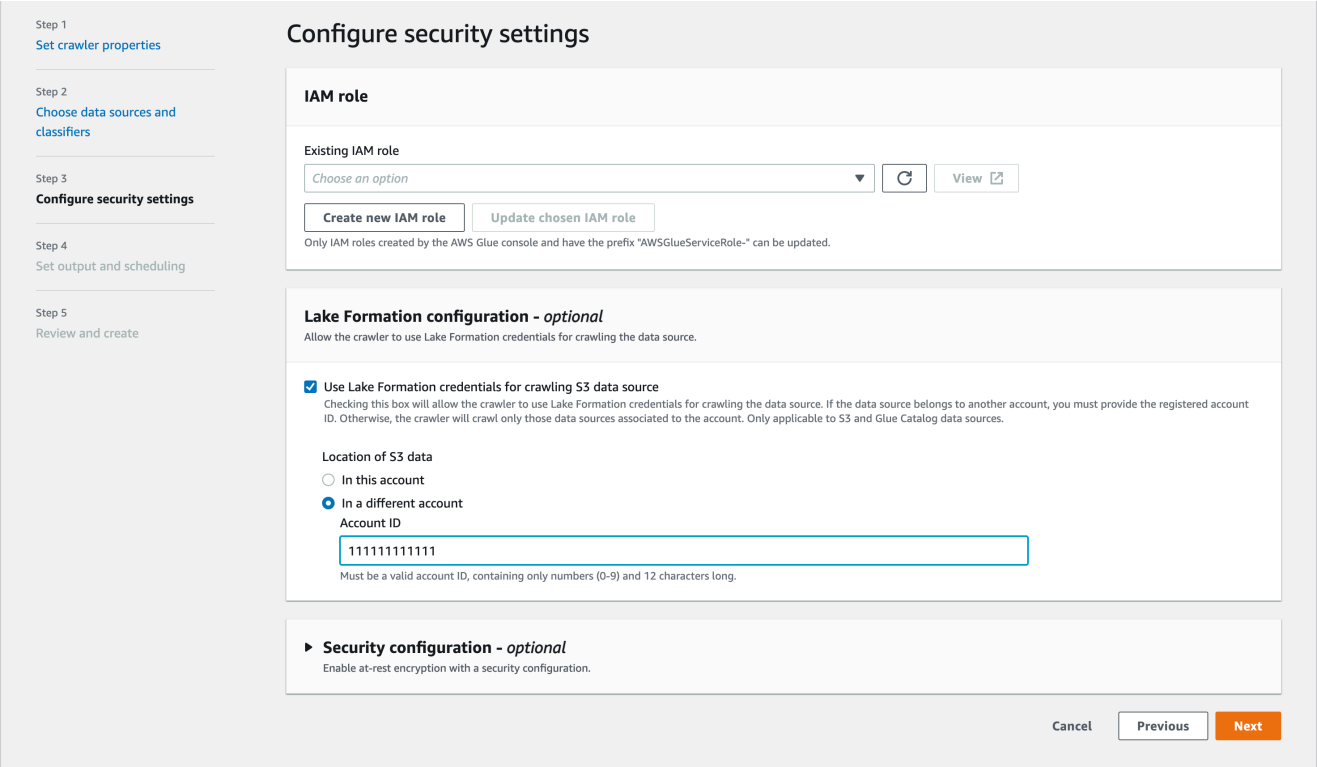
c. Di konsol Lake Formation (https://console.aws.amazon.com/lakeformation/), berikan izin lokasi Data pada lokasi Amazon S3 target ke peran IAM yang digunakan untuk crawler run sehingga crawler dapat membaca data dari tujuan di Lake Formation. Untuk informasi selengkapnya, lihat Memberikan izin lokasi data.

d. Buat tautan sumber daya pada database bersama. Untuk informasi selengkapnya, lihat Membuat tautan sumber daya.

e. Berikan izin akses peran crawler (`Create`) pada database bersama dan (`Describe`) tautan sumber daya. Tautan sumber daya ditentukan dalam output untuk crawler.

f. Di AWS Glue konsol (https://console.aws.amazon.com/glue/), saat mengonfigurasi crawler, pilih opsi Gunakan kredenal Lake Formation untuk merayapi sumber data Amazon S3.

   Untuk crawling lintas akun, tentukan Akun AWS ID tempat lokasi Amazon S3 target terdaftar di Lake Formation. Untuk perayapan dalam akun, bidang accountID bersifat opsional.



AWS CLI

```
aws glue --profile demo create-crawler --debug --cli-input-json '{
    "Name": "prod-test-crawler",
    "Role": "arn:aws:iam::111122223333:role/service-role/AWSGlueServiceRole-prod-
test-run-role",
```

```
    "DatabaseName": "prod-run-db",
    "Description": "",
    "Targets": {
    "S3Targets":[
                {
                 "Path": "s3://crawl-testbucket"
                }
                ]
                },
   "SchemaChangePolicy": {
      "UpdateBehavior": "LOG",
      "DeleteBehavior": "LOG"
  },
  "RecrawlPolicy": {
    "RecrawlBehavior": "CRAWL_EVERYTHING"
  },
  "LineageConfiguration": {
    "CrawlerLineageSettings": "DISABLE"
  },
  "LakeFormationConfiguration": {
    "UseLakeFormationCredentials": true,
    "AccountId": "111111111111"
  },
  "Configuration": {
           "Version": 1.0,
           "CrawlerOutput": {
             "Partitions": { "AddOrUpdateBehavior": "InheritFromTable" },
             "Tables": {"AddOrUpdateBehavior": "MergeNewColumns" }
           },
           "Grouping": { "TableGroupingPolicy": "CombineCompatibleSchemas" }
         },
  "CrawlerSecurityConfiguration": "",
  "Tags": {
    "KeyName": ""
  }
}'
```

Note

- Crawler yang menggunakan kredensi Lake Formation hanya didukung untuk target Amazon S3 dan Katalog Data.

- Untuk target yang menggunakan penjual kredensi Lake Formation, lokasi Amazon S3 yang mendasarinya harus termasuk dalam ember yang sama. Misalnya, pelanggan dapat menggunakan beberapa target (s3://bucket1/folder1, s3://bucket1/folder2) selama semua lokasi target berada di bawah ember yang sama (bucket1). Menentukan bucket yang berbeda (s3://bucket1/folder1, s3://bucket2/folder2) tidak diperbolehkan.
- Saat ini untuk crawler target Katalog Data, hanya satu target katalog dengan satu tabel katalog yang diizinkan.

Tutorial: Menambahkan crawler AWS Glue

Untuk skenario AWS Glue ini, Anda diminta menganalisis data datangnya maskapai penerbangan utama untuk menghitung popularitas bandara keberangkatan dari bulan ke bulan. Anda memiliki data penerbangan tahun 2016 dalam format CSV yang disimpan di Amazon S3. Sebelum Anda mengubah dan menganalisis data Anda, Anda membuat katalog metadata-nya di AWS Glue Data Catalog.

Dalam tutorial ini, mari kita tambahkan sebuah crawler yang menyimpulkan metadata dari log penerbangan ini di Amazon S3 dan menciptakan tabel dalam Katalog Data Anda.

Topik

Prasyarat

Tutorial ini mengasumsikan bahwa Anda telah memiliki akun AWS dan akses ke AWS Glue.

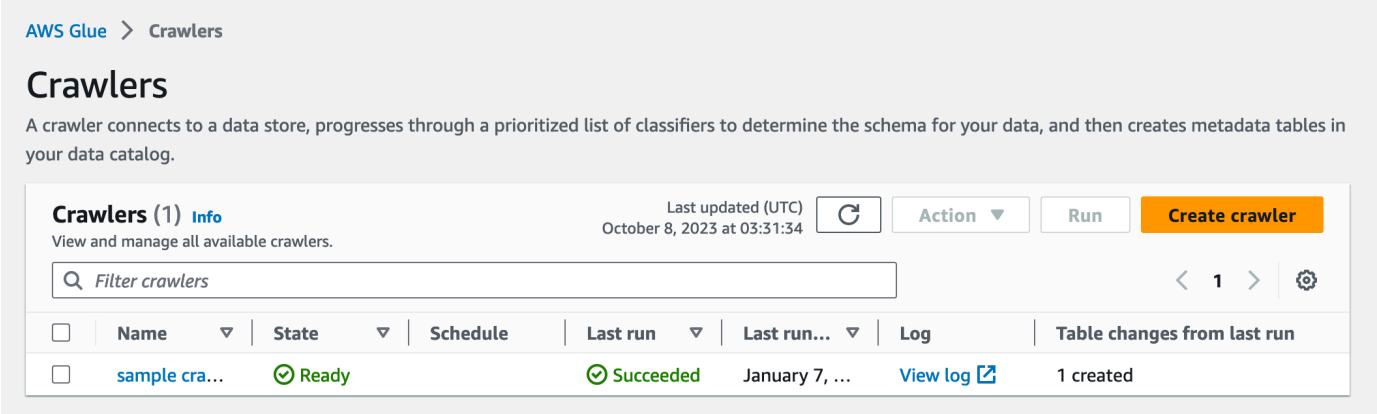
Langkah 1: Menambahkan crawler

Gunakan langkah-langkah berikut untuk mengkonfigurasi dan menjalankan sebuah crawler yang mengekstrak metadata dari file CSV yang disimpan di Amazon S3.

Untuk membuat crawler yang membaca file yang disimpan di Amazon S3

1. Pada konsol layanan AWS Glue, di menu sebelah kiri, pilih Crawler.

2. Pada halaman Crawler, pilih Create crawler. Langkah ini akan memulai serangkaian halaman yang meminta Anda untuk memberikan detail crawler.



3. Dalam kolom nama Crawler, masukkan **`Flights Data Crawler`**, dan pilih Selanjutnya.

   Crawler memanggil pengklasifikasi untuk menyimpulkan skema dari data Anda. Tutorial ini menggunakan pengklasifikasi bawaan untuk CSV secara default.

4. Untuk jenis sumber crawler, pilih Penyimpanan data dan pilih Selanjutnya.

5. Sekarang mari kita arahkan crawler tersebut ke data Anda. Pada halaman Tambahkan penyimpanan data, pilih penyimpanan data Amazon S3. Tutorial ini tidak menggunakan koneksi, jadi biarkan Koneksi kosong jika Anda melihatnya.

   Untuk opsi Lakukan crawling pada data di, pilih Path yang ditentukan di akun lain. Kemudian, untuk Sertakan path, masukkan path di mana crawler dapat menemukan data penerbangan tersebut, yaitu **`s3://crawler-public-us-east-1/flight/2016/csv`**. Setelah Anda memasukkan path-nya, judul kolom ini akan berubah menjadi Sertakan path. Pilih Selanjutnya.

6. Anda dapat meng-crawl beberapa penyimpanan data dengan satu crawler tunggal. Namun, dalam tutorial ini, kita hanya menggunakan satu penyimpanan data tunggal, jadi pilih Tidak, lalu pilih Selanjutnya.

7. Crawler tersebut membutuhkan izin untuk mengakses penyimpanan data dan membuat objek di AWS Glue Data Catalog. Untuk mengkonfigurasi izin ini, pilih Buat IAM role. Nama IAM role dimulai dengan `AWSGlueServiceRole-`, dan di kolom, Anda masukkan bagian terakhir dari nama peran tersebut. Masukkan **`CrawlerTutorial`**, lalu pilih Selanjutnya.

> **Note**
>
> Untuk membuat IAM role, pengguna AWS Anda harus memiliki izin `CreateRole`, `CreatePolicy`, dan `AttachRolePolicy`.

Penuntun menciptakan IAM role bernama `AWSGlueServiceRole-CrawlerTutorial`, melampirkan kebijakan terkelola AWS `AWSGlueServiceRole` ke peran ini, dan menambahkan kebijakan inline yang memungkinkan akses baca ke lokasi Amazon S3 `s3://crawler-public-us-east-1/flight/2016/csv`.

8. Buat satu jadwal untuk crawler. Untuk Frekuensi, pilih Eksekusi sesuai permintaan, lalu pilih Selanjutnya.
9. Crawler membuat tabel di Katalog Data Anda. Tabel terkandung dalam basis data di Katalog Data. Pertama, pilih Tambahkan basis data untuk membuat basis data. Di jendela pop-up, masukkan **`test-flights-db`** sebagai nama basis data, lalu pilih Buat.

   Selanjutnya, masukkan **`flights`** untuk Prefiks yang ditambahkan ke tabel. Gunakan nilai default untuk opsi lainnya, lalu pilih Selanjutnya.
10. Verifikasi pilihan yang Anda buat di penuntun Tambahkan crawler. Jika Anda melihat kesalahan, Anda dapat memilih Kembali untuk kembali ke halaman sebelumnya dan melakukan perubahan.

    Setelah Anda meninjau informasinya, pilih Selesai untuk membuat crawler tersebut.

Langkah 2: Jalankan crawler

Setelah membuat sebuah crawler, penuntun akan mengirimkan Anda ke halaman tampilan Crawler. Karena Anda membuat crawler dengan jadwal sesuai permintaan, maka Anda diberi opsi untuk menjalankan crawler tersebut.

Untuk menjalankan crawler

1. Banner yang ada di dekat bagian atas halaman ini memungkinkan Anda mengetahui bahwa crawler sudah dibuat, dan menanyakan apakah Anda ingin menjalankannya sekarang. Pilih Jalankan sekarang? untuk menjalankan crawler.

   Banner akan berubah untuk menampilkan pesan "Mencoba menjalankan" dan "Berjalan" untuk crawler Anda. Setelah crawler mulai berjalan, banner akan hilang, dan tampilan crawler akan diperbarui untuk menampilkan status Mulai untuk crawler Anda. Setelah satu menit, Anda dapat mengklik ikon Refresh untuk memperbarui status crawler yang ditampilkan dalam tabel tersebut.
2. Saat crawler selesai, akan muncul banner baru yang menjelaskan perubahan yang dilakukan oleh crawler. Anda dapat memilih test-flights-dblink untuk melihat objek Data Catalog.

Langkah 3: Lihat objek AWS Glue Data Catalog

Crawler membaca data di lokasi sumber dan menciptakan tabel di Katalog Data. Sebuah tabel adalah definisi metadata yang mewakili data Anda, termasuk skemanya. Tabel dalam Katalog Data tidak berisi data. Sebaliknya, Anda menggunakan tabel ini sebagai sumber atau target dalam definisi tugas.

Untuk melihat objek Katalog Data yang dibuat oleh crawler

1. Pada navigasi yang ada di sisi kiri, pada Katalog data, pilih Basis data. Di sini Anda dapat melihat basis data `flights-db` yang telah dibuat oleh crawler.
2. Pada navigasi yang ada di sisi kiri, pada Katalog data dan di bawah Basis data, pilih Tabel. Di sini Anda dapat melihat tabel `flightscsv` yang sudah dibuat oleh crawler. Jika Anda memilih nama tabel tersebut, maka Anda dapat melihat pengaturan tabel, parameter, dan properti. Gulir ke bawah dalam tampilan ini, Anda dapat melihat skema, yang merupakan informasi tentang kolom dan jenis data dari tabel tersebut.
3. Jika Anda memilih Lihat partisi pada halaman tampilan tabel, maka Anda dapat melihat partisi yang dibuat untuk data tersebut. Kolom pertama adalah kunci partisi.

Mendefinisikan metadata secara manual

Katalog AWS Glue Data adalah repositori pusat yang menyimpan metadata tentang sumber data dan kumpulan data Anda. Meskipun crawler dapat secara otomatis merayapi dan mengisi metadata untuk sumber data yang didukung, ada skenario tertentu di mana Anda mungkin perlu menentukan metadata secara manual di Katalog Data:

- Format data yang tidak didukung - Jika Anda memiliki sumber data yang tidak didukung oleh crawler, Anda perlu menentukan metadata secara manual untuk sumber data tersebut di Katalog Data.
- Persyaratan metadata khusus — Perayap AWS Glue Metadata menyimpulkan berdasarkan aturan dan konvensi yang telah ditentukan sebelumnya. Jika Anda memiliki persyaratan metadata tertentu yang tidak tercakup oleh metadata yang Perayap AWS Glue disimpulkan, Anda dapat menentukan metadata secara manual untuk memenuhi kebutuhan Anda
- Tata kelola dan standardisasi data — Dalam beberapa kasus, Anda mungkin ingin memiliki kontrol lebih besar atas definisi metadata untuk tata kelola data, kepatuhan, atau alasan keamanan. Mendefinisikan metadata secara manual memungkinkan Anda memastikan bahwa metadata mematuhi standar dan kebijakan organisasi Anda.

- Placeholder for future data ingestion — Jika Anda memiliki sumber data yang tidak segera tersedia atau dapat diakses, Anda dapat membuat tabel skema kosong sebagai placeholder. Setelah sumber data tersedia, Anda dapat mengisi tabel dengan data aktual, sambil mempertahankan struktur yang telah ditentukan.

Untuk menentukan metadata secara manual, Anda dapat menggunakan AWS Glue konsol, konsol Lake Formation, AWS Glue API, atau AWS Command Line Interface ()AWS CLI. Anda dapat membuat database, tabel, dan partisi, dan menentukan properti metadata seperti nama kolom, tipe data, deskripsi, dan atribut lainnya.

Membuat database

Basis data digunakan untuk mengatur tabel metadata di AWS Glue. Ketika Anda mendefinisikan tabel di AWS Glue Data Catalog, Anda menambahkannya ke database. Sebuah tabel dapat hanya berada dalam satu basis data.

Basis data Anda dapat berisi tabel yang mendefinisikan data dari banyak penyimpanan data yang berbeda. Data ini dapat mencakup objek yang ada di Amazon Simple Storage Service (Amazon S3) dan tabel relasional di Amazon Relational Database Service.

> **Note**
>
> Ketika Anda menghapus basis data dari Katalog Data AWS Glue, semua tabel dalam basis data juga dihapus.

Untuk melihat daftar database, masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/. Pilih Basis data, lalu pilih nama basis data pada daftar untuk melihat detail-nya.

Dari tab Basis data pada konsol AWS Glue, Anda dapat menambahkan, mengedit, dan menghapus basis data:

- Untuk membuat basis data baru, pilih Tambahkan basis data dan berikan nama dan deskripsi. Untuk kompatibilitas dengan penyimpanan metadata lainnya, seperti Apache Hive, nama dibuat menjadi karakter huruf kecil.

> **Note**
>
> Jika Anda berencana untuk mengakses basis data dari Amazon Athena, maka berikan nama dengan hanya karakter alfanumerik dan garis bawah. Untuk informasi lebih lanjut, lihat nama Athena.

- Untuk mengedit deskripsi database, pilih kotak centang di sebelah nama database dan pilih Edit.
- Untuk menghapus database, pilih kotak centang di sebelah nama database dan pilih Hapus.
- Untuk menampilkan daftar tabel yang terdapat dalam database, pilih nama database dan properti database akan menampilkan semua tabel dalam database.

Untuk mengubah basis data yang padanya crawler melakukan penulisan, Anda harus mengubah definisi crawler. Untuk informasi selengkapnya, lihat Menggunakan crawler untuk mengisi Katalog Data .

Tautan sumber daya basis data

AWS GlueKonsol baru-baru ini diperbarui. Versi konsol saat ini tidak mendukung Database Resource Links.

Katalog Data juga dapat berisi tautan sumber daya ke basis data. Tautan sumber daya basis data adalah sebuah tautan ke basis data lokal atau bersama. Saat ini, Anda dapat membuat tautan sumber daya hanya di AWS Lake Formation. Setelah Anda membuat tautan sumber daya ke sebuah basis data, Anda dapat menggunakan nama tautan sumber daya di mana pun Anda akan menggunakan nama basis data tersebut. Bersama dengan basis data yang Anda miliki atau yang dibagi dengan Anda, tautan sumber daya basis data dikembalikan oleh `glue:GetDatabases()` dan muncul sebagai entri pada halaman Basis data pada konsol AWS Glue.

Katalog Data juga dapat berisi tautan sumber daya tabel.

Untuk informasi lebih lanjut tentang tautan sumber daya, lihat Membuat Tautan Sumber Daya di Panduan Developer AWS Lake Formation .

Membuat tabel

Meskipun menjalankan crawler adalah metode yang disarankan untuk mengambil inventaris data di penyimpanan data Anda, Anda dapat menambahkan tabel metadata secara manual. AWS Glue

Data Catalog Pendekatan ini memungkinkan Anda untuk memiliki kontrol lebih besar atas definisi metadata dan menyesuaikannya sesuai dengan kebutuhan spesifik Anda.

Anda juga dapat menambahkan tabel ke Katalog Data secara manual dengan cara berikut:

- Gunakan konsol AWS Glue untuk secara manual membuat tabel di AWS Glue Data Catalog. Untuk informasi selengkapnya, lihat Bekerja dengan tabel di AWS Glue konsol.
- Gunakan operasi `CreateTable` di AWS Glue API untuk membuat tabel di AWS Glue Data Catalog. Untuk informasi selengkapnya, lihat CreateTable tindakan (Python: create_table).
- Gunakan AWS CloudFormation templat. Untuk informasi selengkapnya, lihat AWS CloudFormationuntuk AWS Glue.

Ketika Anda menentukan sebuah tabel secara manual dengan menggunakan konsol atau API, Anda menentukan skema tabel dan nilai dari sebuah bidang klasifikasi yang menunjukkan jenis dan format data dalam sumber data. Jika sebuah crawler membuat tabel, maka format data dan skemanya ditentukan oleh pengklasifikasi bawaan atau pengklasifikasi kustom. Untuk informasi lebih lanjut tentang membuat sebuah tabel menggunakan konsol AWS Glue, lihat Bekerja dengan tabel di AWS Glue konsol.

Topik

- Partisi tabel
- Tautan sumber daya tabel
- Memperbarui tabel Katalog Data yang dibuat secara manual menggunakan crawler
- Properti tabel Katalog Data
- Bekerja dengan tabel di AWS Glue konsol
- Bekerja dengan indeks partisi di AWS Glue

Partisi tabel

Sebuah definisi tabel AWS Glue dari sebuah folder Amazon Simple Storage Service (Amazon S3) dapat menggambarkan sebuah tabel yang dipartisi. Sebagai contoh, untuk meningkatkan performa kueri, sebuah tabel yang dipartisi mungkin memisahkan data bulanan ke file yang berbeda dengan menggunakan nama bulan sebagai kunci. Di AWS Glue, definisi tabel termasuk kunci partisi dari sebuah tabel. Saat AWS Glue mengevaluasi data dalam folder Amazon S3 untuk membuat katalog sebuah tabel, ia menentukan apakah tabel individu atau tabel dipartisi ditambahkan.

Anda dapat membuat indeks partisi pada sebuah tabel untuk mengambil subset dari partisi alih-alih memuat semua partisi dalam tabel. Untuk informasi tentang cara menggunakan indeks partisi, lihat Bekerja dengan indeks partisi di AWS Glue.

Semua syarat berikut harus BETUL untuk AWS Glue untuk membuat sebuah tabel dipartisi untuk folder Amazon S3:

- Skema file-nya serupa, seperti yang ditentukan oleh AWS Glue.
- Format data dari file tersebut sama.
- Format kompresi dari file tersebut sama.

Sebagai contoh, Anda mungkin memiliki sebuah bucket Amazon S3 bernama `my-app-bucket`, tempat Anda menyimpan data penjualan aplikasi iOS dan Android. Data tersebut dipartisi berdasarkan tahun, bulan, dan hari. File data untuk penjualan iOS dan Android memiliki skema yang sama, format data, dan format kompresi yang juga sama. Dalam AWS Glue Data Catalog, AWS Glue crawler membuat satu definisi tabel dengan kunci partisi untuk tahun, bulan, dan hari.

Pendaftaran Amazon S3 atas `my-app-bucket` berikut ini menunjukkan beberapa partisi. Simbol = digunakan untuk menetapkan nilai kunci partisi.

```
my-app-bucket/Sales/year=2010/month=feb/day=1/iOS.csv
my-app-bucket/Sales/year=2010/month=feb/day=1/Android.csv
my-app-bucket/Sales/year=2010/month=feb/day=2/iOS.csv
my-app-bucket/Sales/year=2010/month=feb/day=2/Android.csv
...
my-app-bucket/Sales/year=2017/month=feb/day=4/iOS.csv
my-app-bucket/Sales/year=2017/month=feb/day=4/Android.csv
```

Tautan sumber daya tabel

> AWS GlueKonsol baru-baru ini diperbarui. Versi konsol saat ini tidak mendukung Tautan Sumber Daya Tabel.

Katalog Data juga dapat berisi tautan sumber daya ke tabel. Sebuah tautan sumber daya tabel adalah tautan ke tabel lokal atau bersama. Saat ini, Anda dapat membuat tautan sumber daya hanya di AWS Lake Formation. Setelah Anda membuat tautan sumber daya ke sebuah tabel, Anda

dapat menggunakan nama tautan sumber daya di mana pun Anda akan menggunakan nama tabel tersebut. Bersama dengan tabel yang Anda miliki atau yang dibagi dengan Anda, tautan sumber daya tabel dikembalikan oleh `glue:GetTables()` dan muncul sebagai entri pada halaman Tabel pada konsol AWS Glue.

Katalog Data juga dapat berisi tautan sumber daya basis data.

Untuk informasi lebih lanjut tentang tautan sumber daya, lihat Membuat Tautan Sumber Daya di Panduan Developer AWS Lake Formation .

Memperbarui tabel Katalog Data yang dibuat secara manual menggunakan crawler

Anda mungkin ingin membuat AWS Glue Data Catalog tabel secara manual dan kemudian memperbaruinya dengan AWS Glue crawler. Crawler yang berjalan berdasarkan jadwal dapat menambahkan partisi baru dan memperbarui tabel dengan perubahan skema. Hal ini juga berlaku untuk tabel yang telah bermigrasi dari metastore Apache Hive.

Caranya, ketika Anda menentukan sebuah crawler, alih-alih menentukan satu atau beberapa penyimpanan data sebagai sumber perayapan, Anda tentukan satu atau beberapa tabel Katalog Data yang ada. Crawler tersebut kemudian melakukan crawling pada penyimpanan data yang ditentukan oleh tabel katalog. Dalam kasus ini, tidak ada tabel baru yang dibuat; sebaliknya, tabel Anda yang dibuat secara manual diperbarui.

Berikut ini adalah alasan-alasan lain mengapa Anda mungkin ingin membuat tabel katalog secara manual dan menentukan tabel katalog sebagai sumber crawler:

- Anda ingin memilih nama tabel katalog dan tidak bergantung pada algoritme penamaan tabel katalog.
- Anda ingin mencegah tabel baru dibuat dalam kasus di mana file dengan format yang dapat mengganggu deteksi partisi keliru disimpan di path sumber data.

Untuk informasi selengkapnya, lihat Langkah 2: Pilih sumber data dan pengklasifikasi.

Properti tabel Katalog Data

Properti tabel, atau parameter, seperti yang dikenal dalam AWS CLI, adalah string kunci dan nilai yang tidak divalidasi. Anda dapat mengatur properti Anda sendiri di atas tabel untuk mendukung penggunaan Katalog Data di luar AWS Glue. Layanan lain yang menggunakan Katalog Data dapat melakukannya juga. AWS Glue menetapkan beberapa properti tabel saat menjalankan pekerjaan atau crawler. Kecuali dijelaskan lain, properti ini untuk penggunaan internal, kami tidak mendukung

bahwa properti tersebut akan terus ada dalam bentuknya saat ini, atau mendukung perilaku produk jika properti ini diubah secara manual.

Untuk informasi selengkapnya tentang properti tabel yang ditetapkan oleh AWS Glue crawler, lihatthe section called “Parameter diatur pada tabel Katalog Data oleh crawler”.

Bekerja dengan tabel di AWS Glue konsol

Tabel dalam AWS Glue Data Catalog adalah definisi metadata yang mewakili data dalam penyimpanan data. Anda membuat tabel ketika Anda menjalankan sebuah crawler, atau Anda dapat membuat sebuah tabel secara manual di konsol AWS Glue . Daftar Tabel di konsol AWS Glue menampilkan nilai-nilai metadata tabel Anda. Anda menggunakan definisi tabel untuk menentukan sumber dan target ketika Anda membuat tugas ETL (ekstrak, transform, and load).

> **Note**
>
> Dengan perubahan terbaru pada konsol AWS manajemen, Anda mungkin perlu mengubah peran IAM yang ada untuk mendapatkan `SearchTables`izin. Untuk pembuatan peran baru, izin `SearchTables` API telah ditambahkan sebagai default.

Untuk memulai, masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/. Pilih tab Tabel, dan gunakan tombol Tambahkan tabel untuk membuat tabel baik dengan crawler atau dengan mengetik atribut secara manual.

Menambahkan tabel di konsol

Untuk menggunakan sebuah crawler untuk menambahkan tabel, pilih Tambahkan tabel, Tambahkan tabel menggunakan crawler. Kemudian ikuti instruksi di penuntun Tambahkan crawler. Ketika sebuah crawler berjalan, tabel ditambahkan ke AWS Glue Data Catalog. Untuk informasi selengkapnya, lihat Menggunakan crawler untuk mengisi Katalog Data .

Jika Anda tahu atribut yang diperlukan untuk membuat definisi tabel Amazon Simple Storage Service (Amazon S3) dalam Katalog Data Anda, maka Anda dapat membuatnya dengan penuntun tabel. Pilih Tambahkan tabel, Tambahkan Tabel secara manual, lalu ikuti petunjuk di penuntun Tambahkan tabel.

Saat menambahkan tabel secara manual melalui konsol, pertimbangkan hal-hal berikut:

- Jika Anda berencana untuk mengakses tabel dari Amazon Athena, maka berikan nama dengan hanya karakter alfanumerik dan garis bawah. Untuk informasi selengkapnya, lihat Nama Athena.

- Lokasi sumber data Anda harus sebuah path Amazon S3.
- Format data dari data tersebut harus sesuai dengan salah satu format yang tercantum di penuntun. Klasifikasi yang sesuai SerDe,, dan properti tabel lainnya secara otomatis diisi berdasarkan format yang dipilih. Anda dapat menentukan tabel dengan format berikut ini:

  Avro

  Format biner Apache Avro JSON.

  CSV

  Nilai dipisahkan karakter. Anda juga menentukan pembatas baik koma, pipa, titik koma, tab, atau Ctrl-A.

  JSON

  JavaScript Notasi Objek.

  XML

  Format Markup Extensible Language. Tentukan tag XML yang mendefinisikan sebuah baris dalam data. Kolom didefinisikan dalam tag baris.

  Parquet

  penyimpanan kolumnar Apache Parquet.

  ORC

  Format file Row Columnar (ORC) yang dioptimalkan. Format yang dirancang untuk menyimpan data Hive secara efisien.
- Anda dapat menentukan sebuah kunci partisi untuk tabel.
- Saat ini, tabel dipartisi yang Anda buat dengan konsol tidak dapat digunakan dalam tugas ETL.

Atribut tabel

Berikut ini adalah beberapa atribut penting dari tabel Anda:

Nama

Nama ditentukan ketika tabel dibuat, dan Anda tidak dapat mengubahnya. Anda mengacu pada nama tabel di banyak operasi AWS Glue.

Basis Data

Objek kontainer di mana tabel Anda berada. Objek ini berisi organisasi tabel Anda yang ada di dalam AWS Glue Data Catalog dan mungkin berbeda dari organisasi di penyimpanan data Anda. Ketika Anda menghapus basis data, semua tabel yang terkandung dalam basis data juga dihapus dari Katalog Data.

Deskripsi

Deskripsi tabel. Anda dapat menulis deskripsi untuk membantu Anda memahami isi tabel tersebut.

Format tabel

Tentukan membuat AWS Glue tabel standar, atau tabel dalam format Apache Iceberg.

Aktifkan pemadatan

Pilih Aktifkan pemadatan untuk memadatkan objek Amazon S3 kecil di tabel menjadi objek yang lebih besar.

Peran IAM

Untuk menjalankan pemadatan, layanan mengasumsikan peran IAM atas nama Anda. Anda dapat memilih peran IAM menggunakan drop-down. Pastikan peran memiliki izin yang diperlukan untuk mengaktifkan pemadatan.

Untuk mempelajari lebih lanjut tentang izin yang diperlukan untuk peran IAM, lihat. Prasyarat pengoptimalan tabel

Lokasi

Penunjuk ke lokasi data dalam penyimpanan data yang diwakili oleh definisi tabel ini.

Klasifikasi

Nilai kategorisasi yang diberikan saat tabel dibuat. Biasanya, nilai ini ditulis ketika crawler menjalankan dan menentukan format data sumber.

Terakhir diperbarui

Waktu dan tanggal (UTC) saat tabel ini diperbarui dalam Katalog Data.

Tanggal ditambahkan

Waktu dan tanggal (UTC) saat tabel ini ditambahkan ke Katalog Data.

Dihentikan

Jika AWS Glue menemukan bahwa tabel di Katalog Data tidak lagi ada di penyimpanan data semula, maka ia akan menandai tabel sebagai tidak lagi digunakan dalam katalog data. Jika Anda menjalankan tugas yang me-referensi tabel yang tidak lagi digunakan, tugas mungkin gagal. Edit tugas yang me-referensi tabel yang tidak lagi digunakan untuk menghapusnya sebagai sumber dan target. Kami sarankan Anda menghapus tabel yang tidak lagi digunakan ketika tidak lagi diperlukan.

Koneksi

Jika AWS Glue memerlukan koneksi ke penyimpanan data Anda, maka nama koneksi dikaitkan dengan tabel tersebut.

Melihat dan mengedit detail tabel

Untuk melihat detail tabel yang ada, pilih nama tabel dalam daftar, dan kemudian pilih Tindakan, Lihat detail.

Detail tabel termasuk properti dari tabel Anda dan skema-nya. Tampilan ini menampilkan skema tabel, termasuk nama kolom dalam urutan yang ditetapkan untuk tabel, tipe data, dan kolom kunci untuk partisi. Jika kolom merupakan tipe kolom yang kompleks, Anda dapat memilih Lihat properti untuk menampilkan detail struktur dari bidang itu, seperti yang ditunjukkan dalam contoh berikut:

```
{
"StorageDescriptor":
    {
      "cols": {
         "FieldSchema": [
           {
             "name": "primary-1",
             "type": "CHAR",
             "comment": ""
           },
           {
             "name": "second ",
             "type": "STRING",
             "comment": ""
           }
         ]
      },
```

```
      "location": "s3://aws-logs-111122223333-us-east-1",
      "inputFormat": "",
      "outputFormat": "org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat",
      "compressed": "false",
      "numBuckets": "0",
      "SerDeInfo": {
           "name": "",
           "serializationLib": "org.apache.hadoop.hive.serde2.OpenCSVSerde",
           "parameters": {
               "separatorChar": "|"
            }
      },
      "bucketCols": [],
      "sortCols": [],
      "parameters": {},
      "SkewedInfo": {},
      "storedAsSubDirectories": "false"
    },
    "parameters": {
       "classification": "csv"
    }
}
```

Untuk informasi lebih lanjut tentang properti tabel, seperti `StorageDescriptor`, lihat StorageDescriptor struktur.

Untuk mengubah skema tabel, pilih Edit skema untuk menambah dan menghapus kolom, mengubah nama kolom, dan mengubah tipe data.

Untuk membandingkan versi tabel yang berbeda, termasuk skemanya, pilih Bandingkan versi untuk melihat side-by-side perbandingan dua versi skema untuk tabel. Untuk informasi selengkapnya, lihat Bandingkan versi skema tabel .

Untuk menampilkan file yang membentuk sebuah partisi Amazon S3, pilih Lihat partisi. Untuk tabel Amazon S3, kolom Kunci menampilkan kunci partisi yang digunakan untuk partisi tabel di penyimpanan data sumber. Pemartisian adalah cara untuk membagi tabel menjadi bagian-bagian terkait berdasarkan nilai-nilai kolom kunci, seperti tanggal, lokasi, atau departemen. Untuk informasi lebih lanjut tentang partisi, cari di internet untuk informasi tentang "pemartisian hive."

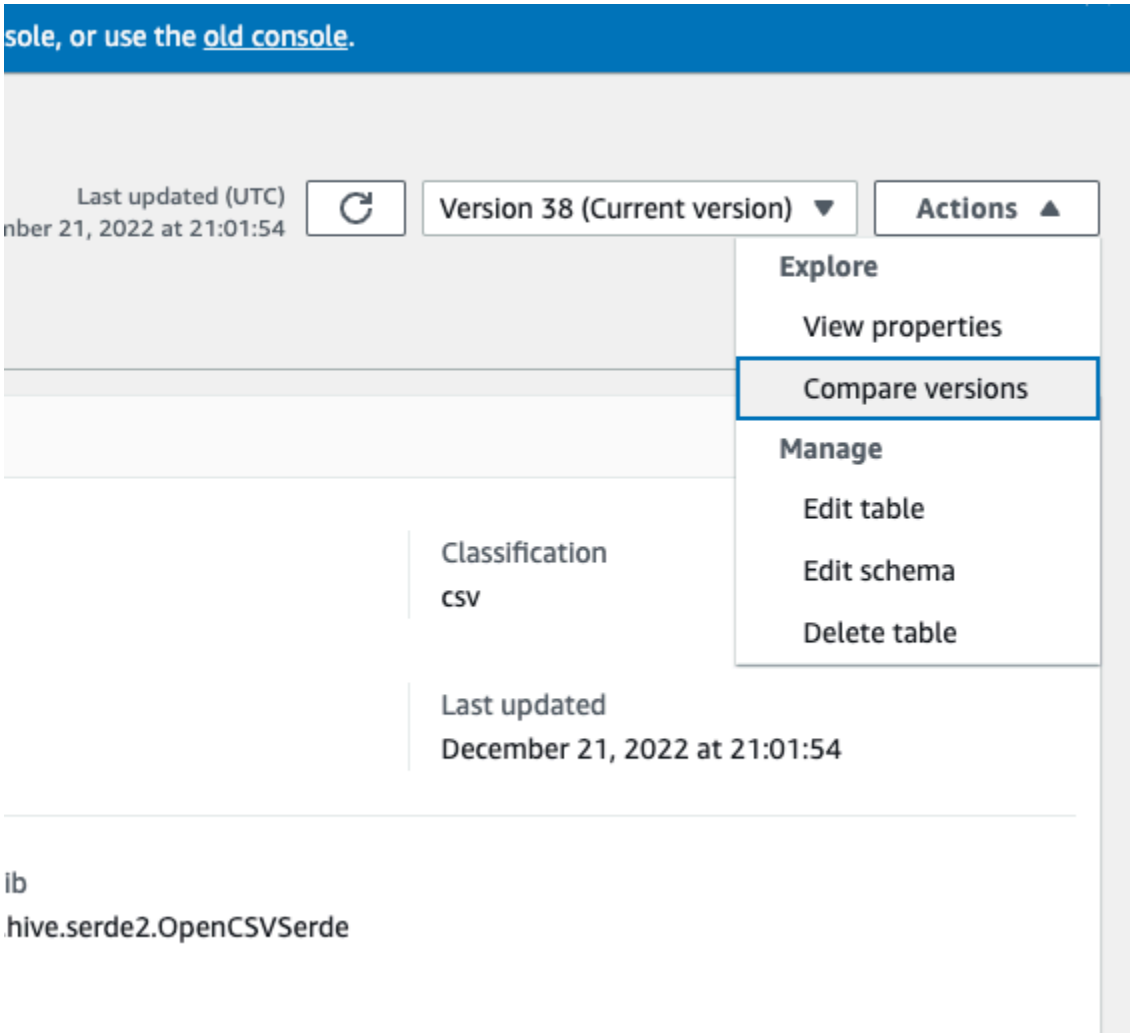
> **Note**
>
> Untuk mendapatkan step-by-step panduan untuk melihat detail tabel, lihat tutorial Jelajahi tabel di konsol.

Bandingkan versi skema tabel

Saat membandingkan dua versi skema tabel, Anda dapat membandingkan perubahan baris bersarang dengan memperluas dan menciutkan baris bersarang, membandingkan skema dua versi side-by-side, dan melihat properti tabel. side-by-side

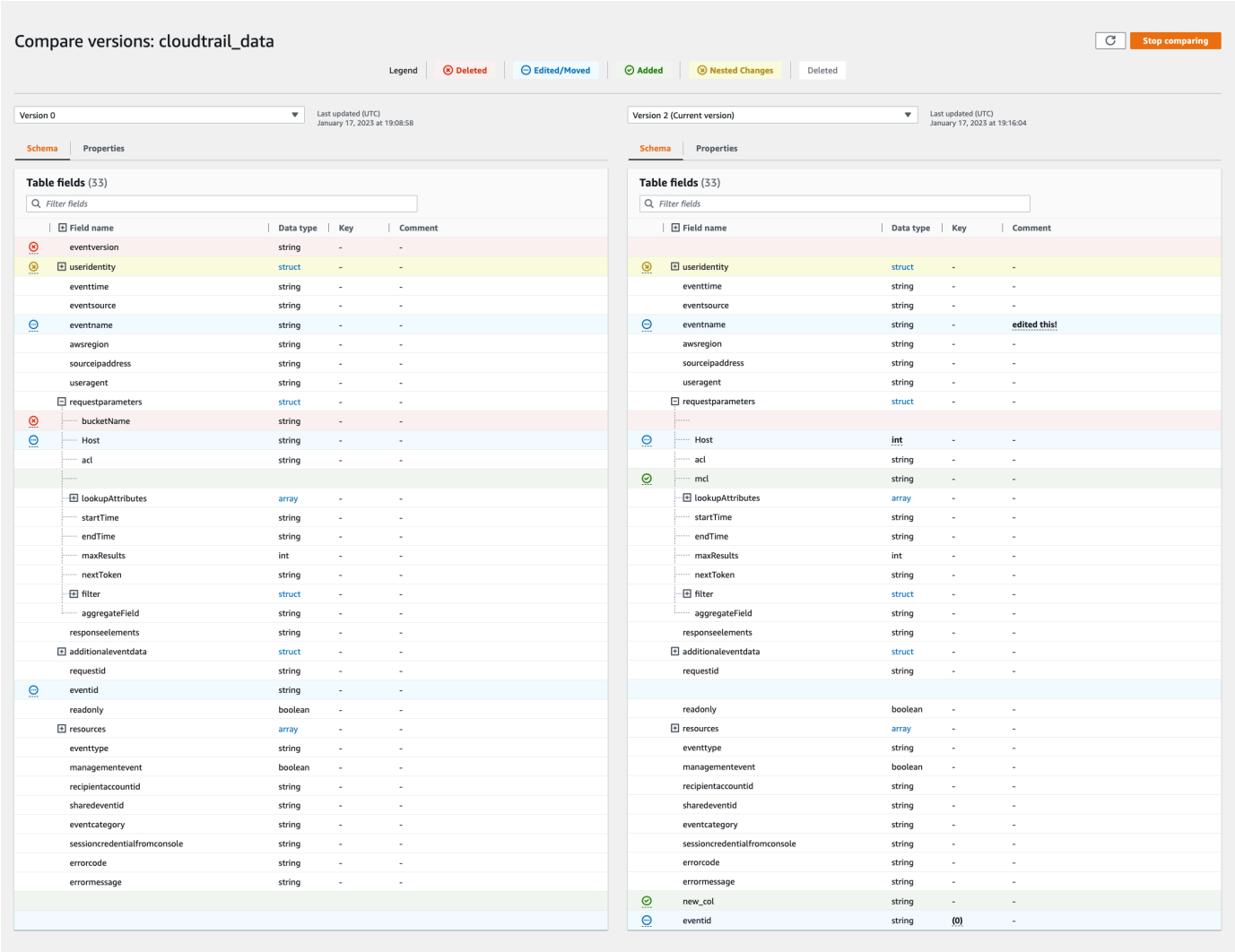
Untuk membandingkan versi

1. Dari AWS Glue konsol, pilih Tabel, lalu Tindakan dan pilih Bandingkan versi.



2. Pilih versi untuk membandingkan dengan memilih menu drop-down versi. Saat membandingkan skema, tab Skema disorot dengan warna oranye.
3. Ketika Anda membandingkan tabel antara dua versi, skema tabel disajikan kepada Anda di sisi kiri dan kanan layar. Ini memungkinkan Anda untuk menentukan perubahan secara visual dengan membandingkan nama Kolom, tipe data, kunci, dan kolom komentar side-by-side. Ketika ada perubahan, ikon berwarna menampilkan jenis perubahan yang dibuat.

   - Dihapus - ditampilkan oleh ikon merah menunjukkan di mana kolom telah dihapus dari versi sebelumnya dari skema tabel.
   - Diedit atau Dipindahkan — ditampilkan oleh ikon biru menunjukkan di mana kolom dimodifikasi atau dipindahkan dalam versi skema tabel yang lebih baru.
   - Ditambahkan - ditampilkan oleh ikon hijau menunjukkan di mana kolom ditambahkan ke versi yang lebih baru dari skema tabel.
   - Perubahan bersarang - ditampilkan oleh ikon kuning menunjukkan di mana kolom bersarang berisi perubahan. Pilih kolom untuk memperluas dan melihat kolom yang telah dihapus, diedit, dipindahkan, atau ditambahkan.



4. Gunakan bilah pencarian bidang filter untuk menampilkan bidang berdasarkan karakter yang Anda masukkan di sini. Jika Anda memasukkan nama kolom di salah satu versi tabel, bidang yang difilter akan ditampilkan di kedua versi tabel untuk menunjukkan di mana perubahan telah terjadi.
5. Untuk membandingkan properti, pilih tab Properties.
6. Untuk berhenti membandingkan versi, pilih Berhenti membandingkan untuk kembali ke daftar tabel.

Bekerja dengan indeks partisi di AWS Glue

Seiring waktu, ratusan ribu partisi ditambahkan ke sebuah tabel. GetPartitions API digunakan untuk mengambil partisi dalam tabel. API mengembalikan partisi yang cocok dengan ekspresi yang diberikan dalam permintaan.

Mari kita ambil tabel sales_data sebagai contoh yang dipartisi oleh kunci Negara, Kategori, Tahun, Bulan, dan CreationDate. Jika Anda ingin mendapatkan data penjualan untuk semua item yang dijual untuk kategori Buku pada tahun 2020 setelah 2020-08-15, Anda harus membuat `GetPartitions` permintaan dengan ungkapan “Kategori = 'Buku' dan CreationDate > '2020-08-15'” ke Katalog Data.

Jika tidak ada indeks partisi yang ada pada tabel tersebut, AWS Glue akan memuat semua partisi dari tabel, dan kemudian menyaring partisi yang dimuat dengan menggunakan ekspresi permintaan yang disediakan oleh pengguna dalam permintaan `GetPartitions` tersebut. Kueri membutuhkan lebih banyak waktu untuk dieksekusi karena jumlah partisi yang meningkat pada sebuah tabel yang tidak memiliki indeks. Dengan indeks, kueri `GetPartitions` akan mencoba untuk mengambil subset dari partisi alih-alih memuat semua partisi dalam tabel tersebut.

Topik

Tentang indeks partisi

Saat membuat sebuah indeks partisi, Anda menentukan daftar kunci partisi yang sudah ada di tabel tertentu. Indeks partisi merupakan sub daftar dari kunci partisi yang didefinisikan dalam tabel. Sebuah indeks partisi dapat dibuat pada setiap permutasi dari kunci partisi yang didefinisikan pada tabel. Untuk tabel sales_data di atas, indeks yang mungkin adalah (negara, kategori, CreationDate), (negara, kategori, tahun), (negara, kategori), (negara), (kategori, negara, tahun, bulan), dan sebagainya.

Katalog Data akan menggabungkan nilai-nilai partisi dalam urutan yang ditetapkan pada saat pembuatan indeks. Indeks dibangun secara konsisten sebagai partisi ditambahkan ke tabel tersebut. Indeks dapat dibuat untuk jenis kolom String (string, char, dan varchar), Numeric (int, bigint, long, tinyint, dan smallint), dan Date (yyyy-mm-dd).

Tipe data yang didukung

- Tanggal — Tanggal dalam format ISO, sepertiYYYY-MM-DD. Misalnya, tanggal2020-08-15. Formatnya menggunakan tanda hubung (-) untuk memisahkan tahun, bulan, dan hari. Rentang yang diizinkan untuk tanggal untuk pengindeksan mencakup dari hingga0000-01-01. 9999-12-31
- String — Sebuah string literal tertutup dalam tanda kutip tunggal atau ganda.
- Char - Data karakter panjang tetap, dengan panjang tertentu antara 1 dan 255, seperti char (10).
- Varchar — Data karakter panjang variabel, dengan panjang tertentu antara 1 dan 65535, seperti varchar (10).
- Numerik - int, bigint, panjang, tinyint, dan smallint

Indeks tipe data Numerik, String, dan Tanggal mendukung =, >, >=, <, <= dan antar operator. Solusi pengindeksan saat ini hanya mendukung operator logis AND. Sub-ekspresi dengan operator "LIKE", "IN", "OR", dan "NOT" diabaikan dalam ekspresi untuk penyaringan menggunakan sebuah indeks. Penyaringan untuk sub-ekspresi yang diabaikan dilakukan pada partisi yang diambil setelah menerapkan penyaringan indeks.

Untuk setiap partisi yang ditambahkan ke sebuah tabel, dibuat item indeks yang sesuai. Untuk tabel dengan partisi ‘n’, 1 indeks partisi akan menghasilkan item partisi indeks 'n'. Indeks partisi 'm' pada tabel yang sama akan menghasilkan item indeks partisi 'm*n'. Setiap item indeks partisi akan dikenakan biaya sesuai dengan kebijakan harga AWS Glue saat ini untuk penyimpanan katalog data. Untuk detail tentang harga objek penyimpanan, lihat Harga AWS Glue.

Membuat tabel dengan indeks partisi

Anda dapat membuat sebuah indeks partisi saat pembuatan tabel. Permintaan `CreateTable` mengambil daftar Objek `PartitionIndex` sebagai masukan. Maksimum 3 indeks partisi dapat dibuat pada tabel tertentu. Setiap indeks partisi membutuhkan sebuah nama dan sebuah daftar `partitionKeys` yang didefinisikan untuk tabel. Indeks yang dibuat pada tabel dapat diambil menggunakan API `GetPartitionIndexes`

Menambahkan indeks partisi ke tabel yang ada

Untuk menambahkan sebuah indeks partisi ke tabel yang ada, gunakan operasi `CreatePartitionIndex`. Anda dapat membuat satu `PartitionIndex` per operasi `CreatePartitionIndex`. Menambahkan sebuah indeks tidak akan mempengaruhi ketersediaan tabel, karena tabel terus akan tersedia saat indeks sedang dibuat.

Status indeks untuk partisi tambahan diatur ke MEMBUAT dan pembuatan data indeks dimulai. Jika proses untuk membuat indeks berhasil, maka IndexStatus diperbarui menjadi AKTIF dan untuk proses gagal, status indeks diperbarui menjadi GAGAL. Pembuatan indeks dapat gagal karena beberapa alasan, dan Anda dapat menggunakan operasi `GetPartitionIndexes` untuk mengambil detail kegagalan. Kegagalan yang mungkin terjadi adalah:

- ENCRYPTED_PARTITION_ERROR — Pembuatan indeks pada tabel dengan partisi terenkripsi tidak didukung.
- INVALID_PARTITION_TYPE_DATA_ERROR — Terlihat saat nilai `partitionKey` bukan nilai yang valid untuk tipe data `partitionKey` yang sesuai. Sebagai contoh: `partitionKey` dengan tipe data 'int' memiliki nilai 'foo'.
- MISSING_PARTITION_VALUE_ERROR — Terlihat bila `partitionValue` untuk `indexedKey` tidak ada. Hal ini dapat terjadi ketika sebuah tabel tidak dipartisi secara konsisten.
- UNSUPPORTED_PARTITION_CHARACTER_ERROR — Terlihat ketika nilai untuk kunci partisi diindeks berisi karakter \u0000, \u0001 atau \u0002
- INTERNAL_ERROR — Sebuah kesalahan internal terjadi saat indeks sedang dibuat.

Menjelaskan indeks partisi di atas meja

Untuk mengambil indeks partisi yang dibuat pada sebuah tabel, gunakan operasi `GetPartitionIndexes`. Respons mengembalikan semua indeks pada tabel tersebut, bersama dengan status saat ini dari masing-masing indeks (`IndexStatus`).

`IndexStatus` untuk sebuah indeks partisi merupakan salah satu dari berikut ini:

- `CREATING` — Indeks sedang dibuat, dan belum tersedia untuk digunakan.
- `ACTIVE` — Indeks siap digunakan. Permintaan dapat menggunakan indeks untuk melakukan kueri dioptimalkan.

- `DELETING` — Indeks saat ini sedang dihapus, dan tidak dapat lagi digunakan. Indeks dalam status aktif dapat dihapus menggunakan permintaan `DeletePartitionIndex`, yang memindahkan status dari AKTIF untuk MENGHAPUS.
- `FAILED` — Pembuatan indeks pada tabel yang ada gagal. Setiap tabel menyimpan 10 indeks gagal terakhir.

Transisi status yang mungkin terjadi untuk indeks yang dibuat pada tabel yang ada adalah:

- MEMBUAT → AKTIF → MENGHAPUS
- MEMBUAT → GAGAL

Keterbatasan penggunaan indeks partisi

Setelah Anda membuat sebuah indeks partisi, perhatikan perubahan ini ke fungsionalitas tabel dan partisi:

Pembuatan partisi baru (setelah Penambahan Indeks)

Setelah sebuah indeks partisi dibuat pada sebuah tabel, semua partisi baru yang ditambahkan ke tabel tersebut akan divalidasi untuk pemeriksaan tipe data untuk kunci diindeks. Nilai partisi kunci diindeks akan divalidasi format tipe data-nya. Jika pemeriksaan tipe data gagal, maka operasi membuat partisi akan gagal. Untuk tabel sales_data, jika indeks dibuat untuk kunci (kategori, tahun) di mana kategori adalah tipe `string` dan tahun merupakan jenis `int`, maka pembuatan partisi baru dengan nilai TAHUN sebagai "foo" akan gagal.

Setelah indeks diaktifkan, penambahan partisi dengan nilai kunci diindeks yang memiliki karakter U +0000, U+00001, dan U+0002 akan mulai gagal.

Pembaruan tabel

Setelah sebuah indeks partisi dibuat pada sebuah tabel, Anda tidak dapat mengubah nama kunci partisi untuk kunci partisi yang ada, dan Anda tidak dapat mengubah jenis, atau urutan, kunci yang terdaftar dengan indeks tersebut.

Menggunakan indeks untuk panggilan yang dioptimalkan GetPartitions

Ketika Anda memanggil `GetPartitions` pada sebuah tabel dengan indeks, Anda dapat menyertakan ekspresi, dan jika berlaku Katalog Data akan menggunakan sebuah indeks jika mungkin. Kunci pertama dari indeks tersebut harus diberikan dalam ekspresi untuk indeks yang akan

digunakan dalam penyaringan. Optimalisasi indeks dalam penyaringan diterapkan sebagai upaya terbaik. Katalog Data mencoba untuk menggunakan indeks optimasi sebanyak mungkin, tetapi jika indeks tidak ada, atau operator tidak didukung, maka ia kembali ke implementasi yang ada, yakni memuat semua partisi.

Untuk tabel sales_data di atas, mari kita tambahkan indeks [Negara, Kategori, Tahun]. Jika "Negara" tidak diberikan dalam ekspresi, maka indeks terdaftar tidak akan dapat mem-filter partisi menggunakan indeks. Anda dapat menambahkan hingga 3 indeks untuk men-support berbagai pola kueri.

Mari kita mengambil beberapa contoh ekspresi dan melihat bagaimana indeks bekerja pada ekspresi-ekspresi tersebut:

Ekspresi	Bagaimana indeks akan digunakan
Negara = 'ID'	Indeks akan digunakan untuk mem-filter partisi.
Negara = 'AS' dan Kategori = 'Sepatu'	Indeks akan digunakan untuk mem-filter partisi.
Kategori = 'Sepatu'	Indeks tidak akan digunakan karena "negara" tidak ada dalam ekspresi. Semua partisi akan dimuat untuk mengembalikan sebuah respon.
Negara = 'AS' dan Kategori = 'Sepatu' serta Tahun > '2018'	Indeks akan digunakan untuk mem-filter partisi.
Negara = 'AS' dan Kategori = 'Sepatu' dan Tahun > '2018' dan bulan = 2	Indeks akan digunakan untuk mengambil semua partisi dengan negara = "AS" dan kategori = "sepatu" dan tahun > 2018. Kemudian, penyaringan pada ekspresi bulan akan dilakukan.
Negara = 'AS' DAN Kategori = 'Sepatu' ATAU Tahun > '2018'	Indeks tidak akan digunakan karena sebuah operator 0R ada dalam ekspresi.
Negara = 'AS' DAN Kategori = 'Sepatu' DAN (Tahun = 2017 ATAU Tahun = '2018')	Indeks akan digunakan untuk mengambil semua partisi dengan negara = "AS" dan kategori = "sepatu", dan kemudian penyaringan pada ekspresi tahun akan dilakukan.

Ekspresi	Bagaimana indeks akan digunakan
Negara di ('AS', 'UK') DAN Kategori = 'Sepatu'	Indeks tidak akan digunakan untuk penyaringan karena operator `IN` tidak didukung saat ini.
Negara = 'AS' DAN Kategori dalam ('Sepatu', 'Buku')	Indeks akan digunakan untuk mengambil semua partisi dengan country = “US”, dan kemudian pemfilteran pada ekspresi Kategori akan dilakukan.
Negara = 'AS' DAN Kategori dalam ('Sepatu', 'Buku') DAN (CreationDate> '2023-9-01'	Indeks akan digunakan untuk mengambil semua partisi dengan country = “US”, dengan creationDate > '2023-9-01', dan kemudian pemfilteran pada ekspresi Kategori akan dilakukan.

Integrasi dengan mesin

Redshift Spectrum, Amazon EMR dan AWS Glue ETL Spark dapat memanfaatkan indeks untuk mengambil partisi setelah indeks DataFrames berada dalam status AKTIF. AWS Glue Athena dan AWS GlueETL Dynamic frame mengharuskan Anda mengikuti langkah-langkah tambahan untuk memanfaatkan indeks untuk peningkatan kueri.

Aktifkan penyaringan partisi

Untuk mengaktifkan pemfilteran partisi di Athena, Anda perlu memperbarui properti tabel sebagai berikut:

1. Di AWS Glue konsol, di bawah Katalog Data, pilih Tabel.
2. Pilih meja.
3. Di bawah Tindakan, pilih Edit tabel.
4. Di bawah properti Tabel, tambahkan yang berikut ini:
   - Kunci — `partition_filtering.enabled`
   - Nilai - `true`
5. Pilih Terapkan.

Atau, Anda dapat mengatur parameter ini dengan menjalankan kueri ALTER TABLE SET PROPERTIES di Athena.

```
ALTER TABLE partition_index.table_with_index
SET TBLPROPERTIES ('partition_filtering.enabled' = 'true')
```

Integrasi dengan layanan lain AWS

Meskipun Anda dapat menggunakan Perayap AWS Glue s untuk mengisi AWS Glue Data Catalog, ada beberapa AWS layanan yang dapat secara otomatis terintegrasi dengan dan mengisi katalog untuk Anda. Bagian berikut memberikan informasi selengkapnya tentang kasus penggunaan spesifik yang didukung oleh AWS layanan yang dapat mengisi Katalog Data.

Topik

- AWS Lake Formation
- Amazon Athena

AWS Lake Formation

AWS Lake Formation adalah layanan yang membuatnya lebih mudah untuk mengatur danau data yang aman di AWS. Lake Formation dibangun di atas AWS Glue, dan Lake Formation dan AWS Glue berbagi hal yang sama AWS Glue Data Catalog. Anda dapat mendaftarkan lokasi data Amazon S3 Anda dengan Lake Formation, dan menggunakan konsol Lake Formation untuk membuat database dan tabel di Katalog AWS Glue Data, menentukan kebijakan akses data, dan mengaudit akses data di seluruh danau data Anda dari tempat sentral. Anda dapat menggunakan kontrol akses berbutir halus Lake Formation untuk mengelola sumber daya Katalog Data dan lokasi data Amazon S3 yang ada.

Dengan data yang terdaftar di Lake Formation, Anda dapat berbagi sumber daya Katalog Data dengan aman di seluruh kepala sekolah, AWS akun, organisasi, dan unit AWS organisasi IAM.

Untuk informasi selengkapnya tentang membuat sumber daya Katalog Data menggunakan Lake Formation, lihat Membuat tabel dan database Katalog Data di Panduan AWS Lake Formation Pengembang.

Amazon Athena

Amazon Athena menggunakan Katalog Data untuk menyimpan dan mengambil metadata tabel untuk data Amazon S3 di akun Anda. AWS Metadata tabel memungkinkan mesin permintaan Athena tahu bagaimana menemukan, membaca, dan memproses data yang ingin Anda kueri.

Anda dapat mengisi AWS Glue Data Catalog dengan menggunakan pernyataan `CREATE TABLE` Athena secara langsung. Anda dapat secara manual menentukan dan mengisi skema dan metadata partisi di Katalog Data tanpa perlu menjalankan crawler.

1. Di konsol Athena, buat database yang akan menyimpan metadata tabel di Katalog Data.
2. Gunakan `CREATE EXTERNAL TABLE` pernyataan untuk menentukan skema sumber data Anda.
3. Gunakan `PARTITIONED BY` klausa untuk menentukan kunci partisi apa pun jika data Anda dipartisi.
4. Gunakan `LOCATION` klausa untuk menentukan jalur Amazon S3 tempat file data aktual Anda disimpan.
5. Jalankan pernyataan `CREATE TABLE`.

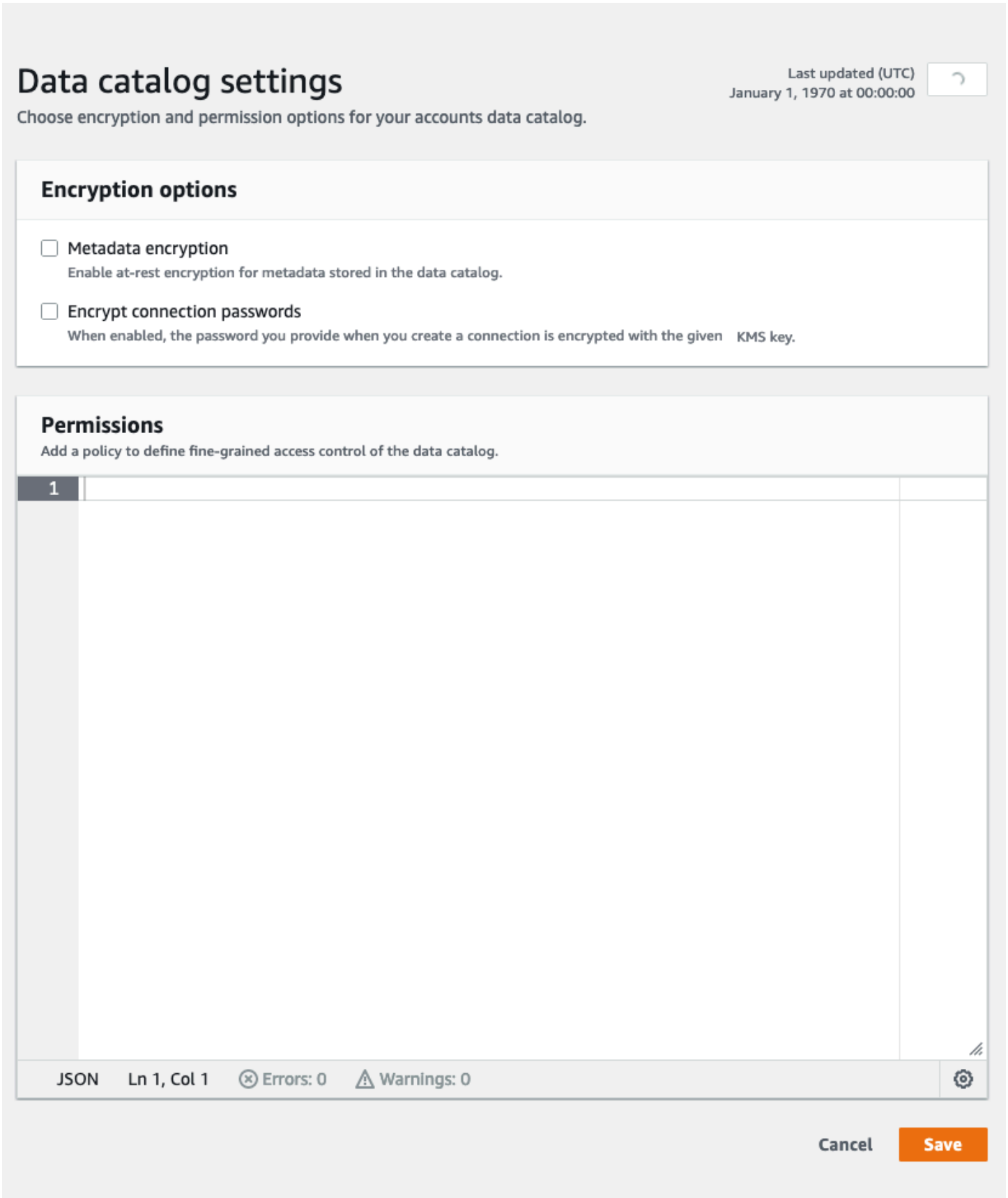
   Kueri ini membuat metadata tabel di Katalog Data berdasarkan skema dan partisi yang Anda tentukan, tanpa benar-benar merayapi data.

Anda dapat menanyakan tabel di Athena, dan itu akan menggunakan metadata dari Katalog Data untuk mengakses dan menanyakan file data Anda di Amazon S3.

Untuk informasi selengkapnya, lihat Membuat database dan tabel di Panduan Pengguna Amazon Athena.

Pengaturan Katalog Data

Pengaturan Katalog Data berisi opsi untuk mengatur opsi enkripsi dan izin untuk Katalog Data di akun Anda.


Data catalog settings
Choose encryption and permission options for your accounts data catalog.
Last updated (UTC)
January 1, 1970 at 00:00:00
Encryption options
Metadata encryption
Enable at-rest encryption for metadata stored in the data catalog.
Encrypt connection passwords
When enabled, the password you provide when you create a connection is encrypted with the given KMS key.
Permissions
Add a policy to define fine-grained access control of the data catalog.
1
JSON
Ln 1, Col 1
Errors: 0
Warnings: 0
Cancel
Save

Untuk mengubah kendali izin akses detail Katalog Data

1. Masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/.
2. Pilih opsi enkripsi.

   - Enkripsi metadata — Pilih kotak centang ini untuk mengenkripsi metadata di Katalog Data Anda. Metadata dienkripsi saat istirahat menggunakan kunci AWS Key Management Service (AWS KMS) yang Anda tentukan. Untuk informasi selengkapnya, lihat Mengenkripsi Katalog Data Anda.
   - Enkripsi kata sandi koneksi - Pilih kotak centang ini untuk mengenkripsi kata sandi di objek AWS Glue koneksi saat koneksi dibuat atau diperbarui. Kata sandi dienkripsi menggunakan AWS KMS kunci yang Anda tentukan. Ketika kata sandi dikembalikan, kata sandi tersebut akan dienkripsi. Pilihan ini adalah pengaturan global untuk semua koneksi AWS Glue dalam Katalog Data. Jika Anda mengosongkan kotak centang ini, kata sandi yang dienkripsi sebelumnya tetap dienkripsi dengan menggunakan kunci yang digunakan saat kata sandi itu dibuat atau diperbarui. Untuk informasi selengkapnya tentang koneksi AWS Glue, lihat Menghubungkan ke data.

     Saat Anda mengaktifkan opsi ini, pilih AWS KMS kunci, atau pilih Masukkan ARN kunci dan berikan Nama Sumber Daya Amazon (ARN) untuk kunci tersebut. Masukkan ARN dalam bentuk `arn:aws:kms:region:account-id:key/key-id`. Anda juga dapat memberikan ARN sebagai alias kunci, seperti `arn:aws:kms:region:account-id:alias/alias-name`.

     > **Important**
     >
     > Jika opsi ini dipilih, maka setiap pengguna atau peran yang membuat atau memperbarui koneksi harus memiliki izin `kms:Encrypt` pada kunci KMS yang ditentukan.

     Untuk informasi selengkapnya, lihat Mengenkripsi kata sandi koneksi.
3. Pilih Pengaturan, dan kemudian di editor Izin, tambahkan pernyataan kebijakan untuk mengubah kendali akses detail Katalog Data untuk akun Anda. Hanya satu kebijakan yang dapat dilampirkan ke Katalog Data pada satu waktu. Anda dapat menempelkan kebijakan sumber daya

JSON ke kendali ini. Untuk informasi selengkapnya, lihat Kebijakan berbasis sumber daya dalam Glue AWS.

4. Pilih Simpan untuk memperbarui Katalog Data Anda dengan perubahan yang Anda buat.

Anda juga dapat menggunakan operasi API AWS Glue untuk menempatkan, mendapatkan, dan menghapus kebijakan sumber daya. Untuk informasi selengkapnya, lihat API Keamanan di AWS Glue.

Mengisi dan mengelola tabel transaksional

Apache Iceberg, Apache Hudi, dan Linux Foundation Delta Lake adalah format tabel open-source yang dirancang untuk menangani analisis data skala besar dan beban kerja data lake di Apache Spark.

Anda dapat mengisi tabel Iceberg, Hudi, dan Delta Lake dengan menggunakan metode berikut: AWS Glue Data Catalog

- Perayap AWS Glue; — Perayap AWS Glue s dapat secara otomatis menemukan dan mengisi metadata tabel Iceberg, Hudi dan Delta Lake di Katalog Data. Untuk informasi selengkapnya, lihat Menggunakan crawler untuk mengisi Katalog Data .
- AWS Glue Pekerjaan ETL — Anda dapat membuat pekerjaan ETL untuk menulis data ke tabel Iceberg, Hudi, dan Delta Lake dan mengisi metadata mereka di Katalog Data. Untuk informasi selengkapnya, lihat Menggunakan kerangka kerja data lake dengan pekerjaan AWS Glue ETL.
- AWS Glue konsol, AWS Lake Formation konsol, AWS CLI atau API — Anda dapat menggunakan AWS Glue konsol, konsol Lake Formation, atau API untuk membuat dan mengelola definisi tabel Iceberg di Katalog Data.

Topik

Membuat tabel Apache Iceberg

Anda dapat membuat tabel Apache Iceberg yang menggunakan format data Apache Parquet di AWS Glue Data Catalog dengan data yang berada di Amazon S3. Tabel dalam Katalog Data adalah

definisi metadata yang mewakili data dalam penyimpanan data. Secara default, AWS Glue membuat tabel Iceberg v2. Untuk perbedaan antara tabel v1 dan v2, lihat Format perubahan versi dalam dokumentasi Apache Iceberg.

Apache Iceberg adalah format tabel terbuka untuk dataset analitik yang sangat besar. Iceberg memungkinkan perubahan mudah pada skema Anda, juga dikenal sebagai evolusi skema, yang berarti bahwa pengguna dapat menambahkan, mengganti nama, atau menghapus kolom dari tabel data tanpa mengganggu data yang mendasarinya. Iceberg juga menyediakan dukungan untuk pembuatan versi data, yang memungkinkan pengguna untuk melacak perubahan data dari waktu ke waktu. Ini memungkinkan fitur perjalanan waktu, yang memungkinkan pengguna untuk mengakses dan menanyakan versi historis data dan menganalisis perubahan data antara pembaruan dan penghapusan.

Anda dapat menggunakan AWS Glue atau konsol Lake Formation atau `CreateTable` operasi di AWS Glue API untuk membuat tabel Gunung Es di Katalog Data. Untuk informasi selengkapnya, lihat CreateTable tindakan (Python: create_table).

Saat Anda membuat tabel Gunung Es di Katalog Data, Anda harus menentukan format tabel dan jalur file metadata di Amazon S3 agar dapat melakukan pembacaan dan penulisan.

Anda dapat menggunakan Lake Formation untuk mengamankan tabel Gunung Es menggunakan izin kontrol akses berbutir halus saat Anda mendaftarkan lokasi data Amazon S3. AWS Lake Formation Untuk data sumber di Amazon S3 dan metadata yang tidak terdaftar di Lake Formation, akses ditentukan oleh kebijakan izin IAM untuk Amazon S3 dan tindakan. AWS Glue Untuk informasi selengkapnya, lihat Mengelola izin.

> **Note**
>
> Data Catalog tidak mendukung pembuatan partisi dan menambahkan properti tabel Iceberg.

Prasyarat

Untuk membuat tabel Gunung Es di Katalog Data, dan mengatur izin akses data Lake Formation, Anda harus melengkapi persyaratan berikut:

1. Izin diperlukan untuk membuat tabel Gunung Es tanpa data yang terdaftar di Lake Formation.

   Selain izin yang diperlukan untuk membuat tabel di Katalog Data, pembuat tabel memerlukan izin berikut:

- `s3:PutObject`pada sumber daya arn:aws:s3::: {bucketName}
- `s3:GetObject`pada sumber daya arn:aws:s3::: {bucketName}
- `s3:DeleteObject`pada sumber daya arn:aws:s3::: {bucketName}

2. Izin yang diperlukan untuk membuat tabel Gunung Es dengan data yang terdaftar di Lake Formation:

   Untuk menggunakan Lake Formation untuk mengelola dan mengamankan data di danau data Anda, daftarkan lokasi Amazon S3 Anda yang memiliki data untuk tabel dengan Lake Formation. Ini agar Lake Formation dapat menjual kredensyal ke layanan AWS analitis seperti Athena, Redshift Spectrum, dan Amazon EMR untuk mengakses data. Untuk informasi selengkapnya tentang mendaftarkan lokasi Amazon S3, lihat Menambahkan lokasi Amazon S3 ke data lake Anda.

   Kepala sekolah yang membaca dan menulis data dasar yang terdaftar di Lake Formation memerlukan izin berikut:

   - `lakeformation:GetDataAccess`
   - `DATA_LOCATION_ACCESS`

     Kepala sekolah yang memiliki izin lokasi data di lokasi juga memiliki izin lokasi di semua lokasi anak.

     Untuk informasi selengkapnya tentang izin lokasi data, lihat Ulink kontrol akses data yang mendasari.

Untuk mengaktifkan pemadatan, layanan perlu mengambil peran IAM yang memiliki izin untuk memperbarui tabel di Katalog Data. Untuk detailnya, lihat Prasyarat pengoptimalan tabel

Membuat tabel Iceberg

Anda dapat membuat tabel Iceberg v1 dan v2 menggunakan atau konsol Lake AWS Glue Formation atau AWS Command Line Interface seperti yang didokumentasikan di halaman ini. Anda juga dapat membuat tabel Iceberg menggunakan. Perayap AWS Glue Untuk informasi selengkapnya, lihat Katalog Data dan Crawler di Panduan AWS Glue Pengembang.

Untuk membuat tabel Iceberg

Console

1. Masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/.
2. Di bawah Katalog Data, pilih Tabel, dan gunakan tombol Buat tabel untuk menentukan atribut berikut:
   - Nama tabel - Masukkan nama untuk tabel. Jika Anda menggunakan Athena untuk mengakses tabel, gunakan tips penamaan ini di Panduan Pengguna Amazon Athena.
   - Database — Pilih database yang ada atau buat yang baru.
   - Deskripsi — Deskripsi tabel. Anda dapat menulis deskripsi untuk membantu Anda memahami isi tabel tersebut.
   - Format tabel - Untuk format Tabel, pilih Apache Iceberg.
   - Aktifkan pemadatan — Pilih Aktifkan pemadatan untuk memadatkan objek Amazon S3 kecil dalam tabel menjadi objek yang lebih besar.
   - Peran IAM — Untuk menjalankan pemadatan, layanan mengasumsikan peran IAM atas nama Anda. Anda dapat memilih peran IAM menggunakan drop-down. Pastikan peran memiliki izin yang diperlukan untuk mengaktifkan pemadatan.

     Untuk mempelajari lebih lanjut tentang izin yang diperlukan, lihat Prasyarat pengoptimalan tabel .
   - Lokasi - Tentukan jalur ke folder di Amazon S3 yang menyimpan tabel metadata. Iceberg membutuhkan file metadata dan lokasi di Katalog Data untuk dapat melakukan pembacaan dan penulisan.
   - Skema - Pilih Tambahkan kolom untuk menambahkan kolom dan tipe data kolom. Anda memiliki opsi untuk membuat tabel kosong dan memperbarui skema nanti. Katalog Data mendukung tipe data Hive. Untuk informasi selengkapnya, lihat Tipe data sarang.

     Iceberg memungkinkan Anda untuk mengembangkan skema dan partisi setelah Anda membuat tabel. Anda dapat menggunakan kueri Athena untuk memperbarui skema tabel dan kueri Spark untuk memperbarui partisi.

AWS CLI

```
aws glue create-table \
    --database-name iceberg-db \
    --region us-west-2 \
```

```
--open-table-format-input '{
  "IcebergInput": {
       "MetadataOperation": "CREATE",
       "Version": "2"
     }
  }' \
--table-input '{"Name":"test-iceberg-input-demo",
        "TableType": "EXTERNAL_TABLE",
        "StorageDescriptor":{
           "Columns":[
               {"Name":"col1", "Type":"int"},
               {"Name":"col2", "Type":"int"},
               {"Name":"col3", "Type":"string"}
            ],
           "Location":"s3://DOC_EXAMPLE_BUCKET_ICEBERG/"
        }
    }'
```

Mengoptimalkan tabel Iceberg

Danau data Amazon S3 menggunakan format tabel terbuka seperti Apache Iceberg menyimpan data sebagai objek Amazon S3. Memiliki ribuan objek Amazon S3 kecil dalam tabel data lake meningkatkan overhead metadata pada tabel Iceberg dan memengaruhi kinerja baca. Untuk kinerja pembacaan yang lebih baik oleh layanan AWS analitik seperti Amazon Athena dan Amazon EMR, dan pekerjaan AWS Glue ETL, AWS Glue Data Catalog menyediakan pemadatan terkelola (proses yang memadatkan objek Amazon S3 kecil menjadi objek yang lebih besar) untuk tabel Iceberg di Katalog Data. Anda dapat menggunakan konsol, AWS Glue konsol AWS CLI, atau AWS API Lake Formation untuk mengaktifkan atau menonaktifkan pemadatan untuk tabel Iceberg individual yang ada di Katalog Data.

Pengoptimal tabel terus memantau partisi tabel dan memulai proses pemadatan ketika ambang batas terlampaui untuk jumlah file dan ukuran file. Tabel Iceberg memenuhi syarat untuk pemadatan jika ukuran file ditentukan dalam penulisan. target-file-size-bytes properti berada dalam kisaran 128MB hingga 512MB. Dalam Katalog Data, proses pemadatan dimulai jika tabel memiliki lebih dari lima file, masing-masing lebih kecil dari 75% penulisan. target-file-size-bytes properti.

Misalnya, Anda memiliki tabel dengan ambang ukuran file yang disetel ke 512MB dalam penulisan. target-file-size-bytes properti (dalam kisaran 128MB hingga 512MB yang ditentukan), dan tabel berisi

10 file. Jika 6 dari 10 file masing-masing kurang dari 384MB (.75* 512), maka Katalog Data memicu pemadatan.

Katalog Data melakukan pemadatan tanpa mengganggu kueri bersamaan. Data Catalog mendukung pemadatan data hanya untuk tabel dalam format Parket.

Untuk tipe data yang didukung, format kompresi, dan batasan, lihatFormat dan batasan yang didukung untuk pemadatan data terkelola .

Topik

Prasyarat pengoptimalan tabel

Pengoptimal tabel mengasumsikan izin peran AWS Identity and Access Management (IAM) yang Anda tentukan saat Anda mengaktifkan pemadatan untuk tabel. Peran IAM harus memiliki izin untuk membaca data dan memperbarui metadata di Katalog Data. Anda dapat membuat peran IAM dan melampirkan kebijakan inline berikut:

- Tambahkan kebijakan sebaris berikut yang memberikan izin baca/tulis Amazon S3 di lokasi untuk data yang tidak terdaftar di Lake Formation. Kebijakan ini juga mencakup izin untuk memperbarui tabel di Katalog Data, dan AWS Glue mengizinkan menambahkan log di Amazon CloudWatch log dan metrik publikasi. Untuk data sumber di Amazon S3 yang tidak terdaftar di Lake Formation, akses ditentukan oleh kebijakan izin IAM untuk Amazon S3 dan tindakan. AWS Glue

  Dalam kebijakan inline berikut, ganti `bucket-name` dengan nama bucket Amazon S3 Anda`aws-account-id`, `region` dan dengan nomor akun dan Wilayah Katalog Data yang AWS valid`database_name`, dengan nama database Anda, `table_name` dan dengan nama tabel.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket-name>/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket-name>"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:UpdateTable",
                "glue:GetTable"
            ],
            "Resource": [
                "arn:aws:glue:<region>:<aws-account-id>:table/<database-name>/<table-
name>",
                "arn:aws:glue:<region>:<aws-account-id>:database/<database-name>",
                "arn:aws:glue:<region>:<aws-account-id>:catalog"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:<region>:<aws-account-id>:log-group:/aws-glue/
iceberg-compaction/logs:*"
        }
```

```
    ]
}
```

- Gunakan kebijakan berikut untuk mengaktifkan pemadatan data yang terdaftar di Lake Formation.

  Jika peran pemadatan tidak memiliki izin `IAM_ALLOWED_PRINCIPALS` grup yang diberikan pada tabel, peran tersebut memerlukan Lake Formation `ALTER`, `DESCRIBE`, `INSERT` dan `DELETE` izin di atas tabel.

  Untuk informasi selengkapnya tentang mendaftarkan bucket Amazon S3 dengan Lake Formation, lihat Menambahkan lokasi Amazon S3 ke data lake Anda.

```
{
 "Version": "2012-10-17",
 "Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "lakeformation:GetDataAccess"
        ],
        "Resource": "*"
    },
    {
        "Effect": "Allow",
        "Action": [
        "glue:UpdateTable",
        "glue:GetTable"
        ],

        "Resource": [
            "arn:aws:glue:<region>:<aws-account-
id>:table/<databaseName>/<tableName>",
            "arn:aws:glue:<region>:<aws-account-id>:database/<database-name>",
            "arn:aws:glue:<region>:<aws-account-id>:catalog"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogGroup",
            "logs:CreateLogStream",
            "logs:PutLogEvents"
        ],
```

```
        "Resource": "arn:aws:logs:<region>:<aws-account-id>:log-group:/aws-glue/
iceberg-compaction/logs:*"
        }
    ]
 }
```

- (Opsional) Untuk memadatkan tabel Iceberg dengan data di bucket Amazon S3 yang dienkripsi menggunakan enkripsi sisi Server, peran pemadatan memerlukan izin untuk mendekripsi objek Amazon S3 dan menghasilkan kunci data baru untuk menulis objek ke bucket terenkripsi. Tambahkan kebijakan berikut ke AWS KMS kunci yang diinginkan. Kami hanya mendukung enkripsi tingkat ember.

```
{
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::<aws-account-id>:role/<compaction-role-name>"
    },
    "Action": [
        "kms:Decrypt",
        "kms:GenerateDataKey"
    ],
    "Resource": "*"
}
```

- (Opsional) Untuk lokasi data yang terdaftar dengan Lake Formation, peran yang digunakan untuk mendaftarkan lokasi memerlukan izin untuk mendekripsi objek Amazon S3 dan menghasilkan kunci data baru untuk menulis objek ke bucket terenkripsi. Untuk informasi selengkapnya, lihat Mendaftarkan lokasi Amazon S3 terenkripsi.
- (Opsional) Jika AWS KMS kunci disimpan di AWS akun yang berbeda, Anda harus menyertakan izin berikut ke peran pemadatan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:GenerateDataKey"
            ],
```

```
            "Resource": ["arn:aws:kms:<REGION>:<KEY_OWNER_ACCOUNT_ID>:key/<KEY_ID>" ]
        }
    ]
}
```

- Peran yang Anda gunakan untuk menjalankan pemadatan harus memiliki `iam:PassRole` izin pada peran tersebut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::<account-id>:role/<compaction-role-name>"
            ]
        }
    ]
}
```

- Tambahkan kebijakan kepercayaan berikut ke peran AWS Glue layanan untuk mengambil peran IAM untuk menjalankan proses pemadatan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "glue.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
 }
```

Mengaktifkan pemadatan

Anda dapat menggunakan konsol, AWS Glue konsol AWS CLI, atau AWS API Lake Formation untuk mengaktifkan pemadatan tabel Apache Iceberg di Katalog Data. Untuk tabel baru, Anda dapat memilih Apache Iceberg sebagai format tabel dan mengaktifkan pemadatan saat Anda membuat tabel. Pemadatan dinonaktifkan secara default untuk tabel baru.

Console

Untuk mengaktifkan pemadatan

1. Buka AWS Glue konsol di https://console.aws.amazon.com/glue/ dan masuk sebagai administrator data lake, pembuat tabel, atau pengguna yang telah diberikan `lakeformation:GetDataAccess` izin `glue:UpdateTable` dan di atas tabel.
2. Di panel navigasi, di bawah Katalog Data, pilih Tabel.
3. Pada halaman Tabel, pilih tabel dalam format tabel terbuka yang ingin Anda aktifkan pemadatan, lalu di bawah menu Tindakan, pilih Aktifkan pemadatan.
4. Anda juga dapat mengaktifkan pemadatan dengan memilih tabel dan membuka halaman rincian Tabel. Pilih tab Pengoptimalan tabel di bagian bawah halaman, dan pilih Aktifkan pemadatan.
5. Selanjutnya, pilih peran IAM yang ada dari drop-down dengan izin yang ditunjukkan di bagian Prasyarat pengoptimalan tabel .

   Saat Anda memilih opsi Buat peran IAM baru, layanan akan membuat peran kustom dengan izin yang diperlukan untuk menjalankan pemadatan.

   Ikuti langkah-langkah di bawah ini untuk memperbarui peran IAM yang ada:

   a. Untuk memperbarui kebijakan izin untuk peran IAM, di konsol IAM, buka peran IAM yang digunakan untuk menjalankan pemadatan.

   b. Di bagian Tambahkan izin, pilih Buat kebijakan. Di jendela browser yang baru dibuka, buat kebijakan baru untuk digunakan dengan peran Anda.

   c. Di halaman Buat kebijakan, pilih tab JSON. Salin kode JSON yang ditampilkan di Prasyarat ke bidang editor kebijakan.

AWS CLI

Contoh berikut menunjukkan cara mengaktifkan pemadatan. Ganti ID akun dengan ID AWS akun yang valid. Ganti nama database dan nama tabel dengan nama tabel Iceberg yang sebenarnya dan nama database. Ganti `roleArn` dengan Nama AWS Sumber Daya (ARN) peran IAM dan nama peran IAM yang memiliki izin yang diperlukan untuk menjalankan pemadatan.

```
aws glue create-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/compaction_role", "enabled":'true'}' \
  --type compaction
```

AWS API

Panggilan `CreateTableOptimizer` operasi untuk mengaktifkan pemadatan untuk tabel.

Setelah Anda mengaktifkan pemadatan, tab pengoptimalan tabel menunjukkan detail pemadatan berikut (setelah sekitar 15-20 menit):

Waktu mulai

Waktu di mana proses pemadatan dimulai dalam Lake Formation. Nilainya adalah stempel waktu dalam waktu UTC.

Waktu akhir

Waktu di mana proses pemadatan berakhir di Katalog Data. Nilainya adalah stempel waktu dalam waktu UTC.

Status

Status pemadatan berjalan. Nilai adalah sukses atau gagal.

File dipadatkan

Jumlah total file yang dipadatkan.

Byte dipadatkan

Jumlah total byte yang dipadatkan.

Menonaktifkan pemadatan

Anda dapat menonaktifkan pemadatan otomatis untuk tabel Apache Iceberg tertentu menggunakan konsol atau. AWS Glue AWS CLI

Console

1. Pilih Katalog Data dan pilih Tabel. Dari daftar tabel, pilih tabel dalam format tabel terbuka yang ingin Anda nonaktifkan pemadatan.
2. Anda dapat memilih tabel Iceberg, dan memilih Nonaktifkan pemadatan di bawah Tindakan.

   Anda juga dapat menonaktifkan pemadatan untuk tabel dengan memilih Nonaktifkan pemadatan di bagian bawah halaman detail Tabel.
3. Pilih Nonaktifkan pemadatan pada pesan konfirmasi. Anda dapat mengaktifkan kembali pemadatan di lain waktu.

   Setelah Anda mengonfirmasi, pemadatan dinonaktifkan dan status pemadatan untuk tabel kembali ke. `Off`

AWS CLI

Pada contoh berikut, ganti ID akun dengan ID AWS akun yang valid. Ganti nama database dan nama tabel dengan nama tabel Iceberg yang sebenarnya dan nama database. Ganti `roleArn` dengan Nama AWS Sumber Daya (ARN) dari peran IAM dan nama sebenarnya dari peran IAM yang memiliki izin yang diperlukan untuk menjalankan pemadatan.

```
aws glue update-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --table-optimizer-configuration
 '{"roleArn":"arn:aws:iam::123456789012:role/compaction_role", "enabled":'false'}'\
  --type compaction
```

AWS API

Panggilan UpdateTableOptimizer operasi untuk menonaktifkan pemadatan untuk tabel tertentu.

Melihat detail pemadatan

Anda dapat melihat status pemadatan untuk Apache Iceberg di konsol Lake Formation AWS CLI, atau menggunakan operasi API. AWS

Console

Untuk melihat status pemadatan untuk tabel Iceberg (konsol)

- Anda dapat melihat status pemadatan untuk tabel Gunung Es di konsol Lake Formation dengan memilih Tabel di bawah Katalog Data. Bidang status pemadatan menunjukkan status proses pemadatan. Anda dapat menampilkan format tabel dan status pemadatan menggunakan preferensi tabel.
- Untuk melihat riwayat proses pemadatan untuk tabel tertentu, pilih Tabel di bawah AWS Glue Data Catalog, dan pilih tabel untuk melihat detail tabel. Tab optimasi tabel menunjukkan riwayat pemadatan untuk tabel.

AWS CLI

Anda dapat melihat detail pemadatan menggunakan AWS CLI.

Dalam contoh berikut, ganti ID akun dengan ID akun yang valid AWS , nama database, dan nama tabel dengan nama tabel Iceberg yang sebenarnya.

- Untuk mendapatkan detail proses pemadatan terakhir untuk sebuah tabel

```
aws get-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --type compaction
```

- Gunakan contoh berikut untuk mengambil riwayat pengoptimal untuk tabel tertentu.

```
aws list-table-optimizer-runs \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --type compaction
```

- Contoh berikut menunjukkan cara mengambil proses pemadatan dan detail konfigurasi untuk beberapa pengoptimal. Anda dapat menentukan maksimal 20 pengoptimal.

```
aws glue batch-get-table-optimizer \
--entries '[{"catalogId":"123456789012", "databaseName":"iceberg_db",
 "tableName":"iceberg_table", "type":"compaction"}]'
```

AWS API

- Gunakan `GetTableOptimizer` operasi untuk mengambil detail run terakhir dari pengoptimal.
- Gunakan `ListTableOptimizerRuns` operasi untuk mengambil riwayat pengoptimal yang diberikan pada tabel tertentu. Anda dapat menentukan 20 pengoptimal dalam satu panggilan API.
- Gunakan `BatchGetTableOptimizer` operasi untuk mengambil detail konfigurasi untuk beberapa pengoptimal di akun Anda. Operasi ini tidak mendukung panggilan lintas akun.

Melihat Amazon CloudWatch metrik

Setelah berhasil menjalankan pemadatan, layanan membuat Amazon CloudWatch metrik pada kinerja pekerjaan pemadatan. Anda dapat pergi ke CloudWatch Metrik dan memilih Metrik, Semua metrik. Anda dapat memfilter metrik berdasarkan namespace tertentu (misalnya AWS Glue), nama tabel, atau nama database.

Untuk informasi selengkapnya, lihat Melihat metrik yang tersedia di Panduan Amazon CloudWatch Pengguna.

- Jumlah byte yang dipadatkan
- Jumlah file yang dipadatkan
- Jumlah DPU yang dialokasikan untuk pekerjaan
- Durasi Pekerjaan (Jam)

Menghapus pengoptimal

Anda dapat menghapus pengoptimal dan metadata terkait untuk tabel yang menggunakan AWS CLI atau AWS operasi API.

Jalankan AWS CLI perintah berikut untuk menghapus riwayat pemadatan untuk sebuah tabel.

```
aws glue delete-table-optimizer \
  --catalog-id 123456789012 \
  --database-name iceberg_db \
  --table-name iceberg_table \
  --type compaction
```

Gunakan `DeleteTableOptimizer` operasi untuk menghapus pengoptimal untuk tabel.

Format dan batasan yang didukung untuk pemadatan data terkelola

Untuk kinerja pembacaan yang lebih baik oleh layanan AWS analitik seperti Amazon Athena, Amazon EMR, dan pekerjaan AWS Glue ETL, AWS Glue Data Catalog menyediakan pemadatan terkelola (proses yang memadatkan objek Amazon S3 kecil menjadi objek yang lebih besar) untuk tabel Iceberg di Katalog Data.

Pemadatan data mendukung berbagai tipe data dan format kompresi untuk membaca dan menulis data, termasuk membaca data dari tabel terenkripsi.

Pemadatan data mendukung:

- Jenis file — Parket
- Tipe data — Boolean, Integer, Panjang, Float, Ganda, String, Desimal, Tanggal, Waktu, Timestamp, String, UUID, Biner
- Kompresi - zstd, gzip, tajam, tidak terkompresi
- Enkripsi - Pemadatan data hanya mendukung enkripsi Amazon S3 default (SSE-S3) dan enkripsi KMS sisi server (SSE-KMS).
- Pemadatan paket bin
- Evolusi skema
- Tabel dengan ukuran file target (tulis. target-file-size-bytes properti dalam konfigurasi gunung es) dalam kisaran inklusif 128MB hingga 512 MB.
- Daerah
  - Asia Pasifik (Tokyo)
  - Asia Pacific (Seoul)
  - Asia Pacific (Mumbai)

  - Asia Pasifik (Singapura)
  - Eropa (Irlandia)
  - Europe (London)
  - Eropa (Frankfurt)
  - AS Timur (N. Virginia)
  - AS Timur (Ohio)
  - AS Barat (California Utara)
  - Amerika Selatan (Sao Paulo)
- Anda dapat menjalankan pemadatan dari akun tempat Katalog Data berada saat bucket Amazon S3 yang menyimpan data yang mendasarinya ada di akun lain. Untuk melakukan ini, peran pemadatan memerlukan akses ke bucket Amazon S3.

Pemadatan data saat ini tidak mendukung:

- Jenis file - Avro, ORC
- Tipe data - Tetap
- Kompresi - brotli, lz4
- Pemadatan file sementara spesifikasi partisi berkembang.
- Penyortiran reguler atau penyortiran urutan-z
- Gabungkan atau hapus file — Proses pemadatan melewatkan file data yang telah menghapus file yang terkait dengannya.
- Pemadatan pada tabel lintas akun — Anda tidak dapat menjalankan pemadatan pada tabel lintas akun.
- Pemadatan pada tabel Lintas wilayah — Anda tidak dapat menjalankan pemadatan pada tabel Lintas wilayah.
- Mengaktifkan pemadatan pada tautan sumber daya
- Titik akhir VPC untuk bucket Amazon S3
- Manajer kunci DynamoDB — Saat menggunakan pemadatan data, tidak ada pekerjaan pemuatan data lain yang harus digunakan sebagai org.apache.iceberg.aws.dynamodb. `lock-impl` DynamoDbLockManager.

Mengelola Katalog Data

AWS Glue Data Catalog Ini adalah repositori metadata pusat yang menyimpan metadata struktural dan operasional untuk kumpulan data Amazon S3 Anda. Mengelola Katalog Data secara efektif sangat penting untuk menjaga kualitas, kinerja, keamanan, dan tata kelola data.

Dengan memahami dan menerapkan praktik pengelolaan Katalog Data ini, Anda dapat memastikan metadata Anda tetap akurat, berkinerja, aman, dan diatur dengan baik seiring perkembangan lanskap data Anda.

Bagian ini mencakup aspek-aspek berikut dari manajemen Katalog Data:

- Memperbarui skema tabel dan partisi Saat data Anda berkembang, Anda mungkin perlu memperbarui skema tabel atau struktur partisi yang ditentukan dalam Katalog Data. Untuk informasi selengkapnya tentang cara membuat pembaruan ini secara terprogram menggunakan AWS Glue ETL, lihat. Memperbarui skema, dan menambahkan partisi baru di Katalog Data menggunakan AWS Glue pekerjaan ETL
- Mengelola statistik kolom: Statistik kolom yang akurat membantu mengoptimalkan rencana kueri dan meningkatkan kinerja. Untuk informasi selengkapnya tentang cara membuat, memperbarui, dan mengelola statistik kolom, lihatMengoptimalkan kinerja kueri menggunakan statistik kolom.
- Mengenkripsi Katalog Data Untuk melindungi metadata sensitif, Anda dapat mengenkripsi Katalog Data Anda menggunakan (). AWS Key Management Service AWS KMS Bagian ini menjelaskan cara mengaktifkan dan mengelola enkripsi untuk Katalog Data Anda.
- Mengamankan Katalog Data dengan AWS Lake Formation Lake Formation memberikan pendekatan komprehensif untuk keamanan data lake dan kontrol akses. Anda dapat menggunakan Lake Formation untuk mengamankan dan mengatur akses ke Katalog Data dan data yang mendasarinya.

Topik

Memperbarui skema, dan menambahkan partisi baru di Katalog Data menggunakan AWS Glue pekerjaan ETL

Tugas extract, transform, and load (ETL) Anda mungkin membuat partisi tabel baru di penyimpanan data target. Skema set data Anda dapat berevolusi dan menyimpang dari skema Katalog Data AWS Glue dari waktu ke waktu. AWS Glue Tugas ETL sekarang menyediakan beberapa fitur yang dapat Anda gunakan dalam skrip ETL Anda untuk memperbarui skema dan partisi dalam Katalog Data. Fitur ini memungkinkan Anda untuk melihat hasil tugas ETL Anda di Katalog Data, tanpa harus menjalankan kembali crawler.

Partisi baru

Jika Anda ingin melihat partisi baru di AWS Glue Data Catalog, Anda dapat melakukan salah satu hal berikut:

- Setelah tugas selesai, jalankan kembali crawler, dan lihat partisi baru di konsol tersebut saat crawler selesai.
- Setelah tugas selesai, segera lihat partisi baru di konsol tersebut, tanpa harus menjalankan ulang crawler. Anda dapat mengaktifkan fitur ini dengan menambahkan beberapa baris kode pada skrip ETL Anda, seperti yang ditunjukkan dalam contoh berikut. Kode menggunakan argumen `enableUpdateCatalog` untuk menunjukkan bahwa Katalog Data akan diperbarui selama eksekusi tugas saat partisi baru dibuat.

Metode 1

Berikan `enableUpdateCatalog` dan `partitionKeys` dalam sebuah argumen pilihan.

Python

```
additionalOptions = {"enableUpdateCatalog": True}
additionalOptions["partitionKeys"] = ["region", "year", "month", "day"]


sink = glueContext.write_dynamic_frame_from_catalog(frame=last_transform,
 database=<target_db_name>,

 table_name=<target_table_name>, transformation_ctx="write_sink",

 additional_options=additionalOptions)
```

Scala

```
val options = JsonOptions(Map(
    "path" -> <S3_output_path>,
    "partitionKeys" -> Seq("region", "year", "month", "day"),
    "enableUpdateCatalog" -> true))
val sink = glueContext.getCatalogSink(
    database = <target_db_name>,
    tableName = <target_table_name>,
    additionalOptions = options)sink.writeDynamicFrame(df)
```

Metode 2

Berikan `enableUpdateCatalog` dan `partitionKeys` dalam `getSink()`, dan panggil `setCatalogInfo()` di objek `DataSink`.

Python

```
sink = glueContext.getSink(
    connection_type="s3",
    path="<S3_output_path>",
    enableUpdateCatalog=True,
    partitionKeys=["region", "year", "month", "day"])
sink.setFormat("json")
sink.setCatalogInfo(catalogDatabase=<target_db_name>,
 catalogTableName=<target_table_name>)
sink.writeFrame(last_transform)
```

Scala

```
val options = JsonOptions(
   Map("path" -> <S3_output_path>,
       "partitionKeys" -> Seq("region", "year", "month", "day"),
       "enableUpdateCatalog" -> true))
val sink = glueContext.getSink("s3", options).withFormat("json")
sink.setCatalogInfo(<target_db_name>, <target_table_name>)
sink.writeDynamicFrame(df)
```

Sekarang, Anda dapat membuat tabel katalog baru, memperbarui tabel yang ada dengan skema yang sudah dimodifikasi, dan menambahkan partisi tabel baru dalam Katalog Data dengan menggunakan tugas ETL AWS Glue itu sendiri, tanpa perlu kembali menjalankan crawler.

Memperbarui skema tabel

Jika Anda ingin menimpa skema tabel Katalog Data Anda, Anda dapat melakukan salah satu hal berikut:

- Setelah tugas selesai, jalankan kembali crawler dan pastikan crawler dikonfigurasi untuk memperbarui definisi tabel juga. Lihat partisi baru di konsol tersebut beserta pembaruan skema apa pun, saat crawler selesai. Untuk informasi selengkapnya, lihat Mengkonfigurasi Crawler Menggunakan API.
- Setelah tugas selesai, segera lihat skema yang sudah dimodifikasi di konsol tersebut, tanpa harus menjalankan ulang crawler. Anda dapat mengaktifkan fitur ini dengan menambahkan beberapa baris kode pada skrip ETL Anda, seperti yang ditunjukkan dalam contoh berikut. Kode menggunakan `enableUpdateCatalog` yang diatur ke BETUL, dan juga `updateBehavior` yang diatur ke `UPDATE_IN_DATABASE`, yang menunjukkan untuk menimpa skema dan menambahkan partisi baru dalam Katalog Data selama eksekusi tugas.

Python

```
additionalOptions = {
    "enableUpdateCatalog": True,
    "updateBehavior": "UPDATE_IN_DATABASE"}
additionalOptions["partitionKeys"] = ["partition_key0", "partition_key1"]

sink = glueContext.write_dynamic_frame_from_catalog(frame=last_transform,
 database=<dst_db_name>,
    table_name=<dst_tbl_name>, transformation_ctx="write_sink",
    additional_options=additionalOptions)
job.commit()
```

Scala

```
val options = JsonOptions(Map(
    "path" -> outputPath,
    "partitionKeys" -> Seq("partition_0", "partition_1"),
    "enableUpdateCatalog" -> true))
val sink = glueContext.getCatalogSink(database = nameSpace, tableName = tableName,
 additionalOptions = options)
sink.writeDynamicFrame(df)
```

Anda juga dapat mengatur nilai `updateBehavior` ke `LOG` jika Anda ingin mencegah skema tabel Anda agar tidak ditimpa, tapi masih ingin menambahkan partisi baru. Nilai default dari `updateBehavior` adalah `UPDATE_IN_DATABASE`, jadi jika Anda tidak secara eksplisit mendefinisikannya, maka skema tabel akan ditimpa.

Jika `enableUpdateCatalog` tidak diatur ke BETUL, terlepas dari mana pilihan yang dipilih untuk `updateBehavior`, tugas ETL tidak akan memperbarui tabel di Katalog Data.

Membuat tabel baru

Anda juga dapat menggunakan opsi yang sama untuk membuat sebuah tabel baru di Katalog Data. Anda dapat menentukan basis data dan nama tabel baru dengan menggunakan `setCatalogInfo`.

Python

```
sink = glueContext.getSink(connection_type="s3", path="s3://path/to/data",
    enableUpdateCatalog=True, updateBehavior="UPDATE_IN_DATABASE",
    partitionKeys=["partition_key0", "partition_key1"])
sink.setFormat("<format>")
sink.setCatalogInfo(catalogDatabase=<dst_db_name>, catalogTableName=<dst_tbl_name>)
sink.writeFrame(last_transform)
```

Scala

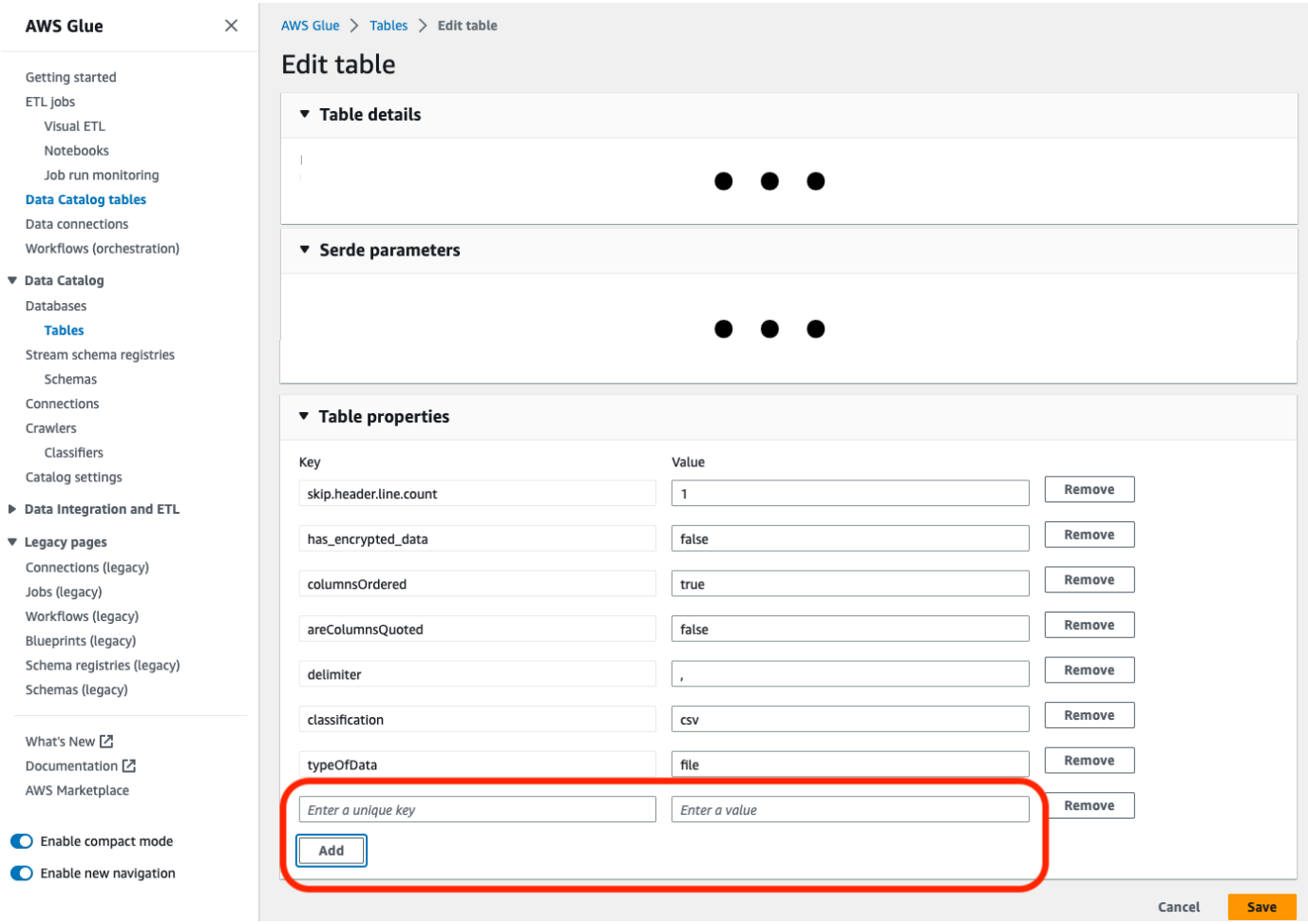
```
val options = JsonOptions(Map(
    "path" -> outputPath,
    "partitionKeys" -> Seq("<partition_1>", "<partition_2>"),
    "enableUpdateCatalog" -> true,
    "updateBehavior" -> "UPDATE_IN_DATABASE"))
val sink = glueContext.getSink(connectionType = "s3", connectionOptions =
 options).withFormat("<format>")
sink.setCatalogInfo(catalogDatabase = “<dst_db_name>”, catalogTableName =
 “<dst_tbl_name>”)
sink.writeDynamicFrame(df)
```

Pembatasan

Perhatikan pembatasan-pembatasan berikut ini:

- Hanya target Amazon Simple Storage Service (Amazon S3) saja yang didukung.
- `enableUpdateCatalog`Fitur ini tidak didukung untuk tabel yang diatur.

- Hanya format berikut ini didukung: `json`, `csv`, `avro`, dan `parquet`.
- Untuk membuat atau memperbarui tabel dengan `parquet` klasifikasi, Anda harus menggunakan penulis parket yang AWS Glue dioptimalkan untuk. DynamicFrames Ini dapat dicapai dengan salah satu dari yang berikut:
  - Jika Anda memperbarui tabel yang ada dalam katalog dengan `parquet` klasifikasi, tabel harus memiliki properti `"useGlueParquetWriter"` tabel yang disetel ke `true` sebelum Anda memperbaruinya. Anda dapat mengatur properti ini melalui AWS Glue APIS/SDK, melalui konsol atau melalui pernyataan Athena DDL.



Setelah properti tabel katalog diatur, Anda dapat menggunakan potongan kode berikut untuk memperbarui tabel katalog dengan data baru:

```
glueContext.write_dynamic_frame.from_catalog(
    frame=frameToWrite,
    database="dbName",
    table_name="tableName",
    additional_options={
```

```
        "enableUpdateCatalog": True,
        "updateBehavior": "UPDATE_IN_DATABASE"
    }
)
```

- Jika tabel belum ada dalam katalog, Anda dapat menggunakan `getSink()` metode dalam skrip Anda `connection_type="s3"` untuk menambahkan tabel dan partisi ke katalog, bersama dengan menulis data ke Amazon S3. Berikan yang sesuai `partitionKeys` dan `compression` untuk alur kerja Anda.

```
s3sink = glueContext.getSink(
    path="s3://bucket/folder/",
    connection_type="s3",
    updateBehavior="UPDATE_IN_DATABASE",
    partitionKeys=[],
    compression="snappy",
    enableUpdateCatalog=True
)

s3sink.setCatalogInfo(
    catalogDatabase="dbName", catalogTableName="tableName"
)

s3sink.setFormat("parquet", useGlueParquetWriter=true)
s3sink.writeFrame(frameToWrite)
```

  - Nilai `glueparquet` format adalah metode warisan yang memungkinkan penulis AWS Glue parket.

- Saat `updateBehavior` diatur ke `LOG`, partisi baru akan ditambahkan hanya jika skema `DynamicFrame` setara dengan atau berisi sebuah subset dari kolom yang didefinisikan dalam skema tabel Katalog Data.
- Pembaruan skema tidak didukung untuk tabel non-partisi (tidak menggunakan opsi “PartitionKeys”).
- PartitionKeys Anda harus setara, dan dalam urutan yang sama, antara parameter Anda yang diberikan dalam skrip ETL dan PartitionKeys dalam skema tabel Katalog Data Anda.
- Fitur ini saat ini belum men-support pembaruan/pembuatan tabel di mana skema yang memperbarui bersarang (misalnya, array dalam struct).

Untuk informasi selengkapnya, lihat the section called “AWSGlue untuk Spark”.

Bekerja dengan koneksi MongoDB pada Tugas ETL

Anda dapat membuat koneksi untuk MongoDB dan kemudian menggunakan koneksi yang di tugas AWS Glue Anda. Untuk informasi selengkapnya, lihat the section called “Koneksi MongoDB” di panduan AWS Glue pemrograman. Koneksi `url`, `username` dan `password` disimpan dalam koneksi MongoDB. Pilihan lain dapat ditentukan dalam skrip tugas ETL Anda dengan menggunakan parameter `additionalOptions` dari `glueContext.getCatalogSource`. Pilihan lainnya bisa meliputi:

- `database`: (Wajib) Basis data MongoDB untuk dibaca.
- `collection`: (Wajib) Kumpulan MongoDB untuk dibaca.

Dengan menempatkan informasi `database` dan `collection` dalam skrip tugas ETL, Anda dapat menggunakan koneksi yang sama untuk beberapa tugas.

1. Buat koneksi AWS Glue Data Catalog untuk sumber data MongoDB. Lihat "connectionType": "mongodb" untuk deskripsi parameter koneksi. Anda dapat membuat koneksi menggunakan konsol, API atau CLI.
2. Membuat basis data di AWS Glue Data Catalog untuk menyimpan definisi tabel untuk data MongoDB Anda. Lihat Membuat database untuk informasi selengkapnya.
3. Buat crawler yang melakukan perayapan pada data yang ada dalam MongoDB dengan menggunakan informasi dalam koneksi tersebut untuk connect ke MongoDB. Crawler menciptakan tabel di AWS Glue Data Catalog yang mendeskripsikan tabel dalam basis data MongoDB yang Anda gunakan dalam tugas Anda. Lihat Menggunakan crawler untuk mengisi Katalog Data untuk informasi selengkapnya.
4. Buat tugas dengan sebuah skrip kustom. Anda dapat membuat tugas menggunakan konsol, API atau CLI. Untuk informasi selengkapnya, lihat Menambahkan Tugas di AWS Glue.
5. Pilih target data untuk tugas Anda. Tabel yang mewakili target data dapat didefinisikan dalam Katalog Data Anda, atau tugas Anda dapat membuat tabel target ketika ia berjalan. Anda memilih lokasi target ketika Anda menulis tugas. Jika target memerlukan sebuah koneksi, maka koneksi tersebut juga direferensikan dalam tugas Anda. Jika tugas Anda memerlukan beberapa target data, maka Anda dapat menambahkannya nanti dengan mengedit skrip.
6. Sesuaikan lingkungan pemrosesan tugas dengan memberikan argumen untuk tugas Anda dan skrip yang dihasilkan.

Berikut adalah contoh membuat `DynamicFrame` dari basis data MongoDB berdasarkan struktur tabel yang ditentukan dalam Katalog Data. Kode menggunakan `additionalOptions` untuk memberikan informasi sumber data tambahan:

Scala

```
val resultFrame: DynamicFrame = glueContext.getCatalogSource(
        database = catalogDB,
        tableName = catalogTable,
        additionalOptions = JsonOptions(Map("database" -> DATABASE_NAME,
                "collection" -> COLLECTION_NAME))
      ).getDynamicFrame()
```

Python

```
glue_context.create_dynamic_frame_from_catalog(
        database = catalogDB,
        table_name = catalogTable,
        additional_options = {"database":"database_name",
            "collection":"collection_name"})
```

7. Jalankan tugas, baik sesuai permintaan atau melalui pemicu.

Mengoptimalkan kinerja kueri menggunakan statistik kolom

Anda dapat menghitung statistik tingkat kolom untuk AWS Glue Data Catalog tabel dalam format data seperti Parket, ORC, JSON, ION, CSV, dan XMLtanpa menyiapkan pipeline data tambahan. Statistik kolom membantu Anda memahami profil data dengan mendapatkan wawasan tentang nilai dalam kolom. Data Catalog mendukung menghasilkan statistik untuk nilai kolom seperti nilai minimum, nilai maksimum, total nilai nol, total nilai yang berbeda, panjang rata-rata nilai, dan total kemunculan nilai sebenarnya.

AWS layanan analisis seperti Amazon Redshift dan Amazon Athena dapat menggunakan statistik kolom ini untuk menghasilkan rencana eksekusi kueri, dan memilih paket optimal yang meningkatkan kinerja kueri.

Anda dapat mengonfigurasi untuk menjalankan tugas pembuatan statistik kolom menggunakan AWS Glue konsol atau AWS CLI. Saat Anda memulai proses, AWS Glue mulailah pekerjaan Spark di latar belakang dan perbarui metadata AWS Glue tabel di Katalog Data. Anda dapat melihat

statistik kolom menggunakan AWS Glue konsol atau AWS CLI atau dengan memanggil operasi GetColumnStatisticsForTabel API.

> **Note**
>
> Jika Anda menggunakan izin Lake Formation untuk mengontrol akses ke tabel, peran yang diasumsikan oleh tugas statistik kolom memerlukan akses tabel penuh untuk menghasilkan statistik.

Topik

Prasyarat untuk menghasilkan statistik kolom

Untuk menghasilkan atau memperbarui statistik kolom, tugas pembuatan statistik mengasumsikan peran AWS Identity and Access Management (IAM) atas nama Anda. Berdasarkan izin yang diberikan untuk peran tersebut, tugas pembuatan statistik kolom dapat membaca data dari penyimpanan data Amazon S3.

> **Note**
>
> Untuk menghasilkan statistik untuk tabel yang dikelola oleh Lake Formation, peran IAM yang digunakan untuk menghasilkan statistik memerlukan akses tabel penuh.

Untuk menggunakan kontrol akses berbasis peran, Anda harus membuat peran IAM dengan izin yang tercantum dalam kebijakan di bawah ini, dan menambahkan peran tersebut ke tugas pembuatan statistik kolom.

Untuk membuat peran IAM untuk menghasilkan statistik kolom

1. Untuk membuat peran IAM, lihat Membuat peran IAM untuk. AWS Glue
2. Untuk memperbarui peran yang ada, di konsol IAM, buka peran IAM yang digunakan oleh proses statistik kolom generate.
3. Di bagian Tambahkan izin, pilih Lampirkan kebijakan. Di jendela browser yang baru dibuka, pilih kebijakan `AWSGlueServiceRole` AWS terkelola.
4. Anda juga perlu menyertakan izin untuk membaca data dari lokasi data Amazon S3.

   Di bagian Tambahkan izin, pilih Buat kebijakan. Di jendela browser yang baru dibuka, buat kebijakan baru untuk digunakan dengan peran Anda.
5. Di halaman Buat kebijakan, pilih tab JSON. Salin `JSON` kode berikut ke kolom editor kebijakan.

   > **Note**
   >
   > Dalam kebijakan berikut, ganti ID akun dengan yang valid Akun AWS, lalu ganti `region` dengan Wilayah tabel, dan `bucket-name` dengan nama bucket Amazon S3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3BucketAccess",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::<bucket-name>/*",
                "arn:aws:s3:::<bucket-name>"
            ]
        }
     ]
}
```

6. (Opsional) Jika Anda menggunakan izin Lake Formation untuk menyediakan akses ke data Anda, peran IAM memerlukan `lakeformation:GetDataAccess` izin.

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
            "Sid": "LakeFormationDataAccess",
            "Effect": "Allow",
            "Action": "lakeformation:GetDataAccess",
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Jika lokasi data Amazon S3 terdaftar dengan Lake Formation, dan peran IAM yang diasumsikan oleh tugas pembuatan statistik kolom tidak memiliki izin `IAM_ALLOWED_PRINCIPALS` grup yang diberikan pada tabel, peran tersebut memerlukan Lake Formation `ALTER` dan `DESCRIBE` izin pada tabel. Peran yang digunakan untuk mendaftarkan bucket Amazon S3 memerlukan Lake Formation `INSERT` dan `DELETE` izin di atas meja.

Jika lokasi data Amazon S3 tidak terdaftar dengan Lake Formation, dan peran IAM tidak memiliki izin `IAM_ALLOWED_PRINCIPALS` grup yang diberikan pada tabel, peran tersebut memerlukan Lake Formation `ALTERDESCRIBE`, `INSERT` dan `DELETE` izin pada tabel.

7. (Opsional) Tugas pembuatan statistik kolom yang menulis terenkripsi Amazon CloudWatch Logs memerlukan izin berikut dalam kebijakan utama.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "CWLogsKmsPermissions",
        "Effect":  "Allow",
        "Action": [
            "logs:CreateLogGroup",
            "logs:CreateLogStream",
            "logs:PutLogEvents",
            "logs:AssociateKmsKey"
        ],
        "Resource": [
            "arn:aws:logs:<region>:111122223333:log-group:/aws-glue:*"
```

```
        ]
    },
    {
        "Sid": "KmsPermissions",
        "Effect": "Allow",
        "Action": [
            "kms:GenerateDataKey",
            "kms:Decrypt",
            "kms:Encrypt"
        ],
        "Resource": [
        "arn:aws:kms:<region>:111122223333:key/"arn of key used for ETL cloudwatch
 encryption"
        ],
        "Condition": {
            "StringEquals": {
                "kms:ViaService": ["glue.<region>.amazonaws.com"]
            }
        }
    }
    ]
}
```

8. Peran yang Anda gunakan untuk menjalankan statistik kolom harus memiliki `iam:PassRole` izin pada peran tersebut.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "iam:PassRole"
        ],
        "Resource": [
            "arn:aws:iam::111122223333:role/<columnstats-role-name>"
        ]
    }]
}
```

9. Saat Anda membuat peran IAM untuk menghasilkan statistik kolom, peran tersebut juga harus memiliki kebijakan kepercayaan berikut yang memungkinkan layanan untuk mengambil peran tersebut.

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Sid": "TrustPolicy",
        "Effect": "Allow",
        "Principal": {
          "Service": "glue.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
      }
   ]
}
```

Menghasilkan statistik kolom

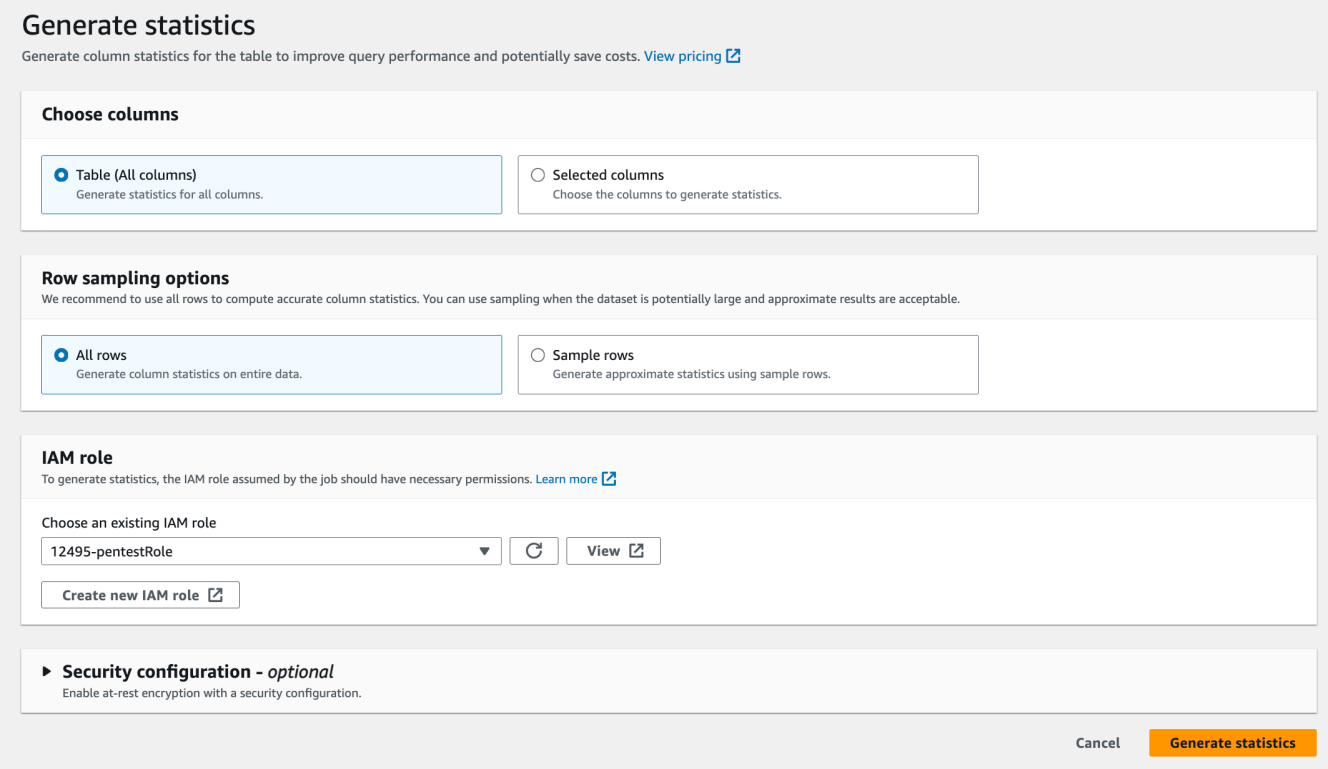
Ikuti langkah-langkah ini untuk mengelola pembuatan statistik di Katalog Data menggunakan AWS Glue konsol atauAWS CLI.

Console

Menghasilkan statistik kolom menggunakan konsol

1. Masuk ke AWS Glue konsol di https://console.aws.amazon.com/glue/.
2. Pilih tabel Katalog Data.
3. Pilih tabel dari daftar.
4. Pilih Hasilkan statistik di bawah menu Tindakan.

   Anda juga dapat memilih tombol Hasilkan statistik di bawah tab Statistik kolom di bagian bawah halaman Tabel.
5. Pada halaman Hasilkan statistik, tentukan opsi berikut:



- Tabel (semua kolom) - Pilih opsi ini untuk menghasilkan statistik untuk semua kolom dalam tabel.
- Kolom yang dipilih - Pilih opsi ini untuk menghasilkan statistik untuk kolom tertentu. Anda dapat memilih kolom dari daftar menurun.
- Semua baris - Pilih semua baris dari tabel untuk menghasilkan statistik yang akurat.
- Baris sampel - Pilih hanya persentase baris tertentu dari tabel untuk menghasilkan statistik. Defaultnya adalah semua baris. Gunakan panah atas dan bawah untuk menambah atau mengurangi nilai persen.

> **Note**
>
> Kami merekomendasikan untuk memasukkan semua baris dalam tabel untuk menghitung statistik yang akurat. Gunakan baris sampel untuk menghasilkan statistik kolom hanya jika nilai perkiraan dapat diterima.

6. (Opsional) Selanjutnya, pilih konfigurasi keamanan untuk mengaktifkan enkripsi saat istirahat untuk log.
7. Pilih Hasilkan statistik untuk menjalankan proses.

AWS CLI

Dalam contoh berikut, ganti nilai untuk`DatabaseName`,`TableName`, dan `ColumnNameList` dengan database aktual, tabel, dan nama kolom. Ganti ID akun dengan nama peran yang valid Akun AWS dan nama peran dengan nama peran IAM yang Anda gunakan untuk menghasilkan statistik.

```
aws glue start-column-statistics-task-run --input-cli-json file://input.json
{
    "DatabaseName": "<test-db>",
    "TableName": "<test-table>",
    "ColumnNameList": [
        "<column1>",
        "<column2>",
    ],
    "Role": "arn:aws:iam::<123456789012>:role/<Stats-Role>",
    "SampleSize": 10.0
}
```

Anda dapat menghasilkan statistik kolom juga dengan memanggil StartColumnStatisticsTaskRunoperasi.

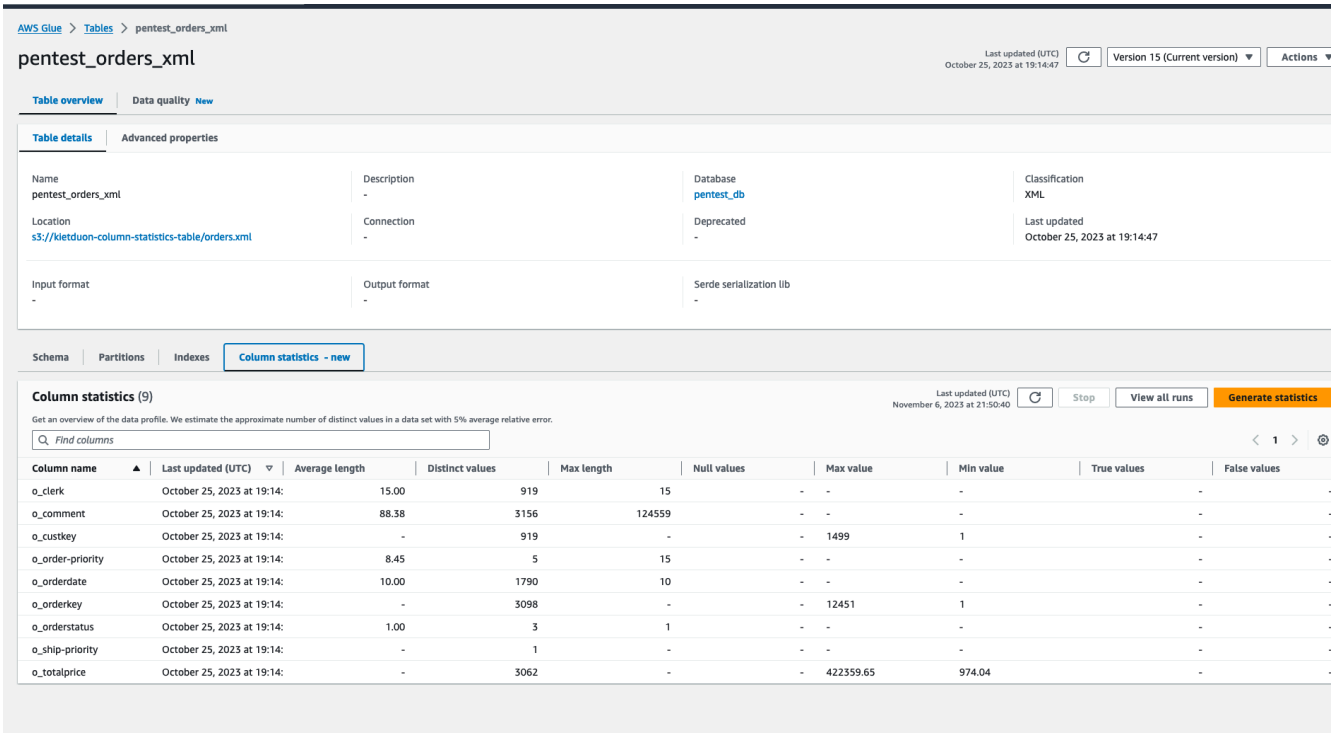
Melihat statistik kolom

Setelah berhasil menghasilkan statistik, Data Catalog menyimpan informasi ini untuk pengoptimal berbasis biaya di dan Amazon Amazon Athena Redshift untuk membuat pilihan optimal saat menjalankan kueri. Statistik bervariasi berdasarkan jenis kolom.

AWS Management Console

Untuk melihat statistik kolom untuk tabel

- Setelah menjalankan tugas statistik kolom, tab Statistik kolom pada halaman Rincian tabel menunjukkan statistik untuk tabel.



Statistik berikut tersedia:

- Nama kolom: Nama kolom yang digunakan untuk menghasilkan statistik
- Terakhir diperbarui: Data dan waktu ketika statistik dihasilkan
- Panjang rata-rata: Panjang rata-rata nilai di kolom
- Nilai yang berbeda: Jumlah nilai yang berbeda dalam kolom. Kami memperkirakan jumlah nilai yang berbeda dalam kolom dengan kesalahan relatif 5%.
- Nilai maksimum: Nilai terbesar dalam kolom.
- Nilai min: Nilai terkecil dalam kolom.
- Panjang maks: Panjang nilai tertinggi dalam kolom.
- Nilai nol: Jumlah total nilai nol dalam kolom.
- Nilai yang sebenarnya: Jumlah nilai yang sebenarnya dalam kolom.
- Nilai SALAH: Jumlah nilai SALAH dalam kolom.

AWS CLI

Contoh berikut menunjukkan cara mengambil statistik kolom. AWS CLI

```
aws glue get-column-statistics-for-table \
```

```
    --database-name <test_db> \
    --table-name <test_tble> \
    --column-names <col1>
```

Anda juga dapat melihat statistik kolom menggunakan operasi GetColumnStatisticsForTableAPI.

Memperbarui statistik kolom

Menjaga statistik terkini meningkatkan kinerja kueri dengan memungkinkan perencana kueri untuk memilih paket yang optimal. Anda perlu menjalankan tugas Hasilkan statistik secara eksplisit dari AWS Glue konsol untuk menyegarkan statistik kolom. Katalog Data tidak secara otomatis menyegarkan statistik.

Jika Anda tidak menggunakan AWS Glue fitur pembuatan statistik di konsol,
Anda dapat memperbarui statistik kolom secara manual menggunakan operasi UpdateColumnStatisticsForTableAPI atauAWS CLI. Contoh berikut menunjukkan cara memperbarui statistik menggunakanAWS CLI.

```
aws glue update-column-statistics-for-table --cli-input-json:

{
    "CatalogId": "111122223333",
    "DatabaseName": "test_db",
    "TableName": "test_table",
    "ColumnStatisticsList": [
        {
            "ColumnName": "col1",
            "ColumnType": "Boolean",
            "AnalyzedTime": "1970-01-01T00:00:00",
            "StatisticsData": {
                "Type": "BOOLEAN",
                "BooleanColumnStatisticsData": {
                    "NumberOfTrues": 5,
                    "NumberOfFalses": 5,
                    "NumberOfNulls": 0
                }
            }
        }
    ]
}
```

Menghapus statistik kolom

Anda dapat menghapus statistik kolom menggunakan operasi DeleteColumnStatisticsForTableAPI atauAWS CLI. Contoh berikut menunjukkan cara menghapus statistik kolom menggunakan AWS Command Line Interface (AWS CLI).

```
aws glue delete-column-statistics-for-table \
    --database-name test_db \
    --table-name test_table \
    --column-name col1
```

Melihat tugas statistik kolom berjalan

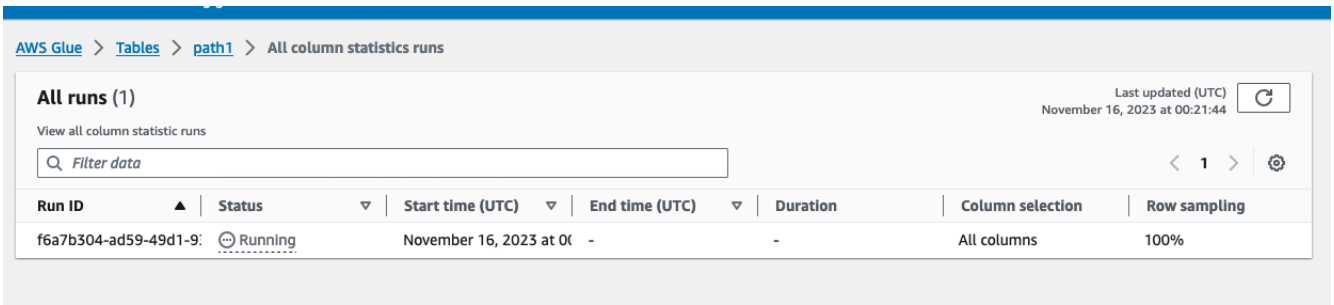
Setelah menjalankan tugas statistik kolom, Anda dapat menjelajahi detail menjalankan tugas untuk tabel menggunakan AWS Glue konsol, AWS CLI atau menggunakan GetColumnStatisticsTaskRunsoperasi.

Console

Untuk melihat rincian tugas statistik kolom

1. Di AWS Glue konsol, pilih Tabel di bawah Katalog Data.
2. Pilih tabel dengan statistik kolom.
3. Pada halaman Rincian tabel, pilih Statistik kolom.
4. Pilih Lihat berjalan.

   Anda dapat melihat informasi tentang semua proses yang terkait dengan tabel yang ditentukan.



AWS CLI

Dalam contoh berikut, ganti nilai untuk `DatabaseName` dan `TableName` dengan database dan nama tabel yang sebenarnya.

```
aws glue get-column-statistics-task-runs --input-cli-json file://input.json
{
    "DatabaseName": "<test-db>",
    "TableName": "<test-table>"
}
```

Menghentikan tugas statistik kolom

Anda dapat menghentikan tugas statistik kolom yang dijalankan untuk tabel menggunakan AWS Glue konsol, AWS CLI atau menggunakan StopColumnStatisticsTaskRunoperasi.

Console

Untuk menghentikan tugas statistik kolom, jalankan

1. Di AWS Glue konsol, pilih Tabel di bawah Katalog Data.
2. Pilih tabel dengan tugas statistik kolom yang sedang berjalan.
3. Pada halaman Rincian tabel, pilih Statistik kolom.
4. Pilih Berhenti.

   Jika Anda menghentikan tugas sebelum proses selesai, statistik kolom tidak akan dihasilkan untuk tabel.

AWS CLI

Dalam contoh berikut, ganti nilai untuk `DatabaseName` dan `TableName` dengan database dan nama tabel yang sebenarnya.

```
aws glue stop-column-statistics-task-run --input-cli-json file://input.json
{
    "DatabaseName": "<test-db>",
    "TableName": "<test-table>"
}
```

Pertimbangan dan batasan

Pertimbangan dan batasan berikut berlaku untuk menghasilkan statistik kolom.

Pertimbangan-pertimbangan

- Menggunakan sampling untuk menghasilkan statistik mengurangi waktu berjalan, tetapi dapat menghasilkan statistik yang tidak akurat.
- Setiap statistik kolom yang dijalankan memerlukan pemrosesan seluruh kumpulan data.
- Katalog Data tidak menyimpan versi statistik yang berbeda.
- Anda hanya dapat menjalankan satu tugas pembuatan statistik pada satu waktu per tabel.
- Jika tabel dienkripsi menggunakan AWS KMS kunci pelanggan yang terdaftar dengan Katalog Data, AWS Glue gunakan kunci yang sama untuk mengenkripsi statistik.

Tugas statistik kolom mendukung menghasilkan statistik:

- Ketika peran IAM memiliki izin tabel lengkap (IAM atau Lake Formation).
- Ketika peran IAM memiliki izin di atas meja menggunakan mode akses hibrida Lake Formation.

Tugas statistik kolom tidak mendukung pembuatan statistik untuk:

- Tabel dengan kontrol akses berbasis sel Lake Formation.
- Danau data transaksional - Yayasan Linux Delta Lake, Apache Iceberg, Apache Hudi.
- Tabel dalam database federasi - Hive metastore, datashares Amazon Redshift
- Kolom bersarang, array, dan tipe data struct.
- Tabel yang dibagikan dengan Anda dari akun lain.

Mengenkripsi Katalog Data Anda

Anda dapat melindungi metadata yang disimpan dalam keadaan diam menggunakan kunci enkripsi yang dikelola oleh AWS Key Management Service ()AWS KMS. AWS Glue Data Catalog Anda dapat mengaktifkan enkripsi Katalog Data untuk Katalog Data baru, dengan menggunakan pengaturan

Katalog Data. Anda dapat mengaktifkan atau menonaktifkan enkripsi untuk Katalog Data yang ada sesuai kebutuhan. Saat diaktifkan, AWS Glue mengenkripsi semua metadata baru yang ditulis ke katalog, sementara metadata yang ada tetap tidak terenkripsi.

Untuk informasi rinci tentang mengenkripsi Katalog Data Anda, lihat. Mengenkripsi Katalog Data Anda

Mengamankan Katalog Data Anda menggunakan Lake Formation

AWS Lake Formation adalah layanan yang membuatnya lebih mudah untuk mengatur danau data yang aman di AWS. Ini menyediakan tempat sentral untuk membuat dan mengelola danau data Anda dengan aman dengan mendefinisikan izin kontrol akses yang diberikan dengan baik. Lake Formation menggunakan Katalog Data untuk menyimpan dan mengambil metadata tentang data lake Anda, seperti definisi tabel, informasi skema, dan pengaturan kontrol akses data.

Anda dapat mendaftarkan lokasi data Amazon S3 dari tabel metadata atau database dengan Lake Formation dan menggunakannya untuk menentukan izin tingkat metadata pada sumber daya Katalog Data. Anda juga dapat menggunakan Lake Formation untuk mengelola izin akses penyimpanan pada data dasar yang disimpan di Amazon S3 atas nama mesin analitik terintegrasi.

Untuk informasi lebih lanjut lihat Apa itu AWS Lake Formation? .

Mengakses Katalog Data

Anda dapat menggunakan AWS Glue Data Catalog untuk menemukan dan memahami data Anda. Katalog Data menyediakan cara yang konsisten untuk mempertahankan definisi skema, tipe data, lokasi, dan metadata lainnya. Anda dapat mengakses Katalog Data menggunakan metode berikut:

- AWS Glue konsol — Anda dapat mengakses dan mengelola Katalog Data melalui AWS Glue konsol, antarmuka pengguna berbasis web. Konsol memungkinkan Anda untuk menelusuri dan mencari database, tabel, dan metadata terkait, serta membuat, memperbarui, dan menghapus definisi metadata.
- Perayap AWS Glue — Crawler adalah program yang secara otomatis memindai sumber data Anda dan mengisi Katalog Data dengan metadata. Anda dapat membuat dan menjalankan crawler untuk menemukan dan membuat katalog data dari berbagai sumber seperti Amazon S3, Amazon RDS, Amazon DynamoDB,, dan database relasional yang sesuai dengan JDBC seperti MySQL Amazon CloudWatch, dan PostgreSQL serta beberapa non-sumber seperti Snowflake dan Google.AWS BigQuery

- AWS Glue API — Anda dapat mengakses Katalog Data secara terprogram menggunakan API. AWS Glue API ini memungkinkan Anda berinteraksi dengan Katalog Data secara terprogram, memungkinkan otomatisasi dan integrasi dengan aplikasi dan layanan lain.
- AWS Command Line Interface (AWS CLI) — Anda dapat menggunakan AWS CLI untuk mengakses dan mengelola Katalog Data dari baris perintah. CLI menyediakan perintah untuk membuat, memperbarui, dan menghapus definisi metadata, serta menanyakan dan mengambil informasi metadata.
- Integrasi dengan AWS layanan lain - Katalog Data terintegrasi dengan berbagai AWS layanan lain, memungkinkan Anda untuk mengakses dan memanfaatkan metadata yang disimpan dalam katalog. Misalnya, Anda dapat menggunakan Amazon Athena untuk menanyakan sumber data menggunakan metadata di Katalog Data, dan gunakan AWS Lake Formation untuk mengelola akses dan tata kelola data untuk sumber daya Katalog Data.

AWS Glue Praktik terbaik Katalog Data

Bagian ini mencakup praktik terbaik untuk mengelola dan memanfaatkan secara efektif. AWS Glue Data Catalog Ini menekankan praktik seperti penggunaan crawler yang efisien, organisasi metadata, keamanan, optimasi kinerja, otomatisasi, tata kelola data, dan integrasi dengan layanan lainnya. AWS

- Gunakan crawler secara efektif — Jalankan crawler secara teratur untuk menjaga Katalog Data up-to-date dengan perubahan dalam sumber data Anda. Gunakan crawl inkremental untuk sering mengubah sumber data untuk meningkatkan kinerja. Konfigurasikan crawler untuk secara otomatis menambahkan partisi baru atau memperbarui skema saat perubahan terdeteksi.
- Mengatur dan memberi nama tabel metadata — Menetapkan konvensi penamaan yang konsisten untuk database dan tabel dalam Katalog Data. Kelompokkan sumber data terkait ke dalam database atau folder logis untuk organisasi yang lebih baik. Gunakan nama deskriptif yang menyampaikan tujuan dan isi dari setiap tabel.
- Kelola skema secara efektif — Manfaatkan kemampuan inferensi skema crawler. AWS Glue Tinjau dan perbarui perubahan skema sebelum menerapkannya untuk menghindari kerusakan aplikasi hilir. Gunakan fitur evolusi skema untuk menangani perubahan skema dengan anggun.
- Amankan Katalog Data — Aktifkan enkripsi data saat istirahat dan dalam perjalanan untuk Katalog Data. Menerapkan kebijakan kontrol akses berbutir halus untuk membatasi akses ke data sensitif. Secara teratur mengaudit dan meninjau izin Katalog Data dan log aktivitas.

- Integrasikan dengan AWS layanan lain Katalog Data Gunakan Katalog Data sebagai lapisan metadata terpusat untuk layanan seperti Amazon Athena, Redshift Spectrum, dan. AWS Lake Formation Manfaatkan pekerjaan AWS Glue ETL untuk mengubah dan memuat data ke berbagai penyimpanan data sambil mempertahankan metadata di Katalog Data.
- Memantau dan mengoptimalkan kinerja Katalog Data Memantau kinerja crawler dan pekerjaan ETL menggunakan Amazon CloudWatch metrik. Partisi dataset besar dalam Katalog Data untuk meningkatkan kinerja kueri. Menerapkan pengoptimalan kinerja untuk metadata yang sering diakses.
- Tetap diperbarui dengan AWS Glue dokumentasi dan praktik terbaik Katalog Data Periksa AWS Glue dokumentasi dan AWS Glue sumber daya secara teratur untuk pembaruan, praktik terbaik, dan rekomendasi terbaru. Hadiri AWS Glue webinar, lokakarya, dan acara lainnya untuk belajar dari para ahli dan tetap mendapat informasi tentang fitur dan kemampuan baru.

Registri Skema AWS Glue

> **Note**
>
> AWS GlueRegistri Skema tidak didukung di Wilayah berikut di AWS Glue konsol: Asia Pasifik (Jakarta) dan Timur Tengah (UEA).

Registri Skema AWS Glue adalah sebuah fitur baru yang memungkinkan Anda untuk menemukan, mengontrol, dan mengembangkan skema aliran data secara terpusat. Sebuah skema mendefinisikan struktur dan format catatan data. Dengan AWS Glue Schema Registry, Anda dapat mengelola dan menerapkan skema pada aplikasi streaming data Anda menggunakan integrasi yang nyaman dengan Apache Kafka,, Amazon Kinesis Amazon Managed Streaming for Apache KafkaData Streams, Amazon Managed Service untuk Apache Flink, dan. AWS Lambda

Registri AWS Glue Skema mendukung format data AVRO (v1.10.2), format Data JSON dengan format Skema JSON untuk skema (spesifikasi Draft-04, Draft-06, dan Draft-07) dengan validasi skema JSON menggunakan pustaka Everit, Protokol Buffer (Protobuf) versi proto2 dan proto3 tanpa dukungan untuk atau, dan dukungan bahasa Java, dengan format data dan bahasa lain yang akan datang. `extensions` `groups` Fitur yang didukung meliputi kompatibilitas, penyumberan skema melalui metadata, pendaftaran otomatis skema, kompatibilitas IAM, dan kompresi ZLIB opsional untuk mengurangi penyimpanan dan transfer data. AWS Glue Registri Skema adalah nirserver dan bisa digunakan gratis.

Menggunakan sebuah skema sebagai kontrak format data antara produsen dan konsumen akan mengantarkan pada peningkatan tata kelola data, data berkualitas lebih tinggi, dan memungkinkan konsumen data untuk tahan terhadap perubahan hulu yang kompatibel.

Registri Skema memungkinkan sistem yang berbeda untuk berbagi skema untuk serialisasi dan de-serialisasi. Misalnya, anggaplah Anda memiliki produser dan konsumen data. Produsen mengetahui skemanya ketika menerbitkan data. Registri Skema memasok pen-serialisasi dan pen-deserialisasi untuk sistem tertentu seperti Amazon MSK atau Apache Kafka.

Untuk informasi selengkapnya, lihat Bagaimana Schema Registry bekerja.

Topik

Skema

Sebuah skema mendefinisikan struktur dan format catatan data. Sebuah skema adalah sebuah spesifikasi berversi untuk publikasi data yang handal, konsumsi, atau penyimpanan.

Dalam contoh ini skema untuk Avro, format dan strukturnya didefinisikan oleh tata letak dan nama bidang, dan format nama bidang didefinisikan oleh tipe data (misalnya, `string`, `int`).

```
{
    "type": "record",
    "namespace": "ABC_Organization",
    "name": "Employee",
    "fields": [
        {
            "name": "Name",
```

```
            "type": "string"
        },
        {
            "name": "Age",
            "type": "int"
        },
        {
            "name": "address",
            "type": {
                "type": "record",
                "name": "addressRecord",
                "fields": [
                    {
                        "name": "street",
                        "type": "string"
                    },
                    {
                        "name": "zipcode",
                        "type": "int"
                    }
                ]
            }
        }
    ]
}
```

Dalam contoh JSON Schema Draft-07 for JSON ini, format didefinisikan oleh Organisasi Skema JSON.

```
{
 "$id": "https://example.com/person.schema.json",
 "$schema": "http://json-schema.org/draft-07/schema#",
 "title": "Person",
 "type": "object",
 "properties": {
  "firstName": {
   "type": "string",
   "description": "The person's first name."
  },
  "lastName": {
   "type": "string",
   "description": "The person's last name."
  },
```

```
  "age": {
   "description": "Age in years which must be equal to or greater than zero.",
   "type": "integer",
   "minimum": 0
  }
 }
}
```

Dalam contoh ini untuk Protobuf, format didefinisikan oleh versi 2 dari bahasa Protocol Buffers (proto2).

```
syntax = "proto2";

package tutorial;

option java_multiple_files = true;
option java_package = "com.example.tutorial.protos";
option java_outer_classname = "AddressBookProtos";

message Person {
  optional string name = 1;
  optional int32 id = 2;
  optional string email = 3;

  enum PhoneType {
    MOBILE = 0;
    HOME = 1;
    WORK = 2;
  }

  message PhoneNumber {
    optional string number = 1;
    optional PhoneType type = 2 [default = HOME];
  }

  repeated PhoneNumber phones = 4;
}

message AddressBook {
  repeated Person people = 1;
}
```

Registrasi

Sebuah registri adalah sebuah kontainer logis dari skema. Registri memungkinkan Anda untuk mengatur skema Anda, serta mengelola kontrol akses untuk aplikasi Anda. Sebuah registri memiliki Amazon Resource Name (ARN) untuk memungkinkan Anda untuk mengorganisasi dan mengatur izin akses yang berbeda untuk skema operasi dalam registri.

Anda dapat menggunakan registri default atau membuat pendaftar baru sebanyak yang diperlukan.

Hirarki Registri Skema AWS Glue

- RegistryName: [string]
  - RegistryArn: [AWS ARN]
  - CreatedTime: [stempel waktu]
  - UpdatedTime: [stempel waktu]
- SchemaName: [string]
  - SchemaArn: [AWS ARN]
  - DataFormat: [Avro, Json, atau Protobuf]
  - Kompatibilitas: [misalnya. BACKWARD, BACKWARD_ALL, FORWARD, FORWARD_ALL, FULL, FULL_ALL, NONE, DISABLED]
  - Status: [misalnya. PENDING, AVAILABLE, DELETING]
  - SchemaCheckpoint: [bilangan bulat]
  - CreatedTime: [stempel waktu]
  - UpdatedTime: [stempel waktu]
- SchemaVersion: [string]
  - SchemaVersionNumber: [bilangan bulat]
  - Status: [misalnya. PENDING, AVAILABLE, DELETING, FAILURE]
  - SchemaDefinition: [string, Nilai: JSON]
  - CreatedTime: [stempel waktu]
- SchemaVersionMetadata: [daftar]
  - MetadataKey: [string]
  - MetadataInfo

- MetadataValue: [string]
- CreatedTime: [stempel waktu]

Pembuatan versi dan kompatibilitas skema

Setiap skema dapat memiliki beberapa versi. Versioning diatur oleh sebuah aturan kompatibilitas yang diterapkan pada sebuah skema. Permintaan untuk mendaftar versi skema baru diperiksa berdasarkan aturan ini oleh Registri Skema sebelum mereka dapat berhasil.

Sebuah versi skema yang ditandai sebagai pos pemeriksaan digunakan untuk menentukan kompatibilitas pendaftaran versi baru dari skema. Ketika sebuah skema pertama kali akan membuat pos pemeriksaan default akan menjadi versi pertama. Saat skema berkembang dengan lebih banyak versi, Anda dapat menggunakan CLI/SDK untuk mengubah pos pemeriksaan ke sebuah versi skema menggunakan API `UpdateSchema` yang mematuhi serangkaian pembatasan-pembatasan. Di konsol, mengedit definisi skema atau mode kompatibilitas, secara default akan mengubah pos pemeriksaan ke versi terbaru.

Mode kompatibilitas akan memungkinkan Anda untuk mengontrol bagaimana skema dapat atau tidak dapat berkembang dari waktu ke waktu. Mode ini membentuk kontrak antara aplikasi yang memproduksi dan mengkonsumsi data. Ketika versi baru dari sebuah skema dikirimkan ke registri, aturan kompatibilitas diterapkan ke nama skema yang digunakan untuk menentukan apakah versi baru dapat diterima. Ada 8 mode kompatibilitas: NONE, DISABLED, BACKWARD, BACKWARD_ALL, FORWARD, FORWARD_ALL, FULL, FULL_ALL.

Dalam format data Avro, bidang bisa opsional atau wajib. Sebuah bidang opsional adalah bidang di mana `Type` menyertakan nol. Bidang wajib tidak memiliki nol sebagai `Type`.

Dalam format data Protobuf, bidang dapat bersifat opsional (termasuk berulang) atau diperlukan dalam sintaks proto2, sementara semua bidang bersifat opsional (termasuk berulang) dalam sintaks proto3. Semua aturan kompatibilitas ditentukan berdasarkan pemahaman spesifikasi Protocol Buffer serta panduan dari dokumentasi Google Protocol Buffer.

- NONE: Tidak ada mode kompatibilitas yang berlaku. Anda dapat menggunakan pilihan ini dalam skenario pengembangan atau jika Anda tidak tahu mode kompatibilitas yang ingin Anda terapkan untuk skema. Setiap versi baru yang ditambahkan akan diterima tanpa menjalani pemeriksaan kompatibilitas terlebih dahulu.
- DISABLED: Pilihan kompatibilitas ini mencegah versioning untuk skema tertentu. Tidak ada versi baru yang dapat ditambahkan.

- BACKWARD: Pilihan kompatibilitas ini dianjurkan karena memungkinkan konsumen untuk membaca versi skema saat ini dan sebelumnya. Anda dapat menggunakan pilihan ini untuk memeriksa kompatibilitas terhadap versi skema sebelumnya saat Anda menghapus bidang atau menambahkan bidang opsional. Kasus penggunaan khas untuk BACKWARD adalah saat aplikasi Anda telah dibuat untuk skema terbaru.

  AVRO

  Sebagai contoh, anggaplah Anda memiliki sebuah skema yang ditentukan oleh nama depan (wajib), nama belakang (wajib), email (wajib), dan nomor telepon (opsional).

  Jika versi skema berikutnya menghapus bidang email wajib, ini akan berhasil mendaftar. Kompatibilitas BACKWARD mengharuskan konsumen untuk dapat membaca versi skema saat ini dan sebelumnya. Konsumen Anda akan dapat membaca skema baru karena bidang email tambahan dari pesan lama diabaikan.

  Jika Anda memiliki versi skema baru yang diusulkan yang menambahkan bidang wajib, misalnya, kode pos, maka hal ini tidak akan berhasil mendaftar dengan kompatibilitas BACKWARD. Konsumen Anda pada versi yang baru tidak akan dapat membaca pesan lama sebelum perubahan skema, karena mereka kehilangan kolom kode pos yang diperlukan. Namun demikian, jika bidang kode pos ditetapkan sebagai opsional dalam skema baru, maka versi yang diusulkan akan berhasil mendaftar karena konsumen dapat membaca skema lama tanpa bidang kode pos opsional.

  JSON

  Sebagai contoh, anggaplah Anda memiliki versi skema yang ditentukan berdasarkan nama depan (opsional), nama belakang (opsional), email (opsional) dan nomor telepon (opsional).

  Jika versi skema berikutnya menambahkan properti nomor telepon opsional, maka hal ini akan berhasil mendaftar selama versi skema asli tidak mengizinkan properti tambahan dengan menetapkan bidang `additionalProperties` ke SALAH. Kompatibilitas BACKWARD mengharuskan konsumen untuk dapat membaca versi skema saat ini dan sebelumnya. Konsumen Anda akan dapat membaca data yang dihasilkan dengan skema asli di mana properti nomor telepon tidak ada.

  Jika Anda memiliki versi skema baru yang diusulkan yang menambahkan properti nomor telepon opsional, maka hal ini tidak akan berhasil mendaftar dengan kompatibilitas BACKWARD ketika versi skema asli menetapkan bidang `additionalProperties` ke BETUL, yaitu memungkinkan setiap properti tambahan. Konsumen Anda pada versi baru tidak akan dapat membaca pesan lama

sebelum ada perubahan skema, karena mereka tidak dapat membaca data dengan properti nomor telepon dalam jenis yang berbeda, misalnya string bukan nomor.

PROTOBUF

Misalnya, anggap Anda memiliki versi skema yang ditentukan oleh Message `Person` dengan bidang (required), `first name` (required), `last name` (required), dan `email` `phone number` (opsional) di bawah sintaks proto2.

Mirip dengan skenario AVRO, jika versi skema Anda berikutnya menghapus `email` bidang yang diperlukan, ini akan berhasil mendaftar. Kompatibilitas BACKWARD mengharuskan konsumen untuk dapat membaca versi skema saat ini dan sebelumnya. Konsumen Anda akan dapat membaca skema baru karena `email` bidang tambahan dari pesan lama diabaikan.

Jika Anda memiliki versi skema baru yang diusulkan yang menambahkan bidang wajib, misalnya`zip code`, ini tidak akan berhasil mendaftar dengan kompatibilitas BACKWARD. Konsumen Anda pada versi baru tidak akan dapat membaca pesan lama sebelum skema berubah, karena mereka kehilangan `zip code` bidang yang diperlukan. Namun, jika `zip code` bidang ditetapkan sebagai opsional dalam skema baru, maka versi yang diusulkan akan berhasil mendaftar karena konsumen dapat membaca skema lama tanpa bidang opsional`zip code`.

Dalam kasus penggunaan gRPC, menambahkan layanan RPC baru atau metode RPC adalah perubahan yang kompatibel ke belakang. Misalnya, asumsikan Anda memiliki versi skema yang ditentukan oleh layanan RPC `MyService` dengan dua metode RPC dan. `Foo` `Bar`

Jika versi skema Anda berikutnya menambahkan metode RPC baru yang dipanggil`Baz`, ini akan berhasil mendaftar. Konsumen Anda akan dapat membaca data yang dihasilkan dengan skema asli sesuai dengan kompatibilitas BACKWARD karena metode `Baz` RPC yang baru ditambahkan adalah opsional.

Jika Anda memiliki versi skema baru yang diusulkan yang menghapus metode RPC yang ada`Foo`, ini tidak akan berhasil mendaftar dengan kompatibilitas BACKWARD. Konsumen Anda pada versi baru tidak akan dapat membaca pesan lama sebelum skema berubah, karena mereka tidak dapat memahami dan membaca data dengan metode RPC yang tidak ada dalam `Foo` aplikasi gRPC.

- BACKWARD_ALL: Pilihan kompatibilitas ini memungkinkan konsumen untuk membaca versi skema saat ini dan semua versi skema sebelumnya. Anda dapat menggunakan pilihan ini untuk memeriksa kompatibilitas terhadap semua versi skema sebelumnya saat Anda menghapus bidang atau menambahkan bidang opsional.

- FORWARD: Pilihan kompatibilitas ini memungkinkan konsumen untuk membaca versi skema saat ini dan versi skema berikutnya, tetapi tidak selalu versi yang lebih baru. Anda dapat menggunakan pilihan ini untuk memeriksa kompatibilitas terhadap versi skema terakhir ketika Anda menambahkan bidang atau menghapus bidang opsional. Kasus penggunaan khas untuk FORWARD adalah ketika aplikasi Anda telah dibuat untuk skema sebelumnya dan harus mampu memproses sebuah skema yang lebih baru.

  AVRO

  Sebagai contoh, anggaplah Anda memiliki sebuah versi skema yang ditentukan berdasarkan nama depan (wajib), nama belakang (wajib), email (opsional).

  Jika Anda memiliki versi skema baru yang menambahkan bidang wajib, misalnya nomor telepon, ini akan berhasil mendaftar. Kompatibilitas FORWARD mengharuskan konsumen untuk dapat membaca data yang dihasilkan dengan skema baru dengan menggunakan versi skema sebelumnya.

  Jika Anda memiliki versi skema yang diusulkan yang menghapus bidang nama pertama yang wajib, maka ini tidak akan berhasil mendaftar dengan kompatibilitas FORWARD. Konsumen Anda pada versi sebelumnya tidak akan dapat membaca skema yang diusulkan karena mereka tidak memiliki bidang nama pertama wajib. Namun, jika bidang nama pertama awalnya bersifat opsional, maka skema baru yang diusulkan akan berhasil mendaftar karena konsumen dapat membaca data berdasarkan skema baru yang tidak memiliki bidang nama depan opsional.

  JSON

  Sebagai contoh, anggaplah Anda memiliki versi skema yang ditentukan berdasarkan nama depan (opsional), nama belakang (opsional), email (opsional) dan nomor telepon (opsional).

  Jika Anda memiliki sebuah versi skema baru yang menghapus properti nomor telepon opsional, maka ini akan berhasil mendaftar selama versi skema baru tidak mengizinkan properti tambahan dengan menetapkan bidang `additionalProperties` ke SALAH. Kompatibilitas FORWARD mengharuskan konsumen untuk dapat membaca data yang dihasilkan dengan skema baru dengan menggunakan versi skema sebelumnya.

  Jika Anda memiliki versi skema yang diusulkan yang menghapus properti nomor telepon opsional, maka ini tidak akan berhasil mendaftar dengan kompatibilitas FORWARD ketika versi skema baru menetapkan bidang `additionalProperties` ke BETUL, yakni memungkinkan setiap properti tambahan. Konsumen Anda pada versi sebelumnya tidak akan dapat membaca skema

yang diusulkan karena mereka dapat memiliki properti nomor telepon dalam jenis yang berbeda, misalnya string bukan nomor.

PROTOBUF

Misalnya, anggap Anda memiliki versi skema yang ditentukan oleh Pesan `Person` dengan bidang (wajib), `first name` (wajib), `last name` `email` (opsional) di bawah sintaks proto2.

Mirip dengan skenario AVRO, jika Anda memiliki versi skema baru yang menambahkan bidang wajib, misalnya`phone number`, ini akan berhasil mendaftar. Kompatibilitas FORWARD mengharuskan konsumen untuk dapat membaca data yang dihasilkan dengan skema baru dengan menggunakan versi skema sebelumnya.

Jika Anda memiliki versi skema yang diusulkan yang menghapus `first name` bidang wajib, ini tidak akan berhasil mendaftar dengan kompatibilitas FORWARD. Konsumen Anda pada versi sebelumnya tidak akan dapat membaca skema yang diusulkan karena mereka kehilangan `first name` bidang yang diperlukan. Namun, jika `first name` bidang tersebut awalnya opsional, maka skema baru yang diusulkan akan berhasil mendaftar karena konsumen dapat membaca data berdasarkan skema baru yang tidak memiliki bidang opsional`first name`.

Dalam kasus penggunaan gRPC, menghapus layanan RPC atau metode RPC adalah perubahan yang kompatibel ke depan. Misalnya, asumsikan Anda memiliki versi skema yang ditentukan oleh layanan RPC `MyService` dengan dua metode RPC dan. `Foo` `Bar`

Jika versi skema Anda berikutnya menghapus metode RPC yang ada bernama`Foo`, ini akan berhasil mendaftar sesuai dengan kompatibilitas FORWARD karena konsumen dapat membaca data yang dihasilkan dengan skema baru dengan menggunakan versi sebelumnya. Jika Anda memiliki versi skema baru yang diusulkan yang menambahkan metode RPC`Baz`, ini tidak akan berhasil mendaftar dengan kompatibilitas FORWARD. Konsumen Anda pada versi sebelumnya tidak akan dapat membaca skema yang diusulkan karena mereka kehilangan metode RPC. `Baz`

- FORWARD_ALL: Pilihan kompatibilitas ini memungkinkan konsumen untuk membaca data yang ditulis oleh produsen dari setiap skema terdaftar baru. Anda dapat menggunakan pilihan ini ketika Anda harus menambahkan bidang atau menghapus bidang opsional, dan memeriksa kompatibilitas terhadap semua versi skema sebelumnya.
- FULL: Pilihan kompatibilitas ini memungkinkan konsumen untuk membaca data yang ditulis oleh produsen dengan menggunakan skema versi sebelumnya atau versi berikutnya, tetapi tidak versi yang lebih awal atau versi yang lebih baru. Anda dapat menggunakan pilihan ini untuk memeriksa

kompatibilitas terhadap versi skema terakhir ketika Anda menambahkan atau menghapus bidang opsional.

- FULL_ALL: Pilihan kompatibilitas ini memungkinkan konsumen untuk membaca data yang ditulis oleh produsen dengan menggunakan semua versi skema sebelumnya. Anda dapat menggunakan pilihan ini untuk memeriksa kompatibilitas terhadap semua versi skema sebelumnya saat Anda menambahkan atau menghapus bidang opsional.

Pustaka Serde sumber terbuka

AWS menyediakan pustaka Serde open-source sebagai kerangka kerja untuk membuat serial dan deserialisasi data. Desain sumber terbuka dari perpustakaan ini memungkinkan aplikasi dan kerangka kerja sumber terbuka umum untuk mendukung perpustakaan ini dalam proyek mereka.

Untuk detail lebih lanjut tentang bagaimana perpustakaan Serde bekerja, lihat Bagaimana Schema Registry bekerja.

Kuota Registri Skema

Kuota, juga disebut sebagai batas dalam AWS, adalah nilai maksimum untuk sumber daya, tindakan, dan item di AWS akun Anda. Berikut ini adalah batas lunak untuk Registri Skema di AWS Glue.

Pasangan nilai kunci metadata versi skema

Anda dapat memiliki hingga 10 pasangan nilai kunci SchemaVersion per AWS Wilayah.

Anda dapat melihat atau mengatur pasangan metadata kunci-nilai menggunakan QuerySchemaVersionMetadata tindakan (Python: query_schema_version_metadata) atau API PutSchemaVersionMetadata tindakan (Python: put_schema_version_metadata).

Berikut ini adalah batasan sulit untuk Schema Registry diAWS Glue.

Registrasi

Anda dapat memiliki hingga 100 pendaftar per AWS Wilayah untuk akun ini.

SchemaVersion

Anda dapat memiliki hingga 10000 versi skema per AWS Wilayah untuk akun ini.

Setiap skema baru membuat versi skema baru, sehingga Anda secara teoritis dapat memiliki hingga 10000 skema per akun per wilayah, jika setiap skema hanya memiliki satu versi.

Beban skema

Ada batas ukuran 170KB untuk beban skema.

Bagaimana Schema Registry bekerja

Bagian ini menjelaskan bagaimana proses serialisasi dan deserialisasi dalam pekerjaan Registri Skema.

1. Mendaftar skema: Jika skema belum ada di registri, maka skema dapat didaftarkan dengan nama skema yang sama dengan nama tujuan (misalnya, test_topic, test_stream, prod_firehose) atau produsen dapat memberikan nama kustom untuk skema tersebut. Produsen juga dapat menambahkan pasangan kunci-nilai ke skema sebagai metadata, seperti sumber: msk_kafka_topic_A, atau menerapkan tag AWS untuk skema pada saat membuat skema. Setelah sebuah skema terdaftar, Registri Skema mengembalikan skema ID versi ke pen-serialisasi. Jika skema ada tetapi pen-serialisasi menggunakan versi baru yang tidak ada, maka Registri Skema akan memeriksa referensi skema untuk aturan kompatibilitas untuk memastikan bahwa versi baru sudah kompatibel sebelum mendaftarkannya sebagai sebuah versi baru.

   Ada dua metode untuk mendaftarkan sebuah skema: registrasi manual dan registrasi otomatis. Anda dapat mendaftarkan sebuah skema secara manual melalui konsol AWS Glue atau CLI/SDK.

   Ketika pendaftaran otomatis diaktifkan dalam pengaturan serializer, pendaftaran otomatis skema akan dilakukan. Jika `REGISTRY_NAME` tidak disediakan dalam konfigurasi produsen, maka pendaftaran otomatis akan mendaftarkan versi skema baru di bawah registri default (default-registry). Lihat Instalasi SerDe Perpustakaan untuk informasi tentang menentukan properti pendaftaran otomatis.
2. Serializer memvalidasi catatan data terhadap skema: Ketika aplikasi yang memproduksi data telah terdaftar skemanya, serializer Registri Skema memvalidasi rekaman yang dihasilkan oleh aplikasi terstruktur dengan bidang dan jenis data yang cocok dengan skema terdaftar. Jika skema catatan tidak cocok dengan skema terdaftar, maka serializer akan mengembalikan pengecualian dan aplikasi akan gagal untuk memberikan catatan ke tujuan.

   Jika tidak ada skema yang ada dan jika nama skema tidak disediakan melalui konfigurasi produsen, maka skema akan dibuat dengan nama yang sama dengan nama topik (jika Apache Kafka atau Amazon MSK) atau nama pengaliran (jika Kinesis Data Streams).

   Setiap catatan memiliki sebuah definisi skema dan data. Definisi skema tersebut di-kueri terhadap skema dan versi yang ada di Registri Skema.

Secara default, produsen meng-cache definisi skema dan ID versi skema dari skema terdaftar. Jika definisi versi skema catatan tidak cocok dengan apa yang tersedia dalam cache, maka produsen akan mencoba untuk memvalidasi skema dengan Registri Skema. Jika versi skema valid, maka ID versi dan definisi skema akan di-cache secara lokal pada produsen.

Anda dapat menyesuaikan periode cache default (24 jam) dalam properti produsen opsional di langkah #3 dari Instalasi SerDe Perpustakaan.

3. Serialisasi dan kirimkan catatan: Jika rekaman sesuai dengan skema, serializer menghiasi setiap rekaman dengan ID versi skema, membuat serial rekaman berdasarkan format data yang dipilih (AVRO, JSON, Protobuf, atau format lain segera hadir), mengompres catatan (konfigurasi produsen opsional), dan mengirimkannya ke tujuan.
4. Konsumen melakukan deserialisasi data: Konsumen membaca data ini menggunakan perpustakaan deserializer Registri Skema yang mem-parsing ID versi skema dari muatan catatan.
5. Deserializer dapat meminta skema dari Registri Skema: Jika ini adalah pertama kalinya deserializer melihat catatan dengan ID versi skema tertentu, dengan menggunakan ID versi skema, deserializer akan meminta skema dari Registri Skema dan meng-cache skema tersebut secara lokal pada konsumen. Jika Registri Skema tidak dapat melakukan deserialisasi catatan, konsumen dapat mencatat log data dari catatan dan melanjutkan, atau menghentikan aplikasi.
6. Deserializer menggunakan skema untuk melakukan deserialisasi catatan: Ketika deserializer mengambil ID versi skema dari Registri Skema, deserializer melakukan dekompresi catatan (jika catatan yang dikirim oleh produsen dikompresi) dan menggunakan skema tersebut untuk melakukan deserialisasi catatan. Aplikasi sekarang memproses catatan tersebut.

> **Note**
>
> Enkripsi: Klien Anda berkomunikasi dengan Registri Skema melalui panggilan API yang mengenkripsi data dalam transit dengan menggunakan enkripsi TLS melalui HTTPS. Skema yang disimpan dalam Registri Skema selalu dienkripsi secara at rest dengan menggunakan kunci AWS Key Management Service (AWS KMS) terkelola layanan.

> **Note**
>
> Otorisasi Pengguna: Registri Skema mendukung kebijakan IAM berbasis identitas.

Memulai dengan Schema Registry

Bagian berikut memberikan gambaran umum dan memandu Anda dalam menyiapkan dan menggunakan Registri Skema. Untuk informasi tentang konsep dan komponen Registri Skema, lihat Registri Skema AWS Glue.

Topik

Instalasi SerDe Perpustakaan

> **Note**
>
> Prasyarat: Sebelum menyelesaikan langkah-langkah berikut, Anda harus memiliki sebuah klaster Amazon Managed Streaming for Apache Kafka (Amazon MSK) atau Apache Kafka yang berjalan. Produsen dan konsumen Anda harus berjalan pada Java 8 atau yang lebih tinggi.

SerDe Pustaka menyediakan kerangka kerja untuk serialisasi dan deserialisasi data.

Anda akan menginstal serializer sumber terbuka untuk aplikasi Anda yang menghasilkan data (secara kolektif “"-serializer"). Serializer menangani serialisasi, kompresi, dan interaksi dengan Registri Skema. Serializer secara otomatis mengekstrak skema dari catatan yang ditulis ke

tujuan yang kompatibel dengan Registri Skema, seperti Amazon MSK. Demikian juga, Anda akan menginstal deserializer sumber terbuka pada aplikasi Anda yang mengkonsumsi data.

Untuk menginstal perpustakaan pada produsen dan konsumen:

1. Di dalam file pom.xml baik dari produsen dan konsumen, tambahkan dependensi ini melalui kode di bawah ini:

```
<dependency>
    <groupId>software.amazon.glue</groupId>
    <artifactId>schema-registry-serde</artifactId>
    <version>1.1.5</version>
</dependency>
```

   Atau, Anda dapat mengkloning Repositori Github Registri Skema AWS Glue.

2. Siapkan produsen Anda dengan properti yang diperlukan ini:

```
props.put(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,
 StringSerializer.class.getName()); // Can replace StringSerializer.class.getName())
 with any other key serializer that you may use
props.put(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,
 GlueSchemaRegistryKafkaSerializer.class.getName());
props.put(AWSSchemaRegistryConstants.AWS_REGION, "us-east-2");
properties.put(AWSSchemaRegistryConstants.DATA_FORMAT, "JSON"); // OR "AVRO"
```

   Jika tidak ada skema yang ada, maka pendaftaran otomatis harus Anda aktifkan (langkah berikutnya). Jika Anda memiliki skema yang ingin Anda terapkan, maka ganti "my-schema" dengan nama skema Anda. Selain itu, "registry-name" harus disediakan jika skema pendaftaran otomatis dimatikan. Jika skema dibuat menggunakan "default-registry", maka nama registri dapat dihilangkan.

3. (Opsional) Mengatur salah satu properti produsen opsional ini. Untuk deskripsi properti terperinci, lihat ReadMe file.

```
props.put(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING, "true"); // If
 not passed, uses "false"
props.put(AWSSchemaRegistryConstants.SCHEMA_NAME, "my-schema"); // If not passed,
 uses transport name (topic name in case of Kafka, or stream name in case of Kinesis
 Data Streams)
props.put(AWSSchemaRegistryConstants.REGISTRY_NAME, "my-registry"); // If not passed,
 uses "default-registry"
```

```
props.put(AWSSchemaRegistryConstants.CACHE_TIME_TO_LIVE_MILLIS, "86400000"); // If
 not passed, uses 86400000 (24 Hours)
props.put(AWSSchemaRegistryConstants.CACHE_SIZE, "10"); // default value is 200
props.put(AWSSchemaRegistryConstants.COMPATIBILITY_SETTING, Compatibility.FULL); //
 Pass a compatibility mode. If not passed, uses Compatibility.BACKWARD
props.put(AWSSchemaRegistryConstants.DESCRIPTION, "This registry is used for several
 purposes."); // If not passed, constructs a description
props.put(AWSSchemaRegistryConstants.COMPRESSION_TYPE,
 AWSSchemaRegistryConstants.COMPRESSION.ZLIB); // If not passed, records are sent
 uncompressed
```

Pendaftaran otomatis mendaftarkan versi skema dengan menggunakan registri default ("default-registry"). Jika SCHEMA_NAME tidak ditentukan dalam langkah sebelumnya, maka nama topik disimpulkan sebagai SCHEMA_NAME.

Lihat Pembuatan versi dan kompatibilitas skema untuk informasi lebih lanjut tentang mode kompatibilitas.

4. Siapkan konsumen Anda dengan properti yang diperlukan berikut ini:

```
props.put(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,
 StringDeserializer.class.getName());
props.put(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,
 GlueSchemaRegistryKafkaDeserializer.class.getName());
props.put(AWSSchemaRegistryConstants.AWS_REGION, "us-east-2"); // Pass an Wilayah AWS
props.put(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.GENERIC_RECORD.getName()); // Only required for AVRO data format
```

5. (Opsional) Mengatur properti konsumen opsional ini. Untuk deskripsi properti terperinci, lihat ReadMe file.

```
properties.put(AWSSchemaRegistryConstants.CACHE_TIME_TO_LIVE_MILLIS, "86400000"); //
 If not passed, uses 86400000
props.put(AWSSchemaRegistryConstants.CACHE_SIZE, "10"); // default value is 200
props.put(AWSSchemaRegistryConstants.SECONDARY_DESERIALIZER,
 "com.amazonaws.services.schemaregistry.deserializers.external.ThirdPartyDeserializer"); //
 For migration fall back scenario
```

Menggunakan AWS CLI untuk API Registri Skema AWS Glue

Untuk menggunakan AWS CLI untuk API Registri Skema AWS Glue, pastikan untuk memperbarui AWS CLI ke versi terbaru.

Membuat registri

Anda dapat menggunakan registri default atau membuat banyak registri baru yang diperlukan dengan menggunakan API AWS Glue atau konsol AWS Glue.

API AWS Glue

Anda dapat menggunakan langkah-langkah ini untuk melakukan tugas ini menggunakan API AWS Glue.

Untuk menambahkan sebuah registri baru, gunakan API CreateRegistry tindakan (Python: create_registry). Menentukan `RegistryName` sebagai nama registri yang akan dibuat, dengan panjang maksimal 255 karakter, hanya berisi huruf, angka, tanda hubung, garis bawah, tanda dolar, atau tanda hash.

Tentukan `Description` sebagai string yang panjangnya tidak lebih dari 2048 byte, cocok dengan pola string multi-baris alamat URI.

Opsional, menentukan satu atau beberapa `Tags` untuk registri Anda, sebagai sebuah susunan peta pasangan nilai kunci.

```
aws glue create-registry --registry-name registryName1 --description description
```

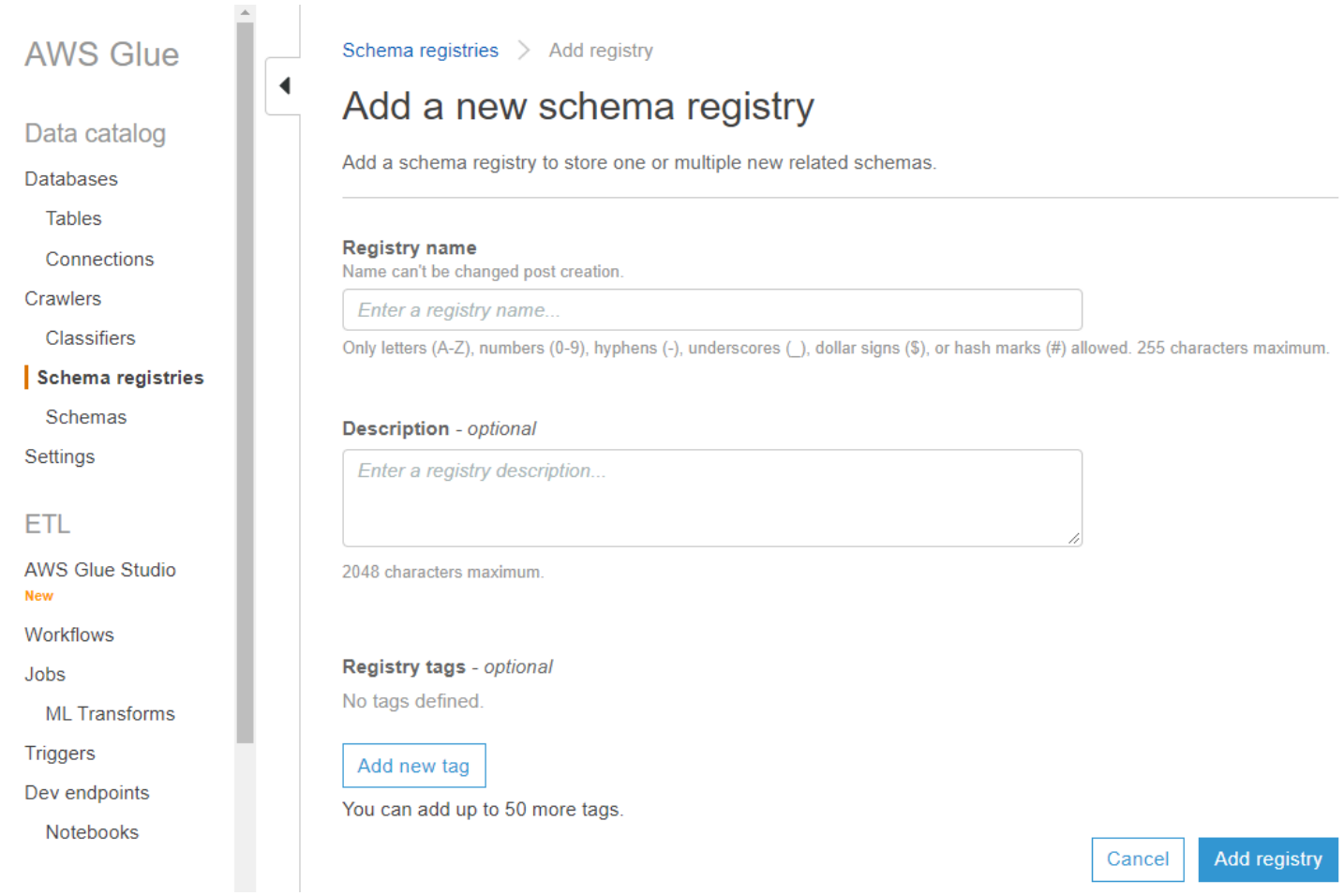
Ketika registri Anda sudah dibuat, Amazon Resource Name (ARN) ditetapkan untuknya, yang dapat Anda lihat di `RegistryArn` dari respons API. Setelah Anda membuat sebuah registri, buatlah satu atau beberapa skema untuk registri tersebut.

Konsol AWS Glue

Untuk menambahkan sebuah registri baru di konsol AWS Glue:

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada Katalog data, pilih Registri Skema.
3. Pilih Tambah registri.

4. Masukkan sebuah Nama Registri untuk registri tersebut, yang terdiri dari huruf, angka, tanda hubung, atau garis bawah. Nama ini tidak dapat diubah.
5. Masukkan Deskripsi (opsional) untuk registri.
6. Opsional, terapkan satu atau beberapa tag ke registri Anda. Pilih Tambahkan tag dan tentukan Kunci tag dan Nilai tag opsional.
7. Pilih Tambah registri.



Ketika registri Anda dibuat, Amazon Resource Name (ARN) ditetappkan padanya, yang dapat Anda lihat dengan memilih registri dari daftar di Registri Skema. Setelah Anda membuat sebuah registri, buatlah satu atau beberapa skema untuk registri tersebut.

Berurusan dengan catatan tertentu (JAVA POJO) untuk JSON

Anda dapat menggunakan sebuah plain old Java object (POJO) dan memberikan objek tersebut sebagai catatan. Ini mirip dengan maksud dari catatan spesifik di AVRO. mbknor-jackson-

jsonschemaDapat menghasilkan skema JSON untuk POJO yang dilewati. Perpustakaan ini juga dapat memasukkan informasi tambahan dalam skema JSON.

Perpustakaan Registri Skema AWS Glue menggunakan bidang "ClassName" yang dimasukkan dalam skema untuk memberikan nama kelas terklasifikasi sepenuhnya. Bidang "ClassName" digunakan oleh deserializer untuk melakukan deserialisasi ke objek dari kelas itu.

```
 Example class :

@JsonSchemaDescription("This is a car")
@JsonSchemaTitle("Simple Car Schema")
@Builder
@AllArgsConstructor
@EqualsAndHashCode
// Fully qualified class name to be added to an additionally injected property
// called className for deserializer to determine which class to deserialize
// the bytes into
@JsonSchemaInject(
        strings = {@JsonSchemaString(path = "className",
                value =
 "com.amazonaws.services.schemaregistry.integrationtests.generators.Car")}
)
// List of annotations to help infer JSON Schema are defined by https://github.com/
mbknor/mbknor-jackson-jsonSchema
public class Car {
    @JsonProperty(required = true)
    private String make;

    @JsonProperty(required = true)
    private String model;

    @JsonSchemaDefault("true")
    @JsonProperty
    public boolean used;

    @JsonSchemaInject(ints = {@JsonSchemaInt(path = "multipleOf", value = 1000)})
    @Max(200000)
    @JsonProperty
    private int miles;

    @Min(2000)
    @JsonProperty
    private int year;
```

```
    @JsonProperty
    private Date purchaseDate;

    @JsonProperty
    @JsonFormat(shape = JsonFormat.Shape.NUMBER)
    private Date listedDate;

    @JsonProperty
    private String[] owners;

    @JsonProperty
    private Collection<Float> serviceChecks;

    // Empty constructor is required by Jackson to deserialize bytes
    // into an Object of this class
    public Car() {}
}
```

Membuat skema

Anda dapat membuat sebuah skema menggunakan API AWS Glue atau konsol AWS Glue.

API AWS Glue

Anda dapat menggunakan langkah-langkah ini untuk melakukan tugas ini menggunakan API AWS Glue.

Untuk menambahkan sebuah skema baru, gunakan API CreateSchema tindakan (Python: create_schema).

Menentukan sebuah struktur `RegistryId` untuk menunjukkan registri untuk skema tersebut. Atau, menghilangkan `RegistryId` untuk menggunakan registri default.

Menentukan `SchemaName` yang terdiri dari huruf, angka, tanda hubung, atau garis bawah, dan `DataFormat` sebagai **`AVRO`** atau **`JSON`**. `DataFormat` setelah ditetapkan pada skema tidak dapat diubah.

Menentukan mode `Compatibility`:

- Backward (disarankan) — Konsumen dapat membaca versi saat ini dan sebelumnya.
- Backward all — Konsumen dapat membaca versi terkini dan semua versi sebelumnya.

- Forward — Konsumen dapat membaca baik versi saat ini dan versi berikutnya.
- Forward all — Konsumen dapat membaca baik versi saat ini dan semua versi berikutnya.
- Full — Kombinasi dari Backward dan Forward.
- Full all — Kombinasi dari Backward all dan Forward all.
- None — Tidak ada pemeriksaan kompatibilitas yang dilakukan.
- Disabled — Mencegah versioning apapun untuk skema ini.

Opsional, tentukan `Tags` untuk skema Anda.

Tentukan a `SchemaDefinition` untuk menentukan skema dalam format data Avro, JSON, atau Protobuf. Lihat contoh-contohnya.

Untuk format data Avro:

```
aws glue create-schema --registry-id RegistryName="registryName1" --schema-name
 testschema --compatibility NONE --data-format AVRO --schema-definition "{\"type\":
 \"record\", \"name\": \"r1\", \"fields\": [ {\"name\": \"f1\", \"type\": \"int\"},
 {\"name\": \"f2\", \"type\": \"string\"} ]}"
```

```
aws glue create-schema --registry-id RegistryArn="arn:aws:glue:us-
east-2:901234567890:registry/registryName1" --schema-name testschema --compatibility
 NONE --data-format AVRO  --schema-definition "{\"type\": \"record\", \"name\": \"r1\",
 \"fields\": [ {\"name\": \"f1\", \"type\": \"int\"}, {\"name\": \"f2\", \"type\":
 \"string\"} ]}"
```

Untuk format data JSON:

```
aws glue create-schema --registry-id RegistryName="registryName" --schema-name
 testSchemaJson --compatibility NONE --data-format JSON --schema-definition "{\"$schema
\": \"http://json-schema.org/draft-07/schema#\",\"type\":\"object\",\"properties\":
{\"f1\":{\"type\":\"string\"}}}"
```

```
aws glue create-schema --registry-id RegistryArn="arn:aws:glue:us-
east-2:901234567890:registry/registryName" --schema-name testSchemaJson --compatibility
 NONE --data-format JSON --schema-definition "{\"$schema\": \"http://json-schema.org/
draft-07/schema#\",\"type\":\"object\",\"properties\":{\"f1\":{\"type\":\"string\"}}}"
```

Untuk format data Protobuf:

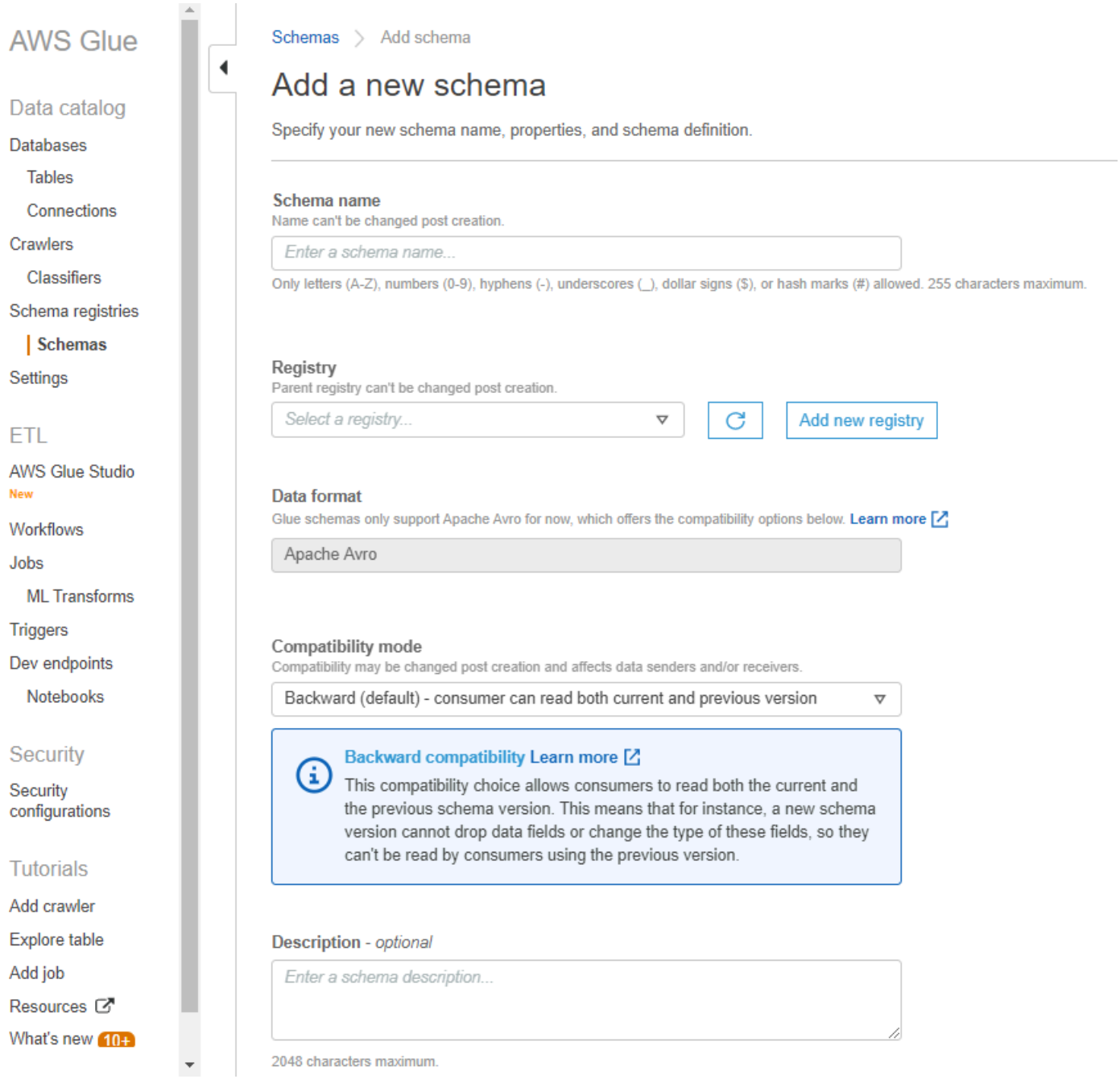
```
aws glue create-schema --registry-id RegistryName="registryName" --schema-name
 testSchemaProtobuf --compatibility NONE --data-format PROTOBUF --schema-definition
 "syntax = \"proto2\";package org.test;message Basic { optional int32 basic = 1;}"
```

```
aws glue create-schema --registry-id RegistryArn="arn:aws:glue:us-
east-2:901234567890:registry/registryName" --schema-name testSchemaProtobuf
 --compatibility NONE --data-format PROTOBUF --schema-definition "syntax =
 \"proto2\";package org.test;message Basic { optional int32 basic = 1;}"
```

Konsol AWS Glue

Untuk menambahkan sebuah skema baru menggunakan konsol AWS Glue:

1. Masuk ke Konsol Manajemen AWS dan buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada Katalog data, pilih Skema.
3. Pilih Tambahkan skema.
4. Masukkan Nama skema, terdiri dari huruf, angka, tanda hubung, garis bawah, tanda dolar, atau tanda hash. Nama ini tidak dapat diubah.
5. Pilih Registri di mana skema akan disimpan dari menu drop-down. Registri induk tidak dapat diubah setelah dibuat.
6. Biarkan Format data sebagai Apache Avro atau JSON. Format ini berlaku untuk semua versi skema ini.
7. Pilih Mode kompatibilitas.
   - Backward (disarankan) — Penerima dapat membaca versi saat ini dan sebelumnya.
   - Backward all — Penerima dapat membaca versi terkini dan semua versi sebelumnya.
   - Forward — Pengirim dapat menulis baik versi terkini dan versi sebelumnya.
   - Forward all — Pengirim dapat menulis versi terkini dan semua versi sebelumnya.
   - Full — Kombinasi dari Backward dan Forward.
   - Full all — Kombinasi dari Backward All dan Forward All.
   - None — Tidak ada pemeriksaan kompatibilitas yang dilakukan.
   - Disabled — Mencegah versioning apapun untuk skema ini.
8. Masukkan Deskripsi opsional untuk registri yang terdiri hingga 250 karakter.

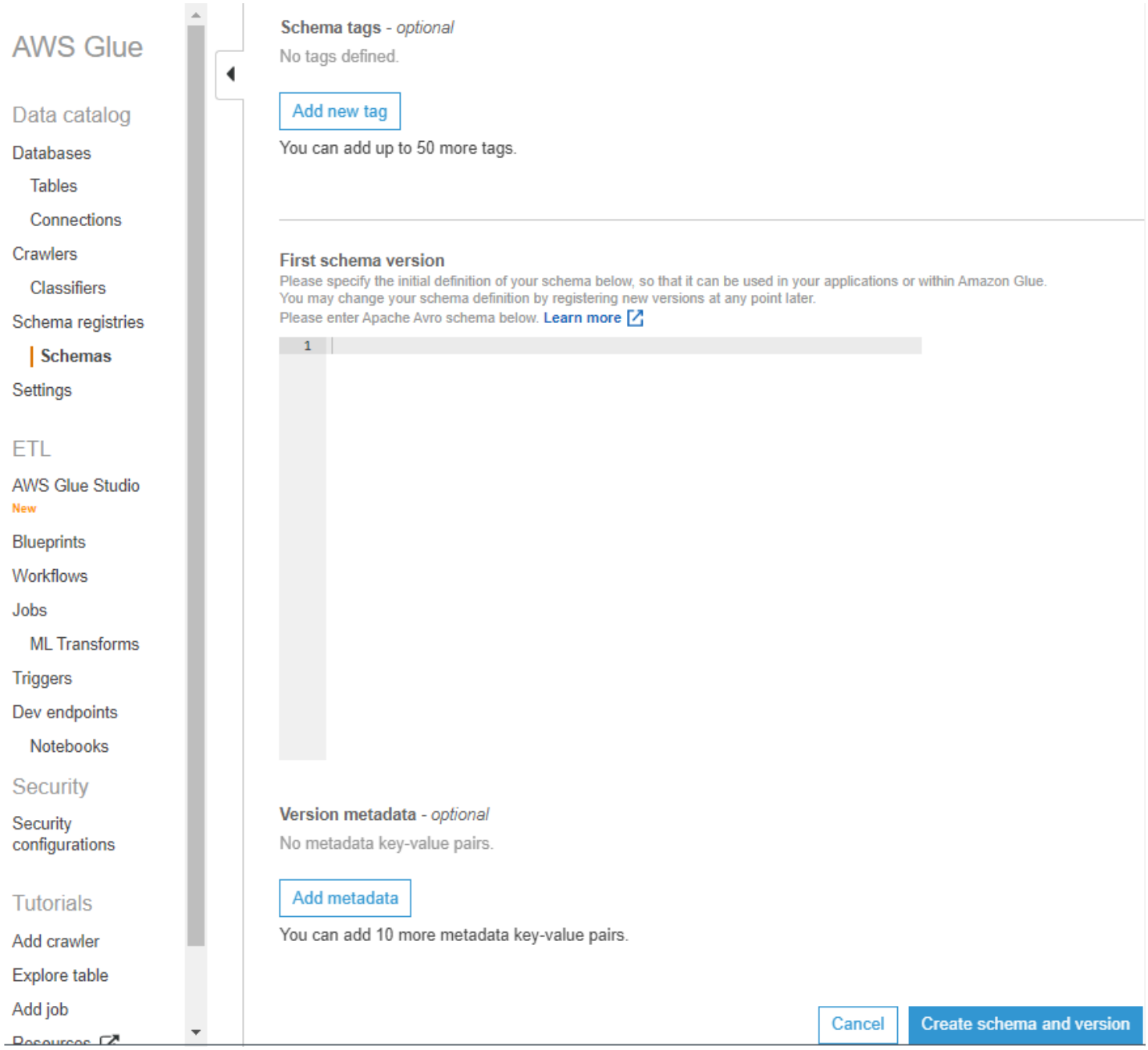


9. Opsional, terapkan satu atau beberapa tag untuk skema Anda. Pilih Tambahkan tag dan tentukan Kunci tag dan Nilai tag opsional.
10. Pada kotak Versi skema pertama, masukkan atau tempel skema awal Anda.

    Untuk format Avro, lihat Bekerja dengan format data Avro

    Untuk format JSON, lihat Bekerja dengan format data JSON

11. Opsional, pilih Tambahkan metadata untuk menambahkan metadata versi untuk membuat anotasi atau mengklasifikasikan versi skema Anda.
12. Pilih Buat skema dan versi.



Skema dibuat dan muncul dalam daftar pada Skema.

Bekerja dengan format data Avro

Avro menyediakan layanan serialisasi data dan pertukaran data. Avro menyimpan definisi data dalam format JSON sehingga mudah untuk dibaca dan ditafsirkan. Data itu sendiri disimpan dalam format biner.

Untuk informasi tentang cara mendefinisikan skema Apache Avro, lihat Spesifikasi Apache Avro.

Bekerja dengan format data JSON

Data dapat diserialisasi dengan format JSON. Format Skema JSON mendefinisikan standar untuk format Skema JSON.

Memperbarui skema atau registri

Setelah dibuat Anda dapat mengedit skema Anda, versi skema, atau registri Anda.

Memperbarui registri

Anda dapat memperbarui sebuah registri dengan menggunakan API AWS Glue atau konsol AWS Glue. Nama registri yang sudah ada tidak dapat Anda edit. Anda dapat mengedit deskripsi untuk sebuah registri.

API AWS Glue

Untuk memperbarui registri yang sudah ada, gunakan API UpdateRegistry tindakan (Python: update_registry).

Menentukan sebuah struktur `RegistryId` untuk menunjukkan registri yang ingin Anda perbarui. Berikan sebuah `Description` untuk mengubah deskripsi untuk sebuah registri.

```
aws glue update-registry --description updatedDescription --registry-id
 RegistryArn="arn:aws:glue:us-east-2:901234567890:registry/registryName1"
```

Konsol AWS Glue

Untuk memperbarui sebuah registri menggunakan konsol AWS Glue:

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.

2. Di panel navigasi, pada Katalog data, pilih Registri Skema.
3. Pilih sebuah registri dari daftar registri, dengan mencentang kotaknya.
4. Di menu Tindakan, pilih Edit registri.

Memperbarui skema

Anda dapat memperbarui pengaturan deskripsi atau kompatibilitas untuk sebuah skema.

Untuk memperbarui skema yang sudah ada, gunakan API UpdateSchema tindakan (Python: update_schema).

Menentukan sebuah struktur `SchemaId` untuk menunjukkan skema yang ingin Anda perbarui. Salah satu dari `VersionNumber` atau `Compatibility` harus disediakan.

Contoh kode 11:

```
aws glue update-schema --description testDescription --schema-id
 SchemaName="testSchema1",RegistryName="registryName1" --schema-version-number
 LatestVersion=true --compatibility NONE
```

```
aws glue update-schema --description testDescription --schema-id
 SchemaArn="arn:aws:glue:us-east-2:901234567890:schema/registryName1/testSchema1" --
schema-version-number LatestVersion=true --compatibility NONE
```

Menambahkan versi skema

Ketika Anda menambahkan sebuah versi skema, Anda akan perlu untuk membandingkan versi untuk memastikan skema baru akan diterima.

Untuk menambahkan versi baru ke sebuah skema yang ada, gunakan API RegisterSchemaVersion tindakan (Python: register_schema_version).

Menentukan sebuah struktur `SchemaId` untuk menunjukkan skema yang Anda ingin tambah versinya, dan sebuah `SchemaDefinition` untuk menentukan skemanya.

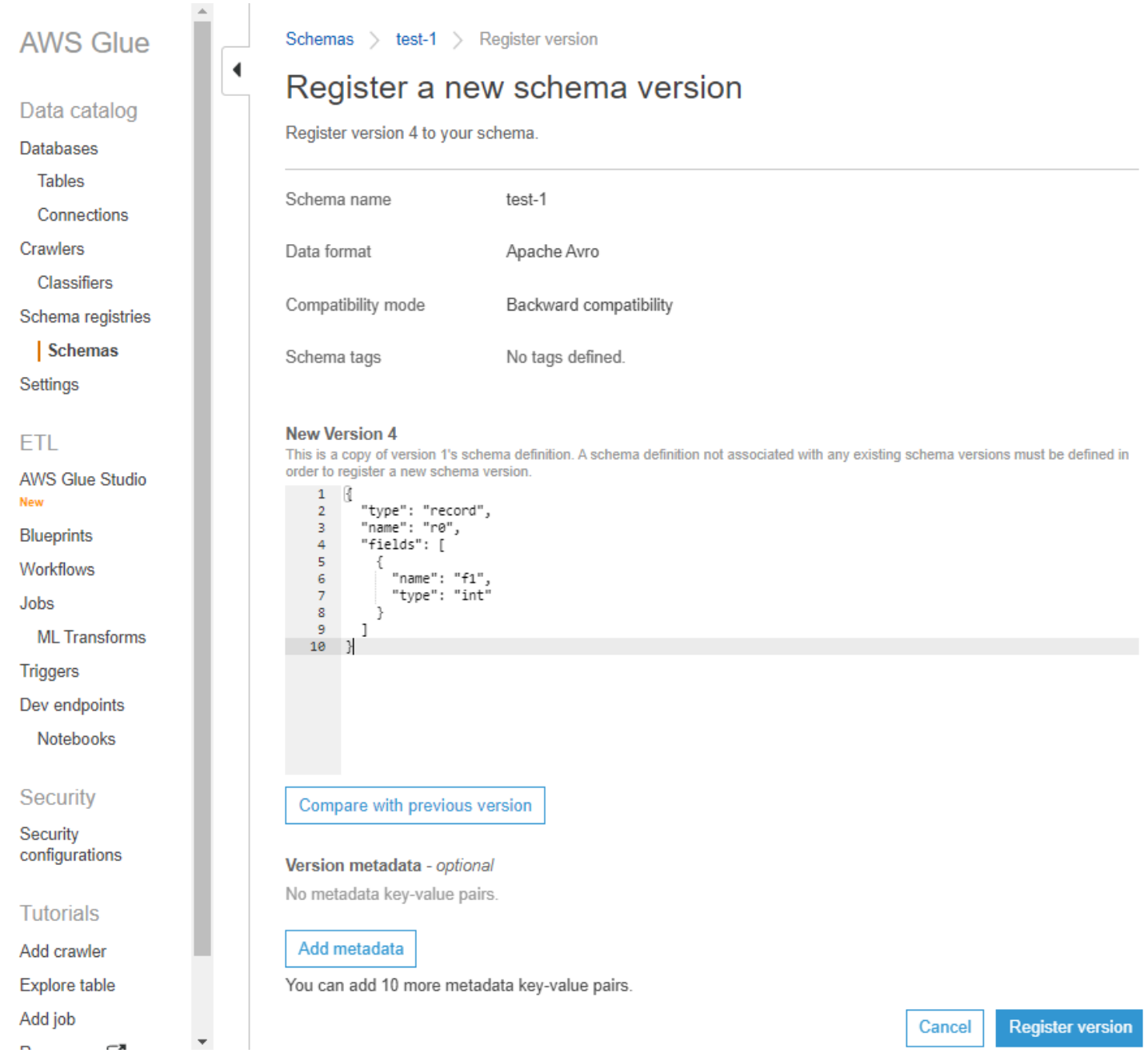
Contoh kode 12:

```
aws glue register-schema-version --schema-definition "{\"type\": \"record\", \"name\":
 \"r1\", \"fields\": [ {\"name\": \"f1\", \"type\": \"int\"}, {\"name\": \"f2\", \"type
```

```
\": \"string\"} ]}" --schema-id SchemaArn="arn:aws:glue:us-east-1:901234567890:schema/
registryName/testschema"
```

```
aws glue register-schema-version --schema-definition "{\"type\": \"record\", \"name\":
 \"r1\", \"fields\": [ {\"name\": \"f1\", \"type\": \"int\"}, {\"name\": \"f2\", \"type
\": \"string\"} ]}" --schema-id SchemaName="testschema",RegistryName="testregistry"
```

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada Katalog data, pilih Skema.
3. Pilih skema dari daftar skema, dengan mencentang kotaknya.
4. Pilih satu atau beberapa skema dari daftar, dengan mencentang kotak.
5. Pada menu Tindakan, pilih Daftarkan versi baru.
6. Pada kotak Versi baru, masukkan atau tempel skema baru Anda.
7. Pilih Bandingkan dengan versi sebelumnya untuk melihat perbedaan dengan versi skema sebelumnya.
8. Opsional, pilih Tambahkan metadata untuk menambahkan metadata versi untuk membuat anotasi atau mengklasifikasikan versi skema Anda. Masukkan Kunci dan Nilai opsional.
9. Pilih Versi Registri.



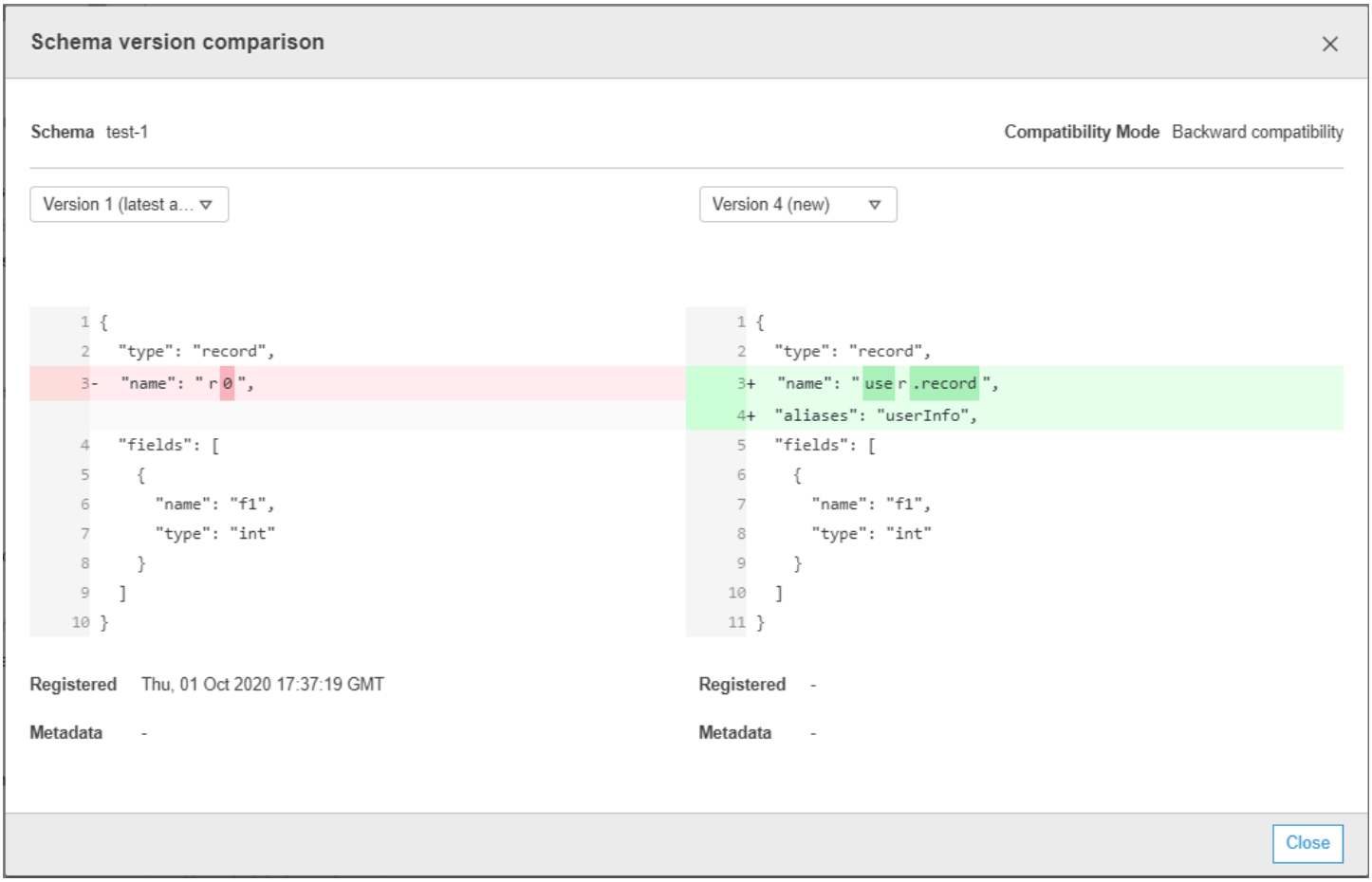
Versi skema muncul pada daftar versi. Jika versi mengubah mode kompatibilitas, maka versi akan ditandai sebagai sebuah pos pemeriksaan.

Contoh perbandingan versi skema

Ketika Anda memilih untuk Bandingkan dengan versi sebelumnya, maka Anda akan melihat versi sebelumnya dan versi baru ditampilkan bersama-sama. Informasi yang diubah akan disorot sebagai berikut:

- Kuning: menunjukkan ada perubahan informasi.
- Hijau: menunjukkan ada penambahan konten dalam versi terbaru.
- Merah: menunjukkan ada penghapusan konten dalam versi terbaru.

Anda juga dapat membandingkan dengan versi yang sebelumnya.



Menghapus skema atau registri

Menghapus sebuah skema, versi skema, atau registri adalah tindakan permanen yang tidak dapat dibatalkan.

Menghapus skema

Anda mungkin ingin menghapus sebuah skema ketika itu akan tidak lagi digunakan dalam sebuah registri, dengan menggunakan AWS Management Console, atau API DeleteSchema tindakan (Python: delete_schema).

Menghapus satu atau beberapa skema adalah tindakan permanen yang tidak dapat dibatalkan. Pastikan bahwa skema atau skema-skema itu tidak lagi diperlukan.

Untuk menghapus sebuah skema dari registri, panggil API DeleteSchema tindakan (Python: delete_schema), dengan menentukan struktur `SchemaId` untuk mengidentifikasi skema.

Sebagai contoh:

```
aws glue delete-schema --schema-id SchemaArn="arn:aws:glue:us-
east-2:901234567890:schema/registryName1/schemaname"
```

```
aws glue delete-schema --schema-id SchemaName="TestSchema6-
deleteschemabyname",RegistryName="default-registry"
```

Konsol AWS Glue

Untuk menghapus sebuah skema dari konsol AWS Glue:

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada Katalog data, pilih Registri Skema.
3. Pilih registri yang berisi skema Anda dari daftar registri.
4. Pilih satu atau beberapa skema dari daftar, dengan mencentang kotak.
5. Di menu Tindakan, pilih Hapus skema.
6. Masukkan teks **`Delete`** di bidang untuk mengonfirmasi penghapusan.
7. Pilih Hapus.

Skema yang Anda tentukan akan dihapus dari registri.

Menghapus versi skema

Karena skema terakumulasi dalam registri, Anda mungkin ingin menghapus versi skema yang tidak diinginkan menggunakan AWS Management Console, atau API DeleteSchemaVersions tindakan (Python: delete_schema_versions). Menghapus satu atau beberapa versi skema adalah tindakan permanen yang tidak dapat dibatalkan. Pastikan bahwa versi skema tersebut tidak diperlukan lagi.

Ketika menghapus versi skema, perhatikan batasan-batasan berikut:

- Anda tidak dapat menghapus versi yang dikenai pemeriksaan.
- Rentang versi berdekatan tidak boleh lebih dari 25.
- Versi skema terbaru tidak boleh dalam status tertunda.

Menentukan struktur `SchemaId` untuk mengidentifikasi skema, dan menentukan `Versions` sebagai rentang versi yang akan dihapus. Untuk informasi lebih lanjut tentang menentukan versi atau rentang versi, lihat DeleteRegistry tindakan (Python: delete_registry). Versi skema yang Anda tentukan akan dihapus dari registri.

Memanggil API ListSchemaVersions tindakan (Python: list_schema_versions) setelah panggilan ini akan mencantumkan status versi yang dihapus.

Sebagai contoh:

```
aws glue delete-schema-versions --schema-id
 SchemaName="TestSchema6",RegistryName="default-registry" --versions "1-1"
```

```
aws glue delete-schema-versions --schema-id SchemaArn="arn:aws:glue:us-
east-2:901234567890:schema/default-registry/TestSchema6-NON-Existent" --versions "1-1"
```

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada Katalog data, pilih Registri Skema.
3. Pilih registri yang berisi skema Anda dari daftar registri.
4. Pilih satu atau beberapa skema dari daftar, dengan mencentang kotak.
5. Di menu Tindakan, pilih Hapus skema.
6. Masukkan teks **`Delete`** di bidang untuk mengonfirmasi penghapusan.
7. Pilih Hapus.

Versi skema yang Anda tentukan akan dihapus dari registri.

Menghapus registri

Anda mungkin ingin menghapus sebuah registri ketika skema yang ada di dalamnya tidak lagi diatur berdasarkan registri itu. Anda harus menetapkan kembali skema tersebut ke registri yang lain.

Menghapus satu atau beberapa registri adalah tindakan permanen yang tidak dapat dibatalkan. Pastikan bahwa registri atau registri-registri tersebut tidak lagi diperlukan.

Registri default dapat dihapus menggunakan AWS CLI.

API AWS Glue

Untuk menghapus registri secara keseluruhan termasuk skema dan semua versinya, panggil API DeleteRegistry tindakan (Python: delete_registry). Menentukan sebuah struktur `RegistryId` untuk mengidentifikasi registri.

Sebagai contoh:

```
aws glue delete-registry --registry-id RegistryArn="arn:aws:glue:us-
east-2:901234567890:registry/registryName1"
```

```
aws glue delete-registry --registry-id RegistryName="TestRegistry-deletebyname"
```

Untuk mendapatkan status operasi hapus, Anda dapat memanggil API `GetRegistry` setelah panggilan asinkron.

Konsol AWS Glue

Untuk menghapus sebuah registri dari konsol AWS Glue:

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada Katalog data, pilih Registri Skema.
3. Pilih sebuah registri dari daftar, dengan mencentang kotaknya.
4. Di menu Tindakan, pilih Hapus registri.
5. Masukkan teks **`Delete`** di bidang untuk mengonfirmasi penghapusan.
6. Pilih Hapus.

Registri yang Anda pilih akan dihapus dari AWS Glue.

Contoh IAM untuk serializer

> **Note**
>
> Kebijakan terkelola AWS memberikan izin yang diperlukan untuk kasus penggunaan umum. Untuk informasi tentang menggunakan kebijakan terkelola untuk mengelola Registri Skema, lihat AWS kebijakan terkelola (standar) untuk AWS Glue.

Untuk serializer, Anda harus membuat sebuah kebijakan yang paling tidak serupa dengan yang di bawah ini untuk memberikan Anda kemampuan untuk menemukan `schemaVersionId` untuk definisi skema yang diberikan. Catatan, Anda harus memiliki izin baca pada registri agar dapat membaca skema dalam registri tersebut. Anda dapat membatasi registri yang dapat dibaca dengan menggunakan klausul `Resource`.

Contoh kode 13:

```
{
    "Sid" : "GetSchemaByDefinition",
    "Effect" : "Allow",
    "Action" :
 [
        "glue:GetSchemaByDefinition"
    ],
        "Resource" : ["arn:aws:glue:us-east-2:012345678:registry/registryname-1",
                      "arn:aws:glue:us-east-2:012345678:schema/registryname-1/
schemaname-1",
                      "arn:aws:glue:us-east-2:012345678:schema/registryname-1/
schemaname-2"
                     ]
}
```

Selanjutnya, Anda juga dapat memungkinkan produsen untuk membuat skema dan versi baru dengan memasukkan metode tambahan berikut. Catatan, Anda harus dapat memeriksa registri agar dapat menambahkan/menghapus/mengembangkan skema di dalamnya. Anda dapat membatasi registri yang dapat diperiksa dengan menggunakan klausul `Resource`.

Contoh kode 14:

```
{
    "Sid" : "RegisterSchemaWithMetadata",
```

```
    "Effect" : "Allow",
    "Action" :
 [
        "glue:GetSchemaByDefinition",
        "glue:CreateSchema",
        "glue:RegisterSchemaVersion",
        "glue:PutSchemaVersionMetadata",
    ],
    "Resource" : ["arn:aws:glue:aws-region:123456789012:registry/registryname-1",
                  "arn:aws:glue:aws-region:123456789012:schema/registryname-1/
schemaname-1",
                  "arn:aws:glue:aws-region:123456789012:schema/registryname-1/
schemaname-2"
                ]
}
```

Contoh IAM untuk deserializer

Untuk deserializer (sisi konsumen), Anda harus membuat kebijakan yang serupa dengan yang di bawah ini untuk memungkinkan deserializer mengambil skema dari Registri Skema untuk melakukan deserialisasi. Catatan, Anda harus dapat memeriksa registri agar dapat mengambil skema di dalamnya.

Contoh kode 15:

```
{
    "Sid" : "GetSchemaVersion",
    "Effect" : "Allow",
    "Action" :
 [
        "glue:GetSchemaVersion"
    ],
    "Resource" : ["*"]
}
```

Konektivitas pribadi menggunakan AWS PrivateLink

Anda dapat menggunakan AWS PrivateLink untuk menghubungkan VPC produsen data Anda ke AWS Glue dengan menentukan VPC endpoint antarmuka untuk AWS Glue. Ketika Anda menggunakan titik akhir antarmuka VPC, komunikasi antara VPC dan AWS Glue dilakukan sepenuhnya di jaringan AWS. Untuk informasi selengkapnya, lihat Menggunakan AWS Glue dengan Titik Akhir VPC.

Mengakses metrik Amazon CloudWatch

CloudWatch Metrik Amazon tersedia sebagai bagian dari CloudWatch tingkat gratis. Anda dapat mengakses metrik ini di CloudWatch Konsol. Metrik tingkat API meliputi CreateSchema (Sukses dan Latensi),, (Sukses dan Latensi) GetSchemaByDefinition, (Sukses dan Latensi), GetSchemaVersion (Sukses dan Latensi), RegisterSchemaVersion (Sukses dan Latensi). PutSchemaVersionMetadata Metrik tingkat sumber daya termasuk Registry. ThrottledByLimit, SchemaVersion. ThrottledByLimit, SchemaVersion .Ukuran.

Contoh AWS CloudFormation template untuk Schema Registry

Berikut ini adalah contoh templat untuk membuat sumber daya Registri Skema di AWS CloudFormation. Untuk membuat tumpukan ini di akun Anda, salin templat di atas ke dalam file `SampleTemplate.yaml`, dan jalankan perintah berikut:

```
aws cloudformation create-stack --stack-name ABCSchemaRegistryStack --template-body
 "'cat SampleTemplate.yaml'"
```

Contoh ini menggunakan `AWS::Glue::Registry` untuk membuat registri, `AWS::Glue::Schema` untuk membuat skema, `AWS::Glue::SchemaVersion` untuk membuat versi skema, dan `AWS::Glue::SchemaVersionMetadata` untuk mengisi metadata versi skema.

```
Description: "A sample CloudFormation template for creating Schema Registry resources."
Resources:
  ABCRegistry:
    Type: "AWS::Glue::Registry"
    Properties:
      Name: "ABCSchemaRegistry"
      Description: "ABC Corp. Schema Registry"
      Tags:
        - Key: "Project"
          Value: "Foo"
  ABCSchema:
    Type: "AWS::Glue::Schema"
    Properties:
      Registry:
        Arn: !Ref ABCRegistry
      Name: "TestSchema"
      Compatibility: "NONE"
      DataFormat: "AVRO"
      SchemaDefinition: >
```

```
        {"namespace":"foo.avro","type":"record","name":"user","fields":
[{"name":"name","type":"string"},{"name":"favorite_number","type":"int"}]}
      Tags:
        - Key: "Project"
          Value: "Foo"
  SecondSchemaVersion:
    Type: "AWS::Glue::SchemaVersion"
    Properties:
      Schema:
        SchemaArn: !Ref ABCSchema
      SchemaDefinition: >
        {"namespace":"foo.avro","type":"record","name":"user","fields":
[{"name":"status","type":"string", "default":"ON"}, {"name":"name","type":"string"},
{"name":"favorite_number","type":"int"}]}
  FirstSchemaVersionMetadata:
    Type: "AWS::Glue::SchemaVersionMetadata"
    Properties:
      SchemaVersionId: !GetAtt ABCSchema.InitialSchemaVersionId
      Key: "Application"
      Value: "Kinesis"
  SecondSchemaVersionMetadata:
    Type: "AWS::Glue::SchemaVersionMetadata"
    Properties:
      SchemaVersionId: !Ref SecondSchemaVersion
      Key: "Application"
      Value: "Kinesis"
```

Mengintegrasikan dengan Registri Skema AWS Glue

Bagian ini menjelaskan integrasi dengan Registri Skema AWS Glue. Contoh-contoh dalam bagian ini menunjukkan skema dengan format data AVRO. Untuk contoh lainnya, termasuk skema dengan format data JSON, lihat tes integrasi dan ReadMe informasi dalam repositori open source AWS Glue Schema Registry.

Topik

- Kasus penggunaan: Menghubungkan Registri Skema ke Amazon MSK atau Apache Kafka
- Kasus penggunaan: Mengintegrasikan Amazon Kinesis Data Streams dengan Registri Skema AWS Glue
- Kasus penggunaan: Amazon Managed Service untuk Apache Flink
- Kasus Penggunaan: Integrasi dengan AWS Lambda

- Kasus penggunaan: AWS Glue Data Catalog
- Kasus penggunaan: AWS Glue streaming
- Kasus penggunaan: Apache Kafka Streams
- Kasus penggunaan: Apache Kafka Connect

Kasus penggunaan: Menghubungkan Registri Skema ke Amazon MSK atau Apache Kafka

Mari kita anggap Anda sedang menulis data ke topik Apache Kafka, dan Anda dapat mengikuti langkah-langkah untuk memulai.

1. Buat klaster Amazon Managed Streaming for Apache Kafka (Amazon MSK) atau Apache Kafka dengan setidaknya satu topik. Jika membuat sebuah klaster Amazon MSK, maka Anda dapat menggunakan AWS Management Console. Ikuti instruksi berikut: Mulai Menggunakan Amazon MSK di Panduan Developer Amazon Managed Streaming for Apache Kafka.
2. Ikuti langkah-langkah di atas Instalasi SerDe Perpustakaan.
3. Untuk membuat skema registri, skema, atau skema versi, ikuti petunjuk pada bagian Memulai dengan Schema Registry dalam dokumen ini.
4. Mulai produsen dan konsumen Anda untuk menggunakan Registri Skema untuk menulis dan membaca catatan ke/dari topik Amazon MSK atau Apache Kafka. Contoh kode produsen dan konsumen dapat ditemukan dalam ReadMe file dari pustaka Serde. Perpustakaan Registri Skema pada produsen akan secara otomatis melakukan serialisasi pada catatan dan menghias catatan dengan ID versi skema.
5. Jika skema dari catatan ini telah diinput, atau jika pendaftaran otomatis telah diaktifkan, maka skema akan telah terdaftar dalam Registri Skema.
6. Pembacaan konsumen dari topik Amazon MSK atau Apache Kafka, menggunakan perpustakaan Registri Skema AWS Glue, secara otomatis akan mencari skema dari Registri Skema.

Kasus penggunaan: Mengintegrasikan Amazon Kinesis Data Streams dengan Registri Skema AWS Glue

Integrasi ini mengharuskan Anda memiliki pengaliran data Amazon Kinesis Data Streams yang sudah ada. Untuk informasi selengkapnya, lihat Memulai Amazon Kinesis Data Streams? dalam Panduan Developer Amazon Kinesis Data Streams.

Ada dua cara untuk berinteraksi dengan data dalam pengaliran data Kinesis Data Streams.

- Melalui perpustakaan Kinesis Producer Library (KPL) dan Kinesis Client Library (KCL) di Java. Support multi-bahasa tidak tersedia.
- Melalui `PutRecords`, `PutRecord`, dan `GetRecords`, API Kinesis Data Streams tersedia di AWS SDK for Java.

Jika saat ini Anda menggunakan pustaka KPL/KCL, kami rekomendasikan Anda untuk terus menggunakan metode tersebut. Ada versi KCL dan KPL yang diperbarui dengan Registri Skema terintegrasi, seperti yang ditunjukkan dalam contoh. Jika tidak, Anda dapat menggunakan kode sampel untuk memanfaatkan Registri Skema AWS Glue jika menggunakan API KDS secara langsung.

Integrasi Registri Skema ini hanya tersedia dengan KPL v0.14.2 atau yang lebih baru dan dengan KCL v2.3 atau yang lebih baru. Integrasi Registri Skema dengan format data JSON tersedia dengan KPL v0.14.8 atau yang lebih baru dan dengan KCL v2.3.6 atau yang lebih baru.

Berinteraksi dengan Data Menggunakan Kinesis SDK V2

Bagian ini menjelaskan cara berinteraksi dengan Kinesis menggunakan Kinesis SDK V2

```
// Example JSON Record, you can construct a AVRO record also
private static final JsonDataWithSchema record =
 JsonDataWithSchema.builder(schemaString, payloadString);
private static final DataFormat dataFormat = DataFormat.JSON;

//Configurations for Schema Registry
GlueSchemaRegistryConfiguration gsrConfig = new GlueSchemaRegistryConfiguration("us-
east-1");

GlueSchemaRegistrySerializer glueSchemaRegistrySerializer =
        new GlueSchemaRegistrySerializerImpl(awsCredentialsProvider, gsrConfig);
GlueSchemaRegistryDataFormatSerializer dataFormatSerializer =
        new GlueSchemaRegistrySerializerFactory().getInstance(dataFormat, gsrConfig);

Schema gsrSchema =
        new Schema(dataFormatSerializer.getSchemaDefinition(record), dataFormat.name(),
 "MySchema");

byte[] serializedBytes = dataFormatSerializer.serialize(record);
```

```
byte[] gsrEncodedBytes = glueSchemaRegistrySerializer.encode(streamName, gsrSchema,
 serializedBytes);

PutRecordRequest putRecordRequest = PutRecordRequest.builder()
        .streamName(streamName)
        .partitionKey("partitionKey")
        .data(SdkBytes.fromByteArray(gsrEncodedBytes))
        .build();
shardId = kinesisClient.putRecord(putRecordRequest)
        .get()
        .shardId();

GlueSchemaRegistryDeserializer glueSchemaRegistryDeserializer = new
 GlueSchemaRegistryDeserializerImpl(awsCredentialsProvider, gsrConfig);

GlueSchemaRegistryDataFormatDeserializer gsrDataFormatDeserializer =
        glueSchemaRegistryDeserializerFactory.getInstance(dataFormat, gsrConfig);

GetShardIteratorRequest getShardIteratorRequest = GetShardIteratorRequest.builder()
        .streamName(streamName)
        .shardId(shardId)
        .shardIteratorType(ShardIteratorType.TRIM_HORIZON)
        .build();

String shardIterator = kinesisClient.getShardIterator(getShardIteratorRequest)
        .get()
        .shardIterator();

GetRecordsRequest getRecordRequest = GetRecordsRequest.builder()
        .shardIterator(shardIterator)
        .build();
GetRecordsResponse recordsResponse = kinesisClient.getRecords(getRecordRequest)
        .get();

List<Object> consumerRecords = new ArrayList<>();
List<Record> recordsFromKinesis = recordsResponse.records();

for (int i = 0; i < recordsFromKinesis.size(); i++) {
    byte[] consumedBytes = recordsFromKinesis.get(i)
            .data()
            .asByteArray();

    Schema gsrSchema = glueSchemaRegistryDeserializer.getSchema(consumedBytes);
```

```
    Object decodedRecord =
 gsrDataFormatDeserializer.deserialize(ByteBuffer.wrap(consumedBytes),

 gsrSchema.getSchemaDefinition());
    consumerRecords.add(decodedRecord);
}
```

Berinteraksi dengan data menggunakan pustaka KPL/KCL

Bagian ini menjelaskan cara mengintegrasikan Kinesis Data Streams dengan Registri Skema dengan menggunakan pustaka KPL/KCL. Untuk informasi selengkapnya tentang KPL/KCL, lihat Mengembangkan Produsen Menggunakan Amazon Kinesis Producer Library dalam Panduan Developer Amazon Kinesis Data Streams.

Menyiapkan Registri Skema di KPL

1. Menentukan definisi skema untuk data, format data dan nama skema yang ditulis dalam Registri Skema AWS Glue.
2. Mengkonfigurasi objek `GlueSchemaRegistryConfiguration`, opsional.
3. Berikan objek skema ke `addUserRecord API`.

```
private static final String SCHEMA_DEFINITION = "{"namespace": "example.avro",\n"
+ " "type": "record",\n"
+ " "name": "User",\n"
+ " "fields": [\n"
+ " {"name": "name", "type": "string"},\n"
+ " {"name": "favorite_number", "type": ["int", "null"]},\n"
+ " {"name": "favorite_color", "type": ["string", "null"]}\n"
+ " ]\n"
+ "}";

KinesisProducerConfiguration config = new KinesisProducerConfiguration();
config.setRegion("us-west-1")

//[Optional] configuration for Schema Registry.

GlueSchemaRegistryConfiguration schemaRegistryConfig =
new GlueSchemaRegistryConfiguration("us-west-1");

schemaRegistryConfig.setCompression(true);

config.setGlueSchemaRegistryConfiguration(schemaRegistryConfig);
```

```
///Optional configuration ends.

final KinesisProducer producer =
      new KinesisProducer(config);

final ByteBuffer data = getDataToSend();

com.amazonaws.services.schemaregistry.common.Schema gsrSchema =
    new Schema(SCHEMA_DEFINITION, DataFormat.AVRO.toString(), "demoSchema");

ListenableFuture<UserRecordResult> f = producer.addUserRecord(
config.getStreamName(), TIMESTAMP, Utils.randomExplicitHashKey(), data, gsrSchema);

private static ByteBuffer getDataToSend() {
      org.apache.avro.Schema avroSchema =
        new org.apache.avro.Schema.Parser().parse(SCHEMA_DEFINITION);

      GenericRecord user = new GenericData.Record(avroSchema);
      user.put("name", "Emily");
      user.put("favorite_number", 32);
      user.put("favorite_color", "green");

      ByteArrayOutputStream outBytes = new ByteArrayOutputStream();
      Encoder encoder = EncoderFactory.get().directBinaryEncoder(outBytes, null);
      new GenericDatumWriter<>(avroSchema).write(user, encoder);
      encoder.flush();
      return ByteBuffer.wrap(outBytes.toByteArray());
 }
```

Menyiapkan pustaka klien Kinesis

Anda akan mengembangkan konsumen Perpustakaan Klien Kinesis di Java. Untuk informasi selengkapnya tentang KCL, lihat Mengembangkan Konsumen Kinesis Client Librarydi Java dalam Panduan Developer Amazon Kinesis Data Streams.

1. Buat sebuah instans `GlueSchemaRegistryDeserializer` dengan memberikan sebuah objek `GlueSchemaRegistryConfiguration`.
2. Berikan `GlueSchemaRegistryDeserializer` ke `retrievalConfig.glueSchemaRegistryDeserializer`.
3. Mengakses skema pesan masuk dengan memanggil `kinesisClientRecord.getSchema()`.

```
GlueSchemaRegistryConfiguration schemaRegistryConfig =
    new GlueSchemaRegistryConfiguration(this.region.toString());

 GlueSchemaRegistryDeserializer glueSchemaRegistryDeserializer =
    new
 GlueSchemaRegistryDeserializerImpl(DefaultCredentialsProvider.builder().build(),
 schemaRegistryConfig);

 RetrievalConfig retrievalConfig =
 configsBuilder.retrievalConfig().retrievalSpecificConfig(new
 PollingConfig(streamName, kinesisClient));
 retrievalConfig.glueSchemaRegistryDeserializer(glueSchemaRegistryDeserializer);

  Scheduler scheduler = new Scheduler(
            configsBuilder.checkpointConfig(),
            configsBuilder.coordinatorConfig(),
            configsBuilder.leaseManagementConfig(),
            configsBuilder.lifecycleConfig(),
            configsBuilder.metricsConfig(),
            configsBuilder.processorConfig(),
            retrievalConfig
        );

 public void processRecords(ProcessRecordsInput processRecordsInput) {
            MDC.put(SHARD_ID_MDC_KEY, shardId);
            try {
                log.info("Processing {} record(s)",
                processRecordsInput.records().size());
                processRecordsInput.records()
                .forEach(
                    r ->
                        log.info("Processed record pk: {} -- Seq: {} : data {} with
 schema: {}",
                        r.partitionKey(),
 r.sequenceNumber(), recordToAvroObj(r).toString(), r.getSchema()));
            } catch (Throwable t) {
                log.error("Caught throwable while processing records. Aborting.");
                Runtime.getRuntime().halt(1);
            } finally {
                MDC.remove(SHARD_ID_MDC_KEY);
            }
 }
```

```
private GenericRecord recordToAvroObj(KinesisClientRecord r) {
   byte[] data = new byte[r.data().remaining()];
   r.data().get(data, 0, data.length);
   org.apache.avro.Schema schema = new
org.apache.avro.Schema.Parser().parse(r.schema().getSchemaDefinition());
   DatumReader datumReader = new GenericDatumReader<>(schema);

   BinaryDecoder binaryDecoder = DecoderFactory.get().binaryDecoder(data, 0,
data.length, null);
   return (GenericRecord) datumReader.read(null, binaryDecoder);
}
```

Berinteraksi dengan data menggunakan API Kinesis Data Streams

Bagian ini menjelaskan cara mengintegrasikan Kinesis Data Streams dengan Registri Skema menggunakan API Kinesis Data Streams.

1. Memperbarui dependensi Maven ini:

```
<dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>com.amazonaws</groupId>
                <artifactId>aws-java-sdk-bom</artifactId>
                <version>1.11.884</version>
                <type>pom</type>
                <scope>import</scope>
            </dependency>
        </dependencies>
    </dependencyManagement>

    <dependencies>
        <dependency>
            <groupId>com.amazonaws</groupId>
            <artifactId>aws-java-sdk-kinesis</artifactId>
        </dependency>

        <dependency>
            <groupId>software.amazon.glue</groupId>
            <artifactId>schema-registry-serde</artifactId>
            <version>1.1.5</version>
        </dependency>
```

```
        <dependency>
            <groupId>com.fasterxml.jackson.dataformat</groupId>
            <artifactId>jackson-dataformat-cbor</artifactId>
            <version>2.11.3</version>
        </dependency>
    </dependencies>
```

2. Dalam produsen, tambahkan informasi header skema menggunakan `PutRecords` atau API `PutRecord` di Kinesis Data Streams.

```
//The following lines add a Schema Header to the record
        com.amazonaws.services.schemaregistry.common.Schema awsSchema =
            new com.amazonaws.services.schemaregistry.common.Schema(schemaDefinition,
 DataFormat.AVRO.name(),
                schemaName);
        GlueSchemaRegistrySerializerImpl glueSchemaRegistrySerializer =
            new
 GlueSchemaRegistrySerializerImpl(DefaultCredentialsProvider.builder().build(), new
 GlueSchemaRegistryConfiguration(getConfigs()));
        byte[] recordWithSchemaHeader =
            glueSchemaRegistrySerializer.encode(streamName, awsSchema,
 recordAsBytes);
```

3. Di produsen, gunakan `PutRecords` atau API `PutRecord` untuk menempatkan catatan ke dalam aliran data.
4. Dalam konsumen, hapus catatan skema dari header, dan lakukan serialisasi pada catatan skema Avro.

```
//The following lines remove Schema Header from record
        GlueSchemaRegistryDeserializerImpl glueSchemaRegistryDeserializer =
            new
 GlueSchemaRegistryDeserializerImpl(DefaultCredentialsProvider.builder().build(),
 getConfigs());
        byte[] recordWithSchemaHeaderBytes = new
 byte[recordWithSchemaHeader.remaining()];
        recordWithSchemaHeader.get(recordWithSchemaHeaderBytes, 0,
 recordWithSchemaHeaderBytes.length);
        com.amazonaws.services.schemaregistry.common.Schema awsSchema =
            glueSchemaRegistryDeserializer.getSchema(recordWithSchemaHeaderBytes);
        byte[] record =
 glueSchemaRegistryDeserializer.getData(recordWithSchemaHeaderBytes);
```

```
        //The following lines serialize an AVRO schema record
        if (DataFormat.AVRO.name().equals(awsSchema.getDataFormat())) {
            Schema avroSchema = new
  org.apache.avro.Schema.Parser().parse(awsSchema.getSchemaDefinition());
            Object genericRecord = convertBytesToRecord(avroSchema, record);
            System.out.println(genericRecord);
        }
```

Berinteraksi dengan data menggunakan API Kinesis Data Streams

Berikut ini adalah kode contoh untuk menggunakan `PutRecords` dan API `GetRecords`.

```
//Full sample code
import
 com.amazonaws.services.schemaregistry.deserializers.GlueSchemaRegistryDeserializerImpl;
import
 com.amazonaws.services.schemaregistry.serializers.GlueSchemaRegistrySerializerImpl;
import com.amazonaws.services.schemaregistry.utils.AVROUtils;
import com.amazonaws.services.schemaregistry.utils.AWSSchemaRegistryConstants;
import org.apache.avro.Schema;
import org.apache.avro.generic.GenericData;
import org.apache.avro.generic.GenericDatumReader;
import org.apache.avro.generic.GenericDatumWriter;
import org.apache.avro.generic.GenericRecord;
import org.apache.avro.io.Decoder;
import org.apache.avro.io.DecoderFactory;
import org.apache.avro.io.Encoder;
import org.apache.avro.io.EncoderFactory;
import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.services.glue.model.DataFormat;

import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.IOException;
import java.nio.ByteBuffer;
import java.util.Collections;
import java.util.HashMap;
import java.util.Map;


public class PutAndGetExampleWithEncodedData {
    static final String regionName = "us-east-2";
```

```
    static final String streamName = "testStream1";
    static final String schemaName = "User-Topic";
    static final String AVRO_USER_SCHEMA_FILE = "src/main/resources/user.avsc";
    KinesisApi kinesisApi = new KinesisApi();

    void runSampleForPutRecord() throws IOException {
        Object testRecord = getTestRecord();
        byte[] recordAsBytes = convertRecordToBytes(testRecord);
        String schemaDefinition =
AVROUtils.getInstance().getSchemaDefinition(testRecord);

        //The following lines add a Schema Header to a record
        com.amazonaws.services.schemaregistry.common.Schema awsSchema =
            new com.amazonaws.services.schemaregistry.common.Schema(schemaDefinition,
DataFormat.AVRO.name(),
                schemaName);
        GlueSchemaRegistrySerializerImpl glueSchemaRegistrySerializer =
            new
GlueSchemaRegistrySerializerImpl(DefaultCredentialsProvider.builder().build(), new
GlueSchemaRegistryConfiguration(regionName));
        byte[] recordWithSchemaHeader =
            glueSchemaRegistrySerializer.encode(streamName, awsSchema, recordAsBytes);

        //Use PutRecords api to pass a list of records
        kinesisApi.putRecords(Collections.singletonList(recordWithSchemaHeader),
streamName, regionName);

        //OR
        //Use PutRecord api to pass single record
        //kinesisApi.putRecord(recordWithSchemaHeader, streamName, regionName);
    }

    byte[] runSampleForGetRecord() throws IOException {
        ByteBuffer recordWithSchemaHeader = kinesisApi.getRecords(streamName,
regionName);

        //The following lines remove the schema registry header
        GlueSchemaRegistryDeserializerImpl glueSchemaRegistryDeserializer =
            new
GlueSchemaRegistryDeserializerImpl(DefaultCredentialsProvider.builder().build(), new
GlueSchemaRegistryConfiguration(regionName));
        byte[] recordWithSchemaHeaderBytes = new
byte[recordWithSchemaHeader.remaining()];
```

```
        recordWithSchemaHeader.get(recordWithSchemaHeaderBytes, 0,
recordWithSchemaHeaderBytes.length);

        com.amazonaws.services.schemaregistry.common.Schema awsSchema =
            glueSchemaRegistryDeserializer.getSchema(recordWithSchemaHeaderBytes);

        byte[] record =
glueSchemaRegistryDeserializer.getData(recordWithSchemaHeaderBytes);

        //The following lines serialize an AVRO schema record
        if (DataFormat.AVRO.name().equals(awsSchema.getDataFormat())) {
            Schema avroSchema = new
org.apache.avro.Schema.Parser().parse(awsSchema.getSchemaDefinition());
            Object genericRecord = convertBytesToRecord(avroSchema, record);
            System.out.println(genericRecord);
        }

        return record;
    }

    private byte[] convertRecordToBytes(final Object record) throws IOException {
        ByteArrayOutputStream recordAsBytes = new ByteArrayOutputStream();
        Encoder encoder = EncoderFactory.get().directBinaryEncoder(recordAsBytes,
null);
        GenericDatumWriter datumWriter = new
GenericDatumWriter<>(AVROUtils.getInstance().getSchema(record));
        datumWriter.write(record, encoder);
        encoder.flush();
        return recordAsBytes.toByteArray();
    }

    private GenericRecord convertBytesToRecord(Schema avroSchema, byte[] record) throws
IOException {
        final GenericDatumReader<GenericRecord> datumReader = new
GenericDatumReader<>(avroSchema);
        Decoder decoder = DecoderFactory.get().binaryDecoder(record, null);
        GenericRecord genericRecord = datumReader.read(null, decoder);
        return genericRecord;
    }

    private Map<String, String> getMetadata() {
        Map<String, String> metadata = new HashMap<>();
        metadata.put("event-source-1", "topic1");
        metadata.put("event-source-2", "topic2");
```

```
        metadata.put("event-source-3", "topic3");
        metadata.put("event-source-4", "topic4");
        metadata.put("event-source-5", "topic5");
        return metadata;
    }

    private GlueSchemaRegistryConfiguration getConfigs() {
        GlueSchemaRegistryConfiguration configs = new
 GlueSchemaRegistryConfiguration(regionName);
        configs.setSchemaName(schemaName);
        configs.setAutoRegistration(true);
        configs.setMetadata(getMetadata());
        return configs;
    }

    private Object getTestRecord() throws IOException {
        GenericRecord genericRecord;
        Schema.Parser parser = new Schema.Parser();
        Schema avroSchema = parser.parse(new File(AVRO_USER_SCHEMA_FILE));

        genericRecord = new GenericData.Record(avroSchema);
        genericRecord.put("name", "testName");
        genericRecord.put("favorite_number", 99);
        genericRecord.put("favorite_color", "red");

        return genericRecord;
    }
}
```

Kasus penggunaan: Amazon Managed Service untuk Apache Flink

Apache Flink adalah sebuah kerangka kerja sumber terbuka populer dan mesin pengolahan terdistribusi untuk komputasi stateful atas aliran data yang tak terbatas dan dibatasi. Amazon Managed Service untuk Apache Flink adalah AWS layanan terkelola penuh yang memungkinkan Anda membangun dan mengelola aplikasi Apache Flink untuk memproses data streaming.

Apache Flink sumber terbuka menyediakan sejumlah sumber dan sink. Sebagai contoh, sumber data yang telah ditetapkan termasuk membaca dari file, direktori, dan soket, dan menyerap data dari koleksi dan iterator. DataStream Konektor Apache Flink menyediakan kode untuk Apache Flink untuk berinteraksi dengan berbagai sistem pihak ketiga, seperti Apache Kafka atau Kinesis sebagai sumber dan/atau sink.

Untuk informasi lebih lanjut, lihat Panduan Developer Amazon Kinesis Data Analytics.

Konektor Apache Flink Kafka

Apache Flink menyediakan sebuah konektor aliran data Apache Kafka untuk membaca data dari dan menulis data untuk topik Kafka dengan jaminan persis-satu-kali. Konsumen Kafka Flink,`FlinkKafkaConsumer`, menyediakan akses untuk membaca dari satu atau lebih topik Kafka. Produsen Kafka dari Apache Flink, `FlinkKafkaProducer`, memungkinkan menulis aliran catatan untuk satu atau beberapa topik Kafka. Untuk informasi lebih lanjut, lihat Konektor Apache Kafka.

Konektor aliran Kinesis Apache Flink

Konektor pengaliran data Kinesis menyediakan akses ke Amazon Kinesis Data Streams. `FlinkKinesisConsumer` adalah sumber data streaming paralel yang persis-satu-kali langsung berlangganan beberapa pengaliran Kinesis dalam wilayah layanan AWS, dan dapat secara transparan menangani re-sharding pada pengaliran saat tugas sedang berjalan. Setiap subtugas konsumen bertanggung jawab untuk mengambil catatan data dari beberapa serpihan Kinesis. Jumlah serpihan yang diambil oleh setiap subtugas akan berubah karena serpihan ditutup dan dibuat oleh Kinesis. `FlinkKinesisProducer` menggunakan Kinesis Producer Library (KPL) untuk memasukkan data dari pengaliran Apache Flink ke pengaliran Kinesis. Untuk informasi selengkapnya, lihat Konektor Amazon Kinesis Streams.

Untuk informasi lebih lanjut, lihat Repositori GitHub Skema AWS Glue.

Mengintegrasikan dengan Apache Flink

SerDes Perpustakaan yang disediakan dengan Schema Registry terintegrasi dengan Apache Flink. Untuk bekerja dengan Apache Flink, Anda diharuskan untuk menerapkan antarmuka `SerializationSchema` dan `DeserializationSchema` yang disebut `GlueSchemaRegistryAvroSerializationSchema` dan `GlueSchemaRegistryAvroDeserializationSchema`, yang dapat Anda hubungkan ke konektor Apache Flink.

Menambahkan ketergantungan AWS Glue Schema Registry ke dalam aplikasi Apache Flink

Untuk menyiapkan dependensi integrasi ke Registri Skema AWS Glue dalam aplikasi Apache Flink:

1. Tambahkan dependensi ke file `pom.xml` Anda.

```
<dependency>
```

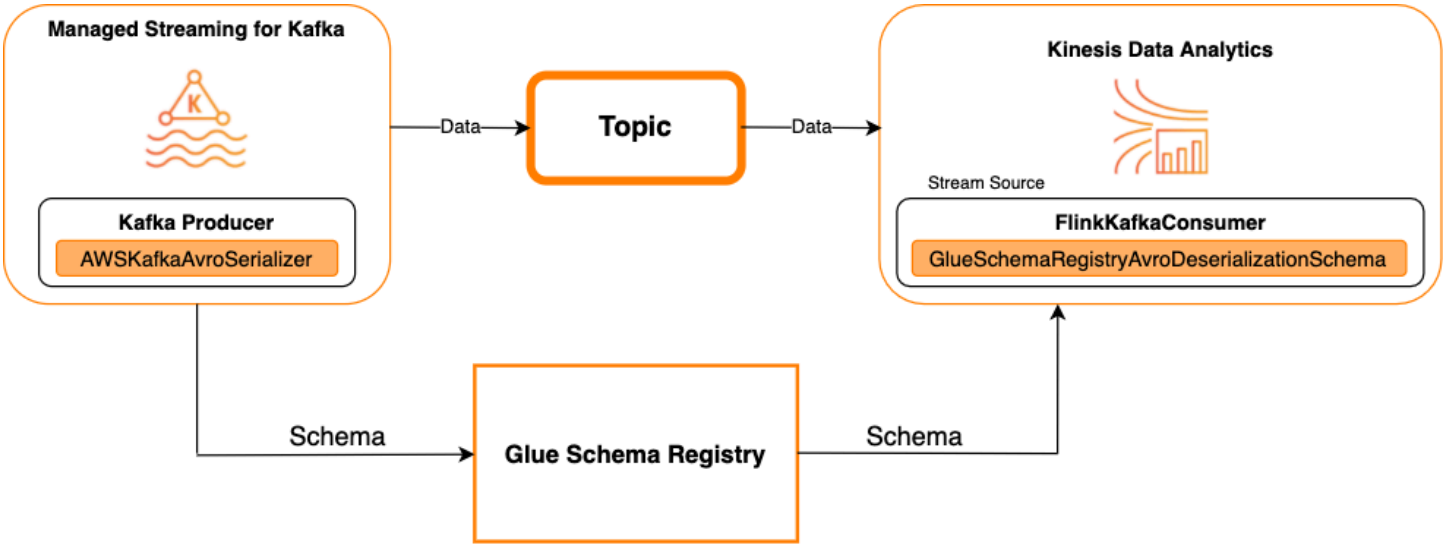
```
    <groupId>software.amazon.glue</groupId>
    <artifactId>schema-registry-flink-serde</artifactId>
    <version>1.0.0</version>
</dependency>
```

Mengintegrasikan Kafka atau Amazon MSK dengan Apache Flink

Anda dapat menggunakan Managed Service untuk Apache Flink untuk Apache Flink, dengan Kafka sebagai sumber atau Kafka sebagai wastafel.

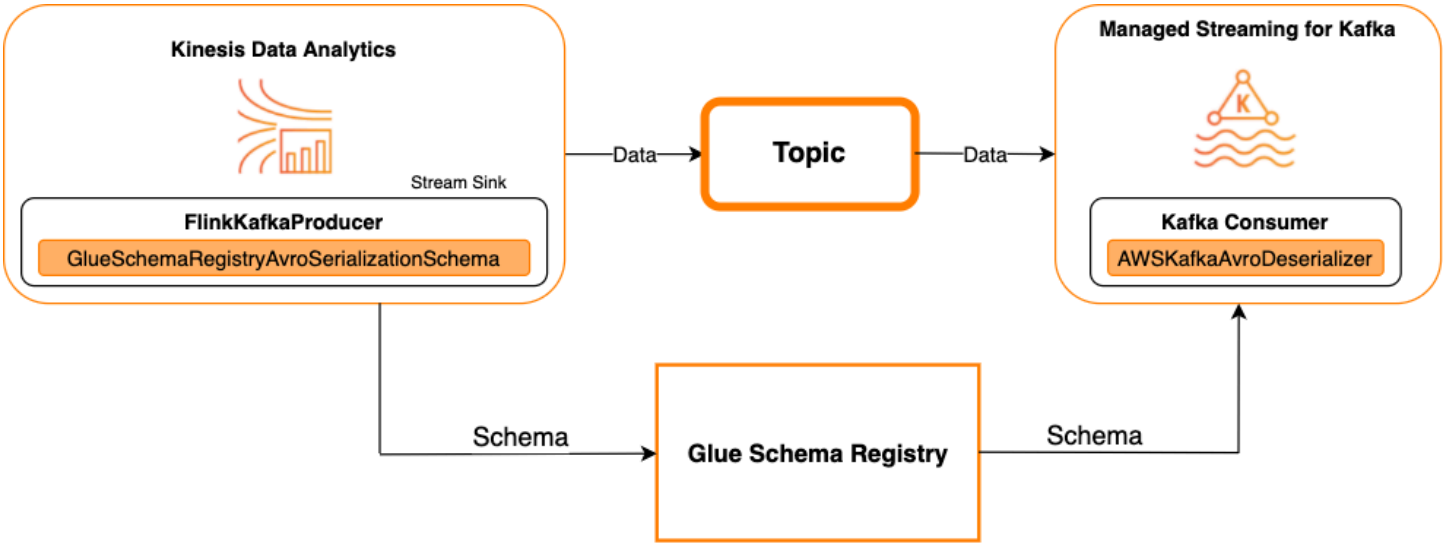
Kafka sebagai sumber

Diagram berikut menunjukkan integrasi Kinesis Data Streams dengan Managed Service untuk Apache Flink untuk Apache Flink, dengan Kafka sebagai sumber.



Kafka sebagai sebuah sink

Diagram berikut menunjukkan integrasi Kinesis Data Streams dengan Managed Service untuk Apache Flink untuk Apache Flink, dengan Kafka sebagai wastafel.



Untuk mengintegrasikan Kafka (atau Amazon MSK) dengan Managed Service untuk Apache Flink untuk Apache Flink, dengan Kafka sebagai sumber atau Kafka sebagai wastafel, buat perubahan kode di bawah ini. Tambahkan blok kode ditebalkan untuk kode Anda masing-masing di bagian analog.

Jika Kafka adalah sumbernya, maka gunakan kode deserializer (blok 2). Jika Kafka adalah sink-nya, maka gunakan kode serializer (blok 3).

```
StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();

String topic = "topic";
Properties properties = new Properties();
properties.setProperty("bootstrap.servers", "localhost:9092");
properties.setProperty("group.id", "test");

// block 1
Map<String, Object> configs = new HashMap<>();
configs.put(AWSSchemaRegistryConstants.AWS_REGION, "aws-region");
configs.put(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING, true);
configs.put(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.GENERIC_RECORD.getName());

FlinkKafkaConsumer<GenericRecord> consumer = new FlinkKafkaConsumer<>(
    topic,
    // block 2
    GlueSchemaRegistryAvroDeserializationSchema.forGeneric(schema, configs),
    properties);
```

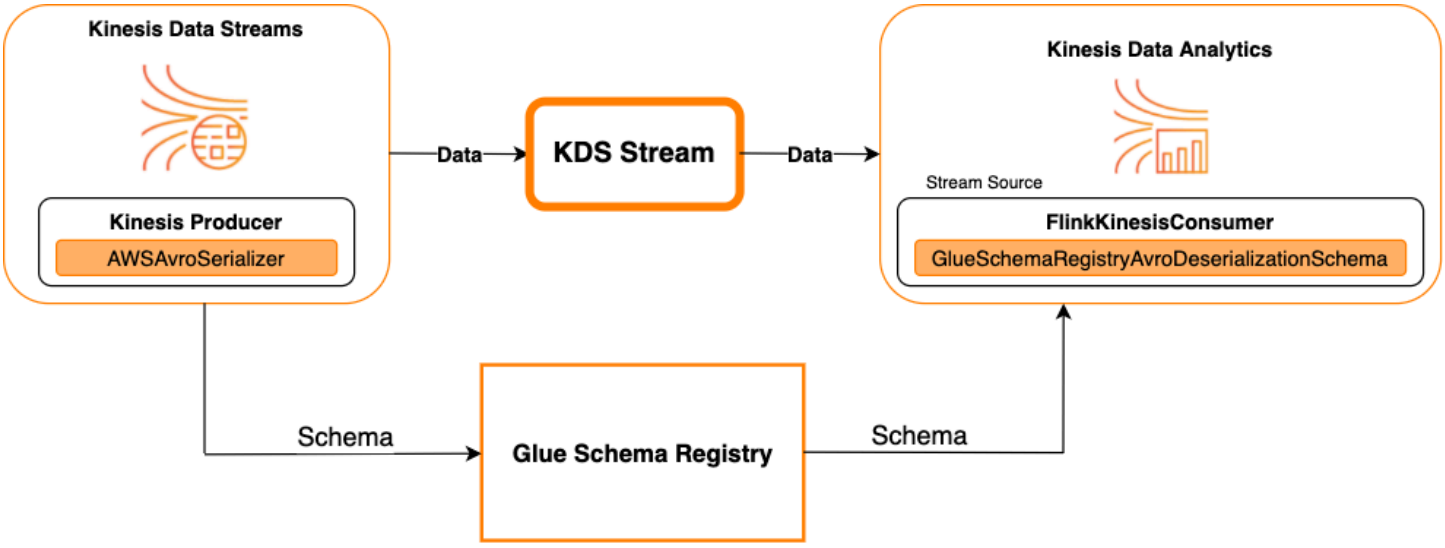
```
FlinkKafkaProducer<GenericRecord> producer = new FlinkKafkaProducer<>(
    topic,
    // block 3
    GlueSchemaRegistryAvroSerializationSchema.forGeneric(schema, topic, configs),
    properties);

DataStream<GenericRecord> stream = env.addSource(consumer);
stream.addSink(producer);
env.execute();
```

Mengintegrasikan Kinesis Data Streams dengan Apache Flink

Anda dapat menggunakan Managed Service for Apache Flink untuk Apache Flink dengan Kinesis Data Streams sebagai sumber atau wastafel.

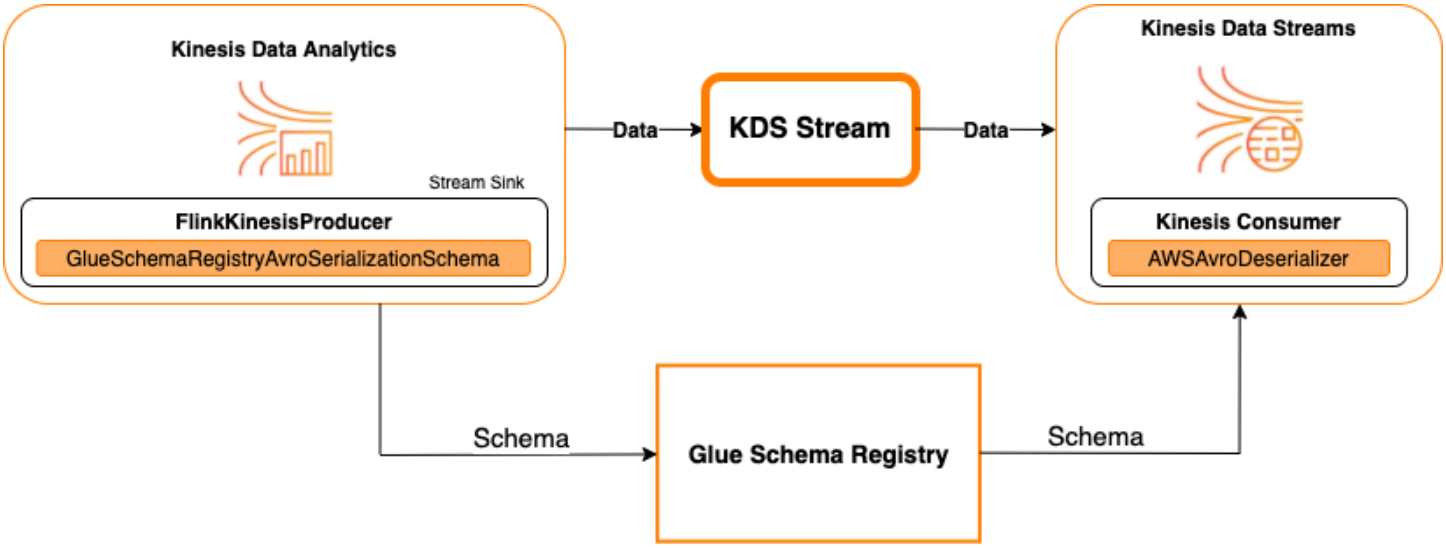
Kinesis Data Streams sebagai sebuah sumber

Diagram berikut menunjukkan integrasi Kinesis Data Streams dengan Managed Service untuk Apache Flink untuk Apache Flink, dengan Kinesis Data Streams sebagai sumber.



Kinesis Data Streams sebagai sebuah sink

Diagram berikut menunjukkan integrasi Kinesis Data Streams dengan Managed Service untuk Apache Flink untuk Apache Flink, dengan Kinesis Data Streams sebagai wastafel.



Untuk mengintegrasikan Kinesis Data Streams dengan Managed Service untuk Apache Flink untuk Apache Flink, dengan Kinesis Data Streams sebagai sumber atau Kinesis Data Streams sebagai sink, buat perubahan kode di bawah ini. Tambahkan blok kode ditebalkan untuk kode Anda masing-masing di bagian analog.

Jika Kinesis Data Streams adalah sumbernya, maka gunakan kode deserializer (blok 2). Jika Kinesis Data Streams adalah sink-nya, maka gunakan kode serializer (blok 3).

```
StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();

String streamName = "stream";
Properties consumerConfig = new Properties();
consumerConfig.put(AWSConfigConstants.AWS_REGION, "aws-region");
consumerConfig.put(AWSConfigConstants.AWS_ACCESS_KEY_ID, "aws_access_key_id");
consumerConfig.put(AWSConfigConstants.AWS_SECRET_ACCESS_KEY, "aws_secret_access_key");
consumerConfig.put(ConsumerConfigConstants.STREAM_INITIAL_POSITION, "LATEST");

// block 1
Map<String, Object> configs = new HashMap<>();
configs.put(AWSSchemaRegistryConstants.AWS_REGION, "aws-region");
configs.put(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING, true);
configs.put(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.GENERIC_RECORD.getName());

FlinkKinesisConsumer<GenericRecord> consumer = new FlinkKinesisConsumer<>(
    streamName,
    // block 2
    GlueSchemaRegistryAvroDeserializationSchema.forGeneric(schema, configs),
```

```
    properties);

FlinkKinesisProducer<GenericRecord> producer = new FlinkKinesisProducer<>(
    // block 3
    GlueSchemaRegistryAvroSerializationSchema.forGeneric(schema, topic, configs),
    properties);
producer.setDefaultStream(streamName);
producer.setDefaultPartition("0");

DataStream<GenericRecord> stream = env.addSource(consumer);
stream.addSink(producer);
env.execute();
```

Kasus Penggunaan: Integrasi dengan AWS Lambda

Untuk menggunakan fungsi AWS Lambda sebagai konsumen Apache Kafka/Amazon MSK dan melakukan deserialisasi pada pesan dikodekan-AVRO dengan menggunakan Registri Skema AWS Glue, kunjungi halaman MSK Labs.

Kasus penggunaan: AWS Glue Data Catalog

Tabel AWS Glue mendukung skema yang Anda dapat tentukan secara manual atau dengan me-referensi ke Registri Skema AWS Glue. Registri Skema terintegrasi dengan Katalog Data untuk memungkinkan Anda untuk secara opsional menggunakan skema yang disimpan dalam Registri Skema saat membuat atau memperbarui tabel atau partisi AWS Glue dalam Katalog Data. Untuk mengidentifikasi sebuah definisi skema dalam Registri Skema, minimal, Anda perlu mengetahui ARN dari skema yang ia menjadi bagiannya. Sebuah versi skema dari sebuah skema, yang berisi definisi skema, dapat direferensikan oleh UUID atau versi nomor. Selalu ada satu versi skema, yakni versi "terbaru", yang dapat dicari tanpa mengetahui nomor versi atau UUID.

Ketika memanggil operasi `CreateTable` atau `UpdateTable`, Anda akan memberikan sebuah struktur `TableInput` yang berisi `StorageDescriptor`, yang mungkin memiliki sebuah `SchemaReference` ke skema yang sudah ada di Registri Skema. Demikian pula, ketika Anda memanggil `GetTable` atau API `GetPartition`, responsnya mungkin berisi skema dan `SchemaReference`. Ketika sebuah tabel atau partisi dibuat menggunakan sebuah referensi skema, Katalog Data akan mencoba untuk mengambil skema tersebut untuk referensi skema ini. Jika ia tidak dapat menemukan skema di Registri Skema, maka ia akan mengembalikan sebuah skema kosong di respons `GetTable`; jika tidak, responsnya akan memiliki skema dan referensi skema.

Anda juga dapat melakukan tindakan dari konsol AWS Glue.

Untuk melakukan operasi ini dan membuat, memperbarui, atau melihat informasi skema, Anda harus memberikan peran IAM kepada pengguna panggilan yang memberikan izin untuk API. `GetSchemaVersion`

Menambahkan tabel atau memperbarui skema untuk tabel

Menambahkan sebuah tabel baru dari skema yang ada mengikat tabel ke versi skema tertentu. Setelah versi skema baru didaftarkan, Anda dapat memperbarui definisi tabel ini dari halaman Lihat tabel di konsol AWS Glue atau menggunakan API UpdateTable tindakan (Python: update_table).

Menambahkan tabel dari skema yang ada

Anda dapat membuat sebuah tabel AWS Glue dari versi skema dalam registri dengan menggunakan konsol AWS Glue atau API `CreateTable`.

API AWS Glue

Ketika memanggil API `CreateTable`, Anda akan memberikan sebuah `TableInput` yang berisi `StorageDescriptor` yang memiliki `SchemaReference` ke sebuah skema yang sudah ada di Registri Skema.

Konsol AWS Glue

Untuk membuat sebuah tabel dari konsol AWS Glue:

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada Katalog data, pilih Tabel.
3. Di menu Tambahkan Tabel, pilih Tambahkan tabel dari skema yang ada.
4. Mengkonfigurasi properti tabel dan penyimpanan data sebagaimana dalam Panduan Developer AWS Glue.
5. Di halaman Pilih skema Glue, pilih Registri tempat skema berada.
6. Pilih Nama skema dan pilih Versi skema yang akan diterapkan.
7. Tinjau pratinjau skema, dan pilih Selanjutnya.
8. Tinjau dan buat tabel.

Skema dan versi yang diterapkan ke tabel muncul di kolom Skema Glue dalam daftar tabel. Anda dapat melihat tabel tersebut untuk melihat lebih detail.

Memperbarui skema untuk tabel

Ketika sebuah versi skema baru tersedia, Anda mungkin ingin memperbarui skema tabel menggunakan API UpdateTable tindakan (Python: update_table) atau konsol AWS Glue.

> **Important**
>
> Ketika memperbarui skema untuk sebuah tabel yang sudah ada yang memiliki skema AWS Glue yang ditentukan secara manual, skema baru yang direferensikan dalam Registri Skema mungkin tidak kompatibel. Hal ini dapat menyebabkan tugas Anda gagal.

API AWS Glue

Ketika memanggil API `UpdateTable`, Anda akan memberikan sebuah `TableInput` yang berisi `StorageDescriptor` yang memiliki `SchemaReference` ke sebuah skema yang sudah ada di Registri Skema.

Konsol AWS Glue

Untuk memperbarui skema untuk sebuah tabel dari konsol AWS Glue:

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada Katalog data, pilih Tabel.
3. Melihat tabel dari daftar tabel.
4. Klik Perbarui skema di kotak yang memberitahu Anda tentang versi baru.
5. Tinjau perbedaan antara skema saat ini dan skema baru.
6. Pilih Tampilkan semua perbedaan skema untuk melihat detail lebih lanjut.
7. Pilih Simpan tabel untuk menyetujui versi baru.

Kasus penggunaan: AWS Glue streaming

AWS Gluestreaming mengkonsumsi data dari sumber streaming dan melakukan operasi ETL sebelum menulis ke sink output. Sumber streaming input dapat ditentukan menggunakan Tabel Data atau langsung dengan menentukan konfigurasi sumber.

AWS Gluestreaming mendukung tabel Katalog Data untuk sumber streaming yang dibuat dengan skema yang ada di Registri AWS Glue Skema. Anda dapat membuat skema di AWS Glue Schema Registry dan membuat AWS Glue tabel dengan sumber streaming menggunakan skema ini. AWS GlueTabel ini dapat digunakan sebagai input ke pekerjaan AWS Glue streaming untuk deserialisasi data dalam aliran input.

Satu hal yang perlu diperhatikan di sini adalah ketika AWS Glue skema di Registri Skema berubah, Anda perlu memulai ulang pekerjaan AWS Glue streaming yang perlu mencerminkan perubahan dalam skema.

Kasus penggunaan: Apache Kafka Streams

API Apache Kafka Streams adalah sebuah perpustakaan klien untuk memproses dan menganalisis data yang disimpan di Apache Kafka. Bagian ini menjelaskan integrasi Apache Kafka Streams dengan Registri Skema AWS Glue, yang memungkinkan Anda untuk mengelola dan menegakkan skema pada aplikasi streaming data Anda. Untuk informasi lebih lanjut tentang Apache Kafka Streams, lihat Apache Kafka Streams.

Integrasi dengan Perpustakaan SerDes

Ada sebuah kelas `GlueSchemaRegistryKafkaStreamsSerde` yang dapat Anda konfigurasikan dengan sebuah aplikasi Streams.

Kode contoh aplikasi Kafka Streams

Untuk menggunakan Registri Skema AWS Glue dalam aplikasi Apache Kafka Streams:

1. Konfigurasikan aplikasi Kafka Streams.

```
final Properties props = new Properties();
    props.put(StreamsConfig.APPLICATION_ID_CONFIG, "avro-streams");
    props.put(StreamsConfig.BOOTSTRAP_SERVERS_CONFIG, "localhost:9092");
    props.put(StreamsConfig.CACHE_MAX_BYTES_BUFFERING_CONFIG, 0);
    props.put(StreamsConfig.DEFAULT_KEY_SERDE_CLASS_CONFIG,
 Serdes.String().getClass().getName());
    props.put(StreamsConfig.DEFAULT_VALUE_SERDE_CLASS_CONFIG,
 AWSKafkaAvroSerDe.class.getName());
    props.put(ConsumerConfig.AUTO_OFFSET_RESET_CONFIG, "earliest");

    props.put(AWSSchemaRegistryConstants.AWS_REGION, "aws-region");
    props.put(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING, true);
```

```
    props.put(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.GENERIC_RECORD.getName());
 props.put(AWSSchemaRegistryConstants.DATA_FORMAT, DataFormat.AVRO.name());
```

2. Buat sebuah pengaliran dari topik avro-input.

```
StreamsBuilder builder = new StreamsBuilder();
final KStream<String, GenericRecord> source = builder.stream("avro-input");
```

3. Proses catatan data (contoh memfilter catatan-catatanyang nilai dari favorite_color adalah merah muda atau di mana nilainya adalah 15).

```
final KStream<String, GenericRecord> result = source
    .filter((key, value) -
> !"pink".equals(String.valueOf(value.get("favorite_color"))));
    .filter((key, value) -> !"15.0".equals(String.valueOf(value.get("amount"))));
```

4. Tulis hasilnya kembali ke topik avro-output.

```
result.to("avro-output");
```

5. Mulai aplikasi Apache Kafka Streams.

```
KafkaStreams streams = new KafkaStreams(builder.build(), props);
streams.start();
```

Hasil implementasi

Hasil ini menunjukkan proses penyaringan catatan yang disaring dalam langkah 3 sebagai favorite_color "merah muda" atau nilai "15,0".

Catatan sebelum penyaringan:

```
{"name": "Sansa", "favorite_number": 99, "favorite_color": "white"}
{"name": "Harry", "favorite_number": 10, "favorite_color": "black"}
{"name": "Hermione", "favorite_number": 1, "favorite_color": "red"}
```

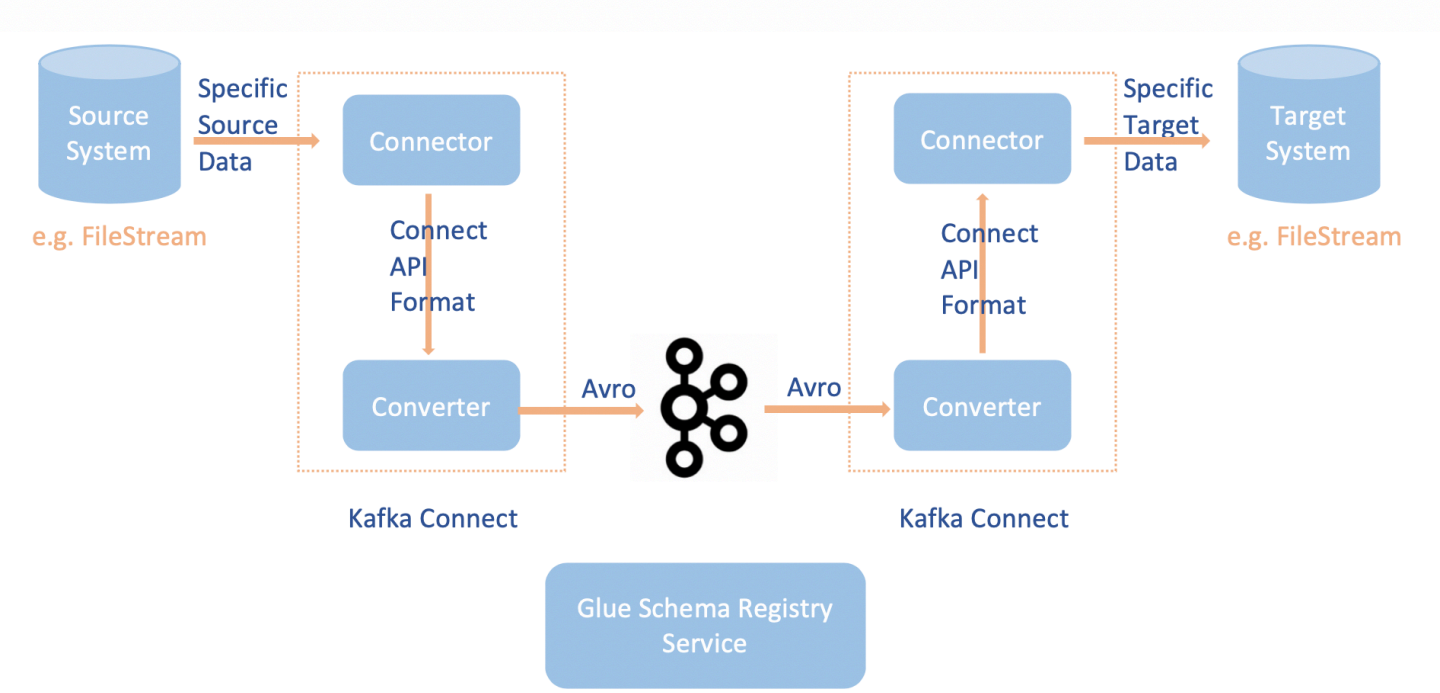
```
{"name": "Ron", "favorite_number": 0, "favorite_color": "pink"}
{"name": "Jay", "favorite_number": 0, "favorite_color": "pink"}

{"id": "commute_1","amount": 3.5}
{"id": "grocery_1","amount": 25.5}
{"id": "entertainment_1","amount": 19.2}
{"id": "entertainment_2","amount": 105}
 {"id": "commute_1","amount": 15}
```

Catatan setelah penyaringan:

```
{"name": "Sansa", "favorite_number": 99, "favorite_color": "white"}
{"name": "Harry", "favorite_number": 10, "favorite_color": "black"}
{"name": "Hermione", "favorite_number": 1, "favorite_color": "red"}
{"name": "Ron", "favorite_number": 0, "favorite_color": "pink"}

{"id": "commute_1","amount": 3.5}
{"id": "grocery_1","amount": 25.5}
{"id": "entertainment_1","amount": 19.2}
{"id": "entertainment_2","amount": 105}
```

Kasus penggunaan: Apache Kafka Connect

Integrasi Apache Kafka Connect dengan Registri Skema AWS Glue memungkinkan Anda untuk mendapatkan informasi skema dari konektor. Konverter Apache Kafka menentukan format data dalam Apache Kafka dan bagaimana menerjemahkannya ke data Apache Kafka Connect. Setiap pengguna Apache Kafka Connect akan diharuskan mengkonfigurasi konverter ini berdasarkan format data yang mereka inginkan di saat dimuat dari atau disimpan ke Apache Kafka. Dengan cara ini, Anda dapat menentukan konverter Anda sendiri untuk menerjemahkan data Apache Kafka Connect ke dalam jenis data yang digunakan dalam Registri Skema AWS Glue (misalnya: Avro) dan memanfaatkan serializer kami untuk mendaftarkan skemanya dan melakukan serialisasi. Kemudian konverter juga dapat menggunakan deserializer kami untuk melakukan deserialisasi data yang diterima dari Apache Kafka dan mengubahnya kembali ke data Apache Kafka Connect. Contoh diagram alur kerja ditunjukkan di bawah ini.



1. Menginstal proyek `aws-glue-schema-registry` dengan mengkloning Repositori Github untuk Registri Skema AWS Glue.

```
git clone git@github.com:awslabs/aws-glue-schema-registry.git
cd aws-glue-schema-registry
mvn clean install
mvn dependency:copy-dependencies
```

2. Jika Anda berencana untuk menggunakan Apache Kafka Connect dalam Mode Standalone, maka perbarui connect-standalone.properties dengan menggunakan langkah-langkah dalam petunjuk di bawah ini. Jika Anda berencana menggunakan Apache Kafka Connect dalam mode Distributed, perbarui connect-avro-distributed.properties menggunakan instruksi yang sama.

   a. Tambahkan properti ini juga ke file properti connect Apache Kafka:

```
key.converter.region=aws-region
value.converter.region=aws-region
key.converter.schemaAutoRegistrationEnabled=true
value.converter.schemaAutoRegistrationEnabled=true
key.converter.avroRecordType=GENERIC_RECORD
value.converter.avroRecordType=GENERIC_RECORD
```

   b. Tambahkan perintah di bawah ini ke bagian Mode peluncuran di kafka-run-classbawah.sh:

```
-cp $CLASSPATH:"<your AWS GlueSchema Registry base directory>/target/dependency/*"
```

3. Tambahkan perintah di bawah ini ke bagian Mode peluncuran di kafka-run-classbawah.sh

```
-cp $CLASSPATH:"<your AWS GlueSchema Registry base directory>/target/dependency/*"
```

Seharusnya terlihat seperti ini:

```
# Launch mode
if [ "x$DAEMON_MODE" = "xtrue" ]; then
  nohup "$JAVA" $KAFKA_HEAP_OPTS $KAFKA_JVM_PERFORMANCE_OPTS $KAFKA_GC_LOG_OPTS
 $KAFKA_JMX_OPTS $KAFKA_LOG4J_OPTS -cp $CLASSPATH:"/Users/johndoe/aws-glue-schema-
registry/target/dependency/*" $KAFKA_OPTS "$@" > "$CONSOLE_OUTPUT_FILE" 2>&1 < /dev/
null &
else
  exec "$JAVA" $KAFKA_HEAP_OPTS $KAFKA_JVM_PERFORMANCE_OPTS $KAFKA_GC_LOG_OPTS
 $KAFKA_JMX_OPTS $KAFKA_LOG4J_OPTS -cp $CLASSPATH:"/Users/johndoe/aws-glue-schema-
registry/target/dependency/*" $KAFKA_OPTS "$@"
fi
```

4. Jika menggunakan bash, jalankan perintah di bawah ini untuk menyiapkan CLASSPATH Anda di bash_profile Anda. Untuk shell yang lain, perbarui lingkungannya sesuai dengan itu.

```
echo 'export GSR_LIB_BASE_DIR=<>' >>~/.bash_profile
echo 'export GSR_LIB_VERSION=1.0.0' >>~/.bash_profile
echo 'export KAFKA_HOME=<your Apache Kafka installation directory>' >>~/.bash_profile
echo 'export CLASSPATH=$CLASSPATH:$GSR_LIB_BASE_DIR/avro-kafkaconnect-converter/
target/schema-registry-kafkaconnect-converter-$GSR_LIB_VERSION.jar:$GSR_LIB_BASE_DIR/
common/target/schema-registry-common-$GSR_LIB_VERSION.jar:$GSR_LIB_BASE_DIR/
avro-serializer-deserializer/target/schema-registry-serde-$GSR_LIB_VERSION.jar'
 >>~/.bash_profile
source ~/.bash_profile
```

5. (Opsional) Jika Anda ingin menguji dengan sebuah sumber file sederhana, maka lakukan kloning pada konektor sumber file.

```
git clone https://github.com/mmolimar/kafka-connect-fs.git
cd kafka-connect-fs/
```

a. Pada konfigurasi konektor sumber, edit format data ke Avro, pembaca file ke `AvroFileReader` dan perbarui contoh objek Avro dari path file yang Anda gunakan untuk membacanya. Sebagai contoh:

```
vim config/kafka-connect-fs.properties
```

```
fs.uris=<path to a sample avro object>
policy.regexp=^.*\.avro$
file_reader.class=com.github.mmolimar.kafka.connect.fs.file.reader.AvroFileReader
```

b. Instal konektor sumber.

```
mvn clean package
echo "export CLASSPATH=\$CLASSPATH:\"\$(find target/ -type f -name '*.jar'| grep
 '\-package' | tr '\n' ':')\"" >>~/.bash_profile
source ~/.bash_profile
```

c. Perbarui properti sink pada *<your Apache Kafka installation directory>*`/config/connect-file-sink.properties`, perbarui nama topik dan nama file yang keluar.

```
file=<output file full path>
topics=<my topic>
```

6. Mulai Konektor Sumber (dalam contoh ini, ia adalah konektor sumber file).

```
$KAFKA_HOME/bin/connect-standalone.sh $KAFKA_HOME/config/connect-
standalone.properties config/kafka-connect-fs.properties
```

7. Jalankan Konektor Sink (dalam contoh ini ia adalah konektor sink file).

```
$KAFKA_HOME/bin/connect-standalone.sh $KAFKA_HOME/config/connect-
standalone.properties $KAFKA_HOME/config/connect-file-sink.properties
```

Untuk contoh penggunaan Kafka Connect, lihat run-local-tests skrip.sh di bawah folder integration-tests di repositori Github untuk Schema Registry. AWS Glue

Migrasi dari registri skema pihak ketiga ke AWS Glue Schema Registry

Migrasi dari registri skema pihak ketiga ke Registri Skema AWS Glue yang memiliki dependensi pada registri skema pihak ketiga yang ada saat ini. Jika ada catatan dalam topik Apache Kafka yang dikirim menggunakan sebuah registri skema pihak ketiga, maka konsumen memerlukan registri skema pihak ketiga tersebut untuk melakukan deserialisasi pada catatan tersebut. `AWSKafkaAvroDeserializer` menyediakan kemampuan untuk menentukan kelas deserializer sekunder yang mengarahkan ke deserializer pihak ketiga dan digunakan untuk melakukan deserialisasi pada catatan-catatan tersebut.

Ada dua kriteria untuk mempensiunkan skema pihak ketiga. Pertama, pensiun dapat terjadi hanya setelah catatan dalam topik Apache Kafka yang menggunakan registri skema pihak ketiga yang tidak lagi diperlukan oleh dan untuk konsumen manapun. Kedua, pensiun dapat terjadi dengan habisnya usia dari topik Apache Kafka, tergantung pada periode retensi yang ditentukan untuk topik-topik tersebut. Perhatikan bahwa jika Anda memiliki topik yang memiliki retensi tak terbatas, maka Anda masih dapat bermigrasi ke Registri Skema AWS Glue tetapi Anda tidak akan dapat mempensiunkan registri skema pihak ketiga. Solusinya, Anda dapat menggunakan aplikasi atau Mirror Maker 2 untuk membaca dari topik saat ini dan menghasilkan topik baru dengan Registri Skema AWS Glue.

Untuk bermigrasi dari registri skema pihak ketiga ke Registri Skema AWS Glue:

1. Buat registri di Registri Skema AWS Glue, atau gunakan registri default.
2. Hentikan konsumennya. Modifikasi sehingga menyertakan Registri Skema AWS Glue sebagai deserializer utama, dan registri skema pihak ketiga sebagai yang sekunder.
   - Mengatur properti konsumen. Dalam contoh ini, secondary_deserializer diatur ke sebuah deserializer yang berbeda. Perilakunya adalah sebagai berikut: konsumen mengambil catatan dari Amazon MSK dan pertama kali mencoba untuk menggunakan `AWSKafkaAvroDeserializer`. Jika tidak dapat membaca byte ajaib yang berisi ID Skema Avro untuk skema dari Registri Skema AWS Glue, maka `AWSKafkaAvroDeserializer` akan mencoba untuk menggunakan kelas deserializer yang disediakan di secondary_deserializer. Properti spesifik untuk deserializer sekunder juga perlu disediakan dalam properti konsumen, seperti schema_registry_url_config dan specific_avro_reader_config, seperti yang ditunjukkan di bawah ini.

```
consumerProps.setProperty(ConsumerConfig.KEY_DESERIALIZER_CLASS_CONFIG,
 StringDeserializer.class.getName());
consumerProps.setProperty(ConsumerConfig.VALUE_DESERIALIZER_CLASS_CONFIG,
 AWSKafkaAvroDeserializer.class.getName());
```

```
consumerProps.setProperty(AWSSchemaRegistryConstants.AWS_REGION,
 KafkaClickstreamConsumer.gsrRegion);
consumerProps.setProperty(AWSSchemaRegistryConstants.SECONDARY_DESERIALIZER,
 KafkaAvroDeserializer.class.getName());
consumerProps.setProperty(KafkaAvroDeserializerConfig.SCHEMA_REGISTRY_URL_CONFIG,
 "URL for third-party schema registry");
consumerProps.setProperty(KafkaAvroDeserializerConfig.SPECIFIC_AVRO_READER_CONFIG,
 "true");
```

3. Mulai ulang konsumen.
4. Hentikan produsen dan arahkan produsen ke Registri Skema AWS Glue.
   a. Mengatur properti produsen. Dalam contoh ini, produsen akan menggunakan default-registry dan versi skema register otomatis.

```
producerProps.setProperty(ProducerConfig.KEY_SERIALIZER_CLASS_CONFIG,
 StringSerializer.class.getName());
producerProps.setProperty(ProducerConfig.VALUE_SERIALIZER_CLASS_CONFIG,
 AWSKafkaAvroSerializer.class.getName());
producerProps.setProperty(AWSSchemaRegistryConstants.AWS_REGION, "us-east-2");
producerProps.setProperty(AWSSchemaRegistryConstants.AVRO_RECORD_TYPE,
 AvroRecordType.SPECIFIC_RECORD.getName());
producerProps.setProperty(AWSSchemaRegistryConstants.SCHEMA_AUTO_REGISTRATION_SETTING,
 "true");
```

5. (Opsional) Secara manual, pindahkan skema yang ada dan versi skema dari registri skema pihak ketiga saat ini ke Registri Skema AWS Glue, baik untuk default-registry di Registri Skema AWS Glue atau registri non-default tertentu di Registri Skema AWS Glue. Hal ini dapat dilakukan dengan mengekspor skema dari registri skema pihak ketiga dalam format JSON dan menciptakan skema baru di Registri Skema AWS Glue dengan menggunakan AWS Management Console atau AWS CLI.

   Langkah ini mungkin penting jika Anda perlu mengaktifkan pemeriksaan kompatibilitas dengan versi skema sebelumnya untuk versi skema yang baru dibuat dengan menggunakan AWS CLI dan AWS Management Console, atau ketika produsen mengirim pesan dengan skema baru dengan pendaftaran otomatis versi skema yang sudah diaktifkan.
6. Mulai produsen.

Menghubungkan ke data

AWS GlueKoneksi adalah objek Katalog Data yang menyimpan kredensil login, string URI, informasi virtual private cloud (VPC), dan lainnya untuk penyimpanan data tertentu. AWS Glue crawler, pekerjaan, dan titik akhir pengembangan menggunakan koneksi untuk mengakses jenis penyimpanan data tertentu. Anda dapat menggunakan koneksi untuk sumber dan target, dan menggunakan kembali koneksi yang sama di beberapa crawler atau mengekstrak, mengubah, dan memuat (ETL) pekerjaan.

AWS Glue mendukung jenis koneksi berikut:

- Amazon DocumentDB
- OpenSearch Layanan Amazon, untuk digunakan dengan AWS Glue Spark.
- Amazon Redshift
- Azure Cosmos, untuk penggunaan Azure Cosmos DB untuk NoSQL dengan pekerjaan ETL AWS Glue
- Azure SQL, untuk digunakan dengan AWS Glue untuk Spark.
- Google BigQuery, untuk digunakan dengan AWS Glue untuk Spark.
- JDBC
- Kafka
- MongoDB
- MongoDB Atlas
- Salesforce
- SAP HANA, untuk digunakan dengan AWS Glue Spark.
- Snowflake, untuk digunakan dengan AWS Glue untuk Spark.
- Teradata Vantage, saat menggunakan untuk Spark. AWS Glue
- Vertica, untuk digunakan dengan AWS Glue untuk Spark.
- Berbagai penawaran Amazon Relational Database Service (Amazon RDS).
- Jaringan (menunjuk koneksi ke sumber data yang ada di Amazon Virtual Private Cloud (Amazon VPC))
- Aurora (didukung jika driver JDBC asli sedang digunakan. Tidak semua fitur driver dapat dimanfaatkan)

DenganAWS Glue Studio, Anda juga dapat membuat koneksi untuk konektor. Konektor adalah paket kode opsional yang membantu mengakses penyimpanan data di. AWS Glue Studio Untuk informasi selengkapnya, lihat Menggunakan konektor dan koneksi dengan AWS Glue Studio

Untuk informasi tentang cara menyambung ke database lokal, lihat Cara mengakses dan menganalisis penyimpanan data lokal menggunakan AWS Glue situs web AWS Big Data Blog.

Bagian ini mencakup topik-topik berikut untuk membantu Anda menggunakan AWS Glue koneksi:

- AWS Glueproperti koneksi
- Menyimpan kredensi koneksi di AWS Secrets Manager
- Menambahkan AWS Glue koneksi
- Menguji AWS Glue koneksi
- Mengkonfigurasi AWS panggilan untuk melalui VPC Anda
- Menghubungkan ke penyimpanan data JDBC di VPC
- Menggunakan koneksi MongoDB atau MongoDB Atlas
- Merayapi penyimpanan data Amazon S3 menggunakan titik akhir VPC
- Memecahkan masalah koneksi di AWS Glue
- Tutorial: Menggunakan AWS Glue Konektor untuk Elasticsearch

AWS Glueproperti koneksi

Topik ini mencakup informasi tentang properti untuk AWS Glue koneksi.

Topik

- Properti koneksi yang diperlukan
- AWS GlueProperti koneksi JDBC
- AWS GlueProperti koneksi MongoDB dan MongoDB Atlas
- Properti koneksi Salesforce
- Koneksi kepingan salju
- Koneksi Vertica
- Koneksi SAP HANA
- Koneksi Azure SQL
- Koneksi Teradata Vantage

- OpenSearch Koneksi layanan
- Koneksi Azure Cosmos
- AWS GlueProperti koneksi SSL
- Properti koneksi Apache Kafka untuk otentikasi klien
- BigQuery Koneksi Google
- Koneksi Vertica

Properti koneksi yang diperlukan

Ketika Anda menentukan sebuah koneksi pada konsol AWS Glue, Anda harus memberikan nilai untuk properti-properti berikut:

Nama koneksi

Masukkan nama unik untuk koneksi Anda.

Tipe koneksi

Pilih JDBC atau salah satu jenis koneksi tertentu.

Untuk detail tentang jenis koneksi JDBC, lihat the section called “Properti koneksi JDBC”

Pilih Jaringan untuk connect ke sumber data di lingkungan Amazon Virtual Private Cloud (Amazon VPC)).

Tergantung pada jenis yang Anda pilih, konsol AWS Glue menampilkan bidang lain yang diperlukan. Misalnya, jika Anda memilih Amazon RDS, maka Anda kemudian harus memilih mesin basis data.

Wajib koneksi SSL

Bila Anda memilih opsi ini, AWS Glue harus memverifikasi bahwa koneksi ke penyimpanan data terhubung melalui Lapisan Soket Aman (SSL) yang terpercaya.

Untuk informasi selengkapnya, termasuk opsi tambahan yang tersedia bila Anda memilih opsi ini, lihat the section called “Properti koneksi SSL”.

Pilih klaster MSK (Amazon Managed Streaming for Apache Kafka (MSK) saja)

Menentukan cluster MSK dari akun lain AWS .

URL server bootstrap Kafka (Kafka saja)

Menentukan daftar dipisahkan koma dari URL server bootstrap. Termasuk nomor port. Sebagai contoh: b-1.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094, b-2.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094, b-3.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094

AWS GlueProperti koneksi JDBC

AWS Glue dapat connect ke penyimpanan data berikut melalui koneksi JDBC:

- Amazon Redshift
- Amazon Aurora
- Microsoft SQL Server
- MySQL
- Oracle
- PostgreSQL
- Snowflake, saat menggunakan AWS Glue crawler.
- Aurora (didukung jika driver JDBC asli sedang digunakan. Tidak semua fitur driver dapat dimanfaatkan)
- Amazon RDS for MariaDB

Important

Saat ini, sebuah tugas ETL dapat menggunakan koneksi JDBC hanya dalam satu subnet saja. Jika Anda memiliki beberapa penyimpanan data dalam suatu pekerjaan, mereka harus berada di subnet yang sama, atau dapat diakses dari subnet.
Jika Anda memilih untuk membawa versi driver JDBC Anda sendiri untuk AWS Glue crawler, crawler Anda akan menggunakan sumber daya dalam pekerjaan dan AWS Glue Amazon S3 untuk memastikan driver yang Anda berikan dijalankan di lingkungan Anda. Penggunaan sumber daya tambahan akan tercermin di akun Anda. Selain itu, menyediakan driver JDBC Anda sendiri tidak berarti bahwa crawler dapat memanfaatkan semua fitur pengemudi. Driver terbatas pada properti yang dijelaskan dalam Mendefinisikan koneksi di Katalog Data.

Berikut ini adalah properti tambahan untuk jenis koneksi JDBC.

URL JDBC

Masukkan URL untuk penyimpanan data JDBC Anda. Untuk kebanyakan mesin basis data, bidang ini adalah dalam format berikut. Dalam format ini, ganti *protokol*, *host*, *port*, dan *db_name* dengan informasi Anda sendiri.

`jdbc:`*protocol*`://`*host*`:`*port*`/`*db_name*

Tergantung pada mesin basis data, format URL JDBC yang berbeda mungkin diperlukan. Format ini dapat memiliki penggunaan yang sedikit berbeda untuk penggunaan titik dua (:) dan garis miring (/) atau kata kunci yang berbeda untuk menentukan basis data.

Untuk JDBC yang akan connect ke penyimpanan data, diperlukan sebuah db_name di penyimpanan data. db_name digunakan untuk membuat koneksi jaringan dengan `username` dan `password`. Saat terhubung, AWS Glue dapat mengakses basis data lain di penyimpanan data untuk menjalankan sebuah crawler atau menjalankan tugas ETL.

Contoh URL JDBC berikut menunjukkan sintaksis untuk beberapa mesin basis data.

- Untuk connect ke penyimpanan data klaster Amazon Redshift dengan basis data `dev`:

  ```
  jdbc:redshift://xxx.us-east-1.redshift.amazonaws.com:8192/dev
  ```

- Untuk connect ke penyimpanan data Amazon RDS for MySQL dengan basis data `employee`:

  ```
  jdbc:mysql://xxx-cluster.cluster-xxx.us-east-1.rds.amazonaws.com:3306/
  employee
  ```

- Untuk connect ke penyimpanan data Amazon RDS for PostgreSQL dengan basis data `employee`:

  ```
  jdbc:postgresql://xxx-cluster.cluster-xxx.us-
  east-1.rds.amazonaws.com:5432/employee
  ```

- Untuk connect ke penyimpanan data Amazon RDS for Oracle data store dengan nama layanan `employee`:

  ```
  jdbc:oracle:thin://@xxx-cluster.cluster-xxx.us-
  east-1.rds.amazonaws.com:1521/employee
  ```

  Sintaksis untuk Amazon RDS for Oracle dapat mengikuti pola berikut. Dalam pola ini, ganti *host*, *port*, *service_name*, dan *SID* dengan informasi Anda sendiri.

  - `jdbc:oracle:thin://@host:port/service_name`
  - `jdbc:oracle:thin://@host:port:SID`
- Untuk connect ke penyimpanan data Amazon RDS for Microsoft SQL Server dengan basis data `employee`:

  ```
  jdbc:sqlserver://xxx-cluster.cluster-xxx.us-
  east-1.rds.amazonaws.com:1433;databaseName=employee
  ```

  Sintaksis untuk Amazon RDS for SQL Server dapat mengikuti pola berikut. Dalam pola ini, ganti *`server_name`*, *`port`*, dan *`db_name`* dengan informasi Anda sendiri.

  - `jdbc:sqlserver://server_name:port;database=db_name`
  - `jdbc:sqlserver://server_name:port;databaseName=db_name`
- Untuk terhubung ke Amazon Aurora PostgreSQL instance `employee` database, tentukan titik akhir untuk instance database, port, dan nama database:

  ```
  jdbc:postgresql://employee_instance_1.xxxxxxxxxxxx.us-
  east-2.rds.amazonaws.com:5432/employee
  ```
- Untuk terhubung ke penyimpanan Amazon RDS for MariaDB data dengan `employee` database, tentukan titik akhir untuk instance database, port, dan nama database:

  ```
  jdbc:mysql://xxx-cluster.cluster-xxx.aws-
  region.rds.amazonaws.com:3306/employee
  ```
-

  > **Warning**
  >
  > Koneksi Snowflake JDBC hanya didukung oleh crawler. AWS Glue Saat menggunakan konektor Snowflake dalam AWS Glue pekerjaan, gunakan jenis koneksi Snowflake.

  Untuk terhubung ke instance Snowflake dari `sample` database, tentukan titik akhir untuk instance snowflake, pengguna, nama database, dan nama peran. Anda dapat menambahkan `warehouse` parameter secara opsional.

  ```
  jdbc:snowflake://account_name.snowflakecomputing.com/?
  user=user_name&db=sample&role=role_name&warehouse=warehouse_name
  ```

> **Important**
>
> Untuk koneksi Snowflake melalui JDBC, urutan parameter dalam URL diberlakukan dan harus diurutkan sebagai`user`,,, dan. db `role_name` `warehouse`

- Untuk terhubung ke instance Snowflake dari `sample` database dengan tautan AWS pribadi, tentukan URL Snowflake JDBC sebagai berikut:

  `jdbc:snowflake://`*`account_name`*`.`*`region`*`.privatelink.snowflakecomputing.com/?user=`*`user_name`*`&db=sample&role=`*`role_name`*`&warehouse=`*`warehouse_name`*

nama pengguna

> **Note**
>
> Kami menyarankan Anda menggunakan AWS rahasia untuk menyimpan kredensyal koneksi alih-alih memasok nama pengguna dan kata sandi Anda secara langsung. Untuk informasi selengkapnya, lihat Menyimpan kredensi koneksi di AWS Secrets Manager.

Berikan nama pengguna yang memiliki izin untuk mengakses penyimpanan data JDBC.

Kata sandi

Masukkan kata sandi untuk nama pengguna yang memiliki izin akses ke penyimpanan data JDBC.

Port

Masukkan port yang digunakan dalam URL JDBC untuk terhubung ke instans Amazon RDS Oracle. Bidang ini hanya ditampilkan ketika Wajib koneksi SSL dipilih untuk instans Amazon RDS Oracle.

VPC

Pilih nama virtual private cloud (VPC) yang berisi penyimpanan data Anda. Konsol AWS Glue mencantumkan semua VPC untuk Wilayah saat ini.

> **Important**
>
> Saat bekerja melalui koneksi JDBC yang di-host AWS, seperti dengan data dari Snowflake, VPC Anda harus memiliki gateway NAT yang membagi lalu lintas menjadi

subnet publik dan pribadi. Subnet publik digunakan untuk koneksi ke sumber eksternal, dan subnet internal digunakan untuk diproses oleh. AWS Glue Untuk informasi tentang mengonfigurasi VPC Amazon Anda untuk koneksi eksternal, baca Connect to the internet atau jaringan lain menggunakan perangkat NAT dan. Menyiapkan Amazon VPC untuk koneksi JDBC ke penyimpanan data Amazon RDS AWS Glue

Subnet

Pilih subnet dalam VPC yang berisi penyimpanan data Anda. Konsol AWS Glue mencantumkan semua subnet untuk penyimpanan data di VPC Anda.

Grup keamanan

Pilih grup keamanan yang dikaitkan dengan penyimpanan data Anda. AWS Glue memerlukan satu atau beberapa grup keamanan dengan aturan sumber inbound yang memungkinkan AWS Glue untuk connect. Konsol AWS Glue mencantumkan semua grup keamanan yang diberikan akses masuk ke VPC Anda. AWS Glue mengaitkan grup keamanan ini dengan antarmuka jaringan elastis yang dilampirkan pada subnet VPC Anda.

Nama Kelas Pengemudi JDBC - opsional

Berikan nama kelas driver JDBC kustom:

- Postgres - org.PostgreSQL.driver
- MySQL - com.mysql.jdbc.driver, com.mysql.cj.jdbc.driver
- Redshift — com.amazon.redshift.jdbc.driver, com.amazon.redshift.jdbc42.driver
- Oracle — oracle.jdbc.driver. OracleDriver
- SQL Server - com.Microsoft.SqlServer.JDBC.SQL ServerDriver

Jalur S3 Driver JDBC - opsional

Berikan lokasi Amazon S3 ke driver JDBC khusus. Ini adalah jalur absolut ke file.jar. Jika Anda ingin menyediakan driver JDBC Anda sendiri untuk terhubung ke sumber data Anda untuk database yang didukung crawler Anda, Anda dapat menentukan nilai untuk parameter `customJdbcDriverS3PathdancustomJdbcDriverClassName`.

Menggunakan driver JDBC yang disediakan oleh pelanggan terbatas pada yang diperlukan. Properti koneksi yang diperlukan

AWS GlueProperti koneksi MongoDB dan MongoDB Atlas

Berikut ini adalah properti tambahan untuk jenis koneksi MongoDB atau MongoDB Atlas.

URL MongoDB

Masukkan URL untuk penyimpanan data MongoDB atau MongoDB Atlas Anda:

- Untuk MongoDB: mongodb: //host:port/database. Host dapat berupa nama host, alamat IP, atau soket domain UNIX. Jika string koneksi tidak menentukan port, ia menggunakan port MongoDB default, 27017.
- Untuk MongoDB Atlas: mongodb+srv: //server.example.com/database. Host dapat berupa nama host yang mengikuti sesuai dengan catatan DNS SRV. Format SRV tidak memerlukan port dan akan menggunakan port MongoDB default, 27017.

nama pengguna

> **Note**
>
> Kami menyarankan Anda menggunakan AWS rahasia untuk menyimpan kredensyal koneksi alih-alih memasok nama pengguna dan kata sandi Anda secara langsung. Untuk informasi selengkapnya, lihat Menyimpan kredensi koneksi di AWS Secrets Manager.

Berikan nama pengguna yang memiliki izin untuk mengakses penyimpanan data JDBC.

Kata sandi

Masukkan kata sandi untuk nama pengguna yang memiliki izin akses ke penyimpanan data MongoDB atau MongoDB Atlas.

Properti koneksi Salesforce

Berikut ini adalah properti tambahan untuk jenis koneksi Salesforce.

- `ENTITY_NAME`(String) - (Diperlukan) Digunakan untuk Baca/Tulis. Nama Objek Anda di Salesforce.

- `API_VERSION`(String) - (Diperlukan) Digunakan untuk Baca/Tulis. Salesforce Rest API versi yang ingin Anda gunakan.
- `SELECTED_FIELDS`(Daftar<String>) - Default: kosong (SELECT *). Digunakan untuk Baca. Kolom yang ingin Anda pilih untuk objek.
- `FILTER_PREDICATE`(String) - Default: kosong. Digunakan untuk Baca. Itu harus dalam format Spark SQL.
- `QUERY`(String) - Default: kosong. Digunakan untuk Baca. Kueri SQL Spark penuh.
- `PARTITION_FIELD`(String) - Digunakan untuk Baca. Bidang yang akan digunakan untuk mempartisi kueri.
- `LOWER_BOUND`(String) - Digunakan untuk Baca. Nilai batas bawah inklusif dari bidang partisi yang dipilih.
- `UPPER_BOUND`(String) - Digunakan untuk Baca. Nilai batas atas eksklusif dari bidang partisi yang dipilih.
- `NUM_PARTITIONS`(Integer) - Default: 1. Digunakan untuk Baca. Jumlah partisi untuk dibaca.
- `IMPORT_DELETED_RECORDS`(String) - Default: FALSE. Digunakan untuk membaca. Untuk mendapatkan catatan hapus saat melakukan kueri.
- `WRITE_OPERATION`(String) - Default: SISIPKAN. Digunakan untuk menulis. Nilai harus INSERT, UPDATE, UPSERT, DELETE.
- `ID_FIELD_NAMES`(String) - Default: null. Digunakan hanya untuk UPSERT.

Koneksi kepingan salju

Properti berikut digunakan untuk mengatur koneksi Snowflake yang digunakan dalam pekerjaan AWS Glue ETL. Saat merayapi Snowflake, gunakan koneksi JDBC.

URL Kepingan Salju

URL titik akhir Snowflake Anda. Untuk informasi selengkapnya tentang URL titik akhir Snowflake, lihat Menghubungkan ke Akun Anda di dokumentasi Snowflake.

AWS Rahasia

Nama Rahasia dari sebuah rahasia di AWS Secrets Manager AWS Glue akan terhubung ke Snowflake menggunakan `sfUser` dan `sfPassword` kunci rahasia Anda.

Peran kepingan salju (opsional)

Peran keamanan kepingan salju AWS Glue akan digunakan saat menghubungkan.

Gunakan properti berikut saat mengonfigurasi sambungan ke titik akhir Snowflake yang dihosting di Amazon VPC menggunakan. AWS PrivateLink

VPC

Pilih nama virtual private cloud (VPC) yang berisi penyimpanan data Anda. Konsol AWS Glue mencantumkan semua VPC untuk Wilayah saat ini.

Subnet

Pilih subnet dalam VPC yang berisi penyimpanan data Anda. Konsol AWS Glue mencantumkan semua subnet untuk penyimpanan data di VPC Anda.

Grup keamanan

Pilih grup keamanan yang dikaitkan dengan penyimpanan data Anda. AWS Glue memerlukan satu atau beberapa grup keamanan dengan aturan sumber inbound yang memungkinkan AWS Glue untuk connect. Konsol AWS Glue mencantumkan semua grup keamanan yang diberikan akses masuk ke VPC Anda. AWS Glue mengaitkan grup keamanan ini dengan antarmuka jaringan elastis yang dilampirkan pada subnet VPC Anda.

Koneksi Vertica

Gunakan properti berikut untuk menyiapkan koneksi Vertica untuk pekerjaan AWS Glue ETL.

Tuan Rumah Vertica

Nama host instalasi Vertica Anda.

Pelabuhan Vertica

Port instalasi Vertica Anda tersedia melalui.

AWS Rahasia

Nama Rahasia dari sebuah rahasia di AWS Secrets Manager AWS Glue akan terhubung ke Vertica menggunakan kunci rahasia Anda.

Gunakan properti berikut saat mengonfigurasi sambungan ke titik akhir Vertica yang dihosting di Amazon VPC.

VPC

Pilih nama virtual private cloud (VPC) yang berisi penyimpanan data Anda. Konsol AWS Glue mencantumkan semua VPC untuk Wilayah saat ini.

Subnet

Pilih subnet dalam VPC yang berisi penyimpanan data Anda. Konsol AWS Glue mencantumkan semua subnet untuk penyimpanan data di VPC Anda.

Grup keamanan

Pilih grup keamanan yang dikaitkan dengan penyimpanan data Anda. AWS Glue memerlukan satu atau beberapa grup keamanan dengan aturan sumber inbound yang memungkinkan AWS Glue untuk connect. Konsol AWS Glue mencantumkan semua grup keamanan yang diberikan akses masuk ke VPC Anda. AWS Glue mengaitkan grup keamanan ini dengan antarmuka jaringan elastis yang dilampirkan pada subnet VPC Anda.

Koneksi SAP HANA

Gunakan properti berikut untuk mengatur koneksi SAP HANA untuk pekerjaan AWS Glue ETL.

SAP HANA URL

URL JDBC SAP.

URL SAP HANA JDBC ada dalam bentuk `jdbc:sap://saphanaHostname:saphanaPort/?databaseName=saphanaDBname,ParameterNa`

AWS Glue memerlukan parameter URL JDBC berikut:

- `databaseName`— Database default di SAP HANA untuk terhubung ke.

AWS Rahasia

Nama Rahasia dari sebuah rahasia di AWS Secrets Manager AWS Glue akan terhubung ke SAP HANA menggunakan kunci rahasia Anda.

Gunakan properti berikut saat mengonfigurasi koneksi ke titik akhir SAP HANA yang dihosting di Amazon VPC:

VPC

Pilih nama virtual private cloud (VPC) yang berisi penyimpanan data Anda. Konsol AWS Glue mencantumkan semua VPC untuk Wilayah saat ini.

Subnet

Pilih subnet dalam VPC yang berisi penyimpanan data Anda. Konsol AWS Glue mencantumkan semua subnet untuk penyimpanan data di VPC Anda.

Grup keamanan

Pilih grup keamanan yang dikaitkan dengan penyimpanan data Anda. AWS Glue memerlukan satu atau beberapa grup keamanan dengan aturan sumber inbound yang memungkinkan AWS Glue untuk connect. Konsol AWS Glue mencantumkan semua grup keamanan yang diberikan akses masuk ke VPC Anda. AWS Glue mengaitkan grup keamanan ini dengan antarmuka jaringan elastis yang dilampirkan pada subnet VPC Anda.

Koneksi Azure SQL

Gunakan properti berikut untuk menyiapkan koneksi Azure SQL untuk pekerjaan AWS Glue ETL.

URL SQL Azure

URL JDBC dari titik akhir Azure SQL.

URL harus dalam format berikut:`jdbc:sqlserver://`*`databaseServerName`*`:`*`databasePort`*`;databaseName=`*`azuresqlDB`*

AWS Glue membutuhkan properti URL berikut:

- `databaseName`— Database default di Azure SQL untuk terhubung ke.

Untuk informasi selengkapnya tentang URL JDBC untuk Instans Terkelola Azure SQL, lihat dokumentasi Microsoft.

AWS Rahasia

Nama Rahasia dari sebuah rahasia di AWS Secrets Manager AWS Glue akan terhubung ke Azure SQL menggunakan kunci rahasia Anda.

Koneksi Teradata Vantage

Gunakan properti berikut untuk menyiapkan koneksi Teradata Vantage untuk pekerjaan ETL. AWS Glue

URL Teradata

Untuk menyambung ke instance Teradata, tentukan nama host untuk instance database dan parameter Teradata yang relevan:

```
jdbc:teradata://teradataHostname/ParameterName=ParameterValue,ParameterName=Pa
```

AWS Glue mendukung parameter URL JDBC berikut:

- `DATABASE_NAME`— Database default di Teradata untuk terhubung ke.
- `DBS_PORT`- Menentukan port Teradata, jika tidak standar.

AWS Rahasia

Nama Rahasia dari sebuah rahasia di AWS Secrets Manager AWS Glue akan terhubung ke Teradata Vantage menggunakan kunci rahasia Anda.

Gunakan properti berikut saat mengonfigurasi sambungan ke titik akhir Teradata Vantage yang dihosting di Amazon VPC:

VPC

Pilih nama virtual private cloud (VPC) yang berisi penyimpanan data Anda. Konsol AWS Glue mencantumkan semua VPC untuk Wilayah saat ini.

Subnet

Pilih subnet dalam VPC yang berisi penyimpanan data Anda. Konsol AWS Glue mencantumkan semua subnet untuk penyimpanan data di VPC Anda.

Grup keamanan

Pilih grup keamanan yang dikaitkan dengan penyimpanan data Anda. AWS Glue memerlukan satu atau beberapa grup keamanan dengan aturan sumber inbound yang memungkinkan AWS Glue untuk connect. Konsol AWS Glue mencantumkan semua grup keamanan yang diberikan akses masuk ke VPC Anda. AWS Glue mengaitkan grup keamanan ini dengan antarmuka jaringan elastis yang dilampirkan pada subnet VPC Anda.

OpenSearch Koneksi layanan

Gunakan properti berikut untuk menyiapkan koneksi OpenSearch Layanan untuk pekerjaan AWS Glue ETL.

Titik akhir domain

Titik akhir domain OpenSearch Layanan Amazon akan memiliki formulir default berikut, https://search - *domainName* -. *unstructuredIdContent* *wilayah* .es.amazonaws.com. Untuk informasi selengkapnya tentang mengidentifikasi titik akhir domain Anda, lihat Membuat dan mengelola domain OpenSearch Layanan Amazon di dokumentasi OpenSearch Layanan Amazon.

Port

Port terbuka di titik akhir.

AWS Rahasia

Nama Rahasia dari sebuah rahasia di AWS Secrets Manager AWS Glue akan terhubung ke OpenSearch Layanan menggunakan kunci rahasia Anda.

Gunakan properti berikut saat mengonfigurasi sambungan ke titik akhir OpenSearch Layanan yang dihosting di Amazon VPC:

VPC

Pilih nama virtual private cloud (VPC) yang berisi penyimpanan data Anda. Konsol AWS Glue mencantumkan semua VPC untuk Wilayah saat ini.

Subnet

Pilih subnet dalam VPC yang berisi penyimpanan data Anda. Konsol AWS Glue mencantumkan semua subnet untuk penyimpanan data di VPC Anda.

Grup keamanan

Pilih grup keamanan yang dikaitkan dengan penyimpanan data Anda. AWS Glue memerlukan satu atau beberapa grup keamanan dengan aturan sumber inbound yang memungkinkan AWS Glue untuk connect. Konsol AWS Glue mencantumkan semua grup keamanan yang diberikan akses masuk ke VPC Anda. AWS Glue mengaitkan grup keamanan ini dengan antarmuka jaringan elastis yang dilampirkan pada subnet VPC Anda.

Koneksi Azure Cosmos

Gunakan properti berikut untuk menyiapkan koneksi Azure Cosmos untuk pekerjaan AWS Glue ETL.

URI Titik Akhir Akun Azure Cosmos DB

Titik akhir yang digunakan untuk terhubung ke Azure Cosmos. Untuk informasi selengkapnya, lihat dokumentasi Azure.

AWS Rahasia

Nama Rahasia dari sebuah rahasia di AWS Secrets Manager AWS Glue akan terhubung ke Azure Cosmos menggunakan kunci rahasia Anda.

AWS GlueProperti koneksi SSL

Berikut ini adalah rincian tentang properti koneksi Required SSL.

Jika Anda tidak memerlukan koneksi SSL, AWS Glue abaikan kegagalan saat menggunakan SSL untuk mengenkripsi koneksi ke penyimpanan data. Lihat dokumentasi untuk penyimpanan data Anda untuk melihat petunjuk konfigurasi. Saat Anda memilih opsi ini, pernyataan job run, crawler, atau ETL di titik akhir pengembangan gagal saat AWS Glue tidak dapat terhubung.

> **Note**
>
> Snowflake mendukung koneksi SSL secara default, jadi properti ini tidak berlaku untuk Snowflake.

Opsi ini divalidasi pada sisi klien AWS Glue. Untuk koneksi JDBC, AWS Glue hanya menghubungkan melalui SSL dengan sertifikat dan validasi nama host. Support koneksi SSL tersedia untuk:

- Basis data Oracle
- Microsoft SQL Server
- PostgreSQL
- Amazon Redshift
- MySQL (instans Amazon RDS saja)
- Amazon Aurora MySQL (instans Amazon RDS saja)

- Amazon Aurora PostgreSQL (Hanya instans Amazon RDS)
- Kafka, yang meliputi Amazon Managed Streaming for Apache Kafka
- MongoDB

> **Note**
>
> Untuk mengaktifkan sebuah penyimpanan data Amazon RDS Oracle untuk menggunakan Wajib koneksi SSL, Anda harus membuat dan melampirkan grup pilihan untuk instans Oracle.
>
> 1. Masuk ke AWS Management Console dan buka konsol Amazon RDS di https://console.aws.amazon.com/rds/.
> 2. Tambahkan Grup opsi untuk instans Amazon RDS Oracle. Untuk informasi selengkapnya tentang cara menambahkan grup opsi di konsol Amazon RDS, lihat Membuat grup opsi
> 3. Tambahkan sebuah Opsi ke grup opsi untuk SSL. Port yang Anda tentukan untuk SSL kemudian digunakan saat Anda membuat URL koneksi JDBC AWS Glue untuk instans Amazon RDS Oracle. Untuk informasi selengkapnya tentang cara menambahkan opsi di konsol Amazon RDS, lihat Menambahkan Opsi ke Grup Opsi dalam Panduan Pengguna Amazon RDS. Untuk informasi selengkapnya tentang opsi Oracle SSL, lihat Oracle SSL dalam Panduan Pengguna Amazon RDS.
> 4. Pada konsol AWS Glue, buat koneksi ke instans Amazon RDS Oracle. Dalam definisi koneksi, pilih Wajib koneksi SSL. Ketika diminta, masukkan Port yang Anda gunakan di opsi Amazon RDS Oracle SSL.

Properti opsional tambahan berikut tersedia ketika Wajib koneksi SSL dipilih untuk sebuah koneksi:

Sertifikat JDBC kustom di S3

Jika Anda memiliki sertifikat yang saat ini Anda gunakan untuk komunikasi SSL dengan basis data on-premise atau cloud, maka Anda dapat menggunakan sertifikat tersebut untuk koneksi SSL ke sumber data atau target AWS Glue. Masukkan lokasi Amazon Simple Storage Service (Amazon S3) yang berisi sertifikat akar kustom. AWS Glue menggunakan sertifikat ini untuk membuat koneksi SSL ke basis data. AWS Glue hanya menangani sertifikat X.509 saja. Sertifikat harus dikodekan-DER dan disediakan dalam format PEM encoding base64.

Jika bidang ini dibiarkan kosong, sertifikat default akan digunakan.

String sertifikat JDBC kustom

Masukkan informasi sertifikat yang spesifik untuk basis data JDBC Anda. String ini digunakan untuk pencocokan domain atau pencocokan nama yang dibedakan (DN). Untuk Basis Data Oracle, string ini memetakan ke parameter `SSL_SERVER_CERT_DN` di bagian keamanan file `tnsnames.ora`. Untuk Microsoft SQL Server, string ini digunakan sebagai `hostNameInCertificate`.

Berikut ini adalah contoh untuk parameter `SSL_SERVER_CERT_DN` Basis Data Oracle.

```
cn=sales,cn=OracleContext,dc=us,dc=example,dc=com
```

Lokasi sertifikat CA pribadi Kafka

Jika Anda memiliki sertifikat yang saat ini Anda gunakan untuk komunikasi SSL dengan penyimpanan data Kafka Anda, maka Anda dapat menggunakan sertifikat tersebut dengan koneksi AWS Glue. Opsi ini diperlukan untuk penyimpanan data Kafka, dan opsional untuk penyimpanan Amazon Managed Streaming for Apache Kafka data. Masukkan lokasi Amazon Simple Storage Service (Amazon S3) yang berisi sertifikat akar kustom. AWS Glue menggunakan sertifikat ini untuk membuat koneksi SSL ke penyimpanan data Kafka. AWS Glue hanya menangani sertifikat X.509 saja. Sertifikat harus dikodekan-DER dan disediakan dalam format PEM encoding base64.

Lewati validasi sertifikat

Pilih kotak centang Lewati validasi sertifikat untuk melewatkan validasi sertifikat kustom oleh AWS Glue. Jika Anda memilih untuk memvalidasi, AWS Glue akan memvalidasi algoritme tanda tangan dan algoritme kunci publik subjek untuk sertifikat. Jika sertifikat gagal validasi, maka setiap tugas ETL atau crawler yang menggunakan koneksi tersebut akan gagal.

Algoritme tanda tangan yang diizinkan adalah SHA256withRSA, SHA384withRSA, atau SHA512withRSA. Untuk algoritme kunci publik subjek, panjang kunci minimal harus 2048.

Lokasi keystore klien Kafka

Lokasi Amazon S3 dari file keystore klien untuk autentikasi sisi klien Kafka. Path harus dalam bentuk s3://bucket/prefix/filename.jks. Ini harus diakhiri dengan nama file dan ekstensi.jks.

Kata sandi keystore klien Kafka (opsional)

Kata sandi untuk mengakses keystore yang disediakan.

Kata sandi kunci klien Kafka (opsional)

Sebuah keystore dapat terdiri dari beberapa kunci, jadi ini adalah kata sandi untuk mengakses kunci klien yang akan digunakan dengan kunci sisi server Kafka.

Properti koneksi Apache Kafka untuk otentikasi klien

AWS Gluemendukung kerangka kerja Simple Authentication and Security Layer (SASL) untuk otentikasi saat Anda membuat koneksi Apache Kafka. Kerangka SASL mendukung berbagai mekanisme otentikasi, dan AWS Glue menawarkan SCRAM (nama pengguna dan kata sandi), GSSAPI (protokol Kerberos), dan protokol PLAIN.

Gunakan AWS Glue Studio untuk mengkonfigurasi salah satu metode otentikasi klien berikut. Untuk informasi selengkapnya, lihat Membuat koneksi untuk konektor di panduan AWS Glue Studio pengguna.

- Tidak ada - Tidak ada otentikasi. Ini berguna jika membuat koneksi untuk tujuan pengujian.
- SASL/SCRAM-SHA-512 - Memilih metode otentikasi ini akan memungkinkan Anda untuk menentukan kredensyal otentikasi. Ada dua opsi yang tersedia:
  - Gunakan AWS Secrets Manager (disarankan) - jika Anda memilih opsi ini, Anda dapat menyimpan nama pengguna dan kata sandi Anda di AWS Secrets Manager dan membiarkan AWS Glue mengaksesnya bila diperlukan. Tentukan rahasia yang menyimpan kredensyal otentikasi SSL atau SASL. Untuk informasi selengkapnya, lihat Menyimpan kredensi koneksi di AWS Secrets Manager.
  - Berikan nama pengguna dan kata sandi secara langsung.
- SASL/GSSAPI (Kerberos) - jika Anda memilih opsi ini, Anda dapat memilih lokasi file keytab, file krb5.conf dan masukkan nama utama Kerberos dan nama layanan Kerberos. Lokasi untuk file tab tombol dan file krb5.conf harus berada di lokasi Amazon S3. Karena MSK belum mendukung SASL/GSSAPI, opsi ini hanya tersedia untuk cluster Apache Kafka yang dikelola pelanggan. Untuk informasi selengkapnya, lihat Dokumentasi MIT Kerberos: Keytab.
- SASL/PLAIN - pilih metode otentikasi ini untuk menentukan kredensyal otentikasi. Ada dua opsi yang tersedia:
  - Gunakan AWS Secrets Manager (disarankan) - jika Anda memilih opsi ini, Anda dapat menyimpan kredensyal Anda di AWS Secrets Manager dan membiarkan AWS Glue mengakses informasi bila diperlukan. Tentukan rahasia yang menyimpan kredensyal otentikasi SSL atau SASL.

  - Berikan nama pengguna dan kata sandi secara langsung.
- Otentikasi Klien SSL - jika Anda memilih opsi ini, Anda dapat memilih lokasi keystore klien Kafka dengan menjelajahi Amazon S3. Secara opsional, Anda dapat memasukkan kata sandi keystore klien Kafka dan kata sandi kunci klien Kafka.

BigQuery Koneksi Google

Properti berikut digunakan untuk mengatur BigQuery koneksi Google yang digunakan dalam pekerjaan AWS Glue ETL. Untuk informasi selengkapnya, lihat the section called “BigQuery koneksi”.

AWS Rahasia

Nama Rahasia dari sebuah rahasia di AWS Secrets Manager AWS Glue Pekerjaan ETL akan terhubung ke Google BigQuery menggunakan `credentials` kunci rahasia Anda.

Koneksi Vertica

Properti berikut digunakan untuk mengatur koneksi Vertica yang digunakan dalam pekerjaan AWS Glue ETL. Untuk informasi selengkapnya, lihat the section called “Koneksi Vertica”.

Menyimpan kredensi koneksi di AWS Secrets Manager

Kami menyarankan Anda menggunakan AWS Secrets Manager untuk menyediakan kredensil koneksi untuk penyimpanan data Anda. Menggunakan Secrets Manager dengan cara ini memungkinkan AWS Glue akses rahasia Anda saat runtime untuk pekerjaan ETL dan crawler berjalan, dan membantu menjaga kredensil Anda tetap aman.

Prasyarat

Untuk menggunakan Secrets Manager denganAWS Glue, Anda harus memberikan peran IAM Anda untuk AWS Glue izin untuk mengambil nilai rahasia. Kebijakan AWS terkelola `AWSGlueServiceRole` tidak menyertakan AWS Secrets Manager izin. Misalnya kebijakan IAM, lihat Contoh: Izin untuk mengambil nilai rahasia di AWS Secrets ManagerPanduan Pengguna.

Bergantung pada pengaturan jaringan Anda, Anda mungkin juga perlu membuat titik akhir VPC untuk membuat koneksi pribadi antara VPC dan Secrets Manager. Untuk informasi selengkapnya, lihat Menggunakan titik AWS Secrets Manager akhir VPC.

Untuk membuat rahasia untuk AWS Glue

1. Ikuti petunjuk di Buat dan kelola rahasia di Panduan AWS Secrets Manager Pengguna. Contoh berikut JSON menunjukkan cara menentukan kredensil Anda di tab Plaintext saat Anda membuat rahasia untuk. AWS Glue

```
{
  "username": "EXAMPLE-USERNAME",
  "password": "EXAMPLE-PASSWORD"
}
```

2. Kaitkan rahasia Anda dengan koneksi menggunakan AWS Glue Studio antarmuka. Untuk petunjuk terperinci, lihat Membuat koneksi untuk konektor di Panduan AWS Glue Studio Pengguna.

Menambahkan AWS Glue koneksi

Anda dapat terhubung ke sumber data di AWS Glue untuk Spark secara terprogram. Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark

Anda juga dapat menggunakan AWS Glue konsol untuk menambah, mengedit, menghapus, dan menguji koneksi. Untuk informasi tentang koneksi AWS Glue, lihat Menghubungkan ke data.

Untuk menambahkan koneksi AWS Glue

1. Masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/.
2. Pada panel navigasi, di Katalog data, pilih Koneksi.
3. Pilih Tambahkan koneksi dan kemudian selesaikan penuntun, masukkan properti koneksi seperti yang diterangkan dalam the section called “AWS Glueproperti koneksi”.

Menghubungkan ke Amazon Redshift di AWS Glue Studio

> **Note**
>
> Anda dapat menggunakan Spark AWS Glue untuk membaca dan menulis ke tabel di Amazon Redshift database di luar. AWS Glue Studio Untuk mengonfigurasi Amazon Redshift AWS Glue pekerjaan secara terprogram, lihatKoneksi Redshift.

AWS Gluemenyediakan dukungan bawaan untuk Amazon Redshift. AWS Glue Studiomenyediakan antarmuka visual untuk terhubung, membuat pekerjaan integrasi data Amazon Redshift, dan menjalankannya pada runtime Spark AWS Glue Studio tanpa server.

Topik

Membuat Amazon Redshift koneksi

Izin diperlukan

Izin tambahan diperlukan untuk menggunakan Amazon Redshift cluster dan lingkungan tanpa Amazon Redshift server. Untuk informasi selengkapnya tentang cara menambahkan izin ke pekerjaan ETL, lihat Meninjau izin IAM yang diperlukan untuk pekerjaan ETL.

- pergeseran merah: DescribeClusters
- redshift-tanpa server: ListWorkgroups
- redshift-tanpa server: ListNamespaces

Gambaran Umum

Saat menambahkan Amazon Redshift koneksi, Anda dapat memilih Amazon Redshift koneksi yang ada atau membuat koneksi baru saat menambahkan sumber Data - Node Redshift masuk. AWS Glue Studio

AWS Gluemendukung Amazon Redshift cluster dan lingkungan tanpa Amazon Redshift server. Saat Anda membuat koneksi, lingkungan Amazon Redshift tanpa server menampilkan label tanpa server di sebelah opsi koneksi.

Untuk informasi selengkapnya tentang cara membuat Amazon Redshift sambungan, lihat Memindahkan data ke dan dari Amazon Redshift.

Membuat simpul Amazon Redshift sumber

Izin diperlukan

AWS Glue Studiopekerjaan yang menggunakan sumber Amazon Redshift data memerlukan izin tambahan. Untuk informasi selengkapnya tentang cara menambahkan izin ke pekerjaan ETL, lihat Meninjau izin IAM yang diperlukan untuk pekerjaan ETL.

Izin berikut diperlukan untuk menggunakan Amazon Redshift koneksi.

- data pergeseran merah: ListSchemas
- data pergeseran merah: ListTables
- data pergeseran merah: DescribeTable
- data pergeseran merah: ExecuteStatement
- data pergeseran merah: DescribeStatement
- data pergeseran merah: GetStatementResult

Menambahkan sumber Amazon Redshift data

Untuk menambahkan Sumber Data — Amazon Redshift node:

1. Pilih jenis Amazon Redshift akses:
   - Koneksi data langsung (disarankan) - pilih opsi ini jika Anda ingin mengakses Amazon Redshift data Anda secara langsung. Ini adalah opsi yang disarankan dan juga default.
   - Data Catalog tables— pilih opsi ini jika Anda memiliki tabel Katalog Data yang ingin Anda gunakan.
2. Jika Anda memilih Koneksi data langsung, pilih koneksi untuk sumber Amazon Redshift data Anda. Ini mengasumsikan bahwa koneksi sudah ada dan Anda dapat memilih dari koneksi yang ada. Jika Anda perlu membuat koneksi, pilih Buat koneksi Redshift. Untuk informasi selengkapnya, lihat Ikhtisar penggunaan konektor dan koneksi.

Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti. Informasi tentang koneksi terlihat, termasuk URL, grup keamanan, subnet, zona ketersediaan, deskripsi, dan stempel waktu yang dibuat (UTC) dan terakhir diperbarui (UTC).

3. Pilih opsi Amazon Redshift sumber:

   - Pilih satu tabel — ini adalah tabel yang berisi data yang ingin Anda akses dari satu Amazon Redshift tabel.
   - Masukkan kueri kustom — memungkinkan Anda mengakses kumpulan data dari beberapa Amazon Redshift tabel berdasarkan kueri kustom Anda.

4. Jika Anda memilih satu tabel, pilih Amazon Redshift skema. Daftar skema yang tersedia untuk dipilih ditentukan oleh tabel yang dipilih.

   Atau, pilih Masukkan kueri khusus. Pilih opsi ini untuk mengakses dataset kustom dari beberapa Amazon Redshift tabel. Saat Anda memilih opsi ini, masukkan Amazon Redshift kueri.

   Saat menghubungkan ke lingkungan Amazon Redshift tanpa server, tambahkan izin berikut ke kueri kustom:

   ```
   GRANT SELECT ON ALL TABLES IN <schema> TO PUBLIC
   ```

   Anda dapat memilih skema Infer untuk membaca skema berdasarkan kueri yang Anda masukkan. Anda juga dapat memilih editor kueri Open Redshift untuk memasukkan kueri. Amazon Redshift Untuk informasi selengkapnya, lihat Menanyakan database menggunakan editor kueri.

5. Di Performa dan keamanan, pilih direktori pementasan Amazon S3 dan peran IAM.

   - Direktori pementasan Amazon S3 — pilih lokasi Amazon S3 untuk data pementasan sementara.
   - Peran IAM — pilih peran IAM yang dapat menulis ke lokasi Amazon S3 yang Anda pilih.

6. Dalam parameter Custom Redshift - opsional, masukkan parameter dan nilai.

Membuat node Amazon Redshift target

Izin diperlukan

AWS Glue Studio pekerjaan menggunakan target Amazon Redshift data memerlukan izin tambahan. Untuk informasi selengkapnya tentang cara menambahkan izin ke pekerjaan ETL, lihat Meninjau izin IAM yang diperlukan untuk pekerjaan ETL.

Izin berikut diperlukan untuk menggunakan Amazon Redshift koneksi.

- data pergeseran merah: ListSchemas
- data pergeseran merah: ListTables

Menambahkan node Amazon Redshift target

Untuk membuat node Amazon Redshift target:

1. Pilih Amazon Redshift tabel yang ada sebagai target, atau masukkan nama tabel baru.
2. Saat Anda menggunakan target Data - Node target Redshift, Anda dapat memilih dari opsi berikut:

   - APPEND - Jika tabel sudah ada, dump semua data baru ke dalam tabel sebagai insert. Jika tabel tidak ada, buat dan kemudian masukkan semua data baru.

     Selain itu, centang kotak jika Anda ingin memperbarui (UPSERT) catatan yang ada di tabel target. Tabel harus ada terlebih dahulu, jika tidak operasi akan gagal.
   - MERGE - AWS Glue akan memperbarui atau menambahkan data ke tabel target Anda berdasarkan kondisi yang Anda tentukan.

     > **Note**
     >
     > Untuk menggunakan tindakan penggabunganAWS Glue, Anda harus mengaktifkan fungsionalitas Amazon Redshift gabungan. Untuk petunjuk tentang cara mengaktifkan penggabungan untuk Amazon Redshift instans Anda, lihat MENGGABUNGKAN (pratinjau).

     Pilih opsi:

- Pilih kunci dan tindakan sederhana — pilih kolom yang akan digunakan sebagai kunci yang cocok antara data sumber dan kumpulan data target Anda.

  Tentukan opsi berikut jika dicocokkan:

  - Perbarui catatan dalam kumpulan data target Anda dengan data dari sumber.
  - Hapus catatan dalam kumpulan data target Anda.

  Tentukan opsi berikut jika tidak cocok:

  - Masukkan data sumber sebagai baris baru ke dalam kumpulan data target Anda.
  - Jangan lakukan apa-apa.
- Masukkan pernyataan MERGE kustom — Anda kemudian dapat memilih pernyataan Validasi Gabungan untuk memverifikasi bahwa pernyataan tersebut valid atau tidak valid.

- TRUNCATE — Jika tabel sudah ada, potong data tabel dengan terlebih dahulu membersihkan isi tabel target. Jika truncate berhasil, maka masukkan semua data. Jika tabel tidak ada, buat tabel dan masukkan semua data. Jika truncate tidak berhasil, operasi akan gagal.
- DROP — Jika tabel sudah ada, hapus metadata tabel dan data. Jika penghapusan berhasil, maka masukkan semua data. Jika tabel tidak ada, buat tabel dan masukkan semua data. Jika drop tidak berhasil, operasi akan gagal.
- CREATE - Buat tabel baru dengan nama default. Jika nama tabel sudah ada, buat tabel baru dengan nama postfix dari nama `job_datetime` untuk keunikan. Ini akan memasukkan semua data ke dalam tabel baru. Jika tabel ada, nama tabel akhir akan memiliki postfix yang ditambahkan. Jika tabel tidak ada, tabel akan dibuat. Dalam kedua kasus, tabel baru akan dibuat.

Opsi lanjutan

Lihat Menggunakan konektor Amazon Redshift Spark pada AWS Glue.

Menghubungkan ke Azure Cosmos DB di AWS Glue Studio

AWS Gluemenyediakan dukungan bawaan untuk Azure Cosmos DB. AWS Glue Studiomenyediakan antarmuka visual untuk terhubung ke Azure Cosmos DB untuk NoSQL, pekerjaan integrasi data penulis, dan menjalankannya pada runtime Spark tanpa server. AWS Glue Studio

Topik

- Membuat koneksi Azure Cosmos DB

- Membuat node sumber Azure Cosmos DB
- Membuat simpul target Azure Cosmos DB
- Opsi lanjutan

Membuat koneksi Azure Cosmos DB

Prasyarat:

- Di Azure, Anda perlu mengidentifikasi atau membuat Kunci DB Azure Cosmos untuk digunakan oleh,. AWS Glue `cosmosKey` Untuk informasi selengkapnya, lihat Akses aman ke data di Azure Cosmos DB di dokumentasi Azure.

Untuk mengonfigurasi koneksi ke Azure Cosmos DB:

1. DiAWS Secrets Manager, buat rahasia menggunakan Azure Cosmos DB Key Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *`secretName`* untuk langkah selanjutnya.
   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci `spark.cosmos.accountKey` dengan nilai `CosmosKey`.*
2. Di AWS Glue konsol, buat koneksi dengan mengikuti langkah-langkah dithe section called “Menambahkan AWS Glue koneksi”. Setelah membuat koneksi, simpan nama koneksi, *`connectionName`*, untuk digunakan di masa mendatang. AWS Glue
   - Saat memilih jenis Koneksi, pilih Azure Cosmos DB.
   - Saat memilih AWSSecret, berikan *`secretName`*.

Membuat node sumber Azure Cosmos DB

Prasyarat yang dibutuhkan

- Koneksi AWS Glue Azure Cosmos DB, dikonfigurasi dengan AWS Secrets Manager rahasia, seperti yang dijelaskan di bagian sebelumnya,. the section called “Membuat koneksi Azure Cosmos DB”
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.

- Azure Cosmos DB untuk wadah NoSQL yang ingin Anda baca. Anda akan memerlukan informasi identifikasi untuk wadah.

  Sebuah Azure Cosmos untuk wadah NoSQL diidentifikasi oleh database dan wadahnya. Anda harus memberikan nama database, *CosmosDBName*, dan container *cosmosContainerName*, saat menghubungkan ke Azure Cosmos for NoSQL API.

Menambahkan sumber data Azure Cosmos DB

Untuk menambahkan sumber Data — Azure Cosmos DB node:

1. Pilih koneksi untuk sumber data Azure Cosmos DB Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi Azure Cosmos DB. Untuk informasi lebih lanjut, lihat bagian sebelumnyathe section called “Membuat koneksi Azure Cosmos DB”.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.
2. Pilih Nama Database Cosmos DB — berikan nama database yang ingin Anda baca, *CosmosDBName*.
3. Pilih Azure Cosmos DB Container — berikan nama wadah yang ingin Anda baca,. *cosmosContainerName*
4. Secara opsional, pilih Azure Cosmos DB Custom Query — berikan kueri SQL SELECT untuk mengambil informasi spesifik dari Azure Cosmos DB.
5. Di properti Custom Azure Cosmos, masukkan parameter dan nilai sesuai kebutuhan.

Membuat simpul target Azure Cosmos DB

Prasyarat yang dibutuhkan

- Koneksi AWS Glue Azure Cosmos DB, dikonfigurasi dengan AWS Secrets Manager rahasia, seperti yang dijelaskan di bagian sebelumnya,. the section called “Membuat koneksi Azure Cosmos DB”
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- Tabel Azure Cosmos DB yang ingin Anda tulis. Anda akan memerlukan informasi identifikasi untuk wadah. Anda harus membuat wadah sebelum memanggil metode koneksi.

Sebuah Azure Cosmos untuk wadah NoSQL diidentifikasi oleh database dan wadahnya. Anda harus memberikan nama database, *CosmosDBName*, dan container *cosmosContainerName*, saat menghubungkan ke Azure Cosmos for NoSQL API.

Menambahkan target data Azure Cosmos DB

Untuk menambahkan target Data — Azure Cosmos DB node:

1. Pilih koneksi untuk sumber data Azure Cosmos DB Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi Azure Cosmos DB. Untuk informasi lebih lanjut, lihat bagian sebelumnya,the section called “Membuat koneksi Azure Cosmos DB”.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.
2. Pilih Nama Database Cosmos DB — berikan nama database yang ingin Anda baca, *CosmosDBName*.
3. Pilih Azure Cosmos DB Container — berikan nama wadah yang ingin Anda baca,. *cosmosContainerName*
4. Di properti Custom Azure Cosmos, masukkan parameter dan nilai sesuai kebutuhan.

Opsi lanjutan

Anda dapat memberikan opsi lanjutan saat membuat node Azure Cosmos DB. Opsi ini sama dengan yang tersedia saat pemrograman AWS Glue untuk skrip Spark.

Lihat the section called “Koneksi Azure Cosmos DB”.

Menghubungkan ke Azure SQL di AWS Glue Studio

AWS Gluemenyediakan dukungan bawaan untuk Azure SQL. AWS Glue Studiomenyediakan antarmuka visual untuk terhubung ke Azure SQL, pekerjaan integrasi data penulis, dan menjalankannya pada runtime Spark tanpa AWS Glue Studio server.

Topik

- Membuat koneksi Azure SQL

- Membuat node sumber Azure SQL
- Membuat node target Azure SQL
- Opsi lanjutan

Membuat koneksi Azure SQL

Untuk terhubung ke Azure SQL dariAWS Glue, Anda harus membuat dan menyimpan kredensyal Azure SQL Anda secara AWS Secrets Manager rahasia, lalu mengaitkan rahasia itu dengan koneksi Azure SQL. AWS Glue

Untuk mengkonfigurasi koneksi ke Azure SQL:

1. DiAWS Secrets Manager, buat rahasia menggunakan kredensyal Azure SQL Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci user dengan nilai AzuresQLUsername.*
   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci password dengan nilai AzuresQLPassword.*

2. Di AWS Glue konsol, buat koneksi dengan mengikuti langkah-langkah dithe section called "Menambahkan AWS Glue koneksi". Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk digunakan di masa mendatang. AWS Glue

   - Saat memilih jenis Koneksi, pilih Azure SQL.
   - Saat menyediakan URL Azure SQL, berikan URL endpoint JDBC.

     URL harus dalam format berikut:
     `jdbc:sqlserver://`*databaseServerName*`:`*databasePort*`;databaseName=`*azuresqlDBnam*

     AWS Gluemembutuhkan properti URL berikut:

     - `databaseName`— Database default di Azure SQL untuk terhubung ke.

     Untuk informasi selengkapnya tentang URL JDBC untuk Instans Terkelola Azure SQL, lihat dokumentasi Microsoft.

   - Saat memilih AWSSecret, berikan *secretName*.

Membuat node sumber Azure SQL

Prasyarat yang dibutuhkan

- Koneksi AWS Glue Azure SQL, dikonfigurasi dengan AWS Secrets Manager rahasia, seperti yang dijelaskan di bagian sebelumnya,. the section called “Membuat koneksi Azure SQL”
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- *`Tabel Azure SQL yang ingin Anda baca, TableName.`*

  Tabel Azure SQL diidentifikasi oleh database, skema, dan nama tabelnya. Anda harus memberikan nama database dan nama tabel saat menghubungkan ke Azure SQL. Anda juga harus memberikan skema jika bukan default, “publik”. Database disediakan melalui properti URL di *`connectionName`*, skema dan nama tabel melalui. `dbtable`

Menambahkan sumber data Azure SQL

Untuk menambahkan sumber Data — Azure SQL node:

1. Pilih koneksi untuk sumber data Azure SQL Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi SQL Azure. Untuk informasi lebih lanjut, lihat bagian sebelumnya,the section called “Membuat koneksi Azure SQL”.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.
2. Pilih opsi Azure SQL Source:
   - Pilih satu tabel — akses semua data dari satu tabel.
   - Masukkan kueri kustom — akses kumpulan data dari beberapa tabel berdasarkan kueri kustom Anda.
3. Jika Anda memilih satu tabel, masukkan *`TableName`*.

   Jika Anda memilih Masukkan kueri kustom, masukkan kueri TransactSQL SELECT.
4. Di properti Custom Azure SQL, masukkan parameter dan nilai sesuai kebutuhan.

Membuat node target Azure SQL

Prasyarat yang dibutuhkan

- Koneksi AWS Glue Azure SQL, dikonfigurasi dengan AWS Secrets Manager rahasia, seperti yang dijelaskan di bagian sebelumnya,. the section called “Membuat koneksi Azure SQL”
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- *`Tabel Azure SQL yang ingin Anda tulis, TableName.`*

  Tabel Azure SQL diidentifikasi oleh database, skema, dan nama tabelnya. Anda harus memberikan nama database dan nama tabel saat menghubungkan ke Azure SQL. Anda juga harus memberikan skema jika bukan default, “publik”. Database disediakan melalui properti URL di *`connectionName`*, skema dan nama tabel melalui. `dbtable`

Menambahkan target data Azure SQL

Untuk menambahkan target Data - Azure SQL node:

1. Pilih koneksi untuk sumber data Azure SQL Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi SQL Azure. Untuk informasi lebih lanjut, lihat bagian sebelumnya,the section called “Membuat koneksi Azure SQL”.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.
2. Konfigurasikan nama Tabel dengan menyediakan *`TableName`*.
3. Di properti Custom Azure SQL, masukkan parameter dan nilai sesuai kebutuhan.

Opsi lanjutan

Anda dapat memberikan opsi lanjutan saat membuat node Azure SQL. Opsi ini sama dengan yang tersedia saat pemrograman AWS Glue untuk skrip Spark.

Lihat the section called “Koneksi Azure SQL”.

Menghubungkan ke Google BigQuery di AWS Glue Studio

> **Note**
>
> Anda dapat menggunakan Spark AWS Glue untuk membaca dari dan menulis ke tabel di Google BigQuery dalam versi AWS Glue 4.0 dan yang lebih baru. Untuk mengonfigurasi Google BigQuery dengan AWS Glue pekerjaan secara terprogram, lihat BigQuery koneksi.

AWS Glue Studiomenyediakan antarmuka visual untuk terhubung BigQuery, membuat pekerjaan integrasi data, dan menjalankannya pada runtime Spark AWS Glue Studio tanpa server.

Topik

Membuat BigQuery koneksi

Untuk terhubung ke Google BigQuery dariAWS Glue, Anda harus membuat dan menyimpan kredensi Google Cloud Platform Anda secara AWS Secrets Manager rahasia, lalu mengaitkan rahasia itu dengan koneksi Google BigQuery AWS Glue.

Untuk mengkonfigurasi koneksi ke BigQuery:

1. Di Google Cloud Platform, buat dan identifikasi sumber daya yang relevan:
   - Buat atau identifikasi proyek GCP yang berisi BigQuery tabel yang ingin Anda sambungkan.
   - Aktifkan BigQuery API. Untuk informasi selengkapnya, lihat Menggunakan BigQuery Storage Read API untuk membaca data tabel.
2. Di Google Cloud Platform, buat dan ekspor kredensional akun layanan:

   Anda dapat menggunakan panduan BigQuery kredensial untuk mempercepat langkah ini: Buat kredensial.

   Untuk membuat akun layanan di GCP, ikuti tutorial yang tersedia di Buat akun layanan.

- Saat memilih proyek, pilih proyek yang berisi BigQuery tabel Anda.
- Saat memilih peran IAM GCP untuk akun layanan Anda, tambahkan atau buat peran yang akan memberikan izin yang sesuai untuk menjalankan BigQuery pekerjaan untuk membaca, menulis, atau membuat tabel. BigQuery

Untuk membuat kredensi untuk akun layanan Anda, ikuti tutorial yang tersedia di Buat kunci akun layanan.

- Saat memilih jenis kunci, pilih JSON.

Anda seharusnya sudah mengunduh file JSON dengan kredensi untuk akun layanan Anda. Itu terlihat serupa dengan yang berikut ini:

```
{
  "type": "service_account",
  "project_id": "*****",
  "private_key_id": "*****",
  "private_key": "*****",
  "client_email": "*****",
  "client_id": "*****",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "*****",
  "universe_domain": "googleapis.com"
}
```

3. base64 menyandikan file kredensi yang Anda unduh. Pada AWS CloudShell sesi atau serupa, Anda dapat melakukan ini dari baris perintah dengan menjalankan`cat` *`credentialsFile.json`* `| base64 -w 0`. Pertahankan output dari perintah ini, *`CredentialString`*.
4. DiAWS Secrets Manager, buat rahasia menggunakan kredensi Google Cloud Platform Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *`secretName`* untuk langkah selanjutnya.

   - *Saat memilih pasangan* ***kunci/nilai, buat pasangan*** *untuk kunci credentials dengan nilai CredentialString.*

5. Di Katalog AWS Glue Data, buat koneksi dengan mengikuti langkah-langkah dihttps://docs.aws.amazon.com/glue/latest/dg/console-connections.html. Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk langkah selanjutnya.

   - Saat memilih jenis Koneksi, pilih Google BigQuery.
   - Saat memilih AWSSecret, berikan *secretName*.

6. Berikan peran IAM yang terkait dengan izin AWS Glue pekerjaan Anda untuk membaca *secretName*.
7. Dalam konfigurasi AWS Glue pekerjaan Anda, berikan *ConnectionName* sebagai koneksi jaringan tambahan.

Membuat simpul BigQuery sumber

Prasyarat yang dibutuhkan

- Koneksi BigQuery tipe AWS Glue Data Catalog
- AWS Secrets ManagerRahasia untuk BigQuery kredensi Google Anda, yang digunakan oleh koneksi.
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- Nama dan kumpulan data tabel dan proyek Google Cloud terkait yang ingin Anda baca.

Menambahkan sumber BigQuery data

Untuk menambahkan sumber Data — BigQuery node:

1. Pilih koneksi untuk sumber BigQuery data Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat BigQuery koneksi. Untuk informasi selengkapnya, lihat Ikhtisar penggunaan konektor dan koneksi.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.

2. Identifikasi BigQuery data apa yang ingin Anda baca, lalu pilih opsi BigQuery Sumber

   - Pilih satu tabel — memungkinkan Anda untuk menarik semua data dari tabel.
   - Masukkan kueri khusus — memungkinkan Anda menyesuaikan data mana yang diambil dengan memberikan kueri.

3. Jelaskan data yang ingin Anda baca

   (Wajib) setel Proyek Induk ke proyek yang berisi tabel Anda, atau proyek induk penagihan, jika relevan.

   Jika Anda memilih satu tabel, atur Tabel ke nama BigQuery tabel Google dalam format berikut: `[dataset].[table]`

   Jika Anda memilih kueri, berikan ke Query. Dalam kueri Anda, lihat tabel dengan nama tabel yang sepenuhnya memenuhi syarat, dalam format:`[project].[dataset].[tableName]`.
4. Menyediakan BigQuery properti

   Jika Anda memilih satu tabel, Anda tidak perlu memberikan properti tambahan.

   Jika Anda memilih kueri, Anda harus memberikan BigQuery properti Google Kustom berikut:

   - Atur `viewsEnabled` menjadi BETUL.
   - Setel `materializationDataset` ke kumpulan data. Prinsipal GCP yang diautentikasi oleh kredensi yang disediakan melalui AWS Glue koneksi harus dapat membuat tabel dalam kumpulan data ini.

Membuat node BigQuery target

Prasyarat yang dibutuhkan

- Koneksi BigQuery tipe AWS Glue Data Catalog
- AWS Secrets ManagerRahasia untuk BigQuery kredensi Google Anda, yang digunakan oleh koneksi.
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- Nama dan kumpulan data tabel dan proyek Google Cloud yang sesuai yang ingin Anda tulis.

Menambahkan target BigQuery data

Untuk menambahkan target Data — BigQuery node:

1. Pilih koneksi untuk target BigQuery data Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat BigQuery koneksi. Untuk informasi selengkapnya, lihat Ikhtisar penggunaan konektor dan koneksi.

Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.

2. Identifikasi BigQuery tabel apa yang ingin Anda tulis, lalu pilih metode Tulis.

   - Direct — menulis untuk BigQuery langsung menggunakan BigQuery Storage Write API.
   - Tidak langsung — menulis ke Google Cloud Storage, lalu salin ke BigQuery.

   Jika Anda ingin menulis secara tidak langsung, berikan lokasi GCS tujuan dengan bucket GCS Sementara. Anda perlu memberikan konfigurasi tambahan dalam AWS Glue koneksi Anda. Untuk informasi selengkapnya, lihat Menggunakan penulisan tidak langsung dengan Google BigQuery.

3. Jelaskan data yang ingin Anda baca

   (Wajib) setel Proyek Induk ke proyek yang berisi tabel Anda, atau proyek induk penagihan, jika relevan.

   Jika Anda memilih satu tabel, atur Tabel ke nama BigQuery tabel Google dalam format berikut: `[dataset].[table]`

Opsi lanjutan

Anda dapat memberikan opsi lanjutan saat membuat BigQuery simpul. Opsi ini sama dengan yang tersedia saat pemrograman AWS Glue untuk skrip Spark.

Lihat referensi opsi BigQuery koneksi di panduan AWS Glue pengembang.

Menghubungkan ke MongoDB di AWS Glue Studio

AWS Gluemenyediakan dukungan bawaan untuk MongoDB. AWS Glue Studiomenyediakan antarmuka visual untuk terhubung ke MongoDB, membuat pekerjaan integrasi data, dan menjalankannya di AWS Glue Studio runtime Spark tanpa server.

Topik

- Membuat Koneksi MongoDB
- Membuat simpul sumber MongoDB
- Membuat simpul target MongoDB

- Opsi lanjutan

Membuat Koneksi MongoDB

Prasyarat:

- Jika instans MongoDB Anda ada di Amazon VPC, konfigurasikan Amazon VPC untuk memungkinkan pekerjaan AWS Glue Anda berkomunikasi dengan instans MongoDB tanpa lalu lintas melintasi internet publik.

  Di Amazon VPC, identifikasi atau buat grup VPC, Subnet, dan Keamanan yang AWS Glue akan digunakan saat menjalankan pekerjaan. Selain itu, Anda perlu memastikan Amazon VPC dikonfigurasi untuk mengizinkan lalu lintas jaringan antara instans MongoDB Anda dan lokasi ini. Berdasarkan tata letak jaringan Anda, ini mungkin memerlukan perubahan pada aturan grup keamanan, ACL Jaringan, Gateway NAT, dan koneksi Peering.

Untuk mengkonfigurasi koneksi ke MongoDB:

1. Secara opsional, diAWS Secrets Manager, buat rahasia menggunakan kredensyal MongoDB Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci username dengan nilai MongoDbuser.*

     *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci password dengan nilai MongoDBPass.*

2. Di AWS Glue konsol, buat koneksi dengan mengikuti langkah-langkah dithe section called “Menambahkan AWS Glue koneksi”. Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk digunakan di masa mendatang. AWS Glue

   - Saat memilih jenis Koneksi, pilih MongoDB atau MongoDB Atlas.
   - Saat memilih URL MongoDB atau URL MongoDB Atlas, berikan nama host instance MongoDB Anda.

     URL MongoDB disediakan dalam format.
     `mongodb://`*mongoHost*`:`*mongoPort*`/`*mongoDBname*

URL Atlas MongoDB disediakan dalam format. `mongodb +srv://`*`mongoHost`*`:`*`mongoPort`*`/`*`mongoDBname`*

Menyediakan database default untuk koneksi, *`MongoDBName`* adalah opsional.

- Jika Anda memilih untuk membuat rahasia Secrets Manager, pilih jenis AWS Secrets Manager Credential.

  Kemudian, di AWSSecret berikan *`secretName`*.

- *`Jika Anda memilih untuk memberikan`* ***`Nama Pengguna dan kata sandi`****`, berikan MongoDbuser dan MongoDBPass.`*

3. Dalam situasi berikut, Anda mungkin memerlukan konfigurasi tambahan:

   - Untuk instans MongoDB yang dihosting AWS di VPC Amazon
     - Anda harus memberikan informasi koneksi Amazon VPC ke AWS Glue koneksi yang menentukan kredensyal keamanan MongoDB Anda. Saat membuat atau memperbarui koneksi Anda, atur grup VPC, Subnet, dan Keamanan dalam opsi Jaringan.

Setelah membuat koneksi AWS Glue MongoDB, Anda harus melakukan langkah-langkah berikut sebelum menjalankan pekerjaan Anda: AWS Glue

- Saat bekerja dengan AWS Glue pekerjaan di editor visual, Anda harus memberikan informasi koneksi Amazon VPC agar pekerjaan Anda terhubung ke MongoDB. Identifikasi lokasi yang sesuai di Amazon VPC dan berikan ke koneksi AWS Glue MongoDB Anda.
- Jika Anda memilih untuk membuat rahasia Secrets Manager, berikan peran IAM yang terkait dengan izin AWS Glue pekerjaan Anda untuk membaca *`secretName`*.

Membuat simpul sumber MongoDB

Prasyarat yang dibutuhkan

- Koneksi AWS Glue MongoDB, seperti yang dijelaskan di bagian sebelumnya,. the section called “Membuat Koneksi MongoDB”
- Jika Anda memilih untuk membuat rahasia Secrets Manager, izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- Koleksi MongoDB yang ingin Anda baca. Anda akan memerlukan informasi identifikasi untuk koleksi.

Koleksi MongoDB diidentifikasi oleh nama database dan nama koleksi, MongoDBName, MongoDBCollection.

Menambahkan sumber data MongoDB

Untuk menambahkan sumber Data — simpul MongoDB:

1. Pilih koneksi untuk sumber data MongoDB Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi MongoDB. Untuk informasi selengkapnya, lihat bagian sebelumnyathe section called “Membuat Koneksi MongoDB”.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.
2. Pilih Database. Masukkan *MongoDBName*.
3. Pilih Koleksi. Masukkan *MongoDBCollection*.
4. Pilih Partitioner, Partition size (MB) dan Partition key. Untuk informasi selengkapnya tentang parameter partisi, lihatthe section called ““ConnectionType”: “mongodb” sebagai sumber”.
5. Di properti MongoDB Kustom, masukkan parameter dan nilai sesuai kebutuhan.

Membuat simpul target MongoDB

Prasyarat yang dibutuhkan

- Koneksi AWS Glue MongoDB, dikonfigurasi dengan rahasia, seperti AWS Secrets Manager yang dijelaskan di bagian sebelumnya,. the section called “Membuat Koneksi MongoDB”
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- *Tabel MongoDB yang ingin Anda tulis, TableName.*

Menambahkan target data MongoDB

Untuk menambahkan target Data - simpul MongoDB:

1. Pilih koneksi untuk sumber data MongoDB Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi MongoDB.

Untuk informasi selengkapnya lihat bagian sebelumnyathe section called “Membuat Koneksi MongoDB”.

Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.

2. Pilih Database. Masukkan *MongoDBName*.
3. Pilih Koleksi. Masukkan *MongoDBCollection*.
4. Pilih Partitioner, Partition size (MB) dan Partition key. Untuk informasi selengkapnya tentang parameter partisi, lihatthe section called ““ConnectionType”: “mongodb” sebagai sumber”.
5. Pilih Coba Ulang Menulis jika diinginkan.
6. Di properti MongoDB Kustom, masukkan parameter dan nilai sesuai kebutuhan.

Opsi lanjutan

Anda dapat memberikan opsi lanjutan saat membuat simpul MongoDB. Opsi ini sama dengan yang tersedia saat pemrograman AWS Glue untuk skrip Spark.

Lihat the section called “Koneksi MongoDB”.

Menghubungkan ke OpenSearch Layanan di AWS Glue Studio

AWS Gluemenyediakan dukungan bawaan untuk Amazon OpenSearch Service. AWS Glue Studiomenyediakan antarmuka visual untuk terhubung ke Amazon OpenSearch Service, membuat pekerjaan integrasi data, dan menjalankannya di runtime Spark AWS Glue Studio tanpa server. Fitur ini tidak kompatibel dengan OpenSearch Layanan tanpa server.

Topik

- Membuat koneksi OpenSearch Layanan
- Membuat node sumber OpenSearch Layanan
- Membuat node target OpenSearch Service
- Opsi lanjutan

Membuat koneksi OpenSearch Layanan

Prasyarat:

- Identifikasi titik akhir domain, *AoSendPoint* dan port, *AoSport* yang ingin Anda baca, atau buat sumber daya dengan mengikuti petunjuk dalam dokumentasi Layanan Amazon. OpenSearch Untuk informasi selengkapnya tentang membuat domain, lihat Membuat dan mengelola domain OpenSearch Layanan Amazon di dokumentasi OpenSearch Layanan Amazon.

  Titik akhir domain OpenSearch Layanan Amazon akan memiliki formulir default berikut, https:// search - *domainName* -. *unstructuredIdContent wilayah* .es.amazonaws.com. Untuk informasi selengkapnya tentang mengidentifikasi titik akhir domain Anda, lihat Membuat dan mengelola domain OpenSearch Layanan Amazon di dokumentasi OpenSearch Layanan Amazon.

  Identifikasi atau hasilkan kredensyal otentikasi dasar HTTP, AOSuser, dan AOSPassWord untuk domain Anda.

Untuk mengkonfigurasi koneksi ke OpenSearch Layanan:

1. DiAWS Secrets Manager, buat rahasia menggunakan kredensyal OpenSearch Layanan Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

   - *Saat memilih **pasangan kunci/nilai**, buat pasangan untuk kunci opensearch.net.http.auth.user dengan nilai AOSUSER.*
   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci opensearch.net.http.auth.pass dengan nilai AOSPassword.*

2. Di AWS Glue konsol, buat koneksi dengan mengikuti langkah-langkah dithe section called “Menambahkan AWS Glue koneksi”. Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk digunakan di masa mendatang. AWS Glue

   - Saat memilih jenis Koneksi, pilih OpenSearch Layanan.
   - Saat memilih titik akhir Domain, berikan *AoSendPoint*.
   - Saat memilih port, sediakan *AOsport*.
   - Saat memilih AWSSecret, berikan *secretName*.

Membuat node sumber OpenSearch Layanan

Prasyarat yang dibutuhkan

- Koneksi AWS Glue OpenSearch layanan, dikonfigurasi dengan AWS Secrets Manager rahasia, seperti yang dijelaskan di bagian sebelumnya,the section called “Membuat koneksi OpenSearch Layanan”.
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- Indeks OpenSearch Layanan yang ingin Anda baca, *AOsIndex*.

Menambahkan sumber data OpenSearch Layanan

Untuk menambahkan sumber Data — Node OpenSearch layanan:

1. Pilih sambungan untuk sumber data OpenSearch Layanan Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi OpenSearch Layanan. Untuk informasi lebih lanjut lihat bagian sebelumnya,the section called “Membuat koneksi OpenSearch Layanan”.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.
2. Berikan Indeks, indeks yang ingin Anda baca.
3. Secara opsional, berikan Query, OpenSearch kueri untuk memberikan hasil yang lebih spesifik. Untuk informasi lebih lanjut tentang menulis OpenSearch pertanyaan, konsultasikanthe section called “Baca dari OpenSearch Layanan”.
4. Di properti OpenSearch Layanan Kustom, masukkan parameter dan nilai sesuai kebutuhan.

Membuat node target OpenSearch Service

Prasyarat yang dibutuhkan

- Koneksi AWS Glue OpenSearch layanan, dikonfigurasi dengan AWS Secrets Manager rahasia, seperti yang dijelaskan di bagian sebelumnya,the section called “Membuat koneksi OpenSearch Layanan”.
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- Indeks OpenSearch Layanan yang ingin Anda tulis, *AOsIndex*.

Menambahkan target data OpenSearch Layanan

Untuk menambahkan target Data — Node OpenSearch layanan:

1. Pilih sambungan untuk sumber data OpenSearch Layanan Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi OpenSearch Layanan. Untuk informasi lebih lanjut, lihat bagian sebelumnya,the section called “Membuat koneksi OpenSearch Layanan”.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.
2. Berikan Indeks, indeks yang ingin Anda baca.
3. Di properti OpenSearch Layanan Kustom, masukkan parameter dan nilai sesuai kebutuhan.

Opsi lanjutan

Anda dapat memberikan opsi lanjutan saat membuat node OpenSearch Layanan. Opsi ini sama dengan yang tersedia saat memprogram AWS Glue skrip Spark.

Lihat the section called “OpenSearch Koneksi layanan”.

Menghubungkan ke Salesforce di AWS Glue Studio

Salesforce menyediakan perangkat lunak manajemen hubungan pelanggan (CRM) yang membantu Anda dengan penjualan, layanan pelanggan, e-commerce, dan banyak lagi. Jika Anda pengguna Salesforce, Anda dapat terhubung AWS Glue ke akun Salesforce Anda. Kemudian, Anda dapat menggunakan Salesforce sebagai sumber data atau tujuan dalam Pekerjaan ETL Anda. Jalankan pekerjaan ini untuk mentransfer data antara Salesforce dan AWS layanan atau aplikasi lain yang didukung.

Topik

- AWS Glue Dukungan untuk Salesforce
- Kebijakan yang berisi operasi API untuk membuat dan menggunakan koneksi
- Mengkonfigurasi Salesforce
- Mengkonfigurasi koneksi Salesforce
- Membaca dari entitas Salesforce
- Menulis ke Salesforce

- Opsi koneksi Salesforce
- Keterbatasan untuk konektor Salesforce
- Siapkan aliran OAuth pembawa JWT untuk Salesforce

AWS Glue Dukungan untuk Salesforce

AWS Glue mendukung Salesforce sebagai berikut:

Didukung sebagai sumber?

Ya. Anda dapat menggunakan pekerjaan AWS Glue ETL untuk menanyakan data dari Salesforce.

Didukung sebagai target?

Ya. Anda dapat menggunakan AWS Glue ETL untuk menulis catatan ke Salesforce.

Versi API Salesforce yang didukung

Versi API Salesforce berikut didukung

- v58.0
- v59.0
- v60.0

Kebijakan yang berisi operasi API untuk membuat dan menggunakan koneksi

Kebijakan contoh berikut menjelaskan izin AWS IAM yang diperlukan untuk membuat dan menggunakan koneksi. Jika Anda membuat peran baru, buat kebijakan yang berisi hal-hal berikut:

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:ListConnectionTypes",
                "glue:DescribeConnectionType",
                "glue:RefreshOAuth2Tokens"
                "glue:ListEntities",
                "glue:DescribeEntity"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

Anda juga dapat menggunakan kebijakan IAM berikut untuk mengizinkan akses:

- AWSGlueServiceRole— Memberikan akses ke sumber daya yang diperlukan berbagai AWS Glue proses untuk dijalankan atas nama Anda. Sumber daya ini termasuk AWS Glue, Amazon S3, IAM, CloudWatch Log, dan Amazon EC2. Jika Anda mengikuti konvensi penamaan untuk sumber daya yang ditentukan dalam kebijakan ini, AWS Glue proses memiliki izin yang diperlukan. Kebijakan ini biasanya dilampirkan pada peran yang ditentukan saat menentukan crawler, tugas, dan titik akhir pengembangan.
- AWSGlueConsoleFullAccess— Memberikan akses penuh ke AWS Glue sumber daya saat identitas yang dilampirkan kebijakan menggunakan Konsol AWS Manajemen. Jika Anda mengikuti konvensi penamaan untuk sumber daya yang ditentukan dalam kebijakan ini, maka pengguna memiliki kemampuan konsol penuh. Kebijakan ini biasanya dilampirkan ke pengguna AWS Glue konsol.

Mengkonfigurasi Salesforce

Sebelum Anda dapat menggunakan AWS Glue untuk mentransfer data ke atau dari Salesforce, Anda harus memenuhi persyaratan ini:

Persyaratan minimum

Berikut ini adalah persyaratan minimum:

- Anda memiliki akun Salesforce.
- Akun Salesforce Anda diaktifkan untuk akses API. Akses API diaktifkan secara default untuk edisi Enterprise, Unlimited, Developer, dan Performance.
- Akun Salesforce Anda memungkinkan Anda menginstal aplikasi yang terhubung. Jika Anda tidak memiliki akses ke fungsi ini, hubungi administrator Salesforce Anda. Untuk informasi selengkapnya, lihat Aplikasi Terhubung di bantuan Salesforce.

Jika Anda memenuhi persyaratan ini, Anda siap untuk terhubung AWS Glue ke akun Salesforce Anda. AWS Glue menangani persyaratan yang tersisa dengan aplikasi terhubung AWS terkelola.

Aplikasi terhubung AWS terkelola untuk Salesforce

Aplikasi tersambung AWS terkelola membantu Anda membuat koneksi Salesforce dalam langkah yang lebih sedikit. Di Salesforce, aplikasi yang terhubung adalah kerangka kerja yang mengotorisasi aplikasi eksternal, seperti AWS Glue, untuk mengakses data Salesforce Anda.

- Buat koneksi Salesforce dengan menggunakan konsol. AWS Glue
- Saat Anda mengonfigurasi koneksi, setel jenis hibah OAuth ke kode Otorisasi.

Mengkonfigurasi koneksi Salesforce

Untuk mengonfigurasi koneksi Salesforce:

1. Di AWS Secrets Manager, buat rahasia dengan detail berikut:
   a. Untuk jenis hibah JWT_TOKEN - rahasia harus berisi kunci JWT_TOKEN dengan nilainya.
   b. Untuk jenis AuthorizationCode hibah: untuk aplikasi terhubung yang dikelola pelanggan, rahasia harus berisi aplikasi yang terhubung Rahasia Konsumen dengan kunci `USER_MANAGED_CLIENT_APPLICATION_CLIENT_SECRET` sebagai. Untuk aplikasi yang terhubung AWS Terkelola, rahasia kosong atau rahasia dengan beberapa nilai sementara.
   c. Catatan: Anda harus membuat rahasia per koneksi di AWS Glue.
2. Dalam Katalog AWS Glue Data, buat koneksi dengan mengikuti langkah-langkah di bawah ini:
   a. Saat memilih jenis Koneksi, pilih Salesforce.
   b. Berikan INSTANCE_URL dari Salesforce yang ingin Anda sambungkan.
   c. Menyediakan lingkungan Salesforce.
   d. Pilih peran AWS IAM yang AWS Glue dapat mengasumsikan dan memiliki izin untuk tindakan berikut:
   e. Pilih jenis hibah OAuth2 yang ingin Anda gunakan untuk koneksi. Jenis hibah menentukan cara AWS Glue berkomunikasi dengan Salesforce untuk meminta akses ke data Anda. Pilihan Anda memengaruhi persyaratan yang harus Anda penuhi sebelum membuat koneksi. Anda dapat memilih salah satu dari jenis ini:
      - Jenis Hibah JWT_BEARER: Jenis hibah ini berfungsi dengan baik untuk skenario otomatisasi karena memungkinkan Token Web JSON (JWT) dibuat di depan dengan izin pengguna tertentu dalam instance Salesforce. Pembuat memiliki kendali atas berapa lama JWT berlaku. AWS Glue dapat menggunakan JWT untuk mendapatkan token akses yang digunakan untuk memanggil Salesforce API.

Alur ini mengharuskan pengguna telah membuat aplikasi yang terhubung dalam instance Salesforce mereka yang memungkinkan penerbitan token akses berbasis JWT untuk pengguna.

Untuk informasi tentang membuat aplikasi yang terhubung untuk aliran OAuth pembawa JWT, lihat alur pembawa JWT OAuth 2.0 untuk integrasi. server-to-server Untuk mengatur aliran pembawa JWT dengan aplikasi yang terhubung Salesforce, lihat. Siapkan aliran OAuth pembawa JWT untuk Salesforce

- Jenis Hibah AUTHORIZATION_CODE: Jenis hibah ini dianggap sebagai OAuth "berkaki tiga" karena bergantung pada pengalihan pengguna ke server otorisasi pihak ketiga untuk mengautentikasi pengguna. Ini digunakan saat membuat koneksi melalui AWS Glue konsol. Pengguna yang membuat koneksi mungkin secara default mengandalkan aplikasi yang AWS Glue terhubung (aplikasi klien AWS Glue terkelola) di mana mereka tidak perlu memberikan informasi terkait OAuth apa pun kecuali URL instance Salesforce mereka. AWS Glue Konsol akan mengarahkan pengguna ke Salesforce di mana pengguna harus masuk dan mengizinkan AWS Glue izin yang diminta untuk mengakses instance Salesforce mereka.

  Pengguna masih dapat memilih untuk membuat aplikasi terhubung mereka sendiri di Salesforce dan memberikan ID klien dan rahasia klien mereka sendiri saat membuat koneksi melalui konsol. AWS Glue Dalam skenario ini, mereka masih akan diarahkan ke Salesforce untuk masuk dan memberi wewenang AWS Glue untuk mengakses sumber daya mereka.

  Jenis hibah ini menghasilkan token penyegaran dan token akses. Token akses berumur pendek, dan dapat disegarkan secara otomatis tanpa interaksi pengguna menggunakan token penyegaran.

  Untuk informasi tentang membuat aplikasi yang terhubung untuk alur OAuth Kode Otorisasi, lihat Mengonfigurasi Aplikasi Terhubung untuk Kode Otorisasi dan Alur Kredensial.

f. Pilih `secretName` yang ingin Anda gunakan untuk koneksi ini AWS Glue untuk memasukkan token.

g. Pilih opsi jaringan jika Anda ingin menggunakan jaringan Anda. Berikan peran IAM yang terkait dengan izin AWS Glue pekerjaan Anda untuk membaca`secretName`.

3. Berikan peran IAM yang terkait dengan izin AWS Glue pekerjaan Anda untuk membaca`secretName`.

4. Dalam konfigurasi AWS Glue pekerjaan Anda, berikan `connectionName` sebagai koneksi jaringan tambahan.

```
{
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "ec2:CreateNetworkInterface",
                "ec2:DescribeNetworkInterface",
                "ec2:DeleteNetworkInterface",
            ],
            "Resource": "*"
        }
    ]
}
```

Membaca dari entitas Salesforce

Prasyarat

Salesforce SoBject yang ingin Anda baca. Anda akan membutuhkan nama objek seperti `Account` atau `Case` atau`Opportunity`.

Contoh:

```
salesforce_read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforce",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v60.0"
    }
```

Mempartisi kueri

Anda dapat memberikan opsi Spark tambahan`PARTITION_FIELD`,, `LOWER_BOUND`,`UPPER_BOUND`, dan `NUM_PARTITIONS` jika Anda ingin memanfaatkan konkurensi di Spark. Dengan parameter ini,

kueri asli akan dibagi menjadi NUM_PARTITIONS sejumlah sub-kueri yang dapat dijalankan oleh tugas Spark secara bersamaan.

- PARTITION_FIELD: nama bidang yang akan digunakan untuk mempartisi kueri.
- LOWER_BOUND: nilai batas bawah inklusif dari bidang partisi yang dipilih.

  Untuk bidang stempel waktu, kami menerima format stempel waktu Spark yang digunakan dalam kueri Spark SQL.

  Contoh nilai yang valid:

  ```
  "TIMESTAMP \"1707256978123\""
  "TIMESTAMP ’2024-02-06 22:02:58.123 UTC'"
  "TIMESTAMP \"2018-08-08 00:00:00 Pacific/Tahiti\"
  "TIMESTAMP \"2018-08-08 00:00:00\""
  "TIMESTAMP \"-123456789\" Pacific/Tahiti"
  "TIMESTAMP \"1702600882\""
  ```
- UPPER_BOUND: nilai batas atas eksklusif dari bidang partisi yang dipilih.
- NUM_PARTITIONS: jumlah partisi.

Contoh:

```
salesforce_read = glueContext.create_dynamic_frame.from_options(
    connection_type="salesforce",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v60.0",
        "PARTITION_FIELD": "SystemModstamp"
        "LOWER_BOUND": "TIMESTAMP '2021-01-01 00:00:00 Pacific/Tahiti'"
        "UPPER_BOUND": "TIMESTAMP '2023-01-10 00:00:00 Pacific/Tahiti'"
        "NUM_PARTITIONS": "10"
    }
```

Menulis ke Salesforce

Prasyarat

Salesforce SoBject yang ingin Anda tulis. Anda akan membutuhkan nama objek seperti Account atau Case atauOpportunity.

Konektor Salesforce mendukung empat operasi tulis:

- INSERT
- MENEGAKKAN
- UPDATE
- DELETE

Saat menggunakan operasi `UPSERT` tulis, `ID_FIELD_NAMES` harus disediakan untuk menentukan bidang ID eksternal untuk catatan.

Contoh

```
salesforce_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="salesforce",
    connection_options={
        "connectionName": "connectionName",
        "ENTITY_NAME": "Account",
        "API_VERSION": "v60.0",
        "WRITE_OPERATION": "INSERT"
    }
```

Opsi koneksi Salesforce

Berikut ini adalah opsi koneksi untuk Salesforce:

- `ENTITY_NAME`(String) - (Diperlukan) Digunakan untuk Baca/Tulis. Nama Objek Anda di Salesforce.
- `API_VERSION`(String) - (Diperlukan) Digunakan untuk Baca/Tulis. Salesforce Rest API versi yang ingin Anda gunakan.
- `SELECTED_FIELDS`(Daftar<String>) - Default: kosong (SELECT *). Digunakan untuk Baca. Kolom yang ingin Anda pilih untuk objek.
- `FILTER_PREDICATE`(String) - Default: kosong. Digunakan untuk Baca. Itu harus dalam format Spark SQL.
- `QUERY`(String) - Default: kosong. Digunakan untuk Baca. Kueri SQL Spark penuh.
- `PARTITION_FIELD`(String) - Digunakan untuk Baca. Bidang yang akan digunakan untuk mempartisi kueri.

- `LOWER_BOUND`(String) - Digunakan untuk Baca. Nilai batas bawah inklusif dari bidang partisi yang dipilih.
- `UPPER_BOUND`(String) - Digunakan untuk Baca. Nilai batas atas eksklusif dari bidang partisi yang dipilih.
- `NUM_PARTITIONS`(Integer) - Default: 1. Digunakan untuk Baca. Jumlah partisi untuk dibaca.
- `IMPORT_DELETED_RECORDS`(String) - Default: FALSE. Digunakan untuk membaca. Untuk mendapatkan catatan hapus saat melakukan kueri.
- `WRITE_OPERATION`(String) - Default: SISIPKAN. Digunakan untuk menulis. Nilai harus INSERT, UPDATE, UPSERT, DELETE.
- `ID_FIELD_NAMES`(String) - Default: null. Digunakan hanya untuk UPSERT.

Keterbatasan untuk konektor Salesforce

Berikut ini adalah batasan untuk konektor Salesforce:

- Kami hanya mendukung Spark SQL dan Salesforce SOQL tidak didukung.
- Bookmark Job tidak didukung.

Siapkan aliran OAuth pembawa JWT untuk Salesforce

Lihat dokumentasi publik Salesforce untuk mengaktifkan server-to-server integrasi dengan Token Web OAuth 2.0 JSON.

Membuat pasangan sertifikat/kunci file PEM

Buat pasangan sertifikat/kunci file PEM

```
openssl req -newkey rsa:4096 -new -nodes -x509 -days 3650 -keyout key.pem -out cert.pem
```

Membuat aplikasi yang terhubung Salesforce dengan JWT

1. Masuk ke Salesforce dan klik roda gigi pengaturan di kanan atas dan pilih Pengaturan.
2. Di sebelah kiri, navigasikan ke Manajer Aplikasi. (Tool Platform > Aplikasi > Manajer Aplikasi)
3. Pilih Aplikasi Koneksi Baru.
4. Berikan nama aplikasi, biarkan sisanya diisi secara otomatis.

5. Centang kotak untuk Aktifkan Pengaturan OAuth.
6. Tetapkan URL panggilan balik. Ini tidak akan digunakan untuk JWT, jadi Anda dapat menggunakan https://localhost.
7. Centang kotak untuk Gunakan Tanda Tangan Digital.
8. Unggah file cert.pem yang dibuat sebelumnya.
9. Tambahkan izin yang diperlukan:
   a. Kelola data pengguna melalui API (api).
   b. Akses izin khusus (custom_permissions).
   c. Akses layanan URL identitas (id, profil, email, alamat, telepon).
   d. Akses pengidentifikasi pengguna unik (openid).
   e. Lakukan permintaan kapan saja (refresh_token, offline_access).
10. Centang kotak untuk token akses berbasis Issue JSON Web Token (JWT) untuk pengguna bernama.
11. Pilih Simpan.
12. Pilih Lanjutkan.
13. Pilih Kelola Detail Konsumen.
14. Salin kunci konsumen (id klien).
15. Salin rahasia konsumen (rahasia klien).
16. Klik Batal.

Menghasilkan Token Web JSON (JWT)

1. Ubah key pair menjadi pkcs12 (setel kata sandi ekspor saat diminta).

```
openssl pkcs12 -export -in cert.pem -inkey key.pem -name jwtcert > jwtcert.p12
```

2. Buat keystore Java dari pkcs12 (atur kata sandi keystore tujuan saat diminta, dan berikan kata sandi ekspor sebelumnya untuk kata sandi keystore sumber).

```
keytool -importkeystore -srckeystore jwtcert.p12 -destkeystore keystore.jks -
srcstoretype pkcs12 -alias jwtcert
```

3. Konfirmasikan keystore.jks menyertakan alias jwtcert (masukkan kata sandi keystore tujuan sebelumnya saat diminta).

```
keytool -keystore keystore.jks -list
```

4. Gunakan kelas Java JwtExample yang disediakan dalam dokumentasi Salesforce untuk menghasilkan token yang ditandatangani.
   a. Edit nilai di ClaimArray seperlunya:
      - ClaimArray [0] = id klien
      - ClaimArray [1] = id pengguna salesforce
      - ClaimArray [2] = url login salesforce
      - ClaimaRray [4] = tanggal kedaluwarsa di millis sejak zaman. 3660624000000 adalah 2085-12-31.
   b. Ganti path/to/keystore dengan jalur yang benar ke keystore.jks Anda.
   c. Ganti keystorepassword dengan kata sandi keystore tujuan yang Anda masukkan
   d. Ganti privatekeypassword dengan kata sandi keystore sumber yang Anda masukkan
   e. Kompilasi kode. Kode tergantung pada Codec Apache Commons untuk pengkodean base64.

      ```
      javac -classpath ".:./commons-codec-1.16.1.jar" JWTExample.java
      ```

   f. Jalankan kode tersebut.

      ```
      java -classpath ".:commons-codec-1.16.1.jar" JWTExample
      ```

5. Setelah aplikasi yang terhubung dan JWT dibuat, masih pengguna harus diotorisasi untuk aplikasi tersebut. Lihat langkah 3 di https://mannharleen.github.io/2020-03-03-salesforce-jwt/ untuk dua pendekatan.

Dengan langkah-langkah di atas selesai, ini akan menghasilkan JSON Web Token (JWT) yang dapat digunakan untuk mendapatkan token akses dari Salesforce.

Contoh masukan:

```
export password for pkcs12: awsglue
destination keystore password for jks: awsglue
source keystore password for jks: awsglue

claimArray[0] = “client-id”;
claimArray[1] = “my@email.com”;
claimArray[2] = "https://login.salesforce.com“;
```

```
claimArray[3] = "3660624000000";

path to keystore: ./keystore.jks
keystore password: awsglue
privatekey password: awsglue
```

Contoh keluaran:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0Ij
```

Tautan yang berguna:

- https://www.base64encode.org/
- https://jwt.io/
- https://help.salesforce.com/s/articleView?id=sf.remoteaccess_oauth_jwt_flow.htm

JWTExample.java:

```
import org.apache.commons.codec.binary.Base64;
import java.io.*;
import java.security.*;
import java.text.MessageFormat;

public class JWTExample {

  public static void main(String[] args) {

    String header = "{\"alg\":\"RS256\"}";
    String claimTemplate = "'{'\"iss\": \"{0}\", \"sub\": \"{1}\", \"aud\": \"{2}\",
 \"exp\": \"{3}\"'}'";

    try {
      StringBuffer token = new StringBuffer();

      //Encode the JWT Header and add it to our string to sign
      token.append(Base64.encodeBase64URLSafeString(header.getBytes("UTF-8")));

      //Separate with a period
      token.append(".");

      //Create the JWT Claims Object
```

```
      String[] claimArray = new String[5];
      claimArray[0] = "value";
      claimArray[1] = "my@email.com";
      claimArray[2] = "https://login.salesforce.com";
      claimArray[3] = Long.toString( ( System.currentTimeMillis()/1000 ) + 300);
      MessageFormat claims;
      claims = new MessageFormat(claimTemplate);
      String payload = claims.format(claimArray);

      //Add the encoded claims object
      token.append(Base64.encodeBase64URLSafeString(payload.getBytes("UTF-8")));

      //Load the private key from a keystore
      KeyStore keystore = KeyStore.getInstance("JKS");
      keystore.load(new FileInputStream("./keystore.jks"), "awsglue".toCharArray());
      PrivateKey privateKey = (PrivateKey) keystore.getKey("jwtcert",
 "awsglue".toCharArray());

      //Sign the JWT Header + "." + JWT Claims Object
      Signature signature = Signature.getInstance("SHA256withRSA");
      signature.initSign(privateKey);
      signature.update(token.toString().getBytes("UTF-8"));
      String signedPayload = Base64.encodeBase64URLSafeString(signature.sign());

      //Separate with a period
      token.append(".");

      //Add the encoded signature
      token.append(signedPayload);

      System.out.println(token.toString());

    } catch (Exception e) {
        e.printStackTrace();
    }
  }
}
```

Menyambung ke SAP HANA di AWS Glue Studio

AWS Gluemenyediakan dukungan bawaan untuk SAP HANA. AWS Glue Studiomenyediakan antarmuka visual untuk terhubung ke SAP HANA, pekerjaan integrasi data penulis, dan menjalankannya pada runtime Spark AWS Glue Studio tanpa server.

Topik

Membuat koneksi SAP HANA

Untuk terhubung ke SAP HANA dariAWS Glue, Anda harus membuat dan menyimpan kredensyal SAP HANA Anda secara AWS Secrets Manager rahasia, kemudian mengaitkan rahasia itu dengan koneksi SAP HANA. AWS Glue Anda perlu mengkonfigurasi konektivitas jaringan antara layanan SAP HANA Anda danAWS Glue.

Prasyarat:

- Jika layanan SAP HANA Anda berada dalam VPC Amazon, konfigurasikan Amazon VPC untuk memungkinkan pekerjaan AWS Glue Anda berkomunikasi dengan layanan SAP HANA tanpa lalu lintas yang melintasi internet publik.

  Di Amazon VPC, identifikasi atau buat grup VPC, Subnet, dan Keamanan yang AWS Glue akan digunakan saat menjalankan pekerjaan. Selain itu, Anda perlu memastikan Amazon VPC dikonfigurasi untuk mengizinkan lalu lintas jaringan antara titik akhir SAP HANA Anda dan lokasi ini. Pekerjaan Anda perlu membuat koneksi TCP dengan port SAP HANA JDBC Anda. Untuk informasi selengkapnya tentang port SAP HANA, lihat dokumentasi SAP HANA. Berdasarkan tata letak jaringan Anda, ini mungkin memerlukan perubahan pada aturan grup keamanan, ACL Jaringan, Gateway NAT, dan koneksi Peering.

Untuk mengkonfigurasi koneksi ke SAP HANA:

1. DiAWS Secrets Manager, buat rahasia menggunakan kredensyal SAP HANA Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci user dengan nilai saphanaUsername.*

- *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci password dengan nilai saphanaPassword.*

2. Di AWS Glue konsol, buat koneksi dengan mengikuti langkah-langkah dithe section called "Menambahkan AWS Glue koneksi". Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk digunakan di masa mendatang. AWS Glue

   - Saat memilih jenis Koneksi, pilih SAP HANA.
   - Saat memberikan URL SAP HANA, berikan URL untuk instance Anda.

     URL SAP HANA JDBC ada dalam bentuk `jdbc:sap://`*saphanaHostname*`:`*saphanaPort*`/?`*databaseName*`=`*saphanaDBname*`,`*Paramete*

     AWS Gluememerlukan parameter URL JDBC berikut:

     - `databaseName`— Database default di SAP HANA untuk terhubung ke.
   - Saat memilih AWSSecret, berikan *secretName*.

Setelah membuat koneksi AWS Glue SAP HANA, Anda harus melakukan langkah-langkah berikut sebelum menjalankan AWS Glue pekerjaan Anda:

- Berikan peran IAM yang terkait dengan izin AWS Glue pekerjaan Anda untuk membaca *secretName*.

Membuat node sumber SAP HANA

Prasyarat yang dibutuhkan

- Koneksi AWS Glue SAP HANA, dikonfigurasi dengan AWS Secrets Manager rahasia, seperti yang dijelaskan di bagian sebelumnya,the section called "Membuat koneksi SAP HANA".
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- *Tabel SAP HANA yang ingin Anda baca, TableName, atau kueri TargetQuery.*

  Sebuah tabel dapat ditentukan dengan nama tabel SAP HANA dan nama skema, dalam bentuk. *schemaName*`.`*tableName* Nama skema dan pemisah "." tidak diperlukan jika tabel dalam skema default, "publik". Panggil *TableIdentifier* ini. Perhatikan bahwa database disediakan sebagai parameter URL JDBC di. `connectionName`

Menambahkan sumber data SAP HANA

Untuk menambahkan sumber Data — SAP HANA node:

1. Pilih koneksi untuk sumber data SAP HANA Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi SAP HANA. Untuk informasi lebih lanjut, lihat bagian sebelumnya,the section called “Membuat koneksi SAP HANA”.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.
2. Pilih opsi Sumber SAP HANA:

   - Pilih satu tabel — akses semua data dari satu tabel.
   - Masukkan kueri kustom — akses kumpulan data dari beberapa tabel berdasarkan kueri kustom Anda.
3. Jika Anda memilih satu tabel, masukkan *`TableName`*.

   Jika Anda memilih Masukkan kueri kustom, masukkan kueri SQL SELECT.
4. Di properti Custom SAP HANA, masukkan parameter dan nilai sesuai kebutuhan.

Membuat node target SAP HANA

Prasyarat yang dibutuhkan

- Koneksi AWS Glue SAP HANA, dikonfigurasi dengan AWS Secrets Manager rahasia, seperti yang dijelaskan di bagian sebelumnya,the section called “Membuat koneksi SAP HANA”.
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- Tabel SAP HANA yang ingin Anda tulis, *`TableName`*.

  Sebuah tabel dapat ditentukan dengan nama tabel SAP HANA dan nama skema, dalam bentuk. *`schemaName`*`.`*`tableName`* Nama skema dan pemisah “.” tidak diperlukan jika tabel dalam skema default, “publik”. Panggil *`TableIdentifier`* ini. Perhatikan bahwa database disediakan sebagai parameter URL JDBC di. `connectionName`

Menambahkan target data SAP HANA

Untuk menambahkan target Data — SAP HANA node:

1. Pilih koneksi untuk sumber data SAP HANA Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi SAP HANA. Untuk informasi lebih lanjut lihat bagian sebelumnya,the section called “Membuat koneksi SAP HANA”.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.
2. Konfigurasikan nama Tabel dengan menyediakan *`TableName`*.
3. Di properti Custom Teradata, masukkan parameter dan nilai sesuai kebutuhan.

Opsi lanjutan

Anda dapat memberikan opsi lanjutan saat membuat node SAP HANA. Opsi ini sama dengan yang tersedia saat pemrograman AWS Glue untuk skrip Spark.

Lihat the section called “Koneksi SAP HANA”.

Menghubungkan ke Snowflake di AWS Glue Studio

> **Note**
>
> Anda dapat menggunakan Spark AWS Glue untuk membaca dari dan menulis ke tabel di Snowflake di AWS Glue versi 4.0 dan yang lebih baru. Untuk mengonfigurasi koneksi Snowflake dengan AWS Glue pekerjaan secara terprogram, lihat. Koneksi Redshift

AWS Gluemenyediakan dukungan bawaan untuk Snowflake. AWS Glue Studiomenyediakan antarmuka visual untuk terhubung ke Snowflake, pekerjaan integrasi data penulis, dan menjalankannya pada runtime Spark tanpa AWS Glue Studio server.

Topik

- Membuat koneksi Snowflake
- Membuat simpul sumber Snowflake
- Membuat simpul target Snowflake

- Opsi lanjutan

Membuat koneksi Snowflake

Saat menambahkan sumber Data - Snowflake node diAWS Glue Studio, Anda dapat memilih koneksi AWS Glue Snowflake yang ada atau membuat koneksi baru. Anda harus memilih `SNOWFLAKE` jenis koneksi dan bukan jenis koneksi yang `JDBC` dikonfigurasi untuk terhubung ke Snowflake. Ikuti prosedur berikut untuk membuat koneksi AWS Glue Snowflake:

Untuk membuat koneksi Snowflake

1. *Di Snowflake, buat pengguna, SnowFlakeUser dan kata sandi, SnowFlakePassword.*
2. *Tentukan gudang Snowflake mana yang akan berinteraksi dengan pengguna ini, SnowFlakeWarehouse.* Entah mengaturnya sebagai `DEFAULT_WAREHOUSE` untuk *SnowFlakeUser* di Snowflake atau ingat untuk langkah berikutnya.
3. DiAWS Secrets Manager, buat rahasia menggunakan kredensi Snowflake Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

   - Saat memilih pasangan kunci/nilai, buat pasangan untuk *SnowFlakeUser* dengan kuncinya. `sfUser`
   - Saat memilih pasangan kunci/nilai, buat pasangan untuk *SnowFlakePassword* dengan kuncinya. `sfPassword`
   - Saat memilih pasangan kunci/nilai, buat pasangan untuk *SnowFlakeWarehouse* dengan kuncinya. `sfWarehouse` Ini tidak diperlukan jika default diatur di Snowflake.
4. Dalam Katalog AWS Glue Data, buat koneksi dengan mengikuti langkah-langkah dalam Menambahkan AWS Glue koneksi. Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk langkah selanjutnya.

   - Saat memilih jenis Koneksi, pilih Snowflake.
   - Saat memilih URL Snowflake, berikan nama host instance Snowflake Anda. URL akan menggunakan nama host dalam formulir*account_identifier*`.snowflakecomputing.com`.
   - Saat memilih AWSSecret, berikan *secretName*.

Membuat simpul sumber Snowflake

Izin diperlukan

AWS Glue Studiopekerjaan menggunakan sumber data Snowflake memerlukan izin tambahan. Untuk informasi selengkapnya tentang cara menambahkan izin ke pekerjaan ETL, lihat Meninjau izin IAM yang diperlukan untuk pekerjaan ETL.

`SNOWFLAKE`AWS Gluekoneksi menggunakan AWS Secrets Manager rahasia untuk memberikan informasi kredensi. Peran pratinjau pekerjaan dan data Anda AWS Glue Studio harus memiliki izin untuk membaca rahasia ini.

Menambahkan sumber data Snowflake

Prasyarat:

- AWS Secrets ManagerRahasia untuk kredensi Snowflake Anda
- Koneksi Katalog AWS Glue Data tipe Snowflake

Untuk menambahkan Sumber Data — Snowflake node:

1. Pilih koneksi untuk sumber data Snowflake Anda. Ini mengasumsikan bahwa koneksi sudah ada dan Anda dapat memilih dari koneksi yang ada. Jika Anda perlu membuat koneksi, pilih Buat koneksi Snowflake. Untuk informasi selengkapnya, lihat Ikhtisar penggunaan konektor dan koneksi.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti. Informasi tentang koneksi terlihat, termasuk URL, grup keamanan, subnet, zona ketersediaan, deskripsi, dan stempel waktu yang dibuat (UTC) dan terakhir diperbarui (UTC).
2. Pilih opsi sumber Snowflake:

   - Pilih satu tabel — ini adalah tabel yang berisi data yang ingin Anda akses dari satu tabel Snowflake.
   - Masukkan kueri khusus — memungkinkan Anda mengakses kumpulan data dari beberapa tabel Snowflake berdasarkan kueri kustom Anda.
3. Jika Anda memilih satu tabel, masukkan nama skema Snowflake.

   Atau, pilih Masukkan kueri khusus. Pilih opsi ini untuk mengakses dataset kustom dari beberapa tabel Snowflake. Saat Anda memilih opsi ini, masukkan kueri Snowflake.

4. Dalam opsi Kinerja dan keamanan (opsional),
   - Aktifkan pushdown kueri - pilih apakah Anda ingin menurunkan pekerjaan ke instance Snowflake.
5. Di properti Snowflake Kustom (opsional), masukkan parameter dan nilai sesuai kebutuhan.

Membuat simpul target Snowflake

Izin diperlukan

AWS Glue Studiopekerjaan menggunakan sumber data Snowflake memerlukan izin tambahan. Untuk informasi selengkapnya tentang cara menambahkan izin ke pekerjaan ETL, lihat Meninjau izin IAM yang diperlukan untuk pekerjaan ETL.

`SNOWFLAKE`AWS Gluekoneksi menggunakan AWS Secrets Manager rahasia untuk memberikan informasi kredensi. Peran pratinjau pekerjaan dan data Anda AWS Glue Studio harus memiliki izin untuk membaca rahasia ini.

Menambahkan target data Snowflake

Untuk membuat simpul target Snowflake:

1. Pilih tabel Snowflake yang ada sebagai target, atau masukkan nama tabel baru.
2. Bila Anda menggunakan target Data - Snowflake target node, Anda dapat memilih dari opsi berikut:
   - APPEND - Jika tabel sudah ada, dump semua data baru ke dalam tabel sebagai insert. Jika tabel tidak ada, buat dan kemudian masukkan semua data baru.
   - MERGE - AWS Glue akan memperbarui atau menambahkan data ke tabel target Anda berdasarkan kondisi yang Anda tentukan.

     Pilih opsi:

     - Pilih kunci dan tindakan sederhana — pilih kolom yang akan digunakan sebagai kunci yang cocok antara data sumber dan kumpulan data target Anda.

       Tentukan opsi berikut jika dicocokkan:

       - Perbarui catatan dalam kumpulan data target Anda dengan data dari sumber.
       - Hapus catatan dalam kumpulan data target Anda.

Tentukan opsi berikut jika tidak cocok:

- Masukkan data sumber sebagai baris baru ke dalam kumpulan data target Anda.
- Jangan lakukan apa-apa.

- Masukkan pernyataan MERGE kustom — Anda kemudian dapat memilih pernyataan Validasi Gabungan untuk memverifikasi bahwa pernyataan tersebut valid atau tidak valid.

- TRUNCATE — Jika tabel sudah ada, potong data tabel dengan terlebih dahulu membersihkan isi tabel target. Jika truncate berhasil, maka masukkan semua data. Jika tabel tidak ada, buat tabel dan masukkan semua data. Jika pemotongan tidak berhasil, operasi akan gagal.
- DROP — Jika tabel sudah ada, hapus metadata tabel dan data. Jika penghapusan berhasil, maka masukkan semua data. Jika tabel tidak ada, buat tabel dan masukkan semua data. Jika drop tidak berhasil, operasi akan gagal.

Opsi lanjutan

Lihat koneksi Snowflake di panduan AWS Glue pengembang.

Menghubungkan ke Teradata Vantage di AWS Glue Studio

AWS Gluemenyediakan dukungan bawaan untuk Teradata Vantage. AWS Glue Studiomenyediakan antarmuka visual untuk terhubung ke Teradata, membuat pekerjaan integrasi data, dan menjalankannya di runtime Spark tanpa AWS Glue Studio server.

Topik

Membuat koneksi Teradata Vantage

Untuk terhubung ke Teradata Vantage dari AWS Glue, Anda harus membuat dan menyimpan kredenal Teradata Anda secara AWS Secrets Manager rahasia, lalu mengaitkan rahasia itu dengan koneksi Teradata. AWS Glue

Prasyarat:

- Jika Anda mengakses lingkungan Teradata Anda melalui Amazon VPC, konfigurasikan Amazon VPC untuk memungkinkan pekerjaan AWS Glue Anda berkomunikasi dengan lingkungan Teradata. Kami tidak menyarankan mengakses lingkungan Teradata melalui internet publik.

  Di Amazon VPC, identifikasi atau buat grup VPC, Subnet, dan Keamanan yang AWS Glue akan digunakan saat menjalankan pekerjaan. Selain itu, Anda perlu memastikan Amazon VPC dikonfigurasi untuk mengizinkan lalu lintas jaringan antara instans Teradata Anda dan lokasi ini. Pekerjaan Anda perlu membuat koneksi TCP dengan port klien Teradata Anda. Untuk informasi selengkapnya tentang port Teradata, lihat dokumentasi Teradata.

  Berdasarkan tata letak jaringan Anda, konektivitas VPC yang aman mungkin memerlukan perubahan di Amazon VPC dan layanan jaringan lainnya. Untuk informasi lebih lanjut tentang AWS konektivitas, lihat Opsi AWS Konektivitas di dokumentasi Teradata.

Untuk mengkonfigurasi koneksi AWS Glue Teradata:

1. *Dalam konfigurasi Teradata Anda, identifikasi atau buat pengguna dan kata sandi AWS Glue akan terhubung dengan, TeradaTauser dan TeradataPassWord.* Untuk informasi lebih lanjut, lihat Ikhtisar Keamanan Vantage di dokumentasi Teradata.
2. Di AWS Secrets Manager, buat rahasia menggunakan kredenal Teradata Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci user dengan nilai TeradataUsername.*
   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci password dengan nilai TeradataPassword.*
3. Di AWS Glue konsol, buat koneksi dengan mengikuti langkah-langkah dithe section called “Menambahkan AWS Glue koneksi”. Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk langkah selanjutnya.

   - Saat memilih jenis Koneksi, pilih Teradata.
   - Saat memberikan URL JDBC, berikan URL untuk instance Anda. Anda juga dapat membuat hardcode parameter koneksi yang dipisahkan koma

tertentu di URL JDBC Anda. URL harus sesuai dengan format berikut:
`jdbc:teradata://teradataHostname/ParameterName=ParameterValue,ParameterName`

Parameter URL yang didukung meliputi:

- `DATABASE`— nama database pada host untuk mengakses secara default.
- `DBS_PORT`— port database, digunakan saat berjalan pada port yang tidak standar.

- Saat memilih jenis Credential, pilih AWS Secrets Manager, lalu atur AWS Secret ke *`secretName`*.

4. Dalam situasi berikut, Anda mungkin memerlukan konfigurasi tambahan:

   - Untuk instans Teradata yang dihosting di VPC AWS Amazon
     - Anda harus memberikan informasi koneksi Amazon VPC ke AWS Glue koneksi yang menentukan kredenal keamanan Teradata Anda. Saat membuat atau memperbarui koneksi Anda, atur grup VPC, Subnet, dan Keamanan dalam opsi Jaringan.

Membuat node sumber Teradata

Prasyarat

- Koneksi AWS Glue Teradata Vantage, dikonfigurasi dengan AWS Secrets Manager rahasia, seperti yang dijelaskan di bagian sebelumnya,. the section called “Membuat koneksi Teradata Vantage”
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- *`Tabel Teradata yang ingin Anda baca, TableName, atau kueri TargetQuery.`*

Menambahkan sumber data Teradata

Untuk menambahkan sumber Data — Teradata node:

1. Pilih koneksi untuk sumber data Teradata Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Untuk membuat koneksi, pilih Buat koneksi baru. Untuk informasi lebih lanjut, lihat bagian sebelumnyathe section called “Membuat koneksi Teradata Vantage”.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.

2. Pilih opsi Sumber Teradata:

- Pilih satu tabel — akses semua data dari satu tabel.
- Masukkan kueri kustom — akses kumpulan data dari beberapa tabel berdasarkan kueri kustom Anda.

3. Jika Anda memilih satu tabel, masukkan *TableName*.

   Jika Anda memilih Masukkan kueri kustom, masukkan kueri SQL SELECT.

4. Di properti Custom Teradata, masukkan parameter dan nilai sesuai kebutuhan.

Membuat node target Teradata

Prasyarat yang dibutuhkan

- Koneksi AWS Glue Teradata Vantage, dikonfigurasi dengan AWS Secrets Manager rahasia, seperti yang dijelaskan di bagian sebelumnya,. the section called “Membuat koneksi Teradata Vantage”
- Izin yang sesuai pada pekerjaan Anda untuk membaca rahasia yang digunakan oleh koneksi.
- *Tabel Teradata yang ingin Anda tulis, TableName.*

Menambahkan target data Teradata

Untuk menambahkan target Data — Teradata node:

1. Pilih koneksi untuk sumber data Teradata Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi Teradata. Untuk informasi selengkapnya, lihat Gambaran umum menggunakan konektor dan koneksi.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.

2. Konfigurasikan nama Tabel dengan menyediakan *TableName*.
3. Di properti Custom Teradata, masukkan parameter dan nilai sesuai kebutuhan.

Opsi lanjutan

Anda dapat memberikan opsi lanjutan saat membuat simpul Teradata. Opsi ini sama dengan yang tersedia saat pemrograman AWS Glue untuk skrip Spark.

Lihat the section called “Koneksi Teradata Vantage”.

Menghubungkan ke Vertica di AWS Glue Studio

AWS Gluemenyediakan dukungan bawaan untuk Vertica. AWS Glue Studiomenyediakan antarmuka visual untuk terhubung ke Vertica, membuat pekerjaan integrasi data, dan menjalankannya di runtime Spark AWS Glue Studio tanpa server.

Topik

- Membuat koneksi Vertica
- Membuat simpul sumber Vertica
- Membuat simpul target Vertica
- Opsi lanjutan

Membuat koneksi Vertica

Prasyarat:

- *Bucket atau folder Amazon S3 yang akan digunakan untuk penyimpanan sementara saat membaca dari dan menulis ke database, dirujuk oleh Temps3Path.*

> **Note**
>
> *Saat menggunakan Vertica dalam pratinjau data AWS Glue pekerjaan, file sementara mungkin tidak dihapus secara otomatis dari Temps3Path.* Untuk memastikan penghapusan file sementara, langsung akhiri sesi pratinjau data dengan memilih Akhiri sesi di panel pratinjau data.
> Jika Anda tidak dapat menjamin sesi pratinjau data berakhir secara langsung, pertimbangkan untuk menyetel konfigurasi Siklus Hidup Amazon S3 untuk menghapus data lama. Sebaiknya hapus data yang lebih lama dari 49 jam, berdasarkan runtime pekerjaan maksimum ditambah margin. Untuk informasi selengkapnya tentang mengonfigurasi Siklus Hidup Amazon S3, lihat Mengelola siklus hidup penyimpanan di dokumentasi Amazon S3.

- Kebijakan IAM dengan izin yang sesuai ke jalur Amazon S3 yang dapat Anda kaitkan dengan AWS Glue peran pekerjaan Anda.

- Jika instans Vertica Anda ada di VPC Amazon, konfigurasikan Amazon VPC untuk memungkinkan pekerjaan AWS Glue Anda berkomunikasi dengan instans Vertica tanpa lalu lintas melintasi internet publik.

  Di Amazon VPC, identifikasi atau buat grup VPC, Subnet, dan Keamanan yang AWS Glue akan digunakan saat menjalankan pekerjaan. Selain itu, Anda perlu memastikan Amazon VPC dikonfigurasi untuk mengizinkan lalu lintas jaringan antara instans Vertica Anda dan lokasi ini. Pekerjaan Anda perlu membuat koneksi TCP dengan port klien Vertica Anda, (default 5433). Berdasarkan tata letak jaringan Anda, ini mungkin memerlukan perubahan pada aturan grup keamanan, ACL Jaringan, Gateway NAT, dan koneksi Peering.

Untuk mengonfigurasi koneksi ke Vertica:

1. *DiAWS Secrets Manager, buat rahasia menggunakan kredensyal Vertica Anda, VerticaUsername dan VerticaPassWord.* Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci user dengan nilai VerticaUsername.*
   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci password dengan nilai VerticaPassWord.*

2. Di AWS Glue konsol, buat koneksi dengan mengikuti langkah-langkah dithe section called “Menambahkan AWS Glue koneksi”. Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk langkah selanjutnya.

   - Saat memilih jenis Koneksi, pilih Vertica.
   - Saat memilih Vertica Host, berikan nama host instalasi Vertica Anda.
   - Saat memilih Port Vertica, port instalasi Vertica Anda tersedia.
   - Saat memilih AWSSecret, berikan *secretName*.

3. Dalam situasi berikut, Anda mungkin memerlukan konfigurasi tambahan:

   - Untuk instans Vertica yang dihosting AWS di VPC Amazon

- Berikan informasi koneksi Amazon VPC ke AWS Glue koneksi yang menentukan kredensyal keamanan Vertica Anda. Saat membuat atau memperbarui koneksi Anda, atur grup VPC, Subnet, dan Keamanan dalam opsi Jaringan.

Anda perlu melakukan langkah-langkah berikut sebelum menjalankan AWS Glue pekerjaan Anda:

- *Berikan peran IAM yang terkait dengan izin AWS Glue pekerjaan Anda ke Temps3Path.*
- Berikan peran IAM yang terkait dengan izin AWS Glue pekerjaan Anda untuk membaca *secretName*.

Membuat simpul sumber Vertica

Prasyarat yang dibutuhkan

- Koneksi Katalog AWS Glue Data tipe Vertica, *ConnectionName*, dan lokasi Amazon S3 sementara, *Temps3Path, seperti yang dijelaskan di bagian sebelumnya*,. the section called “Membuat koneksi Vertica”
- *Tabel Vertica yang ingin Anda baca, TableName, atau kueri TargetQuery.*

Menambahkan sumber data Vertica

Untuk menambahkan sumber Data — simpul Vertica:

1. Pilih koneksi untuk sumber data Vertica Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi Vertica. Untuk informasi lebih lanjut, lihat bagian sebelumnya,the section called “Membuat koneksi Vertica”.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.
2. Pilih Database yang berisi tabel Anda.
3. *Pilih **area Staging di Amazon S3, masukkan URI** S3A ke Temps3Path.*
4. Pilih Sumber Vertica.

   - Pilih satu tabel — akses semua data dari satu tabel.

- Masukkan kueri kustom — akses kumpulan data dari beberapa tabel berdasarkan kueri kustom Anda.

5. Jika Anda memilih satu tabel, masukkan ***TableName*** dan pilih Skema secara opsional.

   Jika Anda memilih Masukkan kueri kustom, masukkan kueri SQL SELECT dan pilih Skema secara opsional.

6. Di properti Custom Vertica, masukkan parameter dan nilai sesuai kebutuhan.

Membuat simpul target Vertica

Prasyarat yang dibutuhkan

- Koneksi Katalog AWS Glue Data tipe Vertica, *ConnectionName*, dan lokasi Amazon S3 sementara, *Temps3Path, seperti yang dijelaskan di bagian sebelumnya*,. the section called “Membuat koneksi Vertica”

Menambahkan target data Vertica

Untuk menambahkan target Data — simpul Vertica:

1. Pilih koneksi untuk sumber data Vertica Anda. Karena Anda telah membuatnya, itu harus tersedia di dropdown. Jika Anda perlu membuat koneksi, pilih Buat koneksi Vertica. Untuk informasi lebih lanjut, lihat bagian sebelumnya,the section called “Membuat koneksi Vertica”.

   Setelah Anda memilih koneksi, Anda dapat melihat properti koneksi dengan mengklik Lihat properti.

2. Pilih Database yang berisi tabel Anda.
3. *Pilih **area Staging di Amazon S3, masukkan URI** S3A ke Temps3Path.*
4. Masukkan ***TableName*** dan pilih Skema secara opsional.
5. Di properti Custom Vertica, masukkan parameter dan nilai sesuai kebutuhan.

Opsi lanjutan

Anda dapat memberikan opsi lanjutan saat membuat simpul Vertica. Opsi ini sama dengan yang tersedia saat memprogram AWS Glue skrip Spark.

Lihat the section called “Koneksi Vertica”.

Menggunakan konektor dan koneksi dengan AWS Glue Studio

AWS Glue menyediakan dukungan bawaan untuk penyimpanan data yang paling umum digunakan (seperti Amazon Redshift, Amazon Aurora, Microsoft SQL Server, MySQL, MongoDB, dan PostgreSQL) dengan menggunakan koneksi JDBC. AWS Glue juga memungkinkan Anda untuk menggunakan driver JDBC kustom dalam tugas extract, transform, and load (ETL) Anda. Untuk penyimpanan data yang tidak didukung secara asli, seperti aplikasi SaaS, Anda dapat menggunakan konektor.

Konektor adalah paket kode opsional yang membantu mengakses penyimpanan data di. AWS Glue Studio Anda dapat berlangganan beberapa konektor yang ditawarkan di AWS Marketplace.

Saat membuat pekerjaan ETL, Anda dapat menggunakan penyimpanan data yang didukung secara native, konektor dari AWS Marketplace, atau konektor kustom Anda sendiri. Jika Anda menggunakan sebuah konektor, maka Anda harus membuat sebuah koneksi untuk konektor tersebut terlebih dahulu. Sebuah koneksi berisi properti yang diperlukan untuk connect ke penyimpanan data tertentu. Anda menggunakan koneksi tersebut dengan sumber data Anda dan target data dalam tugas ETL. Konektor dan koneksi bekerja sama untuk memfasilitasi akses ke penyimpanan data.

Topik

Gambaran umum menggunakan konektor dan koneksi

Sebuah koneksi berisi properti yang diperlukan untuk connect ke penyimpanan data tertentu. Ketika Anda membuat sebuah koneksi, ia disimpan dalam AWS Glue Data Catalog. Anda memilih sebuah konektor, dan kemudian membuat sebuah koneksi berdasarkan konektor tersebut.

Anda dapat berlangganan konektor untuk penyimpanan data yang tidak didukung secara asli AWS Marketplace, dan kemudian menggunakan konektor tersebut saat Anda membuat koneksi. Para

developer juga dapat membuat konektor mereka sendiri, dan Anda dapat menggunakannya saat membuat koneksi.

> **Note**
>
> Koneksi yang dibuat menggunakan kustom atau AWS Marketplace konektor AWS Glue Studio muncul di AWS Glue konsol dengan jenis yang disetel ke`UNKNOWN`.

Langkah-langkah berikut menjelaskan keseluruhan proses penggunaan konektor diAWS Glue Studio:

1. Berlangganan konektor di AWS Marketplace, atau kembangkan konektor Anda sendiri dan unggah keAWS Glue Studio. Untuk informasi selengkapnya, lihat Menambahkan konektor ke AWS Glue Studio.
2. Tinjau informasi penggunaan konektor. Anda dapat menemukan informasi ini di tab Penggunaan pada halaman produk konektor. Misalnya, jika Anda mengklik tab Penggunaan di halaman produk ini, AWS GlueKonektor untuk Google BigQuery, Anda dapat melihat di bagian Sumber Daya Tambahan tautan ke blog tentang penggunaan konektor ini. Konektor lain mungkin berisi tautan ke petunjuk di bagian Gambaran Umum, seperti yang ditunjukkan pada halaman produk konektor untuk Konektor Cloudwatch Logs untuk AWS Glue.
3. Buat sebuah koneksi. Anda memilih konektor mana yang akan digunakan dan memberikan informasi tambahan untuk koneksi, seperti informasi kredensial login, string URI, dan virtual private cloud (VPC). Untuk informasi selengkapnya, lihat Membuat koneksi untuk konektor.
4. Buat IAM role untuk tugas Anda. Tugas mengambil izin dari IAM role yang Anda tentukan saat Anda membuatnya. IAM role ini harus memiliki izin yang diperlukan untuk melakukan autentikasi dengan, mengekstrak data dari, dan menulis data ke penyimpanan data Anda.
5. Buat sebuah tugas ETL dan konfigurasi properti sumber data untuk tugas ETL Anda. Sediakan opsi koneksi dan informasi autentikasi seperti yang diperintahkan oleh penyedia konektor kustom. Untuk informasi selengkapnya, lihat Menulis tugas dengan konektor kustom.
6. Kustom tugas ETL Anda dengan menambahkan transformasi atau penyimpanan data tambahan, seperti yang dijelaskan dalam Visual ETL dengan AWS Glue Studio.
7. Jika Anda menggunakan sebuah konektor untuk target data, konfigurasi properti target data untuk tugas ETL Anda. Sediakan opsi koneksi dan informasi autentikasi seperti yang diperintahkan oleh penyedia konektor kustom. Untuk informasi selengkapnya, lihat the section called “Menulis tugas dengan konektor kustom”.

8. Sesuaikan lingkungan eksekusi tugas dengan mengkonfigurasi properti tugas, seperti yang dijelaskan dalam Mengubah properti tugas.
9. Jalankan tugas.

Menambahkan konektor ke AWS Glue Studio

Sebuah konektor adalah sepotong kode yang memudahkan komunikasi antara penyimpanan data anda dan AWS Glue. Anda dapat berlangganan konektor yang ditawarkan di AWS Marketplace, atau Anda dapat membuat konektor khusus Anda sendiri.

Topik

- Berlangganan konektor AWS Marketplace
- Membuat konektor kustom

Berlangganan konektor AWS Marketplace

AWS Glue Studiomembuatnya mudah untuk menambahkan konektor dari AWS Marketplace.

Untuk menambahkan konektor dari AWS Marketplace ke AWS Glue Studio

1. Di AWS Glue Studio konsol, pilih Konektor di panel navigasi konsol.
2. Pada halaman Konektor, pilih Buka AWS Marketplace.
3. Di AWS Marketplace, di Produk unggulan, pilih konektor yang ingin Anda gunakan. Anda dapat memilih salah satu konektor unggulan, atau menggunakan kolom pencarian. Anda dapat mencari berdasarkan nama atau jenis konektor, dan Anda dapat menggunakan opsi untuk menyempurnakan hasil pencarian.

   Jika Anda ingin menggunakan salah satu konektor unggulan, pilih Tampilkan Produk. Jika Anda menggunakan kolom pencarian untuk menemukan konektor, pilih nama konektornya.
4. Pada halaman produk untuk konektor, gunakan tab untuk melihat informasi tentang konektor tersebut. Jika Anda memutuskan untuk membeli konektor ini, pilih Lanjutkan ke Berlangganan.
5. Berikan informasi pembayaran, lalu pilih Lanjutkan ke Konfigurasi.
6. Pada halaman Konfigurasi perangkat lunak ini, pilih metode deployment dan versi konektor yang akan digunakan. Pilih Lanjutkan ke Peluncuran.

7. Pada halaman Luncurkan perangkat lunak ini, Anda dapat meninjau Petunjuk Penggunaan yang disediakan oleh penyedia konektor. Saat Anda siap untuk melanjutkan, pilih Aktifkan koneksi di AWS Glue Studio.

   Setelah beberapa saat, konsol menampilkan halaman Buat koneksi marketplace diAWS Glue Studio.

8. Buat sebuah koneksi yang menggunakan konektor ini, seperti yang diterangkan dalam Membuat koneksi untuk konektor.

   Atau, Anda dapat memilih Aktifkan konektor saja untuk melewati pembuatan koneksi pada saat ini. Anda harus membuat sebuah koneksi di kemudian hari sebelum Anda dapat menggunakan konektor.

Membuat konektor kustom

Anda juga dapat membuat konektor Anda sendiri dan kemudian mengunggah kode konektor keAWS Glue Studio.

Konektor khusus diintegrasikan ke dalam AWS Glue Studio melalui API runtime AWS Glue Spark. Waktu aktif Spark AWS Glue memungkinkan Anda untuk mencolokkan konektor yang sesuai dengan antarmuka Spark, Athena, atau JDBC. Ini memungkinkan Anda untuk memberikan konektor kustom dalam setiap opsi koneksi yang tersedia.

Anda dapat merangkum semua properti koneksi Anda dengan Koneksi AWS Glue dan menyediakan nama koneksi untuk tugas ETL Anda. Integrasi dengan koneksi Katalog Data memungkinkan Anda untuk menggunakan properti koneksi yang sama di beberapa panggilan dalam satu aplikasi Spark tunggal atau di aplikasi yang berbeda.

Anda dapat menentukan opsi tambahan untuk koneksi tersebut. Skrip pekerjaan yang AWS Glue Studio dihasilkan berisi `Datasource` entri yang menggunakan koneksi untuk mencolokkan konektor Anda dengan opsi koneksi yang ditentukan. Misalnya:

```
Datasource = glueContext.create_dynamic_frame.from_options(connection_type =
"custom.jdbc", connection_options = {"dbTable":"Account","connectionName":"my-custom-
jdbc-
connection"}, transformation_ctx = "DataSource0")
```

Untuk menambahkan konektor khusus ke AWS Glue Studio

1. Buat kode untuk konektor kustom Anda. Untuk informasi selengkapnya, lihat Mengembangkan konektor kustom.
2. Tambahkan support untuk fitur AWS Glue ke konektor Anda. Berikut adalah beberapa contoh fitur ini dan bagaimana mereka digunakan dalam skrip pekerjaan yang dihasilkan olehAWS Glue Studio:

   - Pemetaan tipe data — Konektor Anda dapat melakukan typecasting pada kolom saat membacanya dari penyimpanan data yang mendasari. Misalnya, sebuah `dataTypeMapping` dari `{"INTEGER":"STRING"}` mengkonversi semua kolom tipe `Integer` ke kolom tipe `String` ketika mengurai catatan dan membangun `DynamicFrame`. Hal ini membantu pengguna untuk mengubah kolom ke jenis pilihan mereka.

     ```
     DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type
     = "custom.jdbc", connection_options = {"dataTypeMapping":{"INTEGER":"STRING"}",
     connectionName":"test-connection-jdbc"}, transformation_ctx = "DataSource0")
     ```

   - Pemartisian untuk pembacaan paralel — AWS Glue memungkinkan data paralel membaca dari penyimpanan data dengan melakukan partisi data pada kolom. Anda harus menentukan kolom partisi, batas partisi bawah, batas partisi atas, dan jumlah partisi. Fitur ini memungkinkan Anda untuk menggunakan paralelisme data dan beberapa pelaksana Spark yang dialokasikan untuk aplikasi Spark.

     ```
     DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type
     = "custom.jdbc", connection_options = {"upperBound":"200","numPartitions":"4",
     "partitionColumn":"id","lowerBound":"0","connectionName":"test-connection-jdbc"},
     transformation_ctx = "DataSource0")
     ```

   - Gunakan AWS Secrets Manager untuk menyimpan kredensial-Koneksi Katalog Data juga dapat berisi `secretId` untuk rahasia yang disimpan di. AWS Secrets Manager AWS Rahasianya dapat menyimpan informasi otentikasi dan kredensil dengan aman dan menyediakannya saat runtime. AWS Glue Atau, Anda dapat menentukan `secretId` dari skrip Spark sebagai berikut:

     ```
     DataSource = glueContext.create_dynamic_frame.from_options(connection_type
     = "custom.jdbc", connection_options = {"connectionName":"test-connection-jdbc",
      "secretId"-> "my-secret-id"}, transformation_ctx = "DataSource0")
     ```

- Menyaring data sumber dengan predikat baris dan proyeksi kolom — Waktu aktif Spark AWS Glue juga memungkinkan pengguna untuk mendorong kueri SQL untuk mem-filter data pada sumber dengan predikat baris dan proyeksi kolom. Hal ini memungkinkan tugas ETL Anda untuk memuat data yang telah difilter lebih cepat dari penyimpanan data yang didorong oleh support tersebut. Contoh kueri SQL yang didorong ke sumber data JDBC adalah: `SELECT id, name, department FROM department WHERE id < 200.`

```
DataSource = glueContext.create_dynamic_frame.from_options(connection_type =
"custom.jdbc", connection_options = {"query":"SELECT id, name, department FROM
 department
WHERE id < 200","connectionName":"test-connection-jdbc"}, transformation_ctx =
"DataSource0")
```

- Bookmark tugas — AWS Glue mendukung pemuatan tambahan data dari sumber JDBC. AWS Glue melacak catatan yang diproses terakhir dari penyimpanan data, dan memproses catatan data baru dalam eksekusi tugas ETL berikutnya. Bookmark tugas menggunakan kunci primer sebagai kolom default untuk kunci bookmark, dengan ketentuan bahwa kolom ini bertambah atau berkurang secara berurutan. Untuk informasi selengkapnya tentang bookmark tugas, lihat Bookmark tugas di Panduan Developer AWS Glue .

```
DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type =
"custom.jdbc", connection_options = {"jobBookmarkKeys":["empno"],
 "jobBookmarkKeysSortOrder"
:"asc", "connectionName":"test-connection-jdbc"}, transformation_ctx =
 "DataSource0")
```

3. Kemas konektor kustom sebagai file JAR dan unggah file ke Amazon S3.
4. Uji konektor kustom Anda. Untuk informasi selengkapnya, lihat petunjuk GitHub di Glue Custom Connectors: Local Validation Tests Guide.
5. Di AWS Glue Studio konsol, pilih Konektor di panel navigasi konsol.
6. Pada halaman Konektor, pilih Buat konektor kustom.
7. Pada halaman Buat konektor kustom, masukkan informasi berikut:

   - Path ke lokasi dari file JAR kode kustom di Amazon S3.
   - Nama untuk konektor yang akan digunakan olehAWS Glue Studio.
   - Jenis konektor Anda, yang dapat berupa JDBC, Spark, atau Athena.
   - Nama titik masuk dalam kode kustom Anda yang AWS Glue Studio memanggil untuk menggunakan konektor.

    - Untuk konektor JDBC, bidang ini harus berupa nama kelas dari driver JDBC Anda.
    - Untuk konektor Spark, bidang ini harus berupa nama kelas sumber data yang memenuhi syarat, atau aliasnya, yang Anda gunakan saat memuat sumber data Spark dengan operator `format`.
  - (JDBC saja) URL dasar yang digunakan oleh koneksi JDBC untuk penyimpanan data.
  - (Opsional) Deskripsi untuk konektor kustom.
8. Pilih Buat konektor.
9. Dari halaman Konektor, buat sebuah koneksi yang menggunakan konektor ini, seperti yang dijelaskan di Membuat koneksi untuk konektor.

Koneksi yang tersedia

Koneksi berikut tersedia saat membuat koneksi untuk konektor:

- Amazon Aurora— mesin database relasional berkinerja tinggi yang dapat diskalakan dengan keamanan bawaan, pencadangan dan pemulihan, dan akselerasi dalam memori.
- Amazon DocumentDB — layanan database dokumen yang dapat diskalakan, sangat tersedia, dan dikelola sepenuhnya yang mendukung MongoDB dan SQL API.
- Amazon Redshift— layanan database dokumen yang dapat diskalakan, sangat tersedia, dan dikelola sepenuhnya yang mendukung MongoDB dan SQL API.
- Azure SQL — layanan database relasional berbasis cloud dari Microsoft Azure yang menyediakan kemampuan penyimpanan dan manajemen data yang terukur, andal, dan aman.
- Cosmos DB — layanan database cloud terdistribusi secara global dari Microsoft Azure yang menyediakan kemampuan penyimpanan dan kueri data berkinerja tinggi yang dapat diskalakan.
- Google BigQuery — gudang data cloud tanpa server untuk menjalankan kueri SQL cepat pada kumpulan data besar.
- JDBC — sistem manajemen basis data relasional (RDBMS) yang menggunakan Java API untuk menghubungkan dan berinteraksi dengan koneksi data.
- Kafka — platform pemrosesan aliran sumber terbuka yang digunakan untuk streaming dan pengiriman pesan data waktu nyata.
- MariaDB — fork MySQL yang dikembangkan komunitas yang menawarkan peningkatan kinerja, skalabilitas, dan fitur.
- MongoDB — database berorientasi dokumen lintas platform yang menyediakan skalabilitas, fleksibilitas, dan kinerja tinggi.

- MongoDB Atlas — database berbasis cloud sebagai layanan (dBaaS) yang menawarkan dari MongoDB yang menyederhanakan pengelolaan dan penskalaan penerapan MongoDB.
- Microsoft SQL Server — sistem manajemen basis data relasional (RDBMS) dari Microsoft yang menyediakan kemampuan penyimpanan, analisis, dan pelaporan data yang kuat.
- MySQL — sistem manajemen basis data relasional sumber terbuka (RDBMS) yang banyak digunakan dalam aplikasi web dan dikenal karena keandalan dan skalabilitasnya.
- Jaringan — sumber data jaringan mewakili sumber daya atau layanan yang dapat diakses jaringan yang dapat diakses oleh platform integrasi data.
- OpenSearchSumber OpenSearch data adalah aplikasi yang OpenSearch dapat terhubung ke dan menelan data dari.
- Oracle — sistem manajemen basis data relasional (RDBMS) dari Oracle Corporation yang menyediakan kemampuan penyimpanan, analisis, dan pelaporan data yang kuat.
- PostgreSQL — sistem manajemen basis data relasional sumber terbuka (RDBMS) yang menyediakan kemampuan penyimpanan, analisis, dan pelaporan data yang kuat.
- Salesforce menyediakan perangkat lunak manajemen hubungan pelanggan (CRM) yang membantu Anda dengan penjualan, layanan pelanggan, e-commerce, dan banyak lagi. Jika Anda pengguna Salesforce, Anda dapat terhubung AWS Glue ke akun Salesforce Anda. Kemudian, Anda dapat menggunakan Salesforce sebagai sumber data atau tujuan dalam pekerjaan ETL Anda. Jalankan pekerjaan ini untuk mentransfer data antara Salesforce dan AWS layanan atau aplikasi lain yang didukung.
- SAP HANA — database dalam memori dan platform analitik yang menyediakan pemrosesan data cepat, analitik canggih, dan integrasi data real-time.
- Snowflake — gudang data berbasis cloud yang menyediakan penyimpanan data dan layanan analitik yang dapat diskalakan dan berkinerja tinggi.
- Teradata — sistem manajemen basis data relasional (RDBMS) yang menyediakan kemampuan penyimpanan, analisis, dan pelaporan data berkinerja tinggi.
- Vertica — gudang data analitik berorientasi kolumnar yang dirancang untuk analitik data besar yang menawarkan kinerja kueri cepat, analitik canggih, dan skalabilitas.

Membuat koneksi untuk konektor

AWS GlueKoneksi adalah Data Catalog objek yang menyimpan informasi koneksi untuk penyimpanan data tertentu. Koneksi menyimpan kredensial login, string URI, informasi virtual private

cloud (VPC), dan informasi lainnya. Membuat koneksi di Data Catalog menghemat upaya harus menentukan semua detail koneksi setiap kali Anda membuat pekerjaan.

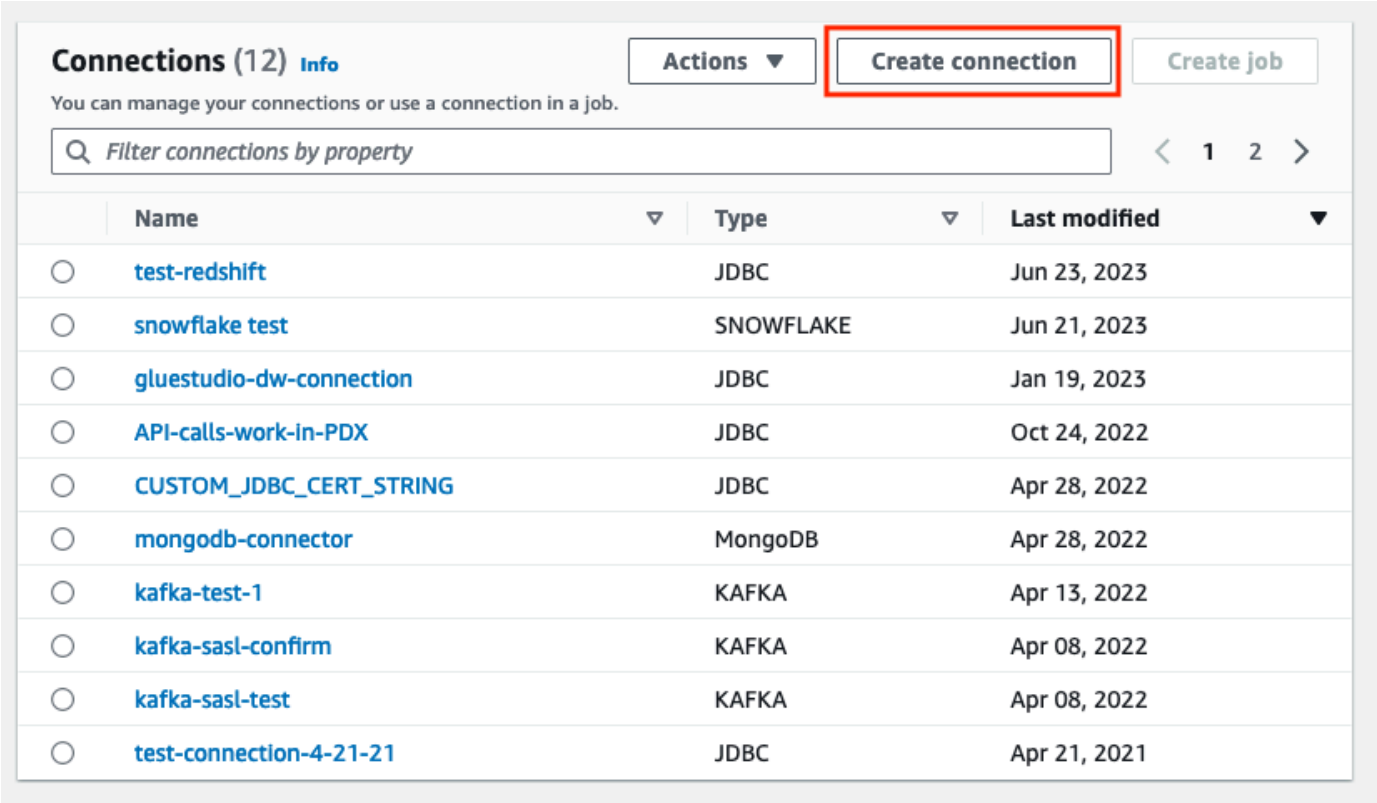
Untuk membuat sebuah koneksi untuk sebuah konektor

1. Di AWS Glue Studio konsol, pilih Konektor di panel navigasi konsol. Di bagian Koneksi, pilih Buat koneksi.
2. Pilih sumber data yang ingin Anda buat koneksi di langkah 1 panduan Buat koneksi data. Ada beberapa cara untuk melihat sumber data yang tersedia, termasuk:

   - Filter sumber data yang tersedia dengan memilih tab. Secara default, Semua konektor dipilih.
   - Alihkan Daftar untuk melihat sumber data sebagai daftar atau beralih kembali ke Grid untuk melihat konektor yang tersedia dalam tata letak grid.
   - Gunakan bilah pencarian untuk mempersempit daftar sumber data. Saat Anda mengetik, kecocokan pencarian ditampilkan dan sumber yang tidak cocok dihapus dari tampilan.

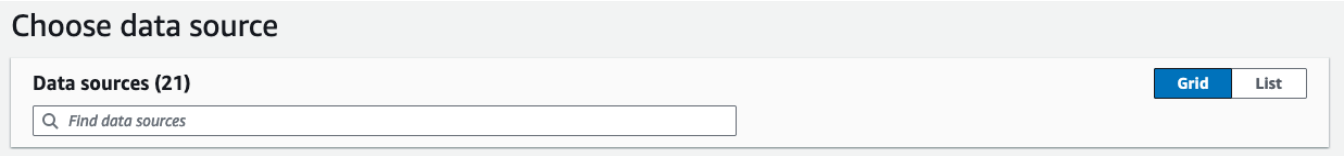
   Setelah Anda memilih sumber data, pilih Berikutnya.
3. Konfigurasikan koneksi di Langkah 2 di wizard.

   Masukkan detail koneksi. Tergantung pada jenis konektor yang dipilih, Anda akan diminta untuk memasukkan informasi tambahan:



4. Pilih sumber data yang ingin Anda buat koneksi di langkah 1 panduan Buat koneksi data. Ada beberapa cara untuk melihat sumber data yang tersedia. Secara default, Anda akan melihat semua sumber data yang tersedia dalam tata letak kisi. Anda juga dapat:

   - Alihkan Daftar untuk melihat sumber data sebagai daftar atau beralih kembali ke Grid untuk melihat konektor yang tersedia dalam tata letak grid.
   - Gunakan bilah pencarian untuk mempersempit daftar sumber data. Saat Anda mengetik, kecocokan pencarian ditampilkan dan sumber yang tidak cocok dihapus dari tampilan.

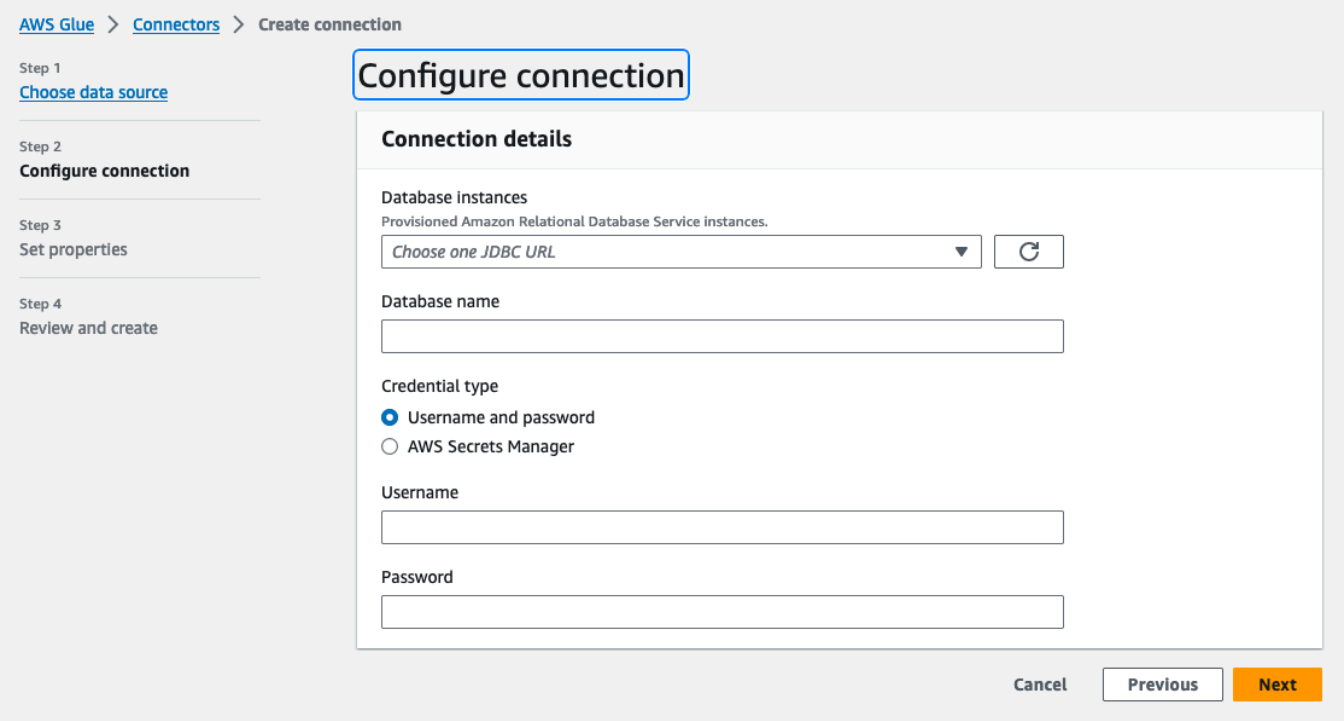
   

   Setelah Anda memilih sumber data, pilih Berikutnya.

5. Konfigurasikan koneksi di Langkah 2 di wizard.

   Masukkan detail koneksi. Tergantung pada jenis konektor yang Anda pilih, Anda mungkin diminta untuk memasukkan informasi koneksi tambahan. Ini dapat mencakup:

- Detail koneksi — bidang ini akan berubah tergantung pada sumber data yang Anda sambungkan. Misalnya, jika Anda terhubung ke database Amazon DocumentDB, Anda akan memasukkan URL Amazon DocumentDB. Jika Anda terhubung keAmazon Aurora, Anda akan memilih instance database dan memasukkan nama database. Berikut ini adalah detail Koneksi yang diperlukan untukAmazon Aurora:



- Jenis kredensi - pilih antara Nama Pengguna dan kata sandi atau AWS Secrets Manager. Masukkan informasi otentikasi yang diminta.
- Untuk konektor yang menggunakan JDBC, masukkan informasi yang diperlukan untuk membuat URL JDBC untuk penyimpanan data.
- Jika Anda menggunakan sebuah Virtual Private Cloud (VPC), masukkan informasi jaringan untuk VPC Anda.

6. Atur properti koneksi di langkah 3 wizard. Anda dapat menambahkan deskripsi dan tag sebagai bagian opsional dari langkah ini. Nama diperlukan dan diisi sebelumnya dengan nilai default. Pilih Selanjutnya.
7. Tinjau sumber koneksi, detail, dan properti. Jika Anda perlu membuat perubahan, pilih Edit untuk langkah di wizard. Saat siap, pilih, Buat koneksi.

   Pilih Buat koneksi.

Anda akan melihat halaman Konektor, dan banner informasi menunjukkan koneksi yang dibuat. Anda sekarang dapat menggunakan koneksi dalam AWS Glue Studio pekerjaan Anda.

Membuat koneksi Kafka

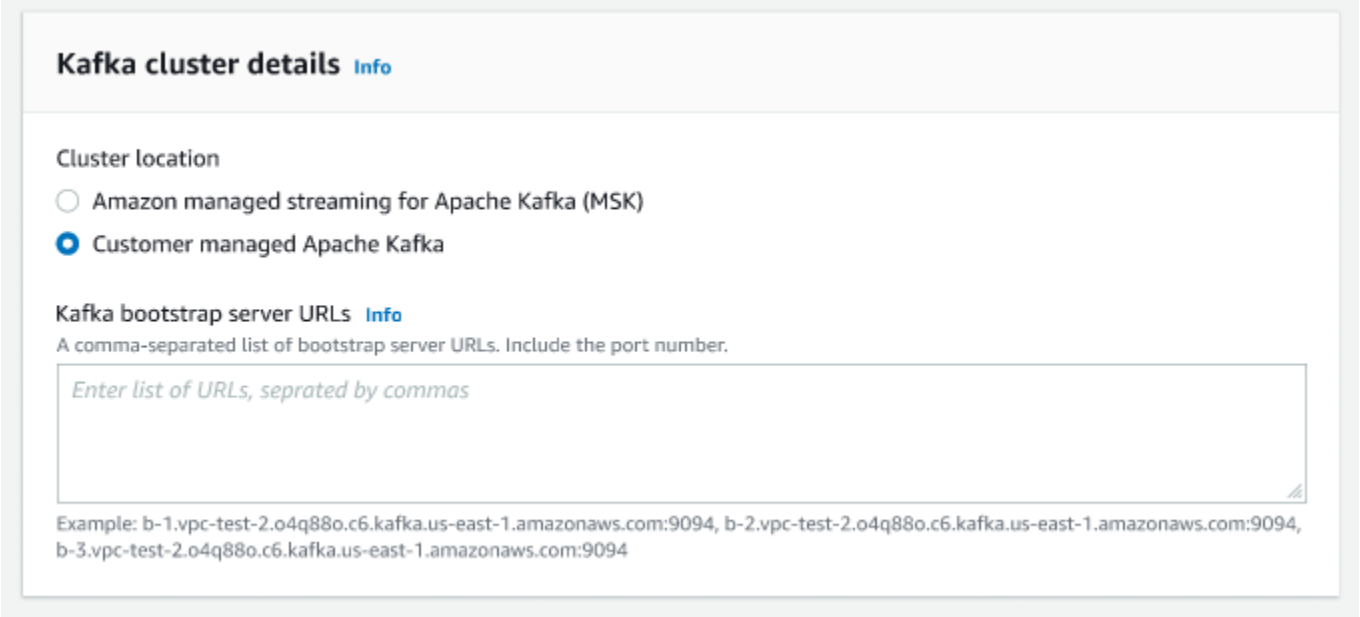
Saat membuat koneksi Kafka, memilih Kafka dari menu drop-down akan menampilkan pengaturan tambahan untuk mengkonfigurasi:

- Rincian cluster Kafka
- Autentikasi
- Enkripsi
- Opsi jaringan

Konfigurasikan detail cluster Kafka

1. Pilih lokasi cluster. Anda dapat memilih dari streaming terkelola Amazon untuk cluster Apache Kafka (MSK) atau cluster Apache Kafka yang dikelola Pelanggan. Untuk informasi selengkapnya tentang Amazon Managed streaming untuk Apache Kafka, lihat Amazon managed streaming untuk Apache Kafka (MSK).

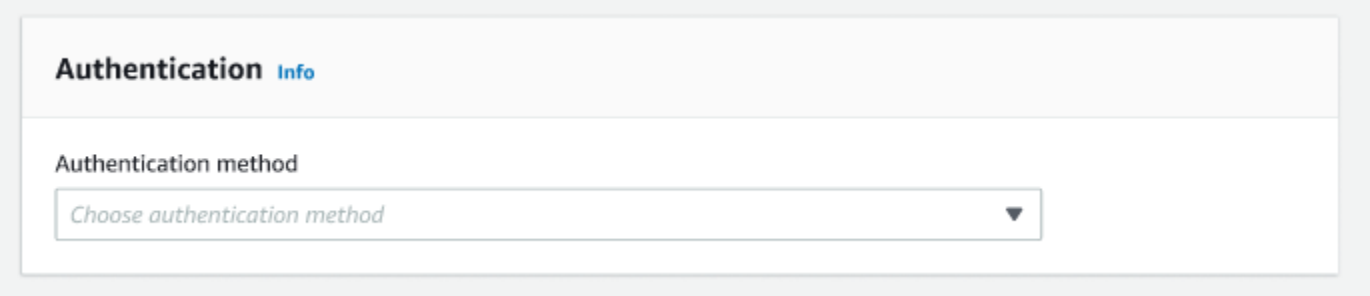
   > **Note**
   >
   > Amazon Managed Streaming for Apache Kafka hanya mendukung metode otentikasi TLS dan SASL/SCRAM-SHA-512.



2. Masukkan URL untuk server bootstrap Kafka Anda. Anda dapat memasukkan lebih dari satu dengan memisahkan setiap server dengan koma. Sertakan nomor port di akhir URL dengan menambahkan. `:<port number>`

   Sebagai contoh: `b-1.vpc-test-2.034a88o.kafka-us-east-1.amazonaws.com:9094`

Pilih metode otentikasi

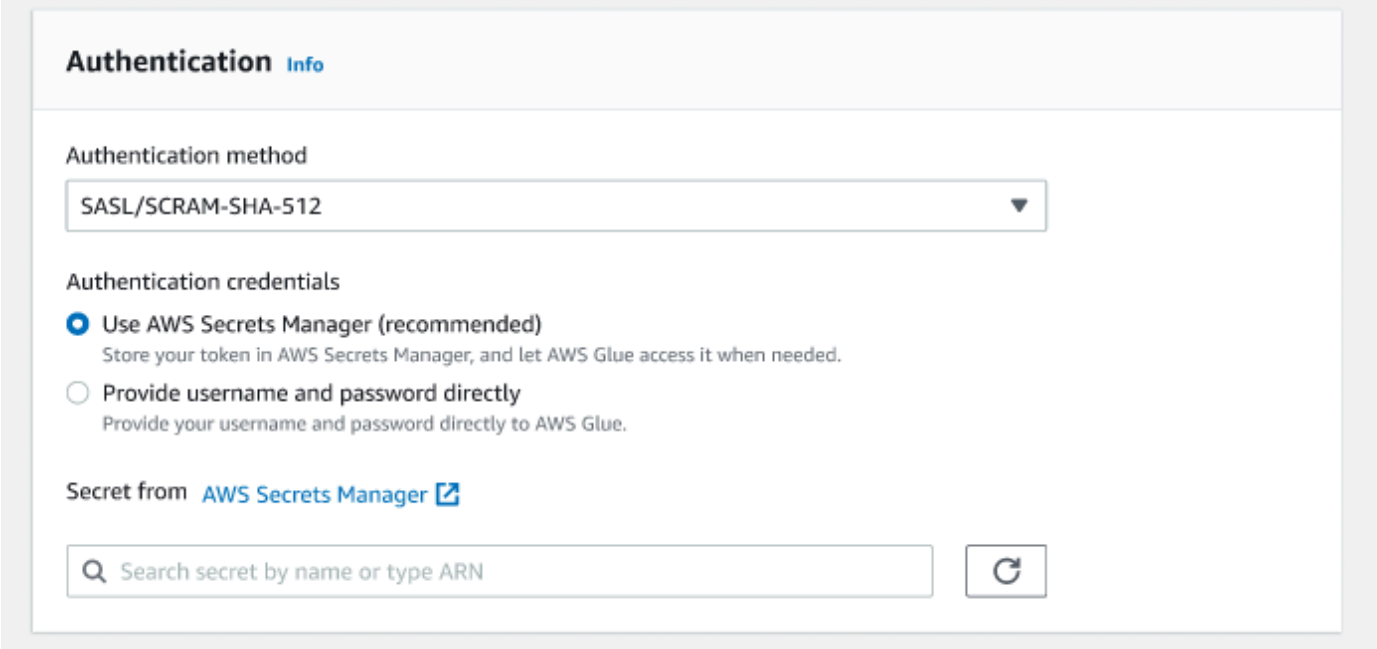


AWS Gluemendukung kerangka kerja Simple Authentication and Security Layer (SASL) untuk otentikasi. Kerangka SASL mendukung berbagai mekanisme otentikasi, dan AWS Glue menawarkan protokol SCRAM (nama pengguna dan kata sandi), GSSAPI (protokol Kerberos), dan PLAIN (nama pengguna dan kata sandi).

Saat memilih metode otentikasi dari menu drop-down, metode otentikasi klien berikut dapat dipilih:

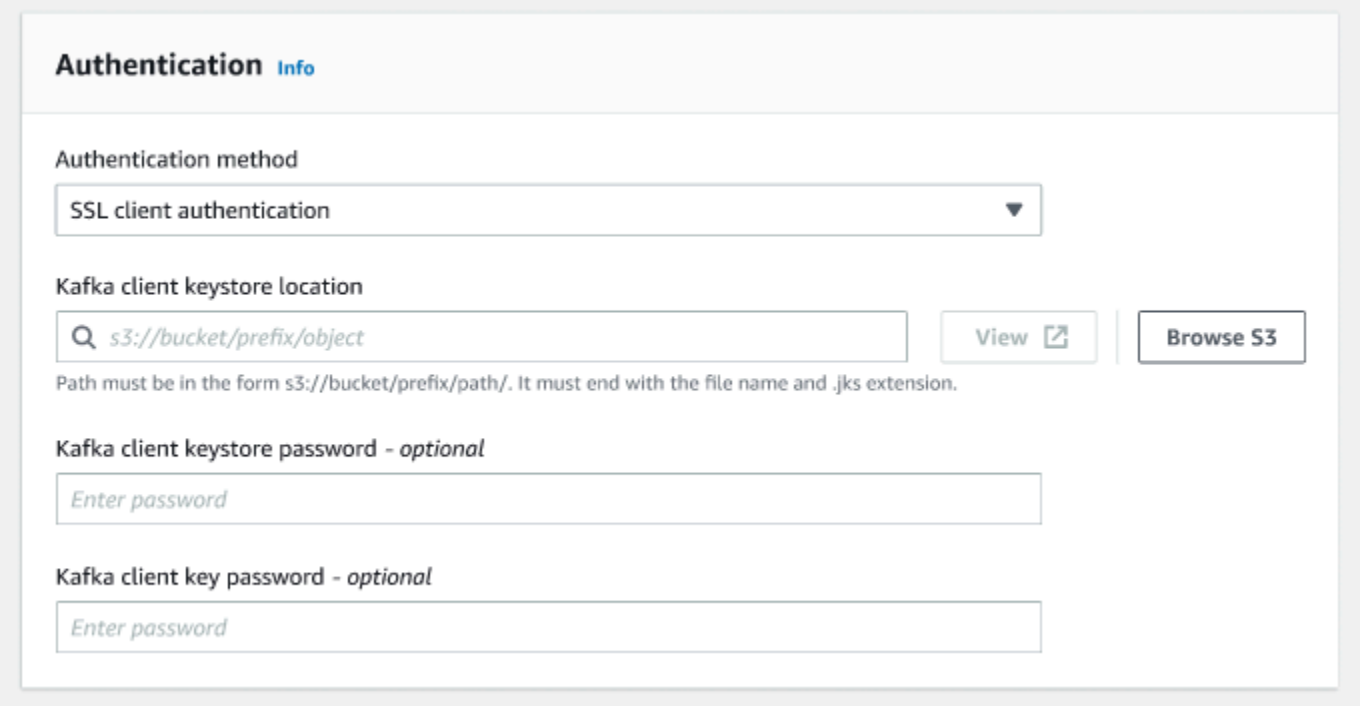
- Tidak ada - Tidak ada otentikasi. Ini berguna jika Anda membuat koneksi untuk tujuan pengujian.

- SASL/SCRAM-SHA-512 - Pilih metode otentikasi ini untuk menentukan kredensyal otentikasi. Ada dua opsi yang tersedia:
  - Gunakan AWS Secrets Manager (disarankan) - jika Anda memilih opsi ini, Anda dapat menyimpan kredensyal Anda di AWS Secrets Manager dan membiarkan AWS Glue mengakses informasi bila diperlukan. Tentukan rahasia yang menyimpan kredensyal otentikasi SSL atau SASL.



  - Berikan nama pengguna dan kata sandi secara langsung.
- SASL/GSSAPI (Kerberos) - jika Anda memilih opsi ini, Anda dapat memilih lokasi file keytab, file krb5.conf dan masukkan nama utama Kerberos dan nama layanan Kerberos. Lokasi untuk file tab tombol dan file krb5.conf harus berada di lokasi Amazon S3. Karena MSK belum mendukung SASL/GSSAPI, opsi ini hanya tersedia untuk cluster Apache Kafka yang dikelola pelanggan. Untuk informasi selengkapnya, lihat Dokumentasi MIT Kerberos: Keytab.
- SASL/PLAIN - Pilih metode otentikasi ini untuk menentukan kredensyal otentikasi. Ada dua opsi yang tersedia:
  - Gunakan AWS Secrets Manager (disarankan) - jika Anda memilih opsi ini, Anda dapat menyimpan kredensyal Anda di AWS Secrets Manager dan membiarkan AWS Glue mengakses informasi bila diperlukan. Tentukan rahasia yang menyimpan kredensyal otentikasi SSL atau SASL.
  - Berikan nama pengguna dan kata sandi secara langsung.

- Otentikasi Klien SSL - jika Anda memilih opsi ini, Anda dapat memilih lokasi keystore klien Kafka dengan menjelajahi Amazon S3. Secara opsional, Anda dapat memasukkan kata sandi keystore klien Kafka dan kata sandi kunci klien Kafka.

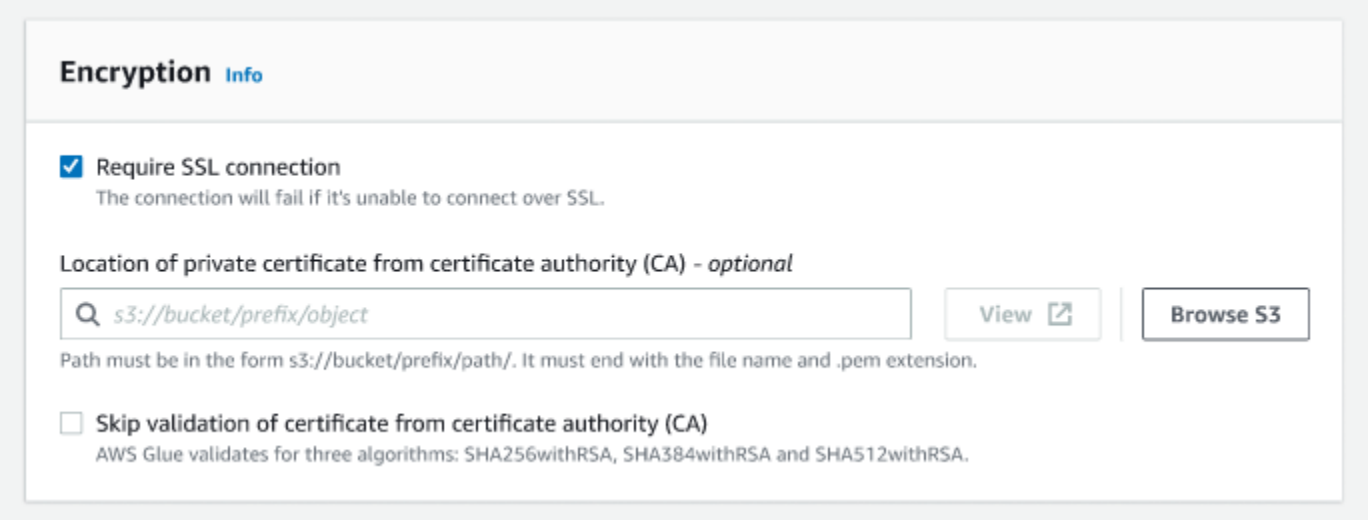


Konfigurasikan pengaturan enkripsi

1. Jika koneksi Kafka memerlukan koneksi SSL, pilih kotak centang untuk Memerlukan koneksi SSL. Perhatikan bahwa koneksi akan gagal jika tidak dapat terhubung melalui SSL. SSL untuk enkripsi dapat digunakan dengan salah satu metode otentikasi (SASL/SCRAM-SHA-512, SASL/GSSAPI, SASL/PLAIN, atau SSL Client Authentication) dan bersifat opsional.

   Jika metode otentikasi diatur ke otentikasi klien SSL, opsi ini akan dipilih secara otomatis dan akan dinonaktifkan untuk mencegah perubahan apa pun.
2. (Opsional). Pilih lokasi sertifikat pribadi dari otoritas sertifikat (CA). Perhatikan bahwa lokasi sertifikasi harus berada di lokasi S3. Pilih Browse untuk memilih file dari bucket S3 yang terhubung. Jalannya harus dalam bentuk`s3://bucket/prefix/filename.pem`. Itu harus diakhiri dengan nama file dan ekstensi.pem.
3. Anda dapat memilih untuk melewati validasi sertifikat dari otoritas sertifikat (CA). Pilih kotak centang Lewati validasi sertifikat dari otoritas sertifikat (CA). Jika kotak ini tidak dicentang, AWS Glue validasi sertifikat untuk tiga algoritma:

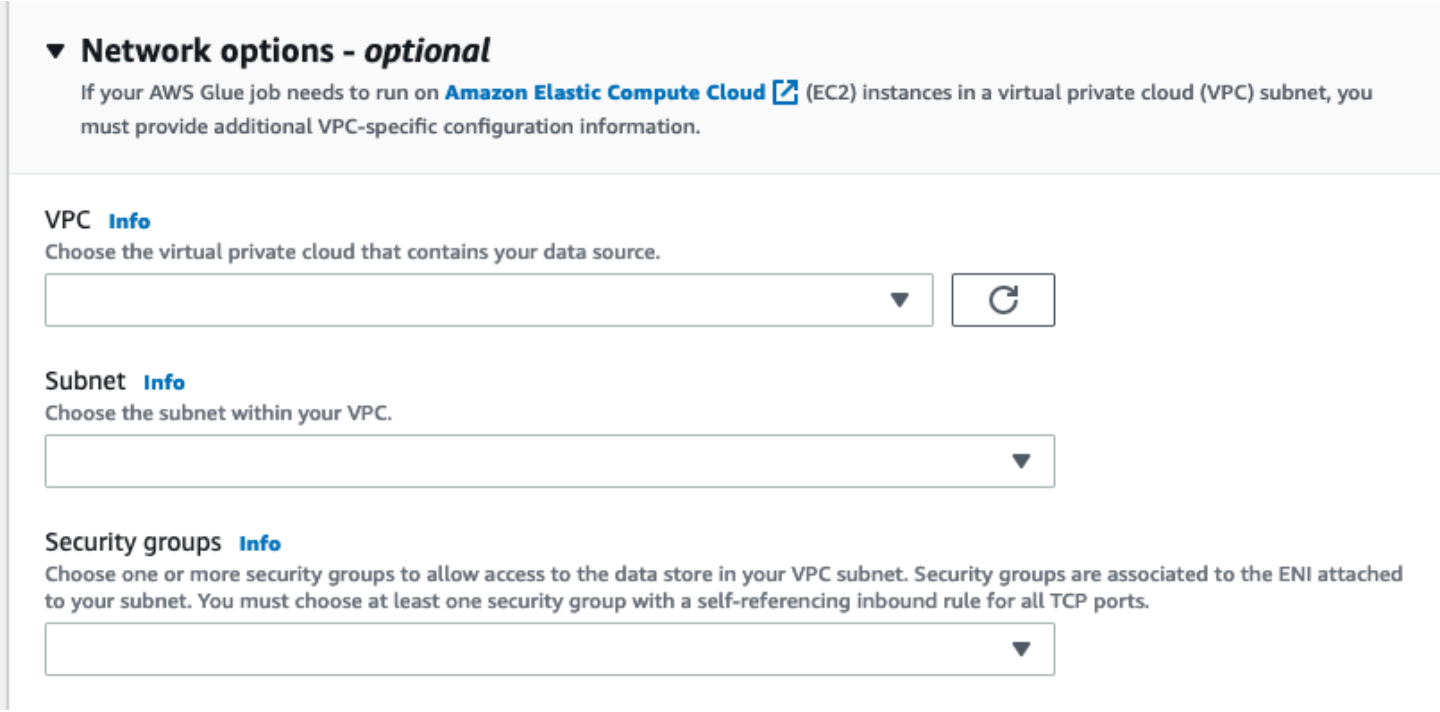
- SHA256denganRSA
- SHA384denganRSA
- SHA512denganRSA



(Opsional) Opsi jaringan

Berikut ini adalah langkah-langkah opsional untuk mengkonfigurasi grup VPC, Subnet dan Keamanan. Jika AWS Glue pekerjaan Anda perlu dijalankan di instans Amazon EC2 di subnet virtual private cloud (VPC), Anda harus memberikan informasi konfigurasi khusus VPC tambahan.

1. Pilih VPC (virtual private cloud) yang berisi sumber data Anda.
2. Pilih subnet dengan VPC Anda.
3. Pilih satu atau beberapa grup keamanan untuk mengizinkan akses ke penyimpanan data di subnet VPC Anda. Grup keamanan terkait dengan ENI yang melekat pada subnet Anda. Anda harus memilih setidaknya satu grup keamanan dengan aturan masuk referensi sendiri untuk semua port TCP.



Menulis tugas dengan konektor kustom

Anda dapat menggunakan konektor dan koneksi untuk node sumber data dan node target data diAWS Glue Studio.

Topik

- Membuat tugas yang menggunakan sebuah konektor untuk sumber data
- Konfigurasi properti sumber untuk simpul yang menggunakan konektor
- Mengkonfigurasi properti target untuk simpul yang menggunakan konektor

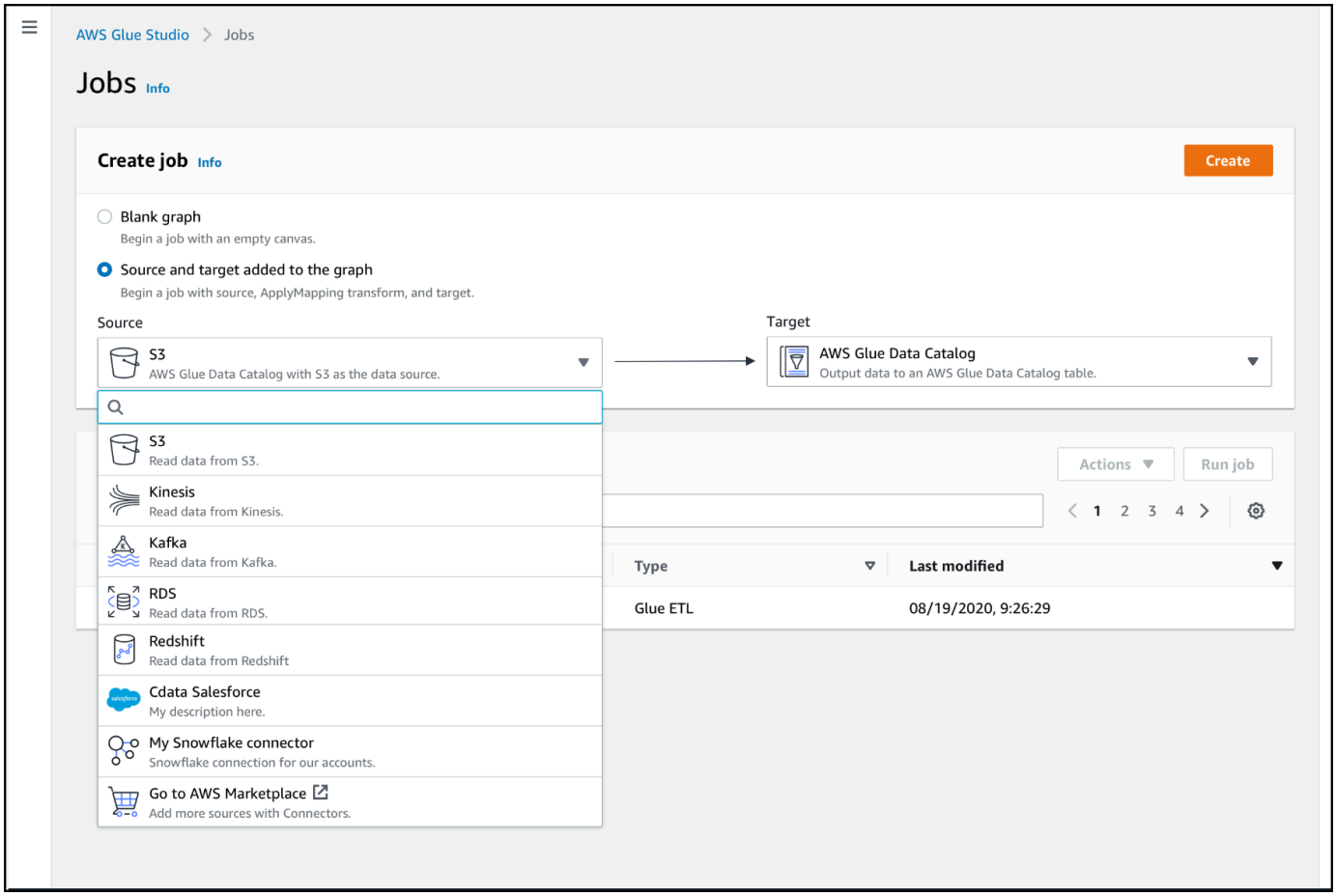
Membuat tugas yang menggunakan sebuah konektor untuk sumber data

Saat membuat sebuah tugas baru, Anda dapat memilih sebuah konektor untuk sumber data dan target data.

Untuk membuat sebuah tugas yang menggunakan konektor untuk sumber data atau target data

1. Masuk ke AWS Management Console dan buka AWS Glue Studio konsol di https://console.aws.amazon.com/gluestudio/.
2. Pada halaman Konektor, di daftar sumber daya Koneksi Anda, pilih koneksi yang ingin Anda gunakan dalam tugas Anda, dan kemudian pilih Buat tugas.

Atau, pada halaman AWS Glue Studio Pekerjaan, di bawah Buat pekerjaan, pilih Sumber dan target ditambahkan ke grafik. pada daftar drop-down Sumber, pilih konektor kustom yang ingin Anda gunakan dalam tugas Anda. Anda juga dapat memilih sebuah konektor untuk Target.



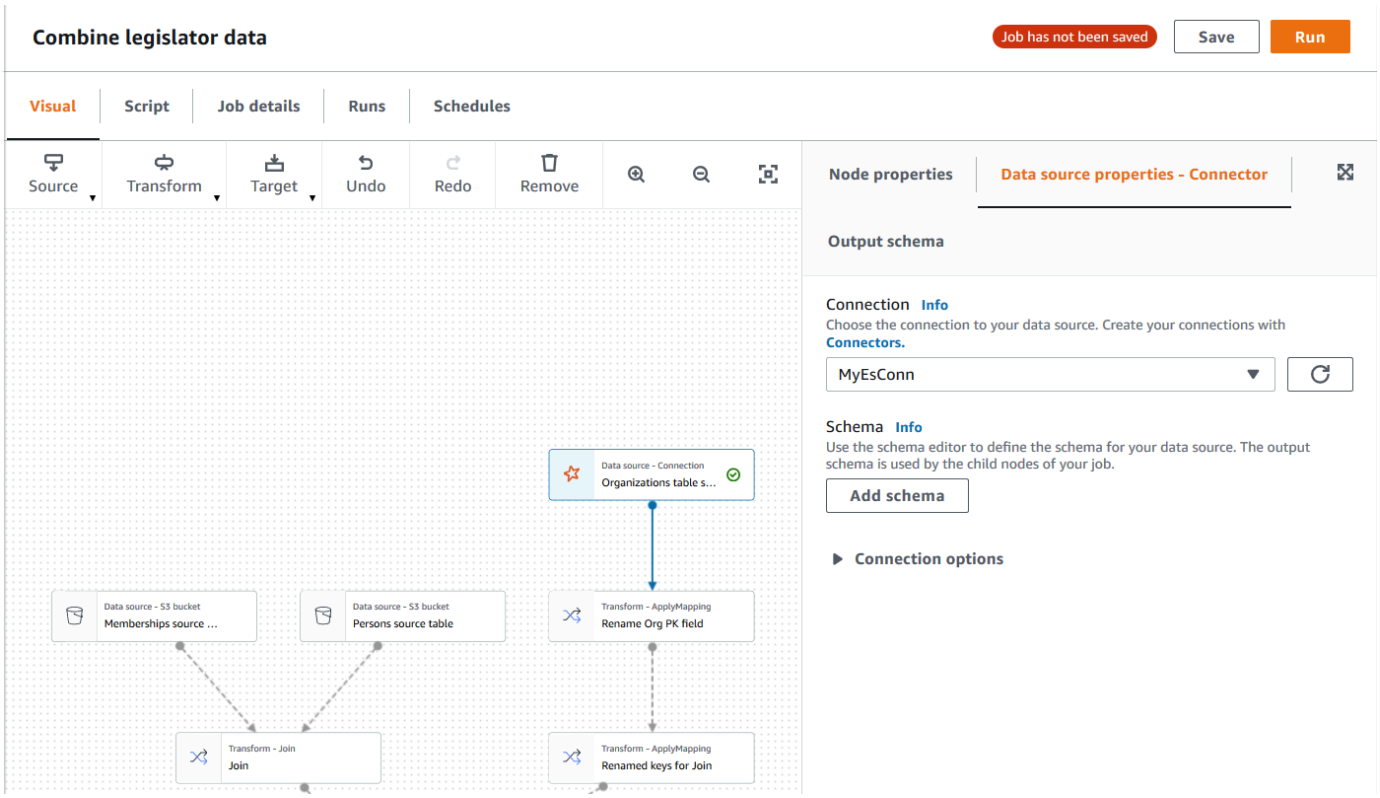
3. Pilih Buat untuk membuka editor tugas visual.
4. Konfigurasi simpul sumber data, seperti yang dijelaskan dalam Konfigurasi properti sumber untuk simpul yang menggunakan konektor.
5. Lanjutkan membuat tugas ETL Anda dengan menambahkan transformasi, penyimpanan data tambahan, dan target data, seperti yang dijelaskan dalam Visual ETL dengan AWS Glue Studio.
6. Sesuaikan lingkungan eksekusi tugas dengan mengkonfigurasi properti tugas, seperti yang dijelaskan dalam Mengubah properti tugas.
7. Simpan dan jalankan tugas.

Konfigurasi properti sumber untuk simpul yang menggunakan konektor

Setelah Anda membuat sebuah tugas yang menggunakan sebuah konektor untuk sumber data, editor tugas visual akan menampilkan grafik tugas dengan simpul sumber data yang dikonfigurasi untuk konektor tersebut. Anda harus mengkonfigurasi properti sumber data untuk simpul tersebut.

Untuk mengkonfigurasi properti untuk simpul sumber data yang menggunakan sebuah konektor

1. Pilih simpul sumber data konektor dalam grafik tugas atau tambahkan sebuah simpul baru dan pilih konektor untuk Jenis Simpul. Kemudian, di sisi kanan, di panel detail simpul, pilih tab Properti sumber data, jika belum dipilih.



2. Di tab Properti sumber data, pilih koneksi yang ingin Anda gunakan untuk tugas ini.

   Masukkan informasi tambahan yang diperlukan untuk masing-masing jenis koneksi:

   JDBC

   - Tipe masukan sumber data: Pilih untuk memberikan nama tabel atau kueri SQL sebagai sumber data. Tergantung pada pilihan Anda, Anda kemudian harus memberikan informasi tambahan berikut:

- Nama tabel: Nama tabel di sumber data. Jika sumber data tidak menggunakan tabel istilah, maka berikan nama struktur data yang sesuai, seperti yang ditunjukkan oleh informasi penggunaan konektor kustom (yang tersedia di AWS Marketplace).
- Predikat filter: Syarat klausul yang akan digunakan ketika membaca sumber data, mirip dengan klausul `WHERE`, yang digunakan untuk mengambil subset dari data.
- Kode kueri: Masukkan kueri SQL yang akan digunakan untuk mengambil set data tertentu dari sumber data. Contoh kueri SQL dasar adalah:

```
SELECT column_list FROM
                          table_name WHERE where_clause
```

- Skema: Karena AWS Glue Studio menggunakan informasi yang disimpan dalam koneksi untuk mengakses sumber data alih-alih mengambil informasi metadata dari tabel Katalog Data, Anda harus menyediakan metadata skema untuk sumber data. Pilih Tambahkan skema untuk membuka editor skema.

  Untuk petunjuk tentang cara menggunakan editor skema, lihat Mengedit skema di simpul transformasi kustom.
- Kolom partisi: (Opsional) Anda dapat memilih untuk melakukan partisi pada pembacaan data dengan memberikan nilai-nilai untuk Kolom partisi, Batas bawah, Batas atas, dan Jumlah partisi.

  Nilai `lowerBound` dan `upperBound` digunakan untuk menentukan langkah partisi, bukan untuk menyaring baris dalam tabel. Semua baris dalam tabel dipartisi dan dikembalikan.

  > **Note**
  >
  > Pemartisian kolom menambahkan syarat pemartisian tambahan untuk kueri yang digunakan untuk membaca data. Bila menggunakan sebuah kueri bukan nama sebuah tabel, maka Anda harus memvalidasi bahwa kueri bekerja dengan syarat pemartisian yang ditentukan. Misalnya:
  >
  > - Jika format kueri Anda adalah `"SELECT col1 FROM table1"`, maka uji kueri dengan menambahkan klausul `WHERE` pada akhir kueri yang menggunakan kolom partisi.

- Jika format kueri Anda adalah `"SELECT col1 FROM table1 WHERE col2=val"`, maka uji kueri dengan memperluas klausul `WHERE` dengan `AND` dan ekspresi yang menggunakan kolom partisi.

- Pengubahan jenis data: Jika sumber data menggunakan tipe data yang tidak tersedia di JDBC, gunakan bagian ini untuk menentukan bagaimana tipe data dari sumber data harus dikonversi ke dalam tipe data JDBC. Anda dapat menentukan hingga 50 konversi tipe data yang berbeda. Semua kolom dalam sumber data yang menggunakan tipe data yang sama akan dikonversi dengan cara yang sama.

  Sebagai contoh, jika Anda memiliki tiga kolom di sumber data yang menggunakan tipe data `Float`, dan Anda menunjukkan bahwa tipe data `Float` harus dikonversi ke tipe data `String` JDBC, maka semua tiga kolom yang menggunakan tipe data `Float` itu akan dikonversi ke tipe data `String`.
- Kunci bookmark tugas: Bookmark tugas membantu AWS Glue menjaga informasi status dan mencegah pengolahan ulang data lama. Tentukan satu lagi satu atau lebih kolom sebagai tombol bookmark. AWS Glue Studiomenggunakan tombol bookmark untuk melacak data yang telah diproses selama menjalankan tugas ETL sebelumnya. Kolom apa pun yang Anda gunakan untuk kunci bookmark kustom harus secara ketat dan secara monoton meningkat atau menurun, namun kesenjangan diizinkan.

  Jika Anda memasukkan beberapa kunci bookmark, maka kunci tersebut digabungkan untuk membentuk satu kunci gabungan. Kunci bookmark tugas gabungan tidak boleh berisi kolom duplikat. Jika Anda tidak menentukan kunci bookmark, secara AWS Glue Studio default menggunakan kunci primer sebagai kunci bookmark, asalkan kunci utama meningkat atau menurun secara berurutan (tanpa celah). Jika tabel tidak memiliki kunci primer, namun properti bookmark tugas diaktifkan, maka Anda harus menyediakan kunci bookmark tugas kustom. Jika tidak, pencarian kunci primer yang akan digunakan sebagai default akan gagal dan eksekusi tugas akan gagal.
- Kunci bookmark tugas yang mengurutkan urutan: Pilih apakah nilai kunci secara berurutan meningkat atau menurun.

Spark

- Skema: Karena AWS Glue Studio menggunakan informasi yang disimpan dalam koneksi untuk mengakses sumber data alih-alih mengambil informasi metadata dari tabel Katalog

Data, Anda harus menyediakan metadata skema untuk sumber data. Pilih Tambahkan skema untuk membuka editor skema.

Untuk petunjuk tentang cara menggunakan editor skema, lihat Mengedit skema di simpul transformasi kustom.

- Opsi koneksi: Masukkan pasangan nilai-kunci tambahan yang diperlukan untuk memberikan informasi koneksi atau pilihan tambahan. Misalnya, Anda dapat memasukkan nama basis data, nama tabel, nama pengguna, dan kata sandi.

  Misalnya, untuk OpenSearch, Anda memasukkan pasangan kunci-nilai berikut, seperti yang dijelaskan dalam: the section called " Tutorial: Menggunakan AWS Glue Konektor untuk Elasticsearch "

  - `es.net.http.auth.user` : *`username`*
  - `es.net.http.auth.pass` : *`password`*
  - `es.nodes` : `https://`*`<Elasticsearch endpoint>`*
  - `es.port` : `443`
  - `path`: *`<Elasticsearch resource>`*
  - `es.nodes.wan.only` : `true`

Untuk contoh opsi koneksi minimum yang akan digunakan, lihat contoh skrip pengujian MinimalSparkConnectorTest.scala aktif GitHub, yang menunjukkan opsi koneksi yang biasanya Anda berikan dalam koneksi.

Athena

- Nama tabel: Nama tabel di sumber data. Jika Anda menggunakan konektor untuk membaca dari Athena- CloudWatch log, Anda akan memasukkan nama tabel. `all_log_streams`
- Nama skema Athena: Pilih skema di sumber data Athena Anda yang sesuai dengan basis data yang berisi tabel. Jika Anda menggunakan konektor untuk membaca dari Athena- CloudWatch log, Anda akan memasukkan nama skema yang mirip dengan. `/aws/glue/`*`name`*
- Skema: Karena AWS Glue Studio menggunakan informasi yang disimpan dalam koneksi untuk mengakses sumber data alih-alih mengambil informasi metadata dari tabel Katalog Data, Anda harus menyediakan metadata skema untuk sumber data. Pilih Tambahkan skema untuk membuka editor skema.

Untuk petunjuk tentang cara menggunakan editor skema, lihat Mengedit skema di simpul transformasi kustom.

- Opsi koneksi tambahan: Masukkan pasangan nilai-kunci tambahan yang diperlukan untuk memberikan informasi koneksi atau pilihan tambahan.

Sebagai contoh, lihat `README.md` file di https://github.com/aws-samples/ aws-glue-samples / tree/master/ /development/Athena GlueCustomConnectors. Dalam langkah-langkah dalam dokumen ini, kode sampel menunjukkan opsi koneksi minimal yang diperlukan, yakni `tableName`, `schemaName`, dan `className`. Contoh kode menentukan pilihan ini sebagai bagian dari variabel `optionsMap`, tetapi Anda dapat menentukan mereka untuk koneksi Anda dan kemudian menggunakan koneksi tersebut.

3. (Opsional) Setelah memberikan informasi yang diperlukan, Anda dapat melihat skema data yang dihasilkan untuk sumber data Anda dengan memilih tab Skema output di panel detail simpul. Skema yang ditampilkan pada tab ini digunakan oleh setiap simpul anak yang Anda tambahkan ke grafik tugas.
4. (Opsional) Setelah mengkonfigurasi properti simpul dan properti sumber data, Anda dapat melihat pratinjau set data dari sumber data Anda dengan memilih tab Pratinjau data di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Ada biaya yang terkait dengan penggunaan fitur ini, dan penagihan akan dimulai langsung setelah Anda memberikan IAM role.

Mengkonfigurasi properti target untuk simpul yang menggunakan konektor

Jika Anda menggunakan sebuah konektor untuk jenis target data, maka Anda harus mengkonfigurasi properti data target simpul.

Untuk mengkonfigurasi properti untuk simpul target data yang menggunakan sebuah konektor

1. Pilih simpul target data konektor dalam grafik tugas. Kemudian, di sisi kanan, di panel detail simpul, pilih tab Properti target data, jika belum dipilih.
2. Di tab Properti target data, pilih koneksi yang akan digunakan untuk menulis ke target.

   Masukkan informasi tambahan yang diperlukan untuk masing-masing jenis koneksi:

JDBC

- Koneksi: Pilih koneksi yang akan digunakan dengan konektor Anda. Untuk informasi tentang cara membuat sebuah koneksi, lihat Membuat koneksi untuk konektor.
- Nama tabel: Nama tabel di target data. Jika target data tidak menggunakan tabel istilah, maka berikan nama struktur data yang sesuai, seperti yang ditunjukkan oleh informasi penggunaan konektor khusus (yang tersedia di AWS Marketplace).
- Ukuran Batch (Opsional): Masukkan jumlah baris atau catatan yang akan disisipkan dalam tabel target dalam satu operasi. Nilai default-nya adalah 1000 baris.

Spark

- Koneksi: Pilih koneksi yang akan digunakan dengan konektor Anda. Jika Anda tidak membuat sebuah koneksi sebelumnya, pilih Buat koneksi untuk membuatnya. Untuk informasi tentang cara membuat sebuah koneksi, lihat Membuat koneksi untuk konektor.
- Opsi koneksi: Masukkan pasangan nilai-kunci tambahan yang diperlukan untuk memberikan informasi koneksi atau pilihan tambahan. Anda dapat memasukkan sebuah nama basis data, nama tabel, nama pengguna, dan kata sandi.

  Misalnya, untuk OpenSearch, Anda memasukkan pasangan kunci-nilai berikut, seperti yang dijelaskan dalam: the section called “ Tutorial: Menggunakan AWS Glue Konektor untuk Elasticsearch ”

  - `es.net.http.auth.user` : *`username`*
  - `es.net.http.auth.pass` : *`password`*
  - `es.nodes` : `https://`*`<Elasticsearch endpoint>`*
  - `es.port` : `443`
  - `path`: *`<Elasticsearch resource>`*
  - `es.nodes.wan.only` : `true`

  Untuk contoh opsi koneksi minimum yang akan digunakan, lihat contoh skrip pengujian MinimalSparkConnectorTest.scala aktif GitHub, yang menunjukkan opsi koneksi yang biasanya Anda berikan dalam koneksi.

3. Setelah memberikan informasi yang diperlukan, Anda dapat melihat skema data yang dihasilkan untuk sumber data Anda dengan memilih tab Skema output di panel detail simpul.

Mengelola konektor dan koneksi

Anda menggunakan halaman Koneksi AWS Glue untuk mengelola konektor dan koneksi Anda.

Topik

Melihat detail konektor dan koneksi

Anda dapat melihat informasi ringkasan tentang konektor dan koneksi Anda di tabel sumber daya Konektor Anda dan Koneksi Anda pada halaman Konektor. Untuk melihat informasi detail, lakukan langkah-langkah berikut.

Untuk melihat detail konektor atau koneksi

1. Di AWS Glue Studio konsol, pilih Konektor di panel navigasi konsol.
2. Pilih konektor atau koneksi yang ingin Anda lihat informasi detailnya.
3. Pilih Tindakan, lalu pilih Lihat detail untuk membuka halaman detail untuk konektor atau koneksi tersebut.
4. Pada halaman detail, Anda dapat memilih untuk Mengedit atau Menghapus konektor atau koneksi.
    - Untuk konektor, Anda dapat memilih Buat koneksi untuk membuat sebuah koneksi baru yang menggunakan konektor.
    - Untuk koneksi, Anda bisa memilih Buat tugas untuk membuat sebuah tugas yang menggunakan koneksi.

Mengedit konektor dan koneksi

Anda gunakan halaman Konektor untuk mengubah informasi yang tersimpan di konektor dan koneksi Anda.

Untuk mengubah sebuah konektor atau koneksi

1. Di AWS Glue Studio konsol, pilih Konektor di panel navigasi konsol.
2. Pilih konektor atau koneksi yang ingin Anda ubah.
3. Pilih Tindakan, dan kemudian pilih Edit.

   Anda juga dapat memilih Lihat detail dan pada halaman detail konektor atau koneksi, Anda bisa memilih Edit.
4. Pada halaman Edit konektor atau Edit koneksi, perbarui informasi, dan kemudian pilih Simpan.

Menghapus konektor dan koneksi

Anda gunakan halaman Konektor untuk menghapus konektor dan koneksi. Jika Anda menghapus sebuah konektor, maka koneksi yang dibuat untuk konektor itu juga harus dihapus.

Untuk menghapus konektor dari AWS Glue Studio

1. Di AWS Glue Studio konsol, pilih Konektor di panel navigasi konsol.
2. Pilih konektor atau koneksi yang ingin dihapus.
3. Pilih Tindakan, lalu pilih Hapus.

   Anda juga dapat memilih Lihat detail, dan pada halaman detail konektor atau koneksi, Anda bisa memilih Hapus.
4. Verifikasi bahwa Anda ingin menghapus konektor atau koneksi dengan memasukkan **`Delete`**, lalu pilih Hapus.

   Saat Anda menghapus sebuah konektor, maka koneksi yang dibuat untuk konektor itu juga dihapus.

Setiap tugas yang menggunakan koneksi yang dihapus tidak akan lagi berfungsi. Anda dapat mengedit tugas untuk menggunakan penyimpanan data yang berbeda, atau menghapus tugas. Untuk informasi tentang cara menghapus tugas, lihat Menghapus tugas.

Jika Anda menghapus sebuah konektor, hal itu tidak membatalkan langganan konektor di AWS Marketplace. Untuk menghapus langganan untuk sebuah konektor yang dihapus, ikuti petunjuk di Membatalkan langganan untuk sebuah konektor .

Membatalkan langganan untuk sebuah konektor

Setelah Anda menghapus koneksi dan konektor dariAWS Glue Studio, Anda dapat membatalkan langganan Anda AWS Marketplace jika Anda tidak lagi memerlukan konektor.

> **Note**
>
> Jika Anda membatalkan langganan ke sebuah konektor, hal itu tidak menghapus konektor atau koneksi dari akun Anda. Setiap tugas yang menggunakan konektor dan koneksi terkait tidak akan lagi dapat menggunakan konektor dan akan gagal.
> Sebelum Anda berhenti berlangganan atau berlangganan ulang konektor dari AWS Marketplace, Anda harus menghapus koneksi dan konektor yang ada yang terkait dengan produk tersebut AWS Marketplace .

Untuk berhenti berlangganan dari konektor di AWS Marketplace

1. Masuk ke AWS Marketplace konsol di https://console.aws.amazon.com/marketplace.
2. Pilih Kelola langganan.
3. Pada halaman Kelola langganan, pilih Kelola yang ada di samping langganan konektor yang ingin Anda batalkan.
4. Pilih Tindakan lalu pilih Batalkan langganan.
5. Pilih kotak centang untuk mengetahui bahwa instans berjalan ditagihkan ke akun Anda, lalu pilih Ya, batalkan langganan..

Mengembangkan konektor kustom

Anda dapat menulis kode yang membaca data dari atau menulis data ke penyimpanan data Anda dan memformat data untuk digunakan dengan AWS Glue Studio pekerjaan. Anda dapat membuat konektor untuk penyimpanan data Spark, Athena, dan JDBC. Kode sampel yang diposting di GitHub memberikan gambaran umum tentang antarmuka dasar yang perlu Anda terapkan.

Anda memerlukan lingkungan pengembangan lokal untuk membuat kode konektor Anda. Anda dapat menggunakan IDE atau bahkan hanya editor baris perintah untuk menulis konektor Anda. Contoh lingkungan pengembangan meliputi:

- Sebuah lingkungan Scala lokal dengan perpustakaan ETL Maven AWS Glue lokal, seperti yang dijelaskan dalam Mengembangkan secara Lokal dengan Scala di AWS Glue Panduan Developer.

- IntelliJ IDE, dengan mengunduh IDE dari https://www.jetbrains.com/idea/.

Topik

Mengembangkan konektor Spark

Anda dapat membuat konektor Spark dengan Spark DataSource API V2 (Spark 2.4) untuk membaca data.

Untuk membuat konektor Spark khusus

Ikuti langkah-langkah di perpustakaan AWS Glue GitHub sampel untuk mengembangkan konektor Spark, yang terletak di https://github.com/aws-samples/ aws-glue-samples /tree/master/ GlueCustomConnectors /development/spark/README.md.

Mengembangkan konektor Athena

Anda dapat membuat konektor Athena untuk digunakan oleh AWS Glue dan AWS Glue Studio untuk menanyakan sumber data kustom.

Untuk membuat konektor Athena kustom

Ikuti langkah-langkah di perpustakaan AWS Glue GitHub sampel untuk mengembangkan konektor Athena, yang terletak di https://github.com/aws-samples/ aws-glue-samples /tree/master/ GlueCustomConnectors /development/Athena.

Mengembangkan konektor JDBC

Anda dapat membuat sebuah konektor yang menggunakan JDBC untuk mengakses penyimpanan data Anda.

Untuk membuat sebuah konektor JDBC kustom

1. Instal perpustakaan waktu aktif Spark AWS Glue di lingkungan pengembangan lokal Anda. Lihat instruksi di perpustakaan AWS Glue GitHub sampel di https://github.com/aws-samples/

aws-glue-samples /tree/master/ /development/ GlueCustomConnectors /README.md. GlueSparkRuntime

2. Menerapkan driver JDBC yang bertanggung jawab untuk mengambil data dari sumber data. Lihat dokumentasi Java untuk Java SE 8.

   Buat titik masuk dalam kode Anda yang AWS Glue Studio digunakan untuk menemukan konektor Anda. Bidang Nama kelas harus berupa path lengkap dari driver JDBC Anda.

3. Gunakan API `GlueContext` untuk membaca data dengan konektor. Pengguna dapat menambahkan lebih banyak opsi input di AWS Glue Studio konsol untuk mengonfigurasi koneksi ke sumber data, jika perlu. Untuk contoh kode yang menunjukkan cara membaca dari dan menulis ke database JDBC dengan konektor JDBC kustom, lihat Nilai kustom dan ConnectionType. AWS Marketplace

Contoh menggunakan konektor khusus dengan AWS Glue Studio

Anda dapat merujuk ke blog berikut untuk contoh cara menggunakan konektor kustom:

- Mengembangkan, menguji, dan menerapkan konektor khusus untuk penyimpanan data Anda dengan AWS Glue
- Apache Hudi: Menulis ke tabel Apache Hudi menggunakan Konektor Kustom AWS Glue
- Google BigQuery: Memigrasi data dari Google BigQuery ke Amazon S3 AWS Glue menggunakan konektor khusus
- Snowflake (JDBC): Melakukan transformasi data menggunakan Snowflake dan AWS Glue
- SingleStore: Membangun ETL cepat menggunakan SingleStore dan AWS Glue
- Salesforce: Menyerap data Salesforce ke Amazon S3 menggunakan konektor kustom CData JDBC dengan AWS Glue -
- MongoDB: Membangun tugas ETL Spark AWS Glue menggunakan Amazon DocumentDB (dengan kompatibilitas MongoDB) dan MongoDB
- Amazon Relational Database Service (Amazon RDS): Membangun pekerjaan AWS Glue Spark ETL dengan membawa driver JDBC Anda sendiri untuk Amazon RDS
- MySQL (JDBC): https://github.com/aws-samples/ /blob/master/ /pengembangan/spark/ sql.scala aws-glue-samples GlueCustomConnectors SparkConnectorMy

Mengembangkan AWS Glue konektor untuk AWS Marketplace

Sebagai AWS mitra, Anda dapat membuat konektor khusus dan mengunggahnya AWS Marketplace untuk dijual kepada AWS Glue pelanggan.

Proses untuk mengembangkan kode konektor adalah sama dengan konektor kustom, tetapi proses mengunggah dan memverifikasi kode konektor lebih terperinci. Lihat instruksi dalam Membuat Konektor untuk AWS Marketplace di GitHub situs web.

Pembatasan untuk menggunakan konektor dan koneksi di AWS Glue Studio

Saat Anda menggunakan konektor atau konektor khusus AWS Marketplace, perhatikan batasan berikut:

- API testConnection tidak didukung dengan koneksi yang dibuat untuk konektor kustom.
- Enkripsi kata sandi koneksi Katalog Data tidak didukung dengan konektor kustom.
- Anda tidak dapat menggunakan bookmark tugas jika menentukan predikat filter untuk simpul sumber data yang menggunakan konektor JDBC.
- Membuat koneksi Marketplace tidak didukung di luar antarmuka AWS Glue Studio pengguna.

Menghubungkan ke sumber data menggunakan pekerjaan Visual ETL

Saat membuat pekerjaan baru, Anda dapat menggunakan koneksi untuk terhubung ke data saat mengedit pekerjaan ETL visual. AWS Glue Anda dapat melakukan ini dengan menambahkan node sumber yang menggunakan konektor untuk membaca data, dan target node untuk menentukan lokasi untuk menulis data.

Topik

Memodifikasi properti dari node sumber data

Untuk menentukan properti sumber data, Anda terlebih dahulu harus memilih sebuah simpul sumber data dalam diagram tugas. Kemudian, di sisi kanan di panel detail simpul, Anda harus mengkonfigurasi properti simpul.

Untuk mengubah properti dari sebuah simpul sumber data

1. Pergi ke editor visual untuk sebuah tugas baru atau yang sudah disimpan.
2. Pilih simpul sumber data dalam diagram tugas.
3. Pilih tab Properti simpul di panel detail simpul, dan kemudian masukkan informasi berikut ini:
   - Nama: (Opsional) Masukkan nama yang akan dikaitkan dengan simpul dalam diagram tugas. Nama ini harus unik di antara semua simpul untuk tugas ini.
   - Jenis Simpul: Jenis simpul menentukan tindakan yang dilakukan oleh simpul. Dalam daftar opsi untuk Jenis simpul, pilih salah satu nilai yang tercantum di bawah judul Sumber data.
4. Mengkonfigurasi informasi Properti sumber data. Untuk informasi selengkapnya, lihat bagian berikut:
   - Menggunakan tabel Katalog Data untuk sumber data
   - Menggunakan sebuah konektor untuk sumber data
   - Menggunakan file di Amazon S3 untuk sumber data
   - Menggunakan sebuah sumber data streaming
5. (Opsional) Setelah mengkonfigurasi properti simpul dan properti sumber data, Anda dapat melihat skema dari sumber data Anda dengan memilih tab Skema output di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Jika Anda belum menentukan IAM role pada tab Detail tugas, maka Anda akan diminta untuk memasukkan IAM role di sini.
6. (Opsional) Setelah mengkonfigurasi properti simpul dan properti sumber data, Anda dapat melihat pratinjau set data dari sumber data Anda dengan memilih tab Pratinjau data di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Ada biaya yang terkait dengan penggunaan fitur ini, dan penagihan akan dimulai langsung setelah Anda memberikan IAM role.

Menggunakan tabel Katalog Data untuk sumber data

Untuk semua sumber data kecuali Amazon S3 dan konektor, tabel harus ada di AWS Glue Data Catalog untuk jenis sumber yang Anda pilih. AWS Gluetidak membuat tabel Katalog Data.

Untuk mengkonfigurasi simpul sumber data berdasarkan tabel Katalog Data

1. Pergi ke editor visual untuk sebuah tugas baru atau yang sudah disimpan.
2. Pilih simpul sumber data dalam diagram tugas.
3. Pilih tab Properti sumber data, dan kemudian masukkan informasi berikut:

   - Tipe sumber S3: (Untuk sumber data Amazon S3 saja) Pilih opsi Pilih tabel Katalog untuk menggunakan tabel AWS Glue Data Catalog.
   - Basis data: Pilih basis data dalam Katalog Data yang berisi tabel sumber yang ingin Anda gunakan untuk tugas ini. Anda dapat menggunakan bidang pencarian untuk mencari basis data berdasarkan namanya.
   - Tabel: Pilih tabel yang telah dikaitkan dengan sumber data dari daftar. Tabel ini harus sudah ada dalam AWS Glue Data Catalog. Anda dapat menggunakan bidang pencarian untuk mencari tabel berdasarkan namanya.
   - Predikat partisi: (Untuk sumber data Amazon S3 saja) Masukkan ekspresi Boolean berdasarkan Spark SQL yang hanya mencakup kolom pemartisian. Misalnya: `"(year=='2020' and month=='04')"`
   - Direktori sementara: (Untuk sumber data Amazon Redshift saja) Masukkan path untuk lokasi direktori kerja di Amazon S3 di mana tugas ETL Anda dapat menulis hasil antara sementara.
   - Peran yang dikaitkan dengan klaster: (Untuk sumber data Amazon Redshift saja) Masukkan sebuah peran untuk tugas ETL Anda untuk menggunakan yang berisi izin untuk klaster Amazon Redshift. Untuk informasi selengkapnya, lihat the section called “Izin sumber data dan target data”.

Menggunakan sebuah konektor untuk sumber data

Jika Anda memilih sebuah konektor untuk Jenis Simpul, ikuti petunjuk di Menulis tugas dengan konektor kustom untuk menyelesaikan konfigurasi properti sumber data.

Menggunakan file di Amazon S3 untuk sumber data

Jika Anda memilih Amazon S3 sebagai sumber data Anda, maka Anda dapat memilih salah satunya:

- Basis data dan tabel Katalog Data.
- Sebuah bucket, folder, atau file di Amazon S3.

Jika Anda menggunakan bucket Amazon S3 sebagai sumber data, AWS Glue mendeteksi skema data di lokasi yang ditentukan dari salah satu file, atau dengan menggunakan file yang Anda tentukan sebagai file sampel. Deteksi skema terjadi ketika Anda menggunakan tombol Simpulkan skema. Jika anda mengganti lokasi Amazon S3 atau file sampel, maka anda mesti memilih Simpulkan skema lagi untuk melakukan deteksi skema menggunakan informasi baru.

Untuk mengkonfigurasi sebuah simpul sumber data yang membaca langsung dari file di Amazon S3

1. Pergi ke editor visual untuk sebuah tugas baru atau yang sudah disimpan.
2. Pilih simpul sumber data dalam diagram tugas untuk sumber Amazon S3.
3. Pilih tab Properti sumber data, dan kemudian masukkan informasi berikut:

   - Tipe sumber S3: (Untuk sumber data Amazon S3 saja) Pilih opsi Lokasi S3.
   - URL S3: Masukkan path ke bucket Amazon S3, folder, atau file yang berisi data untuk tugas Anda. Anda dapat memilih Jelajahi S3 untuk memilih path dari lokasi yang tersedia ke akun Anda.
   - Rekursif: Pilih opsi ini jika Anda AWS Glue ingin membaca data dari file di folder anak di lokasi S3.

     Jika folder anak berisi data yang dipartisi, AWS Glue tidak akan menambahkan informasi partisi apa pun yang ditentukan dalam nama folder ke Katalog Data. Sebagai contoh, pertimbangkan folder di Amazon S3:

     ```
     S3://sales/year=2019/month=Jan/day=1
     S3://sales/year=2019/month=Jan/day=2
     ```

     Jika Anda memilih Recursive dan memilih `sales` folder sebagai lokasi S3 Anda, kemudian AWS Glue membaca data di semua folder anak, tetapi tidak membuat partisi untuk tahun, bulan atau hari.
   - Format data: Pilih format data yang digunakan saat menyimpan data. Anda dapat memilih JSON, CSV, atau Parquet. Nilai yang Anda pilih memberitahu tugas AWS Glue bagaimana cara membaca data dari file sumber.

> **Note**
>
> Jika Anda tidak memilih format yang benar untuk data Anda, AWS Glue mungkin menyimpulkan skema dengan benar, tetapi pekerjaan tidak akan dapat mengurai data dengan benar dari file sumber.

Anda dapat memasukkan opsi konfigurasi tambahan, tergantung pada format data yang Anda pilih.

- JSON (Notasi JavaScript Objek)
  - JsonPath: Masukkan jalur JSON yang menunjuk ke objek yang digunakan untuk mendefinisikan skema tabel. Ekspresi path JSON selalu mengacu pada struktur JSON dengan cara yang sama seperti ekspresi XPath digunakan dalam kombinasi dengan dokumen XML. "objek anggota akar" di path JSON selalu disebut sebagai $, bahkan jika itu adalah sebuah objek atau array. Path JSON dapat ditulis dalam notasi dot atau notasi kurung.

    Untuk informasi lebih lanjut tentang jalur JSON, lihat JsonPathdi situs GitHub web.
  - Catatan dalam file sumber dapat mencapai beberapa baris: Pilih opsi ini jika satu catatan dapat mencapai panjang hingga beberapa baris dalam file CSV.
- CSV (nilai yang dipisahkan koma)
  - Pembatas: Masukkan sebuah karakter untuk menunjukkan apa yang memisahkan masing-masing entri kolom dalam baris, misalnya, `;` atau `,`.
  - Karakter escape: Masukkan karakter yang digunakan sebagai karakter escape. Karakter ini menunjukkan bahwa karakter yang terletak tepat setelah karakter escape harus diambil secara harfiah, dan tidak boleh ditafsirkan sebagai pembatas.
  - Karakter kutipan: Masukkan karakter yang digunakan untuk mengelompokkan string terpisah menjadi nilai tunggal. Misalnya, Anda akan memilih Kutipan ganda (") jika Anda memiliki nilai-nilai seperti `"This is a single value"` di file CSV Anda.
  - Catatan dalam file sumber dapat mencapai beberapa baris: Pilih opsi ini jika satu catatan dapat mencapai panjang hingga beberapa baris dalam file CSV.
  - Baris pertama dari file sumber berisi header kolom: Pilih opsi ini jika baris pertama dalam file CSV berisi header kolom, bukan data.
- Parquet (Penyimpanan kolumnar Apache Parquet)

Tidak ada pengaturan tambahan untuk mengkonfigurasi data yang disimpan dalam format Parquet.

- Predikat partisi: Untuk partisi data yang dibaca dari sumber data, masukkan ekspresi Boolean berdasarkan Spark SQL yang menyertakan hanya kolom pemartisian saja. Misalnya: `"(year=='2020' and month=='04')"`
- Opsi lanjutan: Perluas bagian ini jika Anda AWS Glue ingin mendeteksi skema data Anda berdasarkan file tertentu.
  - Inferensi skema: Pilih opsi Pilih file sampel dari S3 jika Anda ingin menggunakan file tertentu alih-alih membiarkan AWS Glue memilih file.
  - File pengambilan sampel otomatis: Masukkan path ke file di Amazon S3 yang akan digunakan untuk menyimpulkan skema.

  Jika Anda mengedit sebuah simpul sumber data dan mengubah file contoh yang dipilih, pilih Muat ulang skema untuk mendeteksi skema dengan menggunakan file contoh yang baru.

4. Pilih tombol Simpulkan skema untuk mendeteksi skema dari file sumber di Amazon S3. Jika anda mengganti lokasi Amazon S3 atau file sampel, maka anda mesti memilih Simpulkan skema lagi untuk menyimpulkan skema menggunakan informasi baru.

Menggunakan sebuah sumber data streaming

Anda dapat membuat tugas extract, transform, and load (ETL) yang berjalan terus menerus dan mengkonsumsi data dari sumber streaming di Amazon Kinesis Data Streams, Apache Kafka, dan Amazon Managed Streaming for Apache Kafka (Amazon MSK).

Untuk mengkonfigurasi properti untuk sebuah sumber data streaming

1. Pergi ke editor grafik visual untuk sebuah tugas baru atau yang sudah disimpan.
2. Pilih simpul sumber data dalam grafik untuk Kafka atau Kinesis Data Streams.
3. Pilih tab Properti sumber data, dan kemudian masukkan informasi berikut:

   Kinesis

   - Jenis sumber Kinesis: Pilih opsi Streaming detail untuk menggunakan akses langsung ke sumber streaming atau pilih tabel Katalog Data untuk menggunakan informasi yang disimpan di sana.

Jika Anda memilih Detail Streaming, tentukan informasi tambahan berikut.

- Lokasi aliran data: Pilih apakah aliran dikaitkan dengan pengguna saat ini, atau jika dikaitkan dengan pengguna yang berbeda.
- Wilayah: Pilih Wilayah AWS tempat aliran ada. Informasi ini digunakan untuk membangun ARN untuk mengakses aliran data.
- Streaming ARN: Masukkan Nama Sumber Daya Amazon (ARN) untuk aliran data Kinesis. Jika aliran terletak di dalam akun saat ini, Anda dapat memilih nama aliran dari daftar drop-down. Anda dapat menggunakan bidang pencarian untuk mencari aliran data dengan namanya atau ARN.
- Format data: Pilih format yang digunakan oleh aliran data dari daftar.

  AWS Gluesecara otomatis mendeteksi skema dari data streaming.

Jika Anda memilih tabel Katalog Data, tentukan informasi tambahan berikut.

- Database: (Opsional) Pilih database dalam Katalog AWS Glue Data yang berisi tabel yang terkait dengan sumber data streaming Anda. Anda dapat menggunakan bidang pencarian untuk mencari basis data berdasarkan namanya.
- Tabel: (Opsional) Pilih tabel yang telah dikaitkan dengan sumber data dari daftar. Tabel ini harus sudah ada di Katalog AWS Glue Data. Anda dapat menggunakan bidang pencarian untuk mencari tabel berdasarkan namanya.
- Deteksi skema: Pilih opsi ini untuk AWS Glue mendeteksi skema dari data streaming, daripada menggunakan informasi skema dalam tabel Katalog Data. Opsi ini diaktifkan secara otomatis jika Anda memilih opsi Detail Stream.

- Posisi awal: Secara default, pekerjaan ETL menggunakan opsi Paling Awal, yang berarti membaca data yang dimulai dengan catatan tertua yang tersedia di aliran. Sebagai gantinya, Anda dapat memilih Terbaru, yang menunjukkan pekerjaan ETL harus mulai membaca tepat setelah catatan terbaru dalam aliran.
- Ukuran jendela: Secara default, tugas ETL Anda memproses dan menulis data dalam jendela 100-detik. Hal ini memungkinkan data diproses secara efisien dan memungkinkan agregasi untuk dilakukan pada data yang datang lebih lambat dari yang diharapkan. Anda dapat mengubah ukuran jendela ini untuk meningkatkan ketepatan waktu atau akurasi agregasi.

  AWS Glue pekerjaan streaming menggunakan pos pemeriksaan daripada bookmark pekerjaan untuk melacak data yang telah dibaca.

- Opsi koneksi: Perluas bagian ini untuk menambahkan pasangan nilai kunci untuk menentukan opsi koneksi tambahan. Untuk informasi tentang opsi apa yang dapat Anda tentukan di sini, lihat "ConnectionType": "kinesis" di Panduan Pengembang.AWS Glue

Kafka

- Sumber Apache Kafka: Pilih opsi Streaming detail untuk menggunakan akses langsung ke sumber streaming atau pilih tabel Katalog Data untuk menggunakan informasi yang disimpan di sana sebagai gantinya.

  Jika Anda memilih tabel Katalog Data, tentukan informasi tambahan berikut.

  - Database: (Opsional) Pilih database dalam Katalog AWS Glue Data yang berisi tabel yang terkait dengan sumber data streaming Anda. Anda dapat menggunakan bidang pencarian untuk mencari basis data berdasarkan namanya.
  - Tabel: (Opsional) Pilih tabel yang telah dikaitkan dengan sumber data dari daftar. Tabel ini harus sudah ada di Katalog AWS Glue Data. Anda dapat menggunakan bidang pencarian untuk mencari tabel berdasarkan namanya.
  - Deteksi skema: Pilih opsi ini untuk AWS Glue mendeteksi skema dari data streaming, daripada menyimpan informasi skema dalam tabel Katalog Data. Opsi ini diaktifkan secara otomatis jika Anda memilih opsi Detail Stream.

  Jika Anda memilih Detail Streaming, tentukan informasi tambahan berikut.

  - Nama koneksi: Pilih AWS Glue koneksi yang berisi informasi akses dan otentikasi untuk aliran data Kafka. Anda harus menggunakan koneksi dengan sumber data streaming Kafka. Jika koneksi tidak ada, Anda dapat menggunakan AWS Glue konsol untuk membuat koneksi untuk aliran data Kafka Anda.
  - Nama topik: Masukkan nama topik yang akan dibaca.
  - Format data: Pilih format yang akan digunakan saat membaca data dari aliran acara Kafka.
- Posisi awal: Secara default, pekerjaan ETL menggunakan opsi Paling Awal, yang berarti membaca data yang dimulai dengan catatan tertua yang tersedia di aliran. Sebagai gantinya, Anda dapat memilih Terbaru, yang menunjukkan pekerjaan ETL harus mulai membaca tepat setelah catatan terbaru dalam aliran.
- Ukuran jendela: Secara default, tugas ETL Anda memproses dan menulis data dalam jendela 100-detik. Hal ini memungkinkan data diproses secara efisien dan memungkinkan

agregasi untuk dilakukan pada data yang datang lebih lambat dari yang diharapkan. Anda dapat mengubah ukuran jendela ini untuk meningkatkan ketepatan waktu atau akurasi agregasi.

Tugas streaming AWS Glue menggunakan pos pemeriksaan, bukan bookmark tugas, untuk melacak data yang telah dibaca.

- Opsi koneksi: Perluas bagian ini untuk menambahkan pasangan nilai kunci untuk menentukan opsi koneksi tambahan. Untuk informasi tentang opsi apa yang dapat Anda tentukan di sini, lihat "ConnectionType": "kafka" di Panduan Pengembang.AWS Glue

> **Note**
>
> Pratinjau data saat ini tidak didukung untuk sumber data streaming.

References

Praktik Terbaik

- Buat pipeline layanan ETL untuk memuat data secara bertahap dari Amazon S3 ke penggunaan Amazon RedshiftAWS Glue

Pemrograman ETL

- Jenis dan opsi koneksi untuk ETL di AWS Glue
- Nilai ConnectionType JDBC
- Opsi lanjutan untuk memindahkan data ke dan dari Amazon Redshift

Menambahkan koneksi JDBC menggunakan driver JDBC Anda sendiri

Anda dapat menggunakan driver JDBC Anda sendiri saat menggunakan koneksi JDBC. Ketika driver default yang digunakan oleh AWS Glue crawler tidak dapat terhubung ke database, Anda dapat menggunakan Driver JDBC Anda sendiri. Misalnya, jika Anda ingin menggunakan SHA-256 dengan database Postgres Anda, dan driver postgres yang lebih lama tidak mendukung ini, Anda dapat menggunakan driver JDBC Anda sendiri.

Sumber data yang didukung

Sumber data yang didukung	Sumber data yang tidak didukung
MySQL	Kepingan salju
Postgres	
Oracle	
Redshift	
SQL Server	
Aurora*	

* Didukung jika driver JDBC asli sedang digunakan. Tidak semua fitur driver dapat dimanfaatkan.

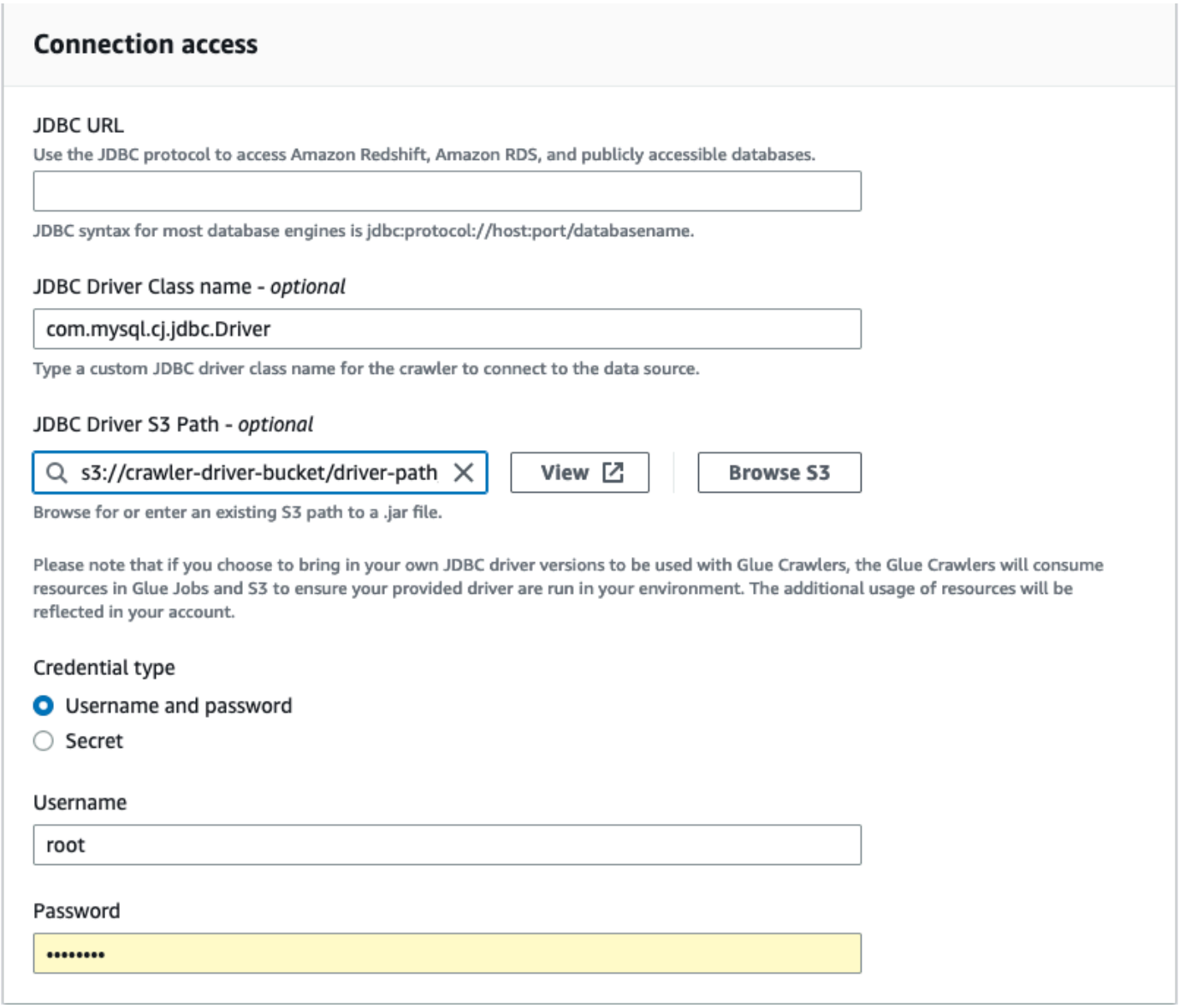
Menambahkan driver JDBC ke koneksi JDBC

> **Note**
>
> Jika Anda memilih untuk membawa versi driver JDBC Anda sendiri, AWS Glue crawler akan menggunakan sumber daya dalam AWS Glue pekerjaan dan bucket Amazon S3 untuk memastikan driver yang Anda berikan dijalankan di lingkungan Anda. Penggunaan sumber daya tambahan akan tercermin di akun Anda. Biaya untuk AWS Glue crawler dan pekerjaan berada di bawah AWS Glue kategori dalam penagihan. Selain itu, menyediakan driver JDBC Anda sendiri tidak berarti bahwa crawler dapat memanfaatkan semua fitur pengemudi.

Untuk menambahkan driver JDBC Anda sendiri ke koneksi JDBC:

1. Tambahkan file driver JDBC ke lokasi Amazon S3. Anda dapat membuat bucket dan/atau folder atau menggunakan bucket dan/atau folder yang ada.
2. Di AWS Glue konsol, pilih Koneksi di menu sebelah kiri di bawah Katalog Data, lalu buat koneksi baru.
3. Lengkapi bidang untuk properti Koneksi dan pilih JDBC untuk jenis Koneksi.

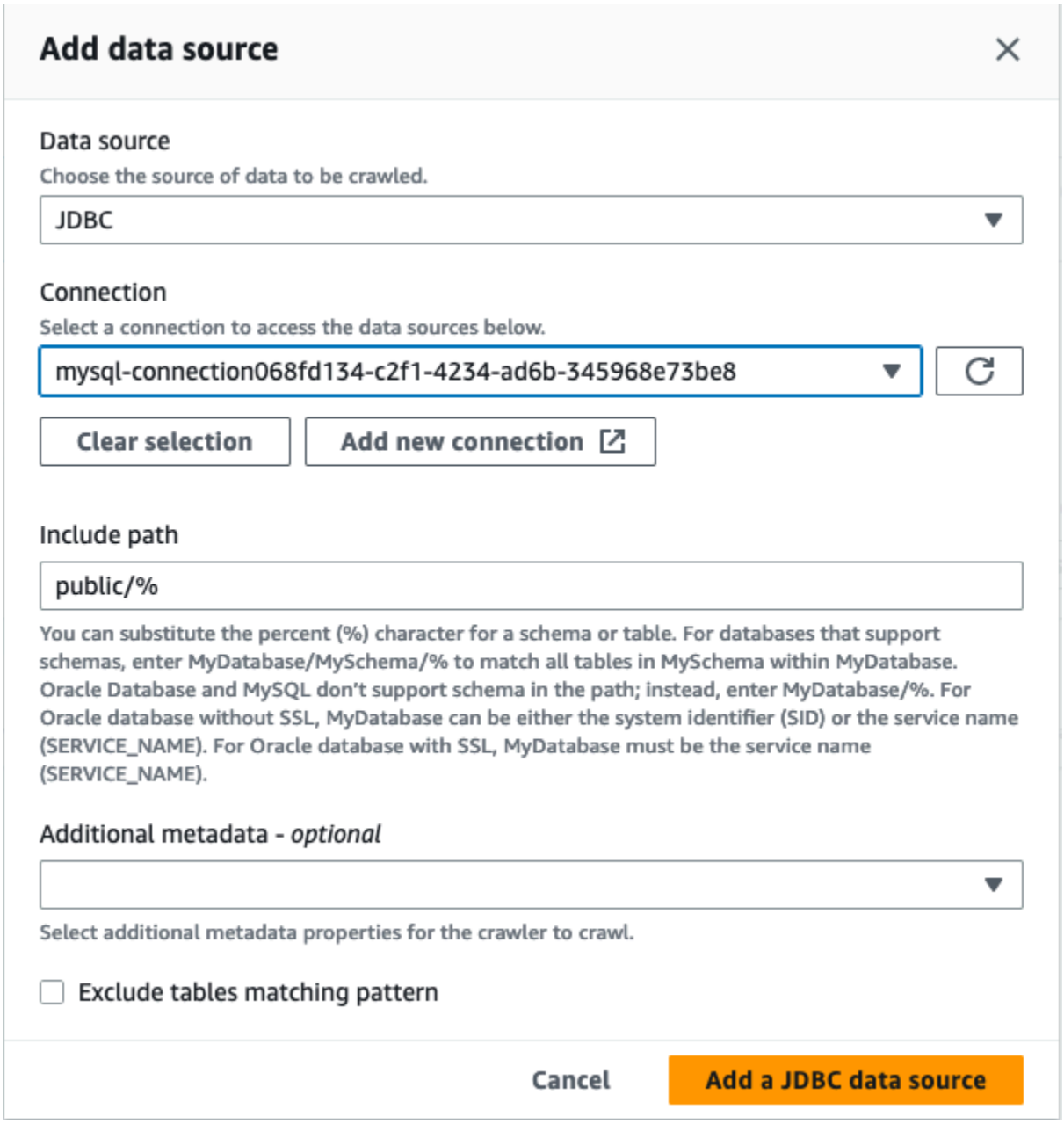
4. Dalam akses Koneksi, masukkan URL JDBC dan nama Kelas Driver JDBC - opsional. Nama kelas driver harus untuk sumber data yang didukung oleh crawler. AWS Glue



5. Pilih jalur Amazon S3 tempat driver JDBC berada di JDBC Driver Amazon S3 Path — bidang opsional.
6. Lengkapi bidang untuk Jenis kredenal jika memasukkan nama pengguna dan kata sandi atau rahasia. Setelah selesai, pilih Buat koneksi.

> **Note**
>
> Koneksi pengujian tidak didukung saat ini. Saat merayapi sumber data dengan driver JDBC yang Anda berikan, crawler melewatkan langkah ini.

7. Tambahkan koneksi yang baru dibuat ke crawler. Di AWS Glue konsol, pilih Crawler di menu sebelah kiri di bawah Katalog Data, lalu buat crawler baru.
8. Di Wizard Add crawler, di Langkah 2 pilih Tambahkan sumber data.



9. Pilih JDBC sebagai sumber data dan pilih koneksi yang dibuat pada langkah sebelumnya. Lengkap
10. Untuk menggunakan driver JDBC Anda sendiri dengan AWS Glue crawler, tambahkan izin berikut ke peran yang digunakan oleh crawler:

- Berikan izin untuk tindakan pekerjaan berikut:`CreateJob`,,`DeleteJob`, `GetJobGetJobRun`,`StartJobRun`.
- Berikan izin untuk tindakan IAM: `iam:PassRole`
- Berikan izin untuk tindakan Amazon S3`s3:DeleteObjects`:`s3:GetObject`,,,`s3:ListBucket`. `s3:PutObject`
- Berikan akses utama layanan ke ember/folder dalam kebijakan IAM.

Contoh kebijakan IAM:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::bucket-name/driver-parent-folder/driver.jar",
                "arn:aws:s3:::bucket-name"
            ]
        }
    ]
}
```

11. Jika Anda menggunakan VPC, Anda harus mengizinkan akses ke titik akhir dengan membuat AWS Glue titik akhir antarmuka dan menambahkannya ke tabel rute Anda. Untuk informasi selengkapnya, lihat Membuat titik akhir VPC antarmuka untuk AWS Glue
12. Jika Anda menggunakan enkripsi di Katalog Data Anda, buat titik akhir AWS KMS antarmuka dan tambahkan ke tabel rute Anda. Untuk informasi selengkapnya, lihat Membuat titik akhir VPC untuk. AWS KMS

Menguji AWS Glue koneksi

Sebagai praktik terbaik, sebelum Anda menggunakan AWS Glue koneksi dalam pekerjaan ETL, gunakan AWS Glue konsol untuk menguji koneksi. AWS Gluemenggunakan parameter dalam koneksi Anda untuk mengonfirmasi bahwa ia dapat mengakses penyimpanan data Anda dan melaporkan kesalahan apa pun. Untuk informasi tentang koneksi AWS Glue, lihat Menghubungkan ke data.

Untuk menguji sebuah koneksi AWS Glue

1. Masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, di bawah Katalog Data, pilih Koneksi. Anda juga dapat memilih Koneksi data di atas Katalog Data di panel navigasi.
3. Di Koneksi, pilih kotak centang di sebelah koneksi yang diinginkan, lalu pilih Tindakan. Di menu tarik-turun, pilih Uji koneksi.
4. Di kotak dialog Uji koneksi, pilih peran atau pilih Buat peran IAM untuk pergi ke konsol AWS Identity and Access Management (IAM) untuk membuat peran baru. Peran harus memiliki izin pada penyimpanan data.
5. Pilih Konfirmasi.

   Pengujian dimulai dan memerlukan waktu beberapa menit untuk selesai. Jika pengujian gagal, pilih Troubleshoot untuk melihat langkah-langkah untuk menyelesaikan masalah.
6. Pilih Log untuk melihat log masuk CloudWatch. Anda harus memiliki izin IAM yang diperlukan untuk melihat log tersebut. Untuk informasi selengkapnya, lihat Kebijakan AWS Terkelola (yang telah ditentukan) untuk CloudWatch Log di Panduan Pengguna CloudWatch Log Amazon.

Mengkonfigurasi AWS panggilan untuk melalui VPC Anda

Parameter pekerjaan khusus `disable-proxy-v2` memungkinkan Anda merutekan panggilan ke layanan seperti Amazon S3, CloudWatch, dan AWS Glue melalui VPC Anda. Secara default, AWS Glue menggunakan proxy lokal untuk mengirim lalu lintas melalui AWS Glue VPC untuk mengunduh skrip dan pustaka dari Amazon S3, untuk mengirim permintaan untuk menerbitkan log dan metrik, dan CloudWatch untuk mengirim permintaan untuk mengakses katalog data. AWS Glue Proxy ini memungkinkan pekerjaan berfungsi secara normal meskipun VPC Anda tidak mengonfigurasi rute yang tepat ke AWS layanan lain, seperti Amazon S3 CloudWatch, dan. AWS

Glue AWS Gluesekarang menawarkan parameter bagi Anda untuk mematikan perilaku ini. Untuk informasi selengkapnya, lihat Parameter Job yang digunakan oleh AWS Glue. AWS Glueakan terus menggunakan proxy lokal untuk menerbitkan CloudWatch log AWS Glue pekerjaan Anda.

> **Note**
>
> - Fitur ini didukung untuk AWS Glue pekerjaan dengan AWS Glue versi 2.0 ke atas. Saat menggunakan fitur ini, Anda perlu memastikan bahwa VPC Anda telah mengonfigurasi rute ke Amazon S3 melalui NAT atau titik akhir VPC layanan.
> - Parameter pekerjaan yang tidak digunakan lagi `disable-proxy` hanya merutekan panggilan Anda ke Amazon S3 untuk mengunduh skrip dan pustaka melalui VPC Anda. Disarankan untuk menggunakan parameter baru `disable-proxy-v2` sebagai gantinya.

Contoh penggunaan

Buat AWS Glue pekerjaan dengan`disable-proxy-v2`:

```
aws glue create-job \
    --name no-proxy-job \
    --role GlueDefaultRole \
    --command "Name=glueetl,ScriptLocation=s3://my-bucket/glue-script.py" \
    --connections Connections="traffic-monitored-connection" \
    --default-arguments '{"--disable-proxy-v2" : "true"}'
```

Menghubungkan ke penyimpanan data JDBC di VPC

Biasanya, Anda membuat sumber daya di dalam Amazon Virtual Private Cloud (Amazon VPC) sehingga sumber daya tidak dapat diakses melalui internet publik. Secara default, AWS Glue tidak dapat mengakses sumber daya di dalam VPC. Untuk mengaktifkan AWS Glue untuk mengakses sumber daya di dalam VPC Anda, Anda harus memberikan informasi konfigurasi spesifik-VPC yang mencakup ID subnet VPC dan ID grup keamanan. AWS Glue menggunakan informasi ini untuk menyiapkan antarmuka jaringan elastis yang memungkinkan fungsi Anda untuk connect dengan aman ke sumber daya lain di VPC privat Anda.

Saat menggunakan titik akhir VPC, tambahkan ke tabel rute Anda. Untuk informasi selengkapnya, lihat Membuat titik akhir VPC antarmuka untuk dan. AWS Glue Prasyarat

Saat menggunakan enkripsi di Katalog Data, buat titik akhir antarmuka KMS dan tambahkan ke tabel rute Anda. Untuk informasi lebih lanjut, lihat Membuat titik akhir VPC untuk. AWS KMS

Mengakses Data VPC Menggunakan antarmuka jaringan elastis

Saat AWS Glue terhubung ke penyimpanan data JDBC di VPC, AWS Glue menciptakan antarmuka jaringan elastis (dengan prefiks `Glue_`) di akun Anda untuk mengakses data VPC Anda. Anda tidak dapat menghapus antarmuka jaringan ini selama dilampirkan ke AWS Glue. Sebagai bagian dari pembuatan antarmuka jaringan elastis, AWS Glue mengaitkan satu atau beberapa grup keamanan padanya. Untuk memungkinkan AWS Glue untuk membuat antarmuka jaringan, grup keamanan yang dikaitkan dengan sumber daya harus memungkinkan akses masuk dengan aturan sumber. Aturan ini berisi grup keamanan yang dikaitkan dengan sumber daya. Hal ini memberikan antarmuka jaringan elastis akses ke penyimpanan data Anda dengan grup keamanan yang sama.

Untuk mengaktifkan AWS Glue untuk berkomunikasi dengan komponen-komponennya, tentukan grup keamanan dengan aturan inbound self-referencing untuk semua port TCP. Dengan membuat aturan self-referencing, Anda dapat membatasi sumber ke grup keamanan yang sama di VPC dan tidak membukanya untuk semua jaringan. Grup keamanan default untuk VPC Anda mungkin sudah memiliki aturan self-referencing inbound untuk `ALL Traffic`.

Anda dapat membuat aturan di konsol Amazon VPC. Untuk memperbarui pengaturan aturan melalui AWS Management Console, buka konsol VPC di (https://console.aws.amazon.com/vpc/), dan pilih grup keamanan yang sesuai. Tentukan aturan inbound untuk `ALL TCP` agar memiliki nama grup keamanan yang sama seperti sumbernya. Untuk informasi selengkapnya tentang aturan grup keamanan, lihat Grup Keamanan untuk VPC Anda.

Setiap antarmuka jaringan elastis ditetapkan untuknya alamat IP privat dari rentang alamat IP dalam subnet yang Anda tentukan. Antarmuka jaringan tidak ditetapkan untuknya alamat IP publik. AWS Glue memerlukan akses internet (contohnya, untuk mengakses AWS yang tidak memiliki VPC endpoint). Anda dapat mengkonfigurasi instans terjemahan alamat jaringan (NAT) di dalam VPC Anda, atau Anda dapat menggunakan gateway NAT Amazon VPC. Untuk informasi lebih lanjut, lihat NAT Gateway di Panduan Pengguna Amazon VPC. Anda tidak dapat langsung menggunakan gateway internet yang dilampirkan ke VPC Anda sebagai rute dalam tabel rute subnet Anda karena itu memerlukan antarmuka jaringan untuk memiliki alamat IP publik.

Atribut jaringan VPC `enableDnsHostnames` dan `enableDnsSupport` harus diatur ke BETUL. Untuk informasi selengkapnya, lihat Menggunakan DNS dengan VPC Anda.

> **Important**
>
> Jangan menaruh penyimpanan data Anda di subnet publik atau di subnet privat yang tidak memiliki akses internet. Sebaliknya, hanya lampirkan itu ke subnet privat yang memiliki akses internet melalui instans NAT atau gateway NAT Amazon VPC.

Properti antarmuka jaringan elastis

Untuk membuat antarmuka jaringan elastis, Anda harus menyediakan properti berikut:

VPC

Nama VPC yang berisi penyimpanan data Anda.

Subnet

Subnet di VPC yang berisi penyimpanan data Anda.

Grup keamanan

Grup keamanan yang dikaitkan dengan penyimpanan data Anda. AWS Glue mengaitkan grup keamanan ini dengan antarmuka jaringan elastis yang dilampirkan pada subnet VPC Anda. Untuk mengizinkan komponen AWS Glue untuk berkomunikasi dan juga mencegah akses dari jaringan lain, setidaknya ada satu grup keamanan yang dipilih yang harus menentukan aturan masuk referensi sendiri untuk semua port TCP.

Untuk informasi tentang mengelola VPC dengan Amazon Redshift, lihat Mengelola Klaster di Amazon Virtual Private Cloud (VPC).

Untuk informasi tentang mengelola VPC dengan Amazon Relational Database Service (Amazon RDS), lihat Bekerja dengan Instans DB Amazon RDS di sebuah VPC.

Menggunakan koneksi MongoDB atau MongoDB Atlas

Setelah membuat koneksi untuk MongoDB atau MongoDB Atlas, Anda dapat menggunakan koneksi tersebut dalam tugas ETL Anda. Anda membuat tabel di AWS Glue Data Catalog dan menentukan koneksi MongoDB atau MongoDB Atlas untuk atribut tabel tersebut. `connection`

AWS Gluemenyimpan koneksi `url` dan kredenal Anda di koneksi MongoDB. Format URI koneksi adalah sebagai berikut:

- Untuk MongoDB: mongodb: //host:port/database. Host dapat berupa nama host, alamat IP, atau soket domain UNIX. Jika string koneksi tidak menentukan port, ia menggunakan port MongoDB default, 27017.
- Untuk MongoDB Atlas: mongodb+srv: //server.example.com/database. Host dapat berupa nama host yang mengikuti sesuai dengan catatan DNS SRV. Format SRV tidak memerlukan port dan akan menggunakan port MongoDB default, 27017.

Selain itu, Anda dapat menentukan opsi dalam skrip tugas Anda. Untuk informasi selengkapnya, lihat the section called “Koneksi MongoDB”.

Merayapi penyimpanan data Amazon S3 menggunakan titik akhir VPC

Untuk tujuan keamanan, audit, atau kontrol, Anda mungkin ingin penyimpanan data Amazon S3 atau tabel Katalog Data yang didukung Amazon S3 hanya dapat diakses melalui lingkungan Amazon Virtual Private Cloud (Amazon VPC). Topik ini menjelaskan cara membuat dan menguji koneksi ke penyimpanan data Amazon S3 atau tabel Katalog Data yang didukung Amazon S3 di titik akhir VPC menggunakan jenis koneksi. `Network`

Lakukan tugas berikut untuk menjalankan crawler di sebuah penyimpanan data:

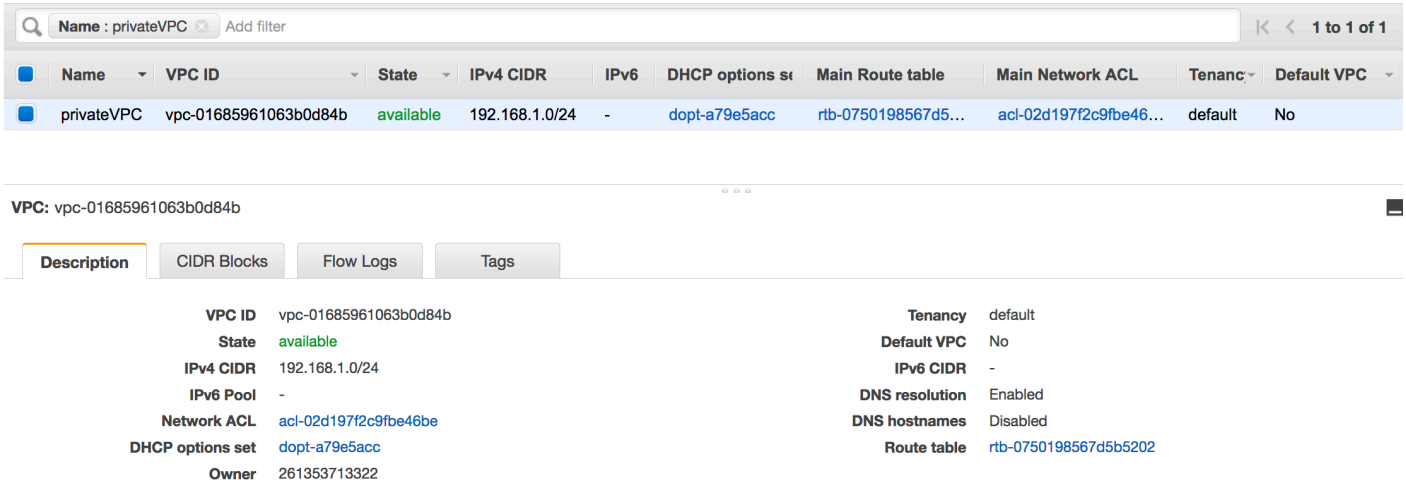
- the section called “Prasyarat”
- the section called “Membuat koneksi ke Amazon S3”
- the section called “Menguji koneksi ke Amazon S3”
- the section called “Membuat crawler untuk penyimpanan data Amazon S3”
- the section called “Menjalankan crawler”

Prasyarat

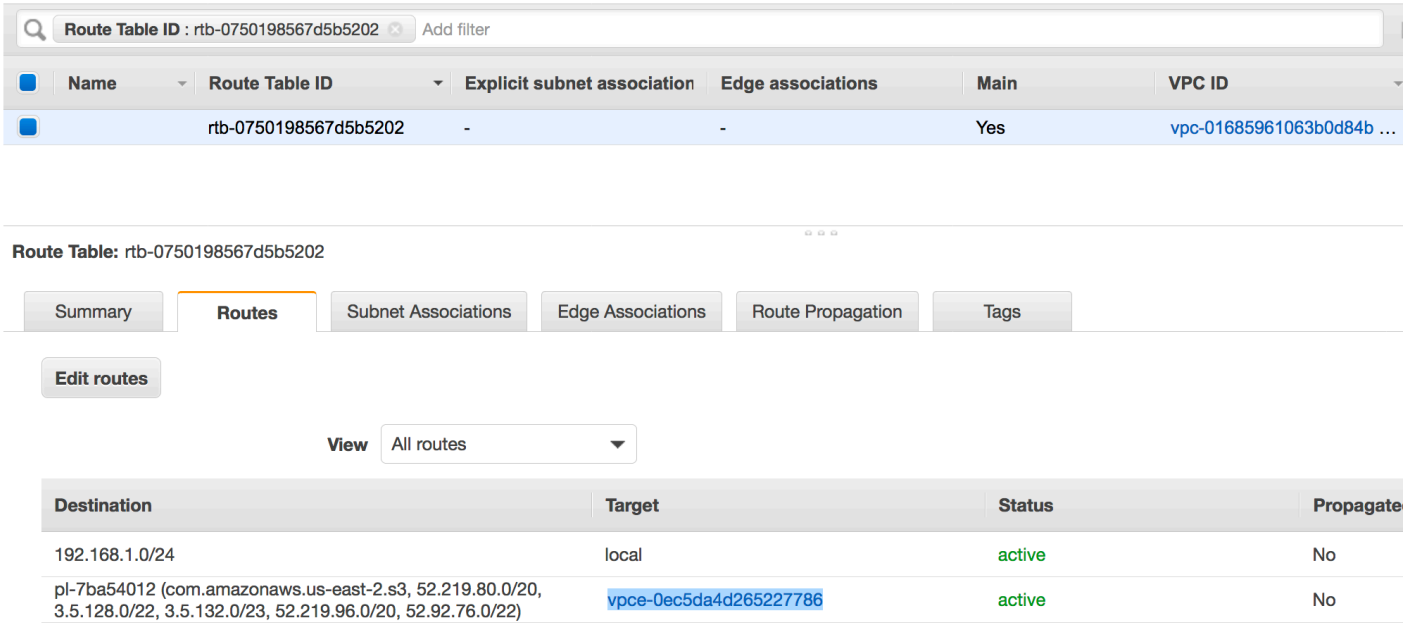
Periksa apakah Anda telah memenuhi prasyarat ini untuk menyiapkan penyimpanan data Amazon S3 atau tabel Katalog Data yang didukung Amazon S3 untuk diakses melalui lingkungan Amazon Virtual Private Cloud (Amazon VPC).

- Sebuah VPC yang sudah dikonfigurasi. Sebagai contoh: vpc-01685961063b0d84b. Untuk informasi selengkapnya, lihat Memulai dengan Amazon VPC dalam Panduan Pengguna Amazon VPC.

- Titik akhir Amazon S3 yang dilampirkan pada VPC. Sebagai contoh: vpc-01685961063b0d84b. Untuk informasi selengkapnya, lihat Titik Akhir untuk Amazon S3 di Panduan Pengguna Amazon VPC.



- Sebuah entri rute yang mengarahkan ke VPC endpoint. Misalnya vpce-0ec5da4d265227786 di tabel rute yang digunakan oleh VPC endpoint (vpce-0ec5da4d265227786).



- ACL jaringan yang dilampirkan ke VPC mengizinkan lalu lintas.
- Sebuah grup keamanan yang dilampirkan pada VPC yang mengizinkan lalu lintas.

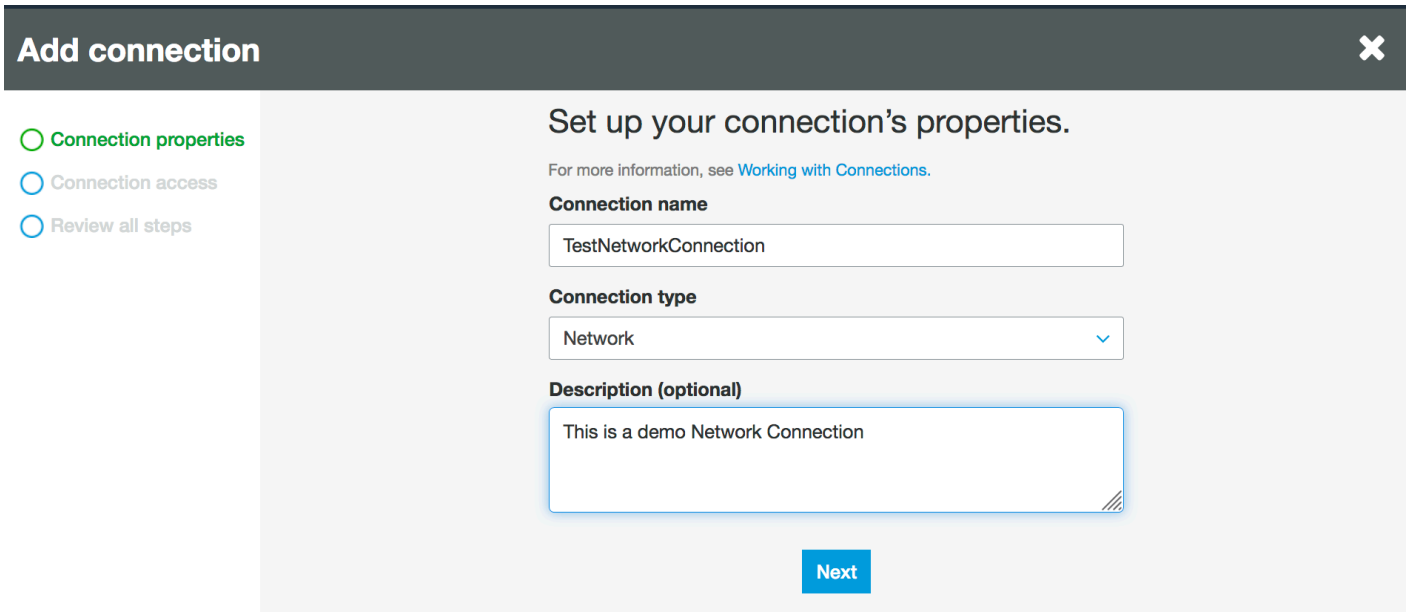
Membuat koneksi ke Amazon S3

Biasanya, Anda membuat sumber daya di dalam Amazon Virtual Private Cloud (Amazon VPC) sehingga sumber daya tidak dapat diakses melalui internet publik. Secara default, AWS Glue tidak dapat mengakses sumber daya di dalam VPC. Untuk memungkinkan AWS Glue mengakses sumber daya di dalam VPC Anda, Anda harus memberikan informasi konfigurasi spesifik-VPC yang mencakup ID subnet VPC dan ID grup keamanan. Untuk membuat koneksi `Network`, Anda harus menentukan informasi berikut:

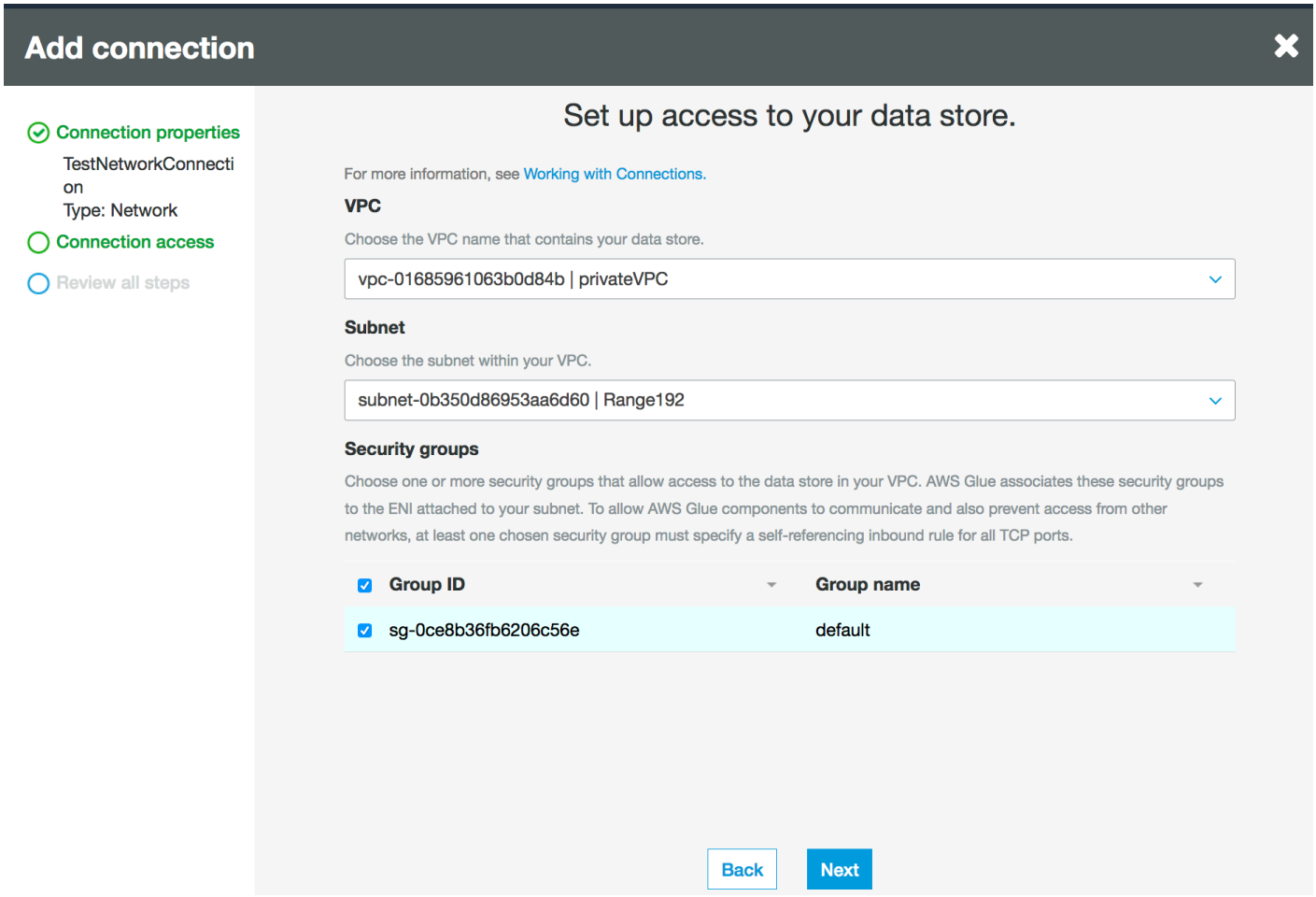
- ID VPC
- Subnet dalam VPC
- Grup keamanan

Untuk mengatur koneksi `Network`:

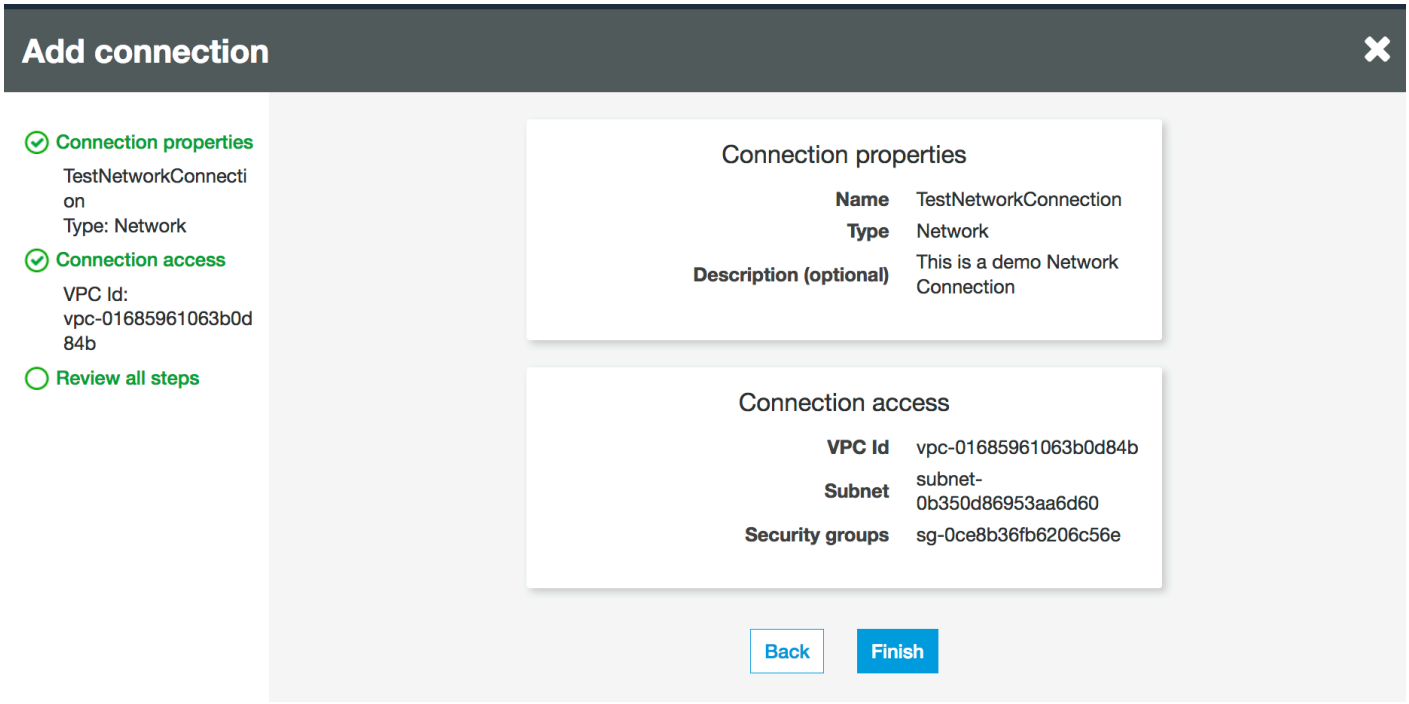
1. Pilih Tambahkan koneksi di panel navigasi konsol AWS Glue.
2. Masukkan nama koneksi, pilih Jaringan sebagai jenis koneksi. Pilih Selanjutnya.



3. Mengkonfigurasi informasi VPC, Subnet dan Grup Keamanan.
   - VPC: pilih nama VPC yang berisi penyimpanan data Anda.
   - Subnet: pilih subnet dalam VPC Anda.
   - Grup keamanan: Pilih satu atau beberapa grup keamanan yang memungkinkan akses ke penyimpanan data di VPC Anda.



4. Pilih Selanjutnya.
5. Verifikasi informasi koneksi dan pilih Selesai.



Menguji koneksi ke Amazon S3

Setelah Anda telah membuat koneksi `Network` Anda, Anda dapat menguji konektivitas ke penyimpanan data Amazon S3 Anda di VPC endpoint.

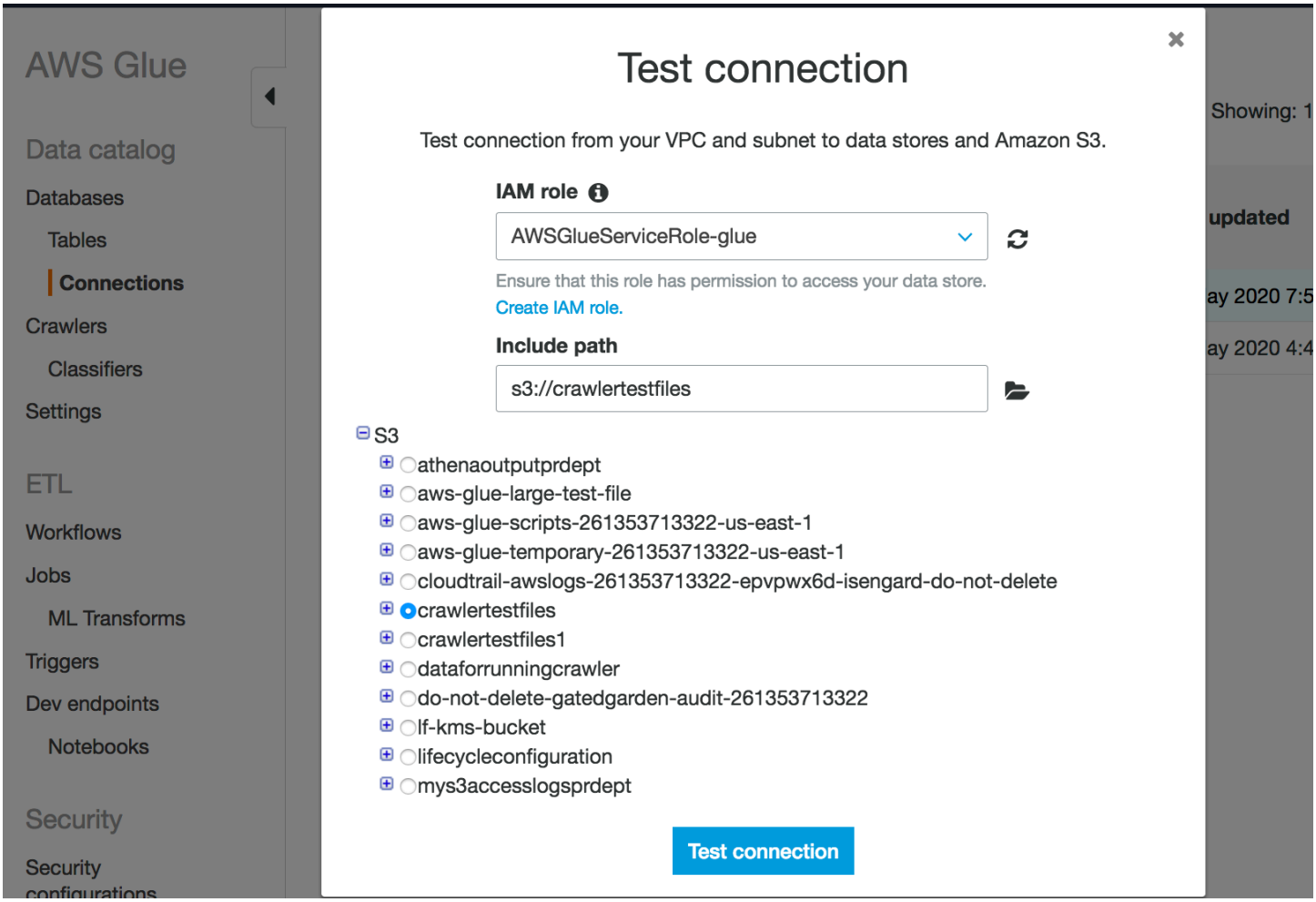
Kesalahan berikut mungkin terjadi saat menguji sebuah koneksi:

- INTERNET CONNECTION ERROR: menunjukkan masalah koneksi Internet
- INVALID BUCKET ERROR: menunjukkan masalah dengan bucket Amazon S3
- S3 CONNECTION ERROR: menunjukkan kegagalan untuk connect ke Amazon S3
- INVALID CONNECTION TYPE: menunjukkan jenis koneksi tidak memiliki nilai yang diharapkan, `NETWORK`
- INVALID CONNECTION TEST TYPE: menunjukkan masalah dengan jenis pengujian koneksi jaringan
- INVALID TARGET: menunjukkan bahwa bucket Amazon S3 belum ditentukan dengan benar

Untuk menguji sebuah koneksi `Network`:

1. Pilih koneksi Jaringan di konsol AWS Glue.
2. Pilih Uji koneksi .

3. Pilih IAM role yang Anda buat di langkah sebelumnya dan tentukan sebuah bucket Amazon S3.
4. Pilih Uji koneksi untuk memulai pengujian. Mungkin perlu beberapa saat untuk menunjukkan hasilnya.



Jika Anda menerima kesalahan, periksa apakah:

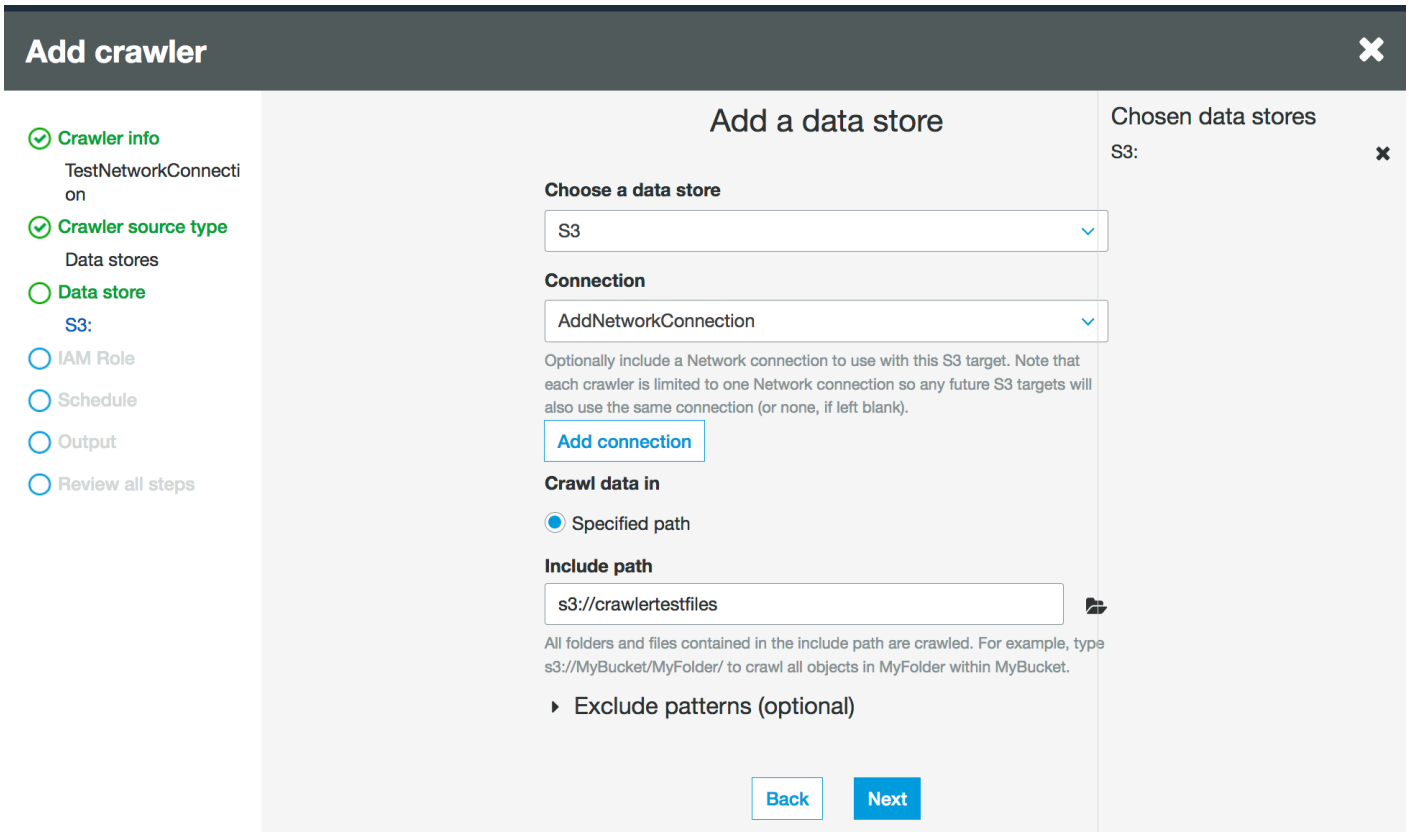
- Hak istimewa yang benar sudah disediakan untuk peran yang dipilih.
- Bucket Amazon S3 yang benar sudah disediakan.
- Grup keamanan dan ACL jaringan memungkinkan lalu lintas masuk dan keluar yang diperlukan.
- VPC yang Anda tentukan terhubung ke VPC endpoint Amazon S3.

Setelah Anda berhasil menguji koneksi, Anda dapat membuat sebuah crawler.

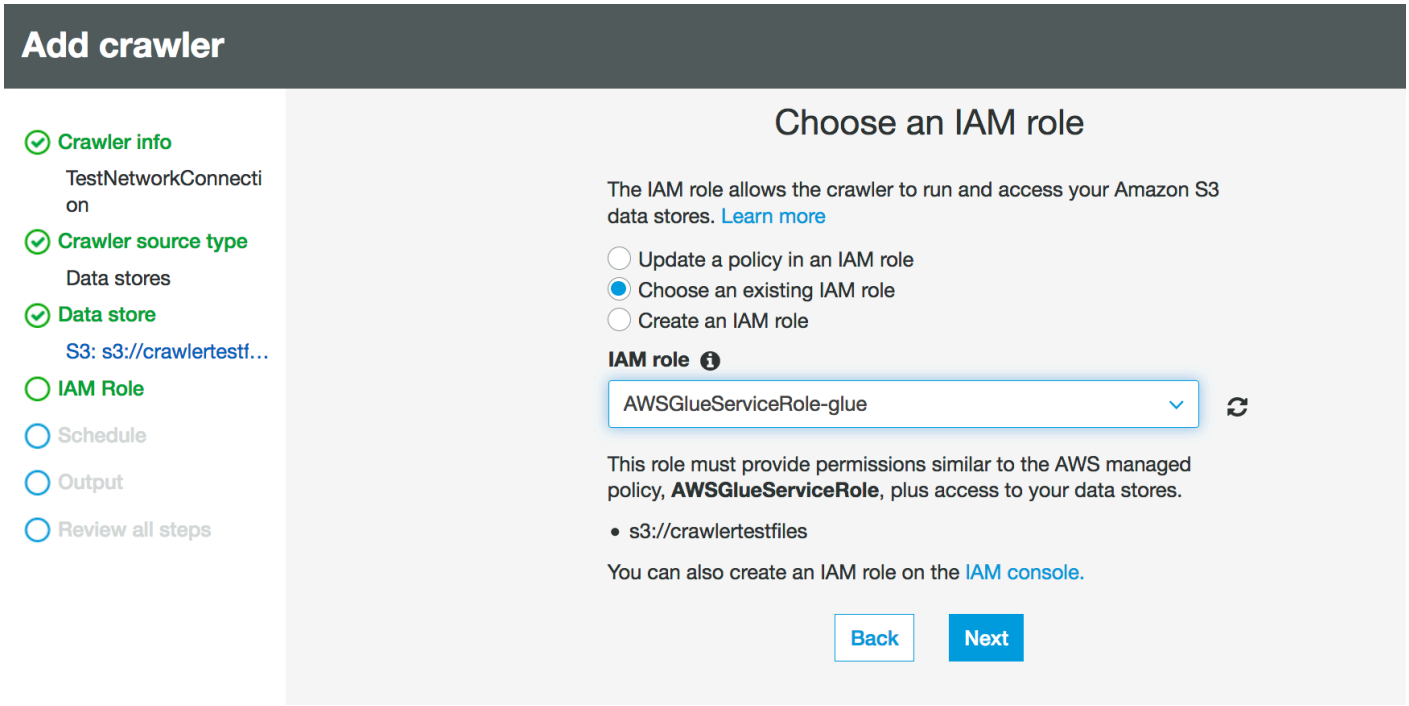
Membuat crawler untuk penyimpanan data Amazon S3

Sekarang Anda dapat membuat sebuah crawler yang menentukan koneksi `Network` yang telah Anda buat. Untuk detail selengkapnya tentang cara membuat crawler, lihat Mengkonfigurasi crawler.

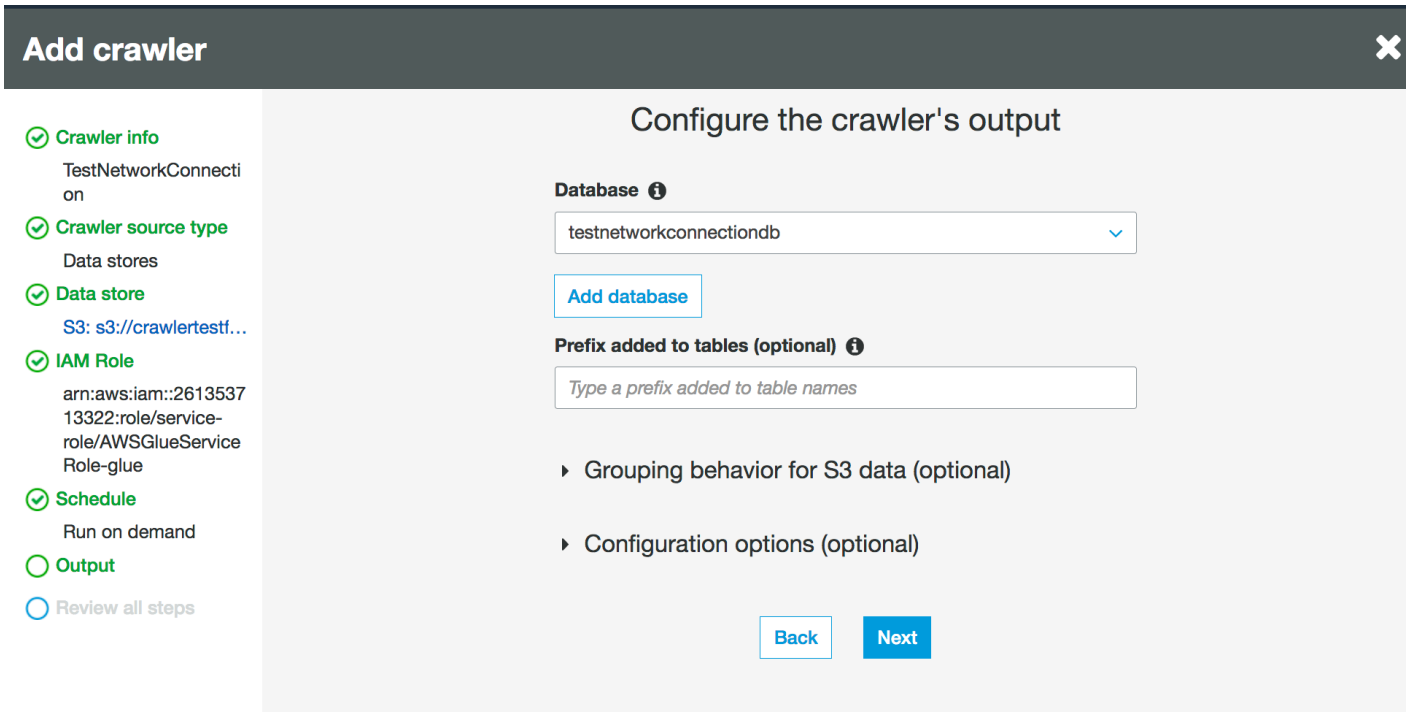
1. Mulailah dengan memilih Crawler di panel navigasi di konsol AWS Glue.
2. Pilih Tambahkan crawler.
3. Tentukan nama crawler dan pilih Selanjutnya.
4. Saat diminta sumber data, pilih S3, dan tentukan prefiks bucket Amazon S3 dan koneksi yang Anda buat sebelumnya.



5. Jika Anda perlu, tambahkan penyimpanan data yang lain pada koneksi jaringan yang sama.
6. Pilih IAM role. IAM role harus mengizinkan akses ke layanan dan bucket Amazon S3 AWS Glue. Untuk informasi selengkapnya, lihat the section called “Mengkonfigurasi crawler”.



7. Tentukan jadwal untuk crawler tersebut.
8. Pilih basis data yang ada di Katalog Data, atau buatlah entri basis data baru.

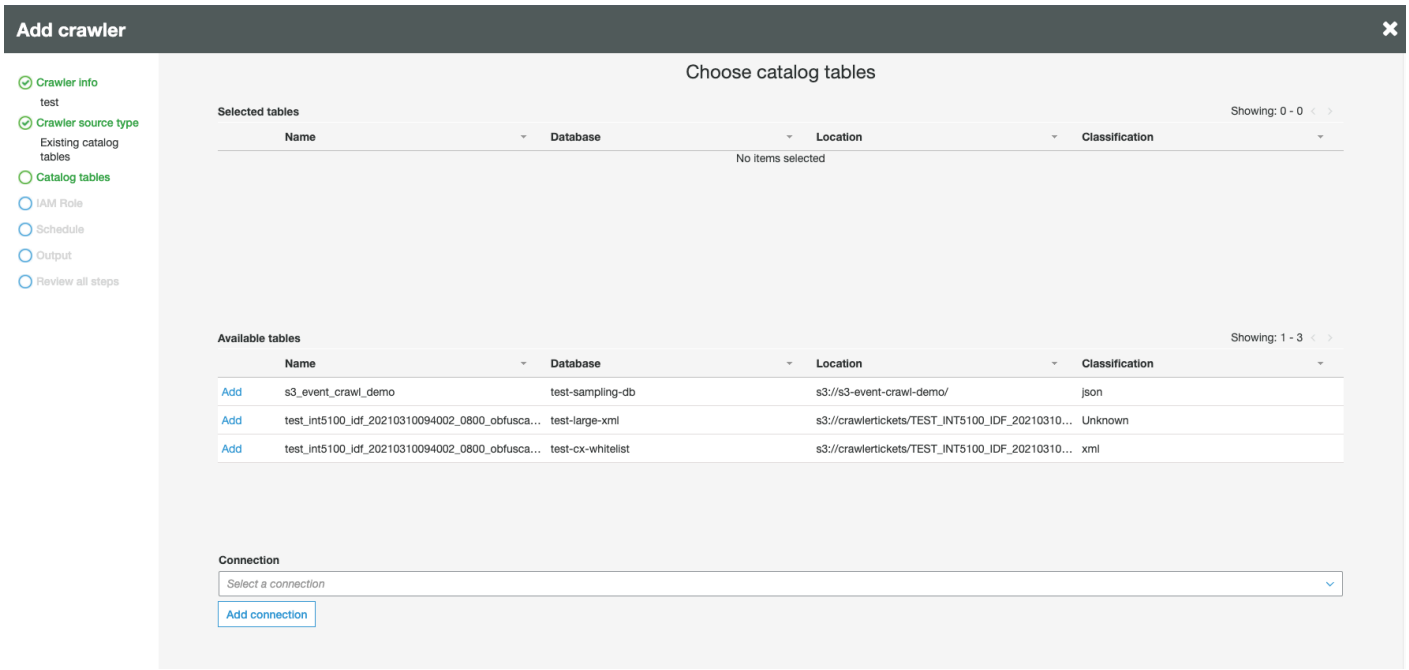


9. Selesaikan pengaturan yang tersisa.

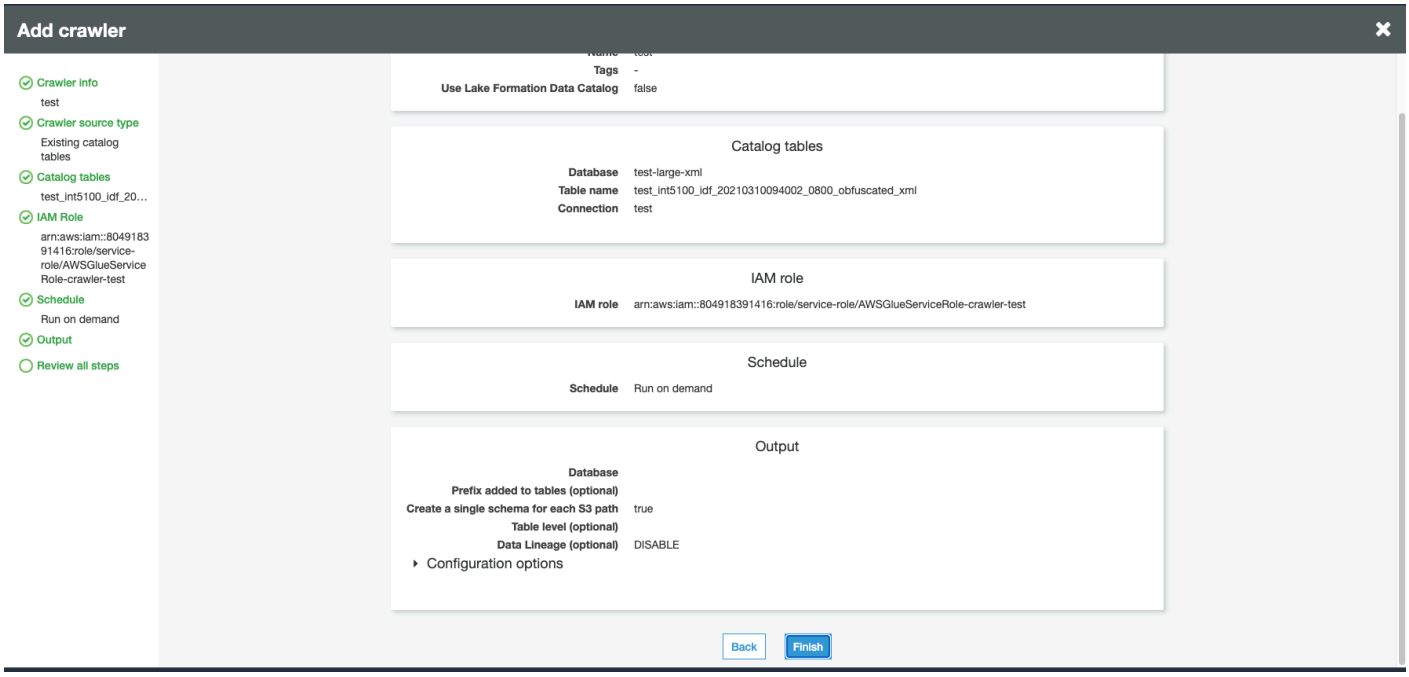
Membuat crawler untuk tabel Katalog Data yang didukung Amazon S3

Sekarang Anda dapat membuat crawler yang menentukan `Network` koneksi yang telah Anda buat dan jenis sumber Katalog. Untuk detail selengkapnya tentang cara membuat crawler, lihat Mengkonfigurasi crawler.

1. Mulailah dengan memilih Crawler di panel navigasi di konsol AWS Glue.
2. Pilih Tambahkan crawler.
3. Tentukan nama crawler dan pilih Selanjutnya.
4. Saat diminta jenis sumber crawler, pilih Tabel katalog yang ada, dan tentukan tabel katalog yang ada untuk dirayapi dari daftar tabel yang tersedia.

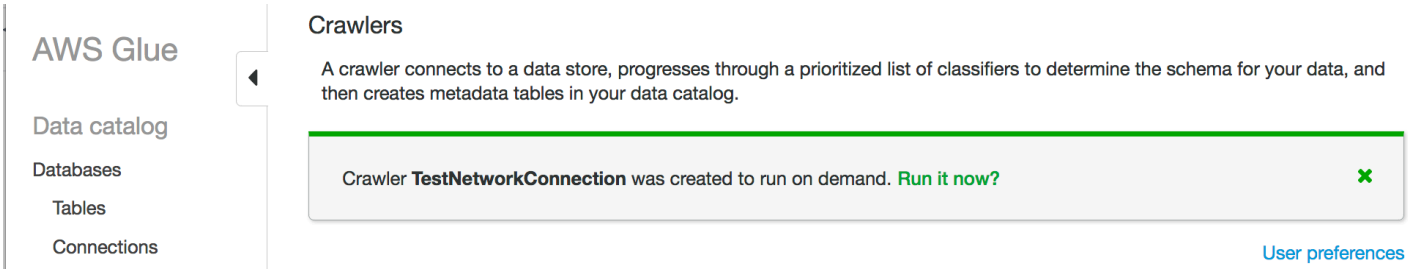


5. Pilih IAM role. IAM role harus mengizinkan akses ke layanan dan bucket Amazon S3 AWS Glue. Untuk informasi selengkapnya, lihat the section called “Mengkonfigurasi crawler”.
6. Tentukan jadwal untuk crawler tersebut.
7. Pilih basis data yang ada di Katalog Data, atau buatlah entri basis data baru.
8. Selesaikan pengaturan yang tersisa dan tinjau langkah-langkah Anda.



Menjalankan crawler

Jalankan crawler Anda.



Pemecahan Masalah

Untuk pemecahan masalah yang berkaitan dengan bucket Amazon S3 menggunakan gateway VPC, lihat Mengapa saya tidak dapat connect ke bucket S3 menggunakan gateway VPC endpoint?

Memecahkan masalah koneksi di AWS Glue

Saat crawler AWS Glue atau tugas menggunakan properti koneksi untuk mengakses penyimpanan data, Anda mungkin mengalami kesalahan saat mencoba connect. AWS Glue menggunakan alamat IP privat di subnet saat membuat antarmuka jaringan elastis di virtual private cloud (VPC) dan subnet yang ditentukan. Grup keamanan yang ditentukan dalam koneksi digunakan pada setiap antara muka

jaringan elastis. Periksa untuk melihat apakah grup keamanan memungkinkan akses keluar dan apakah mengizinkan konektivitas ke klaster basis data.

Selain itu, Apache Spark membutuhkan konektivitas dua arah antara node driver dan executor. Salah satu grup keamanan perlu untuk mengizinkan masuknya aturan ingress pada semua port TCP. Anda dapat mencegahnya terbuka ke seluruh dunia dengan membatasi sumber grup keamanan itu sendiri dengan grup keamanan referensi mandiri.

Berikut adalah beberapa tindakan umum yang dapat dilakukan untuk memecahkan masalah pada koneksi:

- Periksa alamat port koneksi Anda.
- Periksa nama pengguna dan string kata sandi di koneksi atau rahasia Anda.
- Untuk menyimpan data JDBC, verifikasi apakah hal itu memungkinkan koneksi masuk.
- Verifikasi apakah penyimpanan data Anda dapat diakses dalam VPC Anda.
- Jika Anda menyimpan kredenal koneksi Anda menggunakanAWS Secrets Manager, pastikan bahwa peran IAM Anda AWS Glue memiliki izin untuk mengakses rahasia Anda. Untuk informasi selengkapnya, lihat Contoh: Izin untuk mengambil nilai rahasia di Panduan AWS Secrets Manager Pengguna. Bergantung pada pengaturan jaringan Anda, Anda mungkin juga perlu membuat titik akhir VPC untuk membuat koneksi pribadi antara VPC dan Secrets Manager. Untuk informasi selengkapnya, lihat Menggunakan titik AWS Secrets Manager akhir VPC.

Tutorial: Menggunakan AWS Glue Konektor untuk Elasticsearch

Elasticsearch adalah mesin pencarian dan analitik sumber terbuka populer untuk kasus penggunaan seperti analitik log, pemantauan aplikasi waktu nyata, dan analisis clickstream. Anda dapat menggunakan OpenSearch sebagai penyimpanan data untuk pekerjaan ekstrak, transformasi, dan pemuatan (ETL) Anda dengan mengonfigurasi AWS Glue Konektor untuk Elasticsearch di. AWS Glue Studio Konektor ini tersedia secara gratis dari AWS Marketplace.

> **Note**
>
> Konektor Spark AWS Marketplace Elasticsearch tidak digunakan lagi. Silakan gunakan AWS GlueKonektor untuk Elasticsearch sebagai gantinya.

Dalam tutorial ini, kami akan menunjukkan cara menghubungkan ke node OpenSearch Layanan Amazon Anda dengan jumlah langkah minimal.

Topik

Prasyarat

Untuk menggunakan tutorial ini, Anda harus memiliki hal-hal berikut ini:

- Akses ke AWS Glue Studio
- Akses ke OpenSearch klaster di AWS Cloud
- (Opsional) Akses ke AWS Secrets Manager.

Langkah 1: (Opsional) Buat AWS rahasia untuk informasi OpenSearch cluster Anda

Untuk menyimpan dan menggunakan kredensial koneksi Anda dengan aman, simpan kredensial di AWS Secrets Manager. Rahasia yang Anda buat akan digunakan nanti dalam tutorial berdasarkan koneksi. Pasangan nilai kunci kredenal akan dimasukkan ke AWS Glue Konektor untuk Elasticsearch sebagai opsi koneksi normal.

Untuk informasi selengkapnya tentang membuat rahasia, lihat Membuat dan Mengelola Rahasia dengan AWS Secrets Manager di Panduan Pengguna AWS Secrets Manager.

Untuk membuat sebuah rahasia AWS baru

1. Masuk ke konsol AWS Secrets Manager tersebut.
2. Pada halaman pengenalan layanan atau halaman daftar Rahasia, pilih Menyimpan rahasia baru.

3. Pada halaman Menyimpan rahasia baru, pilih Jenis rahasia lainnya. Pilihan ini berarti Anda harus menyediakan struktur dan detail rahasia Anda.
4. Tambahkan pasangan Kunci dan Nilai untuk nama pengguna OpenSearch cluster. Misalnya:

   `es.net.http.auth.user`: *`nama pengguna`*
5. Pilih + Tambahkan baris, dan masukkan pasangan nilai-kunci lain untuk kata sandi. Misalnya:

   `es.net.http.auth.pass`: *`kata sandi`*
6. Pilih Selanjutnya.
7. Masukkan nama rahasia. Misalnya: my-es-secret. Anda dapat menyertakan sebuah deskripsi.

   Catat nama rahasia, yang akan digunakan nanti dalam tutorial ini, dan kemudian pilih Selanjutnya.
8. Pilih Selanjutnya lagi, lalu pilih Menyimpan untuk menciptakan rahasia.

Langkah selanjutnya

Langkah 2: Berlangganan ke konektor

Langkah 2: Berlangganan ke konektor

AWS GlueKonektor untuk Elasticsearch tersedia secara gratis. AWS Marketplace

Untuk berlangganan AWS Glue Konektor untuk Elasticsearch di AWS Marketplace

1. Jika Anda belum mengkonfigurasi akun AWS Anda untuk menggunakan License Manager, lakukan hal berikut:

   a. Buka AWS License Manager konsol di https://console.aws.amazon.com/license-manager.

   b. Pilih Buat lisensi terkelola pelanggan.

   c. Di jendela Izin IAM (pengaturan satu kali), pilih Saya memberikan izin AWS License Manager yang diperlukan, lalu pilih Berikan izin.

      Jika Anda tidak melihat jendela ini, maka berarti Anda telah mengkonfigurasi izin yang diperlukan.
2. Buka AWS Glue Studio konsol di https://console.aws.amazon.com/gluestudio/.

3. Di AWS Glue Studio konsol, perluas ikon menu (☰), lalu pilih Konektor di panel navigasi.
4. Pada halaman Konektor, pilih Buka AWS Marketplace.
5. DiAWS Marketplace, di bagian Cari AWS Glue Studio produk, masukkan AWS GlueKonektor untuk Elasticsearch di bidang pencarian, lalu tekan Enter.
6. Pilih nama konektor, AWS GlueKonektor untuk Elasticsearch.
7. Pada halaman produk untuk konektor, gunakan tab untuk melihat informasi tentang konektor tersebut. Saat Anda siap melanjutkan, pilih Lanjutkan ke Berlangganan.
8. Tinjau ketentuan penggunaan untuk perangkat lunak. Klik Terima Ketentuan.
9. Ketika proses berlangganan selesai, Anda akan melihat pemberitahuan: “Terima kasih telah berlangganan produk ini! Anda sekarang dapat mengkonfigurasi perangkat lunak Anda.” Di atas spanduk akan ada tombol Lanjutkan ke Konfigurasi. Pilih Lanjutkan ke Konfigurasi.
10. Pilih opsi Pemenuhan pada halaman Konfigurasikan perangkat lunak ini. Anda dapat memilih antara AWS Glue 1.0/2.0 atau AWS Glue 3.0. Kemudian, pilih Lanjutkan untuk Meluncurkan.

Langkah selanjutnya

Langkah 3: Aktifkan konektor AWS Glue Studio dan buat koneksi

Langkah 3: Aktifkan konektor AWS Glue Studio dan buat koneksi

Setelah Anda memilih Lanjutkan ke Peluncuran, Anda akan melihat halaman Luncurkan perangkat lunak ini di AWS Marketplace. Setelah Anda menggunakan tautan untuk mengaktifkan konektorAWS Glue Studio, Anda membuat koneksi.

Untuk menyebarkan konektor dan membuat koneksi di AWS Glue Studio

1. Pada halaman Luncurkan perangkat lunak ini di konsol AWS Marketplace, pilih Instruksi Penggunaan, lalu pilih tautan di jendela yang muncul.

   Browser Anda dialihkan ke AWS Glue Studio konsol Buat halaman koneksi marketplace.
2. Masukkan nama untuk koneksi tersebut. Misalnya: my-es-connection.
3. Di bagian Akses koneksi, untuk Jenis kredensial koneksi, pilih Nama pengguna dan kata sandi.
4. Untuk Rahasia AWS, masukkan nama rahasia Anda. Misalnya: my-es-secret.
5. Di bagian Opsi jaringan, masukkan informasi VPC untuk terhubung ke OpenSearch cluster.

6. Pilih Buat koneksi dan aktifkan konektor.

Langkah selanjutnya

Langkah 4: Mengkonfigurasi IAM role untuk tugas ETL Anda

Langkah 4: Mengkonfigurasi IAM role untuk tugas ETL Anda

Saat Anda membuat tugas ETL AWS Glue, Anda menentukan AWS Identity and Access Management (IAM) role yang akan digunakan oleh tugas tersebut. Peran itu harus memberikan akses ke semua sumber daya yang digunakan oleh tugas, termasuk Amazon S3 (untuk sumber, target, skrip, file driver, dan direktori sementara), dan juga objek AWS Glue Data Catalog.

IAM role yang diambil untuk tugas ETL AWS Glue juga harus memiliki akses ke rahasia yang dibuat di bagian sebelumnya. Secara default, peran terkelola AWS `AWSGlueServiceRole` tidak memiliki akses ke rahasia. Untuk mengatur kendali akses ke rahasia Anda, lihat Autentikasi dan Kendali Akses AWS Secrets Manager dan Membatasi Akses ke Rahasia Tertentu.

Untuk mengkonfigurasi IAM role untuk tugas ETL Anda

1. Konfigurasikan izin sebagaimana yang dijelaskan di the section called “Tinjau izin IAM yang diperlukan untuk pekerjaan ETL”.
2. Konfigurasikan izin tambahan yang diperlukan saat menggunakan konektor denganAWS Glue Studio, seperti yang dijelaskan dalamthe section called “Izin diperlukan untuk menggunakan konektor”.

Langkah selanjutnya

Langkah 5: Buat pekerjaan yang menggunakan OpenSearch koneksi

Langkah 5: Buat pekerjaan yang menggunakan OpenSearch koneksi

Setelah membuat peran untuk pekerjaan ETL Anda, Anda dapat membuat pekerjaan AWS Glue Studio yang menggunakan koneksi dan konektor untuk Open Spark ElasticSearch.

Jika tugas Anda berjalan di Amazon Virtual Private Cloud (Amazon VPC), pastikan VPC tersebut dikonfigurasi dengan semestinya. Untuk informasi selengkapnya, lihat the section called “Konfigurasikan VPC untuk pekerjaan ETL Anda”.

Untuk membuat sebuah tugas yang menggunakan Elasticsearch Spark Connector

1. DiAWS Glue Studio, pilih Konektor.
2. Di daftar Koneksi Anda, pilih koneksi yang baru saja Anda buat dan pilih Buat tugas.
3. Dalam editor tugas visual, pilih Simpul sumber data. Di sebelah kanan, di tab Properti sumber data - Konektor, konfigurasi informasi tambahan untuk konektor.

   a. Pilih Tambahkan skema dan masukkan skema set data dalam sumber data. Koneksi tidak menggunakan tabel yang disimpan dalam Katalog Data, yang AWS Glue Studio berarti tidak mengetahui skema data. Anda harus memberikan informasi skema ini secara manual. Untuk petunjuk tentang cara menggunakan editor skema, lihat the section called “Mengedit skema di simpul transformasi kustom”.

   b. Perluas Opsi koneksi.

   c. Pilih Tambahkan opsi baru dan masukkan informasi yang diperlukan untuk konektor yang tidak dimasukkan dalam rahasia AWS:

      - es.nodes: https://< OpenSearch titik akhir domain>
      - es.port: 443
      - jalur: uji
      - es.nodes.wan.only: benar

      Untuk penjelasan tentang pilihan koneksi ini, lihat https://www.elastic.co/guide/en/elasticsearch/hadoop/current/configuration.html.

4. Tambahkan simpul target ke grafik.

   Target data Anda bisa berupa Amazon S3, atau dapat menggunakan formulir informasi dari AWS Glue Data Catalog atau konektor untuk menulis data di lokasi yang berbeda. Sebagai contoh, Anda dapat menggunakan tabel Katalog Data untuk menulis ke basis data di Amazon RDS, atau Anda dapat menggunakan konektor sebagai target data Anda untuk menulis ke penyimpanan data yang didukung secara asli di AWS Glue.

   Jika Anda memilih sebuah konektor untuk target data Anda, maka Anda harus memilih sebuah koneksi yang dibuat untuk konektor tersebut. Selain itu, jika diperlukan oleh penyedia konektor, Anda harus menambahkan opsi untuk memberikan informasi tambahan ke konektor. Jika Anda menggunakan sambungan yang berisi informasi untuk rahasia AWS, maka Anda tidak perlu memberikan nama pengguna dan autentikasi kata sandi dalam pilihan koneksi.

5. Opsional, tambahkan sumber data tambahan dan satu atau beberapa simpul transformasi seperti yang dijelaskan di the section called “Mengedit node transformasi data AWS Glue terkelola”.
6. Konfigurasikan properti tugas seperti yang dijelaskan di the section called “Mengubah properti tugas”, dimulai dengan langkah 3, dan simpan tugas.

Langkah selanjutnya

Langkah 6: Menjalankan tugas

Langkah 6: Menjalankan tugas

Setelah Anda menyimpan tugas Anda, Anda dapat menjalankan tugas untuk melakukan operasi ETL.

Untuk menjalankan pekerjaan yang Anda buat untuk AWS Glue Connector for Elasticsearch

1. Menggunakan AWS Glue Studio konsol, pada halaman editor visual, pilih Jalankan.
2. Dalam banner keberhasilan, pilih Detail Eksekusi, atau Anda dapat memilih tab Eksekusi di editor visual untuk melihat informasi tentang eksekusi tugas.

Membangun AWS Glue pekerjaan dengan sesi interaktif

Insinyur data dapat membuat AWS Glue pekerjaan lebih cepat dan lebih mudah daripada sebelum menggunakan sesi interaktifAWS Glue.

Topik

Ikhtisar sesi AWS Glue interaktif

Dengan sesi AWS Glue interaktif, Anda dapat dengan cepat membangun, menguji, dan menjalankan persiapan data dan aplikasi analitik. Sesi interaktif menyediakan antarmuka terprogram dan visual untuk membangun dan menguji skrip ekstrak, transformasi, dan beban (ETL) untuk persiapan data. Sesi interaktif menjalankan aplikasi analitik Apache Spark dan menyediakan akses sesuai permintaan ke lingkungan runtime Spark jarak jauh. AWS Gluemengelola Spark tanpa server secara transparan untuk sesi interaktif ini.

Sesi interaktif fleksibel, sehingga Anda membangun dan menguji aplikasi Anda dari lingkungan pilihan Anda. Anda dapat membuat dan bekerja dengan sesi interaktif melalui AWS Command Line Interface dan API. Anda dapat menggunakan notebook yang kompatibel dengan Jupyter untuk menulis secara visual dan menguji skrip notebook Anda. Sesi interaktif menyediakan kernel Jupyter open-source yang terintegrasi hampir di mana saja yang dilakukan Jupyter, termasuk mengintegrasikan dengan IDE seperti, IntelliJ, PyCharm dan VS Code. Ini memungkinkan Anda untuk membuat kode di lingkungan lokal Anda dan menjalankannya dengan mulus di backend sesi interaktif.

Dengan menggunakan API sesi interaktif, pelanggan dapat menjalankan aplikasi secara terprogram yang menggunakan analitik Apache Spark tanpa harus mengelola infrastruktur Spark. Anda dapat menjalankan satu atau lebih pernyataan Spark dalam satu sesi interaktif.

Oleh karena itu, sesi interaktif menyediakan cara yang lebih cepat, lebih murah, dan lebih fleksibel untuk membangun dan menjalankan persiapan data dan aplikasi analitik. Untuk mempelajari cara menggunakan sesi interaktif, lihat dokumentasi di bagian ini. Sihir didukung oleh AWS Glue

Batasan

- Bookmark Job tidak didukung dalam sesi interaktif.
- Membuat pekerjaan notebook menggunakan AWS Command Line Interface tidak didukung.

Memulai dengan sesi AWS Glue interaktif

Bagian ini menjelaskan cara menjalankan sesi AWS Glue interaktif secara lokal.

Prasyarat untuk menyiapkan sesi interaktif secara lokal

Berikut ini adalah prasyarat untuk menginstal sesi interaktif:

- Versi Python yang didukung adalah 3.6 - 3.10+.
- Lihat bagian di bawah untuk petunjuk macOS/Linux dan Windows.

Menginstal Jupyter dan sesi AWS Glue interaktif Kernel Jupyter

Gunakan yang berikut ini untuk menginstal kernel secara lokal.

Perintah,`install-glue-kernels`, menginstal jupyter kernelspec untuk kernel pyspark dan spark dan juga menginstal logo di direktori kanan.

```
pip3 install --upgrade jupyter boto3 aws-glue-sessions
```

```
install-glue-kernels
```

Menjalankan Jupyter

Untuk menjalankan Jupyter Notebook, selesaikan langkah-langkah berikut.

1. Jalankan perintah berikut untuk meluncurkan Jupyter Notebook.

```
jupyter notebook
```

2. Pilih Baru, lalu pilih salah satu AWS Glue kernel untuk memulai pengkodean. AWS Glue

Mengkonfigurasi kredensyal sesi dan wilayah

Instruksi macOS/Linux

AWS Gluesesi interaktif memerlukan izin IAM yang sama dengan AWS Glue Jobs dan Dev Endpoints. Tentukan peran yang digunakan dengan sesi interaktif dalam salah satu dari dua cara:

1. Dengan `%iam_role` dan `%region` sihir
2. Dengan baris tambahan di `~/.aws/config`

Mengkonfigurasi peran sesi dengan sihir

Di sel pertama, ketik `%iam_role <YourGlueServiceRole>` sel pertama yang dieksekusi.

Mengkonfigurasi peran sesi dengan **`~/.aws/config`**

AWS GluePeran Layanan untuk sesi interaktif dapat ditentukan dalam buku catatan itu sendiri atau disimpan di samping AWS CLI konfigurasi. Jika Anda memiliki peran yang biasanya Anda gunakan dengan AWS Glue Jobs, ini akan menjadi peran itu. Jika Anda tidak memiliki peran yang Anda gunakan untuk AWS Glue pekerjaan, ikuti panduan ini, Mengonfigurasi izin IAM untukAWS Glue, untuk mengaturnya.

Untuk menetapkan peran ini sebagai peran default untuk sesi interaktif:

1. Dengan editor teks, buka`~/.aws/config`.
2. Cari profil yang Anda gunakan untukAWS Glue. Jika Anda tidak menggunakan profil, gunakan `[Default]` profil.
3. Tambahkan baris di profil untuk peran yang ingin Anda gunakan sepertiglue_role_arn=<AWSGlueServiceRole>.

4. [Opsional]: Jika profil Anda tidak memiliki set wilayah default, saya sarankan menambahkan satu dengan`region=us-east-1`, ganti `us-east-1` dengan wilayah yang Anda inginkan.
5. Simpan konfigurasi.

Untuk informasi selengkapnya, lihat Sesi interaktif dengan IAM.

Instruksi Windows

AWS Gluesesi interaktif memerlukan izin IAM yang sama dengan AWS Glue Jobs dan Dev Endpoints. Tentukan peran yang digunakan dengan sesi interaktif dalam salah satu dari dua cara:

1. Dengan `%iam_role` dan `%region` sihir
2. Dengan baris tambahan di `~/.aws/config`

Mengkonfigurasi peran sesi dengan sihir

Di sel pertama, ketik `%iam_role <YourGlueServiceRole>` sel pertama yang dieksekusi.

Mengkonfigurasi peran sesi dengan **`~/.aws/config`**

AWS GluePeran Layanan untuk sesi interaktif dapat ditentukan dalam buku catatan itu sendiri atau disimpan di samping AWS CLI konfigurasi. Jika Anda memiliki peran yang biasanya Anda gunakan dengan AWS Glue Jobs, ini akan menjadi peran itu. Jika Anda tidak memiliki peran yang Anda gunakan untuk AWS Glue pekerjaan, silakan ikuti panduan ini, Menyiapkan izin IAM untukAWS Glue, untuk mengaturnya.

Untuk menetapkan peran ini sebagai peran default untuk sesi interaktif:

1. Dengan editor teks, buka`~/.aws/config`.
2. Cari profil yang Anda gunakan untukAWS Glue. Jika Anda tidak menggunakan profil, gunakan `[Default]` profil.
3. Tambahkan baris di profil untuk peran yang ingin Anda gunakan sepertiglue_role_arn=<AWSGlueServiceRole>.
4. [Opsional]: Jika profil Anda tidak memiliki set wilayah default, saya sarankan menambahkan satu dengan`region=us-east-1`, ganti `us-east-1` dengan wilayah yang Anda inginkan.
5. Simpan konfigurasi.

Untuk informasi selengkapnya, lihat Sesi interaktif dengan IAM.

Memutakhirkan dari pratinjau sesi interaktif

Kernel ditingkatkan dengan nama baru ketika dirilis dengan versi 0.27. Untuk membersihkan versi pratinjau kernel, jalankan yang berikut ini dari terminal atau PowerShell.

> **Note**
>
> Jika Anda adalah bagian dari AWS Glue pratinjau lain yang memerlukan model layanan khusus, menghapus kernel akan menghapus model layanan khusus.

```
# Remove Old Glue Kernels
jupyter kernelspec remove glue_python_kernel
jupyter kernelspec remove glue_scala_kernel

# Remove Custom Model
cd ~/.aws/models
rm -rf glue/
```

Menggunakan sesi interaktif dengan SageMaker Studio

AWS Glue Sesi Interaktif adalah lingkungan runtime Apache Spark sesuai permintaan, tanpa server yang dapat digunakan oleh ilmuwan dan insinyur data untuk membangun, menguji, dan menjalankan persiapan data dan aplikasi analitik dengan cepat. Anda dapat memulai sesi AWS Glue interaktif dengan memulai notebook Amazon SageMaker Studio Classic.

Untuk informasi selengkapnya, lihat Mempersiapkan Data menggunakan sesi AWS Glue interaktif.

Menggunakan sesi interaktif dengan Microsoft Visual Studio Code

Prasyarat

- Instal sesi AWS Glue interaktif dan verifikasi itu berfungsi dengan Jupyter Notebook.
- Unduh dan instal Visual Studio Code dengan Jupyter. Untuk detailnya, lihat Notebook Jupyter di VS Code.

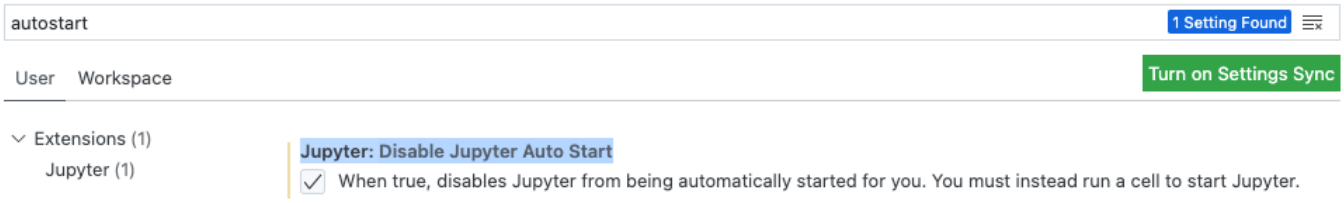
Untuk memulai sesi interaktif dengan VSCode

1. Nonaktifkan Jupyter AutoStart di VS Code.

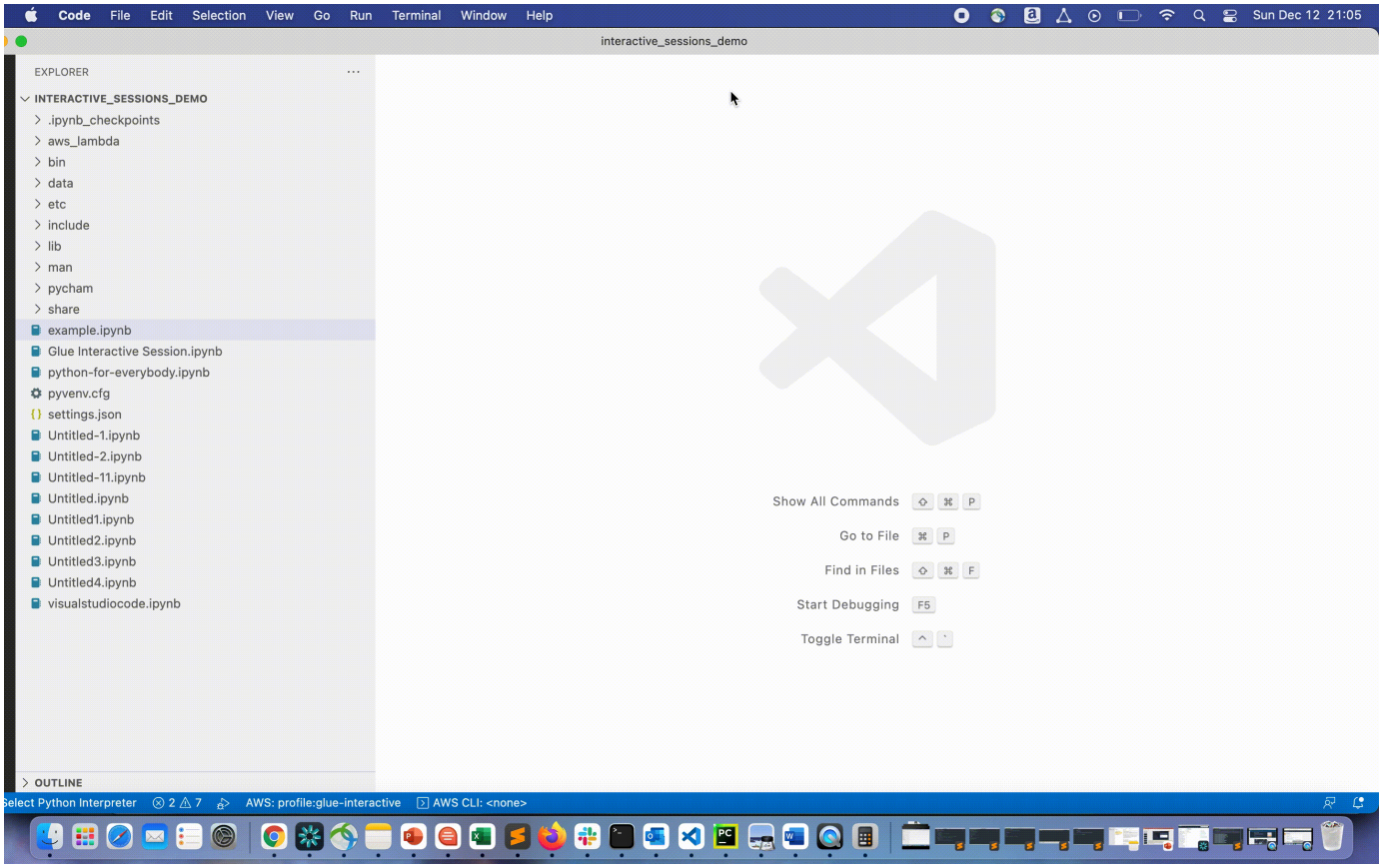
   Dalam Visual Studio Code, kernel Jupyter akan dimulai secara otomatis yang akan mencegah sihir Anda mulai berlaku karena sesi sudah dimulai. Untuk menonaktifkan Mulai Otomatis di Windows, buka File> Preferences > Extensions > Jupyter > klik kanan pada Jupyter lalu pilih Pengaturan Ekstensi.

   Di macOS, buka Kode > Pengaturan > Ekstensi> Jupyter > klik kanan pada Jupyter lalu pilih Pengaturan Ekstensi.

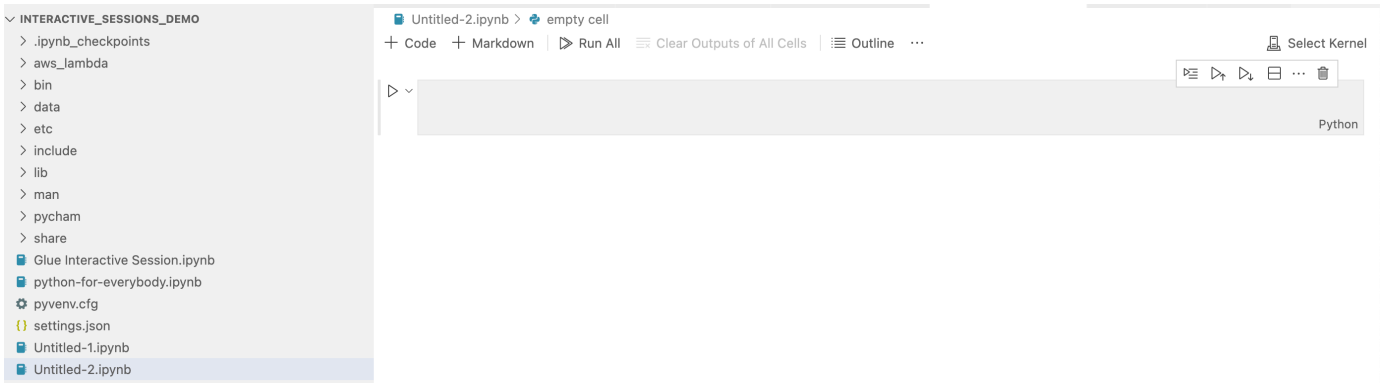
   Gulir ke bawah hingga Anda melihat Jupyter: Nonaktifkan Jupyter Auto Start. Centang kotak "Ketika benar, menonaktifkan Jupyter agar tidak dimulai secara otomatis untuk Anda. Anda harus menjalankan sel untuk memulai Jupyter."



2. Pergi ke File > File Baru > Simpan untuk menyimpan file ini dengan nama pilihan Anda sebagai `.ipynb` ekstensi atau pilih jupyter di bawah pilih bahasa dan simpan file.



3. Klik dua kali pada file tersebut. Shell Jupyter akan ditampilkan dan notebook akan dibuka.



4. Pada Windows, ketika Anda pertama kali membuat file, secara default tidak ada kernel yang dipilih. Klik Select Kernel dan daftar kernel yang tersedia ditampilkan. Pilih Glue PySpark.

   Di macOS, Jika Anda tidak melihat PySpark kernel Glue, coba langkah-langkah berikut:

   1. Jalankan sesi Jupyter lokal untuk mendapatkan URL.

      Misalnya, jalankan perintah berikut untuk meluncurkan Jupyter Notebook.

```
jupyter notebook
```

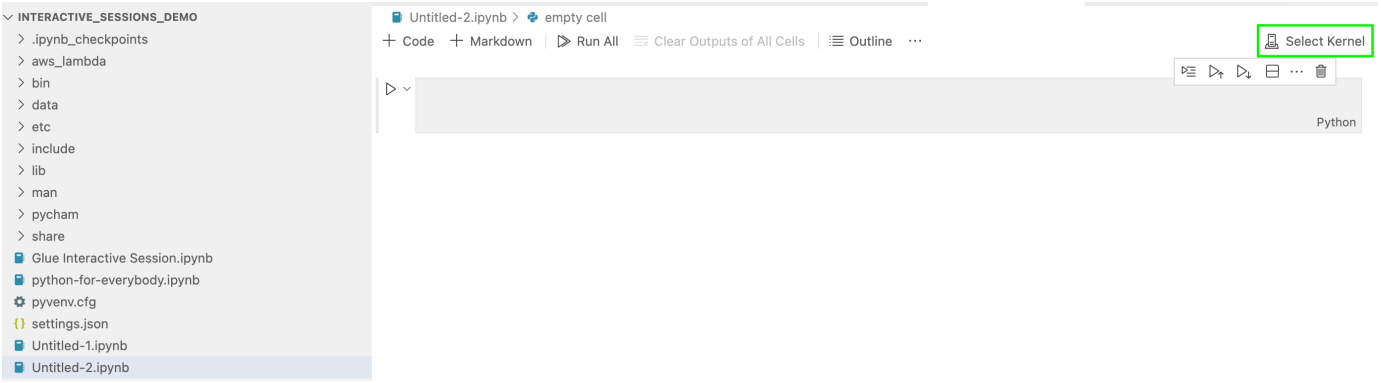
Saat notebook pertama kali berjalan, Anda akan melihat URL yang terlihat seperti`http://localhost:8888/?token=3398XXXXXXXXXXXXXXX`.

Salin URL.

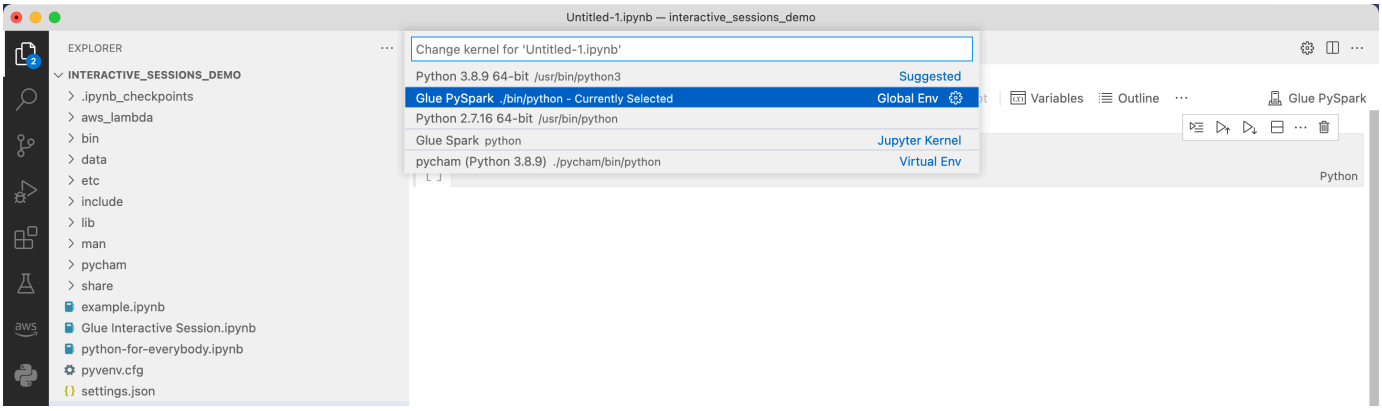
2. Di VS Code, klik kernel saat ini, lalu Pilih Kernel Lain... , lalu pilih Existing Jupyter Server... . Tempel URL yang Anda salin dari langkah di atas.

   Jika Anda menerima pesan kesalahan, lihat wiki VS Code Jupyter.

3. Jika berhasil, ini akan mengatur kernel ke Glue PySpark.



Pilih kernel Glue PySpark atau Glue Spark (masing-masing untuk Python dan Scala).



Jika Anda tidak melihat AWS Glue PySparkdan AWS GlueSpark kernel dalam daftar drop-down, pastikan Anda telah menginstal AWS Glue kernel pada langkah di atas, atau bahwa `python.defaultInterpreterPath` pengaturan Anda di Visual Studio Code sudah benar. Untuk informasi lebih lanjut, lihat python. defaultInterpreterPath deskripsi pengaturan.

5. Buat sesi AWS Glue interaktif. Lanjutkan untuk membuat sesi dengan cara yang sama seperti yang Anda lakukan di Jupyter Notebook. Tentukan sihir apa pun di bagian atas sel pertama Anda dan jalankan pernyataan kode.

Mengkonfigurasi sesi AWS Glue interaktif untuk Jupyter dan notebook AWS Glue Studio

Pengantar Jupyter Magics

Jupyter Magics adalah perintah yang dapat dijalankan di awal sel atau sebagai seluruh badan sel. Sihir dimulai dengan `%` untuk sihir garis dan untuk sihir sel. `%%` Line-magics seperti `%region` dan `%connections` dapat dijalankan dengan beberapa sihir dalam sel, atau dengan kode yang disertakan dalam badan sel seperti contoh berikut.

```
%region us-east-2
%connections my_rds_connection
dy_f = glue_context.create_dynamic_frame.from_catalog(database='rds_tables',
 table_name='sales_table')
```

Sihir sel harus menggunakan seluruh sel dan dapat memiliki rentang perintah beberapa baris. Contoh di bawah `%%sql` ini.

```
%%sql
select * from rds_tables.sales_table
```

Sihir didukung oleh sesi AWS Glue interaktif untuk Jupyter

Berikut ini adalah sihir yang dapat Anda gunakan dengan sesi AWS Glue interaktif untuk notebook Jupyter.

Sesi sihir

Nama	Tipe	Deskripsi
`%help`	T/A	Kembalikan daftar deskripsi dan jenis input untuk semua perintah ajaib.

Nama	Tipe	Deskripsi
`%profile`	String	Tentukan profil dalam AWS konfigurasi Anda untuk digunakan sebagai penyedia kredensyal.
`%region`	String	Tentukan Wilayah AWS; di mana untuk menginisialisasi sesi. Default dari `~/.aws/configure`. Contoh: `%region us-west-1`
`%idle_timeout`	Int	Jumlah menit tidak aktif setelah sesi akan batas waktu setelah sel dieksekusi. Nilai batas waktu idle default untuk sesi Spark ETL adalah batas waktu default, 2880 menit (48 jam). Untuk jenis sesi lainnya, lihat dokumentasi untuk jenis sesi tersebut. Contoh: `%idle_timeout 3000`
`%session_id`	T/A	Kembalikan ID sesi untuk sesi yang sedang berjalan.
`%session_id_prefix`	String	Tentukan string yang akan mendahulu i semua ID sesi dalam format [session_ id_prefix] - [session_id]. Jika ID sesi tidak disediakan, UUID acak akan dihasilkan. Keajaiban ini tidak didukung saat Anda menjalankan Notebook Jupyter di. AWS Glue Studio Contoh: `%session_id_prefix 001`
`%status`		Mengembalikan status AWS Glue sesi saat ini termasuk durasi, konfigurasi dan mengeksekusi pengguna/peran.
`%stop_session`		Hentikan sesi saat ini.

Nama	Tipe	Deskripsi
`%list_sessions`		Daftar semua sesi yang sedang berjalan berdasarkan nama dan ID.
`%session_type`	String	Menetapkan jenis sesi ke salah satu Streaming, ETL, atau Ray. Contoh: `%session_type Streaming`
`%glue_version`	String	Versi yang AWS Glue akan digunakan oleh sesi ini. Contoh: `%glue_version 3.0`

Sihir untuk memilih jenis pekerjaan

Nama	Tipe	Deskripsi
`%streaming`	String	Mengubah jenis sesi ke AWS Glue Streaming.
`%etl`	String	Mengubah jenis sesi ke AWS Glue ETL.
%glue_ray	String	Mengubah jenis sesi menjadi AWS Glue untuk Ray. Lihat Magics didukung oleh sesi interaktif AWS Glue Ray.

AWS Glueuntuk sihir konfigurasi Spark

`%%configure`Keajaiban adalah kamus berformat json yang terdiri dari semua parameter konfigurasi untuk sesi. Setiap parameter dapat ditentukan di sini atau melalui sihir individu.

Nama	Tipe	Deskripsi
`%%configure`	Kamus	Tentukan kamus berformat JSON yang terdiri dari semua parameter konfigura si untuk sesi. Setiap parameter dapat

Nama	Tipe	Deskripsi
		ditentukan di sini atau melalui sihir individu. Untuk daftar parameter dan contoh tentang cara menggunakan`%%configure` , lihat tabel di bawah ini: Menggunakan %%configure.
`%iam_role`	String	Tentukan peran IAM ARN untuk menjalankan sesi Anda. Default dari ~/.aws/configure. Contoh: `%iam_role AWSGlueServiceRole`
`%number_of_workers`	Int	Jumlah pekerja dari worker_type yang ditentukan yang dialokasikan saat pekerjaan berjalan. `worker_type` harus diatur juga. `number_of_workers` Defaultnya adalah 5. Contoh: `%number_of_workers 2`
`%additional_python_modules`	Daftar	Daftar modul Python tambahan yang dipisahkan koma untuk disertakan dalam cluster Anda (bisa dari PyPI atau S3). Contoh:`%additional_python_modules pandas, numpy` .

Nama	Tipe	Deskripsi

`%%tags`	String	Menambahkan tag ke sesi. Tentukan tag dalam kurung kurawal {}. Setiap pasangan nama tag diapit dalam tanda kurung (“”) dan dipisahkan dengan koma (,).
		
		`%%tags`
		`{"billing":"Data-Platform",`
		`"team":"analytics"}`
		
		Gunakan `%status` sihir untuk melihat tag yang terkait dengan sesi.
		
		`%status`
		
		`Session ID: <sessionId>`
		`Status: READY`
		`Role: <example-role>`
		`CreatedOn: 2023-05-26 11:12:17.`
		`056000-07:00`
		`GlueVersion: 3.0`
		`Job Type: glueetl`
		`Tags: {'owner':'example-owner',`
		`'team':'analytics', 'billing'`
		`:'Data-Platform'}`
		`Worker Type: G.4X`
		`Number of Workers: 5`
		`Region: us-west-2`
		`Applying the following default`
		`arguments:`
		`--glue_kernel_version 0.38.0`
		`--enable-glue-datacatalog true`
		`Arguments Passed: ['--glue_`
		`kernel_version: 0.38.0', '--`
		`enable-glue-datacatalog: true']`

Nama	Tipe	Deskripsi

%%assume_role	Kamus	Tentukan kamus berformat json atau string ARN peran IAM untuk membuat sesi untuk akses lintas akun.
		
		Contoh dengan ARN:
		
		`%%assume_role`
		`{`
		`  'arn:aws:iam::XXXXXXXXXXXX:`
		`role/AWSGlueServiceRole'`
		`}`
		
		Contoh dengan kredensyal:
		
		`%%assume_role`
		`{{`
		`  "aws_access_key_id" =`
		`"XXXXXXXXXXXX",`
		`  "aws_secret_access_key" =`
		`"XXXXXXXXXXXX",`
		`  "aws_session_token" =`
		`"XXXXXXXXXXXX"`
		`}}`

%%konfigurasi argumen sihir sel

%%configureKeajaiban adalah kamus berformat json yang terdiri dari semua parameter konfigurasi untuk sesi. Setiap parameter dapat ditentukan di sini atau melalui sihir individu. Lihat di bawah untuk contoh argumen yang didukung oleh sihir %%configure sel. Gunakan -- awalan untuk menjalankan argumen yang ditentukan untuk pekerjaan tersebut. Contoh:

```
%%configure
{
   "--user-jars-first": "true",
   "--enable-glue-datacatalog": "false"
}
```

Untuk informasi selengkapnya tentang parameter pekerjaan, lihatParameter Tugas.

Konfigurasi Sesi

Parameter	Jenis	Deskripsi

`max_retries`	Int	Jumlah waktu maksimum berapa kali percobaan yang bisa dilakukan untuk tugas ini jika gagal.
		`%%configure`
		`{`
		`  "max_retries": "0"`
		`}`

`max_concurrent_runs`	Int	Jumlah maksimum proses bersamaan yang diizinkan untuk suatu pekerjaan.
		Contoh:
		`%%configure`
		`{`
		`  "max_concurrent_runs": "3"`
		`}`

Parameter sesi

Parameter	Jenis	Deskripsi

`--enable-spark-ui`	Boolean	Aktifkan Spark UI untuk memantau dan men-debug pekerjaan AWS Glue ETL.
		`%%configure`
		`{`
		`  "--enable-spark-ui": "true"`

Parameter	Jenis	Deskripsi
		`}`

`--spark-event-logs-path`	String	Menentukan jalur Amazon S3. Saat menggunakan fitur pemantauan UI Spark.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--spark-event-logs-path":`
		`"s3://path/to/event/logs/"`
		`}`

`--script_location`	String	Menentukan jalur S3 ke skrip yang mengeksekusi pekerjaan.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "script_location": "s3://new-folder-here"`
		`}`

`--SECURITY_CONFIGURATION`	String	Nama konfigurasi AWS Glue keamanan
		
		Contoh:
		
		`%%configure`
		`{`
		`    "--SECURITY_CONFIGURATION":  security-configuration-name  ,`
		`}`

Parameter	Jenis	Deskripsi

`--job-language`	String	Bahasa pemrograman skrip. Menerima nilai 'scala' atau 'python'. Defaultnya adalah 'python'.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--job-language": "scala"`
		`}`

`--class`	String	Kelas Scala yang berfungsi sebagai titik masuk untuk skrip Scala Anda. Default adalah null.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--class": "className"`
		`}`

`--user-jars-first`	Boolean	Memprioritaskan file JAR tambahan pelanggan di classpath. Default adalah null.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--user-jars-first": "true"`
		`}`

Parameter	Jenis	Deskripsi

`--use-postgres-driver`	Boolean	Memprioritaskan driver Postgres JDBC di jalur kelas untuk menghindari konflik dengan driver JDBC. Amazon Redshift Default adalah null.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--use-postgres-driver": "true"`
		`}`

`--extra-files`	Daftar (string)	Jalur Amazon S3 ke file tambahan, seperti file konfigurasi yang AWS Glue menyalin ke direktori kerja skrip Anda sebelum menjalankannya.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--extra-files": "s3://path/to/additional/files/"`
		`}`

Parameter	Jenis	Deskripsi

`--job-bookmark-option`	String	Mengontrol perilaku bookmark pekerjaan . Menerima nilai 'job-bookmark-enable', " atau job-bookmark-disable 'job-bookmark-pause'. Defaultnya adalah job-bookmark-disable ".
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--job-bookmark-option": "job-bookmark-enable"`
		`}`

`--TempDir`	String	Menentukan jalur Amazon S3 ke bucket yang dapat digunakan sebagai direktori sementara untuk pekerjaan itu. Default adalah null.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--TempDir": "s3://path/to/temp/dir"`
		`}`

Parameter	Jenis	Deskripsi

`--enable-s3-parquet-optimized-committer`	Boolean	Mengaktifkan EMRFS Amazon S3 committer yang dioptimalkan untuk menulis data Parket ke Amazon S3. Default adalah 'benar'.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--enable-s3-parquet-optimized-committer": "false"`
		`}`

`--enable-rename-algorithm-v2`	Boolean	Menetapkan EMRFS mengubah nama algoritma versi ke versi 2. Default adalah 'benar'.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--enable-rename-algorithm-v2": "true"`
		`}`

Parameter	Jenis	Deskripsi
`--enable-glue-data catalog`	Boolean	Memungkinkan Anda menggunak an Katalog AWS Glue Data sebagai metastore Apache Spark Hive. Contoh: `%%configure { --"enable-glue-datacatalog": "true" }`
`--enable-metrics`	Boolean	Mengaktifkan pengumpulan metrik untuk pembuatan profil pekerjaan untuk menjalankan pekerjaan. Defaultnya adalah 'palsu'. Contoh: `%%configure { "--enable-metrics": "true" }`

Parameter	Jenis	Deskripsi

`--enable-continuous-cloudwatch-log`	Boolean	Mengaktifkan pencatatan berkelanjutan secara real-time untuk AWS Glue pekerjaan. Defaultnya adalah 'palsu'.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--enable-continuous-cloudwatch-log": "true"`
		`}`

`--enable-continuous-log-filter`	Boolean	Menentukan filter standar atau tidak ada filter saat Anda membuat atau mengedit pekerjaan diaktifkan untuk logging berkelanjutan. Default adalah 'benar'.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--enable-continuous-log-filter": "true"`
		`}`

Parameter	Jenis	Deskripsi

`--continuous-log-stream-prefix`	String	Menentukan awalan aliran Amazon CloudWatch log kustom untuk pekerjaan yang diaktifkan untuk logging berkelanjutan. Default adalah null.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--continuous-log-stream-prefix": "prefix"`
		`}`

`--continuous-log-conversionPattern`	String	Menentukan pola log konversi kustom untuk pekerjaan yang diaktifkan untuk logging berkelanjutan. Default adalah null.
		
		Contoh:
		
		`%%configure`
		`{`
		`  "--continuous-log-conversionPattern": "pattern"`
		`}`

Parameter	Jenis	Deskripsi

`--conf`	String	Mengontrol parameter konfigurasi Spark. Ini untuk kasus penggunaan lanjutan. Gunakan `--conf` sebelum setiap parameter. Contoh:
		`%%configure`
		`{`
		`"--conf": "spark.hadoop.hive`
		`.metastore.glue.catalogid=1`
		`23456789012 --conf hive.meta`
		`store.client.factory.class=`
		`com.amazonaws.glue.catalog.`
		`metastore.AWSGlueDataCatalo`
		`gHiveClientFactory --conf`
		`hive.metastore.schema.verif`
		`ication=false"`
		`}`

Sihir pekerjaan Spark (ETL & streaming)

Nama	Tipe	Deskripsi
`%worker_type`	String	Standar, G.1X, atau G.2X. `number_of_workers` harus diatur juga. Worker_type default adalah G.1X.

`%connections`	Daftar	Tentukan daftar koneksi yang dipisahkan koma untuk digunakan dalam sesi.
		Contoh:
		`%connections my_rds_connection`
		`dy_f =`
		`glue_context.create_dynamic`
		`_frame.from_catalog(databas`

Nama	Tipe	Deskripsi
		`e='rds_tables', table_nam e='sales_table')`
`%extra_py_files`	Daftar	Daftar terpisah koma file Python tambahan dari Amazon S3.
`%extra_jars`	Daftar	Daftar stoples tambahan yang dipisahkan koma untuk dimasukkan ke dalam cluster.
`%spark_conf`	String	Tentukan konfigurasi percikan khusus untuk sesi Anda. Misalnya, `%spark_co nf spark.serializer=o rg.apache.spark.serializer. KryoSerializer` .

Pekerjaan Magics for Ray

Nama	Tipe	Deskripsi
`%min_workers`	Int	Jumlah minimum pekerja yang dialokasi kan untuk pekerjaan Ray. Default: 1. Contoh: `%min_workers 2`
`%object_memory_head`	Int	Persentase memori bebas pada node kepala instance setelah awal yang hangat. Minimal: 0. Maksimal: 100. Contoh: `%object_memory_head 100`
%object_memory_worker	Int	Persentase memori bebas pada node pekerja instance setelah awal yang hangat. Minimal: 0. Maksimal: 100. Contoh: `%object_memory_worker 100`

Sihir aksi

Nama	Tipe	Deskripsi
`%%sql`	String	Jalankan kode SQL. Semua baris setelah `%%sql` sihir awal akan diteruskan sebagai bagian dari kode SQL. Contoh: `%%sql select * from rds_tables.sales_table`

`%matplot`	Sosok Matplotlib	Visualisasikan data Anda menggunakan pustaka matplotlib.
		
		Contoh:
		
		`import matplotlib.pyplot as plt`
		
		`# Set X-axis and Y-axis values`
		`x = [5, 2, 8, 4, 9]`
		`y = [10, 4, 8, 5, 2]`
		
		`# Create a bar chart`
		`plt.bar(x, y)`
		
		`# Show the plot`
		`%matplot plt`

`%plotly`	Sosok yang cepat	Visualisasikan data Anda menggunakan pustaka plotly.
		
		Contoh:
		
		`import plotly.express as px`
		
		`#Create a graphical figure`
		`fig = px.line(x=["a","b","c"],`
		` y=[1,3,2], title="sample`
		` figure")`

Nama	Tipe	Deskripsi
		`#Show the figure` `%plotly fig`

Sesi penamaan

AWS Gluesesi interaktif adalah AWS sumber daya dan membutuhkan nama. Nama harus unik untuk setiap sesi dan mungkin dibatasi oleh administrator IAM Anda. Untuk informasi selengkapnya, lihat Sesi interaktif dengan IAM. Kernel Jupyter secara otomatis menghasilkan nama sesi unik untuk Anda. Namun sesi dapat diberi nama secara manual dengan dua cara:

1. Menggunakan file AWS Command Line Interface konfigurasi yang terletak di`~.aws/config`. Lihat Menyiapkan AWS Config dengan file. AWS Command Line Interface
2. Menggunakan `%session_id_prefix` sihir. Lihat Sihir didukung oleh sesi AWS Glue interaktif untuk Jupyter.

Nama sesi dihasilkan sebagai berikut:

- Ketika awalan dan session_id disediakan: nama sesi akan menjadi {prefix} - {UUID}.
- Ketika tidak ada yang disediakan: nama sesi akan menjadi {UUID}.

Awalan nama sesi memungkinkan Anda mengenali sesi saat mencantumkannya di konsol AWS CLI atau.

Menentukan peran IAM untuk sesi interaktif

Anda harus menentukan peran AWS Identity and Access Management (IAM) untuk digunakan dengan kode AWS Glue ETL yang Anda jalankan dengan sesi interaktif.

Peran tersebut memerlukan izin IAM yang sama dengan yang diperlukan untuk menjalankan AWS Glue pekerjaan. Lihat Membuat peran IAM AWS Glue untuk informasi selengkapnya tentang membuat peran untuk AWS Glue pekerjaan dan sesi interaktif.

Peran IAM dapat ditentukan dalam dua cara:

- Menggunakan file AWS Command Line Interface konfigurasi yang terletak di `~.aws/config` (Disarankan). Untuk informasi selengkapnya, lihat Mengonfigurasi sesi dengan ~/.aws/config.

> **Note**
>
> Ketika `%profile` sihir digunakan, konfigurasi untuk `glue_iam_role` profil itu dihormati.

- Menggunakan sihir %iam_role. Untuk informasi selengkapnya, lihat Sihir didukung oleh sesi AWS Glue interaktif untuk Jupyter.

Mengkonfigurasi sesi dengan profil bernama

AWS Gluesesi interaktif menggunakan kredensil yang sama dengan AWS Command Line Interface atau boto3, dan sesi interaktif menghormati dan bekerja dengan profil bernama seperti yang AWS CLI ditemukan di (`~/.aws/config`Linux dan macOS) atau (Windows). `%USERPROFILE%\.aws \config` Untuk informasi selengkapnya, lihat Menggunakan profil bernama.

Sesi interaktif memanfaatkan profil bernama dengan mengizinkan Peran AWS Glue Layanan dan Awalan ID Sesi ditentukan dalam profil. Untuk mengonfigurasi peran profil, tambahkan baris untuk `iam_role` kunci dan/atau `session_id_prefix` profil bernama Anda seperti yang ditunjukkan di bawah ini. `session_id_prefix`Tidak memerlukan tanda kutip. Misalnya, jika Anda ingin menambahkan `session_id_prefix`, masukkan nilai`session_id_prefix=myprefix`.

```
[default]
region=us-east-1
aws_access_key_id=AKIAIOSFODNN7EXAMPLE
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
glue_iam_role=arn:aws:iam::<AccountID>:role/<GlueServiceRole>
session_id_prefix=<prefix_for_session_names>

[user1]
region=eu-west-1
aws_access_key_id=AKIAI44QH8DHBEXAMPLE
aws_secret_access_key=je7MtGbClwBF/2Zp9Utk/h3yCo8nvbEXAMPLEKEY
glue_iam_role=arn:aws:iam::<AccountID>:role/<GlueServiceRoleUser1>
session_id_prefix=<prefix_for_session_names_for_user1>
```

Jika Anda memiliki metode khusus untuk menghasilkan kredensil, Anda juga dapat mengonfigurasi profil Anda untuk menggunakan `credential_process` parameter dalam file Anda`~/.aws/ config`. Sebagai contoh:

```
[profile developer]
region=us-east-1
credential_process = "/Users/Dave/generate_my_credentials.sh" --username helen
```

Anda dapat menemukan detail selengkapnya tentang sumber kredensil melalui `credential_process` parameter di sini: Sumber kredensil dengan proses eksternal.

Jika suatu wilayah atau tidak `iam_role` diatur dalam profil yang Anda gunakan, Anda harus menentukannya menggunakan `%region` dan `%iam_role` sihir di sel pertama yang Anda jalankan.

Memulai sesi interaktif AWS Glue untuk Ray (pratinjau)

Warning

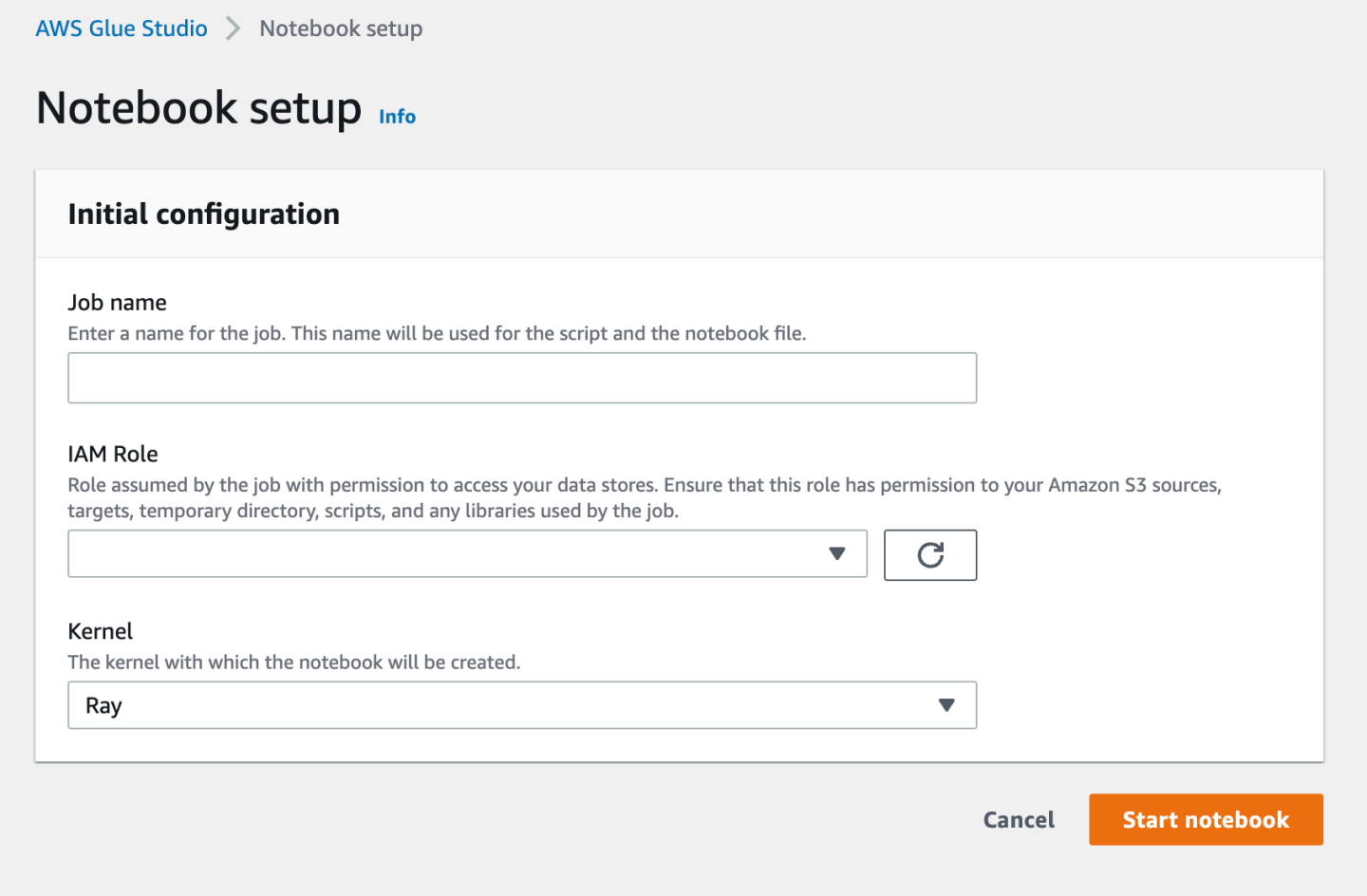
Pratinjau sesi interaktif AWS Glue for Ray berakhir 30 April 2024. Anda tidak akan lagi dapat membuat sesi interaktif baru AWS Glue untuk Ray.

Note

AWS Glue untuk Ray tersedia di AS Timur (Virginia N.), AS Timur (Ohio), AS Barat (Oregon), Asia Pasifik (Tokyo), dan Eropa (Irlandia).

Sesi interaktif Ray di AWS Glue Studio Konsol

Di halaman Pekerjaan di AWS Glue Studio Konsol, pilih opsi Jupyter Notebook yang ada. Ini akan membuka halaman pengaturan Notebook di mana Anda dapat memilih Kernel Anda. Pilih kernel Ray untuk memulai sesi interaktif Ray. Untuk informasi selengkapnya tentang sesi interaktif dan cara penggunaannya, lihatthe section called “Memulai dengan sesi AWS Glue interaktif”.



Sesi interaktif Ray menggunakan Jupyter Kernel

Untuk menggunakan Ray Kernel di luar AWS Glue Studio Konsol, Anda harus menginstal `aws-glue-sessions` paket, yang kami publikasikan di PyPI. Untuk informasi selengkapnya tentang penggunaan paket kernel, lihat the section called “Memulai dengan sesi AWS Glue interaktif” dokumentasi.

Untuk memperbarui atau menginstal kernel, jalankan`pip install --upgrade aws-glue-sessions`. Anda akan membutuhkan versi .37+ untuk menggunakan kernel Ray.

Sesi interaktif Ray memiliki akses ke perpustakaan dan versi Ray yang sama dengan pekerjaan Ray. Dalam pratinjau, semua sesi interaktif Ray akan menggunakan Ray 2.4.0.

Default batas waktu sesi interaktif Ray

- Batas waktu (untuk sesi) default: 8 jam.
- Batas Waktu Idle default: 1 jam.

Magics didukung oleh sesi interaktif AWS Glue Ray

Sihir untuk kernel AWS Glue Jupyter saat memberi daya pada sesi interaktif Ray mirip dengan sesi Spark. Untuk referensi, lihatthe section called " Mengkonfigurasi sesi AWS Glue interaktif untuk Jupyter dan notebook AWS Glue Studio".

Sesi sihir

Sihir sesi sebagian besar sama seperti sebelum pratinjau AWS Glue for Ray. Untuk informasi selengkapnya tentang sihir sesi di luar pratinjau ini, lihatthe section called "Sihir didukung oleh sesi AWS Glue interaktif untuk Jupyter". Kami memperkenalkan keajaiban baru untuk mengatur jenis sesi AWS Glue untuk Ray.

Nama	Tipe	Deskripsi
`%glue_ray`	String	Mengubah jenis sesi menjadi AWS Glue untuk Ray.

AWS Gluesihir konfigurasi

Sihir untuk mengkonfigurasi AWS Glue dalam sesi interaktif mungkin berbeda antara jenis sesi. Saat ini, kami hanya mendukung subset sihir yang ada saat menggunakan AWS Glue untuk Ray.

> **Warning**
>
> Masalah yang Diketahui: **`additional_python_modules`**
> Dalam pratinjau sesi interaktif Ray, penggunaan `additional_python_modules` sihir akan menyebabkan masalah saat menyimpan buku catatan Anda. Untuk mengonfigurasi modul python untuk sesi Ray, gunakan `%%configure` sihir untuk mengatur `pip-install` parameter yang ditentukan. the section called "Parameter pekerjaan ray"

Nama	Tipe	Deskripsi
`%%configure`	Kamus	Tentukan kamus berformat JSON yang terdiri dari semua parameter konfigura si untuk sesi. Setiap parameter dapat

Nama	Tipe	Deskripsi
		ditentukan di sini atau melalui sihir individu.
`%iam_role`	String	Tentukan peran IAM ARN untuk menjalankan sesi Anda. Default dari ~/.aws/configure
`%number_of_workers`	int	Jumlah pekerja dari worker_type yang ditentukan yang dialokasikan saat pekerjaan berjalan. `worker_type` harus diatur juga.
`%worker_type`	String	Dalam pratinjau AWS Glue for Ray, satu-satunya jenis pekerja yang didukung adalah Z.2X.
`%additional_python_modules`	Daftar	Daftar modul Python tambahan yang dipisahkan koma untuk disertakan dalam cluster Anda (bisa dari Pypi atau S3).

Sihir aksi

AWS GlueSesi Ray tidak mendukung sihir aksi apa pun.

Sesi interaktif dengan IAM

Bagian ini menjelaskan pertimbangan keamanan untuk sesi AWS Glue interaktif.

Topik

- Prinsipal IAM digunakan dengan sesi interaktif
- Menyiapkan prinsipal klien
- Menyiapkan peran runtime
- Jadikan sesi Anda pribadi dengan TagOnCreate
- Pertimbangan kebijakan IAM

Prinsipal IAM digunakan dengan sesi interaktif

Anda menggunakan dua prinsip IAM yang digunakan dengan sesi interaktif. AWS Glue

- Prinsipal klien: Prinsipal klien (baik pengguna atau peran) mengotorisasi operasi API untuk sesi interaktif dari AWS Glue klien yang dikonfigurasi dengan kredensyal berbasis identitas prinsipal. Misalnya, ini bisa menjadi peran IAM yang biasanya Anda gunakan untuk mengakses AWS Glue konsol. Ini juga bisa menjadi peran yang diberikan kepada pengguna di IAM yang kredensialnya digunakan untuk AWS Command Line Interface, atau AWS Glue klien yang digunakan oleh sesi interaktif kernel Jupyter.
- Peran runtime: Peran runtime adalah peran IAM yang diteruskan oleh prinsipal klien ke operasi API sesi interaktif. AWS Gluemenggunakan peran ini untuk menjalankan pernyataan di sesi Anda. Misalnya, peran ini bisa menjadi yang digunakan untuk menjalankan pekerjaan AWS Glue ETL.

  Untuk informasi selengkapnya, lihat Menyiapkan peran runtime.

Menyiapkan prinsipal klien

Anda harus melampirkan kebijakan identitas ke kepala klien untuk memungkinkannya memanggil API sesi interaktif. Peran ini harus memiliki `iam:PassRole` akses ke peran eksekusi yang akan Anda teruskan ke API sesi interaktif, seperti`CreateSession`. Misalnya, Anda dapat melampirkan kebijakan AWSGlueConsoleFullAccessterkelola ke peran IAM yang memungkinkan pengguna di akun Anda dengan kebijakan yang dilampirkan untuk mengakses semua sesi yang dibuat di akun Anda (seperti pernyataan waktu proses atau pernyataan pembatalan).

Jika Anda ingin melindungi sesi Anda dan menjadikannya pribadi hanya untuk peran IAM tertentu, seperti yang terkait dengan pengguna yang membuat sesi maka Anda dapat menggunakan Kontrol Otorisasi Berbasis Tag Sesi AWS Glue Interaktif yang disebut. TagOnCreate Untuk informasi selengkapnya, lihat Jadikan sesi Anda pribadi dengan TagOnCreate bagaimana kebijakan terkelola cakupan berbasis tag pemilik dapat membuat sesi Anda pribadi. TagOnCreate Untuk informasi selengkapnya tentang kebijakan berbasis identitas, lihat Kebijakan berbasis identitas untuk. AWS Glue

Menyiapkan peran runtime

Anda harus meneruskan peran IAM ke operasi CreateSession API agar memungkinkan AWS Glue untuk mengasumsikan dan menjalankan pernyataan dalam sesi interaktif. Peran harus memiliki izin IAM yang sama dengan yang diperlukan untuk menjalankan pekerjaan biasaAWS Glue.

Misalnya, Anda dapat membuat peran layanan menggunakan AWSGlueServiceRolekebijakan yang memungkinkan AWS Glue untuk memanggil AWS layanan atas nama Anda. Jika Anda menggunakan AWS Glue konsol, itu akan secara otomatis membuat peran layanan atas nama Anda atau menggunakan yang sudah ada. Anda juga dapat membuat peran IAM Anda sendiri dan melampirkan kebijakan IAM Anda sendiri untuk mengizinkan izin serupa.

Jika Anda ingin melindungi sesi Anda dan menjadikannya pribadi hanya untuk pengguna yang membuat sesi maka Anda dapat menggunakan Kontrol Otorisasi Berbasis Tag Sesi AWS Glue Interaktif yang disebut TagOnCreate. Untuk informasi selengkapnya, lihat Jadikan sesi Anda pribadi dengan TagOnCreate bagaimana kebijakan terkelola cakupan berbasis tag pemilik dapat membuat sesi Anda pribadi. TagOnCreate Untuk informasi selengkapnya tentang kebijakan berbasis identitas, lihat. Kebijakan berbasis identitas untuk Glue AWS Jika Anda membuat peran eksekusi sendiri dari konsol IAM dan Anda ingin menjadikan layanan Anda pribadi dengan TagOnCreate fitur, ikuti langkah-langkah di bawah ini.

1. Buat peran IAM dengan tipe peran yang disetel ke`Glue`.
2. Lampirkan kebijakan AWS Glue terkelola ini: AwsGlueSessionUserRestrictedServiceRole
3. Awalan nama peran dengan nama AwsGlueSessionUserRestrictedServiceRolekebijakan. Misalnya, Anda dapat membuat peran dengan nama AwsGlueSessionUserRestrictedServiceRole-myrole dan melampirkan kebijakan AWS Glue terkelola. AwsGlueSessionUserRestrictedServiceRole
4. Lampirkan kebijakan kepercayaan seperti berikut untuk memungkinkan AWS Glue untuk mengambil peran:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "glue.amazonaws.com"
        ]
      },
      "Action": [
        "sts:AssumeRole"
      ]
    }
  ]
```

```
}
```

Untuk sesi interaktif Jupyter kernel, Anda dapat menentukan `iam_role` kunci di profil Anda AWS Command Line Interface . Untuk informasi selengkapnya, lihat Mengonfigurasi sesi dengan ~/.aws/config. Jika Anda berinteraksi dengan sesi interaktif menggunakan AWS Glue buku catatan, maka Anda dapat meneruskan peran eksekusi dalam `%iam_role` keajaiban di sel pertama yang Anda jalankan.

Jadikan sesi Anda pribadi dengan TagOnCreate

AWS Gluesesi interaktif mendukung penandaan dan Otorisasi Berbasis Tag (TBAC) untuk sesi interaktif sebagai sumber daya bernama. Selain penggunaan TBAC TagResource dan UntagResource API, sesi AWS Glue interaktif mendukung TagOnCreate fitur untuk 'menandai' sesi dengan tag yang diberikan hanya selama pembuatan sesi dengan CreateSession operasi. Ini juga berarti tag tersebut akan dihapus DeleteSession, alias UntagOnDelete.

TagOnCreate menawarkan mekanisme keamanan yang kuat untuk membuat sesi Anda pribadi bagi pembuat sesi. Misalnya, Anda dapat melampirkan kebijakan IAM dengan “pemilik” RequestTag dan nilai $ {AWS:userId} ke prinsipal klien (seperti pengguna) untuk mengizinkan pembuatan sesi hanya jika tag “pemilik” dengan nilai yang cocok dari userId penelepon disediakan sebagai tag userID dalam permintaan. CreateSession Kebijakan ini memungkinkan sesi AWS Glue interaktif untuk membuat sumber daya sesi dan menandai sesi dengan tag userID hanya selama waktu pembuatan sesi. Selain itu, Anda dapat mencakup akses (seperti pernyataan berjalan) ke sesi Anda hanya ke pembuat (alias tag pemilik dengan nilai $ {AWS:userId}) sesi dengan melampirkan kebijakan IAM dengan “pemilik” ResourceTag ke peran eksekusi yang Anda berikan selama. CreateSession

Untuk memudahkan Anda menggunakan TagOnCreate fitur untuk membuat sesi pribadi bagi pembuat sesi, AWS Glue berikan kebijakan terkelola khusus dan peran layanan.

Jika Anda ingin membuat Sesi AWS Glue Interaktif menggunakan AssumeRole prinsipal IAM (yaitu, menggunakan kredensi yang dijual dengan mengasumsikan peran IAM) dan Anda ingin membuat sesi pribadi bagi pembuatnya, maka gunakan kebijakan yang mirip dengan dan masing-masing. AWSGlueSessionUserRestrictedNotebookPolicyAWSGlueSessionUserRestrictedNotebookServiceRole Kebijakan ini memungkinkan AWS Glue untuk menggunakan $ {aws:PrincipalTag} untuk mengekstrak nilai tag pemilik. Ini mengharuskan Anda untuk meneruskan tag userId dengan nilai $ {aws:userId} seperti SessionTag pada kredensi peran asumsi. Lihat tag sesi ID. Jika Anda menggunakan instans Amazon EC2 dengan profil instans yang menjual kredensi dan Anda ingin membuat sesi atau berinteraksi dengan sesi dari dalam instans Amazon EC2, maka Anda

harus meneruskan tag userID dengan nilai $ {AWS:userId} seperti pada kredensi peran asumsi. SessionTag

Misalnya, Jika Anda membuat sesi menggunakan kredensi AssumeRole utama IAM dan Anda ingin menjadikan layanan Anda pribadi dengan TagOnCreate fitur, ikuti langkah-langkah di bawah ini.

1. Buat peran runtime sendiri dari konsol IAM. Harap lampirkan kebijakan AWS Glue terkelola ini AwsGlueSessionUserRestrictedNotebookServiceRoledan awali nama peran dengan nama AwsGlueSessionUserRestrictedNotebookServiceRolekebijakan. Misalnya, Anda dapat membuat peran dengan nama AwsGlueSessionUserRestrictedNotebookServiceRole-myrole dan melampirkan kebijakan AWS Glue terkelola. AwsGlueSessionUserRestrictedNotebookServiceRole
2. Lampirkan kebijakan kepercayaan seperti di bawah ini untuk memungkinkan AWS Glue untuk mengambil peran di atas.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "glue.amazonaws.com"
        ]
      },
      "Action": [
        "sts:AssumeRole"
       ]
    }
   ]
}
```

3. Buat peran lain bernama dengan awalan AwsGlueSessionUserRestrictedNotebookPolicydan lampirkan kebijakan AWS Glue terkelola AwsGlueSessionUserRestrictedNotebookPolicyuntuk membuat sesi menjadi pribadi. Selain kebijakan terkelola, lampirkan kebijakan inline berikut untuk mengizinkan iam: PassRole ke peran yang Anda buat di langkah 1.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::*:role/
AwsGlueSessionUserRestrictedNotebookServiceRole*"
            ],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "glue.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

4. Lampirkan kebijakan kepercayaan seperti mengikuti IAM di atas AWS Glue untuk mengambil peran.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Principal": {
            "Service": [
                "glue.amazonaws.com"
            ]
        },
        "Action": [
            "sts:AssumeRole",
            "sts:TagSession"
        ]
    }]
}
```

> **Note**
>
> Secara opsional, Anda dapat menggunakan satu peran (misalnya, peran buku catatan) dan melampirkan kedua kebijakan terkelola di atas AwsGlueSessionUserRestrictedNotebookServiceRoledan AwsGlueSessionUserRestrictedNotebookPolicy. Juga lampirkan kebijakan inline tambahan untuk mengizinkan `iam:passrole` peran Anda. AWS Glue Dan akhirnya lampirkan kebijakan kepercayaan di atas untuk mengizinkan `sts:AssumeRole` dan`sts:TagSession`.

AWSGlueSessionUserRestrictedNotebookPolicy

AWSGlueSessionUserRestrictedNotebookPolicy Menyediakan akses untuk membuat Sesi AWS Glue Interaktif dari buku catatan hanya jika kunci tag “pemilik” dan nilai yang cocok dengan id AWS pengguna prinsipal (pengguna atau Peran). Untuk informasi selengkapnya, lihat Di mana Anda dapat menggunakan variabel kebijakan. Kebijakan ini dilampirkan pada prinsipal (Pengguna atau peran) yang membuat buku catatan Sesi AWS Glue Interaktif dariAWS Glue Studio. Kebijakan ini juga mengizinkan akses yang cukup ke AWS Glue Studio buku catatan untuk berinteraksi dengan sumber daya Sesi AWS Glue Studio Interaktif yang dibuat dengan nilai tag “pemilik” yang cocok dengan ID AWS pengguna prinsipal. Kebijakan ini menolak izin untuk mengubah atau menghapus tag “pemilik” dari sumber daya AWS Glue sesi setelah sesi dibuat.

AWSGlueSessionUserRestrictedNotebookServiceRole

AWSGlueSessionUserRestrictedNotebookServiceRoleMemberikan akses yang cukup ke AWS Glue Studio buku catatan untuk berinteraksi dengan sumber daya Sesi AWS Glue Interaktif yang dibuat dengan nilai tag “pemilik” yang cocok dengan ID AWS pengguna utama (pengguna atau peran) pembuat buku catatan. Untuk informasi selengkapnya, lihat Di mana Anda dapat menggunakan variabel kebijakan. Kebijakan peran layanan ini dilampirkan pada peran yang diteruskan sebagai sihir ke buku catatan atau diteruskan sebagai peran eksekusi ke API. CreateSession Kebijakan ini juga mengizinkan untuk membuat Sesi AWS Glue Interaktif dari buku catatan hanya jika kunci tag “pemilik” dan nilai cocok dengan ID AWS pengguna prinsipal. Kebijakan ini menolak izin untuk mengubah atau menghapus tag “pemilik” dari sumber daya AWS Glue sesi setelah sesi dibuat. Kebijakan ini juga mencakup izin untuk menulis dan membaca dari bucket Amazon S3, CloudWatch menulis log, membuat dan menghapus tag untuk sumber daya Amazon EC2 yang digunakan oleh. AWS Glue

Jadikan sesi Anda pribadi dengan kebijakan pengguna

Anda dapat melampirkan peran IAM AWSGlueSessionUserRestrictedPolicyke yang dilampirkan ke masing-masing pengguna di akun Anda untuk membatasi mereka membuat sesi hanya dengan tag pemilik dengan nilai yang cocok dengan $ {AWS:userID} mereka sendiri. Alih-alih menggunakan AWSGlueSessionUserRestrictedNotebookPolicydan AWSGlueSessionUserRestrictedNotebookServiceRoleAnda perlu menggunakan kebijakan yang mirip dengan AWSGlueSessionUserRestrictedPolicydan AWSGlueSessionUserRestrictedServiceRolemasing-masing. Untuk informasi selengkapnya, lihat Kebijakan berbasis identitas. Kebijakan ini mencakup akses ke sesi hanya untuk pencipta, $ {AWS:userId} dari pengguna yang membuat sesi dengan tag pemilik yang memuat $ {aws:userId} mereka sendiri. Jika Anda telah membuat peran eksekusi sendiri menggunakan konsol IAM dengan mengikuti langkah-langkahnyaMenyiapkan peran runtime, selain melampirkan kebijakan AwsGlueSessionUserRestrictedPolicyterkelola, lampirkan juga kebijakan sebaris berikut ke setiap pengguna di akun Anda `iam:PassRole` untuk memungkinkan peran eksekusi yang Anda buat sebelumnya.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "iam:PassRole"
        ],
        "Resource": [
            "arn:aws:iam::*:role/AwsGlueSessionUserRestrictedServiceRole*"
        ],
        "Condition": {
            "StringLike": {
                "iam:PassedToService": [
                    "glue.amazonaws.com"
                ]
            }
        }
    }]
}
```

AWSGlueSessionUserRestrictedPolicy

AWSGlueSessionUserRestrictedPolicyIni menyediakan akses untuk membuat Sesi AWS Glue Interaktif menggunakan CreateSession API hanya jika kunci tag “pemilik” dan nilai yang cocok dengan ID AWS pengguna mereka disediakan. Kebijakan identitas ini dilampirkan ke pengguna yang memanggil CreateSession API. Kebijakan ini juga memungkinkan untuk berinteraksi dengan sumber daya Sesi AWS Glue Interaktif yang dibuat dengan tag “pemilik” dan nilai yang cocok dengan id AWS pengguna mereka. Kebijakan ini menolak izin untuk mengubah atau menghapus tag “pemilik” dari sumber daya AWS Glue sesi setelah sesi dibuat.

AWSGlueSessionUserRestrictedServiceRole

AWSGlueSessionUserRestrictedServiceRoleIni menyediakan akses penuh ke semua AWS Glue sumber daya kecuali untuk sesi dan memungkinkan pengguna untuk membuat dan menggunakan hanya sesi interaktif yang terkait dengan pengguna. Kebijakan ini juga mencakup izin lain yang diperlukan AWS Glue untuk mengelola sumber daya Glue di AWS layanan lain. Kebijakan ini juga memungkinkan penambahan tag ke AWS Glue sumber daya di AWS layanan lain.

Pertimbangan kebijakan IAM

Sesi interaktif adalah sumber daya IAM diAWS Glue. Karena mereka adalah sumber daya IAM, akses dan interaksi ke sesi diatur oleh kebijakan IAM. Berdasarkan kebijakan IAM yang dilampirkan pada prinsipal klien atau peran eksekusi yang dikonfigurasi oleh admin, kepala klien (pengguna atau peran) akan dapat membuat sesi baru dan berinteraksi dengan sesi sendiri dan sesi lainnya.

Jika admin telah melampirkan kebijakan IAM seperti AWSGlueConsoleFullAccess atau AWSGlueServiceRole yang memungkinkan akses ke semua AWS Glue sumber daya di akun itu, prinsipal klien akan dapat berkolaborasi satu sama lain. Misalnya, satu pengguna akan dapat berinteraksi dengan sesi yang dibuat oleh pengguna lain jika kebijakan mengizinkan ini.

Jika Anda ingin mengonfigurasi kebijakan yang disesuaikan dengan kebutuhan spesifik Anda, lihat dokumentasi IAM tentang mengonfigurasi sumber daya untuk kebijakan. Misalnya, untuk mengisolasi sesi milik pengguna, Anda dapat menggunakan TagOnCreate fitur yang didukung oleh sesi AWS Glue Interaktif. Lihat Jadikan sesi Anda pribadi dengan TagOnCreate .

Sesi interaktif mendukung pembatasan pembuatan sesi berdasarkan kondisi VPC tertentu. Lihat Kebijakan kontrol yang mengontrol pengaturan menggunakan tombol kondisi.

Mengubah skrip atau buku catatan menjadi pekerjaan AWS Glue

Ada dua cara Anda dapat mengubah skrip atau buku catatan menjadi AWS Glue pekerjaan:

- Gunakan nbconvert untuk mengonversi file dokumen `.ipynb` notebook Jupyter Anda menjadi file. `.py` Untuk informasi selengkapnya, lihat nbconvert: Mengkonversi Notebook ke format lain.
- Unggah file ke AWS Glue Studio Notebook.
  - Di AWS Glue Studio konsol, pilih Jobs dari menu navigasi.
  - Di bagian Buat pekerjaan, pilih Jupyter Notebook.
  - Di bagian Opsi, pilih Unggah dan edit buku catatan yang ada.
  - Pilih Pilih file untuk mengunggah `.ipynb` file.

AWS Gluesesi interaktif untuk streaming

Mengalihkan jenis sesi streaming

Gunakan sihir konfigurasi sesi AWS Glue interaktif,`%streaming`, untuk menentukan pekerjaan yang Anda jalankan dan menginisialisasi sesi interaktif streaming.

Sampling input stream untuk pengembangan interaktif

Salah satu alat yang kami peroleh untuk membantu meningkatkan pengalaman AWS Glue interaktif dalam sesi interaktif adalah penambahan metode baru `GlueContext` untuk mendapatkan snapshot aliran dalam DynamicFrame statis. `GlueContext`memungkinkan Anda untuk memeriksa, berinteraksi, dan mengimplementasikan alur kerja Anda.

Dengan instance `GlueContext` kelas, Anda akan dapat menemukan metode`getSampleStreamingDynamicFrame`. Argumen yang diperlukan untuk metode ini adalah:

- `dataFrame`: Streaming Spark DataFrame
- `options`: Lihat opsi yang tersedia di bawah ini

Pilihan yang tersedia meliputi:

- WindowSize: Ini juga disebut Durasi Microbatch. Parameter ini akan menentukan berapa lama kueri streaming akan menunggu setelah batch sebelumnya dipicu. Nilai parameter ini harus lebih kecil dari`pollingTimeInMs`.

- pollingTimeInMs: Total panjang waktu metode akan berjalan. Ini akan menembakkan setidaknya satu batch mikro untuk mendapatkan catatan sampel dari aliran input.
- recordPollingLimit: Parameter ini membantu Anda membatasi jumlah total catatan yang akan Anda polling dari aliran.
- (Opsional) Anda juga dapat menggunakan `writeStreamFunction` untuk menerapkan fungsi kustom ini ke setiap fungsi pengambilan sampel rekaman. Lihat di bawah untuk contoh di Scala dan Python.

Scala

```
val sampleBatchFunction = (batchDF: DataFrame, batchId: Long) => {//Optional but
 you can replace your own forEachBatch function here}
val jsonString: String = s"""{"pollingTimeInMs": "10000", "windowSize": "5
 seconds"}"""
val dynFrame = glueContext.getSampleStreamingDynamicFrame(YOUR_STREAMING_DF,
 JsonOptions(jsonString), sampleBatchFunction)
dynFrame.show()
```

Python

```
def sample_batch_function(batch_df, batch_id):
       //Optional but you can replace your own forEachBatch function here
options = {
            "pollingTimeInMs": "10000",
            "windowSize": "5 seconds",
        }
glue_context.getSampleStreamingDynamicFrame(YOUR_STREAMING_DF, options,
 sample_batch_function)
```

> **Note**
>
> Ketika sampel `DynFrame` kosong, itu bisa disebabkan oleh beberapa alasan:
>
> - Sumber Streaming diatur ke “Terbaru” dan tidak ada data baru yang tertelan selama periode pengambilan sampel.

- Waktu pemungutan suara tidak cukup untuk memproses catatan yang dicerna. Data tidak akan muncul kecuali seluruh batch telah diproses.

Menjalankan aplikasi streaming dalam sesi interaktif

Dalam sesi AWS Glue interaktif, Anda dapat menjalankan aplikasi AWS Glue streaming seperti bagaimana Anda akan membuat aplikasi streaming di AWS Glue Konsol. Karena sesi interaktif berbasis sesi, menghadapi pengecualian di runtime tidak menyebabkan sesi berhenti. Kami sekarang memiliki manfaat tambahan untuk mengembangkan fungsi batch Anda secara iteratif. Misalnya:

```
def batch_function(data_frame, batch_id):
    log.info(data_frame.count())
    invalid_method_call()
glueContext.forEachBatch(frame=streaming_df, batch_function = batch_function, options =
 {**})
```

Dalam contoh di atas, kami menyertakan penggunaan metode yang tidak valid dan tidak seperti AWS Glue pekerjaan biasa yang akan keluar dari seluruh aplikasi, konteks dan definisi pengkodean pengguna sepenuhnya dipertahankan dan sesi masih beroperasi. Tidak perlu mem-bootstrap cluster baru dan menjalankan kembali semua transformasi sebelumnya. Ini memungkinkan Anda untuk fokus pada iterasi implementasi fungsi batch Anda dengan cepat untuk mendapatkan hasil yang diinginkan.

Penting untuk dicatat bahwa Sesi Interaktif mengevaluasi setiap pernyataan dengan cara pemblokiran sehingga sesi hanya akan mengeksekusi satu pernyataan pada satu waktu. Karena kueri streaming berkelanjutan dan tidak pernah berakhir, sesi dengan kueri streaming aktif tidak akan dapat menangani pernyataan tindak lanjut apa pun kecuali jika terputus. Anda dapat mengeluarkan perintah interupsi langsung dari Jupyter Notebook dan kernel kami akan menangani pembatalan untuk Anda.

Ambil urutan pernyataan berikut yang menunggu eksekusi sebagai contoh:

```
Statement 1:
      val number = df.count()
      #Spark Action with deterministic result
      Result: 5
```

```
Statement 2:
      streamingQuery.start().awaitTermination()
      #Spark Streaming Query that will be executing continously
      Result: Constantly updated with each microbatch

Statement 3:
      val number2 = df.count()
      #This will not be executed as previous statement will be running indefinitely
```

Mengembangkan dan menguji skrip AWS Glue pekerjaan secara lokal

Saat Anda mengembangkan dan menguji skrip AWS Glue pekerjaan Spark, ada beberapa opsi yang tersedia:

- Konsol AWS Glue Studio
  - Editor visual
  - Editor skrip
  - AWS Glue Studiobuku catatan
- Sesi interaktif
  - Buku catatan Jupyter
- Gambar Docker
  - Pembangunan lokal
  - Pengembangan jarak jauh
- AWS Glue StudioPerpustakaan ETL
  - Pembangunan lokal

Anda dapat memilih salah satu opsi di atas berdasarkan kebutuhan Anda.

Jika Anda lebih suka tidak ada kode atau kurang pengalaman kode, editor AWS Glue Studio visual adalah pilihan yang baik.

Jika Anda lebih suka pengalaman notebook interaktif, AWS Glue Studio notebook adalah pilihan yang baik. Untuk informasi selengkapnya, lihat Menggunakan Buku Catatan dengan AWS Glue Studio dan

AWS Glue. Jika Anda ingin menggunakan lingkungan lokal Anda sendiri, sesi interaktif adalah pilihan yang baik. Untuk informasi selengkapnya, lihat Menggunakan sesi interaktif dengan AWS Glue.

Jika Anda lebih suka pengalaman pengembangan lokal/jarak jauh, gambar Docker adalah pilihan yang baik. Ini membantu Anda mengembangkan dan menguji skrip pekerjaan AWS Glue for Spark di mana pun Anda inginkan tanpa menimbulkan AWS Glue biaya.

Jika Anda lebih suka pengembangan lokal tanpa Docker, menginstal direktori pustaka AWS Glue ETL secara lokal adalah pilihan yang baik.

Mengembangkan menggunakan AWS Glue Studio

Editor AWS Glue Studio visual adalah antarmuka grafis yang memudahkan untuk membuat, menjalankan, dan memantau pekerjaan ekstrak, transformasi, dan pemuatan (ETL) diAWS Glue. Anda dapat menyusun alur kerja transformasi data secara visual dan menjalankannya dengan mulus di mesin ETL tanpa server AWS Glue berbasis Apache Spark. Anda dapat memeriksa skema dan hasil data di masing-masing langkah dari tugas tersebut. Untuk informasi selengkapnya, lihat Panduan Pengguna AWS Glue Studio.

Mengembangkan menggunakan sesi interaktif

Sesi interaktif memungkinkan Anda untuk membangun dan menguji aplikasi dari lingkungan pilihan Anda. Untuk informasi selengkapnya, lihat Menggunakan sesi interaktif dengan AWS Glue.

Mengembangkan menggunakan gambar Docker

> **Note**
>
> Petunjuk di bagian ini masih belum diuji pada sistem operasi Microsoft Windows. Untuk pengembangan dan pengujian lokal pada platform Windows, lihat blog Membangun pipeline AWS Glue ETL secara lokal tanpa akun AWS

Untuk platform data siap produksi, proses pengembangan dan pipa CI/CD untuk AWS Glue pekerjaan adalah topik utama. Anda dapat mengembangkan dan menguji AWS Glue pekerjaan secara fleksibel dalam wadah Docker. AWS Gluemeng-host gambar Docker di Docker Hub untuk mengatur lingkungan pengembangan Anda dengan utilitas tambahan. Anda dapat menggunakan IDE, notebook, atau REPL pilihan Anda menggunakan pustaka AWS Glue ETL. Topik ini

menjelaskan cara mengembangkan dan menguji pekerjaan AWS Glue versi 4.0 dalam wadah Docker menggunakan gambar Docker.

Gambar Docker berikut tersedia untuk AWS Glue di Docker Hub.

- Untuk AWS Glue versi 4.0: `amazon/aws-glue-libs:glue_libs_4.0.0_image_01`
- Untuk AWS Glue versi 3.0: `amazon/aws-glue-libs:glue_libs_3.0.0_image_01`
- Untuk AWS Glue versi 2.0: `amazon/aws-glue-libs:glue_libs_2.0.0_image_01`

Gambar-gambar ini untuk x86_64. Disarankan agar Anda menguji arsitektur ini. Namun, dimungkinkan untuk mengerjakan ulang solusi pengembangan lokal pada gambar dasar yang tidak didukung.

Contoh ini menjelaskan penggunaan `amazon/aws-glue-libs:glue_libs_4.0.0_image_01` dan menjalankan wadah pada mesin lokal. Gambar kontainer ini telah diuji untuk pekerjaan Spark AWS Glue versi 3.3. Gambar ini berisi yang berikut:

- Amazon Linux
- AWS GluePerpustakaan ETL () aws-glue-libs
- Apache Spark 3.3.0
- Server riwayat percikan
- Lab Jupyter
- Livy
- Dependensi perpustakaan lainnya (set yang sama dengan yang ada di sistem AWS Glue pekerjaan)

Lengkapi salah satu bagian berikut sesuai dengan kebutuhan Anda:

- Siapkan wadah untuk menggunakan spark-submit
- Siapkan wadah untuk menggunakan REPL shell () PySpark
- Siapkan wadah untuk menggunakan Pytest
- Siapkan wadah untuk menggunakan Jupyter Lab
- Siapkan wadah untuk menggunakan Visual Studio Code

Prasyarat

Sebelum Anda mulai, pastikan Docker diinstal dan daemon Docker sedang berjalan. Untuk petunjuk penginstalan, lihat dokumentasi Docker untuk Mac atau Linux. Mesin yang menjalankan Docker meng-host AWS Glue wadah. Juga pastikan bahwa Anda memiliki setidaknya 7 GB ruang disk untuk gambar pada host yang menjalankan Docker.

Untuk informasi selengkapnya tentang pembatasan saat mengembangkan AWS Glue kode secara lokal, lihat Pembatasan pengembangan lokal.

Mengonfigurasi AWS

Untuk mengaktifkan panggilan AWS API dari penampung, siapkan AWS kredensyal dengan mengikuti langkah-langkah berikut. Di bagian berikut, kami akan menggunakan profil AWS bernama ini.

1. Siapkan AWS CLI, konfigurasi profil bernama. Untuk informasi selengkapnya tentang AWS CLI konfigurasi, lihat Konfigurasi dan pengaturan file kredensyal dalam AWS CLI dokumentasi.
2. Jalankan perintah berikut di terminal:

```
PROFILE_NAME="<your_profile_name>"
```

Anda mungkin juga perlu mengatur variabel lingkungan AWS_REGION untuk menentukan permintaan Wilayah AWS untuk mengirim.

Menyiapkan dan menjalankan wadah

Menyiapkan wadah untuk menjalankan PySpark kode melalui perintah spark-submit mencakup langkah-langkah tingkat tinggi berikut:

1. Tarik gambar dari Docker Hub.
2. Jalankan wadah.

Menarik gambar dari Docker Hub

Jalankan perintah berikut untuk menarik gambar dari Docker Hub:

```
docker pull amazon/aws-glue-libs:glue_libs_4.0.0_image_01
```

Menjalankan wadah

Anda sekarang dapat menjalankan wadah menggunakan gambar ini. Anda dapat memilih salah satu dari berikut berdasarkan kebutuhan Anda.

spark-submit

Anda dapat menjalankan skrip AWS Glue pekerjaan dengan menjalankan `spark-submit` perintah pada wadah.

1. Tulis skrip dan simpan seperti `sample1.py` di bawah `/local_path_to_workspace` direktori. Kode sampel disertakan sebagai lampiran dalam topik ini.

```
$ WORKSPACE_LOCATION=/local_path_to_workspace
$ SCRIPT_FILE_NAME=sample.py
$ mkdir -p ${WORKSPACE_LOCATION}/src
$ vim ${WORKSPACE_LOCATION}/src/${SCRIPT_FILE_NAME}
```

2. Jalankan perintah berikut untuk menjalankan `spark-submit` perintah pada wadah untuk mengirimkan aplikasi Spark baru:

```
$ docker run -it -v ~/.aws:/home/glue_user/.aws -v $WORKSPACE_LOCATION:/
home/glue_user/workspace/ -e AWS_PROFILE=$PROFILE_NAME -e DISABLE_SSL=true
 --rm -p 4040:4040 -p 18080:18080 --name glue_spark_submit amazon/aws-glue-
libs:glue_libs_4.0.0_image_01 spark-submit /home/glue_user/workspace/src/
$SCRIPT_FILE_NAME
...22/01/26 09:08:55 INFO DAGScheduler: Job 0 finished: fromRDD at
 DynamicFrame.scala:305, took 3.639886 s
root
|-- family_name: string
|-- name: string
|-- links: array
| |-- element: struct
| | |-- note: string
| | |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
| |-- element: struct
| | |-- scheme: string
| | |-- identifier: string
|-- other_names: array
| |-- element: struct
```

```
| | |-- lang: string
| | |-- note: string
| | |-- name: string
|-- sort_name: string
|-- images: array
| |-- element: struct
| | |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
| |-- element: struct
| | |-- type: string
| | |-- value: string
|-- death_date: string

...
```

3. (Opsional) Konfigurasikan `spark-submit` agar sesuai dengan lingkungan Anda. Misalnya, Anda dapat meneruskan dependensi Anda dengan konfigurasi. `--jars` Untuk informasi lebih lanjut, lihat Peluncuran Aplikasi dengan spark-submit dalam dokumentasi Spark.

Cangkang REPL (Pyspark)

Anda dapat menjalankan REPL (read-eval-print loop) shell untuk pengembangan interaktif.

Jalankan perintah berikut untuk menjalankan PySpark perintah pada wadah untuk memulai shell REPL:

```
$ docker run -it -v ~/.aws:/home/glue_user/.aws -e AWS_PROFILE=$PROFILE_NAME -e
 DISABLE_SSL=true --rm -p 4040:4040 -p 18080:18080 --name glue_pyspark amazon/aws-glue-
libs:glue_libs_4.0.0_image_01 pyspark
...
 ____ __
 / __/__ ___ _____/ /__
 _\ \/ _ \/ _ `/ __/ '_/
 /__ / .__/\_,_/_/ /_/\_\  version 3.1.1-amzn-0
 /_/

Using Python version 3.7.10 (default, Jun 3 2021 00:02:01)
Spark context Web UI available at http://56e99d000c99:4040
Spark context available as 'sc' (master = local[*], app id = local-1643011860812).
SparkSession available as 'spark'.
```

```
>>>
```

Pytest

Untuk pengujian unit, Anda dapat menggunakan pytest untuk skrip pekerjaan AWS Glue Spark.

Jalankan perintah berikut untuk persiapan.

```
$ WORKSPACE_LOCATION=/local_path_to_workspace
$ SCRIPT_FILE_NAME=sample.py
$ UNIT_TEST_FILE_NAME=test_sample.py
$ mkdir -p ${WORKSPACE_LOCATION}/tests
$ vim ${WORKSPACE_LOCATION}/tests/${UNIT_TEST_FILE_NAME}
```

Jalankan perintah berikut untuk mengeksekusi `pytest` pada test suite:

```
$ docker run -it -v ~/.aws:/home/glue_user/.aws -v $WORKSPACE_LOCATION:/home/glue_user/
workspace/ -e AWS_PROFILE=$PROFILE_NAME -e DISABLE_SSL=true --rm -p 4040:4040 -p
 18080:18080 --name glue_pytest amazon/aws-glue-libs:glue_libs_4.0.0_image_01 -c
 "python3 -m pytest"
starting org.apache.spark.deploy.history.HistoryServer,
 logging to /home/glue_user/spark/logs/spark-glue_user-
org.apache.spark.deploy.history.HistoryServer-1-5168f209bd78.out
*============================================================= test session starts
 ==============================================================
*platform linux -- Python 3.7.10, pytest-6.2.3, py-1.11.0, pluggy-0.13.1
rootdir: /home/glue_user/workspace
plugins: anyio-3.4.0
*collected 1 item  *

tests/test_sample.py . [100%]

============================================================== warnings summary
 ===============================================================
tests/test_sample.py::test_counts
 /home/glue_user/spark/python/pyspark/sql/context.py:79: DeprecationWarning: Deprecated
 in 3.0.0. Use SparkSession.builder.getOrCreate() instead.
 DeprecationWarning)

-- Docs: https://docs.pytest.org/en/stable/warnings.html
======================================================== 1 passed, *1 warning* in
 21.07s =========================================================
```

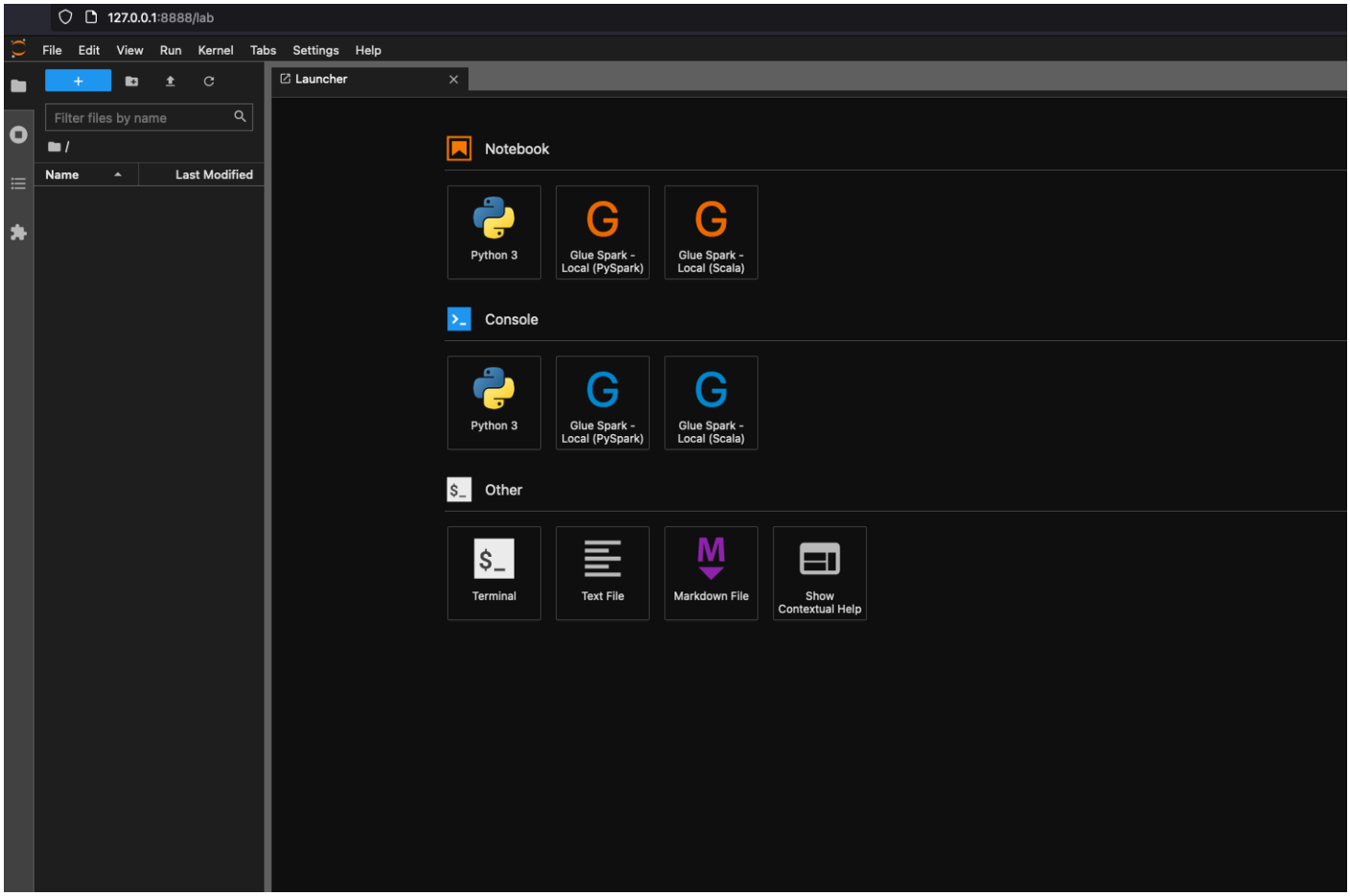
Lab Jupyter

Anda dapat memulai Jupyter untuk pengembangan interaktif dan kueri ad-hoc di notebook.

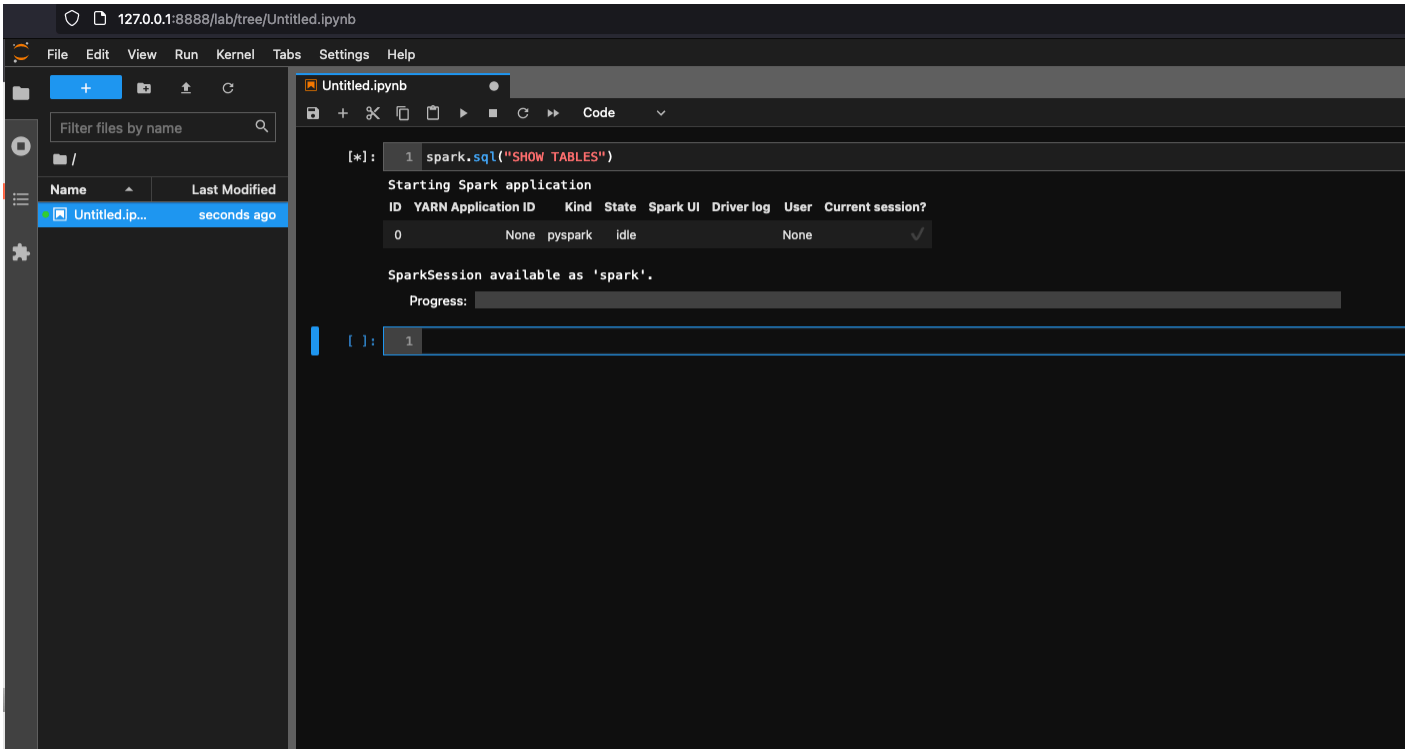
1. Jalankan perintah berikut untuk memulai Jupyter Lab:

```
$ JUPYTER_WORKSPACE_LOCATION=/local_path_to_workspace/jupyter_workspace/
$ docker run -it -v ~/.aws:/home/glue_user/.aws -v $JUPYTER_WORKSPACE_LOCATION:/
home/glue_user/workspace/jupyter_workspace/ -e AWS_PROFILE=$PROFILE_NAME -e
 DISABLE_SSL=true --rm -p 4040:4040 -p 18080:18080 -p 8998:8998 -p 8888:8888 --name
 glue_jupyter_lab amazon/aws-glue-libs:glue_libs_4.0.0_image_01 /home/glue_user/
jupyter/jupyter_start.sh
...
[I 2022-01-24 08:19:21.368 ServerApp] Serving notebooks from local directory: /home/
glue_user/workspace/jupyter_workspace
[I 2022-01-24 08:19:21.368 ServerApp] Jupyter Server 1.13.1 is running at:
[I 2022-01-24 08:19:21.368 ServerApp] http://faa541f8f99f:8888/lab
[I 2022-01-24 08:19:21.368 ServerApp] or http://127.0.0.1:8888/lab
[I 2022-01-24 08:19:21.368 ServerApp] Use Control-C to stop this server and shut down
 all kernels (twice to skip confirmation).
```

2. Buka http://127.0.0.1:8888/lab di browser web Anda di mesin lokal Anda, untuk melihat UI lab Jupyter.



3. Pilih Glue Spark Local (PySpark) di bawah Notebook. Anda dapat mulai mengembangkan kode di UI notebook Jupyter interaktif.



Menyiapkan wadah untuk menggunakan Visual Studio Code

Prasyarat:

1. Instal Kode Visual Studio.
2. Instal Python.
3. Instal Visual Studio Code Remote - Wadah
4. Buka folder ruang kerja di Visual Studio Code.
5. Pilih Pengaturan.
6. Pilih Workspace.
7. Pilih Buka Pengaturan (JSON).
8. Tempel JSON berikut dan simpan.

```
{
    "python.defaultInterpreterPath": "/usr/bin/python3",
    "python.analysis.extraPaths": [
        "/home/glue_user/aws-glue-libs/PyGlue.zip:/home/glue_user/spark/python/lib/
py4j-0.10.9-src.zip:/home/glue_user/spark/python/",
    ]
```

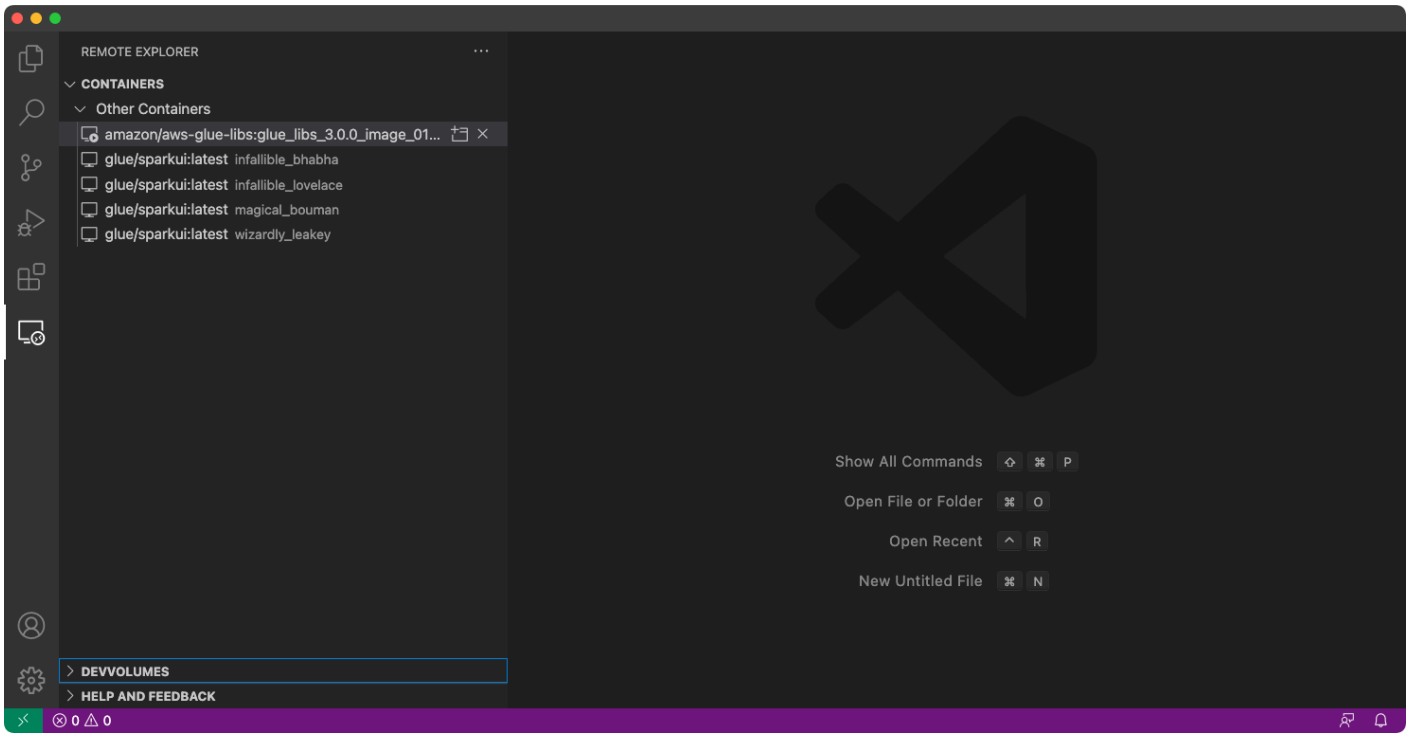
```
}
```

Langkah:

1. Jalankan wadah Docker.

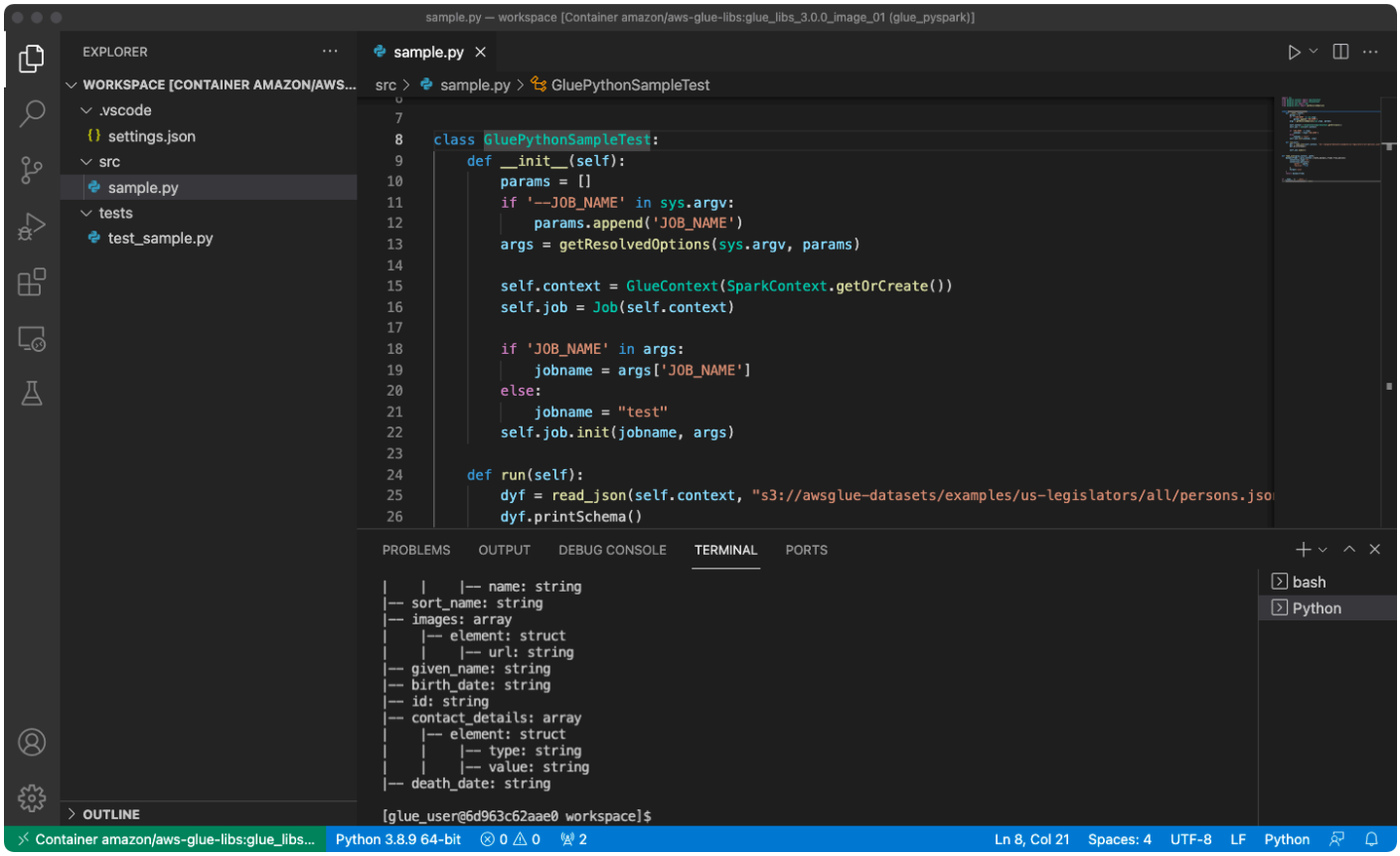
```
$ docker run -it -v ~/.aws:/home/glue_user/.aws -v $WORKSPACE_LOCATION:/
home/glue_user/workspace/ -e AWS_PROFILE=$PROFILE_NAME -e DISABLE_SSL=true
 --rm -p 4040:4040 -p 18080:18080 --name glue_pyspark amazon/aws-glue-
libs:glue_libs_4.0.0_image_01 pyspark
```

2. Mulai Kode Visual Studio.
3. Pilih Remote Explorer di menu sebelah kiri, dan pilih`amazon/aws-glue-libs:glue_libs_4.0.0_image_01`.



4. Klik kanan dan pilih Lampirkan ke Kontainer. Jika dialog ditampilkan, pilih Got it.
5. Buka `/home/glue_user/workspace/`.
6. Buat PySpark skrip Glue dan pilih Run.

   Anda akan melihat keberhasilan menjalankan skrip.



Lampiran: kode contoh AWS Glue pekerjaan untuk pengujian

Lampiran ini menyediakan skrip sebagai kode contoh AWS Glue pekerjaan untuk tujuan pengujian.

sample.py: Contoh kode untuk memanfaatkan pustaka AWS Glue ETL dengan panggilan API Amazon S3

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions


class GluePythonSampleTest:
    def __init__(self):
        params = []
        if '--JOB_NAME' in sys.argv:
            params.append('JOB_NAME')
        args = getResolvedOptions(sys.argv, params)
```

```
        self.context = GlueContext(SparkContext.getOrCreate())
        self.job = Job(self.context)

        if 'JOB_NAME' in args:
            jobname = args['JOB_NAME']
        else:
            jobname = "test"
        self.job.init(jobname, args)

    def run(self):
        dyf = read_json(self.context, "s3://awsglue-datasets/examples/us-legislators/
all/persons.json")
        dyf.printSchema()

        self.job.commit()


def read_json(glue_context, path):
    dynamicframe = glue_context.create_dynamic_frame.from_options(
        connection_type='s3',
        connection_options={
            'paths': [path],
            'recurse': True
        },
        format='json'
    )
    return dynamicframe


if __name__ == '__main__':
    GluePythonSampleTest().run()
```

Kode di atas memerlukan izin Amazon S3 di IAM. AWS Anda harus memberikan kebijakan terkelola IAM `arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess` atau kebijakan kustom IAM yang memungkinkan Anda menelepon `ListBucket` dan `GetObject` untuk jalur Amazon S3.

test_sample.py: Contoh kode untuk unit test sample.py.

```
import pytest
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
```

```
from awsglue.utils import getResolvedOptions
import sys
from src import sample


@pytest.fixture(scope="module", autouse=True)
def glue_context():
    sys.argv.append('--JOB_NAME')
    sys.argv.append('test_count')

    args = getResolvedOptions(sys.argv, ['JOB_NAME'])
    context = GlueContext(SparkContext.getOrCreate())
    job = Job(context)
    job.init(args['JOB_NAME'], args)

    yield(context)

    job.commit()


def test_counts(glue_context):
    dyf = sample.read_json(glue_context, "s3://awsglue-datasets/examples/us-
legislators/all/persons.json")
    assert dyf.toDF().count() == 1961
```

Mengembangkan menggunakan AWS Glue pustaka ETL

Pustaka AWS Glue ETL tersedia dalam bucket Amazon S3 publik, dan dapat dikonsumsi oleh sistem build Apache Maven. Hal ini memungkinkan Anda untuk mengembangkan dan menguji skrip extract, transform, and load (ETL) Python dan Scala secara lokal, tanpa perlu menggunakan koneksi jaringan. Pengembangan lokal dengan image Docker direkomendasikan, karena menyediakan lingkungan yang dikonfigurasi dengan benar untuk penggunaan perpustakaan ini.

Pengembangan lokal tersedia untuk semua AWS Glue versi, termasuk AWS Glue versi 0.9, 1.0, 2.0, dan yang lebih baru. Untuk informasi tentang versi Python dan Apache Spark yang tersedia, lihat. AWS Glue Glue version job property

Perpustakaan tersebut dirilis dengan lisensi Amazon Software (https://aws.amazon.com/asl).

Pembatasan pembangunan lokal

Ingatlah batasan berikut saat menggunakan pustaka AWS Glue Scala untuk mengembangkan secara lokal.

- Hindari membuat toples rakitan (“toples lemak” atau “toples uber”) dengan AWS Glue perpustakaan karena menyebabkan fitur-fitur berikut dinonaktifkan:
  - Bookmark Job
  - AWS GluePenulis parket () Menggunakan format Parket di AWS Glue
  - FillMissingValues transformasi (Scala atau Python)

  Fitur ini hanya tersedia dalam sistem AWS Glue pekerjaan.
- FindMatchesTransformasi ini tidak didukung dengan pembangunan lokal.
- Pembaca SIMD CSV vektor tidak didukung dengan pengembangan lokal.
- Properti customJdbcDriverS3Path untuk memuat driver JDBC dari jalur S3 tidak didukung dengan pengembangan lokal. Atau Anda dapat mengunduh driver JDBC di lokal Anda dan memuat dari sana.
- Kualitas Data Glue tidak didukung dengan pengembangan lokal.

Berkembang secara lokal dengan Python

Selesaikan beberapa langkah prasyarat dan kemudian gunakan AWS Glue utilitas untuk menguji dan mengirimkan skrip ETL Python Anda.

Prasyarat untuk pengembangan Python lokal

Selesaikan langkah-langkah ini untuk mempersiapkan pengembangan Python lokal:

1. Kloning repositori AWS Glue GitHub Python dari (). https://github.com/awslabs/aws-glue-libs
2. Lakukan salah satu dari berikut:
   - Untuk AWS Glue versi 0.9, periksa cabang`glue-0.9`.
   - Untuk AWS Glue versi 1.0, periksa cabang`glue-1.0`. Semua versi di atas AWS Glue 0.9 mendukung Python 3.
   - Untuk AWS Glue versi 2.0, lihat cabang`glue-2.0`.
   - Untuk AWS Glue versi 3.0, periksa cabang`glue-3.0`.
   - Untuk AWS Glue versi 4.0, periksa `master` cabangnya.

3. Instal Apache Maven dari lokasi berikut: https://aws-glue-etl-artifacts.s3.amazonaws.com/glue-common/apache-maven-3.6.0-bin.tar.gz.
4. Menginstal distribusi Apache Spark dari salah satu lokasi berikut:
   - Untuk AWS Glue versi 0.9: https://aws-glue-etl-artifacts.s3.amazonaws.com/glue-0.9/spark-2.2.1-bin-hadoop2.7.tgz
   - Untuk AWS Glue versi 1.0: https://aws-glue-etl-artifacts.s3.amazonaws.com/glue-1.0/spark-2.4.3-bin-hadoop2.8.tgz
   - Untuk AWS Glue versi 2.0: https://aws-glue-etl-artifacts.s3.amazonaws.com/glue-2.0/spark-2.4.3-bin-hadoop2.8.tgz
   - Untuk AWS Glue versi 3.0: https://aws-glue-etl-artifacts.s3.amazonaws.com/glue-3.0/spark-3.1.1-amzn-0-bin-3.2.1-amzn-3.tgz
   - Untuk AWS Glue versi 4.0: https://aws-glue-etl-artifacts.s3.amazonaws.com/glue-4.0/spark-3.3.0-amzn-1-bin-3.3.3-amzn-0.tgz
5. Ekspor variabel lingkungan `SPARK_HOME`, atur ke lokasi akar yang diekstrak dari arsip Spark. Misalnya:
   - Untuk AWS Glue versi 0.9: `export SPARK_HOME=/home/$USER/spark-2.2.1-bin-hadoop2.7`
   - Untuk AWS Glue versi 1.0 dan 2.0: `export SPARK_HOME=/home/$USER/spark-2.4.3-bin-spark-2.4.3-bin-hadoop2.8`
   - Untuk AWS Glue versi 3.0: `export SPARK_HOME=/home/$USER/spark-3.1.1-amzn-0-bin-3.2.1-amzn-3`
   - Untuk AWS Glue versi 4.0: `export SPARK_HOME=/home/$USER/spark-3.3.0-amzn-1-bin-3.3.3-amzn-0`

Menjalankan skrip ETL Python Anda

Dengan file AWS Glue jar yang tersedia untuk pengembangan lokal, Anda dapat menjalankan paket AWS Glue Python secara lokal.

Gunakan utilitas dan kerangka kerja berikut untuk menguji dan menjalankan skrip Python Anda. Perintah yang tercantum dalam tabel berikut dijalankan dari direktori root paket AWS GluePython.

Utilitas	Perintah	Deskripsi
AWS GlueCangkang	`./bin/gluepyspark`	Masukkan dan jalankan skrip Python dalam shell yang terintegrasi dengan pustaka ETL. AWS Glue
AWS GlueKirim	`./bin/gluesparksubmit`	Kirim sebuah skrip Python lengkap untuk eksekusi.
Pytest	`./bin/gluepytest`	Menulis dan menjalankan unit percobaan kode Python Anda. Modul pytest harus diinstal dan tersedia di `PATH`. Untuk informasi selengkapnya, lihat dokumentasi pytest.

Berkembang secara lokal dengan Scala

Selesaikan beberapa langkah prasyarat dan kemudian keluarkan perintah Maven untuk menjalankan skrip ETL Scala Anda secara lokal.

Prasyarat untuk pengembangan Scala lokal

Menyelesaikan langkah-langkah untuk mempersiapkan pengembangan Scala lokal.

Langkah 1: Instal perangkat lunak

Pada langkah ini, Anda menginstal perangkat lunak dan mengatur variabel lingkungan yang diperlukan.

1. Instal Apache Maven dari lokasi berikut: https://aws-glue-etl-artifacts.s3.amazonaws.com/glue-common/apache-maven-3.6.0-bin.tar.gz.
2. Menginstal distribusi Apache Spark dari salah satu lokasi berikut:
   - Untuk AWS Glue versi 0.9: https://aws-glue-etl-artifacts.s3.amazonaws.com/glue-0.9/spark-2.2.1-bin-hadoop2.7.tgz
   - Untuk AWS Glue versi 1.0: https://aws-glue-etl-artifacts.s3.amazonaws.com/glue-1.0/spark-2.4.3-bin-hadoop2.8.tgz
   - Untuk AWS Glue versi 2.0: https://aws-glue-etl-artifacts.s3.amazonaws.com/glue-2.0/spark-2.4.3-bin-hadoop2.8.tgz

- Untuk AWS Glue versi 3.0: https://aws-glue-etl-artifacts.s3.amazonaws.com/glue-3.0/spark-3.1.1-amzn-0-bin-3.2.1-amzn-3.tgz
- Untuk AWS Glue versi 4.0: https://aws-glue-etl-artifacts.s3.amazonaws.com/glue-4.0/spark-3.3.0-amzn-1-bin-3.3.3-amzn-0.tgz

3. Ekspor variabel lingkungan `SPARK_HOME`, atur ke lokasi akar yang diekstrak dari arsip Spark. Misalnya:
   - Untuk AWS Glue versi 0.9: `export SPARK_HOME=/home/$USER/spark-2.2.1-bin-hadoop2.7`
   - Untuk AWS Glue versi 1.0 dan 2.0: `export SPARK_HOME=/home/$USER/spark-2.4.3-bin-spark-2.4.3-bin-hadoop2.8`
   - Untuk AWS Glue versi 3.0: `export SPARK_HOME=/home/$USER/spark-3.1.1-amzn-0-bin-3.2.1-amzn-3`
   - Untuk AWS Glue versi 4.0: `export SPARK_HOME=/home/$USER/spark-3.3.0-amzn-1-bin-3.3.3-amzn-0`

Langkah 2: Konfigurasikan proyek Maven Anda

Gunakan `pom.xml` file berikut sebagai template untuk aplikasi AWS Glue Scala Anda. Ia berisi elemen `dependencies`, `repositories`, dan elemen `plugins` yang diperlukan. Ganti `Glue version` string dengan salah satu dari berikut ini:

- `4.0.0`untuk AWS Glue versi 4.0
- `3.0.0`untuk AWS Glue versi 3.0
- `1.0.0`untuk AWS Glue versi 1.0 atau 2.0
- `0.9.0`untuk AWS Glue versi 0.9

```
  <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://
maven.apache.org/maven-v4_0_0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.amazonaws</groupId>
    <artifactId>AWSGlueApp</artifactId>
    <version>1.0-SNAPSHOT</version>
    <name>${project.artifactId}</name>
    <description>AWS ETL application</description>
```

```
        <properties>
            <scala.version>2.11.1 for AWS Glue 2.0 or below, 2.12.7 for AWS Glue 3.0
 and 4.0</scala.version>
            <glue.version>Glue version with three numbers (as mentioned earlier)</
glue.version>
        </properties>
    <dependencies>
        <dependency>
            <groupId>org.scala-lang</groupId>
            <artifactId>scala-library</artifactId>
            <version>${scala.version}</version>
   <!-- A "provided" dependency, this will be ignored when you package your application
 -->
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>com.amazonaws</groupId>
            <artifactId>AWSGlueETL</artifactId>
   <version>${glue.version}</version>
            <!-- A "provided" dependency, this will be ignored when you package your
 application -->
            <scope>provided</scope>
        </dependency>
    </dependencies>

    <repositories>
        <repository>
            <id>aws-glue-etl-artifacts</id>
            <url>https://aws-glue-etl-artifacts.s3.amazonaws.com/release/</url>
        </repository>
    </repositories>
    <build>
        <sourceDirectory>src/main/scala</sourceDirectory>
        <plugins>
            <plugin>
                <!-- see http://davidb.github.com/scala-maven-plugin -->
                <groupId>net.alchim31.maven</groupId>
                <artifactId>scala-maven-plugin</artifactId>
                <version>3.4.0</version>
                <executions>
                    <execution>
                        <goals>
                            <goal>compile</goal>
                            <goal>testCompile</goal>
```

```
                    </goals>
                </execution>
            </executions>
        </plugin>
        <plugin>
            <groupId>org.codehaus.mojo</groupId>
            <artifactId>exec-maven-plugin</artifactId>
            <version>1.6.0</version>
            <executions>
                <execution>
                    <goals>
                    <goal>java</goal>
                    </goals>
                </execution>
            </executions>
            <configuration>
            <systemProperties>
                <systemProperty>
                    <key>spark.master</key>
                    <value>local[*]</value>
                </systemProperty>
                <systemProperty>
                    <key>spark.app.name</key>
                    <value>localrun</value>
                </systemProperty>
                <systemProperty>
                    <key>org.xerial.snappy.lib.name</key>
                    <value>libsnappyjava.jnilib</value>
                </systemProperty>
            </systemProperties>
            </configuration>
        </plugin>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-enforcer-plugin</artifactId>
            <version>3.0.0-M2</version>
            <executions>
                <execution>
                    <id>enforce-maven</id>
                    <goals>
                        <goal>enforce</goal>
                    </goals>
                    <configuration>
                        <rules>
```

```
                                <requireMavenVersion>
                                    <version>3.5.3</version>
                                </requireMavenVersion>
                            </rules>
                        </configuration>
                    </execution>
                </executions>
            </plugin>
  <!-- The shade plugin will be helpful in building a uberjar or fatjar.
  You can use this jar in the AWS Glue runtime environment. For more information, see
 https://maven.apache.org/plugins/maven-shade-plugin/ -->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-shade-plugin</artifactId>
                <version>3.2.4</version>
                <configuration>
                    <!-- any other shade configurations -->
                </configuration>
                <executions>
                    <execution>
                        <phase>package</phase>
                        <goals>
                            <goal>shade</goal>
                        </goals>
                    </execution>
                </executions>
            </plugin>
        </plugins>
    </build>
</project>
```

Menjalankan skrip Scala ETL Anda

Jalankan perintah berikut dari direktori akar proyek Maven untuk menjalankan skrip ETL Scala Anda.

```
mvn exec:java -Dexec.mainClass="mainClass" -Dexec.args="--JOB-NAME jobName"
```

Ganti *`mainClass`* dengan nama kelas yang memenuhi syarat dari kelas utama skrip. Ganti *`jobName`* dengan nama tugas yang diinginkan.

Mengkonfigurasi lingkungan pengujian

Untuk contoh mengkonfigurasi sebuah lingkungan pengujian lokal, lihat artikel blog berikut:

- Membangun pipa AWS Glue ETL secara lokal tanpa akun AWS
- Mengembangkan pekerjaan AWS Glue ETL secara lokal menggunakan wadah

Jika Anda ingin menggunakan titik akhir pengembangan atau notebook untuk menguji skrip ETL Anda, lihat Mengembangkan skrip menggunakan titik akhir pengembangan.

> **Note**
>
> Titik akhir pengembangan tidak didukung untuk digunakan dengan tugas AWS Glue versi 2.0. Untuk informasi selengkapnya, lihat Menjalankan Tugas Spark ETL dengan Waktu Pemulaian yang Dikurangi.

Titik akhir pengembangan

> **Note**
>
> Pengalaman konsol untuk titik akhir dev telah dihapus pada 31 Maret 2023. Membuat, memperbarui, dan memantau titik akhir dev masih tersedia melalui dan API titik akhir pengembangan CLI AWS Glue.

Kami sangat menyarankan untuk bermigrasi dari titik akhir dev ke sesi interaktif karena alasan yang tercantum di bawah ini. Untuk tindakan yang diperlukan tentang cara bermigrasi dari titik akhir dev ke sesi interaktif, lihat Memigrasi dari titik akhir dev ke sesi interaktif.

Deskripsi	Titik akhir Dev	Sesi interaktif
Dukungan versi Glue	Mendukung AWS Glue versi 0.9 dan 1.0	Mendukung AWS Glue versi 2.0 dan yang lebih baru
Titik akhir Dev tidak tersedia di Asia Pasifik (Jakarta) (`ap-southeast-3`), Timur Tengah (UEA) (), Eropa (Spanyol`me-central-1`) (), Eropa (Zurich`eu-south-`	Sesi interaktif saat ini tidak tersedia di wilayah Timur Tengah (UEA`me-central-1`) (), tetapi mungkin tersedia nanti	

Deskripsi	Titik akhir Dev	Sesi interaktif
2) (eu-central-2), atau wilayah baru lainnya di masa mendatang		
Metode akses ke cluster Spark	Mendukung SSH, shell REPL, notebook Jupyter, IDE (mis.) PyCharm	mendukung AWS Glue Studio notebook, notebook Jupyter, berbagai IDE (misalnya, Visual Studio Code, PyCharm), dan notebook SageMaker
Saatnya kueri pertama	Membutuhkan 10-15 menit untuk menyiapkan cluster Spark	Dapat memakan waktu hingga 1 menit untuk menyiapkan cluster Spark fana
Model harga	AWSbiaya untuk titik akhir pengembangan berdasarkan waktu titik akhir disediakan dan jumlah DPU. Titik akhir pengembangan tidak habis waktu. Ada durasi penagihan minimum 10 menit untuk setiap titik akhir pengembangan yang disediakan. Selain itu, AWS biaya untuk notebook Jupyter di instans Amazon EC2, SageMaker dan notebook saat Anda mengonfigurasinya dengan titik akhir dev.	AWSbiaya untuk sesi interaktif berdasarkan waktu sesi aktif dan jumlah DPU. sesi interaktif memiliki batas waktu idle yang dapat dikonfigurasi. AWS Glue Studionotebook menyediakan antarmuka bawaan untuk sesi interaktif dan ditawarkan tanpa biaya tambahan. Ada durasi penagihan minimum 1 menit untuk setiap sesi interaktif. AWS Glue Studionotebook menyediakan antarmuka bawaan untuk sesi interaktif dan ditawarkan tanpa biaya tambahan
Pengalaman konsol	Hanya tersedia melalui CLI dan API	Tersedia melalui AWS Glue konsol, CLI, dan API

Migrasi dari titik akhir dev ke sesi interaktif

Gunakan daftar periksa berikut untuk menentukan metode yang tepat untuk bermigrasi dari titik akhir dev ke sesi interaktif.

Apakah skrip Anda bergantung pada fitur spesifik AWS Glue 0.9 atau 1.0 (misalnya, HDFS, YARN, dll.)?

Jika jawabannya ya, lihat Migrasi AWS Glue pekerjaan ke AWS Glue versi 3.0. untuk mempelajari cara bermigrasi dari Glue 0.9 atau 1.0 ke Glue 3.0 dan yang lebih baru.

Metode mana yang Anda gunakan untuk mengakses titik akhir dev Anda?

Jika Anda menggunakan metode ini	Kemudian lakukan ini
SageMaker notebook, notebook Jupyter, atau JupyterLab	Migrasi ke AWS Glue Studionotebook dengan mengunduh `.ipynb` file di Jupyter dan buat pekerjaan AWS Glue Studio notebook baru dengan mengunggah file. `.ipynb` Atau, Anda juga dapat menggunakan SageMaker Studio dan memilih AWS Glue kernel.
Notebook Zeppelin	Konversi notebook ke notebook Jupyter secara manual dengan menyalin dan menempelk an kode atau secara otomatis menggunak an konverter pihak ketiga seperti ze2nb. Kemudian, gunakan notebook di AWS Glue Studio notebook atau SageMaker Studio.
IDE	Lihat AWS Gluepekerjaan Penulis dengan PyCharm menggunakan sesi AWS Glue interaktif, atau Menggunakan sesi interaktif dengan Microsoft Visual Studio Code.
REPL	Instal `aws-glue-session package` secara lokal, lalu jalankan perintah berikut: • Untuk Python: `jupyter console --kernal glue_pyspark`

Jika Anda menggunakan metode ini	Kemudian lakukan ini
	• Untuk Scala: `jupyter console --kernal glue_spark`
SSH	Tidak ada opsi yang sesuai pada sesi interaktif. Atau, Anda dapat menggunakan gambar Docker. Untuk mempelajari lebih lanjut, lihat Mengembangkan menggunakan image Docker.

Bagian berikut memberikan informasi tentang penggunaan titik akhir dev untuk mengembangkan pekerjaan di AWS Glue versi 1.0.

Topik

- Mengembangkan skrip menggunakan titik akhir pengembangan
- Mengelola buku catatan

Mengembangkan skrip menggunakan titik akhir pengembangan

> **Note**
>
> Titik Akhir Pengembangan hanya didukung untuk versi AWS Glue sebelum 2.0. Untuk lingkungan interaktif tempat Anda dapat membuat dan menguji skrip ETL, gunakan Notebook di Studio. AWS Glue

AWS Glue dapat menciptakan lingkungan—yang dikenal sebagai titik akhir pengembangan—yang dapat Anda gunakan untuk mengembangkan dan menguji skrip extract, transform, and load (ETL) Anda. Anda dapat membuat, mengedit, dan menghapus titik akhir pengembangan menggunakan konsol AWS Glue atau API.

Mengelola lingkungan pengembangan Anda

Ketika Anda membuat sebuah titik akhir pengembangan, Anda memberikan nilai-nilai konfigurasi untuk penyediaan lingkungan pengembangan. Nilai-nilai ini memberitahu cara AWS Glue menyiapkan jaringan sehingga Anda dapat mengakses titik akhir dengan aman, dan sehingga titik akhir Anda dapat mengakses penyimpanan data Anda.

Anda kemudian dapat membuat sebuah notebook yang terhubung ke titik akhir, dan menggunakan notebook Anda untuk penulis dan menguji skrip ETL Anda. Ketika Anda sudah puas dengan hasil proses pengembangan Anda, Anda dapat membuat tugas ETL yang menjalankan skrip Anda. Dengan proses ini, Anda dapat menambahkan fungsi dan melakukan debug pada skrip Anda secara interaktif.

Ikuti tutorial di bagian ini untuk mempelajari cara menggunakan titik akhir pengembangan Anda dengan notebook.

Topik

Alur kerja titik akhir pengembangan

Untuk menggunakan sebuah titik akhir pengembangan AWS Glue, Anda dapat mengikuti alur kerja ini:

1. Buat titik akhir pengembangan menggunakan API. Titik akhir tersebut diluncurkan di virtual private cloud (VPC) dengan grup keamanan yang Anda tetapkan.
2. API melakukan polling titik akhir pengembangan hingga disediakan dan siap untuk bekerja. Ketika sudah siap, connect ke titik akhir pengembangan dengan menggunakan salah satu metode berikut untuk membuat dan menguji skrip AWS Glue.

    - Buat SageMaker buku catatan di akun Anda. Untuk informasi lebih lanjut tentang cara membuat sebuah notebook, lihat the section called “Menulis kode dengan notebook AWS Glue Studio”.
    - Buka sebuah jendela terminal untuk connect langsung ke titik akhir pengembangan.

- Jika Anda memiliki edisi profesional IDE JetBrains PyCharm Python, hubungkan ke titik akhir pengembangan dan gunakan untuk mengembangkan secara interaktif. Jika Anda menyisipkan `pydevd` pernyataan dalam skrip Anda, PyCharm dapat mendukung breakpoint jarak jauh.

3. Setelah Anda selesai melakukan debug dan pengujian di titik akhir pengembangan Anda, Anda dapat menghapusnya.

Bagaimana titik akhir AWS Glue pengembangan bekerja dengan notebook SageMaker

Salah satu cara umum untuk mengakses titik akhir pengembangan Anda adalah dengan menggunakan Jupyter di notebook. SageMaker Notebook Jupyter adalah sebuah aplikasi web sumber terbuka yang banyak digunakan dalam visualisasi, analitik, machine learning, dll. AWS Glue SageMaker Notebook memberi Anda pengalaman notebook Jupyter dengan titik akhir AWS Glue pengembangan. Di AWS Glue SageMaker notebook, lingkungan notebook Jupyter sudah dikonfigurasi sebelumnya dengan SparkMagic, plugin Jupyter open source untuk mengirimkan pekerjaan Spark ke cluster Spark jarak jauh. Apache Livy adalah sebuah layanan yang memungkinkan interaksi dengan klaster Spark jarak jauh melalui API REST. Di AWS Glue SageMaker notebook, SparkMagic dikonfigurasi untuk memanggil REST API terhadap server Livy yang berjalan pada titik akhir AWS Glue pengembangan.

Alur teks berikut menjelaskan bagaimana masing-masing komponen bekerja:

AWS Glue SageMaker notebook: (Jupyter → SparkMagic) → (jaringan) → titik akhir AWS Glue pengembangan: (Apache Livy → Apache Spark)

Setelah Anda menjalankan skrip Spark yang ditulis di setiap paragraf pada notebook Jupyter, kode Spark dikirimkan ke server Livy melalui SparkMagic, kemudian pekerjaan Spark bernama “Livy-session-N” berjalan di cluster Spark. Tugas ini disebut sesi Livy. Tugas Spark akan berjalan saat sesi notebook masih hidup. Tugas Spark akan diakhiri saat Anda mematikan kernel Jupyter dari notebook, atau saat sesi waktunya habis. Satu tugas Spark diluncurkan untuk setiap file (.ipynb) notebook.

Anda dapat menggunakan satu titik akhir AWS Glue pengembangan dengan beberapa instance SageMaker notebook. Anda dapat membuat beberapa file notebook di setiap instance SageMaker notebook. Saat Anda membuka setiap file notebook dan menjalankan paragraf, maka sesi Livy diluncurkan per file notebook di cluster Spark via. SparkMagic Setiap sesi Livy sesuai dengan saru tugas Spark.

Perilaku default untuk titik akhir AWS Glue pengembangan dan notebook SageMaker

Tugas Spark berjalan berdasarkan Konfigurasi Spark. Ada beberapa cara untuk mengatur konfigurasi Spark (misalnya, konfigurasi cluster Spark, konfigurasi, SparkMagic dll.).

Secara default, Spark mengalokasikan sumber daya klaster untuk sesi Livy berdasarkan konfigurasi klaster Spark. Di titik akhir pengembangan AWS Glue, konfigurasi klaster tergantung pada jenis pekerja. Berikut adalah tabel yang menjelaskan konfigurasi umum per jenis pekerja.

	Standar	G.1X	G.2X
`spark.driver.memory`	5G	10G	20G
`spark.executor.memory`	5G	10G	20G
`spark.executor.cores`	4	8	16
`spark.dynamicAllocation.enabled`	BETUL	BETUL	BETUL

Jumlah maksimum pelaksana Spark secara otomatis dihitung berdasarkan kombinasi DPU (atau `NumberOfWorkers`) dan jenis pekerja.

	Standar	G.1X	G.2X
Jumlah maksimal pelaksana Spark	`(DPU - 1) * 2 - 1`	`(NumberOfWorkers - 1)`	`(NumberOfWorkers - 1)`

Misalnya, jika titik akhir pengembangan Anda memiliki 10 pekerja dan jenis pekerja-nya adalah `G.1X`, maka Anda akan memiliki 9 pelaksana Spark dan seluruh klaster akan memiliki memori pelaksana sebesar 90G karena setiap pelaksana akan memiliki memori 10G.

Terlepas dari jenis pekerja yang ditentukan, alokasi sumber daya dinamis Spark akan dinyalakan. Jika set data cukup besar, Spark dapat mengalokasikan semua pelaksana untuk satu sesi Livy tunggal karena `spark.dynamicAllocation.maxExecutors` tidak diatur secara default. Ini berarti bahwa sesi Livy lainnya pada titik akhir pengembangan yang sama akan menunggu untuk meluncurkan pelaksana baru. Jika set data kecil, maka Spark akan dapat mengalokasikan pelaksana untuk beberapa sesi Livy pada saat yang sama.

> **Note**
>
> Untuk informasi lebih lanjut tentang bagaimana sumber daya dialokasikan dalam kasus penggunaan yang berbeda dan bagaimana Anda menetapkan konfigurasi untuk mengubah perilaku, lihat Konfigurasi lanjutan: berbagi titik akhir pengembangan di antara banyak pengguna.

Menambahkan titik akhir pengembangan

Gunakan titik akhir pengembangan untuk mengembangkan dan menguji skrip extract, transform, and load (ETL) di AWS Glue. Bekerja dengan titik akhir pengembangan hanya tersedia melalui. AWS Command Line Interface

1. Dalam jendela baris perintah, masukkan sebuah perintah yang serupa dengan perintah berikut.

   ```
   aws glue create-dev-endpoint --endpoint-name "endpoint1" --role-arn
    "arn:aws:iam::account-id:role/role-name" --number-of-nodes "3" --glue-version
    "1.0" --arguments '{"GLUE_PYTHON_VERSION": "3"}' --region "region-name"
   ```

   Perintah ini menentukan AWS Glue versi 1.0. Karena versi ini mendukung Python 2 dan Python 3, Anda dapat menggunakan parameter `arguments` untuk menunjukkan versi Python yang diinginkan. Jika parameter `glue-version` dihilangkan, maka AWS Glue versi 0.9 digunakan. Untuk informasi selengkapnya tentang versi AWS Glue, lihat Glue version job property.

   Untuk informasi tentang parameter baris perintah tambahan, lihat create-dev-endpointdi Referensi AWS CLI Perintah.

2. (Opsional) Masukkan perintah berikut untuk memeriksa status dari titik akhir pengembangan. Ketika status berubah ke READY, titik akhir pengembangan siap digunakan.

```
aws glue get-dev-endpoint --endpoint-name "endpoint1"
```

Mengakses titik akhir pengembangan Anda

Saat Anda membuat sebuah titik akhir pengembangan di virtual private cloud (VPC), AWS Glue hanya mengembalikan alamat IP privat saja. Bidang alamat IP publik tidak diisi. Saat Anda membuat titik akhir pengembangan non-VPC, AWS Glue hanya mengembalikan alamat IP publik saja.

Jika titik akhir pengembangan Anda memiliki sebuah Alamat Publik, maka konfirmasikan bahwa alamat tersebut dapat dijangkau dengan SSH kunci privat untuk titik akhir pengembangan, seperti dalam contoh berikut.

```
ssh -i dev-endpoint-private-key.pem glue@public-address
```

Seandainya titik akhir pengembangan Anda memiliki sebuah Alamat privat, maka subnet VPC Anda dapat diretukan dari internet publik, dan grup keamanannya memungkinkan akses masuk dari klien Anda. Dalam kasus ini, ikuti langkah-langkah berikut untuk melampirkan alamat IP elastis ke titik akhir pengembangan untuk memungkinkan akses dari internet.

> **Note**
>
> Jika Anda ingin menggunakan alamat IP elastis, maka subnet yang sedang digunakan memerlukan gateway internet yang dikaitkan melalui tabel rute.

Untuk mengakses sebuah titik akhir pengembangan dengan melampirkan alamat IP Elastis

1. Buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pilih Titik akhir pengembangan, dan arahkan ke halaman detail titik akhir pengembangan. Catat Alamat privat untuk digunakan di langkah berikutnya.
3. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
4. Pada panel navigasi, pada Jaringan & Keamanan, pilih Antarmuka jaringan.

5. Cari DNS privat (IPv4) yang sesuai dengan Alamat privat pada halaman detail titik akhir pengembangan konsol AWS Glue.

   Anda mungkin perlu memodifikasi kolom yang ditampilkan pada konsol Amazon EC2 Anda. Catat ID antarmuka jaringan (ENI) untuk alamat ini (misalnya, `eni-12345678`).
6. Di konsol Amazon EC2, pada Jaringan & Keamanan, pilih IP elastis.
7. Pilih Alokasikan alamat baru, lalu pilih Alokasikan untuk mengalokasikan alamat IP Elastis baru.
8. Pada halaman IP elastis, pilih IP elastis yang baru saja dialokasikan. Pilih Tindakan, Kaitkan alamat.
9. Pada halaman Kaitkan alamat, lakukan hal berikut:

   - Untuk Jenis sumber daya, pilih Antarmuka jaringan.
   - Pada kotak Antarmuka jaringan, masukkan ID antarmuka jaringan (ENI) untuk alamat privat.
   - Pilih Kaitkan.
10. Konfirmasi bahwa alamat IP Elastis yang baru saja dikaitkan dapat dijangkau dengan kunci privat SSH yang dikaitkan dengan titik akhir pengembangan, seperti dalam contoh berikut.

    ```
    ssh -i dev-endpoint-private-key.pem glue@elastic-ip
    ```

    Untuk informasi tentang menggunakan sebuah host bastion untuk mendapatkan akses SSH ke alamat privat titik akhir pengembangan, lihat posting Blog Keamanan AWS Connect Dengan Aman ke Instans Linux yang Berjalan di Amazon VPC Privat.

Tutorial: Siapkan notebook Jupyter JupyterLab untuk menguji dan men-debug skrip ETL

Dalam tutorial ini, Anda menghubungkan notebook Jupyter saat JupyterLab berjalan di mesin lokal Anda ke titik akhir pengembangan. Anda melakukan ini sehingga Anda dapat secara interaktif menjalankan, melakukan debug, dan menguji skrip extract, transform, and load (ETL) AWS Glue sebelum men-deploy-nya. Tutorial ini menggunakan penerusan port Secure Shell (SSH) untuk menghubungkan mesin lokal Anda ke titik akhir pengembangan AWS Glue. Untuk informasi selengkapnya, lihat Penerusan port di Wikipedia.

Langkah 1: Instal JupyterLab dan Sparkmagic

Anda dapat menginstal JupyterLab dengan menggunakan `conda` atau`pip`. `conda`adalah sistem manajemen paket open-source dan sistem manajemen lingkungan yang berjalan di Windows, macOS, dan Linux. `pip`adalah penginstal paket untuk Python.

Jika Anda menginstal di macOS, maka Anda harus menginstal Xcode sebelum Anda dapat menginstal Sparkmagic.

1. Instal JupyterLab, Sparkmagic, dan ekstensi terkait.

```
$ conda install -c conda-forge jupyterlab
$ pip install sparkmagic
$ jupyter nbextension enable --py --sys-prefix widgetsnbextension
$ jupyter labextension install @jupyter-widgets/jupyterlab-manager
```

2. Periksa direktori `sparkmagic` dari `Location`.

```
$ pip show sparkmagic | grep Location
Location: /Users/username/.pyenv/versions/anaconda3-5.3.1/lib/python3.7/site-
packages
```

3. Ubah direktori Anda ke direktori yang dikembalikan`Location`, dan instal kernel untuk Scala dan. PySpark

```
$ cd /Users/username/.pyenv/versions/anaconda3-5.3.1/lib/python3.7/site-packages
$ jupyter-kernelspec install sparkmagic/kernels/sparkkernel
$ jupyter-kernelspec install sparkmagic/kernels/pysparkkernel
```

4. Mengunduh sampel file `config`.

```
$ curl -o ~/.sparkmagic/config.json https://raw.githubusercontent.com/jupyter-
incubator/sparkmagic/master/sparkmagic/example_config.json
```

   Dalam file konfigurasi ini, Anda dapat mengkonfigurasi parameter terkait Spark seperti `driverMemory` dan `executorCores`.

Langkah 2: Mulai JupyterLab

Ketika Anda mulai JupyterLab, browser web default Anda secara otomatis dibuka, dan URL `http://localhost:8888/lab/workspaces/{workspace_name}` ditampilkan.

```
$ jupyter lab
```

Langkah 3: Memulai penerusan port SSH untuk terhubung ke titik akhir pengembangan Anda

Selanjutnya, gunakan penerusan port lokal SSH untuk meneruskan port lokal (di sini, 8998) ke tujuan jarak jauh yang ditentukan oleh AWS Glue (`169.254.76.1:8998`).

1. Buka jendela terminal terpisah yang memberikan Anda akses ke SSH. Pada Microsoft Windows, Anda dapat menggunakan shell BASH yang disediakan oleh Git untuk Windows, atau Anda dapat menginstal Cygwin.
2. Jalankan perintah SSH berikut, dan lakukan modifikasi sebagai berikut:

   - Ganti *`private-key-file-path`* dengan path ke file `.pem` yang berisi kunci privat yang sesuai dengan kunci publik yang Anda gunakan untuk membuat titik akhir pengembangan Anda.
   - Jika Anda sedang meneruskan port yang berbeda dari 8998, ganti 8998 dengan nomor port yang sebenarnya sedang Anda gunakan secara lokal. Alamat `169.254.76.1:8998` adalah port jarak jauh dan tidak diubah oleh Anda.
   - Ganti *`dev-endpoint-public-dns`* dengan alamat DNS publik dari titik akhir pengembangan Anda. Untuk menemukan alamat ini, arahkan ke titik akhir pengembangan Anda di konsol AWS Glue, pilih nama, dan salin Alamat publik yang tercantum pada halaman Detail titik akhir.

   ```
   ssh -i private-key-file-path -NTL 8998:169.254.76.1:8998 glue@dev-endpoint-public-dns
   ```

   Anda mungkin akan melihat pesan peringatan seperti yang berikut ini:

   ```
   The authenticity of host 'ec2-xx-xxx-xxx-xx.us-west-2.compute.amazonaws.com
    (xx.xxx.xxx.xx)'
   can't be established.  ECDSA key fingerprint is SHA256:4e97875Brt+1wKzRko
   +JflSnp21X7aTP3BcFnHYLEts.
   Are you sure you want to continue connecting (yes/no)?
   ```

   Masuk **`yes`** dan biarkan jendela terminal terbuka saat Anda menggunakan JupyterLab.
3. Periksa apakah penerusan port SSH berfungsi dengan baik dengan titik akhir pengembangan.

```
$ curl localhost:8998/sessions
{"from":0,"total":0,"sessions":[]}
```

Langkah 4: Jalankan fragmen skrip sederhana dalam paragraf notebook

Sekarang notebook Anda JupyterLab harus bekerja dengan titik akhir pengembangan Anda. Masukkan fragmen skrip berikut ke notebook Anda dan jalankan.

1. Periksa apakah Spark berhasil berjalan. Perintah berikut menginstruksikan Spark untuk menghitung 1 dan kemudian mencetak nilainya.

   ```
   spark.sql("select 1").show()
   ```

2. Periksa apakah integrasi AWS Glue Data Catalog bekerja. Perintah berikut mencantumkan tabel dalam Katalog Data.

   ```
   spark.sql("show tables").show()
   ```

3. Periksa apakah fragmen skrip sederhana yang menggunakan perpustakaan AWS Glue bekerja.

   Skrip berikut menggunakan metadata tabel `persons_json` dalam AWS Glue Data Catalog untuk membuat `DynamicFrame` dari data sampel Anda. Ia kemudian mencetak jumlah item dan skema data ini.

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create a Glue context
glueContext = GlueContext(SparkContext.getOrCreate())

# Create a DynamicFrame using the 'persons_json' table
persons_DyF = glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="persons_json")

# Print out information about *this* data
print("Count:  ", persons_DyF.count())
persons_DyF.printSchema()
```

Output dari skrip tersebut adalah sebagai berikut.

```
Count:  1961
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

Pemecahan Masalah

- Selama instalasi JupyterLab, jika komputer Anda berada di belakang proxy perusahaan atau firewall, Anda mungkin mengalami kesalahan HTTP dan SSL karena profil keamanan khusus yang dikelola oleh departemen TI perusahaan.

  Berikut ini adalah contoh kesalahan khas yang terjadi ketika `conda` tidak dapat terhubung ke repositorinya sendiri:

```
CondaHTTPError: HTTP 000 CONNECTION FAILED for url <https://repo.anaconda.com/pkgs/
main/win-64/current_repodata.json>
```

Ini mungkin terjadi karena perusahaan Anda dapat memblokir koneksi ke repositori yang banyak digunakan di Python dan komunitas. JavaScript Untuk informasi selengkapnya, lihat Masalah Instalasi di JupyterLab situs web.

- Jika Anda menemukan kesalahan koneksi ditolak saat mencoba connect ke titik akhir pengembangan, Anda mungkin menggunakan titik akhir pengembangan yang sudah kedaluwarsa. Cobalah membuat sebuah titik akhir pengembangan baru dan hubungkan kembali.

Tutorial: Gunakan buku SageMaker catatan dengan titik akhir pengembangan Anda

DiAWS Glue, Anda dapat membuat titik akhir pengembangan dan kemudian membuat SageMaker buku catatan untuk membantu mengembangkan skrip ETL dan machine learning Anda. SageMaker Notebook adalah instance komputasi pembelajaran mesin yang dikelola sepenuhnya yang menjalankan aplikasi Notebook Jupyter.

1. Pada konsol AWS Glue, pilih Titik akhir pengembangan untuk mengarahkan ke daftar titik akhir pengembangan.
2. Pilih kotak centang di samping nama titik akhir pengembangan yang ingin Anda gunakan, dan pada menu Tindakan, pilih Buat SageMaker buku catatan.
3. Isi halaman Membuat dan mengkonfigurasi notebook seperti berikut:

   a. Masukkan nama notebook.

   b. Pada Lampirkan ke titik akhir pengembangan, verifikasi titik akhir pengembangan.

   c. Buat atau pilih sebuah AWS Identity and Access Management (IAM) role.

      Membuat sebuah peran dianjurkan. Jika Anda menggunakan peran yang sudah ada, pastikan bahwa peran itu memiliki izin yang diperlukan. Untuk informasi selengkapnya, lihat the section called “Langkah 6: Buat kebijakan IAM untuk notebook SageMaker”.

   d. (Opsional) Pilih VPC, subnet, dan satu atau beberapa grup keamanan.

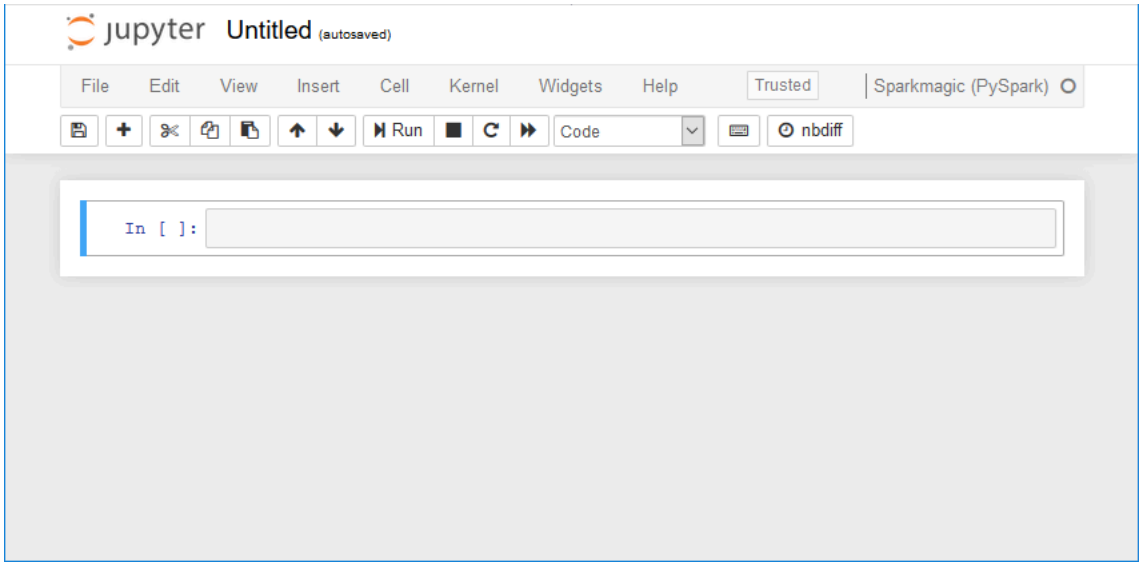
   e. (Opsional) Pilih kunci enkripsi AWS Key Management Service.

   f. (Opsional) Tambahkan tag untuk instans notebook.

4. Pilih Buat Notebook. Pada halaman Notebook, pilih ikon refresh di kanan atas, dan lanjutkan hingga Status menunjukkan `Ready`.

5. Pilih kotak centang di samping nama notebook yang baru, kemudian pilih Buka notebook.
6. Buat buku catatan baru: Pada halaman jupyter, pilih Baru, lalu pilih Sparkmagic (). PySpark

   Layar Anda sekarang akan terlihat seperti berikut ini:



7. (Opsional) Di bagian atas halaman, pilih Tanpa judul, dan beri nama untuk notebook tersebut.
8. Untuk memulai sebuah aplikasi Spark, masukkan perintah berikut ke dalam notebook, lalu di toolbar, pilih Jalankan.

```
spark
```

   Setelah penundaan sesaat, Anda akan melihat respons berikut:

```
In [1]: spark
Starting Spark application
```

ID	YARN Application ID	Kind	State	Spark UI	Driver log	Current session?
0	application_1576209965005_0001	pyspark	idle	Link	Link	✓

```
SparkSession available as 'spark'.
<pyspark.sql.session.SparkSession object at 0x7f3d54913550>
```

9. Membuat bingkai dinamis dan menjalankan kueri terhadapnya: Salin, tempel, dan jalankan kode berikut, yang memberikan output jumlah dan skema tabel `persons_json`.

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import *
```

```
glueContext = GlueContext(SparkContext.getOrCreate())
persons_DyF = glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="persons_json")
print ("Count:  ", persons_DyF.count())
persons_DyF.printSchema()
```

Tutorial: Gunakan shell REPL dengan titik akhir pengembangan Anda

DiAWS Glue, Anda dapat membuat endpoint pengembangan dan kemudian memanggil shell REPL (Read—Evaluate—Print Loop) untuk menjalankan PySpark kode secara bertahap sehingga Anda dapat men-debug skrip ETL secara interaktif sebelum menerapkannya.

Untuk menggunakan REPL pada titik akhir pengembangan, Anda harus memiliki otorisasi ke SSH ke titik akhir.

1. Pada komputer lokal Anda, buka jendela terminal yang dapat menjalankan perintah SSH, dan tempel di perintah SSH yang sudah diedit. Jalankan perintah.

   Dengan asumsi bahwa Anda menerima AWS Glue versi 1.0 dengan Python 3 untuk titik akhir pengembangan, output akan terlihat seperti ini:

   ```
   Python 3.6.8 (default, Aug  2 2019, 17:42:44)
   [GCC 4.8.5 20150623 (Red Hat 4.8.5-28)] on linux
   Type "help", "copyright", "credits" or "license" for more information.
   SLF4J: Class path contains multiple SLF4J bindings.
   SLF4J: Found binding in [jar:file:/usr/share/aws/glue/etl/jars/glue-assembly.jar!/
   org/slf4j/impl/StaticLoggerBinder.class]
   SLF4J: Found binding in [jar:file:/usr/lib/spark/jars/slf4j-log4j12-1.7.16.jar!/
   org/slf4j/impl/StaticLoggerBinder.class]
   SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
   SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
   Setting default log level to "WARN".
   To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use
    setLogLevel(newLevel).
   2019-09-23 22:12:23,071 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
    - Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading
    libraries under SPARK_HOME.
   2019-09-23 22:12:26,562 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
    - Same name resource file:/usr/lib/spark/python/lib/pyspark.zip added multiple
    times to distributed cache
   ```

```
2019-09-23 22:12:26,580 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
 - Same path resource file:///usr/share/aws/glue/etl/python/PyGlue.zip added
 multiple times to distributed cache.
2019-09-23 22:12:26,581 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
 - Same path resource file:///usr/lib/spark/python/lib/py4j-src.zip added multiple
 times to distributed cache.
2019-09-23 22:12:26,581 WARN  [Thread-5] yarn.Client (Logging.scala:logWarning(66))
 - Same path resource file:///usr/share/aws/glue/libs/pyspark.zip added multiple
 times to distributed cache.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 2.4.3
      /_/

Using Python version 3.6.8 (default, Aug  2 2019 17:42:44)
SparkSession available as 'spark'.
>>>
```

2. Uji apakah shell REPL bekerja dengan benar dengan mengetik pernyataan, `print(spark.version)`. Selama yang ditampilkan adalah versi Spark, REPL Anda sekarang siap untuk digunakan.
3. Sekarang Anda dapat mencoba mengeksekusi skrip sederhana berikut, baris demi baris, di shell:

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import *
glueContext = GlueContext(SparkContext.getOrCreate())
persons_DyF = glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="persons_json")
print ("Count:  ", persons_DyF.count())
persons_DyF.printSchema()
```

Tutorial: Siapkan PyCharm profesional dengan titik akhir pengembangan

Tutorial ini menunjukkan cara menghubungkan IDE Python PyCharm Profesional yang berjalan di mesin lokal Anda ke titik akhir pengembangan sehingga Anda dapat menjalankan, men-debug, dan menguji skrip AWS Glue ETL (ekstrak, transfer, dan muat) secara interaktif sebelum menerapkannya. Instruksi dan tangkapan layar dalam tutorial didasarkan pada versi PyCharm Profesional 2019.3.

Untuk terhubung ke titik akhir pengembangan secara interaktif, Anda harus menginstal PyCharm Profesional. Anda tidak dapat melakukan ini dengan menggunakan edisi gratis.

> **Note**
>
> Tutorial ini menggunakan Amazon S3 sebagai sumber data. Jika Anda ingin menggunakan sumber data JDBC sebagai gantinya, maka Anda harus menjalankan titik akhir pengembangan Anda di virtual private cloud (VPC). Untuk terhubung dengan SSH ke titik akhir pengembangan di sebuah VPC, Anda harus membuat terowongan SSH. Tutorial ini tidak menyertakan petunjuk untuk membuat terowongan SSH. Untuk informasi tentang cara menggunakan SSH untuk terhubung ke titik akhir pengembangan dalam sebuah VPC, lihat Connect Secara Aman ke Instans Linux yang Berjalan di Amazon VPC Privat di blog keamanan AWS.

Topik

Menghubungkan PyCharm profesional ke titik akhir pengembangan

1. Buat proyek pure-Python baru dengan nama. PyCharm `legislators`
2. Membuat file bernama `get_person_schema.py` dalam proyek tersebut dengan konten berikut:

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext


def main():
    # Create a Glue context
    glueContext = GlueContext(SparkContext.getOrCreate())

    # Create a DynamicFrame using the 'persons_json' table
    persons_DyF =
 glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="persons_json")
```

```
    # Print out information about this data
    print("Count:  ", persons_DyF.count())
    persons_DyF.printSchema()


if __name__ == "__main__":
    main()
```

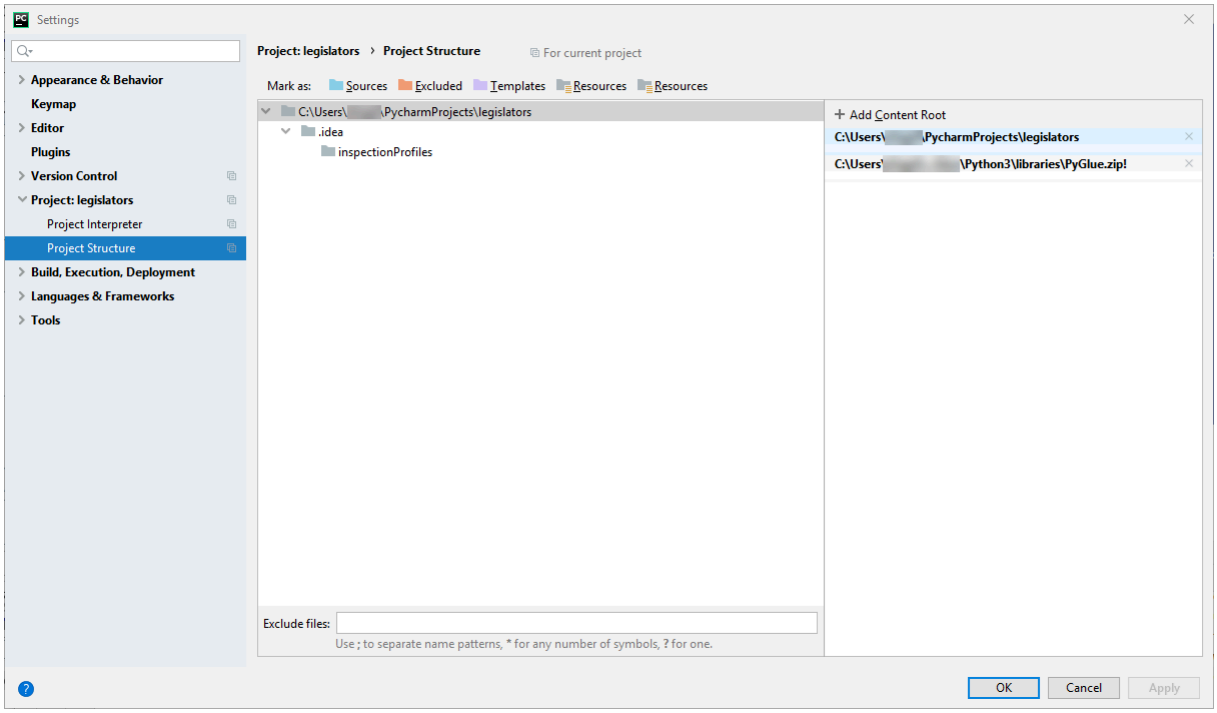
3. Lakukan salah satu dari berikut:

   - Untuk AWS Glue versi 0.9, unduh file perpustakaan Python AWS Glue, `PyGlue.zip`, dari `https://s3.amazonaws.com/aws-glue-jes-prod-us-east-1-assets/etl/python/PyGlue.zip` ke lokasi yang nyaman pada mesin lokal Anda.
   - Untuk AWS Glue versi 1.0 dan versi yang lebih baru, unduh file perpustakaan Python AWS Glue, `PyGlue.zip`, dari `https://s3.amazonaws.com/aws-glue-jes-prod-us-east-1-assets/etl-1.0/python/PyGlue.zip` ke lokasi yang nyaman pada mesin lokal Anda.

4. Tambahkan `PyGlue.zip` sebagai root konten untuk proyek Anda di PyCharm:

   - Di PyCharm, pilih File, Pengaturan untuk membuka kotak dialog Pengaturan. (Anda juga dapat menekan `Ctrl+Alt+S`.)
   - Perluas proyek `legislators` dan pilih Struktur Proyek. Kemudian di panel kanan, pilih +Tambah Root Konten.
   - Navigasi ke lokasi tempat Anda menyimpan `PyGlue.zip`, pilih, lalu pilih Terapkan.

   Layar Pengaturan akan terlihat seperti berikut ini:



Biarkan kotak dialog Pengaturan terbuka setelah Anda memilih Terapkan.

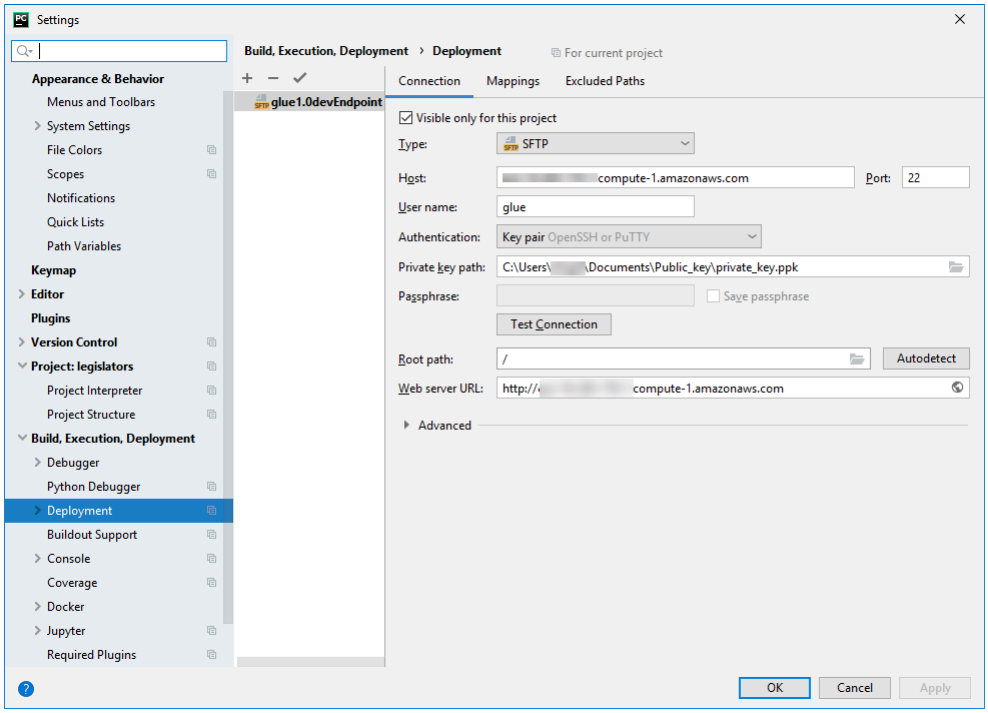
5. Konfigurasikan opsi penerapan untuk mengunggah skrip lokal ke titik akhir pengembangan Anda menggunakan SFTP (kemampuan ini hanya tersedia di Profesional): PyCharm

   - Di kotak dialog Pengaturan, perluas bagian Membangun, Eksekusi, Deployment. Pilih subbagian Deployment.
   - Pilih ikon + di bagian atas panel tengah untuk menambahkan sebuah server baru. Atur Jenis-nya ke SFTP dan beri nama.
   - Atur Host SFTP ke Alamat Publik dari titik akhir pengembangan Anda, seperti yang tercantum pada halaman detailnya. (Pilih nama titik akhir pengembangan Anda di konsol AWS Glue untuk menampilkan halaman detail). Untuk titik akhir pengembangan yang berjalan di sebuah VPC, atur Host SFTP ke alamat host dan atur port lokal terowongan SSH Anda ke titik akhir pengembangan.
   - Atur Nama pengguna ke `glue`.
   - Atur Jenis autentikasi ke Pasangan kunci (OpenSSH atau Putty). Atur File kunci privat dengan menelusuri ke lokasi di mana file kunci privat dari titik akhir pengembangan Anda terletak. Perhatikan bahwa PyCharm hanya mendukung tipe kunci OpenSSH DSA, RSA dan ECDSA, dan tidak menerima kunci dalam format pribadi Putty. Anda dapat menggunakan up-to-date versi `ssh-keygen` untuk menghasilkan tipe pasangan kunci yang PyCharm menerima, menggunakan sintaks seperti berikut:

```
ssh-keygen -t rsa -f <key_file_name> -C "<your_email_address>"
```

- Pilih Uji koneksi, dan biarkan koneksi diuji. Jika koneksi berhasil, pilih Terapkan.

Layar Pengaturan sekarang akan terlihat seperti berikut ini:

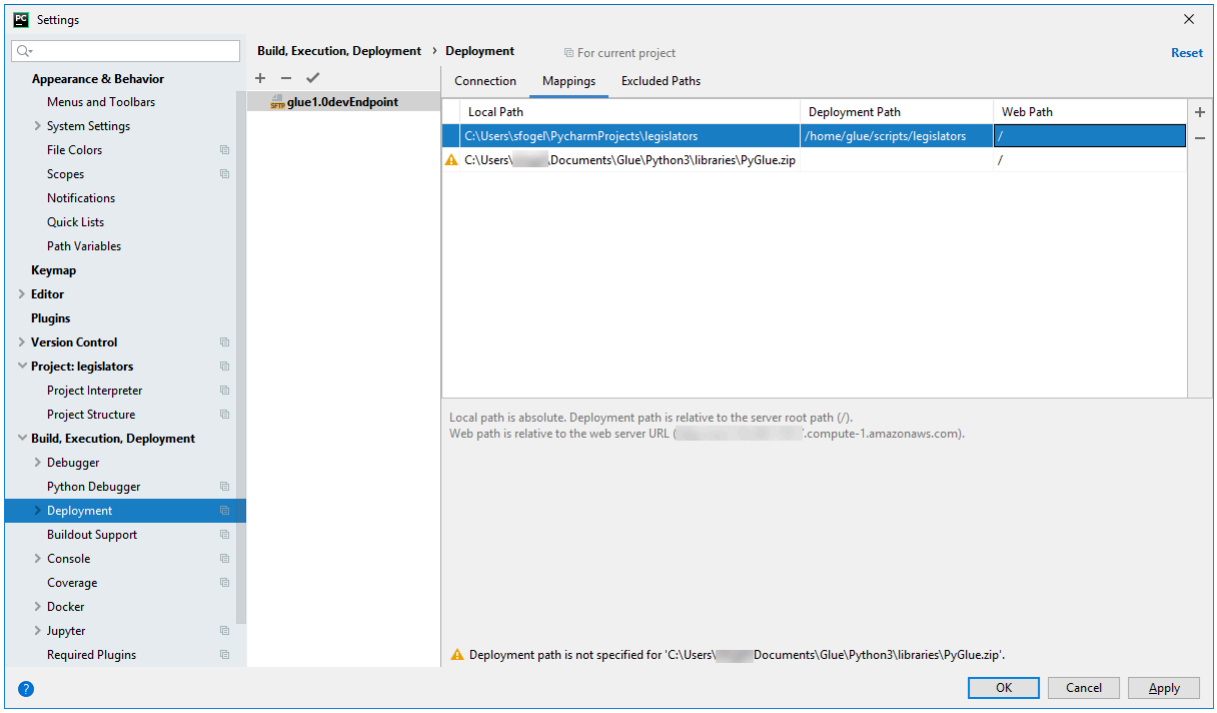


Sekali lagi, biarkan kotak dialog Pengaturan terbuka setelah Anda memilih Terapkan.

6. Memetakan direktori lokal ke direktori jarak jauh untuk deployment:

   - Dalam panel kanan di halaman Deployment, pilih tab tengah di bagian atas, yang berlabel Pemetaan.
   - Di kolom Path Deployment, masukkan sebuah path di bawah `/home/glue/scripts/` untuk deployment path proyek Anda. Misalnya: `/home/glue/scripts/legislators`.
   - Pilih Apply (Terapkan).

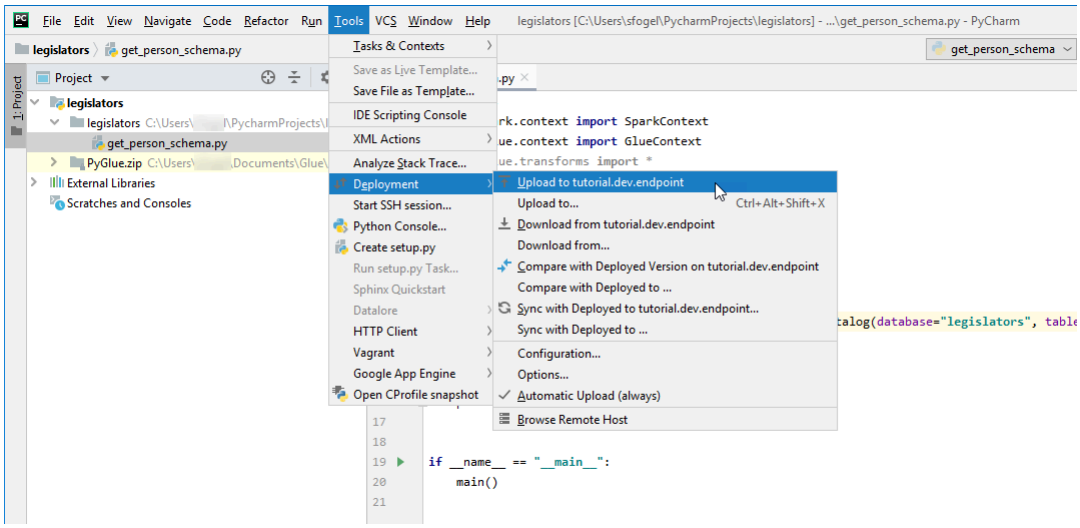
   Layar Pengaturan sekarang akan terlihat seperti berikut ini:



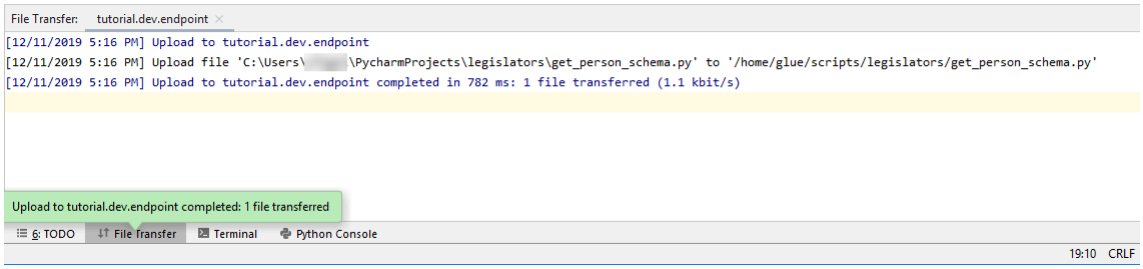
Pilih OK untuk menutup kotak dialog Pengaturan.

Menerapkan skrip ke titik akhir pengembangan Anda

1. Pilih Alat, Deployment, kemudian pilih nama di mana Anda menyiapkan titik akhir pengembangan Anda, seperti yang ditunjukkan pada gambar berikut ini:



Setelah skrip Anda di-deploy, bagian bawah layar akan terlihat seperti berikut ini:



2. Pada bilah menu, pilih Alat, Deployment, Unggah Otomatis (selalu). Pastikan bahwa tanda centang muncul di samping Unggah Otomatis (selalu).

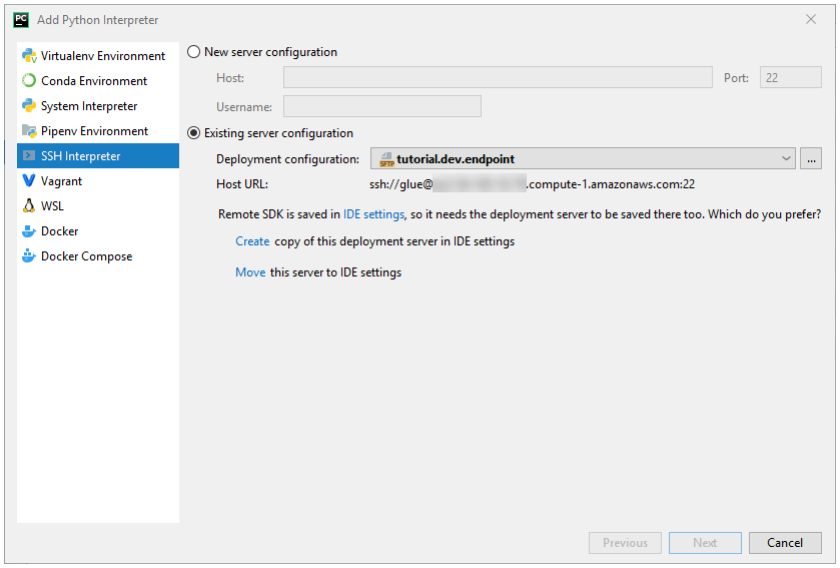
   Ketika opsi ini diaktifkan, PyCharm secara otomatis mengunggah file yang diubah ke titik akhir pengembangan.

Mengkonfigurasi penerjemah jarak jauh

Konfigurasikan PyCharm untuk menggunakan interpreter Python pada titik akhir pengembangan.

1. Dari menu File, pilih Pengaturan.
2. Perluas legislator proyek dan pilih Penerjemah proyek.
3. Pilih ikon roda gigi di samping Penerjemah proyek, dan kemudian pilih Tambahkan.
4. Di kotak dialog Tambahkan Penerjemah Python, di panel sebelah kiri, pilih Penerjemah SSH.
5. Pilih Konfigurasi server yang ada, dan di daftar Konfigurasi deployment, pilih konfigurasi Anda.

   Layar Anda akan terlihat seperti gambar berikut.



6. Pilih Pindahkan server ini ke pengaturan IDE, lalu pilih Selanjutnya.

7. Di bidang Penerjemah, ubah path ke `/usr/bin/gluepython` jika Anda menggunakan Python 2, atau ke `/usr/bin/gluepython3` jika Anda menggunakan Python 3. Lalu pilih Selesai.

Menjalankan skrip Anda di titik akhir pengembangan

Untuk menjalankan skrip:

- Di panel sebelah kiri, klik kanan nama file dan pilih Jalankan '***<filename>***'.

  Setelah serangkaian pesan, output akhir seharusnya menunjukkan jumlah dan skema.

```
Count:  1961
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

```
Process finished with exit code 0
```

Anda sekarang disiapkan untuk melakukan debug pada skrip Anda dari jarak jauh pada titik akhir pengembangan Anda.

Konfigurasi lanjutan: berbagi titik akhir pengembangan di antara banyak pengguna

Bagian ini menjelaskan bagaimana Anda dapat memanfaatkan titik akhir pengembangan dengan SageMaker notebook dalam kasus penggunaan umum untuk berbagi titik akhir pengembangan di antara beberapa pengguna.

Konfigurasi penyewaan tunggal

Dalam kasus penggunaan satu penyewa tunggal, untuk menyederhanakan pengalaman developer dan untuk menghindari perebutan sumber daya, dianjurkan agar Anda mengharuskan setiap developer menggunakan titik akhir pengembangan mereka sendiri dengan ukuran yang disesuaikan untuk proyek yang mereka kerjakan. Hal ini juga menyederhanakan keputusan yang berkaitan dengan jenis pekerja dan jumlah DPU dengan menyerahkannya pada kebijaksanaan developer dan proyek yang sedang mereka kerjakan.

Anda tidak perlu mengurus alokasi sumber daya kecuali Anda menjalankan beberapa file notebook secara bersamaan. Jika Anda menjalankan kode dalam beberapa file notebook sekaligus, maka beberapa sesi Livy akan diluncurkan bersamaan. Untuk memisahkan konfigurasi klaster Spark agar menjalankan beberapa sesi Livy pada saat yang sama, Anda dapat mengikuti langkah-langkah yang diperkenalkan dalam kasus penggunaan multi penghuni.

Misalnya, jika titik akhir pengembangan Anda memiliki 10 pekerja dan jenis pekerja-nya adalah `G.1X`, maka Anda akan memiliki 9 pelaksana Spark dan seluruh klaster akan memiliki memori pelaksana sebesar 90G karena setiap pelaksana akan memiliki memori 10G.

Terlepas dari jenis pekerja yang ditentukan, alokasi sumber daya dinamis Spark akan dinyalakan. Jika set data cukup besar, Spark dapat mengalokasikan semua pelaksana untuk satu sesi Livy tunggal karena `spark.dynamicAllocation.maxExecutors` tidak diatur secara default. Ini berarti bahwa sesi Livy lainnya pada titik akhir pengembangan yang sama akan menunggu untuk meluncurkan pelaksana baru. Jika set data kecil, maka Spark akan dapat mengalokasikan pelaksana untuk beberapa sesi Livy pada saat yang sama.

> **Note**
>
> Untuk informasi lebih lanjut tentang bagaimana sumber daya dialokasikan dalam kasus penggunaan yang berbeda dan bagaimana Anda menetapkan konfigurasi untuk mengubah perilaku, lihat Konfigurasi lanjutan: berbagi titik akhir pengembangan di antara banyak pengguna.

Konfigurasi multi-tenancy

> **Note**
>
> Perlu dicatat bahwa, titik akhir pengembangan dimaksudkan untuk melakukan emulasi pada lingkungan ETL AWS Glue sebagai lingkungan penghuni tunggal. Meskipun penggunaan multi-penghuni dimungkinkan, namun konfigurasi ini merupakan kasus penggunaan lanjutan dan dianjurkan sebagian besar pengguna untuk tetap mempertahankan pola penghuni tunggal untuk setiap titik akhir pengembangan.

Dalam kasus penggunaan multi penghuni, Anda mungkin perlu mengurus alokasi sumber daya. Faktor utamanya adalah jumlah pengguna bersamaan yang menggunakan notebook Jupyter secara bersamaan. Jika tim Anda bekerja dalam alur kerja follow-the-sun "" dan hanya ada satu pengguna Jupyter di setiap zona waktu, maka jumlah pengguna bersamaan hanya satu, jadi Anda tidak perlu khawatir dengan alokasi sumber daya. Namun, jika notebook Anda dibagi di antara beberapa pengguna dan setiap pengguna mengirimkan kode secara sementara waktu, maka Anda akan perlu mempertimbangkan poin di bawah ini.

Untuk mempartisi sumber daya cluster Spark di antara beberapa pengguna, Anda dapat menggunakan SparkMagic konfigurasi. Ada dua cara berbeda untuk mengkonfigurasi SparkMagic.

(A) Gunakan direktif %%configure -f

Jika Anda ingin mengubah konfigurasi per sesi Livy dari notebook, maka Anda dapat menjalankan direktif `%%configure -f` pada paragraf notebook.

Misalnya, jika Anda ingin menjalankan aplikasi Spark pada 5 pelaksana, maka Anda dapat menjalankan perintah berikut pada paragraf notebook.

```
%%configure -f
```

```
{"numExecutors":5}
```

Maka Anda akan melihat hanya 5 pelaksana yang berjalan untuk tugas pada Spark UI.

Kami merekomendasikan untuk membatasi jumlah maksimum pelaksana untuk alokasi sumber daya dinamis.

```
%%configure -f
{"conf":{"spark.dynamicAllocation.maxExecutors":"5"}}
```

(B) Ubah SparkMagic file konfigurasi

SparkMagic bekerja berdasarkan Livy API. SparkMagic membuat sesi Livy dengan konfigurasi sepertidriverMemory,,, driverCores,executorMemory, executorCores numExecutorsconf, dll. Itulah faktor kunci yang menentukan berapa banyak sumber daya yang dikonsumsi dari seluruh cluster Spark. SparkMagic memungkinkan Anda menyediakan file konfigurasi untuk menentukan parameter yang dikirim ke Livy. Anda dapat melihat contoh file config dalam Repositori Github.

Jika Anda ingin mengubah konfigurasi di semua sesi Livy dari buku catatan, Anda dapat memodifikasi /home/ec2-user/.sparkmagic/config.json untuk menambahkansession_config.

Untuk memodifikasi file konfigurasi pada instance SageMaker notebook, Anda dapat mengikuti langkah-langkah ini.

1. Buka buku SageMaker catatan.
2. Buka kernel Terminal.
3. Jalankan perintah berikut:

   ```
   sh-4.2$ cd .sparkmagic
   sh-4.2$ ls
   config.json logs
   sh-4.2$ sudo vim config.json
   ```

   Misalnya, Anda dapat menambahkan baris ini ke /home/ec2-user/.sparkmagic/config.json dan memulai ulang kernel Jupyter dari notebook.

   ```
     "session_configs": {
   ```

```
    "conf": {
      "spark.dynamicAllocation.maxExecutors":"5"
    }
  },
```

Panduan dan praktik terbaik

Untuk menghindari konflik sumber daya semacam ini, Anda dapat menggunakan beberapa pendekatan dasar, misalnya:

- Memiliki klaster Spark yang lebih besar dengan meningkatkan `NumberOfWorkers` (penskalaan horizontal) dan meningkatkan `workerType` (penskalaan vertikal)
- Mengalokasikan sumber daya yang lebih sedikit untuk setiap pengguna (lebih sedikit sumber daya per sesi Livy)

Pendekatan Anda akan tergantung pada kasus penggunaan Anda. Jika Anda memiliki titik akhir pengembangan yang lebih besar, dan data-nya tidak dalam jumlah besar, kemungkinan konflik sumber daya akan menurun secara signifikan karena Spark dapat mengalokasikan sumber daya berdasarkan strategi alokasi yang dinamis.

Seperti dijelaskan di atas, jumlah pelaksana Spark dapat dihitung secara otomatis berdasarkan kombinasi DPU (atau `NumberOfWorkers`) dan jenis pekerja. Masing-masing aplikasi Spark meluncurkan satu driver dan beberapa pelaksana. Untuk menghitung Anda akan membutuhkan `NumberOfWorkers` = `NumberOfExecutors + 1`. Matriks di bawah ini menjelaskan berapa banyak kapasitas yang Anda butuhkan dalam titik akhir pengembangan Anda berdasarkan jumlah pengguna bersamaan.

Jumlah pengguna notebook bersamaan	Jumlah pelaksana Spark yang ingin Anda alokasikan per pengguna	Total NumberOfWorkers untuk titik akhir dev Anda
3	5	18
10	5	60
50	5	300

Jika Anda ingin mengalokasikan sumber daya yang lebih sedikit untuk setiap pengguna, `spark.dynamicAllocation.maxExecutors` (atau `numExecutors`) akan menjadi parameter termudah untuk dikonfigurasi sebagai parameter sesi Livy. Jika Anda mengatur konfigurasi di bawah ini`/home/ec2-user/.sparkmagic/config.json`, maka SparkMagic akan menetapkan maksimum 5 pelaksana per sesi Livy. Hal ini akan membantu memisahkan sumber daya per sesi Livy.

```
"session_configs": {
    "conf": {
      "spark.dynamicAllocation.maxExecutors":"5"
    }
  },
```

Misalkan ada titik akhir pengembangan dengan 18 pekerja (G.1X) dan ada 3 pengguna notebook bersamaan pada saat yang sama. Jika config sesi Anda memiliki `spark.dynamicAllocation.maxExecutors=5` maka setiap pengguna dapat menggunakan 1 driver dan 5 pelaksana. Tidak akan ada konflik sumber daya bahkan ketika Anda menjalankan beberapa paragraf buku catatan pada saat yang bersamaan.

Trade-off

Dengan config sesi `"spark.dynamicAllocation.maxExecutors":"5"` ini, Anda akan dapat menghindari kesalahan konflik sumber daya dan Anda tidak perlu menunggu alokasi sumber daya ketika ada akses pengguna secara bersamaan. Namun demikian, bahkan ketika ada banyak sumber daya gratis (misalnya, tidak ada pengguna bersamaan lainnya), Spark tidak dapat menetapkan lebih dari 5 pelaksana untuk sesi Livy Anda.

Catatan lainnya

Hal ini merupakan praktik yang baik untuk menghentikan kernel Jupyter saat Anda berhenti menggunakan sebuah notebook. Ini akan membebaskan sumber daya dan pengguna notebook lainnya dapat langsung menggunakan sumber daya tersebut tanpa perlu menunggu kernel menjadi kedaluwarsa (auto-shutdown).

Masalah umum

Bahkan ketika Anda sudah mengikuti pedoman, Anda mungkin masih akan mengalami masalah tertentu.

Sesi tidak ditemukan

Ketika Anda mencoba untuk menjalankan sebuah paragraf notebook meskipun sesi Livy Anda telah dihentikan, Anda akan melihat pesan di bawah ini. Untuk mengaktifkan sesi Livy, Anda perlu memulai ulang kernel Jupyter dengan memilih Kernel > Mulai Ulang di menu Jupyter, lalu jalankan paragraf notebook lagi.

```
An error was encountered:
Invalid status code '404' from http://localhost:8998/sessions/13 with error payload:
 "Session '13' not found."
```

Sumber daya YARN tidak cukup

Ketika Anda mencoba untuk menjalankan sebuah paragraf notebook meskipun klaster Spark Anda tidak memiliki cukup sumber daya untuk memulai sesi Livy baru, Anda akan melihat pesan di bawah ini. Anda sering kali dapat menghindari masalah ini dengan mengikuti panduan, namun, mungkin ada kemungkinan bahwa Anda akan menghadapi masalah ini. Untuk mengatasi masalah ini, Anda dapat memeriksa apakah ada sesi Livy aktif yang tidak dibutuhkan. Jika ada sesi Livy yang tidak dibutuhkan, Anda harus mengakhirinya untuk membebaskan sumber daya klaster. Untuk informasi detail, lihat bagian berikutnya.

```
Warning: The Spark session does not have enough YARN resources to start.
The code failed because of a fatal error:
    Session 16 did not start up in 60 seconds..

Some things to try:
a) Make sure Spark has enough available resources for Jupyter to create a Spark
 context.
b) Contact your Jupyter administrator to make sure the Spark magics library is
 configured correctly.
c) Restart the kernel.
```

Pemantauan dan debugging

Bagian ini menjelaskan teknik untuk melakukan pemantauan sumber daya dan sesi.

Memantau dan men-debug alokasi sumber daya cluster

Anda dapat mengawasi Spark UI untuk memantau berapa banyak sumber daya yang dialokasikan per sesi Livy, dan apa konfigurasi Spark yang berlaku pada tugas. Untuk mengaktifkan Spark UI, lihat Mengaktifkan Apache Spark Web UI untuk Titik Akhir Pengembangan.

(Opsional) Jika Anda membutuhkan tampilan waktu nyata dari Spark UI, Anda dapat mengkonfigurasi terowongan SSH terhadap server riwayat Spark yang berjalan pada klaster Spark.

```
ssh -i <private-key.pem> -N -L 8157:<development endpoint public address>:18080
 glue@<development endpoint public address>
```

Anda kemudian dapat membuka http://localhost:8157 pada peramban Anda untuk melihat Spark UI.

Sesi Livy gratis yang tidak dibutuhkan

Tinjau prosedur ini untuk menutup setiap sesi Livy yang tidak diperlukan dari notebook atau klaster Spark.

(a). Mengakhiri sesi Livy dari sebuah notebook

Anda dapat mematikan kernel pada notebook Jupyter untuk mengakhiri sesi Livy yang tidak dibutuhkan.

(b). Mengakhiri sesi Livy dari klaster Spark

Jika ada sesi Livy yang tidak diperlukan yang masih berjalan, Anda dapat menutup sesi Livy pada klaster Spark.

Sebagai prasyarat untuk melakukan prosedur ini, Anda perlu mengkonfigurasi kunci publik SSH Anda untuk titik akhir pengembangan Anda.

Untuk log in ke klaster Spark, Anda dapat menjalankan perintah berikut:

```
$ ssh -i <private-key.pem> glue@<development endpoint public address>
```

Anda dapat menjalankan perintah berikut untuk melihat sesi Livy yang aktif:

```
$ yarn application -list
20/09/25 06:22:21 INFO client.RMProxy: Connecting to ResourceManager at
 ip-255-1-106-206.ec2.internal/172.38.106.206:8032
Total number of applications (application-types: [] and states: [SUBMITTED, ACCEPTED,
 RUNNING]):2
Application-Id Application-Name Application-Type User Queue State Final-State Progress
 Tracking-URL
application_1601003432160_0005 livy-session-4 SPARK livy default RUNNING UNDEFINED 10%
 http://ip-255-1-4-130.ec2.internal:41867
```

```
application_1601003432160_0004 livy-session-3 SPARK livy default RUNNING UNDEFINED 10%
 http://ip-255-1-179-185.ec2.internal:33727
```

Anda kemudian dapat menutup sesi Livy dengan perintah berikut ini:

```
$ yarn application -kill application_1601003432160_0005
20/09/25 06:23:38 INFO client.RMProxy: Connecting to ResourceManager at
 ip-255-1-106-206.ec2.internal/255.1.106.206:8032
Killing application application_1601003432160_0005
20/09/25 06:23:39 INFO impl.YarnClientImpl: Killed application
 application_1601003432160_0005
```

Mengelola buku catatan

> **Note**
>
> Titik Akhir Pengembangan hanya didukung untuk versi AWS Glue sebelum 2.0. Untuk lingkungan interaktif tempat Anda dapat membuat dan menguji skrip ETL, gunakan Notebook di Studio. AWS Glue

Notebook memungkinkan pengembangan dan pengujian interaktif skrip ETL (ekstrak, transformasi, dan muat) Anda pada titik akhir pengembangan. AWS Gluemenyediakan antarmuka untuk SageMaker notebook Jupyter. DenganAWS Glue, Anda membuat dan mengelola SageMaker notebook. Anda juga dapat membuka SageMaker notebook dari AWS Glue konsol.

Selain itu, Anda dapat menggunakan Apache Spark dengan SageMaker titik akhir AWS Glue pengembangan yang mendukung SageMaker (tetapi bukan pekerjaan AWS Glue ETL). SageMaker Spark adalah perpustakaan Apache Spark open source untuk. SageMaker Untuk informasi selengkapnya, lihat Menggunakan Apache Spark dengan Amazon. SageMaker

> **Important**
>
> Mengelola SageMaker buku catatan dengan titik akhir AWS Glue pengembangan tersedia di Wilayah berikut: AWS

Wilayah	Code
US East (Ohio)	`us-east-2`
US East (N. Virginia)	`us-east-1`
US West (N. California)	`us-west-1`
US West (Oregon)	`us-west-2`
Asia Pacific (Tokyo)	`ap-northeast-1`
Asia Pacific (Seoul)	`ap-northeast-2`
Asia Pacific (Mumbai)	`ap-south-1`
Asia Pacific (Singapore)	`ap-southeast-1`
Asia Pacific (Sydney)	`ap-southeast-2`
Canada (Central)	`ca-central-1`
Europe (Frankfurt)	`eu-central-1`
Europe (Ireland)	`eu-west-1`
Europe (London)	`eu-west-2`

Membangun pekerjaan ETL visual dengan AWS Glue Studio

Sebuah tugas AWS Glue merangkum skrip yang terhubung ke sumber data Anda, memprosesnya, dan kemudian menuliskannya ke target data Anda. Biasanya, sebuah tugas menjalankan skrip extract, transform, and load (ETL). Jobs dapat menjalankan skrip yang dirancang untuk lingkungan runtime Apache Spark dan Ray. Tugas juga dapat menjalankan skrip Python tujuan umum (tugas shell Python.) pemicu AWS Glue dapat memulai tugas berdasarkan jadwal atau peristiwa, atau sesuai permintaan. Anda dapat memantau eksekusi tugas untuk memahami metrik waktu aktif seperti status penyelesaian, durasi, dan waktu mulai.

Anda dapat menggunakan skrip yang dihasilkan AWS Glue atau Anda dapat memberikan milik Anda sendiri. Dengan skema sumber dan lokasi target atau skema, pembuat AWS Glue Studio kode dapat secara otomatis membuat skrip Apache Spark API (). PySpark Anda dapat menggunakan skrip ini sebagai titik awal dan mengedit skrip tersebut untuk memenuhi tujuan Anda.

AWS Gluedapat menulis file output dalam beberapa format data. Setiap jenis pekerjaan dapat mendukung format output yang berbeda. Untuk beberapa format data, format-format kompresi umum dapat ditulis.

Masuk ke AWS Glue konsol

Pekerjaan AWS Glue terdiri dari logika bisnis yang melakukan pekerjaan ekstrak, transformasi, dan beban (ETL). Anda dapat membuat tugas di bagian ETL pada konsol AWS Glue.

Untuk melihat pekerjaan yang ada, masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/. Lalu pilih tab Tugas di AWS Glue. Daftar Tugas menampilkan lokasi skrip yang dikaitkan dengan setiap tugas, kapan tugas terakhir diubah, dan opsi bookmark tugas saat ini.

Saat membuat pekerjaan baru, atau setelah Anda menyimpan pekerjaan Anda, Anda dapat menggunakan kaleng AWS Glue Studio untuk memodifikasi pekerjaan ETL Anda. Anda dapat melakukan ini dengan mengedit simpul di editor visual atau dengan mengedit skrip tugas dalam mode developer. Anda juga dapat menambah dan menghapus simpul dalam editor visual untuk membuat tugas ETL yang lebih rumit.

Langkah selanjutnya untuk membuat pekerjaan di AWS Glue Studio

Anda menggunakan editor tugas visual untuk mengkonfigurasi simpul untuk tugas Anda. Setiap simpul merupakan sebuah tindakan, seperti membaca data dari lokasi sumber atau menerapkan transformasi ke data. Setiap simpul yang Anda tambahkan ke tugas Anda memiliki properti yang memberikan informasi tentang lokasi data atau transformasinya.

Langkah selanjutnya untuk membuat dan mengelola tugas Anda adalah:

- Visual ETL dengan AWS Glue Studio
- Lihat skrip tugas
- Mengubah properti tugas
- Simpan tugas
- Memulai eksekusi tugas
- Melihat informasi untuk eksekusi tugas terbaru
- Mengakses dasbor pemantauan tugas

Visual ETL dengan AWS Glue Studio

Anda dapat menggunakan antarmuka visual sederhana AWS Glue Studio untuk membuat pekerjaan ETL Anda. Anda menggunakan halaman Tugas untuk membuat tugas baru. Anda juga dapat menggunakan editor skrip atau buku catatan untuk bekerja secara langsung dengan kode dalam skrip pekerjaan AWS Glue Studio ETL.

Pada halaman Jobs, Anda dapat melihat semua pekerjaan yang telah Anda buat baik dengan AWS Glue Studio atauAWS Glue. Anda dapat melihat, mengelola, dan menjalankan tugas Anda di halaman ini.

Lihat juga tutorial blog pada contoh lain tentang cara membuat pekerjaan ETL denganAWS Glue Studio.

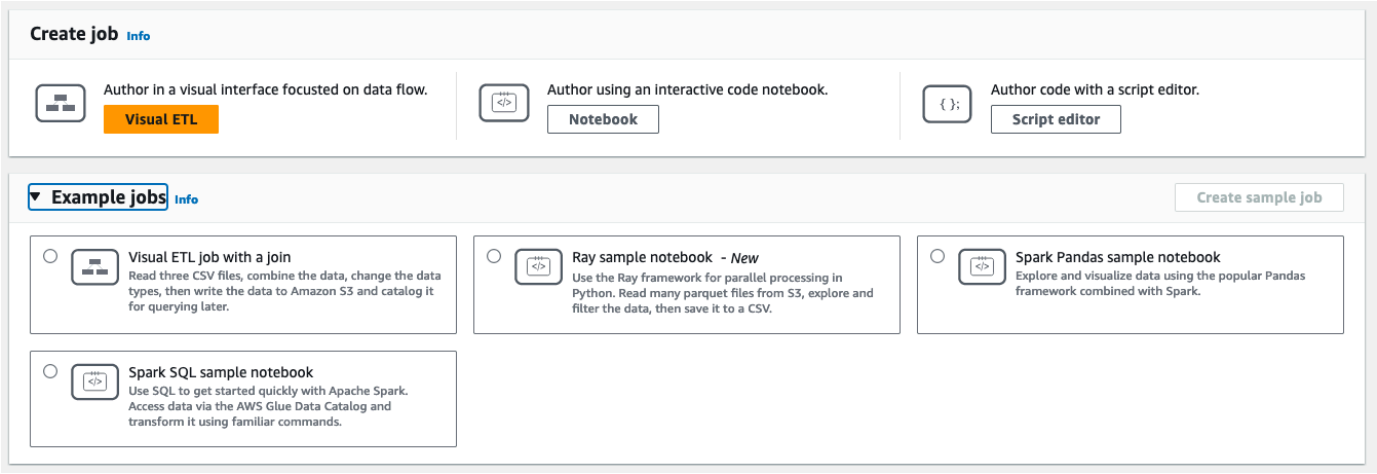
Memulai pekerjaan di AWS Glue Studio

AWS Gluememungkinkan Anda membuat pekerjaan melalui antarmuka visual, buku catatan kode interaktif, atau dengan editor skrip. Anda dapat memulai pekerjaan dengan mengklik salah satu opsi atau membuat pekerjaan baru berdasarkan contoh pekerjaan.

Contoh pekerjaan membuat pekerjaan dengan alat pilihan Anda. Misalnya, pekerjaan sampel memungkinkan Anda membuat pekerjaan ETL visual yang menggabungkan file CSV ke dalam tabel catatlog, membuat pekerjaan di buku catatan kode interaktif dengan AWS Glue untuk Ray atau AWS Glue untuk Spark saat bekerja dengan panda, atau membuat pekerjaan di buku catatan kode interaktif dengan SparkSQL.

Membuat pekerjaan AWS Glue Studio dari awal

1. Masuk ke AWS Management Console dan buka AWS Glue Studio konsol di https://console.aws.amazon.com/gluestudio/.
2. Pilih pekerjaan ETL dari panel navigasi.
3. Di bagian Buat pekerjaan, pilih opsi konfigurasi untuk pekerjaan Anda.



Opsi untuk membuat pekerjaan dari awal:

- Visual ETL — penulis dalam antarmuka visual yang berfokus pada aliran data
- Penulis menggunakan buku catatan kode Interaktif — pekerjaan penulis interaktif di antarmuka notebook berdasarkan Jupyter Notebooks

  Ketika Anda memilih opsi ini, Anda harus memberikan informasi tambahan sebelum membuat sesi penulisan buku catatan. Untuk informasi selengkapnya tentang cara menentukan informasi ini, lihatMemulai dengan notebook di AWS Glue Studio.
- Kode penulis dengan editor skrip — Bagi mereka yang terbiasa dengan pemrograman dan penulisan skrip ETL, pilih opsi ini untuk membuat pekerjaan Spark ETL baru. Pilih mesin (Python shell, Ray, Spark (Python), atau Spark (Scala). Kemudian, pilih Mulai baru atau Unggah skrip. mengunggah skrip yang ada dari file lokal. Jika Anda memilih untuk

menggunakan editor skrip, Anda tidak dapat menggunakan editor pekerjaan visual untuk merancang atau mengedit pekerjaan Anda.

Sebuah tugas Spark dijalankan di lingkungan Apache Spark yang dikelola oleh AWS Glue. Secara default, skrip baru dikodekan dengan Python. Untuk menulis skrip Scala baru, lihatMembuat dan mengedit skrip Scala di AWS Glue Studio.

Membuat pekerjaan AWS Glue Studio dari contoh pekerjaan

Anda dapat memilih untuk membuat pekerjaan dari contoh pekerjaan. Di bagian Contoh pekerjaan, pilih pekerjaan sampel, lalu pilih Buat pekerjaan sampel. Membuat contoh pekerjaan dari salah satu opsi menyediakan templat cepat yang dapat Anda kerjakan.

1. Masuk ke AWS Management Console dan buka AWS Glue Studio konsol di https://console.aws.amazon.com/gluestudio/.
2. Pilih pekerjaan ETL dari panel navigasi.
3. Pilih opsi buat pekerjaan dari pekerjaan sampel:
   - Tugas ETL visual untuk bergabung dengan beberapa sumber — Baca tiga file CSV, gabungkan data, ubah tipe data, lalu tulis data ke Amazon S3 dan katalogkan untuk kueri nanti.
   - Spark notebook menggunakan Pandas — Jelajahi dan visualisasikan data menggunakan kerangka Pandas populer yang dikombinasikan dengan Spark.
   - Spark notebook menggunakan SQL - Gunakan SQL untuk memulai dengan cepat dengan Apache Spark. Akses data melalui Katalog AWS Glue Data dan ubah menggunakan perintah yang sudah dikenal.
4. Pilih Buat pekerjaan sampel.

Fitur editor tugas

Editor tugas menyediakan fitur-fitur berikut untuk membuat dan mengedit tugas.

- Sebuah diagram visual tugas Anda, dengan simpul untuk setiap tugas: simpul sumber data untuk membaca data; simpul transformasi untuk memodifikasi data; simpul target data untuk menulis data.

Anda dapat melihat dan mengkonfigurasi properti dari setiap simpul dalam diagram tugas tersebut. Anda juga dapat melihat skema dan sampel data untuk setiap simpul dalam diagram tugas tersebut. Fitur ini membantu Anda memverifikasi bahwa tugas Anda memodifikasi dan mengubah data dengan cara yang benar, tanpa harus menjalankan tugas.

- Sebuah Skrip melihat dan mengedit tab, di mana Anda dapat memodifikasi kode yang dihasilkan untuk tugas Anda.
- Tab Detail tugas, di mana Anda dapat mengkonfigurasi berbagai pengaturan untuk menyesuaikan lingkungan di mana tugas ETL AWS Glue Anda berjalan.
- Tab Eksekusi, di mana Anda dapat melihat tugas saat ini dan sebelumnya, melihat status eksekusi tugas, dan mengakses log untuk eksekusi tugas tersebut.
- Tab kualitas data, tempat Anda dapat menerapkan aturan kualitas data ke pekerjaan Anda.
- Tab Jadwal, di mana Anda dapat mengkonfigurasi waktu mulai untuk tugas Anda, atau mengatur eksekusi tugas berulang.
- Tab Kontrol Versi, tempat Anda dapat mengonfigurasi layanan Git untuk digunakan dengan pekerjaan Anda.

Menggunakan pratinjau skema dalam editor tugas visual

Saat membuat atau mengedit tugas Anda, Anda dapat menggunakan tab Skema output untuk melihat skema untuk data Anda.

Sebelum Anda dapat melihat skema, editor tugas memerlukan izin untuk mengakses sumber data. Anda dapat menentukan IAM role pada tab Detail tugas pada editor atau pada tab Skema output untuk sebuah simpul. Jika IAM role memiliki semua izin yang diperlukan untuk mengakses sumber data, Anda kemudian dapat melihat skema pada tab Skema output untuk sebuah simpul.

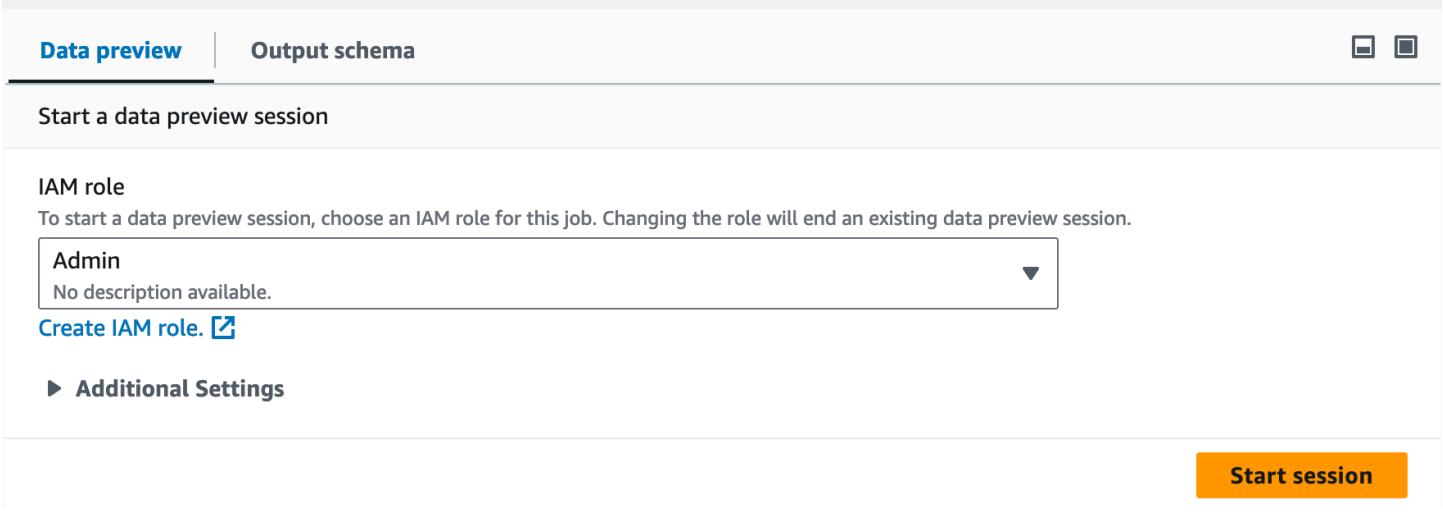
Menggunakan pratinjau data dalam editor tugas visual

Pratinjau data membantu Anda membuat dan menguji pekerjaan menggunakan sampel data Anda tanpa harus berulang kali menjalankan pekerjaan. Dengan menggunakan pratinjau data, Anda dapat:

- Uji peran IAM untuk memastikan Anda memiliki akses ke sumber data atau target data Anda.
- Periksa apakah transformasi memodifikasi data dengan cara yang dimaksud. Misalnya, jika Anda menggunakan Transformasi filter, Anda dapat memastikan bahwa filter memilih subset data yang tepat.

- Periksa data Anda. Jika set data Anda berisi kolom dengan beberapa jenis nilai, maka pratinjau data akan menampilkan daftar tupel untuk kolom-kolom ini. Setiap tupel berisi tipe data dan nilainya.

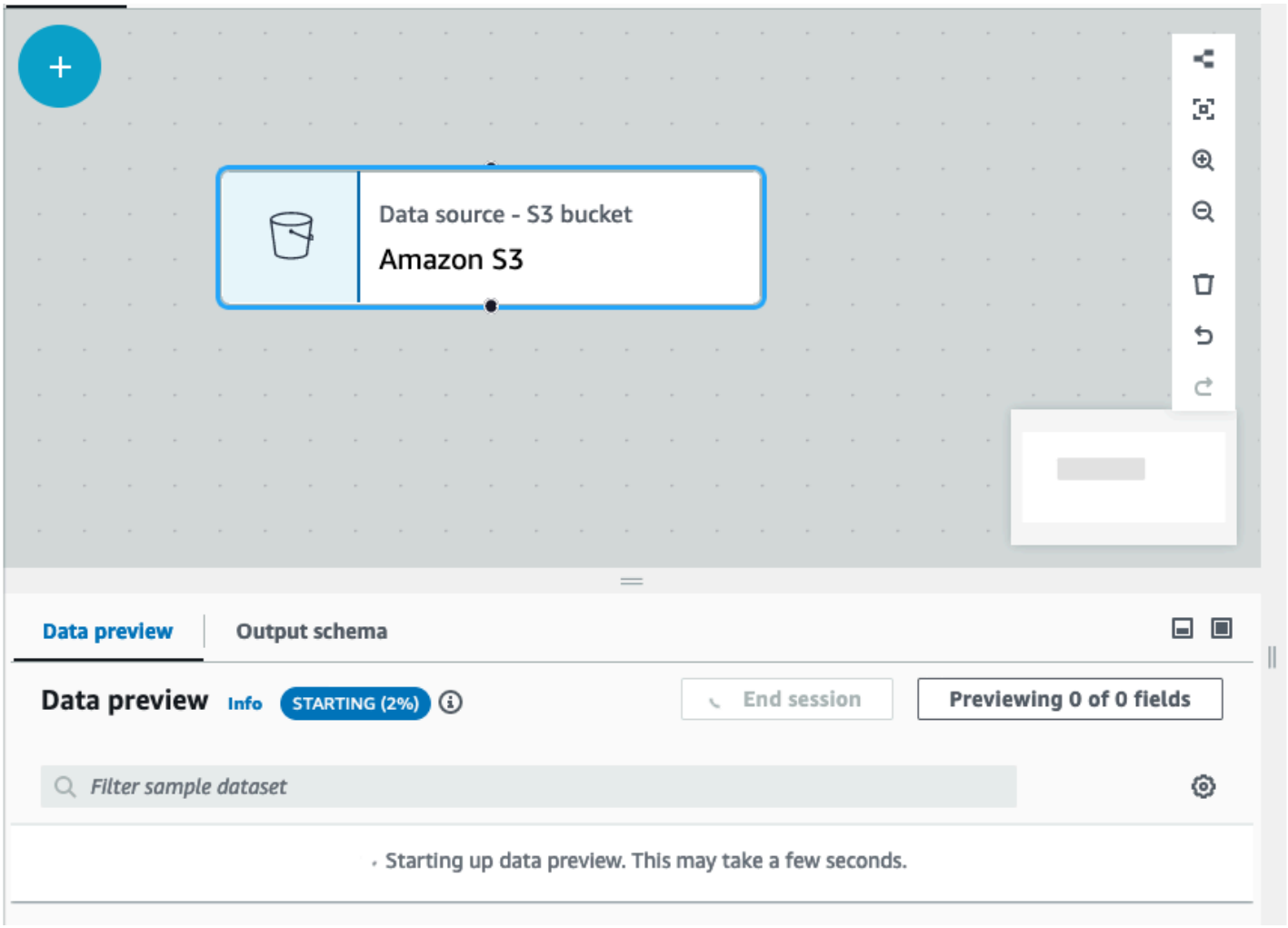
Saat membuat atau mengedit pekerjaan Anda, Anda dapat menggunakan tab Pratinjau data di bawah kanvas pekerjaan untuk melihat sampel data Anda. Sesi pratinjau data baru akan dimulai secara otomatis ketika peran sudah dikonfigurasi pada pekerjaan atau peran IAM default telah disiapkan di akun. Jika peran belum dikonfigurasi sebelumnya, Anda dapat memulai sesi dengan memilih peran.



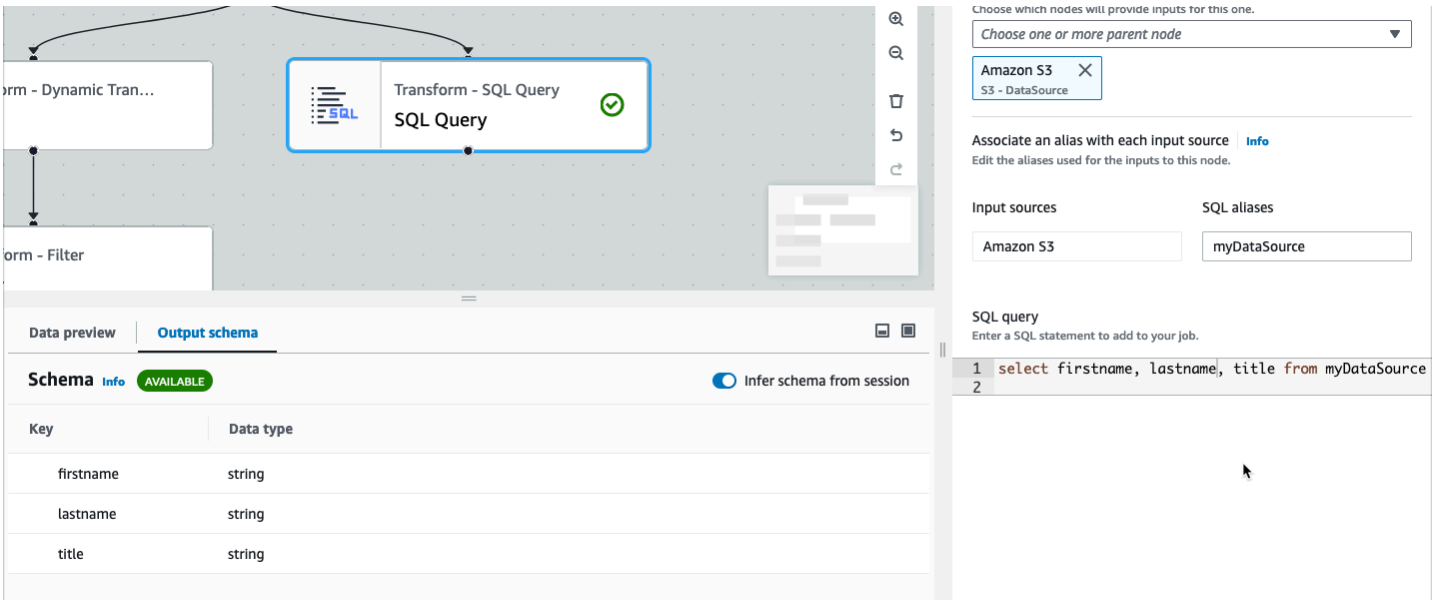
> **Note**
>
> Peran yang Anda pilih untuk sesi pratinjau data juga akan digunakan untuk pekerjaan itu.

Anda dapat melihat status dan kemajuan sesi Anda serta detail sesi dengan mengklik ikon info.

Ketika sesi siap, AWS Glue Studio akan memuat data untuk node yang Anda pilih. Anda dapat melihat% selesai saat berlangsung.



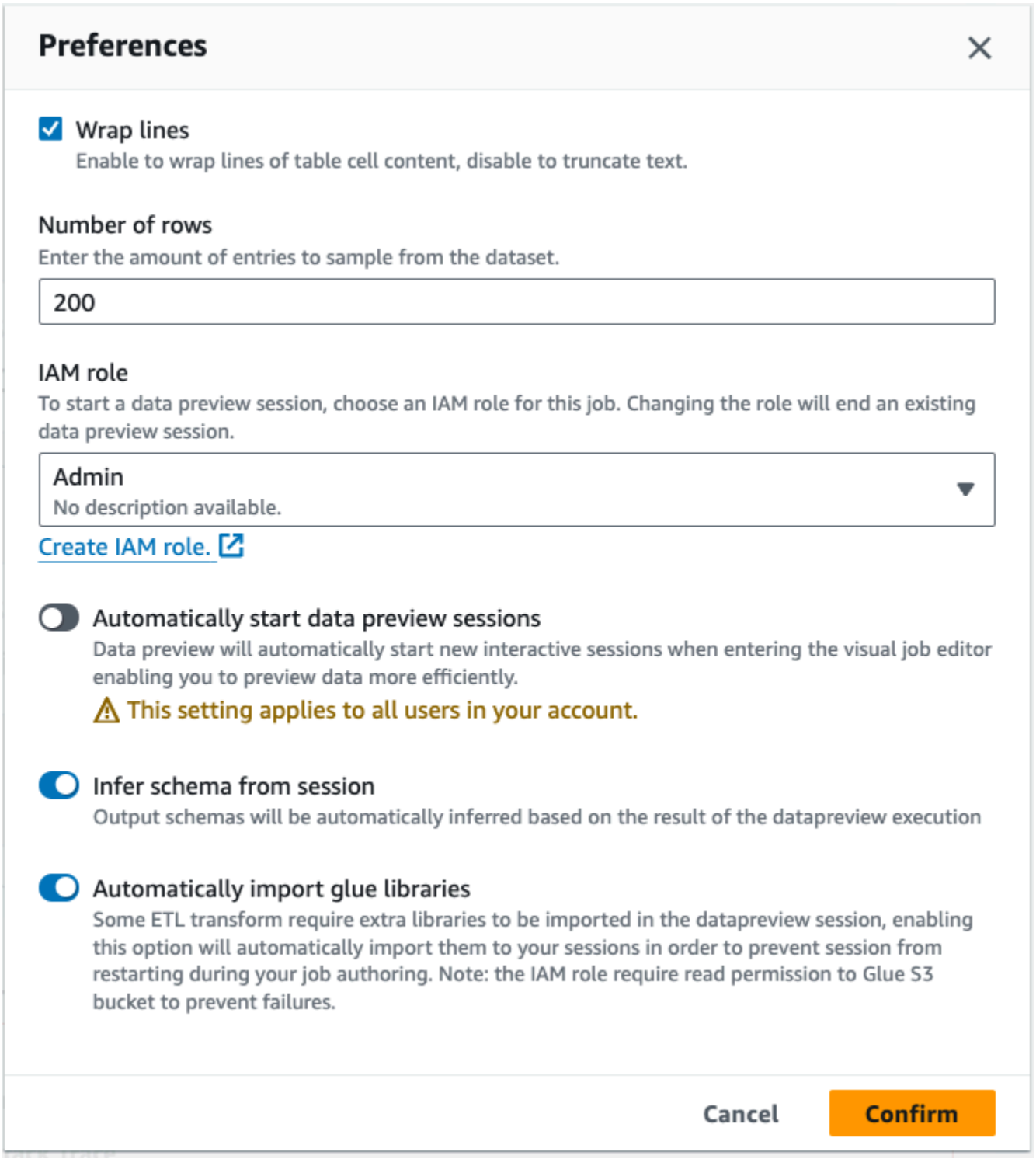
Saat Anda membuat pekerjaan visual Anda, secara otomatis AWS Glue Studio akan memperbarui skema untuk simpul yang dipilih saat Anda beralih skema Inferensi dari sesi di tab skema Output.



Untuk mengonfigurasi preferensi pratinjau data Anda:

Pilih ikon pengaturan (simbol roda gigi) untuk mengkonfigurasi preferensi Anda untuk pratinjau data. Pengaturan ini berlaku untuk semua simpul dalam diagram tugas. Anda dapat:

- Pilih untuk membungkus teks dari satu baris ke baris berikutnya. Opsi ini diaktifkan secara default
- Ubah jumlah baris (default ke 200)
- Pilih peran IAM atau buat peran IAM jika diperlukan
- Pilih untuk secara otomatis memulai sesi baru ketika Anda menulis pekerjaan. Ini memberikan sesi interaktif baru saat menulis pekerjaan. Pengaturan ini berlaku di tingkat akun. Setelah diatur, itu akan berlaku untuk semua pengguna di akun Anda saat mengedit pekerjaan apa pun.
- Pilih untuk secara otomatis menyimpulkan skema. Skema output akan secara otomatis disimpulkan untuk node yang dipilih
- Pilih untuk mengimpor AWS Glue pustaka secara otomatis. Ini berguna karena akan mencegah pratinjau data memulai ulang sesi baru saat menambahkan transformasi baru yang memerlukan restart sesi

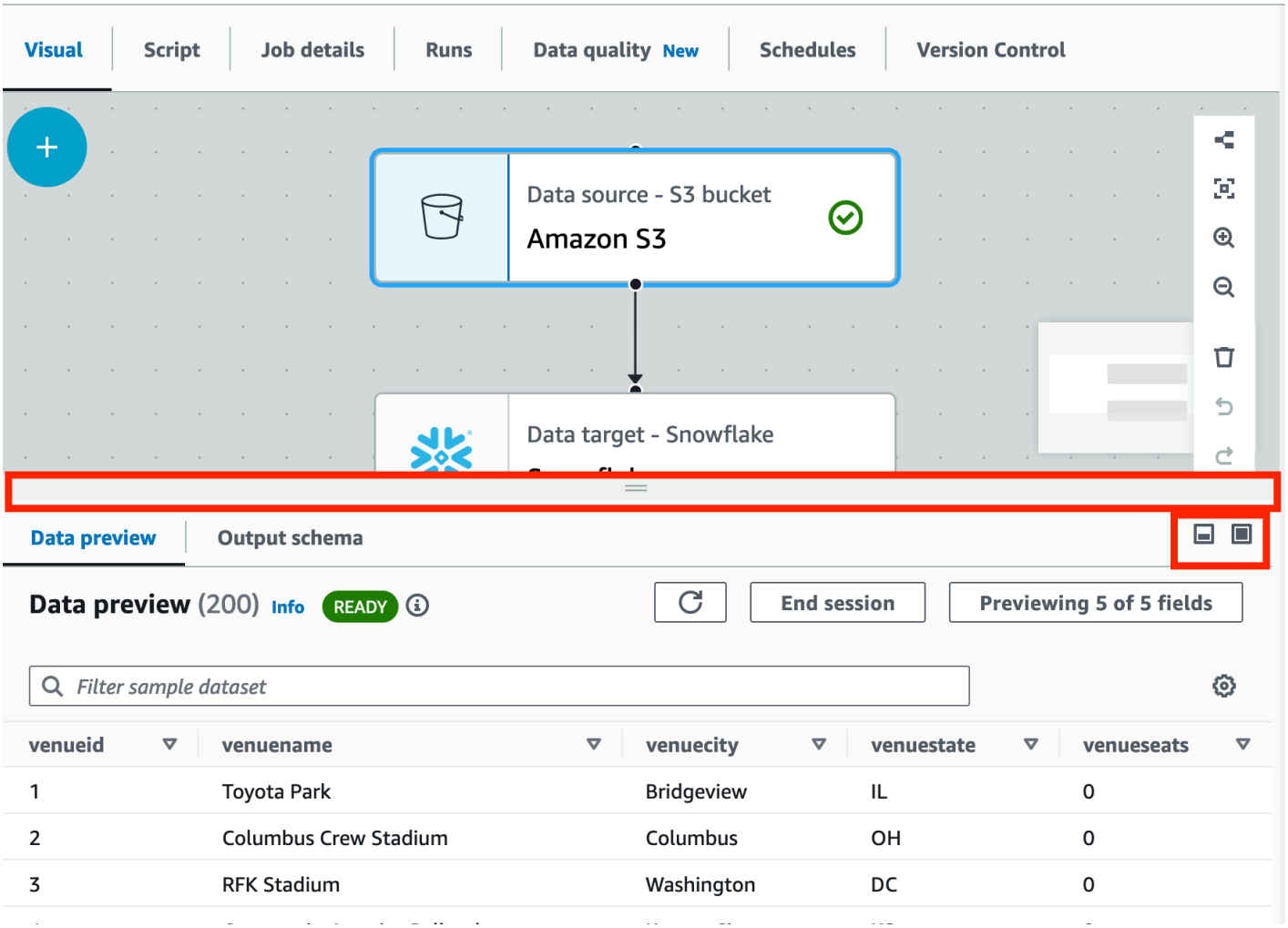


Fitur tambahan termasuk kemampuan untuk:

- Pilih Pratinjau x dari bidang y untuk memilih kolom (bidang) yang akan dipratinjau. Saat Anda mempratinjau data Anda menggunakan pengaturan default, editor tugas menampilkan 5 kolom

pertama dari set data Anda. Anda dapat mengubah ini untuk menampilkan semua atau tidak ada (tidak disarankan).

- Gulir melalui jendela pratinjau data baik secara horizontal maupun vertikal.
- Gunakan tombol maksimalkan untuk memperluas tab Pratinjau data untuk meletakkan grafik pekerjaan secara berlebihan untuk melihat struktur data dan data dengan lebih baik. Demikian pula, gunakan tombol minimalkan untuk meminimalkan tab pratinjau Data. Anda juga dapat mengambil panel pegangan dan menyeret ke atas untuk memperluas tab Pratinjau data.



- Gunakan Sesi akhir untuk menghentikan pratinjau data. Ketika Anda menghentikan sesi, Anda dapat memilih peran IAM baru, dan mengatur pengaturan tambahan (seperti mengaktifkan atau menonaktifkan pengaturan untuk secara otomatis memulai sesi baru, menyimpulkan skema, atau mengimpor AWS Glue pustaka, dan memulai sesi lagi.

Keterbatasan saat menggunakan pratinjau data

Saat menggunakan pratinjau data, Anda mungkin mengalami keterbatasan atau batasan berikut.

- Pertama kali Anda memilih tab Pratinjau data, Anda harus memilih IAM role. Peran ini harus memiliki izin yang diperlukan untuk mengakses data dan sumber daya lain yang diperlukan untuk membuat pratinjau data tersebut.
- Setelah Anda memberikan IAM role, dibutuhkan beberapa saat sebelum data tersedia dan bisa dilihat. Untuk set data dengan data kurang dari 1 GB, dibutuhkan waktu hingga satu menit. Jika Anda memiliki set data yang besar, Anda harus menggunakan partisi untuk meningkatkan waktu unggah. Memigrasikan data secara langsung dari Amazon S3 memiliki performa terbaik.
- Jika Anda memiliki kumpulan data yang sangat besar, dan dibutuhkan lebih dari 15 menit untuk menanyakan data untuk pratinjau data, permintaan akan habis. Pratinjau data memiliki batas waktu IDLE 30 menit. Untuk meringankan hal ini, kurangi ukuran dataset untuk menggunakan pratinjau data.
- Secara default, Anda melihat 50 kolom pertama di tab Pratinjau data. Jika kolom tidak memiliki nilai data, Anda akan mendapatkan sebuah pesan bahwa tidak ada data untuk ditampilkan. Anda dapat meningkatkan jumlah baris sampel, atau kolom berbeda yang telah dipilih untuk melihat nilai data.
- Pratinjau data saat ini tidak didukung untuk sumber data streaming, atau sumber data yang menggunakan konektor kustom.
- Kesalahan pada satu simpul berpengaruh pada seluruh tugas. Jika salah satu simpul memiliki kesalahan dengan data pratinjau, maka kesalahan akan muncul pada semua simpul sampai Anda memperbaikinya.
- Jika Anda mengubah sumber data untuk tugas tersebut, maka simpul anak dari sumber data tersebut mungkin perlu diperbarui agar sesuai dengan skema baru. Misalnya, jika Anda memiliki ApplyMapping node yang memodifikasi kolom, dan kolom tidak ada di sumber data pengganti, Anda perlu memperbarui node ApplyMapping transformasi.
- Jika Anda melihat tab Pratinjau data untuk simpul transformasi kueri SQL, dan kueri SQL menggunakan nama bidang yang salah, maka tab Pratinjau data akan menunjukkan kesalahan.

Pembuatan kode skrip

Saat Anda menggunakan editor visual untuk membuat pekerjaan, kode ETL dibuat secara otomatis untuk Anda. AWS Glue Studiomembuat skrip pekerjaan fungsional dan lengkap, dan menyimpannya di lokasi Amazon S3.

Ada dua bentuk kode yang dihasilkan olehAWS Glue Studio: versi asli, atau Klasik, dan versi yang lebih baru dan efisien. Secara default, generator kode baru digunakan untuk membuat skrip pekerjaan. Anda dapat membuat skrip pekerjaan menggunakan pembuat kode klasik pada tab Script dengan memilih tombol sakelar Hasilkan skrip klasik.

Beberapa perbedaan dalam versi baru dari kode yang dihasilkan meliputi:

- Blok komentar besar tidak lagi ditambahkan ke skrip
- Struktur output dalam kode menggunakan nama node yang Anda tentukan di editor visual. Dalam skrip kelas, struktur output hanya diberi nama`DataSource0`,`DataSource1`,`Transform0`,`Transform1`,`DataSink0`,`DataSink1`, dan sebagainya.
- Perintah panjang dibagi menjadi beberapa baris untuk menghilangkan kebutuhan untuk menggulir di halaman untuk melihat seluruh perintah.

Fitur baru AWS Glue Studio memerlukan versi baru pembuatan kode, dan tidak akan berfungsi dengan skrip kode klasik. Anda diminta untuk memperbarui pekerjaan ini ketika Anda mencoba menjalankannya.

Mengedit node transformasi data AWS Glue terkelola

AWS Glue Studiomenyediakan dua jenis transformasi:

- AWS Glue-native transforms - tersedia untuk semua pengguna dan dikelola oleh. AWS Glue
- Transformasi visual khusus - memungkinkan Anda mengunggah transformasi Anda sendiri untuk digunakan AWS Glue Studio

AWS Gluenode transformasi data terkelola

AWS Glue Studio menyediakan serangkaian transformasi bawaan yang dapat Anda gunakan untuk memproses data Anda. Data Anda melewati dari satu simpul dalam diagram tugas ke simpul yang lain dalam struktur data yang disebut `DynamicFrame`, yang merupakan ekstensi untuk `DataFrame` Apache Spark SQL.

Dalam diagram pra-populasi untuk suatu pekerjaan, antara sumber data dan node target data adalah simpul transformasi Change Schema. Anda dapat mengkonfigurasi simpul transformasi ini untuk memodifikasi data Anda, atau Anda dapat menggunakan transformasi tambahan.

Transformasi bawaan berikut tersedia denganAWS Glue Studio:

- ChangeSchema: Petakan kunci properti data di sumber data ke kunci properti data di target data. Anda dapat mengganti nama kunci, memodifikasi tipe data untuk kunci, dan memilih kunci mana yang akan dibuang dari set data.
- SelectFields: Pilih kunci properti data yang ingin Anda simpan.
- DropFields: Pilih kunci properti data yang ingin Anda jatuhkan.
- RenameField: Ganti nama kunci properti data tunggal.
- Spigot: Menulis sampel data ke sebuah bucket Amazon S3.
- Join: Menggabungkan dua set data menjadi satu set data menggunakan frasa perbandingan pada kunci properti data tertentu. Anda dapat menggunakan join bagian dalam, luar, kiri, kanan, kiri semi, dan lawan kiri.
- Union: Gabungkan baris dari lebih dari satu sumber data yang memiliki skema yang sama.
- SplitFields: Pisahkan kunci properti data menjadi dua`DynamicFrames`. Output adalah sebuah kumpulan `DynamicFrames`: satu dengan kunci properti data yang dipilih, dan satu dengan kunci properti data yang tersisa.
- SelectFromCollection: Pilih salah satu `DynamicFrame` dari koleksi`DynamicFrames`. Outputnya adalah `DynamicFrame` yang dipilih.
- FillMissingValues: Temukan catatan dalam kumpulan data yang memiliki nilai yang hilang dan tambahkan bidang baru dengan nilai yang disarankan yang ditentukan oleh imputasi
- Filter: Membagi set data menjadi dua, berdasarkan syarat filter.
- Jatuhkan Bidang Null: Menghapus kolom dari kumpulan data jika semua nilai di kolom 'null'.
- Jatuhkan Duplikat: Menghapus baris dari sumber data Anda dengan memilih untuk mencocokkan seluruh baris atau menentukan kunci.
- SQL: Memasukkan kode SparkSQL ke bidang entri teks untuk menggunakan kueri SQL untuk mengubah data. Outputnya adalah satu `DynamicFrame`.
- Agregat: Melakukan perhitungan (seperti rata-rata, jumlah, min, maks) pada bidang dan baris yang dipilih, dan membuat bidang baru dengan nilai yang baru dihitung.
- Flatten: Ekstrak bidang di dalam struct ke bidang tingkat atas.
- UUID: Tambahkan kolom dengan Universe Unique Identifier untuk setiap baris.
- Identifier: Tambahkan kolom dengan pengidentifikasi numerik untuk setiap baris.
- Ke stempel waktu: Ubah kolom menjadi tipe stempel waktu.

- Format timestamp: Konversi kolom stempel waktu ke string yang diformat.
- Transformasi Router Bersyarat: Terapkan beberapa kondisi ke data yang masuk. Setiap baris data yang masuk dievaluasi oleh kondisi filter grup dan diproses menjadi grup yang sesuai.
- Transformasi Kolom Gabungan: Bangun kolom string baru menggunakan nilai kolom lain dengan spacer opsional.
- Transformasi String Split: Memecah string menjadi array token menggunakan ekspresi reguler untuk menentukan bagaimana pemisahan dilakukan.
- Array To Columns transform: Ekstrak beberapa atau semua elemen kolom tipe array ke kolom baru.
- Tambahkan transformasi Timestamp Saat Ini: Tandai baris dengan waktu pemrosesan data. Ini berguna untuk tujuan audit atau untuk melacak latensi dalam pipa data.
- Pivot Rows to Columns transform: Agregat kolom numerik dengan memutar nilai unik pada kolom terpilih yang menjadi kolom baru. Jika beberapa kolom dipilih, nilainya digabungkan untuk memberi nama kolom baru.
- Unpivot Columns To Rows transform: Ubah kolom menjadi nilai kolom baru yang menghasilkan baris untuk setiap nilai unik.
- Transformasi Pemrosesan Autobalance: Mendistribusikan kembali data dengan lebih baik di antara para pekerja. Ini berguna jika data tidak seimbang atau karena berasal dari sumber tidak memungkinkan pemrosesan paralel yang cukup di atasnya.
- Transformasi Kolom Derived: Tentukan kolom baru berdasarkan rumus matematika atau ekspresi SQL di mana Anda dapat menggunakan kolom lain dalam data, serta konstanta dan literal.
- Transformasi pencarian: Tambahkan kolom dari tabel katalog yang ditentukan saat kunci cocok dengan kolom pencarian yang ditentukan dalam data.
- Explode Array atau Map Into Rows transform: Ekstrak nilai dari struktur bersarang menjadi baris individual yang lebih mudah dimanipulasi.
- Transformasi pencocokan rekaman: Memanggil transformasi klasifikasi data pembelajaran mesin Record Matching yang ada.
- Hapus baris null transform: Hapus dari baris dataset yang memiliki semua kolom sebagai null, atau kosong.
- Mengurai transformasi kolom JSON: Parse kolom string yang berisi data JSON dan mengubahnya menjadi struct atau kolom array, tergantung apakah JSON adalah objek atau array, masing-masing.
- Ekstrak transformasi jalur JSON: Ekstrak kolom baru dari kolom string JSON.

- Ekstrak fragmen string dari ekspresi reguler: Ekstrak fragmen string menggunakan ekspresi reguler dan buat kolom baru darinya, atau beberapa kolom jika menggunakan grup regex.
- Transformasi kustom: Masukkan kode ke bidang entri teks untuk menggunakan transformasi kustom. Outputnya adalah kumpulan `DynamicFrames`.

Menggunakan resep persiapan data di AWS Glue Studio

AWS Glue Studiomemungkinkan Anda untuk menggunakan AWS Glue DataBrew resep dalam alur kerja visual. Hal ini memungkinkan AWS Glue DataBrew resep pelanggan untuk dijalankan dalam AWS Glue pekerjaan bersama dengan AWS Glue Studio node lainnya.

Dalam DataBrew, resep adalah serangkaian langkah transformasi data. DataBrew resep menentukan bagaimana mengubah data yang telah dibaca dan tidak menjelaskan di mana dan bagaimana membaca data, serta bagaimana dan di mana untuk menulis data. Ini dikonfigurasi di node Sumber dan Target diAWS Glue Studio. Untuk informasi selengkapnya tentang resep, lihat Membuat dan menggunakan AWS Glue DataBrew resep.

Node Resep Persiapan Data tersedia dari panel Sumber Daya. Anda dapat menghubungkan simpul Resep Persiapan Data ke node lain dalam alur kerja visual, apakah itu node sumber Data atau node transformasi lainnya. Setelah memilih AWS Glue DataBrew resep dan versi, langkah-langkah yang diterapkan dalam resep terlihat di tab properti simpul.

Prasyarat

- Anda memiliki AWS Glue DataBrew resep yang dibuat diAWS Glue DataBrew.
- Anda memiliki izin IAM yang diperlukan seperti yang dijelaskan di bagian di bawah ini.

Izin IAM untuk AWS Glue DataBrew

Topik ini memberikan informasi untuk membantu Anda memahami tindakan dan sumber daya yang dapat digunakan oleh administrator IAM dalam kebijakan AWS Identity and Access Management (IAM) untuk transformasi Resep Persiapan Data.

Untuk informasi tambahan tentang keamanan diAWS Glue, lihat Manajemen Akses.

Tabel berikut mencantumkan izin yang dibutuhkan pengguna untuk melakukan operasi tertentu untuk menggunakan transformasi Resep Persiapan Data.

Resep Persiapan Data mengubah tindakan

Action	Deskripsi
`databrew:ListRecipes`	Memberikan izin untuk mengambil resepAWS Glue DataBrew.
`databrew:ListRecipeVersions`	Memberikan izin untuk mengambil versi AWS Glue DataBrew resep.
`databrew:DescribeRecipe`	Memberikan izin untuk mengambil deskripsi AWS Glue DataBrew resep.

Peran yang Anda gunakan untuk mengakses fungsionalitas ini harus memiliki kebijakan yang memungkinkan beberapa AWS Glue DataBrew Anda dapat mencapainya dengan menggunakan `AWSGlueConsoleFullAccess` kebijakan yang menyertakan tindakan yang diperlukan atau menambahkan kebijakan sebaris berikut ke peran Anda:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "databrew:ListRecipes",
                "databrew:ListRecipeVersions",
                "databrew:DescribeRecipe"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Untuk menggunakan transformasi Resep Persiapan Data, Anda harus menambahkan `IAM:PassRole` tindakan ke kebijakan izin.

Izin tambahan yang diperlukan

Action	Deskripsi
`iam:PassRole`	Memberikan izin kepada IAM untuk memungkinkan pengguna melewati peran yang disetujui.

Tanpa izin ini terjadi kesalahan berikut:

```
"errorCode": "AccessDenied"
"errorMessage": "User: arn:aws:sts::account_id:assumed-role/AWSGlueServiceRole is not
authorized to perform: iam:PassRole on resource: arn:aws:iam::account_id:role/service-
role/AWSGlueServiceRole
because no identity-based policy allows the iam:PassRole action"
```

Keterbatasan:

- Tidak semua AWS Glue DataBrew resep didukung olehAWS Glue. Beberapa resep tidak akan dapat dijalankanAWS Glue Studio.
  - Resep dengan transformasi UNION dan JOIN tidak didukung, namun, AWS Glue Studio sudah memiliki node transformasi “Join” dan “Union” yang dapat digunakan sebelum atau sesudah simpul Resep Persiapan Data sebagai gantinya.
- Node Resep Persiapan Data didukung untuk pekerjaan yang dimulai dengan AWS Glue versi 4.0. Versi ini akan dipilih secara otomatis setelah simpul Resep Persiapan Data ditambahkan ke pekerjaan.
- Simpul Resep Persiapan Data membutuhkan Python. Ini secara otomatis diatur ketika simpul Resep Persiapan Data ditambahkan ke pekerjaan.
- Saat menggunakan Data Preview, Anda harus memulai ulang sesi pratinjau data Anda setelah menambahkan simpul Resep Persiapan Data ke pekerjaan Anda.

Cara menggunakan AWS Glue DataBrew resep di AWS Glue Studio

Untuk menggunakan AWS Glue DataBrew resepAWS Glue Studio, mulailah dengan membuat resep diAWS Glue DataBrew. Jika Anda memiliki resep yang ingin Anda gunakan, Anda dapat melewati langkah ini.

Untuk membuat AWS Glue DataBrew resep diAWS Glue DataBrew:

1. Penulis resep diAWS Glue DataBrew. Untuk informasi selengkapnya, lihat Memulai dengan AWS Glue DataBrew.
2. Simpan resep Anda.
3. Publikasikan resep Anda. Ini akan mempublikasikan resep Anda sebagai versi 1.0.

Untuk menggunakan simpul Resep Persiapan Data diAWS Glue Studio:

Anda dapat menggunakan lebih dari satu simpul Resep Persiapan Data dalam pekerjaan ETL visual. Untuk melakukan ini, tambahkan simpul Resep Persiapan Data dengan mengikuti langkah-langkah di bawah ini dan tambahkan simpul Resep Persiapan Data lain ke pekerjaan. Misalnya, alur kerja mungkin mengikuti pola ini:

- Sumber data 1 > resep 1 > keluaran 1
- Sumber data 2 > resep 2 > output 2
- keluaran 1, keluaran 2 > GABUNG

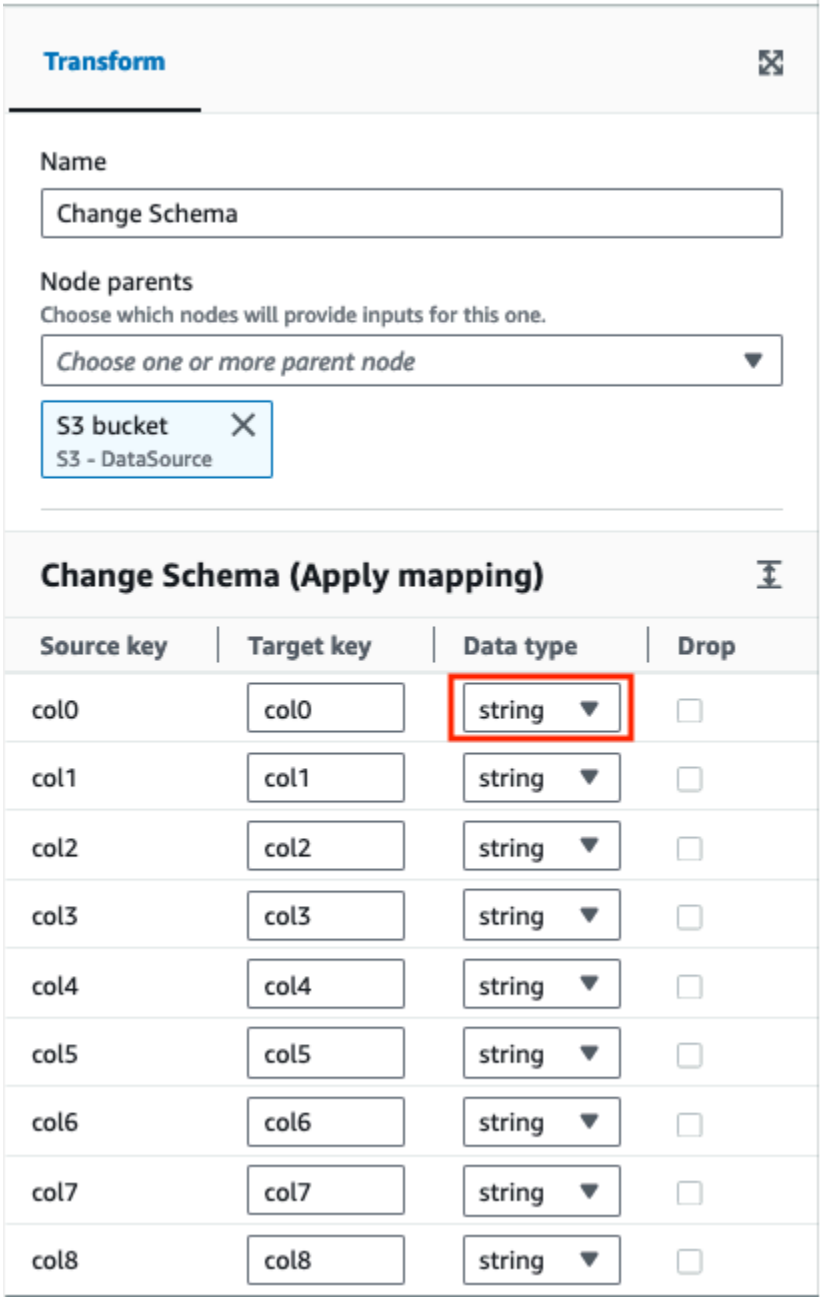
1. Mulai AWS Glue pekerjaan AWS Glue Studio dengan sumber data.
2. Tambahkan simpul Resep Persiapan Data ke sumber data Anda.
3. Filter resep berdasarkan nama dengan mengetikkan nama resep di bidang pencarian.
4. Pilih versi yang diterbitkan. Hanya versi yang diterbitkan yang tersedia.
5. Selesaikan penulisan pekerjaan dengan menambahkan node transformasi lain sesuai kebutuhan dan tambahkan node target Data untuk menyimpan output pekerjaan.
6. Buat perubahan konfigurasi yang diperlukan di tab Detail pekerjaan, seperti menamai pekerjaan Anda dan menyesuaikan kapasitas yang dialokasikan sesuai kebutuhan, dan simpan pekerjaan.
7. Jalankan pekerjaan dengan memilih Jalankan dari menu tarik-turun Tindakan.

Untuk mengubah skema jika sumber datanya Amazon S3 dan format datanya CSV:

Jika semua kolom dalam file CSV awalnya dimuat sebagai tipe data stringAWS Glue Studio, Anda perlu memastikan bahwa tipe data kolom kompatibel dengan langkah-langkah lainnya dalam AWS Glue DataBrew resep.

AWS Glue DataBrewresep hanya menentukan cara mengubah data yang telah dibaca. Ini tidak menjelaskan di mana dan bagaimana membaca data.

1. Tambahkan simpul Ubah Skema sebelum simpul resep Multi-langkah.
2. Klik Ubah Skema simpul dan ubah skema menjadi sama dengan tipe data kolom AWS Glue DataBrew dengan memilih tipe data baru di Transform untuk kolom sesuai kebutuhan.



Untuk mengubah skema jika sumber data tanpa kepala:

AWS Glue DataBrewresep hanya menentukan cara mengubah data yang telah dibaca. Ini tidak menjelaskan di mana dan bagaimana membaca data.

Saat memuat kumpulan data tanpa headerAWS Glue Studio, nama header default berbeda dari yang dimuat. AWS Glue DataBrew

1. Dalam pekerjaan ETL, tambahkan node Change Schema sebelum simpul Resep Persiapan Data.
2. Pilih simpul Ubah Skema dan ubah nama kolom ke nama yang sama yang digunakan dalam AWS Glue DataBrew resep.

Menggunakan Change Schema untuk memetakan ulang kunci properti data

Transformasi Change Schema memetakan ulang kunci properti data sumber ke dalam konfigurasi yang diinginkan untuk data target. Dalam simpul transformasi Skema Perubahan, Anda dapat:

- Mengubah nama beberapa kunci properti data.
- Mengubah tipe data kunci properti data, jika tipe data baru didukung dan ada path transformasi antara dua tipe data.
- Memilih subset kunci properti data dengan menunjukkan kunci properti data yang ingin Anda buang.

Anda juga dapat menambahkan node Change Schema tambahan ke diagram pekerjaan sesuai kebutuhan — misalnya, untuk memodifikasi sumber data tambahan atau mengikuti transformasi Gabung.

Menggunakan Change Schema dengan tipe data desimal

Saat menggunakan transformasi Change Schema dengan tipe data desimal, transformasi Change Schema memodifikasi presisi ke nilai default (10,2). Untuk memodifikasi ini dan mengatur presisi untuk kasus penggunaan Anda, Anda dapat menggunakan transformasi Kueri SQL dan mentransmisikan kolom dengan presisi tertentu.

Misalnya, jika Anda memiliki kolom input bernama "DecimalCol" dari jenis Desimal, dan Anda ingin memetakannya kembali ke kolom keluaran bernama "OutputDecimalCol" dengan presisi spesifik (18,6), Anda akan:

1. Tambahkan transformasi Query SQL berikutnya setelah transformasi Change Schema.
2. Dalam transformasi SQL Query, gunakan kueri SQL untuk mentransmisikan kolom yang dipetakan ulang ke presisi yang diinginkan. Kueri SQL akan terlihat seperti ini:

   ```
   SELECT col1, col2, CAST(DecimalCol AS DECIMAL(18,6)) AS OutputDecimalCol
   FROM __THIS__
   ```

   Dalam query SQL di atas:

   - `col1` dan `col2` adalah kolom lain dalam data Anda yang ingin Anda lewati tanpa modifikasi.
   - `DecimalCol` adalah nama kolom asli dari data input.
   - `CAST (DecimalCol AS DECIMAL (18,6)) `melemparkan` DecimalCol `ke tipe Desimal dengan presisi 18 digit dan 6 tempat desimal.
   - `AS OutputDecimalCol `mengganti nama kolom yang dicor menjadi` `. OutputDecimalCol

Dengan menggunakan transformasi SQL Query, Anda dapat mengganti presisi default yang ditetapkan oleh transformasi Change Schema dan secara eksplisit melemparkan kolom Desimal ke presisi yang diinginkan. Pendekatan ini memungkinkan Anda memanfaatkan transformasi Change Schema untuk mengganti nama dan merestrukturisasi data Anda sambil menangani persyaratan presisi untuk kolom Desimal melalui transformasi SQL Query berikutnya.

Menambahkan transformasi Change Schema ke pekerjaan Anda

> **Note**
>
> Transformasi Change Schema tidak peka huruf besar/kecil.

Untuk menambahkan node transformasi Change Schema ke diagram pekerjaan Anda

1. (Opsional) Buka panel Resource dan kemudian pilih Change Schema untuk menambahkan transformasi baru ke diagram pekerjaan Anda, jika diperlukan.
2. Di panel properti node, masukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pilih tab Transform di panel properti node.
4. Memodifikasi skema input:

- Untuk mengubah nama kunci properti data, masukkan nama baru untuk kunci tersebut di bidang Kunci target.
- Untuk mengubah tipe data untuk kunci properti data, pilih tipe data baru untuk kunci tersebut dari daftar Jenis data.
- Untuk menghapus kunci properti data dari skema target, pilih kotak centang Buang untuk kunci itu.

5. (Opsional) Setelah mengkonfigurasi properti simpul transformasi, Anda dapat melihat skema yang telah diubah untuk data Anda dengan memilih tab Skema output di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Jika Anda belum menentukan IAM role pada tab Detail tugas, maka Anda akan diminta untuk memasukkan IAM role di sini.
6. (Opsional) Setelah mengkonfigurasi properti simpul dan properti transformasi, Anda dapat melihat pratinjau set data yang diubah dengan memilih tab Pratinjau data di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Ada biaya yang terkait dengan penggunaan fitur ini, dan penagihan akan dimulai langsung setelah Anda memberikan IAM role.

Menggunakan Drop Duplikat

Transformasi Drop Duplicates menghapus baris dari sumber data Anda dengan memberi Anda dua opsi. Anda dapat memilih untuk menghapus baris duplikat yang benar-benar sama, atau Anda dapat memilih untuk memilih bidang yang cocok dan hanya menghapus baris tersebut berdasarkan bidang yang Anda pilih.

Misalnya, dalam kumpulan data ini, Anda memiliki baris duplikat di mana semua nilai di beberapa baris persis sama dengan baris lain, dan beberapa nilai dalam baris sama atau berbeda.

Baris	Nama	Email	Umur	Status	Catatan
1	Sukacita	sukacita @gmail	33	NY	
2	Tim	tim @gmail	45	OH	
3	Mawar	mawar @gmail	23	NJ	

Baris	Nama	Email	Umur	Status	Catatan
4	Tim	tim @gmail	42	OH	
5	Mawar	mawar @gmail	23	NJ	
6	Tim	tim @gmail	42	OH	ini adalah baris duplikat dan cocok sepenuhnya pada semua nilai sebagai baris #4
7	Mawar	mawar @gmail	23	NJ	Ini adalah baris duplikat dan cocok sepenuhnya pada semua nilai sebagai baris #5

Jika Anda memilih untuk mencocokkan seluruh baris, baris 6 dan 7 akan dihapus dari kumpulan data. Kumpulan data sekarang:

Baris	Nama	Email	Umur	Status
1	Sukacita	sukacita @gmail	33	NY
2	Tim	tim @gmail	45	OH
3	Mawar	mawar @gmail	23	NJ
4	Tim	tim @gmail	42	OH
5	Mawar	mawar @gmail	23	NJ

Jika Anda memilih untuk menentukan kunci, Anda dapat memilih untuk menghapus baris yang cocok dengan 'nama' dan 'email'. Ini memberi Anda kontrol yang lebih baik tentang apa yang dimaksud dengan 'baris duplikat' untuk kumpulan data Anda. Dengan menentukan 'nama' dan 'email', kumpulan data sekarang:

Baris	Nama	Email	Umur	Status
1	Sukacita	sukacita @gmail	33	NY
2	Tim	tim @gmail	45	OH
3	Mawar	mawar @gmail	23	NJ

Beberapa hal yang perlu diingat:

- Agar baris dikenali sebagai duplikat, nilainya peka huruf besar. semua nilai dalam baris harus memiliki casing yang sama - ini berlaku untuk salah satu opsi yang Anda pilih (Cocokkan seluruh baris atau Tentukan kunci).
- Semua nilai dibaca sebagai string.
- Transformasi Drop Duplicates menggunakan perintah Spark DropDuplicates.
- Saat menggunakan transformasi Drop Duplicates, baris pertama disimpan dan baris lainnya dijatuhkan.
- Transformasi Drop Duplicates tidak mengubah skema kerangka data. Jika Anda memilih untuk menentukan kunci, semua bidang disimpan dalam kerangka data yang dihasilkan.

Menggunakan SelectFields untuk menghapus sebagian besar kunci properti data

Anda dapat membuat subset kunci properti data dari dataset menggunakan transformasi. SelectFields Anda menunjukkan kunci properti data mana yang ingin Anda simpan dan sisanya akan dihapus dari set data.

> **Note**
>
> SelectFieldsTransformasinya peka huruf besar/kecil. Gunakan ApplyMappingjika Anda memerlukan cara case-insensitive untuk memilih bidang.

Untuk menambahkan node SelectFields transformasi ke diagram pekerjaan Anda

1. (Opsional) Buka panel Resource, lalu pilih SelectFieldsuntuk menambahkan transformasi baru ke diagram pekerjaan Anda, jika diperlukan.
2. Pada tab Properti simpul, masukkan nama untuk simpul dalam diagram tugas. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pilih tab Transformasi di panel detail simpul.
4. Di bawah judul SelectFields, pilih kunci properti data dalam kumpulan data yang ingin Anda simpan. Kunci properti data yang tidak dipilih akan dibuang dari set data.

   Anda juga dapat memilih kotak centang yang ada di samping judul kolom Bidang untuk secara otomatis memilih semua kunci properti data dalam set data. Kemudian Anda dapat membatalkan pilihan kunci properti data individu untuk menghapusnya dari set data.
5. (Opsional) Setelah mengkonfigurasi properti simpul transformasi, Anda dapat melihat skema yang telah diubah untuk data Anda dengan memilih tab Skema output di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Jika Anda belum menentukan IAM role pada tab Detail tugas, maka Anda akan diminta untuk memasukkan IAM role di sini.
6. (Opsional) Setelah mengkonfigurasi properti simpul dan properti transformasi, Anda dapat melihat pratinjau set data yang diubah dengan memilih tab Pratinjau data di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Ada biaya yang terkait dengan penggunaan fitur ini, dan penagihan akan dimulai langsung setelah Anda memberikan IAM role.

Menggunakan DropFields untuk menyimpan sebagian besar kunci properti data

Anda dapat membuat subset kunci properti data dari dataset menggunakan transformasi. DropFields Anda menunjukkan kunci properti data mana yang ingin Anda hapus dari set data dan kunci sisanya akan dipertahankan.

> **Note**
>
> DropFieldsTransformasinya peka huruf besar/kecil. Gunakan Ubah Skema jika Anda memerlukan cara yang tidak peka huruf besar/kecil untuk memilih bidang.

Untuk menambahkan node DropFields transformasi ke diagram pekerjaan Anda

1. (Opsional) Buka panel Resource dan kemudian pilih DropFieldsuntuk menambahkan transformasi baru ke diagram pekerjaan Anda, jika diperlukan.
2. Pada tab Properti simpul, masukkan nama untuk simpul dalam diagram tugas. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pilih tab Transformasi di panel detail simpul.
4. Di bawah judul DropFields, pilih kunci properti data untuk dijatuhkan dari sumber data.

   Anda juga dapat memilih kotak centang yang ada di samping judul kolom Bidang untuk secara otomatis memilih semua kunci properti data dalam set data. Kemudian Anda dapat membatalkan pilihan masing-masing kunci properti data untuk mempertahankannya dalam set data.
5. (Opsional) Setelah mengkonfigurasi properti simpul transformasi, Anda dapat melihat skema yang telah diubah untuk data Anda dengan memilih tab Skema output di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Jika Anda belum menentukan IAM role pada tab Detail tugas, maka Anda akan diminta untuk memasukkan IAM role di sini.
6. (Opsional) Setelah mengkonfigurasi properti simpul dan properti transformasi, Anda dapat melihat pratinjau set data yang diubah dengan memilih tab Pratinjau data di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Ada biaya yang terkait dengan penggunaan fitur ini, dan penagihan akan dimulai langsung setelah Anda memberikan IAM role.

Mengganti nama sebuah bidang dalam set data

Anda dapat menggunakan RenameFieldtransformasi untuk mengubah nama kunci properti individual dalam kumpulan data.

> **Note**
>
> RenameFieldTransformasinya peka huruf besar/kecil. Gunakan ApplyMappingjika Anda membutuhkan transformasi case-insensitive.

> **Tip**
>
> Jika Anda menggunakan transformasi Change Schema, Anda dapat mengganti nama beberapa kunci properti data dalam dataset dengan satu transformasi.

Untuk menambahkan node RenameField transformasi ke diagram pekerjaan Anda

1. (Opsional) Buka panel Resource dan kemudian pilih RenameFielduntuk menambahkan transformasi baru ke diagram pekerjaan Anda, jika diperlukan.
2. Pada tab Properti simpul, masukkan nama untuk simpul dalam diagram tugas. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pilih tab Transformasi.
4. Pada judul Bidang data, pilih sebuah kunci properti dari sumber data dan kemudian masukkan nama baru di bidang Nama bidang baru.
5. (Opsional) Setelah mengkonfigurasi properti simpul transformasi, Anda dapat melihat skema yang telah diubah untuk data Anda dengan memilih tab Skema output di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Jika Anda belum menentukan IAM role pada tab Detail tugas, maka Anda akan diminta untuk memasukkan IAM role di sini.
6. (Opsional) Setelah mengkonfigurasi properti simpul dan properti transformasi, Anda dapat melihat pratinjau set data yang diubah dengan memilih tab Pratinjau data di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Ada biaya yang terkait dengan penggunaan fitur ini, dan penagihan akan dimulai langsung setelah Anda memberikan IAM role.

Menggunakan Spigot untuk mengambil sampel dari set data Anda

Untuk menguji transformasi yang dilakukan oleh tugas Anda, Anda mungkin ingin mendapatkan sampel data untuk memeriksa apakah transformasi bekerja sebagaimana mestinya. Transformasi Spigot menulis subset catatan dari set data ke file JSON dalam sebuah bucket Amazon S3. Metode pengambilan sampel data dapat berupa jumlah tertentu dari catatan dari awal file atau faktor probabilitas yang digunakan untuk memilih catatan.

Untuk menambahkan simpul transformasi Spigot ke diagram tugas Anda

1. (Opsional) Buka panel Resource dan kemudian pilih Spigot untuk menambahkan transformasi baru ke diagram pekerjaan Anda, jika diperlukan.
2. Pada tab Properti simpul, masukkan nama untuk simpul dalam diagram tugas. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pilih tab Transformasi di panel detail simpul.
4. Masukkan sebuah path Amazon S3 atau pilih Jelajahi S3 untuk memilih lokasi di Amazon S3. Ini adalah lokasi di mana tugas menulis file JSON yang berisi sampel data.
5. Masukkan informasi untuk metode pengambilan sampel. Anda dapat menentukan nilai untuk Jumlah catatan untuk menulis mulai dari awal set data dan Ambang probabilitas (dimasukkan sebagai nilai desimal dengan nilai maksimum 1) dari pemilihan catatan yang diberikan.

   Misalnya, untuk menulis 50 catatan pertama dari set data, Anda akan mengatur Jumlah catatan ke 50 dan Ambang probabilitas ke 1 (100%).

Menggabungkan set data

Transformasi Join memungkinkan Anda untuk menggabungkan dua set data menjadi satu. Anda menentukan nama-nama kunci dalam skema dari setiap set data yang akan dibandingkan. Output `DynamicFrame` berisi baris di mana kunci memenuhi syarat penggabungan. Baris di setiap set data yang memenuhi syarat penggabungan digabungkan menjadi satu baris dalam output `DynamicFrame` yang berisi semua kolom yang ditemukan di salah satu set data.

Untuk menambahkan simpul transformasi Join ke diagram tugas Anda

1. Jika hanya ada satu sumber data yang tersedia, maka Anda harus menambahkan simpul sumber data baru ke diagram tugas.
2. Pilih salah satu simpul sumber untuk penggabungan. Buka panel Resource dan kemudian pilih Gabung untuk menambahkan transformasi baru ke diagram pekerjaan Anda.
3. Pada tab Properti simpul, masukkan nama untuk simpul dalam diagram tugas.
4. Pada tab Properti simpul, pada judul Simpul, tambahkan simpul induk sehingga ada dua set data yang menyediakan masukan untuk penggabungan. Induk dapat berupa simpul sumber data atau simpul transformasi.

> **Note**
>
> Sebuah transformasi join hanya dapat memiliki dua simpul induk.

5. Pilih tab Transformasi.

   Jika Anda melihat pesan yang menunjukkan bahwa ada nama kunci yang bertentangan, Anda dapat:

   - Pilih Selesaikan untuk secara otomatis menambahkan node ApplyMappingtransformasi ke diagram pekerjaan Anda. ApplyMapping Node menambahkan awalan ke kunci apa pun dalam kumpulan data yang memiliki nama yang sama dengan kunci di kumpulan data lainnya. Sebagai contoh, jika Anda menggunakan nilai default **`right`**, maka setiap kunci dalam set data kanan yang memiliki nama yang sama dengan kunci pada set data kiri akan diubah namanya menjadi `(right)`*`key name`*.
   - Secara manual, tambahkan simpul transformasi sebelumnya dalam diagram tugas untuk menghapus atau mengubah nama kunci yang bertentangan.

6. Pilih jenis penggabungan dalam daftar Jenis penggabungan.

   - Penggabungan dalam: Mengembalikan baris dengan kolom dari kedua set data untuk setiap kecocokan berdasarkan syarat penggabungan. Baris yang tidak memenuhi syarat penggabungan tidak dikembalikan.
   - Penggabungan kiri: Semua baris dari set data kiri dan hanya baris dari set data kanan yang memenuhi syarat penggabungan.
   - Penggabungan kanan: Semua baris dari set data kanan dan hanya baris dari set data kiri yang memenuhi syarat penggabungan.
   - Penggabungan luar: Semua baris dari kedua set data.
   - Penggabungan semi kiri: Semua baris dari set data kiri yang memiliki kecocokan di set data kanan berdasarkan syarat penggabungan.
   - Penggabungan kebalikan kiri: Semua baris di set data kiri yang tidak memiliki kecocokan di set data kanan berdasarkan syarat penggabungan.

7. Pada tab Transformasi, pada judul Syarat penggabungan, pilih Tambahkan syarat. Pilih kunci properti dari setiap set data yang akan dibandingkan. Kunci properti di sisi kiri operator perbandingan disebut sebagai set data kiri dan kunci properti di sebelah kanan disebut sebagai set data kanan.

Untuk syarat penggabungan yang lebih kompleks, Anda dapat menambahkan kunci pencocokan tambahan dengan memilih Tambahkan syarat lebih dari sekali. Jika Anda secara tidak sengaja menambahkan sebuah syarat, Anda dapat memilih ikon hapus ( ) untuk menghapusnya.

8. (Opsional) Setelah mengkonfigurasi properti simpul transformasi, Anda dapat melihat skema yang telah diubah untuk data Anda dengan memilih tab Skema output di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Jika Anda belum menentukan IAM role pada tab Detail tugas, maka Anda akan diminta untuk memasukkan IAM role di sini.
9. (Opsional) Setelah mengkonfigurasi properti simpul dan properti transformasi, Anda dapat melihat pratinjau set data yang diubah dengan memilih tab Pratinjau data di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Ada biaya yang terkait dengan penggunaan fitur ini, dan penagihan akan dimulai langsung setelah Anda memberikan IAM role.

Untuk contoh skema output penggabungan, pertimbangkan penggabungan antara dua set data dengan kunci properti berikut:

```
Left: {id, dept, hire_date, salary, employment_status}
Right: {id, first_name, last_name, hire_date, title}
```

Penggabungan dikonfigurasi untuk mencocokkan berdasarkan kunci `id` dan `hire_date` menggunakan operator perbandingan =.

Karena kedua set data mengandung kunci `id` dan `hire_date`, Anda pilih Ubah untuk secara otomatis menambahkan prefiks **`right`** ke kunci dalam set data kanan.

Kunci dalam skema output akan:

```
{id, dept, hire_date, salary, employment_status,
(right)id, first_name, last_name, (right)hire_date, title}
```

Menggunakan Union untuk menggabungkan baris

Anda menggunakan node transformasi Union ketika Anda ingin menggabungkan baris dari lebih dari satu sumber data yang memiliki skema yang sama.

Ada beberapa jenis transformasi Union:

1. SEMUA — saat menerapkan SEMUA, serikat yang dihasilkan tidak menghapus baris duplikat.
2. DISTINCT — saat menerapkan DISTINCT, gabungan yang dihasilkan menghapus baris duplikat.

Serikat vs Bergabung

Anda menggunakan Union untuk menggabungkan baris. Anda menggunakan Gabung untuk menggabungkan kolom.

Menggunakan transformasi Union di kanvas Visual ETL

1. Tambahkan lebih dari satu sumber data untuk melakukan transformasi serikat. Untuk menambahkan sumber data, buka Panel Sumber Daya, lalu pilih sumber data dari tab Sumber. Sebelum menggunakan transformasi Union, Anda harus memastikan bahwa semua sumber data yang terlibat dalam serikat memiliki skema dan struktur yang sama.
2. Bila Anda memiliki setidaknya dua sumber data yang ingin Anda gabungkan menggunakan transformasi Union, buat transformasi Union dengan menambahkannya ke kanvas. Buka Resource Panel di kanvas dan cari 'Union'. Anda juga dapat memilih tab Transforms di Resource Panel dan gulir ke bawah sampai Anda menemukan Union transform, lalu pilih Union.
3. Pilih node Union pada kanvas pekerjaan. Di jendela properti Node, pilih node induk untuk terhubung ke transformasi Union.
4. AWS Gluememeriksa kompatibilitas untuk memastikan bahwa transformasi Union dapat diterapkan ke semua sumber data. Jika skema untuk sumber data sama, operasi akan diizinkan. Jika sumber data tidak memiliki skema yang sama, pesan kesalahan yang tidak valid ditampilkan: “Skema input dari gabungan ini tidak sama. Pertimbangkan ApplyMapping untuk menggunakan untuk mencocokkan skema.” Untuk memperbaikinya, pilih Gunakan ApplyMapping.
5. Pilih jenis Union.
    1. Semua - Secara default, tipe All Union dipilih; ini akan menghasilkan baris duplikat jika ada dalam kombinasi data.
    2. Distinct - Pilih Distinct jika Anda ingin baris duplikat dihapus dari kombinasi data yang dihasilkan.

Menggunakan SplitFields untuk membagi kumpulan data menjadi dua

SplitFieldsTransformasi memungkinkan Anda untuk memilih beberapa kunci properti data dalam dataset input dan memasukkannya ke dalam satu kumpulan data dan kunci yang tidak dipilih ke dalam kumpulan data terpisah. Output dari transformasi ini adalah sebuah kumpulan `DynamicFrames`.

> **Note**
>
> Anda harus menggunakan SelectFromCollectiontransformasi untuk mengubah koleksi `DynamicFrames` menjadi satu `DynamicFrame` sebelum Anda dapat mengirim output ke lokasi target.

SplitFieldsTransformasinya peka huruf besar/kecil. Tambahkan ApplyMappingtransformasi sebagai node induk jika Anda memerlukan nama kunci properti case-insensitive.

Untuk menambahkan node SplitFields transformasi ke diagram pekerjaan Anda

1. (Opsional) Buka panel Resource dan kemudian pilih SplitFieldsuntuk menambahkan transformasi baru ke diagram pekerjaan Anda, jika diperlukan.
2. Pada tab Properti simpul, masukkan nama untuk simpul dalam diagram tugas. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pilih tab Transformasi.
4. Pilih kunci properti mana yang ingin Anda masukkan ke dalam set data pertama. Kunci yang tidak Anda pilih akan ditempatkan di set data kedua.
5. (Opsional) Setelah mengkonfigurasi properti simpul transformasi, Anda dapat melihat skema yang telah diubah untuk data Anda dengan memilih tab Skema output di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Jika Anda belum menentukan IAM role pada tab Detail tugas, maka Anda akan diminta untuk memasukkan IAM role di sini.
6. (Opsional) Setelah mengkonfigurasi properti simpul dan properti transformasi, Anda dapat melihat pratinjau set data yang diubah dengan memilih tab Pratinjau data di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Ada biaya yang terkait dengan penggunaan fitur ini, dan penagihan akan dimulai langsung setelah Anda memberikan IAM role.

7. Konfigurasikan node SelectFromCollectiontransformasi untuk memproses kumpulan data yang dihasilkan.

Gambaran umum transformasi SelectFromCollection

Transformasi tertentu memiliki beberapa kumpulan data sebagai outputnya, bukan satu kumpulan data, misalnya,. SplitFields SelectFromCollectionTransformasi memilih satu dataset (`DynamicFrame`) dari kumpulan dataset (array). `DynamicFrames` Output untuk transformasi tersebut adalah `DynamicFrame` yang dipilih.

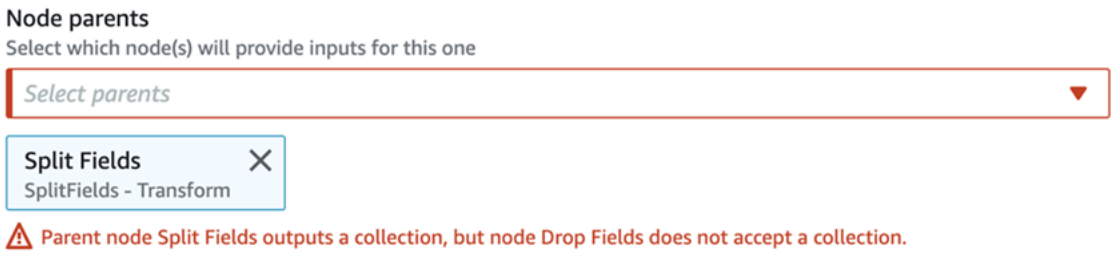
Anda harus menggunakan transformasi ini setelah Anda menggunakan transformasi yang menciptakan koleksi `DynamicFrames`, seperti:

- Transformasi kode kustom
- SplitFields

Jika Anda tidak menambahkan node SelectFromCollectiontransformasi ke diagram pekerjaan Anda setelah salah satu dari transformasi ini, Anda akan mendapatkan kesalahan untuk pekerjaan Anda.

Induk simpul untuk transformasi ini haruslah sebuah simpul yang mengembalikan koleksi `DynamicFrames`. Jika Anda memilih sebuah induk untuk simpul transformasi yang mengembalikan satu `DynamicFrame` ini, seperti transformasi Join, maka tugas Anda akan mengembalikan kesalahan.

Demikian pula, jika Anda menggunakan SelectFromCollectionnode dalam diagram pekerjaan Anda sebagai induk untuk transformasi yang mengharapkan satu `DynamicFrame` sebagai input, pekerjaan Anda mengembalikan kesalahan.



Menggunakan SelectFromCollection untuk memilih kumpulan data mana yang akan disimpan

Gunakan SelectFromCollectiontransformasi untuk mengubah koleksi `DynamicFrames` menjadi satu`DynamicFrame`.

Untuk menambahkan node SelectFromCollection transformasi ke diagram pekerjaan Anda

1. (Opsional) Buka panel Resource dan kemudian pilih SelectFromCollectionuntuk menambahkan transformasi baru ke diagram pekerjaan Anda, jika diperlukan.
2. Pada tab Properti simpul, masukkan nama untuk simpul dalam diagram tugas. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pilih tab Transformasi.
4. Pada judul Indeks bingkai, pilih nomor indeks array yang sesuai dengan `DynamicFrame` yang ingin Anda pilih dari koleksi `DynamicFrames` tersebut.

   Misalnya, jika node induk untuk transformasi ini adalah SplitFieldstransformasi, pada tab skema Output dari node itu Anda dapat melihat skema untuk masing-masing. `DynamicFrame` Jika Anda ingin menyimpan `DynamicFrame` yang dikaitkan dengan skema untuk Output 2, Anda akan memilih **1** untuk nilai Indeks bingkai, yang merupakan nilai kedua dalam daftar.

   Hanya `DynamicFrame` yang Anda pilih yang disertakan dalam output.
5. (Opsional) Setelah mengkonfigurasi properti simpul transformasi, Anda dapat melihat skema yang telah diubah untuk data Anda dengan memilih tab Skema output di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Jika Anda belum menentukan IAM role pada tab Detail tugas, maka Anda akan diminta untuk memasukkan IAM role di sini.
6. (Opsional) Setelah mengkonfigurasi properti simpul dan properti transformasi, Anda dapat melihat pratinjau set data yang diubah dengan memilih tab Pratinjau data di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Ada biaya yang terkait dengan penggunaan fitur ini, dan penagihan akan dimulai langsung setelah Anda memberikan IAM role.

Menemukan dan mengisi nilai yang hilang dalam set data

Anda dapat menggunakan FillMissingValuestransformasi untuk menemukan catatan dalam kumpulan data yang memiliki nilai yang hilang dan menambahkan bidang baru dengan nilai yang ditentukan oleh imputasi. Set data input digunakan untuk melatih model machine learning (ML) yang menentukan apa nilai yang hilang seharusnya. Jika Anda menggunakan kumpulan data tambahan, maka setiap set tambahan digunakan sebagai data pelatihan untuk model ML, sehingga hasilnya mungkin tidak akurat.

Untuk menggunakan node FillMissingValues transformasi dalam diagram pekerjaan Anda

1. (Opsional) Buka panel Resource dan kemudian pilih FillMissingValuesuntuk menambahkan transformasi baru ke diagram pekerjaan Anda, jika diperlukan.
2. Pada tab Properti simpul, masukkan nama untuk simpul dalam diagram tugas. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pilih tab Transformasi.
4. Untuk Bidang data, pilih kolom atau nama kolom dari sumber data yang ingin Anda analisis nilai-nilai hilangnya.
5. (Opsional) Dalam bidang Nama bidang baru, masukkan nama untuk bidang yang ditambahkan ke setiap catatan yang akan menyimpan nilai pengganti perkiraan untuk bidang yang dianalisis. Jika bidang yang dianalisis tidak memiliki nilai yang hilang, maka nilai di bidang yang dianalisis akan disalin ke bidang baru.

   Jika Anda tidak menentukan nama untuk bidang baru, maka nama default-nya adalah nama kolom yang dianalisis dengan ditambahkan `_filled`. Misalnya, jika Anda memasukkan **`Age`** untuk Bidang data dan tidak menentukan nilai untuk Nama bidang baru, maka sebuah bidang baru bernama **`Age_filled`** akan ditambahkan ke setiap catatan.
6. (Opsional) Setelah mengkonfigurasi properti simpul transformasi, Anda dapat melihat skema yang telah diubah untuk data Anda dengan memilih tab Skema output di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Jika Anda belum menentukan IAM role pada tab Detail tugas, maka Anda akan diminta untuk memasukkan IAM role di sini.
7. (Opsional) Setelah mengkonfigurasi properti simpul dan properti transformasi, Anda dapat melihat pratinjau set data yang diubah dengan memilih tab Pratinjau data di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta

untuk memberikan IAM role untuk mengakses data. Ada biaya yang terkait dengan penggunaan fitur ini, dan penagihan akan dimulai langsung setelah Anda memberikan IAM role.

Mem-filter kunci dalam set data

Gunakan transformasi Filter untuk membuat set data baru dengan mem-filter catatan dari set data masukan berdasarkan ekspresi reguler. Baris yang tidak memenuhi syarat filter dihapus dari output.

- Untuk tipe data string, Anda dapat mem-filter baris di mana nilai kunci cocok dengan string yang ditentukan.
- Untuk tipe data numerik, Anda dapat mem-filter baris dengan membandingkan nilai kunci untuk nilai yang ditentukan menggunakan operator perbandingan <, >, =, !=, <=, dan >=.

Jika Anda menentukan beberapa syarat filter, hasilnya digabungkan menggunakan operator `AND` secara default, tetapi Anda dapat memilih `OR` sebagai gantinya.

Transformasi Filter peka huruf besar-kecil. Tambahkan ApplyMappingtransformasi sebagai node induk jika Anda memerlukan nama kunci properti case-insensitive.

Untuk menambahkan simpul transformasi Filter ke diagram tugas Anda

1. (Opsional) Buka panel Resource dan kemudian pilih Filter untuk menambahkan transformasi baru ke diagram pekerjaan Anda, jika diperlukan.
2. Pada tab Properti simpul, masukkan nama untuk simpul dalam diagram tugas. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pilih tab Transformasi.
4. Pilih salah satu Global AND atau Global OR. Ini menentukan bagaimana beberapa syarat filter digabungkan. Semua syarat digabungkan menggunakan operasi `AND` atau `OR`. Jika Anda hanya memiliki satu syarat filter, maka Anda dapat memilih salah satu.
5. Pilih tombol Tambahkan syarat di bagian Syarat filter untuk menambahkan syarat filter.

   Di bidang Kunci, pilih nama sebuah kunci properti dari set data. Di bidang Operasi, pilih operator perbandingan. Di bidang Nilai, masukkan nilai perbandingan. Berikut ini adalah beberapa contoh syarat filter:

   - `year >= 2018`

- `State matches 'CA*'`

  Ketika Anda mem-filter nilai-nilai string, pastikan bahwa nilai perbandingan menggunakan format ekspresi reguler yang cocok dengan bahasa skrip yang dipilih di properti tugas (Python atau Scala).

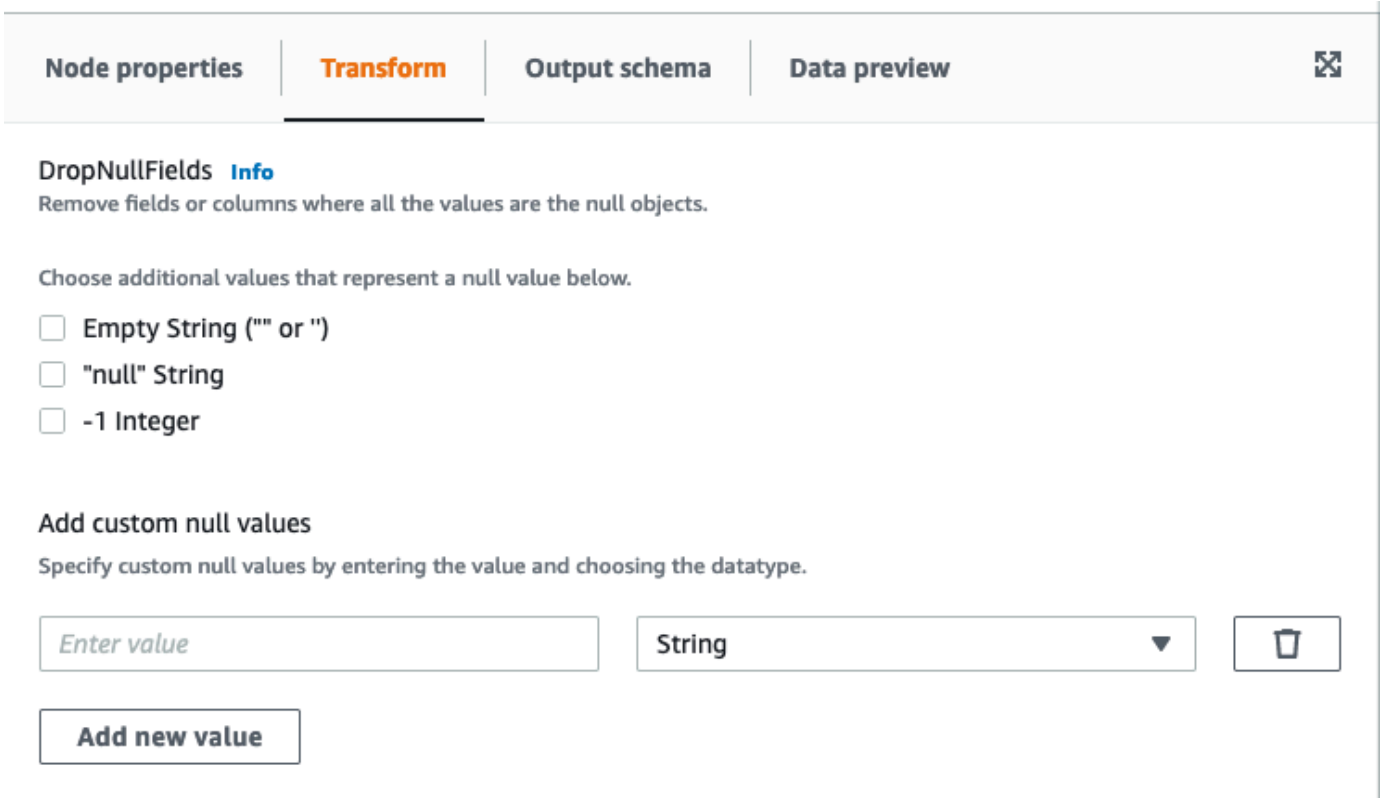
6. Tambahkan syarat filter tambahan, sesuai kebutuhan.
7. (Opsional) Setelah mengkonfigurasi properti simpul transformasi, Anda dapat melihat skema yang telah diubah untuk data Anda dengan memilih tab Skema output di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Jika Anda belum menentukan IAM role pada tab Detail tugas, maka Anda akan diminta untuk memasukkan IAM role di sini.
8. (Opsional) Setelah mengkonfigurasi properti simpul dan properti transformasi, Anda dapat melihat pratinjau set data yang diubah dengan memilih tab Pratinjau data di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Ada biaya yang terkait dengan penggunaan fitur ini, dan penagihan akan dimulai langsung setelah Anda memberikan IAM role.

Menggunakan DropNullFields untuk menghapus bidang dengan nilai null

Gunakan DropNullFieldstransformasi untuk menghapus bidang dari kumpulan data jika semua nilai di bidang adalah 'null'. Secara default, AWS Glue Studio akan mengenali objek null, tetapi beberapa nilai seperti string kosong, string yang “null”, -1 integer atau placeholder lain seperti nol, tidak secara otomatis dikenali sebagai nol.

Untuk menggunakan DropNullFields

1. Tambahkan DropNullFields simpul ke diagram pekerjaan.
2. Pada tab properti Node, pilih nilai tambahan yang mewakili nilai nol. Anda dapat memilih untuk memilih tidak ada atau semua nilai:



- String Kosong (“” atau “) - bidang yang berisi string kosong akan dihapus
- “null string” - bidang yang berisi string dengan kata 'null' akan dihapus
- -1 integer - bidang yang berisi -1 (negatif satu) integer akan dihapus

3. Jika diperlukan, Anda juga dapat menentukan nilai null kustom. Ini adalah nilai nol yang mungkin unik untuk kumpulan data Anda. Untuk menambahkan nilai null kustom, pilih Tambahkan nilai baru.
4. Masukkan nilai null kustom. Misalnya, ini bisa nol, atau nilai apa pun yang digunakan untuk mewakili nol dalam kumpulan data.
5. Pilih tipe data di bidang drop-down. Tipe data dapat berupa String atau Integer.

> **Note**
>
> Nilai null kustom dan tipe datanya harus sama persis agar bidang dikenali sebagai nilai nol dan bidang dihapus. Pencocokan sebagian di mana hanya nilai null kustom yang cocok tetapi tipe data tidak tidak akan mengakibatkan bidang dihapus.

Menggunakan kueri SQL untuk mengubah data

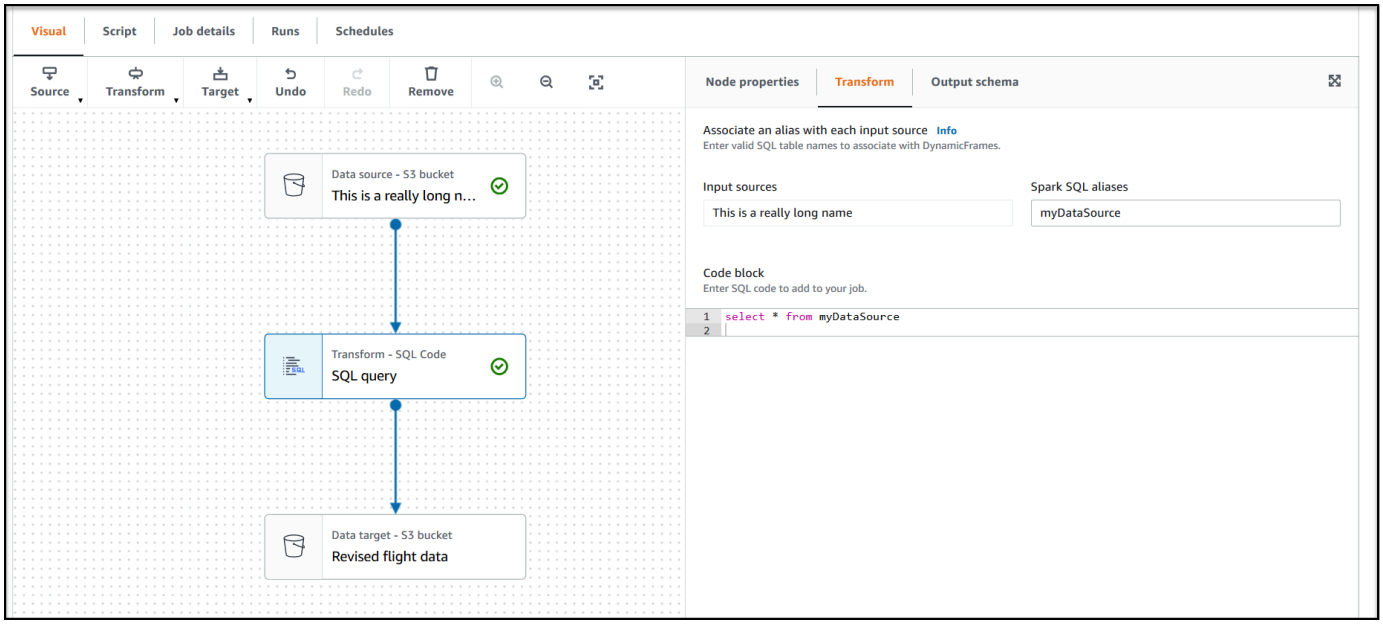
Anda dapat menggunakan transformasi SQL untuk menulis transformasi Anda sendiri dalam bentuk kueri SQL.

Sebuah simpul transformasi SQL dapat memiliki beberapa set data sebagai input, tetapi hanya menghasilkan set data tunggal sebagai output. Ia berisi bidang teks, di mana Anda memasukkan kueri Apache SparkSQL. Anda dapat menetapkan alias untuk setiap set data yang digunakan sebagai masukan, untuk membantu hanya kueri SQL. Untuk informasi lebih lanjut tentang sintaksis SQL, lihat dokumentasi Spark SQL.

> **Note**
>
> Jika Anda menggunakan transformasi Spark SQL dengan sumber data yang terletak di VPC, tambahkan VPC endpoint AWS Glue ke VPC yang berisi sumber data. Untuk informasi selengkapnya tentang konfigurasi titik akhir pengembangan, lihat Menambah Titik Akhir Pengembangan, Menyiapkan Lingkungan Anda untuk Titik Akhir Pengembangan, dan Mengakses Titik Akhir Pengembangan Anda dalam Panduan Developer AWS Glue.

Untuk menggunakan simpul transformasi SQL ke diagram tugas Anda

1. (Opsional) Tambahkan simpul transformasi ke diagram tugas, jika diperlukan. Pilih SQL Spark untuk jenis simpul.
2. Pada tab Properti simpul, masukkan nama untuk simpul dalam diagram tugas. Jika sebuah simpul induk belum dipilih, atau jika Anda ingin beberapa masukan untuk transformasi SQL, pilih sebuah simpul dari Induk simpul yang akan digunakan sebagai sumber masukan untuk transformasi. Tambahkan simpul induk tambahan sesuai kebutuhan.
3. Pilih tab Transformasi di panel detail simpul.
4. Set data sumber untuk kueri SQL diidentifikasi berdasarkan nama yang Anda tentukan dalam bidang Nama untuk setiap simpul. Jika Anda tidak ingin menggunakan nama-nama ini, atau jika nama-nama tidak cocok untuk kueri SQL, maka Anda dapat mengaitkan nama untuk setiap set data. Konsol tersebut menyediakan alias default, seperti `MyDataSource`.



Sebagai contoh, jika induk simpul untuk simpul transformasi SQL bernama `Rename Org PK field`, Anda dapat mengaitkan nama `org_table` dengan set data ini. Alias ini kemudian dapat digunakan dalam kueri SQL sebagai ganti nama simpul.

5. Dalam bidang entri teks pada judul Blok kode, tempel atau masukkan kueri SQL. Bidang teks menampilkan penyorotan sintaksis SQL dan saran kata kunci.
6. Dengan simpul transformasi SQL yang sudah dipilih, pilih tab Skema output, dan kemudian pilih Edit. Berikan kolom dan tipe data yang menggambarkan bidang output kueri SQL.

   Tentukan skema menggunakan tindakan berikut di bagian Skema output pada halaman tersebut:

   - Untuk mengubah nama kolom, tempatkan kursor di kotak teks Kunci untuk kolom (juga disebut sebagai bidang atau kunci properti) dan masukkan nama baru.
   - Untuk mengubah tipe data untuk kolom, pilih tipe data baru untuk kolom tersebut dari daftar drop-down.
   - Untuk menambahkan kolom tingkat atas baru pada skema, pilih tombol Overflow (⋯), dan kemudian pilih Tambah kunci akar. Kolom baru ditambahkan di bagian atas skema.
   - Untuk menghapus kolom dari skema, pilih ikon hapus (🗑) di ujung kanan nama kunci.

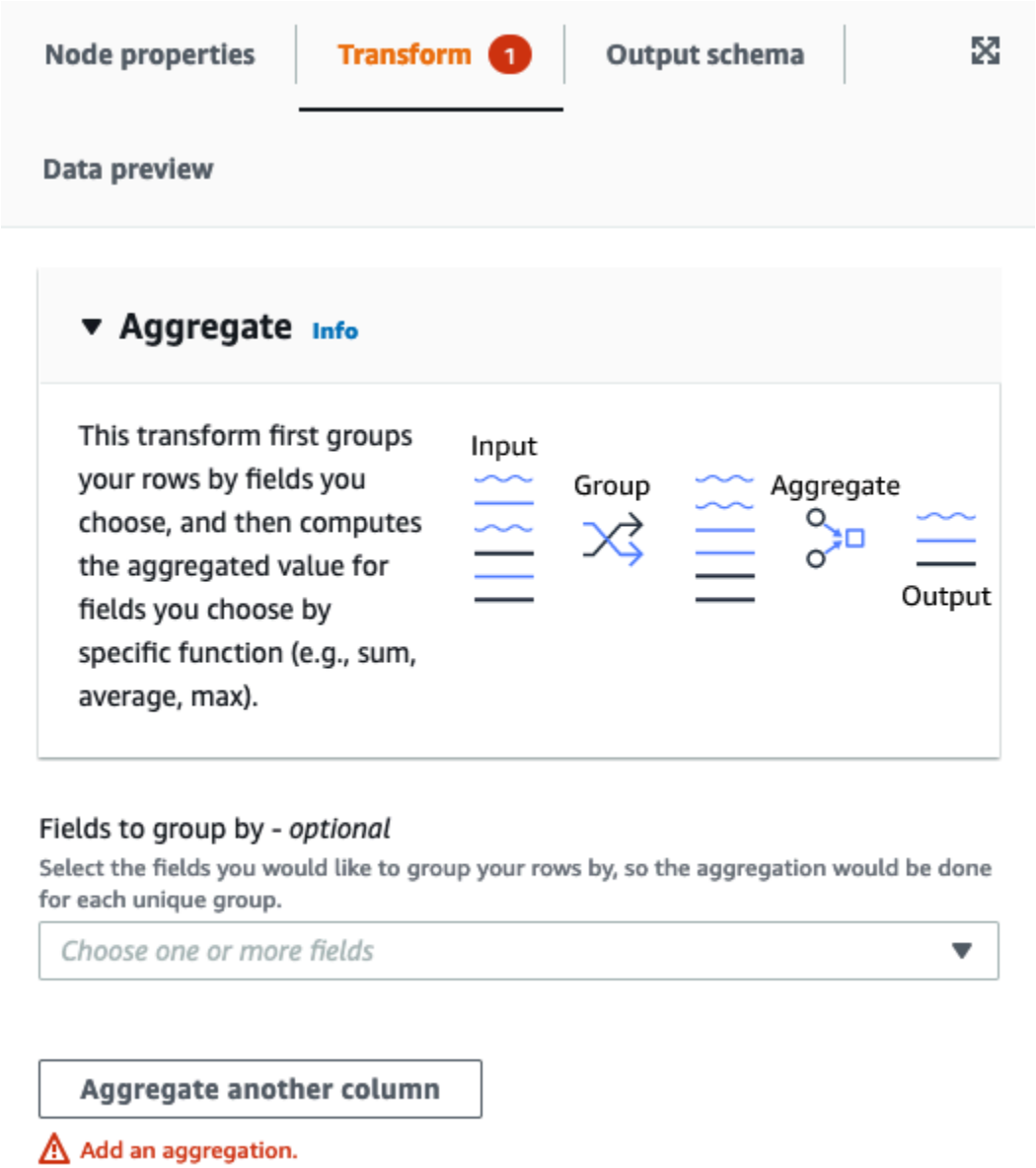
7. Setelah Anda selesai menentukan skema output, pilih Terapkan untuk menyimpan perubahan dan keluar dari editor skema. Jika Anda tidak ingin menyimpan perubahan, pilih Batalkan untuk mengedit editor skema.
8. (Opsional) Setelah mengkonfigurasi properti simpul dan properti transformasi, Anda dapat melihat pratinjau set data yang diubah dengan memilih tab Pratinjau data di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Ada biaya yang terkait dengan penggunaan fitur ini, dan penagihan akan dimulai langsung setelah Anda memberikan IAM role.

Menggunakan Agregat untuk melakukan perhitungan ringkasan pada bidang yang dipilih

Untuk menggunakan transformasi Agregat

1. Tambahkan node Agregat ke diagram pekerjaan.
2. Pada tab Properti Node, pilih bidang untuk dikelompokkan bersama dengan memilih bidang drop-down (opsional). Anda dapat memilih lebih dari satu bidang sekaligus atau mencari nama bidang dengan mengetikkan di bilah pencarian.

   Ketika bidang dipilih, nama dan tipe data ditampilkan. Untuk menghapus bidang, pilih 'X' di bidang.



3. Pilih Agregat kolom lain. Diperlukan untuk memilih setidaknya satu bidang.

4. Pilih bidang di bidang untuk agregat drop-down.
5. Pilih fungsi agregasi untuk diterapkan ke bidang yang dipilih:

- rata-rata - menghitung rata-rata
- CountDistinct - menghitung jumlah nilai non-null yang unik
- count - menghitung jumlah nilai non-null
- pertama - mengembalikan nilai pertama yang memenuhi kriteria 'kelompok dengan'
- last - mengembalikan nilai terakhir yang memenuhi kriteria 'grup dengan'
- kurtosis - menghitung ketajaman puncak kurva distribusi frekuensi
- max - mengembalikan nilai tertinggi yang memenuhi kriteria 'kelompok dengan'
- min - mengembalikan nilai terendah yang memenuhi kriteria 'kelompok dengan'
- kemiringan - ukuran asimetri distribusi probabilitas distribusi normal
- stddev_pop - menghitung standar deviasi populasi dan mengembalikan akar kuadrat dari varians populasi
- jumlah - jumlah semua nilai dalam kelompok
- SumDistinct - jumlah nilai yang berbeda dalam kelompok
- var_samp - varians sampel grup (mengabaikan nol)
- var_pop - varians populasi grup (mengabaikan nol)

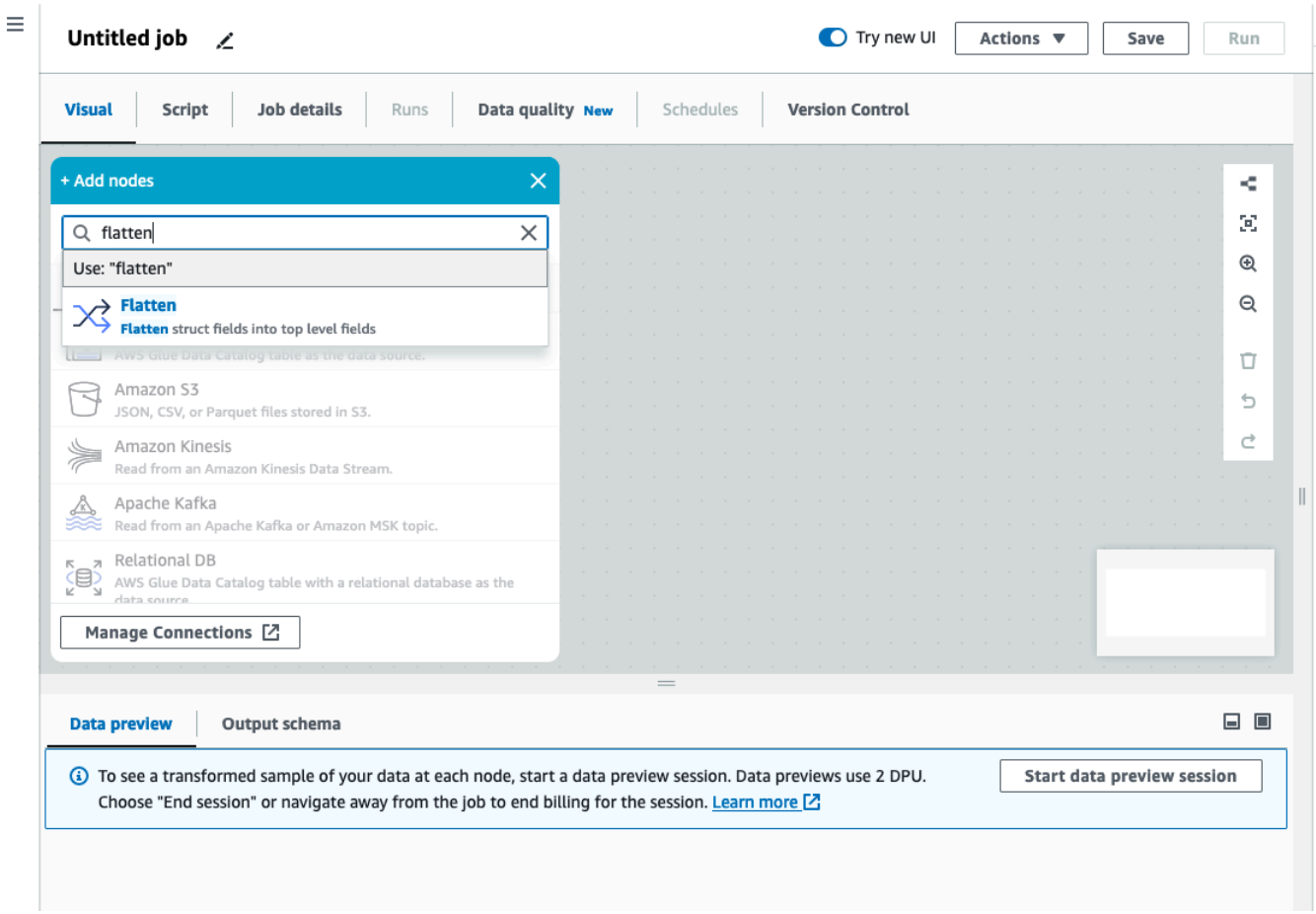
Ratakan struct bersarang

Ratakan bidang struct bersarang dalam data, sehingga menjadi bidang tingkat atas. Bidang baru diberi nama menggunakan nama bidang yang diawali dengan nama bidang struct untuk mencapainya, dipisahkan oleh titik-titik.

Misalnya, jika data memiliki bidang tipe Struct bernama “phone_numbers”, yang di antara bidang lainnya memiliki salah satu tipe “Struct” bernama “home_phone” dengan dua bidang: “country_code” dan “number”. Setelah diratakan, kedua bidang ini akan menjadi bidang tingkat atas bernama: “phone_numbers.home_phone.country_code” dan “phone_numbers.home_phone.number” masing-masing.

Untuk menambahkan node transformasi Flatten dalam diagram pekerjaan Anda

1. Buka panel Resource dan kemudian pilih tab Transforms, lalu Flatten untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Anda juga dapat menggunakan bilah pencarian dengan memasukkan 'Ratakan', lalu mengklik simpul Flatten. Node yang dipilih pada saat menambahkan node akan menjadi induknya.



2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. (Opsional) Pada tab Transform, Anda dapat membatasi level bersarang maksimum untuk meratakan. Misalnya, menyetel nilai itu ke 1 berarti hanya struct tingkat atas yang akan diratakan. Mengatur max ke 2 akan meratakan level atas dan struct langsung di bawahnya.

Tambahkan kolom UUID

Saat Anda menambahkan kolom UUID (Universally Unique Identified), setiap baris akan diberi string 36 karakter yang unik.

Untuk menambahkan node transformasi UUID dalam diagram pekerjaan Anda

1. Buka panel Resource dan kemudian pilih UUID untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.

2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. (Opsional) Pada Transform tab, Anda dapat menyesuaikan nama kolom baru. Secara default akan diberi nama “uuid”.

Tambahkan kolom pengenal

Tetapkan Identifier numerik untuk setiap baris dalam kumpulan data.

Untuk menambahkan node transformasi Identifier dalam diagram pekerjaan Anda

1. Buka panel Resource dan kemudian pilih Identifier untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. (Opsional) Pada Transform tab, Anda dapat menyesuaikan nama kolom baru. Secara default, itu akan diberi nama “id”.
4. (Opsional) Jika pekerjaan memproses dan menyimpan data secara bertahap, Anda ingin menghindari id yang sama untuk digunakan kembali di antara pekerjaan berjalan.

   Pada tab Transform, tandai opsi kotak centang unik. Ini akan menyertakan stempel waktu pekerjaan di pengenal, membuatnya unik di antara beberapa proses. Untuk memungkinkan angka yang lebih besar, kolom bukan tipe panjang akan menjadi desimal.

Ubah kolom menjadi tipe stempel waktu

Anda dapat menggunakan stempel waktu transformasi Ke untuk mengubah tipe data kolom numerik atau string menjadi stempel waktu, sehingga dapat disimpan dengan tipe data tersebut atau diterapkan ke transformasi lain yang memerlukan stempel waktu.

Untuk menambahkan simpul transformasi To timestamp di diagram pekerjaan Anda

1. Buka panel Resource dan kemudian pilih To timestamp untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pada tab Transform, masukkan nama kolom yang akan dikonversi.
4. Pada tab Transform, tentukan cara mengurai kolom yang dipilih dengan memilih jenisnya.

   Jika nilainya adalah angka, itu dapat dinyatakan dalam detik (stempel waktu Unix/Python), milidetik atau mikrodetik, pilih opsi yang sesuai.

   Jika nilainya adalah string yang diformat, pilih tipe “iso”, string harus sesuai dengan salah satu varian format ISO, misalnya: “2022-11-02T 14:40:59.915 Z”.

   Jika Anda tidak tahu jenisnya pada saat ini atau baris yang berbeda menggunakan tipe yang berbeda, maka Anda dapat memilih” deteksi otomatis “dan sistem akan membuat tebakan terbaiknya, dengan biaya kinerja yang kecil.
5. (Opsional) Pada tab Transform, alih-alih mengonversi kolom yang dipilih, Anda dapat membuat yang baru dan menyimpan yang asli dengan memasukkan nama untuk kolom baru.

Ubah kolom stempel waktu menjadi string yang diformat

Format kolom stempel waktu menjadi string berdasarkan pola. Anda dapat menggunakan stempel waktu Format untuk mendapatkan tanggal dan waktu sebagai string dengan format yang diinginkan. Anda dapat menentukan format menggunakan sintaks tanggal Spark serta sebagian besar kode tanggal Python.

Misalnya, jika Anda ingin string tanggal Anda diformat seperti “2023-01-01 00:00”, Anda dapat menentukan format tersebut menggunakan sintaks Spark sebagai “YYYY-MM-DD HH:mm” atau kode tanggal Python yang setara sebagai “%y-%m-%d %H: %M”

Untuk menambahkan simpul transformasi stempel waktu Format dalam diagram pekerjaan Anda

1. Buka panel Resource dan kemudian pilih Format timestamp untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pada tab Transform, masukkan nama kolom yang akan dikonversi.
4. Pada tab Transform, masukkan pola format Timestamp untuk digunakan, dinyatakan menggunakan sintaks tanggal Spark atau kode tanggal Python.
5. (Opsional) Pada tab Transform, alih-alih mengonversi kolom yang dipilih, Anda dapat membuat yang baru dan menyimpan yang asli dengan memasukkan nama untuk kolom baru.

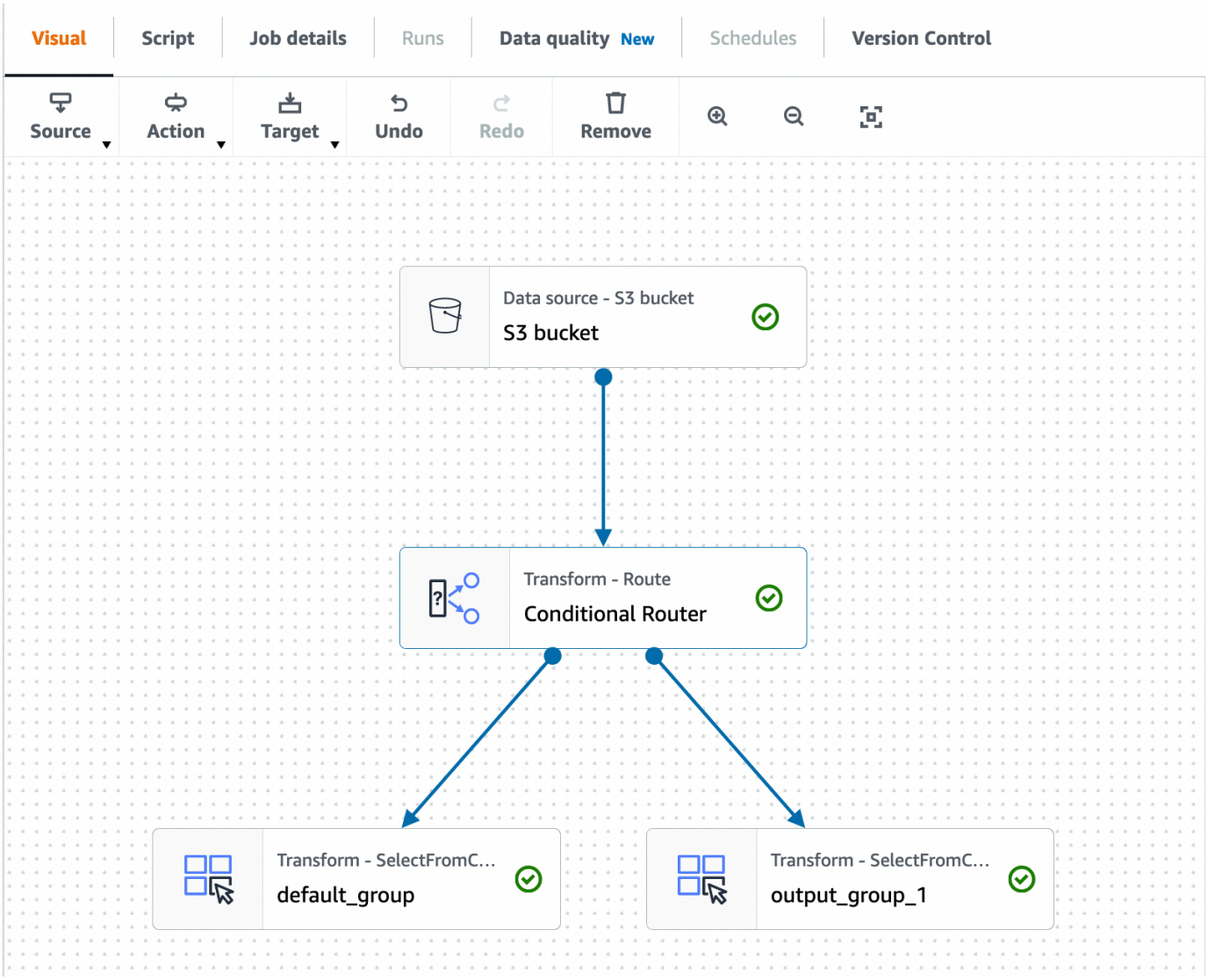
Membuat transformasi Router Bersyarat

Transformasi Router Bersyarat memungkinkan Anda menerapkan beberapa kondisi ke data yang masuk. Setiap baris data yang masuk dievaluasi oleh kondisi filter grup dan diproses menjadi grup yang sesuai. Jika baris memenuhi lebih dari satu kondisi filter grup, transformasi meneruskan baris ke beberapa grup. Jika baris tidak memenuhi kondisi apa pun, itu dapat dijatuhkan atau dirutekan ke grup keluaran default.

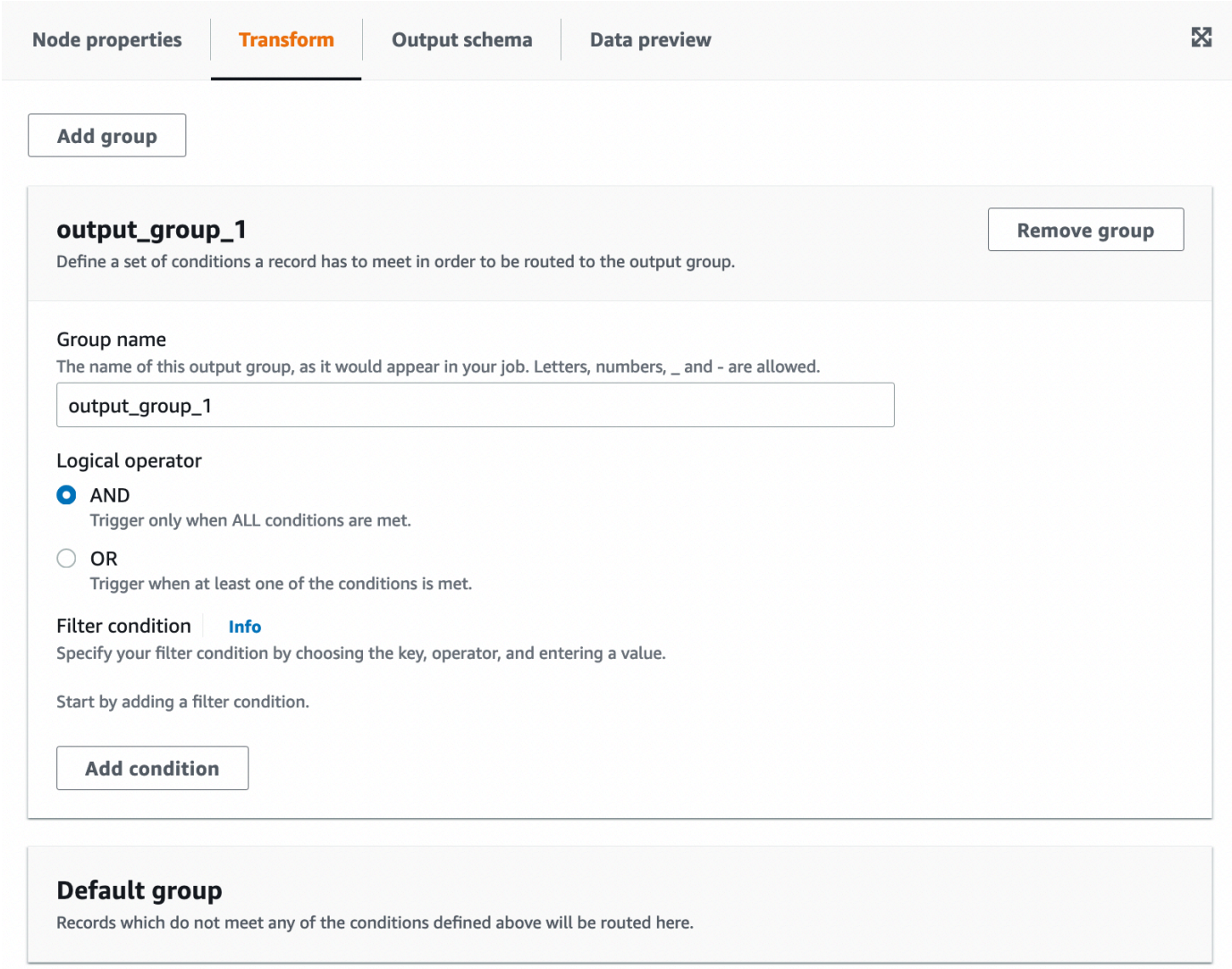
Transformasi ini mirip dengan transformasi filter, tetapi berguna bagi pengguna yang ingin menguji data input yang sama pada beberapa kondisi.

Untuk menambahkan transformasi Router Bersyarat:

1. Pilih node tempat Anda akan melakukan transformasi router bersyarat. Ini bisa berupa simpul sumber atau transformasi lain.
2. Pilih Tindakan, lalu gunakan bilah pencarian untuk menemukan dan memilih 'Conditional Router'. Transformasi Router Bersyarat ditambahkan bersama dengan dua node keluaran. Satu node keluaran, 'Grup default', berisi catatan yang tidak memenuhi salah satu kondisi yang ditentukan dalam node keluaran lainnya. Grup default tidak dapat diedit.

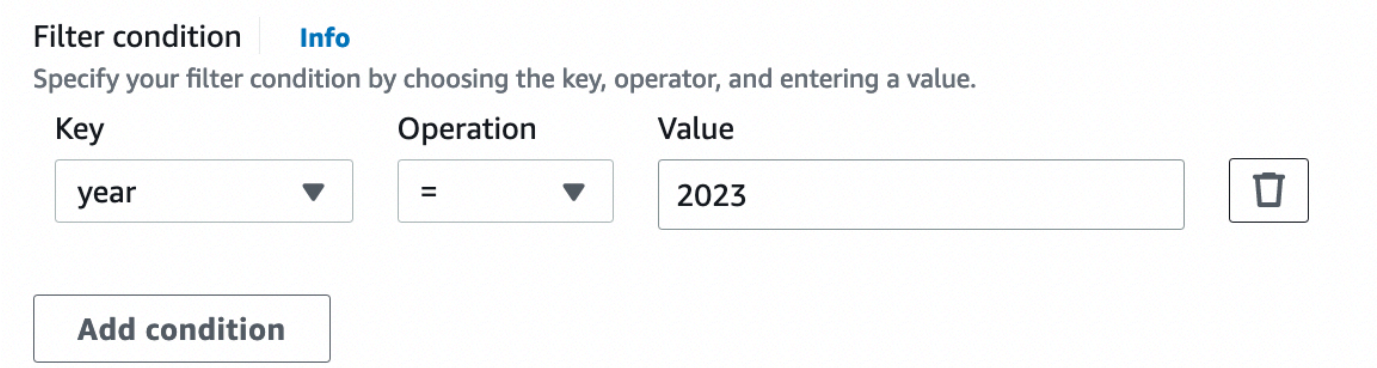


Anda dapat menambahkan grup keluaran tambahan dengan memilih Tambah grup. Untuk setiap grup keluaran, Anda dapat memberi nama grup dan menambahkan kondisi filter dan operator logis.



3. Ganti nama nama grup keluaran dengan memasukkan nama baru untuk grup. AWS Glue Studioakan secara otomatis memberi nama grup Anda untuk Anda (misalnya, 'output_group_1').
4. Pilih operator logis (AND, OR) dan tambahkan kondisi Filter dengan menentukan Kunci, Operasi, dan Nilai. Operator logis memungkinkan Anda untuk menerapkan lebih dari satu kondisi filter dan melakukan operator logis pada setiap kondisi filter yang Anda tentukan.

   Saat menentukan kunci, Anda dapat memilih dari kunci yang tersedia dalam skema Anda. Anda kemudian dapat memilih operasi yang tersedia tergantung pada jenis kunci yang Anda pilih. Misalnya, jika tipe kuncinya adalah 'string', maka operasi yang tersedia untuk dipilih adalah 'cocok'.



5. Masukkan nilai di bidang Nilai. Untuk menambahkan kondisi filter tambahan, pilih Tambahkan kondisi. Untuk menghapus kondisi filter, pilih ikon tempat sampah.

Menggunakan transformasi Kolom Concatenate untuk menambahkan kolom

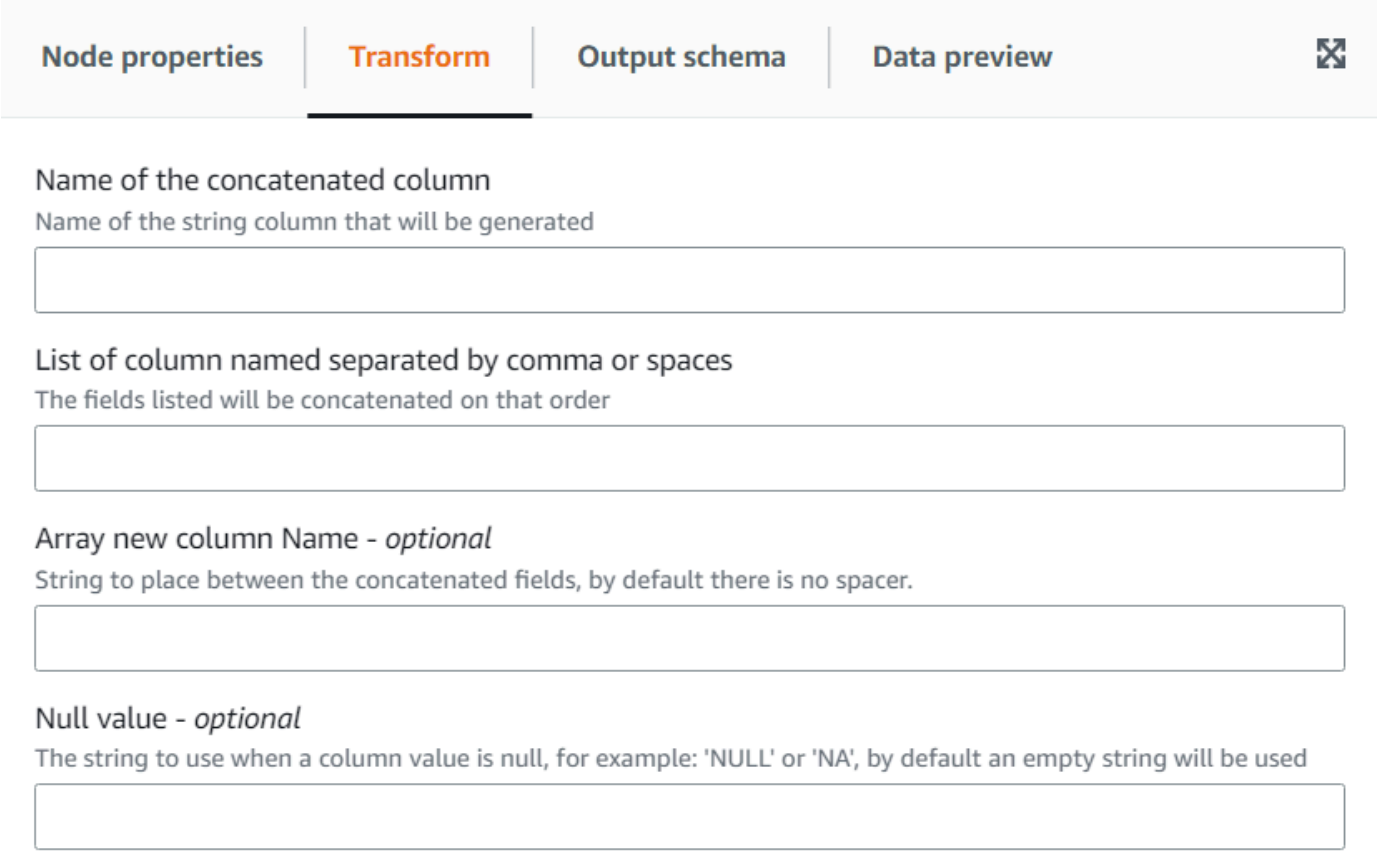
Transformasi Concatenate memungkinkan Anda membangun kolom string baru menggunakan nilai kolom lain dengan spacer opsional. Misalnya, jika kita mendefinisikan kolom gabungan “tanggal” sebagai rangkaian “tahun”, “bulan” dan “hari” (dalam urutan itu) dengan “-” sebagai spacer, kita akan mendapatkan:

hari	bulan	tahun	tanggal
01	01	2020	2020-01-01
02	01	2020	2020-01-02
03	01	2020	2020-01-03
04	01	2020	2020-01-04

Untuk menambahkan transformasi Concatenate:

1. Buka panel Sumber Daya. Kemudian pilih Concatenate Columns untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.

3. Pada tab Transform, masukkan nama kolom yang akan menahan string gabungan serta kolom untuk digabungkan. Urutan di mana Anda memeriksa kolom di dropdown akan menjadi urutan yang digunakan.



4. Spacer - opsional - Masukkan string untuk ditempatkan di antara bidang gabungan. Secara default, tidak ada spacer.
5. Nilai nol - opsional - Masukkan string untuk digunakan ketika nilai kolom adalah nol. Secara default, dalam kasus di mana kolom memiliki nilai 'NULL' atau 'NA', string kosong digunakan.

Menggunakan transformasi Split String untuk memecah kolom string

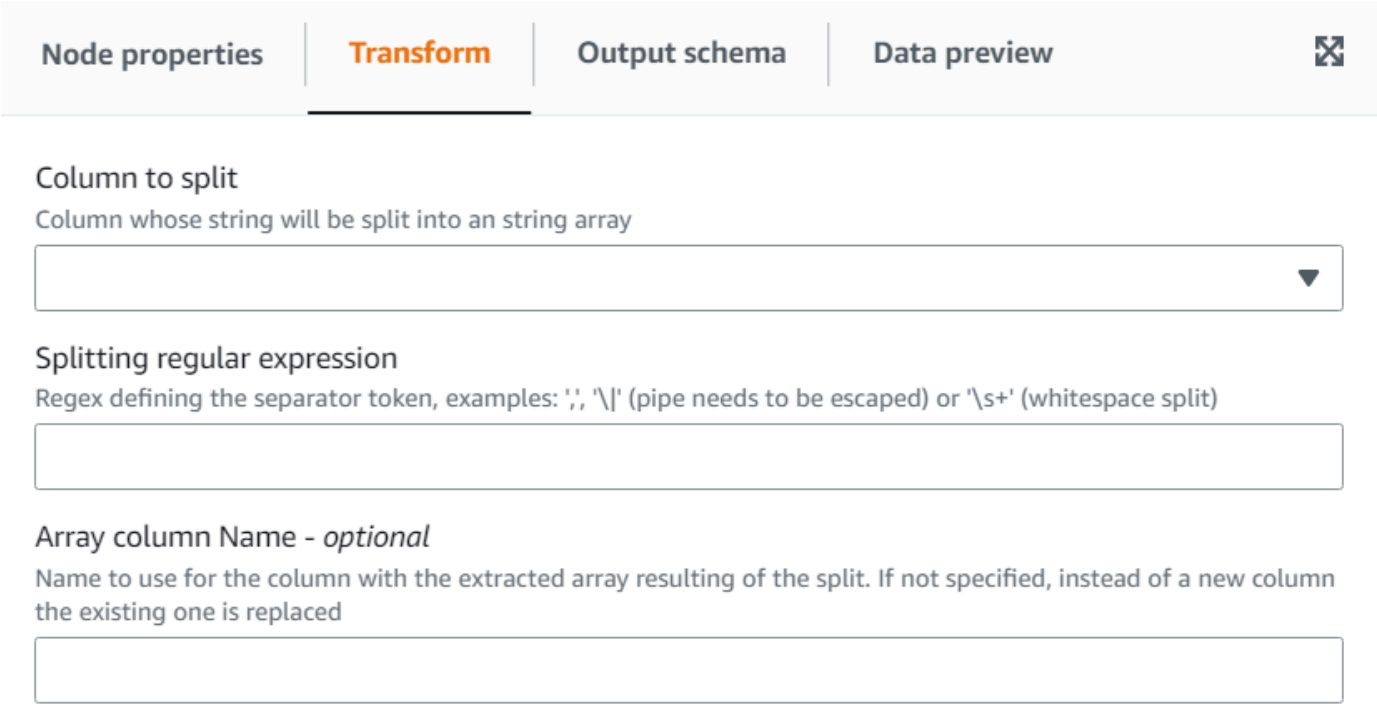
Transformasi Split String memungkinkan Anda memecah string menjadi array token menggunakan ekspresi reguler untuk menentukan bagaimana pemisahan dilakukan. Anda kemudian dapat menyimpan kolom sebagai tipe array atau menerapkan transformasi Array To Columns setelah yang ini, untuk mengekstrak nilai array ke bidang tingkat atas, dengan asumsi bahwa setiap token memiliki makna yang kita ketahui sebelumnya. Juga, jika urutan token tidak relevan (misalnya, satu set kategori), Anda dapat menggunakan transformasi Explode untuk menghasilkan baris terpisah untuk setiap nilai.

Misalnya, Anda dapat membagi kolom “kategori” menggunakan koma sebagai pola untuk menambahkan kolom “categories_arr”.

product_id	kategori	kategori_arr
1	olahraga, musim dingin	[olahraga, musim dingin]
2	kebun, alat-alat	[kebun, alat]
3	videogame	[videogame]
4	permainan, boardgame, sosial	[permainan, boardgame, sosial]

Untuk menambahkan transformasi Split String:

1. Buka panel Resource dan kemudian pilih Split String untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pada tab Transform, pilih kolom untuk dibagi dan masukkan pola yang akan digunakan untuk membagi string. Dalam kebanyakan kasus, Anda hanya dapat memasukkan karakter kecuali jika memiliki arti khusus sebagai ekspresi reguler dan perlu diloloskan. Karakter yang perlu melarikan diri adalah: `\.[]{}()<>*+-=!?^$|` dengan menambahkan garis miring terbalik di depan karakter. Misalnya jika Anda ingin memisahkan dengan titik ('.') Anda harus memasukkan`\.`. Namun, koma tidak memiliki arti khusus dan hanya dapat ditentukan apa adanya:`,`.



4. (Opsional) Jika Anda ingin menyimpan kolom string asli, maka Anda dapat memasukkan nama untuk kolom array baru, dengan cara ini menjaga kolom string asli dan kolom array tokenized baru.

Menggunakan transformasi Array To Columns untuk mengekstrak elemen array ke kolom tingkat atas

Transformasi Array To Columns memungkinkan Anda mengekstrak beberapa atau semua elemen kolom tipe array ke kolom baru. Transformasi akan mengisi kolom baru sebanyak mungkin jika array memiliki nilai yang cukup untuk diekstrak, secara opsional mengambil elemen dalam posisi yang ditentukan.

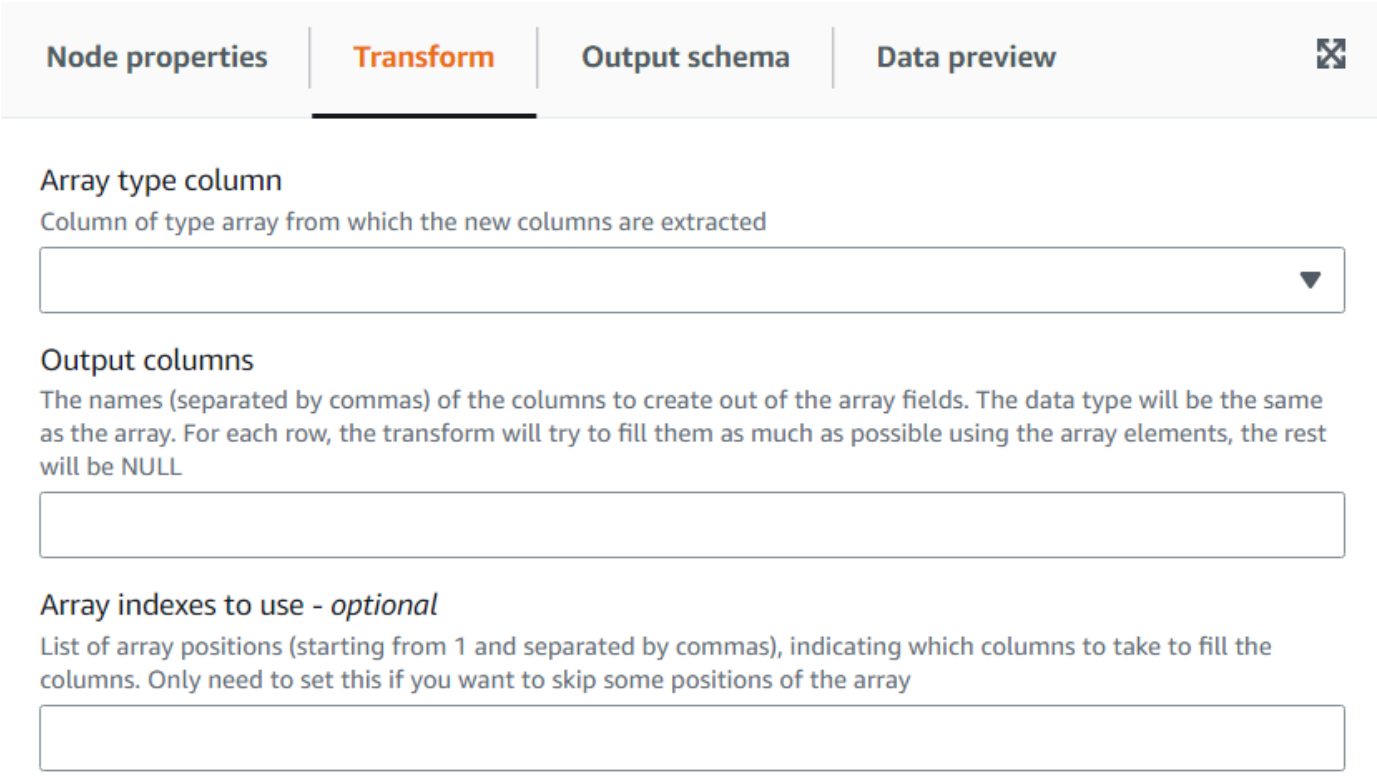
Misalnya, jika Anda memiliki kolom array “subnet”, yang merupakan hasil dari penerapan transformasi “Split String” pada subnet ip v4, Anda dapat mengekstrak posisi pertama dan seterusnya ke kolom baru “first_octect” dan “forth_octect”. Output dari transformasi dalam contoh ini adalah (perhatikan dua baris terakhir memiliki array yang lebih pendek dari yang diharapkan):

subnet	first_octect	keempat_oktek
[54, 240, 197, 238]	54	238

subnet	first_octect	keempat_oktek
[192, 168, 0, 1]	192	1
[192, 168]	192	
[]		

Untuk menambahkan transformasi Array Ke Kolom:

1. Buka panel Resource dan kemudian pilih Array To Columns untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pada tab Transform, pilih kolom array untuk mengekstrak dan masukkan daftar kolom baru untuk token yang diekstrak.



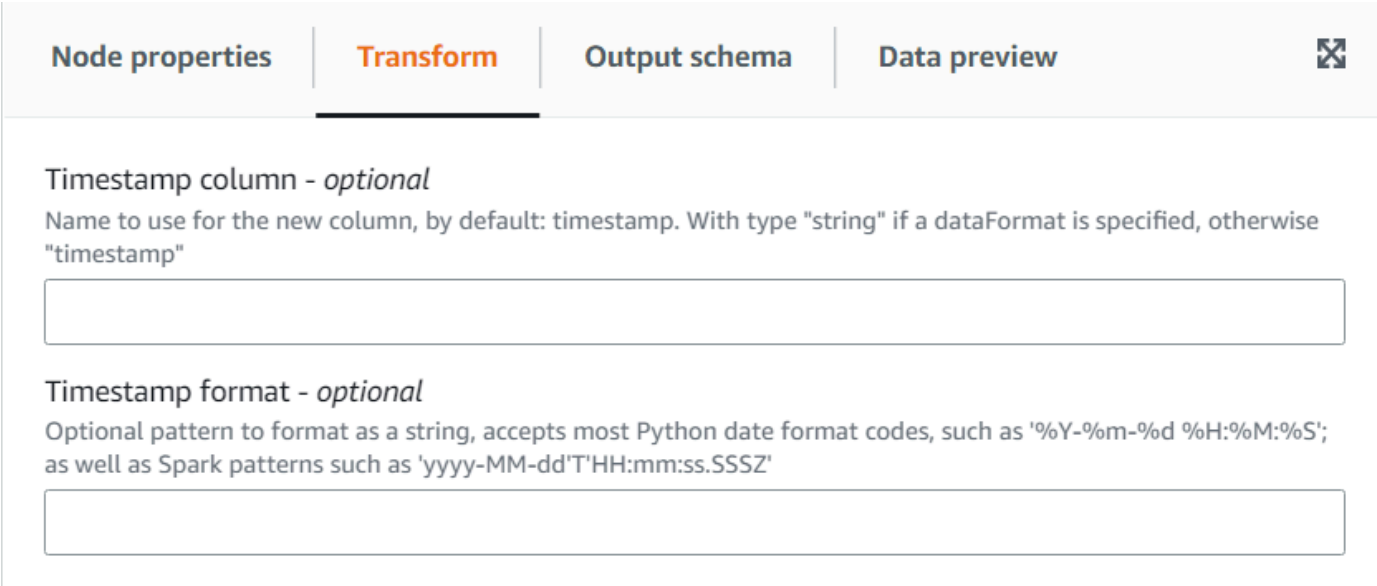
4. (Opsional) Jika Anda tidak ingin mengambil token array untuk menetapkan ke kolom, Anda dapat menentukan indeks yang akan diambil yang akan ditetapkan ke daftar kolom dalam urutan yang sama ditentukan. Misalnya jika kolom output adalah “column1, column2, column3" dan indeks “4, 1, 3", elemen keempat dari array akan pergi ke column1, yang pertama ke column2 dan yang ketiga ke column3 (jika array lebih pendek dari nomor indeks, nilai NULL akan ditetapkan).

Menggunakan transformasi Add Current Timestamp

Transformasi Add Current Timestamp memungkinkan Anda menandai baris dengan waktu pemrosesan data. Ini berguna untuk tujuan audit atau untuk melacak latensi dalam pipa data. Anda dapat menambahkan kolom baru ini sebagai tipe data stempel waktu atau string yang diformat.

Untuk menambahkan transformasi Add Current Timestamp:

1. Buka panel Resource dan kemudian pilih Add Current Timestamp untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.



3. (Opsional) Pada Transform tab, masukkan nama khusus untuk kolom baru dan format jika Anda lebih suka kolom menjadi string tanggal yang diformat.

Menggunakan Pivot Rows to Columns transform

Transformasi Pivot Rows to Columns memungkinkan Anda untuk menggabungkan kolom numerik dengan memutar nilai unik pada kolom terpilih yang menjadi kolom baru (jika beberapa kolom dipilih, nilainya digabungkan untuk memberi nama kolom baru). Dengan begitu baris dikonsolidasikan sementara memiliki lebih banyak kolom dengan agregasi parsi untuk setiap nilai unik. Misalnya, jika Anda memiliki kumpulan data penjualan ini berdasarkan bulan dan negara (diurutkan agar lebih mudah diilustrasikan):

tahun	bulan	negeri	jumlah
2020	Jan	inggris	32
2020	Jan	de	42
2020	Jan	µs	64
2020	Februari	inggris	67
2020	Februari	de	4
2020	Februari	de	7
2020	Februari	µs	6
2020	Februari	µs	12
2020	Jan	µs	90

Jika jumlah pivot dan negara sebagai kolom agregasi, kolom baru akan dibuat dari kolom negara asli. Pada tabel di bawah ini, Anda memiliki kolom baru untuk de, uk, dan kami, bukan kolom negara.

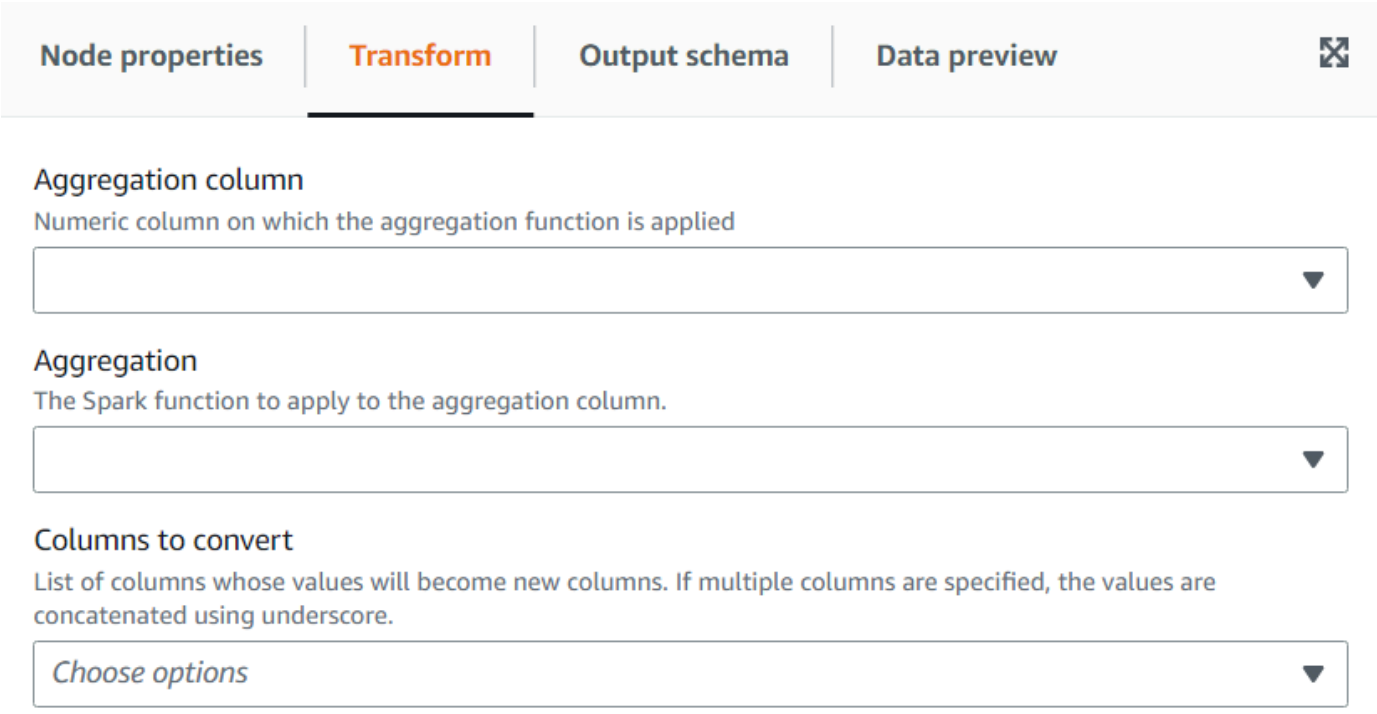
tahun	bulan	de	inggris	µs
2020	Jan	42	32	64
2020	Jan	11	67	18
2021	Jan			90

Jika sebaliknya Anda ingin memutar bulan dan kabupaten, Anda mendapatkan kolom untuk setiap kombinasi nilai kolom tersebut:

tahun	Jan_de	Jan_UK	Jan_us	Feb_de	Feb_Inggris	Feb_US
2020	42	32	64	11	67	18
2021			90			

Untuk menambahkan transformasi Pivot Rows To Columns:

1. Buka panel Resource dan kemudian pilih Pivot Rows To Columns untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pada tab Transform, pilih kolom numerik yang akan digabungkan untuk menghasilkan nilai untuk kolom baru, fungsi agregasi untuk diterapkan dan kolom untuk mengubah nilai uniknya menjadi kolom baru.



Menggunakan transformasi Kolom Unpivot Ke Baris

Transformasi Unpivot memungkinkan Anda mengonversi kolom menjadi nilai kolom baru yang menghasilkan baris untuk setiap nilai unik. Ini kebalikan dari pivot tetapi perhatikan bahwa itu tidak setara karena tidak dapat memisahkan baris dengan nilai identik yang digabungkan atau membagi kombinasi ke dalam kolom asli (Anda dapat melakukannya nanti menggunakan transformasi Split). Misalnya, jika Anda memiliki tabel berikut:

tahun	bulan	de	inggris	us
2020	Jan	42	32	64
2020	Februari	11	67	18
2021	Jan			90

Anda dapat membuka pivot kolom: “de”, “uk” dan “us” ke dalam kolom “negara” dengan nilai “jumlah”, dan dapatkan yang berikut (diurutkan di sini untuk tujuan ilustrasi):

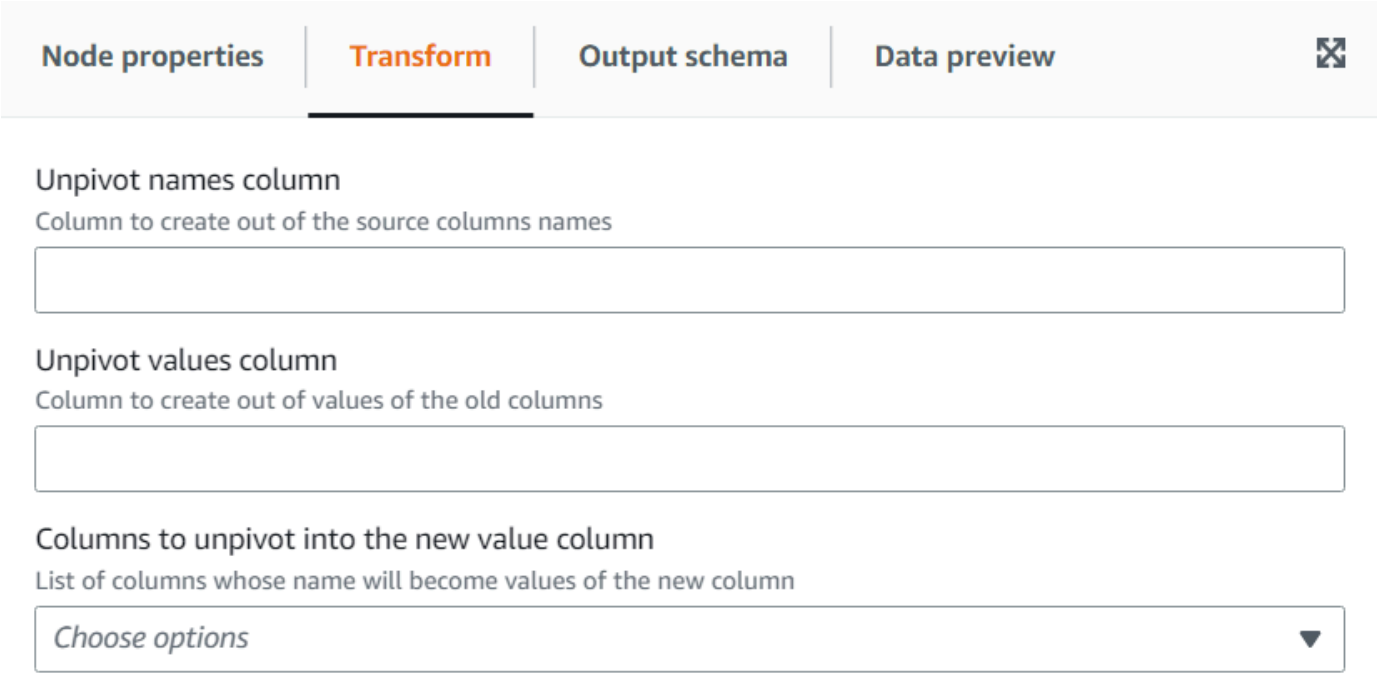
tahun	bulan	negeri	jumlah
2020	Jan	inggris	32
2020	Jan	de	42
2020	Jan	us	64
2020	Februari	inggris	67
2020	Februari	de	11
2020	Februari	us	18
2021	Jan	us	90

Perhatikan kolom yang memiliki nilai NULL (“de” dan “uk of Jan 2021) tidak dihasilkan secara default. Anda dapat mengaktifkan opsi itu untuk mendapatkan:

tahun	bulan	negeri	jumlah
2020	Jan	inggris	32
2020	Jan	de	42
2020	Jan	us	64
2020	Februari	inggris	67
2020	Februari	de	11
2020	Februari	us	18
2021	Jan	us	90
2021	Jan	de	
2021	Jan	inggris	

Untuk menambahkan Kolom Unpivot ke Rows transform:

1. Buka panel Resource dan kemudian pilih Unpivot Columns to Rows untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pada tab Transform, masukkan kolom baru yang akan dibuat untuk menyimpan nama dan nilai kolom yang dipilih untuk di-unpivot.

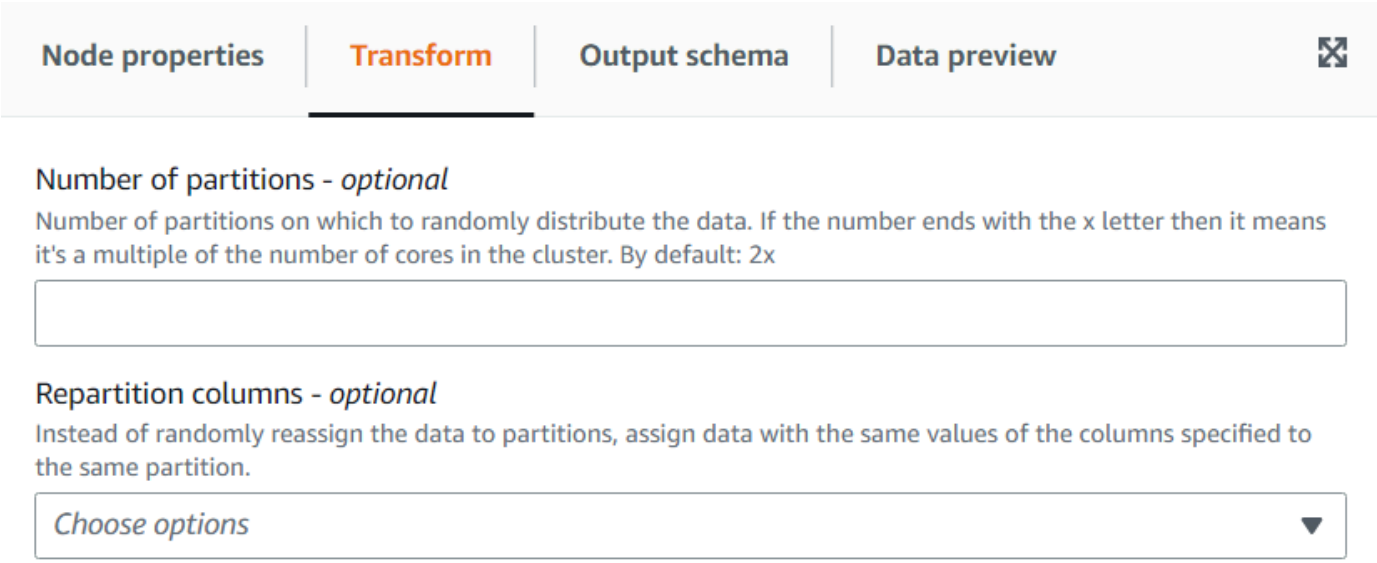


Menggunakan transformasi Autobalance Processing untuk mengoptimalkan runtime Anda

Transformasi Autobalance Processing mendistribusikan kembali data di antara para pekerja untuk kinerja yang lebih baik. Ini membantu dalam kasus di mana data tidak seimbang atau karena berasal dari sumber tidak memungkinkan pemrosesan paralel yang cukup di atasnya. Ini umum terjadi di mana sumbernya di-gzip atau JDBC. Redistribusi data memiliki biaya kinerja yang sederhana, sehingga optimasi mungkin tidak selalu mengkompensasi upaya itu jika data sudah seimbang. Di bawahnya, transformasi menggunakan repartisi Apache Spark untuk menetapkan kembali data secara acak di antara sejumlah partisi yang optimal untuk kapasitas cluster. Untuk pengguna tingkat lanjut, dimungkinkan untuk memasukkan sejumlah partisi secara manual. Selain itu, dapat digunakan untuk mengoptimalkan penulisan tabel yang dipartisi dengan mengatur ulang data berdasarkan kolom yang ditentukan. Ini menghasilkan file output yang lebih terkonsolidasi.

1. Buka panel Resource dan kemudian pilih Autobalance Processing untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.

3. (Opsional) Pada tab Transform, Anda dapat memasukkan sejumlah partisi. Secara umum, disarankan agar Anda membiarkan sistem memutuskan nilai ini, namun Anda dapat menyetel pengganda atau memasukkan nilai tertentu jika Anda perlu mengontrolnya. Jika Anda akan menyimpan data yang dipartisi oleh kolom, Anda dapat memilih kolom yang sama dengan kolom partisi ulang. Dengan cara ini akan meminimalkan jumlah file pada setiap partisi dan menghindari memiliki banyak file per partisi, yang akan menghambat kinerja alat yang menanyakan data tersebut.



Menggunakan transformasi Kolom Derived untuk menggabungkan kolom lain

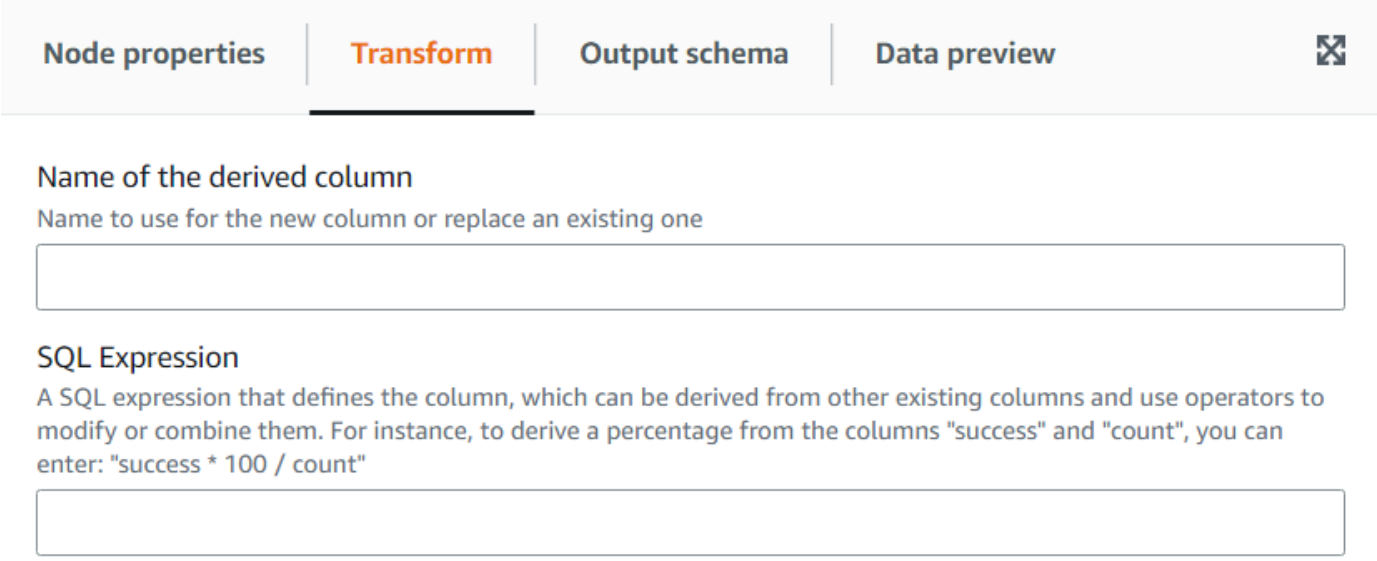
Transformasi Kolom Derived memungkinkan Anda untuk menentukan kolom baru berdasarkan rumus matematika atau ekspresi SQL di mana Anda dapat menggunakan kolom lain dalam data, serta konstanta dan literal. Misalnya, untuk mendapatkan kolom “persentase” dari kolom “sukses” dan “hitung”, Anda dapat memasukkan ekspresi SQL: “sukses * 100/count || '%'”.

Contoh hasil:

keberhasilan	count	persentase
14	100	14%
6	20	3%
3	40	7,5%

Untuk menambahkan transformasi Kolom Derived:

1. Buka panel Resource dan kemudian pilih Derived Column untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pada tab Transform, masukkan nama kolom dan ekspresi untuk kontennya.



Menggunakan transformasi Lookup untuk menambahkan data yang cocok dari tabel katalog

Transformasi Pencarian memungkinkan Anda menambahkan kolom dari tabel katalog yang ditentukan saat kunci cocok dengan kolom pencarian yang ditentukan dalam data. Ini setara dengan melakukan gabungan luar kiri antara data dan tabel pencarian menggunakan kolom pencocokan kondisi.

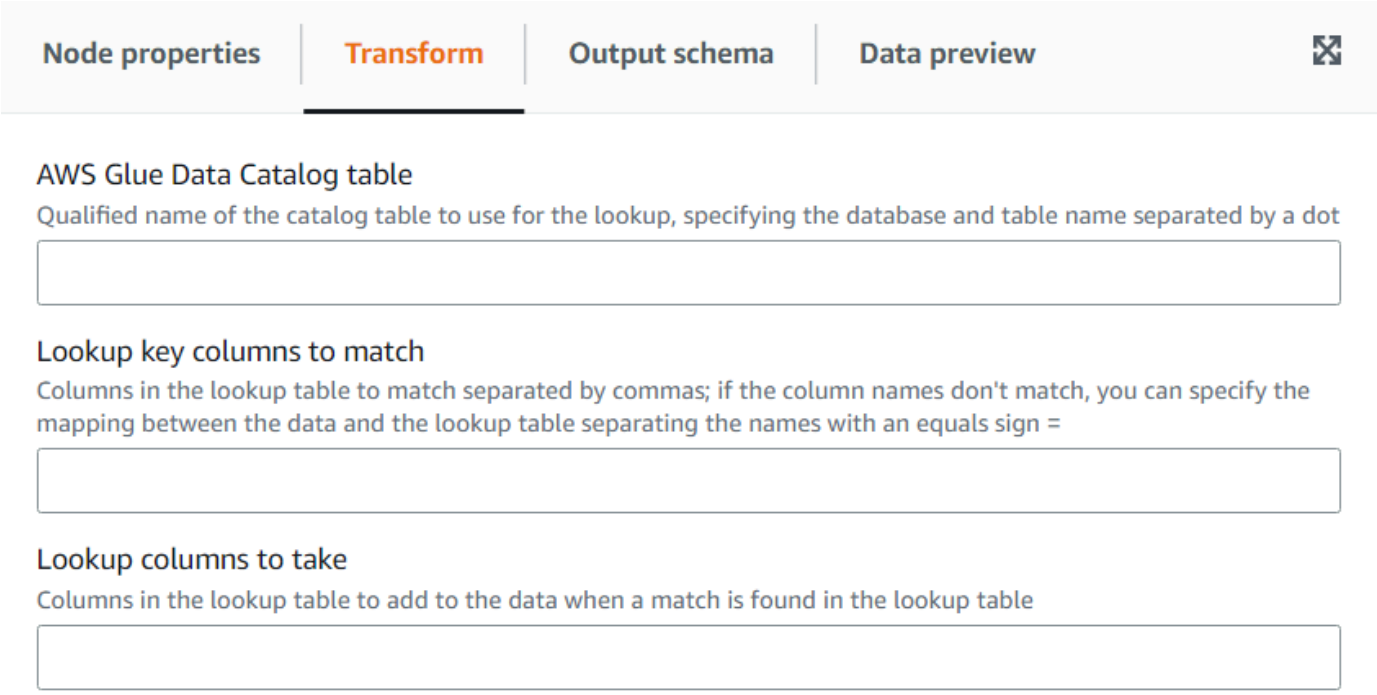
Untuk menambahkan transformasi Pencarian:

1. Buka panel Resource dan kemudian pilih Lookup untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.

2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pada tab Transform, masukkan nama tabel katalog yang memenuhi syarat untuk digunakan untuk melakukan pencarian. Misalnya, jika database Anda adalah “mydb” dan tabel Anda “mytable” kemudian masukkan “mydb.mytable”. Kemudian masukkan kriteria untuk menemukan kecocokan di tabel pencarian, jika kunci pencarian disusun. Masukkan daftar kolom kunci yang dipisahkan dengan koma. Jika satu atau beberapa kolom kunci tidak memiliki nama yang sama maka Anda perlu menentukan pemetaan kecocokan.

   Misalnya, jika kolom data adalah “user_id” dan “region” dan di tabel pengguna kolom yang sesuai diberi nama “id” dan “region “, lalu di kolom Kolom yang cocok, masukkan:” user_id = id, region”. Anda dapat melakukan region=region tetapi tidak diperlukan karena mereka sama.
4. Terakhir, masukkan kolom untuk dibawa dari baris yang cocok di tabel pencarian untuk memasukkannya ke dalam data. Jika tidak ada kecocokan yang ditemukan, kolom tersebut akan disetel ke NULL.

> **Note**
>
> Di bawah transformasi Pencarian, ia menggunakan gabungan kiri agar efisien. Jika tabel pencarian memiliki kunci komposit, pastikan kolom yang cocok diatur agar sesuai dengan semua kolom kunci sehingga hanya satu kecocokan yang dapat terjadi. Jika tidak, beberapa baris pencarian akan cocok dan ini akan menghasilkan baris tambahan yang ditambahkan untuk masing-masing kecocokan tersebut.



Menggunakan transformasi Explode Array atau Map Into Rows

Transformasi Explode memungkinkan Anda mengekstrak nilai dari struktur bersarang menjadi baris individual yang lebih mudah dimanipulasi. Dalam kasus array, transformasi akan menghasilkan baris untuk setiap nilai array, mereplikasi nilai untuk kolom lain di baris. Dalam kasus peta, transformasi akan menghasilkan baris untuk setiap entri dengan kunci dan nilai sebagai kolom ditambah kolom lain di baris.

Misalnya, jika kita memiliki kumpulan data ini yang memiliki kolom array “kategori” dengan beberapa nilai.

product_id	kategori
1	[olahraga, musim dingin]
2	[kebun, alat]
3	[videogame]
4	[permainan, boardgame, sosial]

product_id	kategori
5	[]

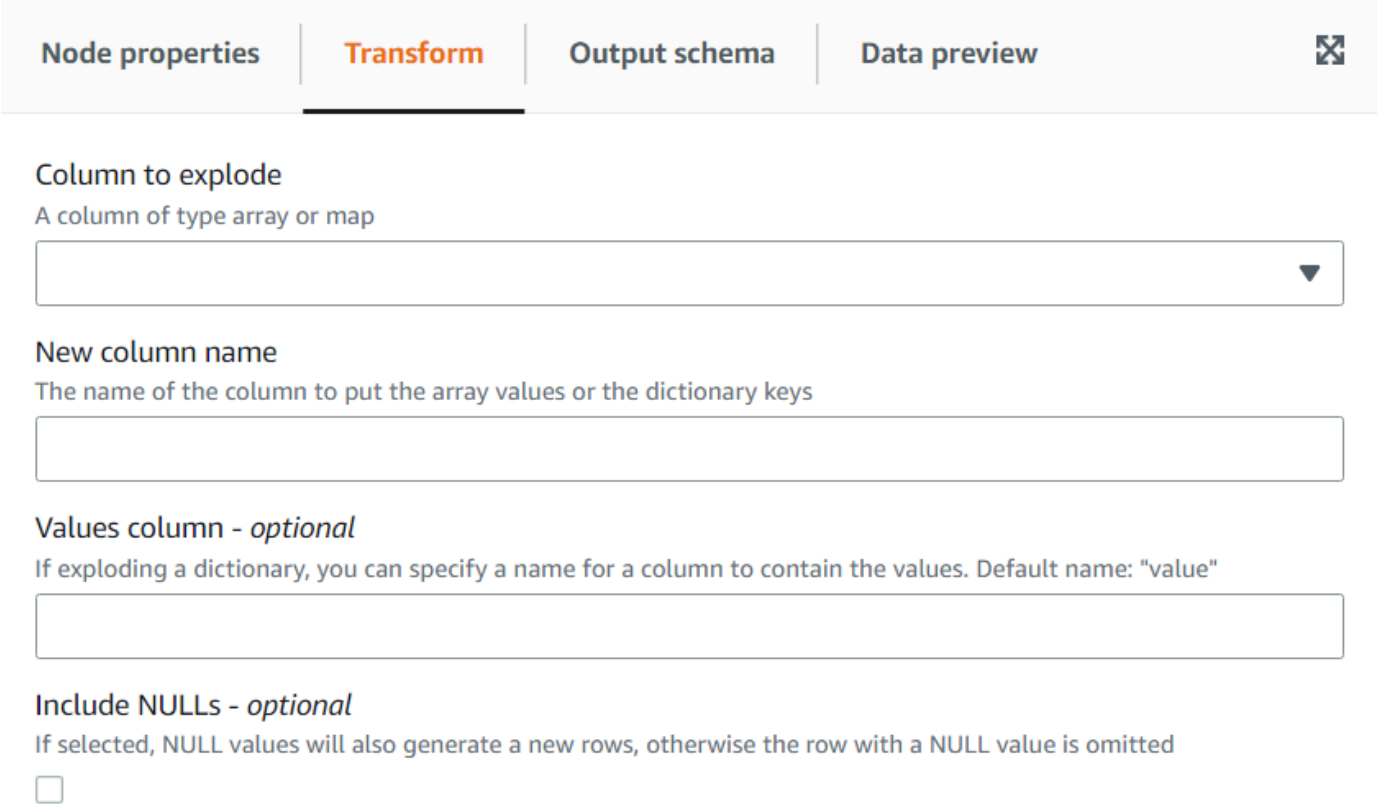
Jika Anda meledakkan kolom 'kategori' ke dalam kolom dengan nama yang sama, Anda akan mengganti kolom tersebut. Anda dapat memilih bahwa Anda ingin NULL disertakan untuk mendapatkan yang berikut (dipesan untuk tujuan ilustrasi):

product_id	kategori
1	olahraga
1	musim dingin
2	taman
2	alat
3	videogame
4	gim
4	permainan papan
4	sosial
5	

Untuk menambahkan transformasi Explode Array Atau Map Into Rows:

1. Buka panel Resource dan kemudian pilih Explode Array Or Map Into Rows untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. (Opsional) Pada tab properti Node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.

3. Pada tab Transform, pilih kolom yang akan meledak (harus berupa array atau tipe peta). Kemudian masukkan nama untuk kolom untuk item array atau nama kolom untuk kunci dan nilai jika Anda meledakkan peta.
4. (Opsional) Pada Transform tab, secara default jika kolom yang akan meledak adalah NULL atau memiliki struktur kosong, itu akan dihilangkan pada kumpulan data yang meledak. Jika Anda ingin menyimpan baris (dengan kolom baru sebagai NULL) maka centang “Sertakan NULLs”.

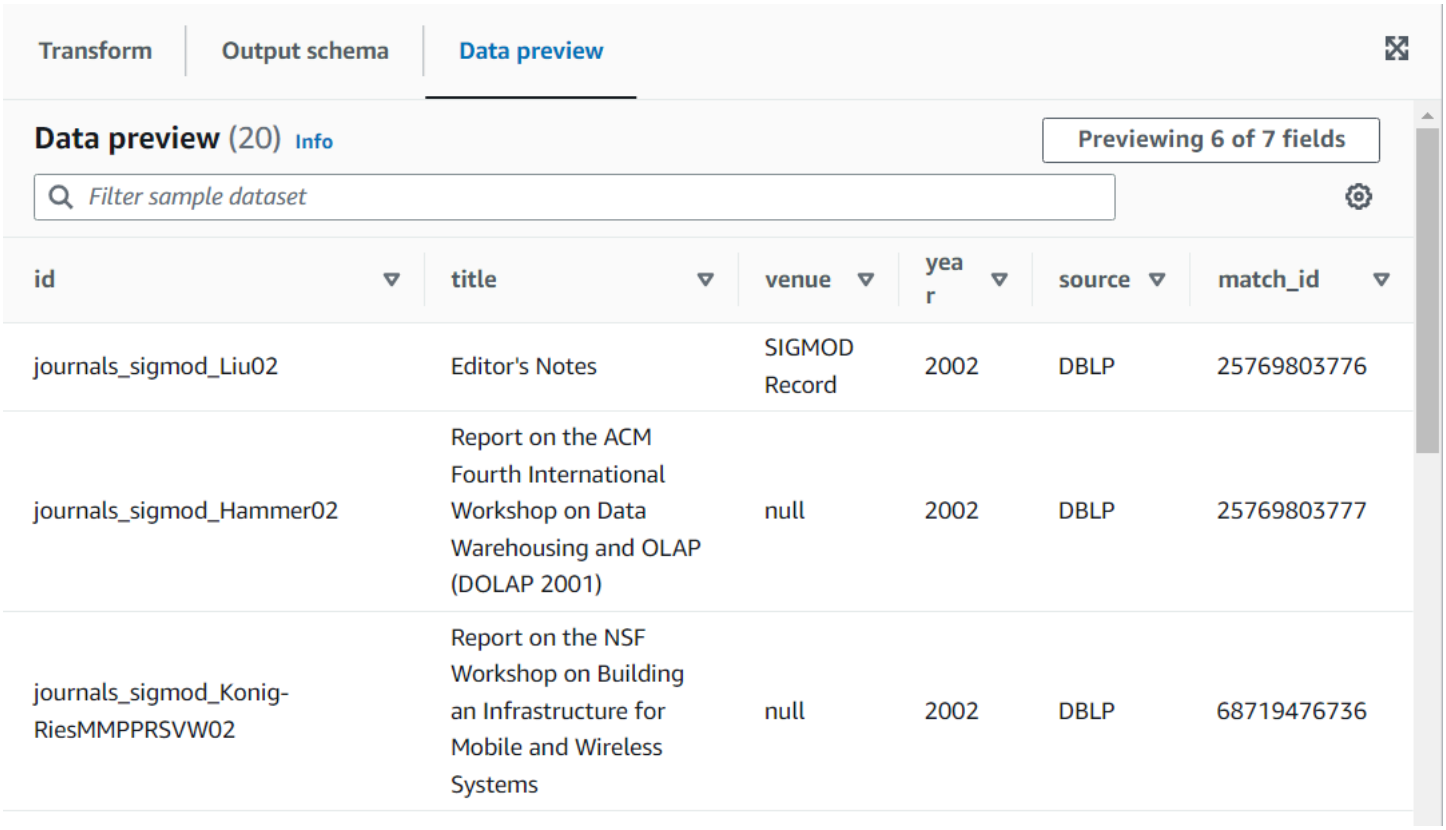


Menggunakan transformasi Record Matching untuk menjalankan transformasi klasifikasi data yang ada

Transformasi ini memanggil transformasi klasifikasi data pembelajaran mesin Record Matching yang ada.

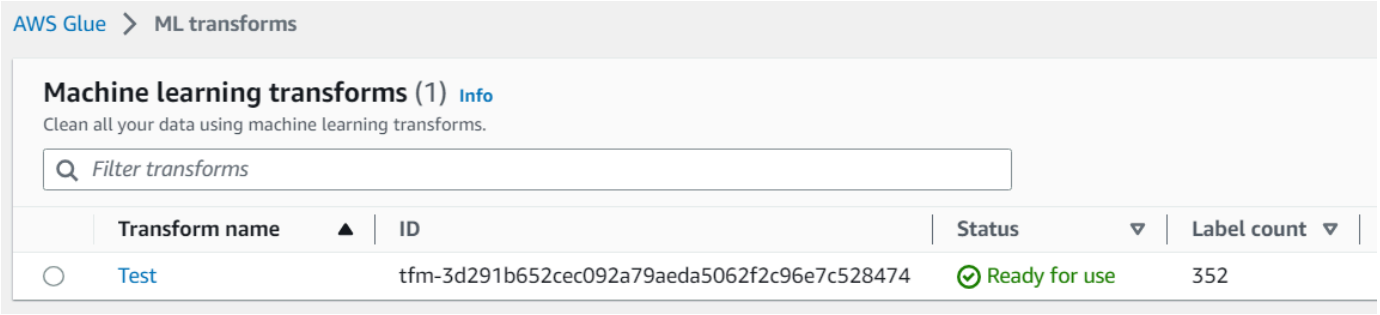
Transformasi mengevaluasi data saat ini terhadap model terlatih berdasarkan label. Kolom “match_id” ditambahkan untuk menetapkan setiap baris ke sekelompok item yang dianggap setara berdasarkan pelatihan algoritma. Untuk informasi lebih lanjut, lihat Rekam pencocokan dengan Lake Formation FindMatches.

> **Note**
>
> Versi yang AWS Glue digunakan oleh pekerjaan visual harus cocok dengan versi yang AWS Glue digunakan untuk membuat transformasi Record Matching.



Untuk menambahkan node transformasi Record Matching ke diagram pekerjaan Anda

1. Buka panel Resource, lalu pilih Record Matching untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. Di panel properti node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pada tab Transform, masukkan ID yang diambil dari halaman Transformasi pembelajaran mesin:



4. (Opsional) Pada Transform tab, Anda dapat mencentang opsi untuk menambahkan skor kepercayaan. Dengan biaya komputasi ekstra, model akan memperkirakan skor kepercayaan untuk setiap pertandingan sebagai kolom tambahan.

Menghapus baris nol

Transformasi ini menghapus dari baris dataset yang memiliki semua kolom sebagai null. Selain itu, Anda dapat memperluas kriteria ini untuk menyertakan bidang kosong, sehingga dapat menjaga baris di mana setidaknya satu kolom tidak kosong.

Untuk menambahkan Remove Null Rows, ubah node ke diagram pekerjaan Anda

1. Buka panel Resource, lalu pilih Remove Null Rows untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. Di panel properti node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. (Opsional) Pada tab Transform, centang opsi Extended jika Anda ingin meminta baris tidak hanya untuk tidak menjadi nol tetapi juga tidak kosong, dengan cara ini string kosong, array atau peta akan dianggap nol untuk tujuan transformasi ini.

Mengurai kolom string yang berisi data JSON

Transformasi ini mengurai kolom string yang berisi data JSON dan mengubahnya menjadi struct atau kolom array, tergantung apakah JSON adalah objek atau array, masing-masing. Secara opsional Anda dapat menyimpan kolom yang diurai dan asli.

Skema JSON dapat disediakan atau disimpulkan (dalam kasus objek JSON), dengan pengambilan sampel opsional.

Untuk menambahkan node transformasi Kolom JSON Parse ke diagram pekerjaan Anda

1. Buka panel Resource, lalu pilih Parse JSON Column untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. Di panel properti node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pada tab Transform, pilih kolom yang berisi string JSON.
4. <STRING>(Opsional) Pada tab Transform, masukkan skema yang diikuti data JSON menggunakan sintaks SQL, misalnya: “field1 STRING, field2 INT” dalam kasus objek atau “ARRAY” dalam kasus array.

   Jika kasus array skema diperlukan tetapi dalam kasus objek, jika skema tidak ditentukan itu akan disimpulkan menggunakan data. Untuk mengurangi dampak menyimpulkan skema (terutama pada kumpulan data besar), Anda dapat menghindari membaca seluruh data dua kali dengan memasukkan Rasio sampel yang akan digunakan untuk menyimpulkan skema. Jika nilainya lebih rendah dari 1, rasio sampel acak yang sesuai digunakan untuk menyimpulkan skema. Jika data dapat diandalkan dan objek konsisten di antara baris, Anda dapat menggunakan rasio kecil seperti 0,1 untuk meningkatkan kinerja.
5. (Opsional) Pada Transform tab, Anda dapat memasukkan nama kolom baru jika Anda ingin menyimpan kolom string asli dan kolom yang diuraikan.

Mengekstrak jalur JSON

Transformasi ini mengekstrak kolom baru dari kolom string JSON. Transformasi ini berguna ketika Anda hanya membutuhkan beberapa elemen data dan tidak ingin mengimpor seluruh konten JSON ke dalam skema tabel.

Untuk menambahkan node transformasi Extract JSON Path ke diagram pekerjaan Anda

1. Buka panel Resource, lalu pilih Extract JSON Path untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. Di panel properti node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.

3. Pada tab Transform, pilih kolom yang berisi string JSON. Masukkan salah satu ekspresi jalur JSON yang dipisahkan dengan koma, masing-masing merujuk cara mengekstrak nilai dari array atau objek JSON. Misalnya, jika kolom JSON berisi objek dengan properti “prop_1" dan “prop2" Anda dapat mengekstrak keduanya dengan menentukan nama mereka “prop_1, prop_2".

   Jika bidang JSON memiliki karakter khusus, misalnya untuk mengekstrak properti dari JSON ini `{"a. a": 1}` Anda dapat menggunakan jalur. `$['a. a']` Pengecualian adalah koma karena dicadangkan untuk jalur terpisah. Kemudian masukkan nama kolom yang sesuai untuk setiap jalur, dipisahkan dengan koma.

4. (Opsional) Pada Transform tab, Anda dapat memeriksa untuk menjatuhkan kolom JSON setelah diekstraksi, ini masuk akal ketika Anda tidak memerlukan sisa data JSON setelah Anda mengekstrak bagian yang Anda butuhkan.

Mengekstrak fragmen string menggunakan ekspresi reguler

Transformasi ini mengekstrak fragmen string menggunakan ekspresi reguler dan membuat kolom baru darinya, atau beberapa kolom jika menggunakan grup regex.

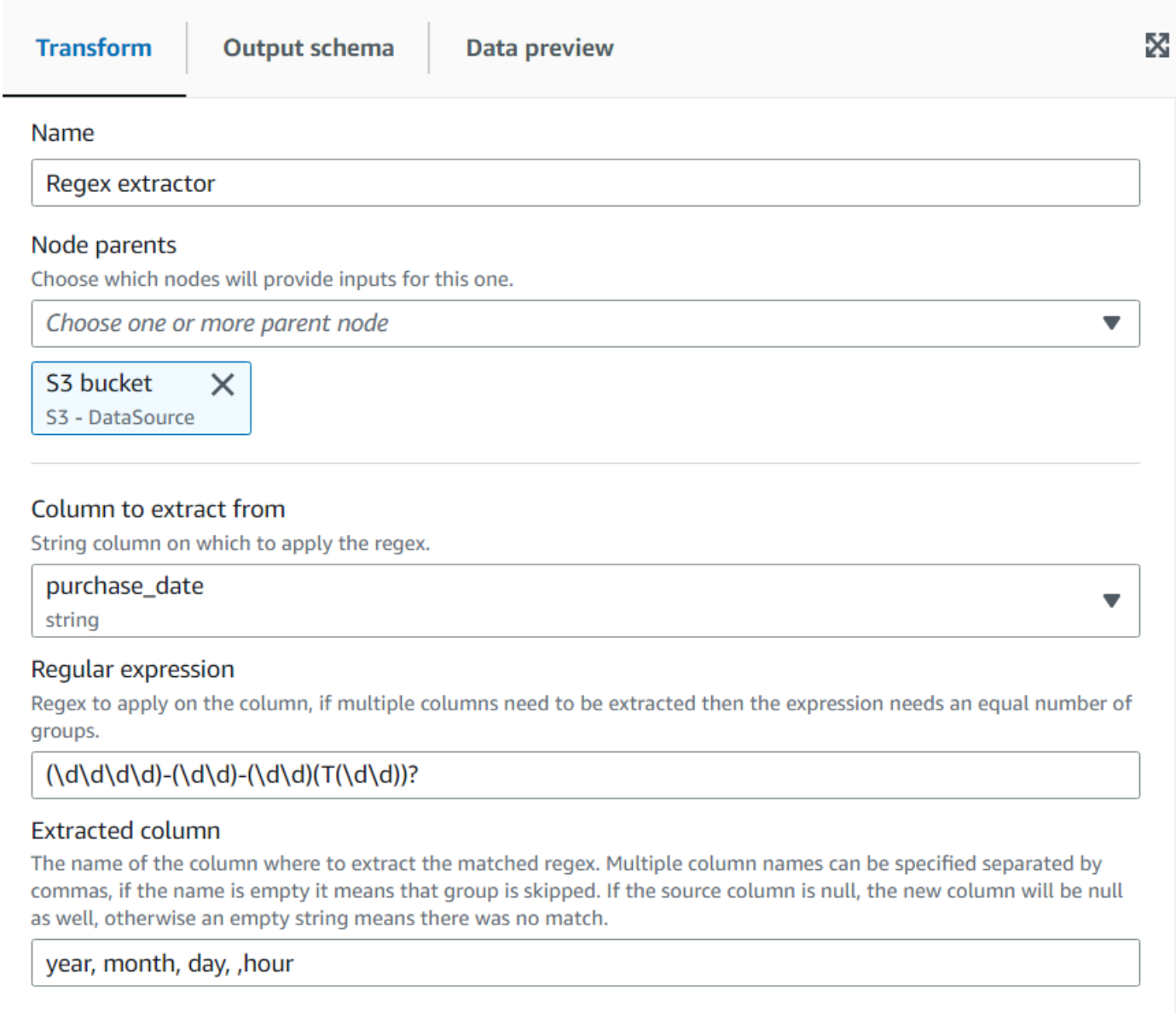
Untuk menambahkan node transformasi Regex Extractor ke diagram pekerjaan Anda

1. Buka panel Resource, lalu pilih Regex Extractor untuk menambahkan transformasi baru ke diagram pekerjaan Anda. Node yang dipilih pada saat menambahkan node akan menjadi induknya.
2. Di panel properti node, Anda dapat memasukkan nama untuk node dalam diagram pekerjaan. Jika sebuah induk simpul belum dipilih, maka pilihlah sebuah simpul dari daftar Induk simpul untuk digunakan sebagai sumber masukan untuk transformasi tersebut.
3. Pada tab Transform, masukkan ekspresi reguler dan kolom yang perlu diterapkan. Kemudian masukkan nama kolom baru untuk menyimpan string yang cocok. Kolom baru akan menjadi null hanya jika kolom sumber adalah null, jika regex tidak cocok kolom akan kosong.

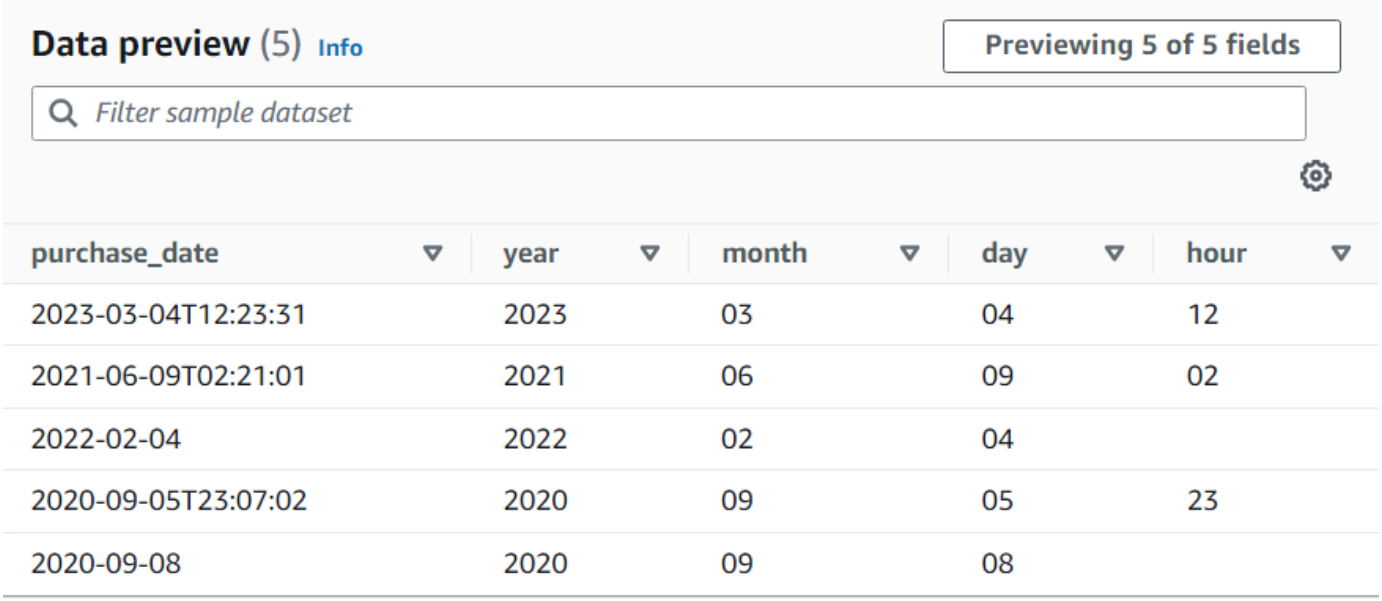
   Jika regex menggunakan grup, ada nama kolom yang sesuai dipisahkan dengan koma tetapi Anda dapat melewati grup dengan membiarkan nama kolom kosong.

   Misalnya, jika Anda memiliki kolom “purchase_date” dengan string menggunakan format tanggal ISO panjang dan pendek, maka Anda ingin mengekstrak tahun, bulan, hari dan jam, bila tersedia. Perhatikan grup jam adalah opsional, jika tidak di baris yang tidak tersedia, semua grup yang diekstraksi akan menjadi string kosong (karena regex tidak cocok). Dalam hal ini, kami

tidak ingin grup membuat waktu opsional tetapi yang dalam, jadi kami membiarkan nama kosong dan tidak diekstraksi (grup itu akan menyertakan karakter T).



Menghasilkan pratinjau data:



purchase_date	year	month	day	hour
2023-03-04T12:23:31	2023	03	04	12
2021-06-09T02:21:01	2021	06	09	02
2022-02-04	2022	02	04	
2020-09-05T23:07:02	2020	09	05	23
2020-09-08	2020	09	08	

Membuat sebuah transformasi kustom

Jika Anda perlu melakukan transformasi yang lebih rumit pada data Anda, atau ingin menambahkan kunci properti data ke set data, maka Anda dapat menambahkan transformasi Kode kustom ke diagram tugas Anda. Simpul kode kustom memungkinkan Anda untuk memasukkan sebuah skrip yang melakukan transformasi.

Saat menggunakan kode kustom, Anda harus menggunakan editor skema untuk menunjukkan perubahan yang dibuat pada output melalui kode kustom tersebut. Saat mengedit skema, Anda dapat melakukan tindakan-tindakan berikut:

- Menambah atau menghapus kunci properti data
- Mengubah jenis data kunci properti data
- Mengubah nama kunci properti data
- Melakukan restrukturisasi kunci properti yang di-nest

Anda harus menggunakan SelectFromCollectiontransformasi untuk memilih satu `DynamicFrame` dari hasil node transformasi Kustom Anda sebelum Anda dapat mengirim output ke lokasi target.

Gunakan tugas berikut untuk menambahkan simpul transformasi kustom ke diagram tugas Anda.

Menambahkan simpul transformasi kode kustom ke diagram tugas

Untuk menambahkan simpul transformasi kustom ke diagram tugas Anda

1. (Opsional) Buka panel Resource dan kemudian pilih Custom transform untuk menambahkan transformasi kustom ke diagram pekerjaan Anda.
2. Pada tab Properti simpul, masukkan nama untuk simpul dalam diagram tugas. Jika sebuah simpul induk belum dipilih, atau jika Anda ingin beberapa masukan untuk transformasi kustom, kemudian pilih sebuah simpul dari Induk simpul yang akan digunakan sebagai sumber masukan untuk transformasi.

Memasukkan kode untuk simpul transformasi kustom

Anda dapat mengetik atau menyalin kode ke dalam bidang input. Tugas menggunakan kode ini untuk melakukan transformasi data. Anda dapat memberikan potongan kode baik Python atau Scala. Kode harus mengambil satu atau beberapa `DynamicFrames` sebagai masukan dan mengembalikan sebuah koleksi `DynamicFrames`.

Untuk memasukkan skrip untuk simpul transformasi kustom

1. Dengan simpul transformasi kustom yang sudah dipilih dalam diagram tugas, pilih tab Transformasi.
2. Dalam bidang entri teks pada judul Blok kode, tempel atau masukkan kode untuk transformasi. Kode yang Anda gunakan harus sesuai dengan bahasa yang ditentukan untuk tugas di tab Detail tugas.

   Saat mengacu pada node input dalam kode Anda, AWS Glue Studio beri nama yang `DynamicFrames` dikembalikan oleh node diagram pekerjaan secara berurutan berdasarkan urutan pembuatan. Gunakan salah satu metode penamaan berikut dalam kode Anda:

   - Pembuatan kode klasik — Gunakan nama fungsional untuk merujuk ke node dalam diagram pekerjaan Anda.
     - Simpul sumber data :`DataSource0`, `DataSource1`, `DataSource2`, dan seterusnya.
     - Simpul transformasi: `Transform0`, `Transform1`, `Transform2`, dan seterusnya.
   - Pembuatan kode baru - Gunakan nama yang ditentukan pada tab properti Node dari sebuah node, ditambahkan dengan '`_node1`', '`_node2`', dan seterusnya. Misalnya,`S3bucket_node1`, `ApplyMapping_node2S3bucket_node2`,`MyCustomNodeName_node1`.

Untuk informasi selengkapnya tentang pembuat kode baru, lihatPembuatan kode skrip.

Contoh berikut menunjukkan format kode yang akan dimasukkan dalam kotak kode:

Python

Contoh berikut mengambil DynamicFrame yang pertama diterima, mengonversinya menjadi DataFrame untuk menerapkan metode filter asli (hanya menyimpan catatan yang memiliki lebih dari 1000 suara), kemudian mengubahnya kembali menjadi DynamicFrame sebelum mengembalikannya.

```
def FilterHighVoteCounts (glueContext, dfc) -> DynamicFrameCollection:
    df = dfc.select(list(dfc.keys())[0]).toDF()
    df_filtered = df.filter(df["vote_count"] > 1000)
    dyf_filtered = DynamicFrame.fromDF(df_filtered, glueContext, "filter_votes")
    return(DynamicFrameCollection({"CustomTransform0": dyf_filtered}, glueContext))
```

Scala

Contoh berikut mengambil DynamicFrame yang pertama diterima, mengonversinya menjadi DataFrame untuk menerapkan metode filter asli (hanya menyimpan catatan yang memiliki lebih dari 1000 suara), kemudian mengubahnya kembali menjadi DynamicFrame sebelum mengembalikannya.

```
object FilterHighVoteCounts {
  def execute(glueContext : GlueContext, input : Seq[DynamicFrame]) :
 Seq[DynamicFrame] = {
    val frame = input(0).toDF()
    val filtered = DynamicFrame(frame.filter(frame("vote_count") > 1000),
 glueContext)
    Seq(filtered)
  }
}
```

Mengedit skema di simpul transformasi kustom

Saat Anda menggunakan simpul transformasi khusus, AWS Glue Studio tidak dapat secara otomatis menyimpulkan skema keluaran yang dibuat oleh transformasi. Anda menggunakan editor skema untuk menggambarkan perubahan skema yang diimplementasikan oleh kode transformasi kustom.

Sebuah simpul kode kustom dapat memiliki sejumlah simpul induk, masing-masing menyediakan `DynamicFrame` sebagai masukan untuk kode kustom Anda. Sebuah simpul kode kustom mengembalikan sebuah koleksi `DynamicFrames`. Setiap `DynamicFrame` yang digunakan sebagai masukan memiliki skema terkait. Anda harus menambahkan skema yang menggambarkan setiap `DynamicFrame` yang dikembalikan oleh simpul kode kustom.

> **Note**
>
> Saat Anda mengatur skema Anda sendiri pada transformasi khusus, AWS Glue Studio tidak mewarisi skema dari node sebelumnya. Untuk memperbarui skema, pilih simpul transformasi kustom, lalu pilih tab Pratinjau data. Setelah pratinjau dibuat, pilih 'Use Preview Schema'. Skema kemudian akan digantikan oleh skema menggunakan data pratinjau.

Untuk mengedit skema output untuk simpul transformasi kustom

1. Dengan simpul transformasi kustom yang sudah dipilih dalam diagram tugas, di panel detail simpul, pilih tab Skema output.
2. Pilih Edit untuk membuat perubahan pada skema.

   Jika Anda memiliki kunci properti data yang di-nest, seperti array atau objek, Anda dapat memilih iko Expand-Rows
   ( )
   di kanan atas dari setiap panel skema untuk memperluas daftar kunci properti
   data anak. Setelah Anda memilih ikon ini, ia akan berubah ke ikon Collapse-Rows
   ( ),
   yang dapat Anda pilih untuk menutup daftar kunci properti anak.
3. Memodifikasi skema menggunakan tindakan berikut di bagian di sisi kanan halaman:

   - Untuk mengubah nama kunci properti, tempatkan kursor di kotak teks Kunci untuk kunci properti, lalu masukkan nama baru.

- Untuk mengubah tipe data untuk kunci properti, gunakan daftar untuk memilih tipe data baru untuk kunci properti.
- Untuk menambahkan kunci properti tingkat atas baru pada skema, pilih ikon Overflow (⋯) di sebelah kiri tombol Batalkan, dan kemudian pilih Tambahkan kunci akar.
- Untuk menambahkan kunci properti anak untuk skema, pilih ikon Add-Key yang dikaitkan dengan kunci induk. Pilih nama untuk kunci anak dan pilih tipe data.
- Untuk menghapus kunci properti dari skema, pilih ikon Hapus ( ) yang ada di ujung kanan nama kunci.

4. Jika kode transformasi kustom Anda menggunakan beberapa `DynamicFrames`, Anda dapat menambahkan skema output tambahan.

   - Untuk menambahkan skema baru dan kosong, pilih ikon Overflow (⋯), dan kemudian pilih Tambah skema output.
   - Untuk menyalin skema yang ada ke skema output baru, pastikan skema yang ingin Anda salin ditampilkan dalam pemilih skema. Pilih ikon Overflow (⋯), dan kemudian pilih Duplikasi.

   Jika Anda ingin menghapus sebuah skema output, pastikan skema yang ingin Anda salin ditampilkan dalam pemilih skema. Pilih ikon Overflow (⋯), dan kemudian pilih Hapus.

5. Menambahkan kunci akar baru untuk skema baru atau mengedit kunci yang diduplikasi.
6. Ketika Anda memodifikasi skema output, pilih Terapkan untuk menyimpan perubahan dan keluar dari editor skema.

   Jika Anda tidak ingin menyimpan perubahan, pilih tombol Batalkan.

Mengkonfigurasi output transformasi kustom

Sebuah transformasi kode kustom mengembalikan sebuah koleksi `DynamicFrames`, bahkan jika hanya ada satu `DynamicFrame` di set yang dihasilkan.

Untuk memproses output dari sebuah simpul transformasi kustom

1. Tambahkan node SelectFromCollectiontransformasi, yang memiliki node transformasi kustom sebagai simpul induknya. Perbarui transformasi ini untuk menunjukkan set data yang ingin Anda gunakan. Lihat Menggunakan SelectFromCollection untuk memilih kumpulan data mana yang akan disimpan untuk informasi selengkapnya.
2. Tambahkan SelectFromCollectiontransformasi tambahan ke diagram pekerjaan jika Anda ingin menggunakan tambahan yang `DynamicFrames` dihasilkan oleh node transformasi kustom.

   Pertimbangkan skenario di mana Anda menambahkan simpul transformasi kustom untuk membagi set data penerbangan menjadi beberapa set data, tetapi duplikat beberapa kunci properti pengidentifikasi di setiap skema output, seperti tanggal penerbangan atau nomor penerbangan. Anda menambahkan node SelectFromCollectiontransformasi untuk setiap skema output, dengan node transformasi kustom sebagai induknya.
3. (Opsional) Anda kemudian dapat menggunakan setiap node SelectFromCollectiontransformasi sebagai input untuk node lain dalam pekerjaan, atau sebagai induk untuk node target data.

AWS Gluetransformasi visual khusus

Transformasi visual khusus memungkinkan Anda membuat transformasi dan membuatnya tersedia untuk digunakan dalam AWS Glue Studio pekerjaan. Transformasi visual khusus memungkinkan pengembang ETL, yang mungkin tidak terbiasa dengan pengkodean, untuk mencari dan menggunakan perpustakaan transformasi yang berkembang menggunakan antarmuka. AWS Glue Studio

Anda dapat membuat transformasi visual khusus, lalu mengunggahnya ke Amazon S3 agar tersedia untuk digunakan melalui editor visual AWS Glue Studio untuk bekerja dengan pekerjaan ini.

Topik

- Memulai dengan transformasi visual khusus
- Langkah 1. Buat file konfigurasi JSON
- Langkah 2. Menerapkan logika transformasi

- Langkah 3. Validasi dan pecahkan masalah transformasi visual khusus di AWS Glue Studio
- Langkah 4. Perbarui transformasi visual khusus sesuai kebutuhan
- Langkah 5. Gunakan transformasi visual khusus di AWS Glue Studio
- Contoh penggunaan
- Contoh skrip visual kustom
- Video

Memulai dengan transformasi visual khusus

Untuk membuat transformasi visual khusus, Anda melalui langkah-langkah berikut.

- Langkah 1. Buat file konfigurasi JSON
- Langkah 2. Menerapkan logika transformasi
- Langkah 3. Validasi transformasi visual kustom
- Langkah 4. Perbarui transformasi visual khusus sesuai kebutuhan
- Langkah 5. Gunakan transformasi visual khusus di AWS Glue Studio

Mulailah dengan menyiapkan bucket Amazon S3 dan lanjutkan ke Langkah 1. Buat file konfigurasi JSON.

Prasyarat

Transformasi yang disediakan pelanggan berada dalam akun pelanggan. AWS Akun itu memiliki transformasi dan karenanya memiliki semua izin untuk melihat (mencari dan menggunakan), mengedit, atau menghapusnya.

Untuk menggunakan transformasi khususAWS Glue Studio, Anda harus membuat dan mengunggah dua file ke bucket aset Amazon S3 di akun tersebutAWS:

- File Python - berisi fungsi transformasi
- File JSON - menjelaskan transformasi. Ini juga dikenal sebagai file konfigurasi yang diperlukan untuk mendefinisikan transformasi.

Untuk memasangkan file bersama-sama, gunakan nama yang sama untuk keduanya. Misalnya:

- MyTransform.json

- myTransform.py

Secara opsional, Anda dapat memberikan transformasi visual kustom Anda ikon kustom dengan menyediakan file SVG yang berisi ikon. Untuk memasangkan file bersama-sama, gunakan nama yang sama untuk ikon:

- myTransform.svg

AWS Glue Studioakan secara otomatis mencocokkan mereka menggunakan nama file masing-masing. Nama file tidak boleh sama untuk modul yang ada.

Konvensi yang direkomendasikan untuk mengubah nama file

AWS Glue Studioakan mengimpor file Anda sebagai modul (misalnya,`import myTransform`) dalam skrip pekerjaan Anda. Oleh karena itu, nama file Anda harus mengikuti aturan penamaan yang sama yang ditetapkan untuk nama variabel python (pengidentifikasi). Secara khusus, mereka harus mulai dengan huruf atau garis bawah dan kemudian seluruhnya terdiri dari huruf, digit, dan/atau garis bawah.

> **Note**
>
> Pastikan nama file transform Anda tidak bertentangan dengan modul python yang dimuat (misalnya, `sys, array, copy` dll.) Untuk menghindari masalah runtime yang tidak terduga.

Menyiapkan bucket Amazon S3

Transformasi yang Anda buat disimpan di Amazon S3 dan dimiliki oleh AWS akun Anda. Anda membuat transformasi visual kustom baru hanya dengan mengunggah file (json dan py) ke folder aset Amazon S3 tempat semua skrip pekerjaan saat ini disimpan (misalnya,). `s3://aws-glue-assets-<accountid>-<region>/transforms` Jika menggunakan ikon khusus, unggah juga. Secara default, AWS Glue Studio akan membaca semua file.json dari folder /transforms di bucket S3 yang sama.

Langkah 1. Buat file konfigurasi JSON

File konfigurasi JSON diperlukan untuk menentukan dan mendeskripsikan transformasi visual kustom Anda. Skema untuk file konfigurasi adalah sebagai berikut.

Struktur file JSON

Bidang

- `name: string` — (diperlukan) nama sistem transformasi yang digunakan untuk mengidentifikasi transformasi. Ikuti aturan penamaan yang sama yang ditetapkan untuk nama variabel python (pengidentifikasi). Secara khusus, mereka harus mulai dengan huruf atau garis bawah dan kemudian seluruhnya terdiri dari huruf, digit, dan/atau garis bawah.
- `displayName: string` — (opsional) nama transformasi yang ditampilkan di editor pekerjaan AWS Glue Studio visual. Jika tidak `displayName` ditentukan, `name` digunakan sebagai nama transformasi diAWS Glue Studio.
- `description: string` — (opsional) deskripsi transformasi ditampilkan AWS Glue Studio dan dapat dicari.
- `functionName: string` — (wajib) nama fungsi Python digunakan untuk mengidentifikasi fungsi untuk memanggil dalam skrip Python.
- `path: string` — (opsional) jalur Amazon S3 lengkap ke file sumber Python. Jika tidak ditentukan, AWS Glue gunakan pencocokan nama file untuk memasangkan file.json dan .py bersama-sama. Misalnya, nama file JSON,`myTransform.json`, akan dipasangkan ke file Python,`myTransform.py`, di lokasi Amazon S3 yang sama.
- `parameters: Array of TransformParameter object` — (opsional) daftar parameter yang akan ditampilkan saat Anda mengonfigurasinya di editor AWS Glue Studio visual.

TransformParameter bidang

- `name: string` — (required) nama parameter yang akan diteruskan ke fungsi python sebagai argumen bernama dalam skrip pekerjaan. Ikuti aturan penamaan yang sama yang ditetapkan untuk nama variabel python (pengidentifikasi). Secara khusus, mereka harus mulai dengan huruf atau garis bawah dan kemudian seluruhnya terdiri dari huruf, digit, dan/atau garis bawah.
- `displayName: string` — (opsional) nama transformasi yang ditampilkan di editor pekerjaan AWS Glue Studio visual. Jika tidak `displayName` ditentukan, `name` digunakan sebagai nama transformasi diAWS Glue Studio.
- `type: string` — (wajib) tipe parameter yang menerima tipe data Python umum. Nilai yang valid: 'str' | 'int' | 'float' | 'list' | 'bool'.
- `isOptional: boolean` — (opsional) menentukan apakah parameter opsional. Secara default semua parameter diperlukan.

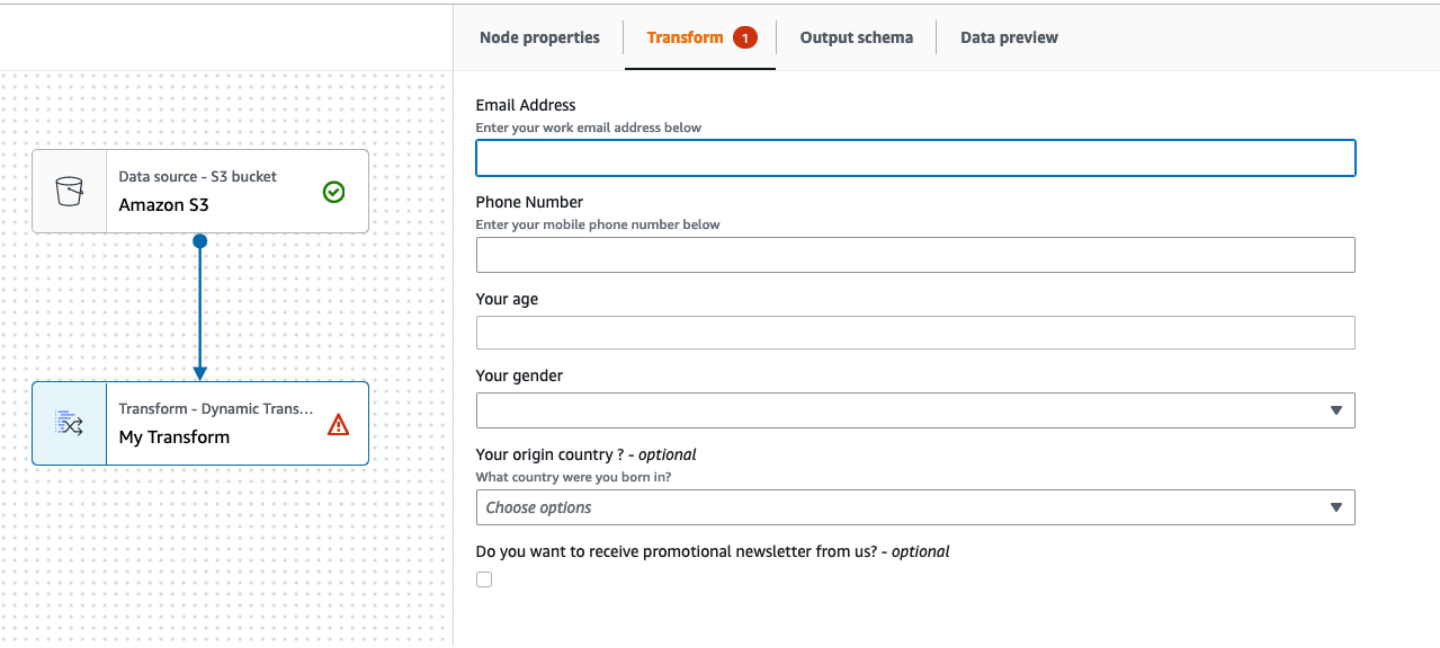
- `description: string`— (opsional) deskripsi ditampilkan AWS Glue Studio untuk membantu pengguna mengkonfigurasi parameter transformasi.
- `validationType: string`— (opsional) mendefinisikan cara parameter ini divalidasi. Saat ini, ini hanya mendukung ekspresi reguler. Secara default, jenis validasi diatur ke`RegularExpression`.
- `validationRule: string`— (opsional) ekspresi reguler yang digunakan untuk memvalidasi input formulir sebelum mengirimkan ketika `validationType` diatur ke`RegularExpression`. Sintaks ekspresi reguler harus kompatibel dengan spesifikasi RegExp Ecmascript.
- `validationMessage: string`— (opsional) pesan untuk ditampilkan ketika validasi gagal.
- `listOptions: An array of TransformParameterListOption objectATAU a string` atau nilai string 'kolom' - (opsional) opsi untuk ditampilkan di kontrol Select atau Multiselect UI. Menerima daftar nilai dipisahkan koma atau tipe objek JSON yang sangat kuat. `TransformParameterListOption` Hal ini juga dapat secara dinamis mengisi daftar kolom dari skema node induk dengan menentukan nilai string “kolom”.
- `listType: string`— (opsional) Tentukan jenis opsi untuk tipe = 'daftar'. Nilai yang valid: 'str' | 'int' | 'float' | 'list' | 'bool'. Jenis parameter menerima tipe data python umum.

TransformParameterListOption bidang

- `value: string | int | float | bool`— (wajib) nilai opsi.
- `label: string`- (opsional) label opsi ditampilkan di pilih dropdown.

Ubah parameter di AWS Glue Studio

Secara default, parameter diperlukan kecuali tandai seperti `isOptional` pada file.json. DalamAWS Glue Studio, parameter ditampilkan di tab Transform. Contoh ini menunjukkan parameter yang ditentukan pengguna seperti Alamat Email, Nomor Telepon, Usia Anda, Jenis kelamin Anda, dan negara asal Anda.

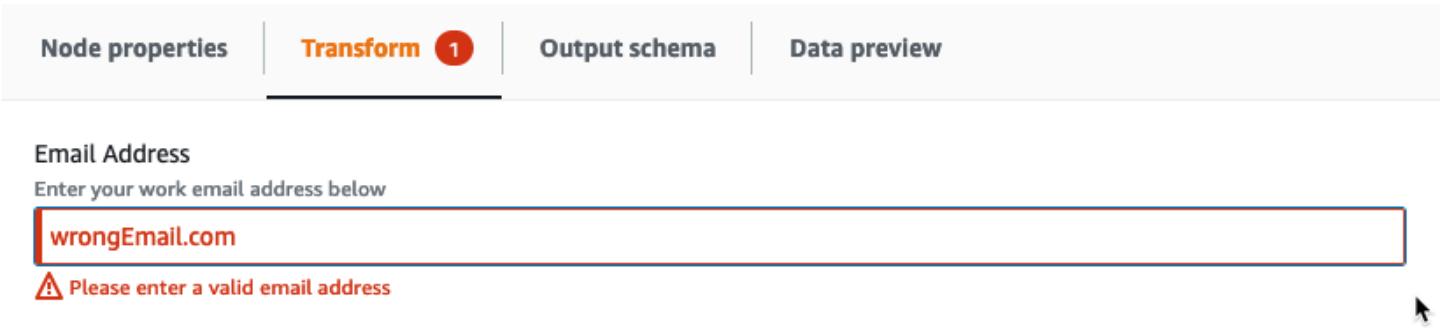


Anda dapat menerapkan beberapa validasi dalam AWS Glue Studio menggunakan ekspresi reguler dalam file json dengan menentukan `validationRule` parameter dan menentukan pesan validasi di. `validationMessage`

```
    "validationRule": "^\\(?(\\d{3})\\)?[- ]?(\\d{3})[- ]?(\\d{4})$",
    "validationMessage": "Please enter a valid US number"
```

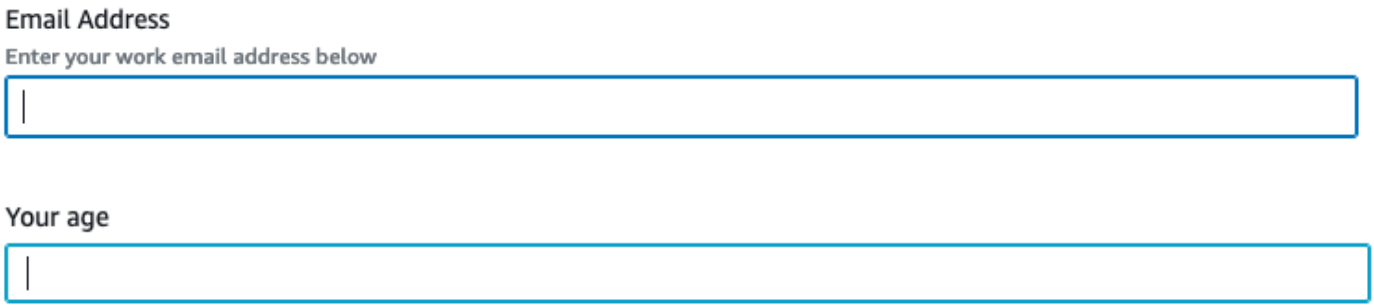
> **Note**
>
> Karena validasi terjadi di browser, sintaks ekspresi reguler Anda harus kompatibel dengan spesifikasi RegExp Ecmascript. Sintaks Python tidak didukung untuk ekspresi reguler ini.

Menambahkan validasi akan mencegah pengguna menyimpan pekerjaan dengan input pengguna yang salah. AWS Glue Studiomenampilkan pesan validasi seperti yang ditampilkan dalam contoh:

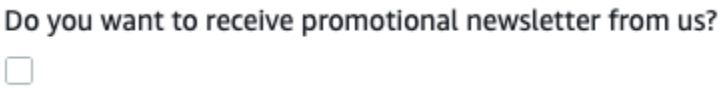


Parameter ditampilkan AWS Glue Studio berdasarkan konfigurasi parameter.

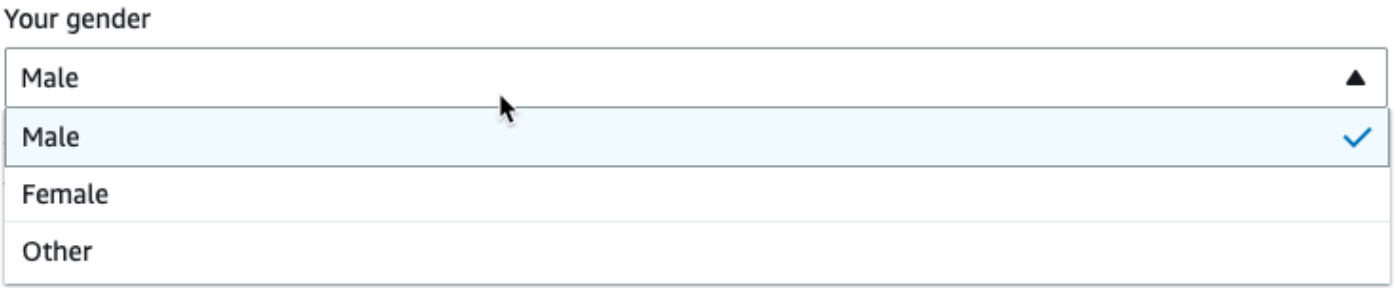
- `type`Kapan salah satu dari berikut ini: `str`, `float`, `int` atau, bidang input teks ditampilkan. Misalnya, tangkapan layar menunjukkan bidang input untuk parameter 'Alamat Email' dan 'Usia Anda'.



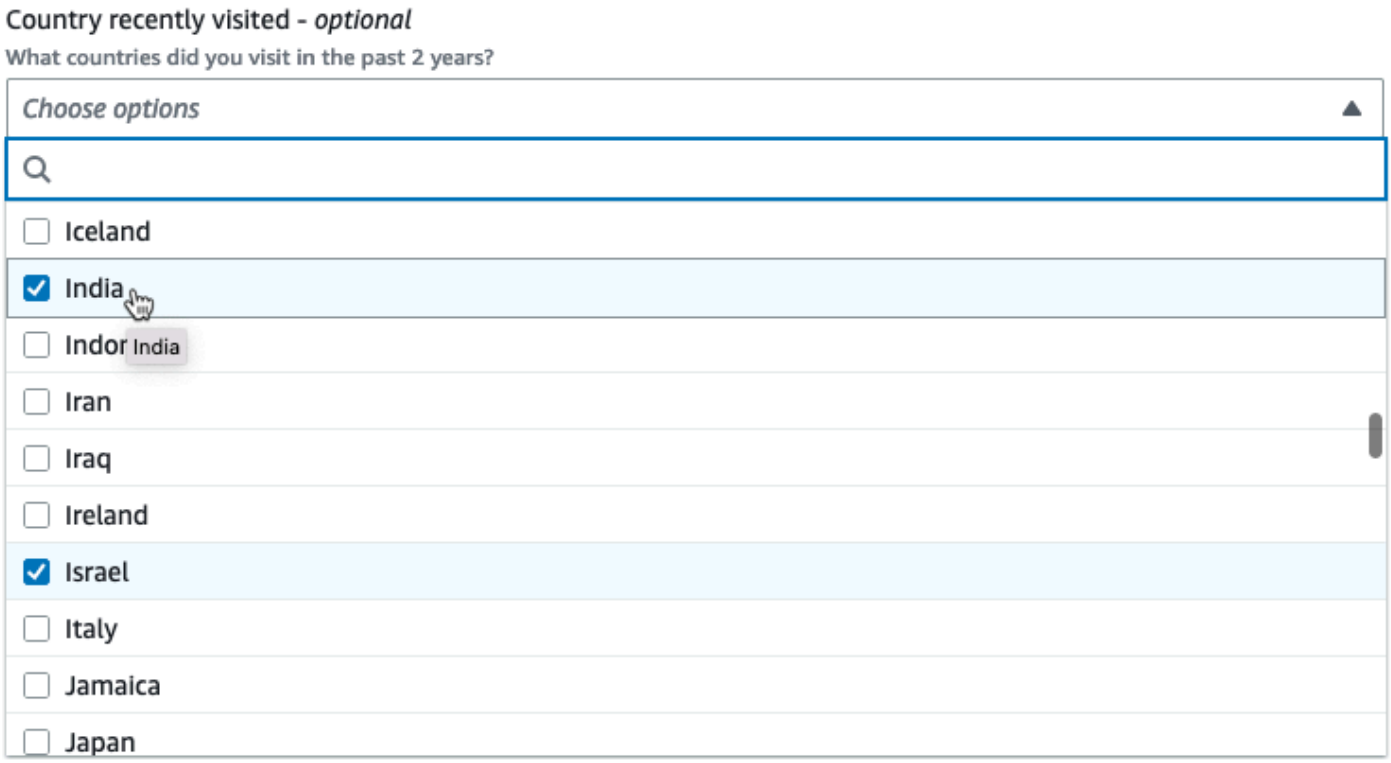
- `type`Kapan`bool`, kotak centang ditampilkan.



- `type`Kapan `str` dan `listOptions` disediakan, satu daftar pilih ditampilkan.



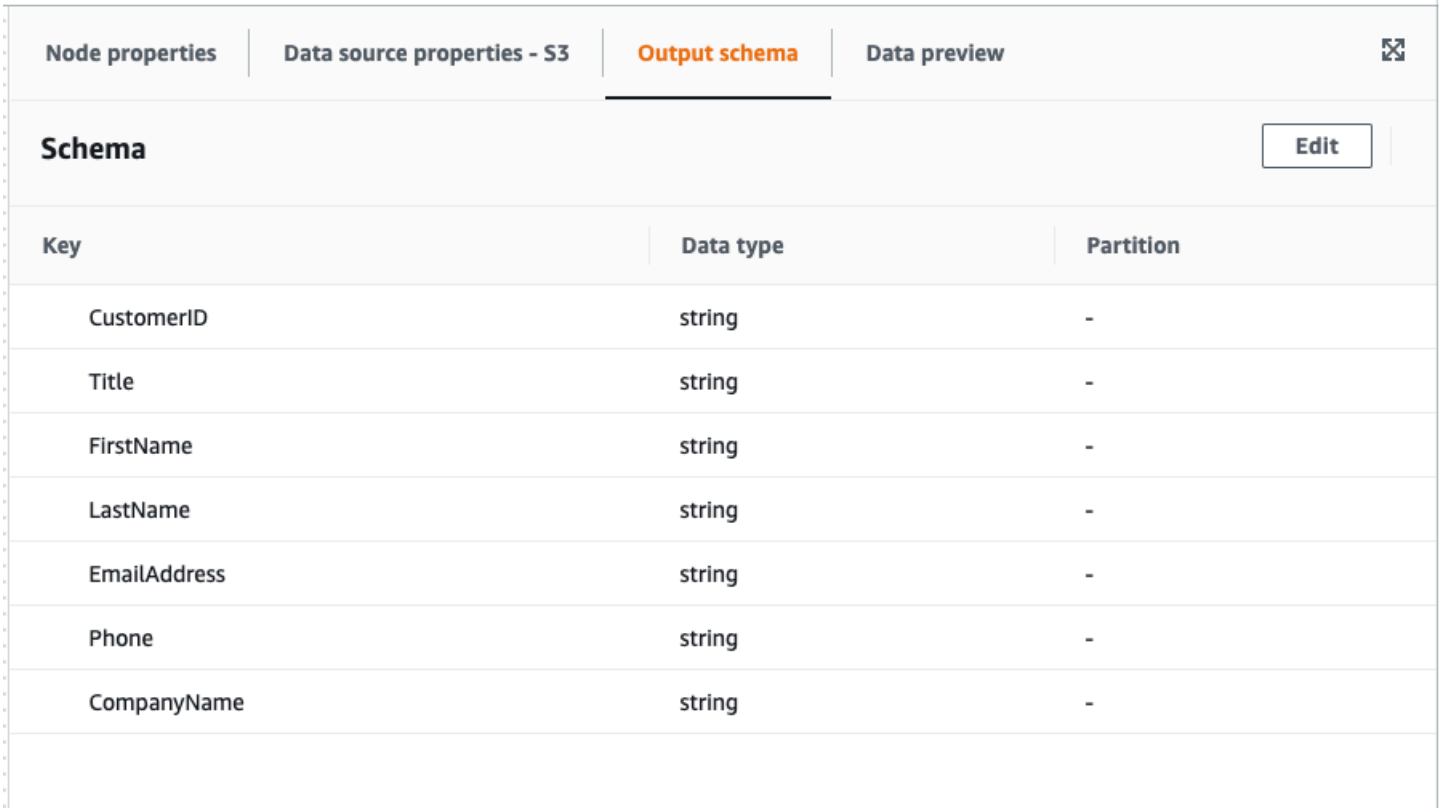
- `type`Kapan `list` dan `listOptions` dan `listType` disediakan, daftar multi-pilih ditampilkan.



Menampilkan pemilih kolom sebagai parameter

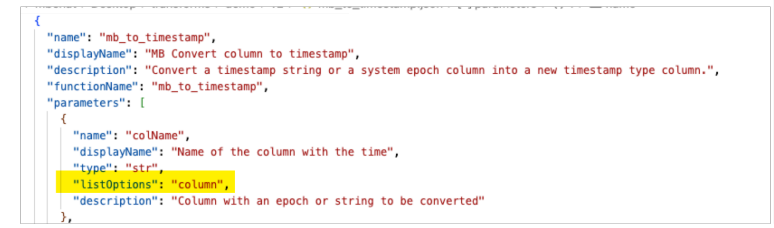
Jika konfigurasi mengharuskan pengguna untuk memilih kolom dari skema, Anda dapat menampilkan pemilih kolom sehingga pengguna tidak diharuskan untuk mengetikkan nama kolom. Dengan mengatur `listOptions` bidang ke '"kolom", AWS Glue Studio secara dinamis menampilkan pemilih kolom berdasarkan skema keluaran node induk. AWS Glue Studiodapat menampilkan pemilih kolom tunggal atau beberapa.

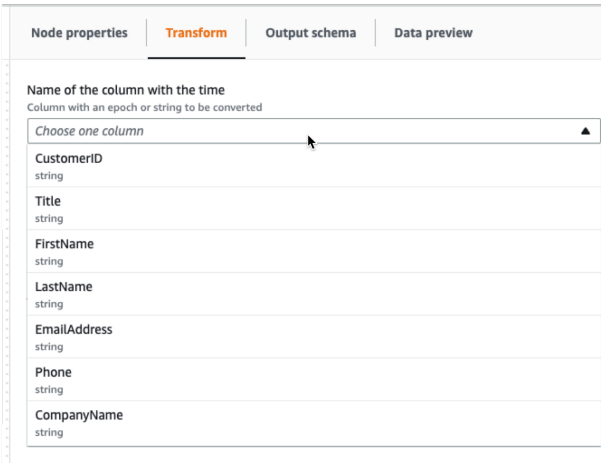
Contoh berikut menggunakan skema:



Untuk menentukan parameter Custom Visual Transform Anda untuk menampilkan satu kolom:

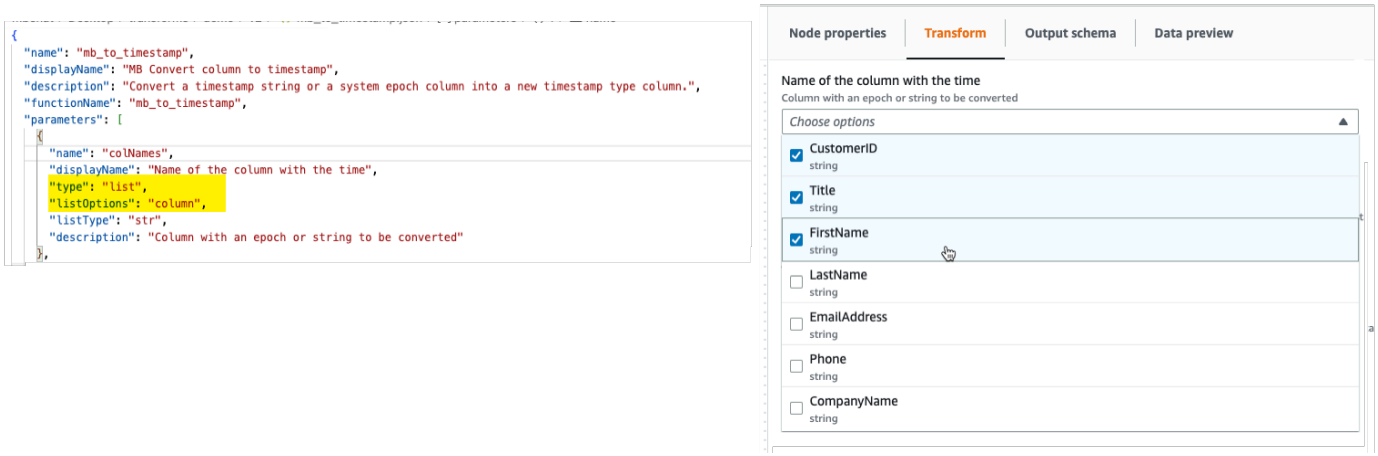
1. Dalam file JSON Anda, untuk `parameters` objek, atur `listOptions` nilainya ke “kolom”. Hal ini memungkinkan pengguna untuk memilih kolom dari daftar pilih diAWS Glue Studio.





2. Anda juga dapat mengizinkan pemilihan beberapa kolom dengan mendefinisikan parameter sebagai:

   - `listOptions: "column"`
   - `type: "list"`



Langkah 2. Menerapkan logika transformasi

> **Note**
>
> Transformasi visual khusus hanya mendukung skrip Python. Scala tidak didukung.

Untuk menambahkan kode yang mengimplementasikan fungsi yang ditentukan oleh file konfigurasi.json, disarankan untuk menempatkan file Python di lokasi yang sama dengan file.json, dengan nama yang sama tetapi dengan ekstensi “.py”. AWS Glue Studiosecara otomatis memasangkan file.json dan .py sehingga Anda tidak perlu menentukan jalur file Python di file konfigurasi.

Dalam file Python, tambahkan fungsi yang dideklarasikan, dengan parameter bernama dikonfigurasi dan daftarkan untuk digunakan. `DynamicFrame` Berikut ini adalah contoh file Python:

```
from awsglue import DynamicFrame

# self refers to the DynamicFrame to transform,
# the parameter names must match the ones defined in the config
# if it's optional, need to provide a default value
def myTransform(self, email, phone, age=None, gender="",
                      country="", promotion=False):
   resulting_dynf = # do some transformation on self
   return resulting_dynf

DynamicFrame.myTransform = myTransform
```

Disarankan untuk menggunakan AWS Glue notebook untuk cara tercepat untuk mengembangkan dan menguji kode python. Lihat Memulai dengan notebook di AWS Glue Studio.

Untuk mengilustrasikan cara menerapkan logika transformasi, transformasi visual khusus pada contoh di bawah ini adalah transformasi untuk memfilter data yang masuk agar hanya menyimpan data yang terkait dengan status AS tertentu. File.json berisi parameter untuk `functionName` as `custom_filter_state` dan dua argumen (“state” dan “colName” dengan tipe “str”).

Contoh file config .json adalah:

```
{
"name": "custom_filter_state",
"displayName": "Filter State",
"description": "A simple example to filter the data to keep only the state indicated.",
"functionName": "custom_filter_state",
"parameters": [
   {
    "name": "colName",
    "displayName": "Column name",
    "type": "str",
    "description": "Name of the column in the data that holds the state postal code"
   },
   {
    "name": "state",
    "displayName": "State postal code",
    "type": "str",
    "description": "The postal code of the state whole rows to keep"
   }
  ]
}
```

Untuk mengimplementasikan skrip pendamping dengan Python

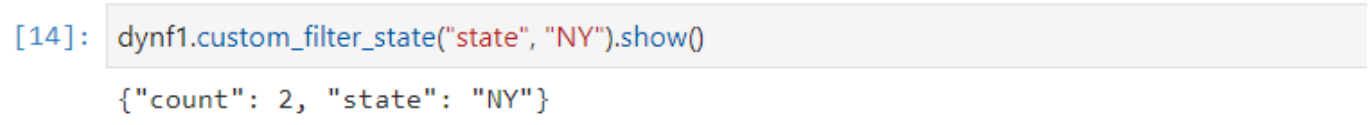
1. Mulai buku AWS Glue catatan dan jalankan sel awal yang disediakan untuk sesi yang akan dimulai. Menjalankan sel awal menciptakan komponen dasar yang diperlukan.
2. Buat fungsi yang melakukan pemfilteran seperti yang dijelaskan dalam contoh dan daftarkan. `DynamicFrame` Salin kode di bawah ini dan tempel ke sel di AWS Glue buku catatan.

```
from awsglue import DynamicFrame

def custom_filter_state(self, colName, state):
    return self.filter(lambda row: row[colName] == state)

DynamicFrame.custom_filter_state = custom_filter_state
```

3. Buat atau muat data sampel untuk menguji kode di sel yang sama atau sel baru. Jika Anda menambahkan data sampel di sel baru, jangan lupa untuk menjalankan sel. Sebagai contoh:

```
# A few of rows of sample data to test
data_sample = [
    {"state": "CA", "count": 4},
    {"state": "NY", "count": 2},
    {"state": "WA", "count": 3}
]
df1 = glueContext.sparkSession.sparkContext.parallelize(data_sample).toDF()
dynf1 = DynamicFrame.fromDF(df1, glueContext, None)
```

4. Uji untuk memvalidasi “custom_filter_state” dengan argumen yang berbeda:



5. Setelah menjalankan beberapa tes, simpan kode dengan ekstensi.py dan beri nama file.py dengan nama yang mencerminkan nama file.json. File.py dan.json harus berada di folder transformasi yang sama.

   Salin kode berikut dan tempel ke file dan ganti nama dengan ekstensi file.py.

```
from awsglue import DynamicFrame

def custom_filter_state(self, colName, state):
    return self.filter(lambda row: row[colName] == state)

DynamicFrame.custom_filter_state = custom_filter_state
```

6. DiAWS Glue Studio, buka pekerjaan visual dan tambahkan transformasi ke pekerjaan dengan memilihnya dari daftar Transformasi yang tersedia.

   Untuk menggunakan kembali transformasi ini dalam kode skrip Python, tambahkan jalur Amazon S3 ke file.py dalam pekerjaan di bawah “Jalur file yang direferensikan” dan dalam skrip, impor nama file python (tanpa ekstensi) dengan menambahkannya ke bagian atas file. Sebagai contoh: `import` <name of the file (without the extension) >

Langkah 3. Validasi dan pecahkan masalah transformasi visual khusus di AWS Glue Studio

AWS Glue Studiomemvalidasi file konfigurasi JSON sebelum transformasi visual khusus dimuat ke dalam. AWS Glue Studio Validasi meliputi:

- Kehadiran bidang wajib
- Validasi format JSON
- Parameter salah atau tidak valid
- Kehadiran file.py dan .json di jalur Amazon S3 yang sama
- Mencocokkan nama file untuk .py dan .json

Jika validasi berhasil, transformasi tercantum dalam daftar Tindakan yang tersedia di editor visual. Jika ikon kustom telah disediakan, itu akan terlihat di samping Action.

Jika validasi gagal, AWS Glue Studio tidak memuat transformasi visual khusus.

Langkah 4. Perbarui transformasi visual khusus sesuai kebutuhan

Setelah dibuat dan digunakan, skrip transformasi dapat diperbarui selama transformasi mengikuti definisi json yang sesuai:

- Nama yang digunakan saat menetapkan untuk DynamicFrame banyak cocok dengan `functionName` json.
- Argumen fungsi harus didefinisikan dalam file json seperti yang dijelaskan dalam Langkah 1. Buat file konfigurasi JSON .
- Jalur Amazon S3 dari file Python tidak dapat berubah, karena pekerjaan bergantung langsung padanya.

> **Note**
>
> Jika ada pembaruan yang perlu dilakukan, pastikan skrip dan file.json diperbarui secara konsisten dan pekerjaan visual apa pun disimpan dengan benar lagi dengan transformasi baru. Jika pekerjaan visual tidak disimpan setelah pembaruan dilakukan, pembaruan tidak akan diterapkan dan divalidasi. Jika file skrip Python diganti namanya atau tidak ditempatkan di sebelah file.json, maka Anda perlu menentukan path lengkap dalam file.json.

Ikon kustom

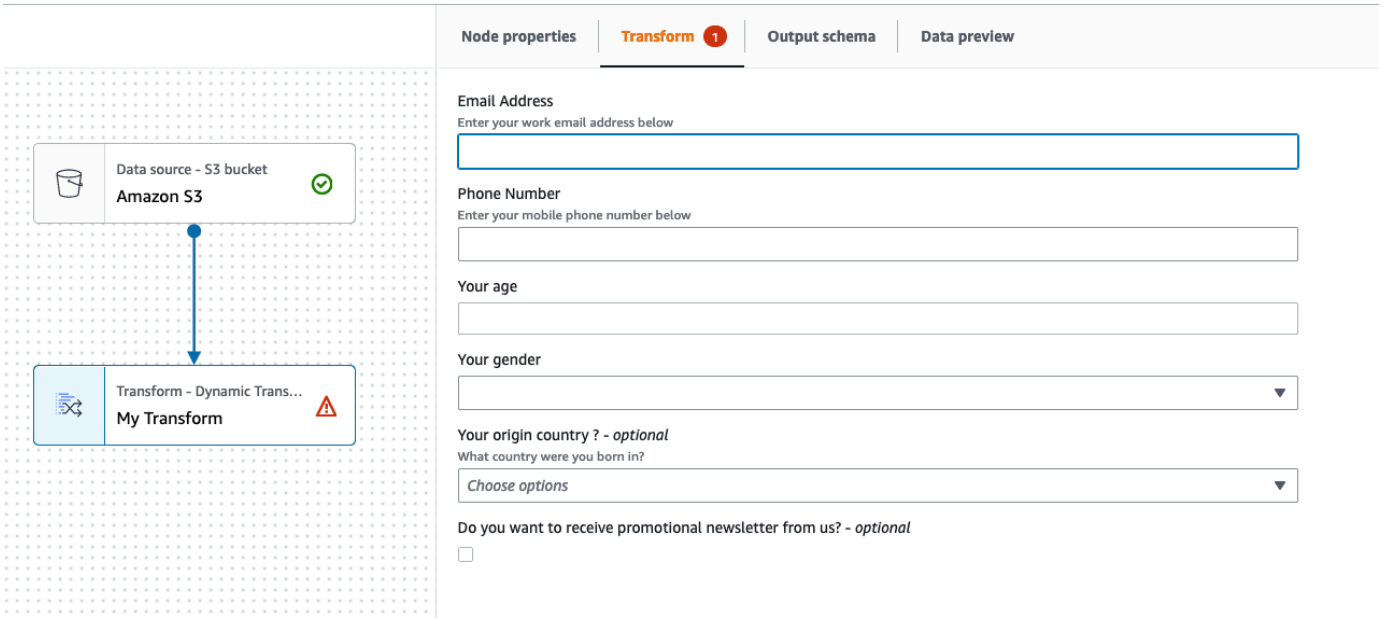
Jika Anda menentukan ikon default untuk Tindakan tidak membedakannya secara visual sebagai bagian dari alur kerja, Anda dapat memberikan ikon kustom, seperti yang dijelaskan dalamthe section called “ Memulai dengan transformasi visual khusus ”. Anda dapat memperbarui ikon dengan memperbarui SVG terkait yang dihosting di Amazon S3.

Untuk hasil terbaik, desain gambar Anda untuk dilihat pada 32x32px mengikuti pedoman dari Sistem Desain Cloudscape. Untuk informasi selengkapnya tentang pedoman Cloudscape, lihat Dokumentasi Cloudscape

Langkah 5. Gunakan transformasi visual khusus di AWS Glue Studio

Untuk menggunakan transformasi visual khususAWS Glue Studio, Anda mengunggah konfigurasi dan file sumber, lalu pilih transformasi dari menu Tindakan. Parameter apa pun yang membutuhkan nilai atau input tersedia untuk Anda di tab Transform.

1. Unggah dua file (file sumber Python dan file konfigurasi JSON) ke folder aset Amazon S3 tempat skrip pekerjaan disimpan. Secara default, AWS Glue menarik semua file JSON dari folder / transforms dalam bucket Amazon S3 yang sama.
2. Dari menu Tindakan, pilih transformasi visual khusus. Ini dinamai dengan transformasi `displayName` atau nama yang Anda tentukan dalam file konfigurasi.json.
3. Masukkan nilai untuk parameter apa pun yang dikonfigurasi dalam file konfigurasi.



Contoh penggunaan

Berikut ini adalah contoh dari semua parameter yang mungkin dalam file konfigurasi.json.

```
{
  "name": "MyTransform",
  "displayName": "My Transform",
  "description": "This transform description will be displayed in UI",
  "functionName": "myTransform",
  "parameters": [
      {
      "name": "email",
      "displayName": "Email Address",
      "type": "str",
      "description": "Enter your work email address below",
      "validationType": "RegularExpression",
      "validationRule": "^\\w+([\\.-]?\\w+)*@\\w+([\\.-]?\\w+)*(\\.\\w{2,3})+$",
      "validationMessage": "Please enter a valid email address"
    },
    {
      "name": "phone",
      "displayName": "Phone Number",
      "type": "str",
      "description": "Enter your mobile phone number below",
      "validationRule": "^\\(?(\\d{3})\\)?[- ]?(\\d{3})[- ]?(\\d{4})$",
      "validationMessage": "Please enter a valid US number"
```

```
    },
    {
      "name": "age",
      "displayName": "Your age",
      "type": "int",
      "isOptional": true
    },
    {
      "name": "gender",
      "displayName": "Your gender",
      "type": "str",
      "listOptions": [
            {"label": "Male", "value": "male"},
            {"label": "Female", "value": "female"},
            {"label": "Other", "value": "other"}
        ],
      "isOptional": true
    },
    {
      "name": "country",
      "displayName": "Your origin country ?",
      "type": "list",
      "listOptions": "Afghanistan,Albania,Algeria,American
 Samoa,Andorra,Angola,Anguilla,Antarctica,Antigua and
 Barbuda,Argentina,Armenia,Aruba,Australia,Austria,Azerbaijan,Bahamas,Bahrain,Bangladesh,Barbad
 and Herzegovina,Botswana,Bouvet Island,Brazil,British Indian Ocean Territory,Brunei
 Darussalam,Bulgaria,Burkina Faso,Burundi,Cambodia,Cameroon,Canada,Cape
 Verde,Cayman Islands,Central African Republic,Chad,Chile,China,Christmas
 Island,Cocos (Keeling Islands),Colombia,Comoros,Congo,Cook Islands,Costa
 Rica,Cote D'Ivoire (Ivory Coast),Croatia (Hrvatska,Cuba,Cyprus,Czech
 Republic,Denmark,Djibouti,Dominica,Dominican Republic,East Timor,Ecuador,Egypt,El
 Salvador,Equatorial Guinea,Eritrea,Estonia,Ethiopia,Falkland Islands (Malvinas),Faroe
 Islands,Fiji,Finland,France,France,Metropolitan,French Guiana,French Polynesia,French
 Southern
 Territories,Gabon,Gambia,Georgia,Germany,Ghana,Gibraltar,Greece,Greenland,Grenada,Guadeloupe,G
Bissau,Guyana,Haiti,Heard and McDonald Islands,Honduras,Hong
 Kong,Hungary,Iceland,India,Indonesia,Iran,Iraq,Ireland,Israel,Italy,Jamaica,Japan,Jordan,Kazak
 (North),Korea
 (South),Kuwait,Kyrgyzstan,Laos,Latvia,Lebanon,Lesotho,Liberia,Libya,Liechtenstein,Lithuania,Lu
 Islands,Martinique,Mauritania,Mauritius,Mayotte,Mexico,Micronesia,Moldova,Monaco,Mongolia,Mont
 Antilles,New Caledonia,New Zealand,Nicaragua,Niger,Nigeria,Niue,Norfolk
 Island,Northern Mariana Islands,Norway,Oman,Pakistan,Palau,Panama,Papua
 New Guinea,Paraguay,Peru,Philippines,Pitcairn,Poland,Portugal,Puerto
 Rico,Qatar,Reunion,Romania,Russian Federation,Rwanda,Saint Kitts and Nevis,Saint
```

```
 Lucia,Saint Vincent and The Grenadines,Samoa,San Marino,Sao Tome and Principe,Saudi
 Arabia,Senegal,Seychelles,Sierra Leone,Singapore,Slovak Republic,Slovenia,Solomon
 Islands,Somalia,South Africa,S. Georgia and S. Sandwich Isls.,Spain,Sri
 Lanka,St. Helena,St. Pierre and Miquelon,Sudan,Suriname,Svalbard and Jan Mayen
 Islands,Swaziland,Sweden,Switzerland,Syria,Tajikistan,Tanzania,Thailand,Togo,Tokelau,Tonga,Tri
 and Tobago,Tunisia,Turkey,Turkmenistan,Turks and Caicos
 Islands,Tuvalu,Uganda,Ukraine,United Arab Emirates,United Kingdom
 (Britain / UK),United States of America (USA),US Minor Outlying
 Islands,Uruguay,Uzbekistan,Vanuatu,Vatican City State (Holy See),Venezuela,Viet
 Nam,Virgin Islands (British),Virgin Islands (US),Wallis and Futuna Islands,Western
 Sahara,Yemen,Yugoslavia,Zaire,Zambia,Zimbabwe",
      "description": "What country were you born in?",
      "listType": "str",
      "isOptional": true
    },
    {
      "name": "promotion",
      "displayName": "Do you want to receive promotional newsletter from us?",
      "type": "bool",
      "isOptional": true
    }
  ]
}
```

Contoh skrip visual kustom

Contoh-contoh berikut melakukan transformasi yang setara. Namun, contoh kedua (SparkSQL) adalah yang terbersih dan paling efisien, diikuti oleh Pandas UDF dan akhirnya pemetaan tingkat rendah pada contoh pertama. Contoh berikut adalah contoh lengkap dari transformasi sederhana untuk menambahkan dua kolom:

```
from awsglue import DynamicFrame

# You can have other auxiliary variables, functions or classes on this file, it won't
 affect the runtime
def record_sum(rec, col1, col2, resultCol):
    rec[resultCol] = rec[col1] + rec[col2]
    return rec


# The number and name of arguments must match the definition on json config file
# (expect self which is the current DynamicFrame to transform
```

```
# If an argument is optional, you need to define a default value here
#  (resultCol in this example is an optional argument)
def custom_add_columns(self, col1, col2, resultCol="result"):
    # The mapping will alter the columns order, which could be important
    fields = [field.name for field in self.schema()]
    if resultCol not in fields:
        # If it's a new column put it at the end
        fields.append(resultCol)
    return self.map(lambda record: record_sum(record, col1, col2,
 resultCol)).select_fields(paths=fields)


# The name we assign on DynamicFrame must match the configured "functionName"
DynamicFrame.custom_add_columns = custom_add_columns
```

Contoh berikut adalah transformasi setara yang memanfaatkan SparkSQL API.

```
from awsglue import DynamicFrame

# The number and name of arguments must match the definition on json config file
# (expect self which is the current DynamicFrame to transform
# If an argument is optional, you need to define a default value here
#  (resultCol in this example is an optional argument)
def custom_add_columns(self, col1, col2, resultCol="result"):
    df = self.toDF()
    return DynamicFrame.fromDF(
        df.withColumn(resultCol, df[col1] + df[col2]) # This is the conversion logic
        , self.glue_ctx, self.name)


# The name we assign on DynamicFrame must match the configured "functionName"
DynamicFrame.custom_add_columns = custom_add_columns
```

Contoh berikut menggunakan transformasi yang sama tetapi menggunakan panda UDF, yang lebih efisien daripada menggunakan UDF biasa. Untuk informasi selengkapnya tentang menulis panda UDF lihat: Dokumentasi Apache Spark SQL.

```
from awsglue import DynamicFrame
import pandas as pd
from pyspark.sql.functions import pandas_udf
```

```
# The number and name of arguments must match the definition on json config file
# (expect self which is the current DynamicFrame to transform
# If an argument is optional, you need to define a default value here
#  (resultCol in this example is an optional argument)
def custom_add_columns(self, col1, col2, resultCol="result"):
    @pandas_udf("integer")  # We need to declare the type of the result column
    def add_columns(value1: pd.Series, value2: pd.Series) # pd.Series:
        return value1 + value2

    df = self.toDF()
    return DynamicFrame.fromDF(
        df.withColumn(resultCol, add_columns(col1, col2)) # This is the conversion
 logic
        , self.glue_ctx, self.name)

# The name we assign on DynamicFrame must match the configured "functionName"
DynamicFrame.custom_add_columns = custom_add_columns
```

Video

Video berikut memberikan pengantar transformasi kustom visual dan menunjukkan cara menggunakannya.

Menggunakan kerangka Data Lake dengan AWS Glue Studio

Gambaran Umum

Kerangka kerja data lake sumber terbuka menyederhanakan pemrosesan data tambahan untuk file yang disimpan di danau data yang dibangun di Amazon S3. AWS Glue3.0 dan yang lebih baru mendukung kerangka kerja penyimpanan data lake sumber terbuka berikut:

- Apache Hudi
- Yayasan Linux Delta Lake
- Gunung Es Apache

Mulai AWS Glue 4.0, AWS Glue menyediakan dukungan asli untuk kerangka kerja ini sehingga Anda dapat membaca dan menulis data yang Anda simpan di Amazon S3 dengan cara yang konsisten secara transaksional. Tidak perlu menginstal konektor terpisah atau menyelesaikan langkah-langkah konfigurasi tambahan untuk menggunakan kerangka kerja ini dalam AWS Glue pekerjaan.

Framework Data Lake dapat digunakan sebagai sumber atau target di dalam AWS Glue Studio melalui pekerjaan Spark Script Editor. Untuk informasi lebih lanjut tentang menggunakan Apache Hudi, Apache Iceberg dan Delta Lake lihat: Menggunakan kerangka data lake dengan pekerjaan ETL. AWS Glue

Membuat format tabel terbuka dari sumber AWS Glue Streaming

AWS Gluestreaming pekerjaan ETL terus mengkonsumsi data dari sumber streaming, membersihkan dan mengubah data dalam penerbangan, dan membuatnya tersedia untuk analisis dalam hitungan detik.

AWSmenawarkan berbagai pilihan layanan untuk mendukung kebutuhan Anda. Layanan replikasi AWS database seperti Database Migration Service dapat mereplikasi data dari sistem sumber Anda ke Amazon S3, yang biasanya menampung lapisan penyimpanan data lake. Meskipun mudah untuk menerapkan pembaruan pada sistem manajemen basis data relasional (RDBMS) yang mendukung aplikasi sumber online, sulit untuk menerapkan proses CDC ini di danau data Anda. Kerangka kerja manajemen data sumber terbuka menyederhanakan pemrosesan data tambahan dan pengembangan pipa data, dan merupakan pilihan yang baik untuk memecahkan masalah ini.

Untuk informasi selengkapnya, lihat:

- Buat danau data near-real-time transaksional berbasis Apache Hudi menggunakan Streaming AWS Glue
- Bangun danau data Apache Iceberg yang selaras dengan GDPR secara real-time

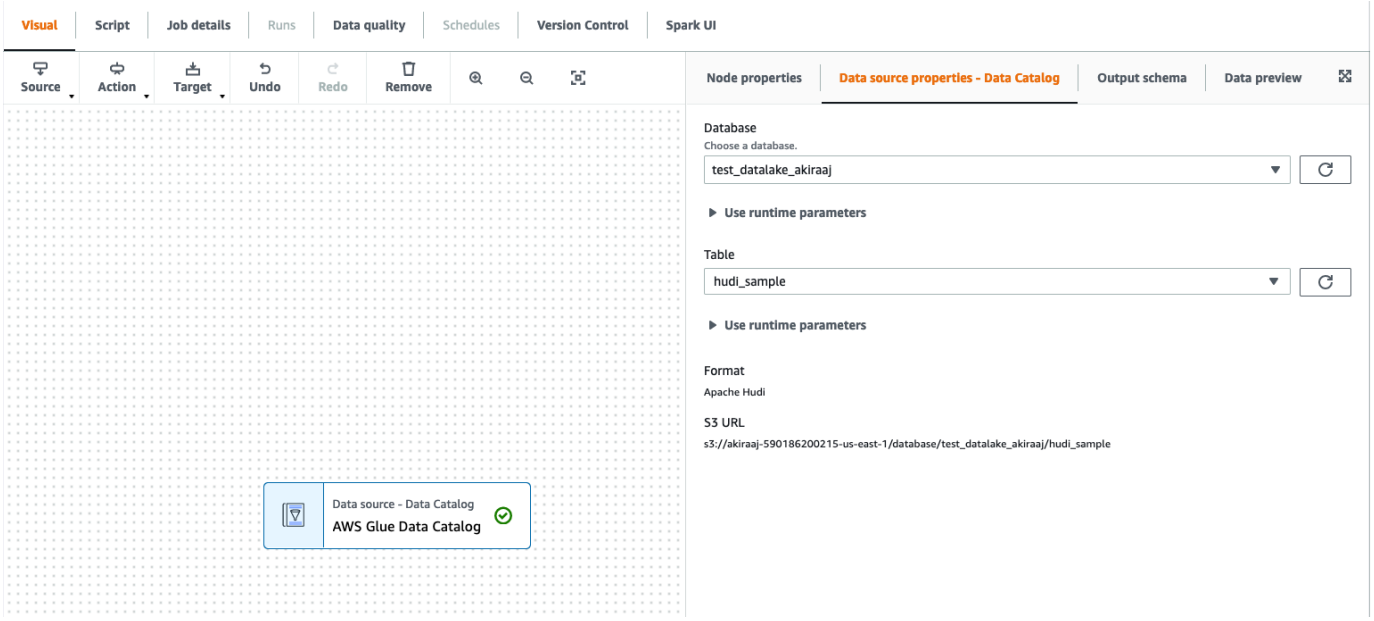
Menggunakan kerangka Hudi di AWS Glue Studio

Saat membuat atau mengedit pekerjaan, AWS Glue Studio secara otomatis menambahkan pustaka Hudi yang sesuai untuk Anda tergantung pada versi yang AWS Glue Anda gunakan. Untuk informasi selengkapnya, lihat Menggunakan kerangka kerja Hudi di. AWS Glue

Menggunakan kerangka Apache Hudi di sumber data Katalog Data

Untuk menambahkan format sumber data Hudi ke pekerjaan:

1. Dari menu Sumber, pilih Katalog AWS Glue Studio Data.
2. Di tab Properti sumber data, pilih database dan tabel.
3. AWS Glue Studiomenampilkan jenis format sebagai Apache Hudi dan URL Amazon S3.

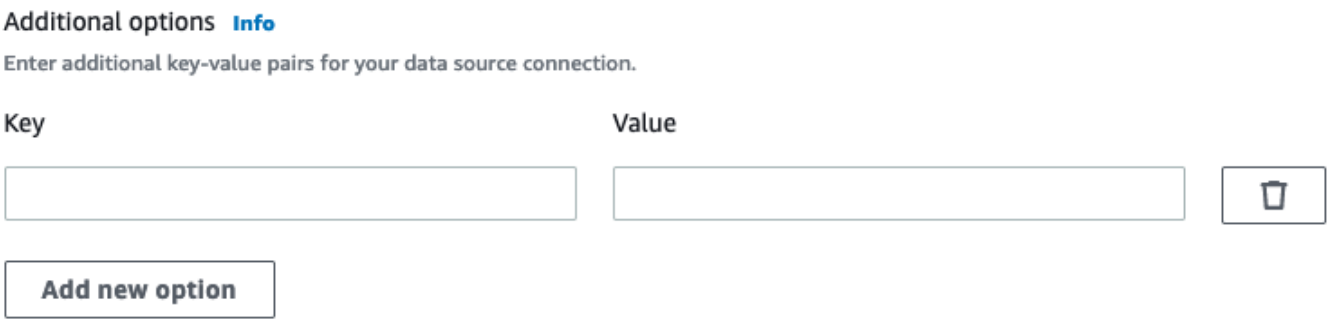


Menggunakan kerangka kerja Hudi di sumber data Amazon S3

1. Dari menu Sumber, pilihAmazon S3.
2. Jika Anda memilih tabel Katalog Data sebagai jenis sumber Amazon S3, pilih database dan tabel.
3. AWS Glue Studiomenampilkan format sebagai Apache Hudi dan URL Amazon S3.
4. Jika Anda memilih lokasi Amazon S3 sebagai jenis sumber Amazon S3, pilih URL Amazon S3 dengan mengklik Jelajahi Amazon S3.
5. Dalam format Data, pilih Apache Hudi.

> **Note**
>
> Jika AWS Glue Studio tidak dapat menyimpulkan skema dari folder Amazon S3 atau file yang Anda pilih, pilih Opsi tambahan untuk memilih folder atau file baru.
> Dalam Opsi tambahan pilih dari opsi berikut di bawah Inferensi skema:
>
> - Biarkan AWS Glue Studio secara otomatis memilih file sampel - AWS Glue Studio akan memilih file sampel di lokasi Amazon S3 sehingga skema dapat disimpulkan. Di bidang File sampel otomatis, Anda dapat melihat file yang dipilih secara otomatis.
> - Pilih file sampel dari Amazon S3 - pilih file Amazon S3 yang akan digunakan dengan mengklik Jelajahi Amazon S3.

6. Klik Skema Infer. Anda kemudian dapat melihat skema output dengan mengklik Skema keluaran tab.
7. Pilih Opsi tambahan untuk memasukkan pasangan kunci-nilai.



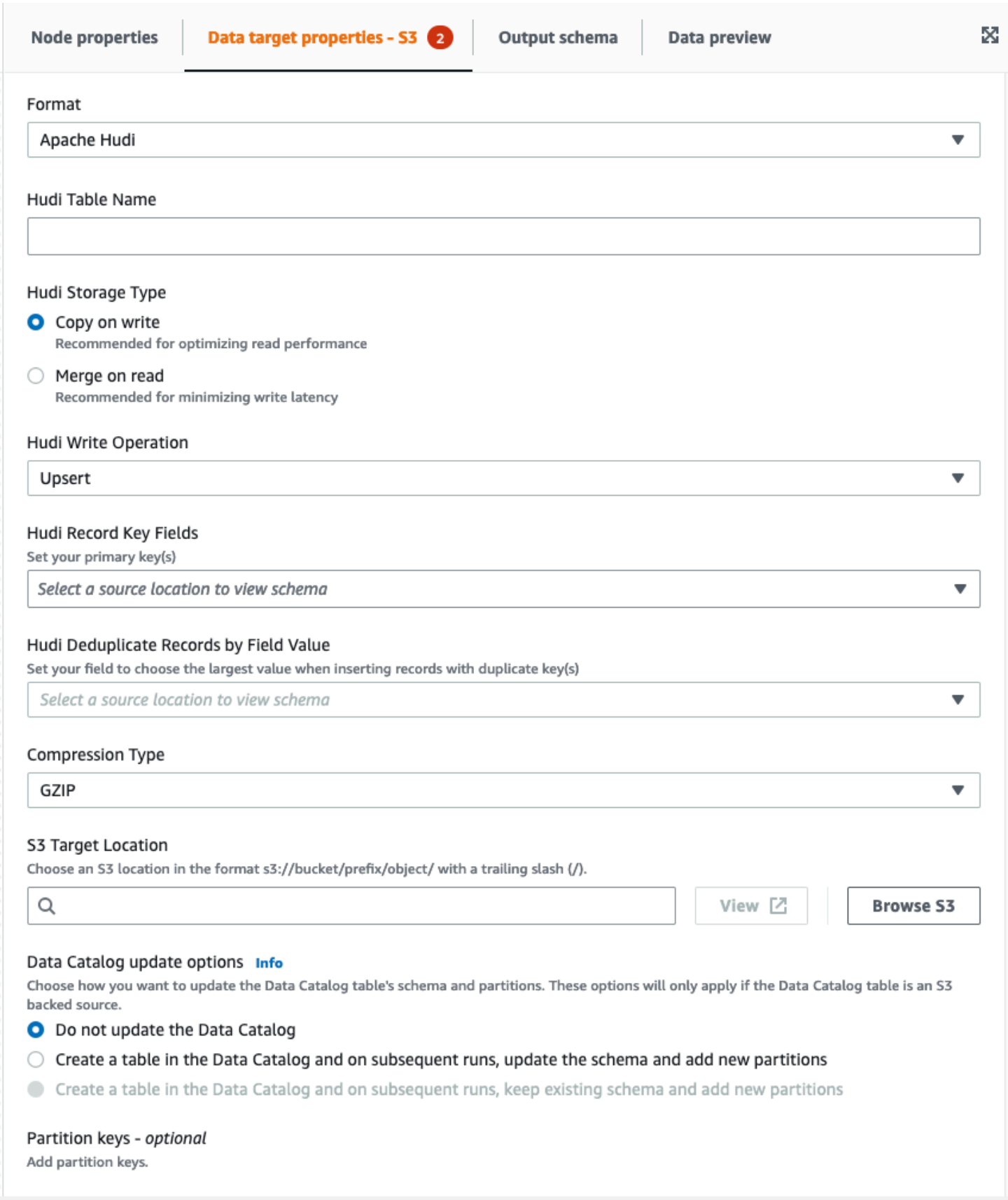
Menggunakan kerangka Apache Hudi dalam target data

Menggunakan kerangka Apache Hudi dalam target data Katalog Data

1. Dari menu Target, pilih Katalog AWS Glue Studio Data.
2. Di tab Properti sumber data, pilih database dan tabel.
3. AWS Glue Studiomenampilkan jenis format sebagai Apache Hudi dan URL Amazon S3.

Menggunakan kerangka Apache Hudi di target data Amazon S3

Masukkan nilai atau pilih dari opsi yang tersedia untuk mengonfigurasi format Apache Hudi. Untuk informasi lebih lanjut tentang Apache Hudi, lihat dokumentasi Apache Hudi.


Node properties
Data target properties - S3 2
Output schema
Data preview
Format
Apache Hudi
Hudi Table Name
Hudi Storage Type
Copy on write
Recommended for optimizing read performance
Merge on read
Recommended for minimizing write latency
Hudi Write Operation
Upsert
Hudi Record Key Fields
Set your primary key(s)
Select a source location to view schema
Hudi Deduplicate Records by Field Value
Set your field to choose the largest value when inserting records with duplicate key(s)
Select a source location to view schema
Compression Type
GZIP
S3 Target Location
Choose an S3 location in the format s3://bucket/prefix/object/ with a trailing slash (/).
View
Browse S3
Data Catalog update options Info
Choose how you want to update the Data Catalog table's schema and partitions. These options will only apply if the Data Catalog table is an S3 backed source.
Do not update the Data Catalog
Create a table in the Data Catalog and on subsequent runs, update the schema and add new partitions
Create a table in the Data Catalog and on subsequent runs, keep existing schema and add new partitions
Partition keys - optional
Add partition keys.

- Nama Tabel Hudi — ini adalah nama tabel hudi Anda.
- Hudi Storage Type - pilih dari dua opsi:
  - Salin saat menulis - direkomendasikan untuk mengoptimalkan kinerja baca. Ini adalah tipe penyimpanan Hudi default. Setiap pembaruan membuat versi file baru selama penulisan.
  - Gabungkan saat dibaca — disarankan untuk meminimalkan latensi tulis. Pembaruan dicatat ke file delta berbasis baris dan dipadatkan sesuai kebutuhan untuk membuat file kolumnar versi baru.
- Operasi Tulis Hudi - pilih dari opsi berikut:
  - Upsert - ini adalah operasi default di mana catatan input pertama kali ditandai sebagai sisipan atau pembaruan dengan mencari indeks. Disarankan di mana Anda memperbarui data yang ada.
  - Sisipkan — ini menyisipkan catatan tetapi tidak memeriksa catatan yang ada dan dapat mengakibatkan duplikat.
  - Sisipan Massal — ini menyisipkan catatan dan direkomendasikan untuk sejumlah besar data.
- Hudi Record Key Fields — gunakan bilah pencarian untuk mencari dan memilih kunci rekam utama. Catatan dalam Hudi diidentifikasi oleh kunci utama yang merupakan sepasang kunci catatan dan jalur partisi tempat catatan berada.
- Hudi Precombine Field - ini adalah bidang yang digunakan dalam PreCombining sebelum menulis aktual. Ketika dua record memiliki nilai kunci yang sama, AWS Glue Studio akan memilih satu dengan nilai terbesar untuk bidang precombine. Tetapkan bidang dengan nilai tambahan (misalnya updated_at) milik.
- Jenis Kompresi - pilih dari salah satu opsi jenis kompresi: Tidak Terkompresi, GZIP, LZO, atau Snappy.
- Lokasi Target Amazon S3 - pilih lokasi target Amazon S3 dengan mengklik Jelajahi S3.
- Opsi pembaruan Katalog Data - pilih dari opsi berikut:
  - Jangan memutakhirkan Katalog Data: (Default) Pilih opsi ini jika Anda tidak ingin tugas memperbarui Katalog Data, bahkan jika skema berubah atau partisi baru ditambahkan.
  - Buat tabel di Katalog Data dan pada proses berikutnya, perbarui skema dan tambahkan partisi baru: Jika Anda memilih opsi ini, pekerjaan membuat tabel di Katalog Data pada proses pertama pekerjaan. Pada eksekusi tugas berikutnya, tugas memutakhirkan tabel Katalog Data jika skema berubah atau partisi baru ditambahkan.

    Anda juga harus memilih sebuah basis data dari Katalog Data dan memasukkan nama tabel.

- Membuat tabel di Katalog Data dan eksekusi berikutnya, mempertahankan skema yang ada dan menambahkan partisi baru: Jika Anda memilih opsi ini, maka tugas akan menciptakan tabel di Katalog Data pada eksekusi pertama tugas. Pada eksekusi tugas berikutnya, tugas memutakhirkan tabel Katalog Data hanya jika partisi baru ditambahkan.

  Anda juga harus memilih sebuah basis data dari Katalog Data dan memasukkan nama tabel.

- Kunci partisi: Pilih kolom mana yang digunakan sebagai kunci partisi dalam output. Untuk menambahkan lebih banyak kunci partisi, pilih Tambahkan kunci partisi.
- Opsi tambahan - masukkan pasangan kunci-nilai sesuai kebutuhan.

Menghasilkan kode melalui AWS Glue Studio

Ketika pekerjaan disimpan, parameter pekerjaan berikut ditambahkan ke pekerjaan jika sumber atau target Hudi terdeteksi:

- `--datalake-formats`— daftar format data lake yang berbeda terdeteksi dalam pekerjaan visual (baik secara langsung dengan memilih “Format” atau secara tidak langsung dengan memilih tabel katalog yang didukung oleh danau data).
- `--conf` — dihasilkan berdasarkan nilai`--datalake-formats`. Misalnya, jika nilai untuk `--datalake-formats` adalah 'hudi', AWS Glue menghasilkan nilai `spark.serializer=org.apache.spark.serializer.KryoSerializer –conf spark.sql.hive.convertMetastoreParquet=false` untuk parameter ini.

Mengesampingkan pustaka AWS Glue yang disediakan

Untuk menggunakan versi Hudi yang AWS Glue tidak mendukung, Anda dapat menentukan file JAR perpustakaan Hudi Anda sendiri. Untuk menggunakan file JAR Anda sendiri:

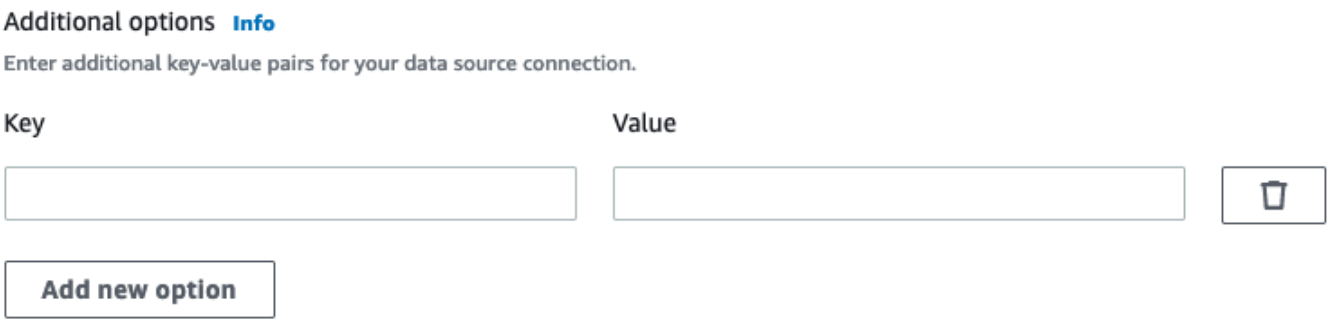
- gunakan parameter `--extra-jars` pekerjaan. Sebagai contoh, `'--extra-jars': 's3pathtojarfile.jar'`. Untuk informasi selengkapnya, lihat parameter AWS Glue pekerjaan.
- tidak termasuk `hudi` sebagai nilai untuk parameter `--datalake-formats` pekerjaan. Memasukkan string kosong sebagai nilai memastikan bahwa tidak ada pustaka data lake yang disediakan untuk Anda secara AWS Glue otomatis. Untuk informasi selengkapnya, lihat Menggunakan kerangka kerja Hudi di. AWS Glue

Menggunakan kerangka Delta Lake di AWS Glue Studio

Menggunakan kerangka Delta Lake dalam sumber data

Menggunakan kerangka Delta Lake di sumber data Amazon S3

1. Dari menu Sumber, pilihAmazon S3.
2. Jika Anda memilih tabel Katalog Data sebagai jenis sumber Amazon S3, pilih database dan tabel.
3. AWS Glue Studiomenampilkan format sebagai Delta Lake dan URL Amazon S3.
4. Pilih Opsi tambahan untuk memasukkan pasangan kunci-nilai. Misalnya, pasangan kunci-nilai dapat berupa: key: timestampAsOf dan value: 2023-02-24 14:16:18.



5. Jika Anda memilih lokasi Amazon S3 sebagai jenis sumber Amazon S3, pilih URL Amazon S3 dengan mengklik Jelajahi Amazon S3.
6. Dalam format Data, pilih Delta Lake.

> **Note**
>
> Jika AWS Glue Studio tidak dapat menyimpulkan skema dari folder Amazon S3 atau file yang Anda pilih, pilih Opsi tambahan untuk memilih folder atau file baru.
> Dalam Opsi tambahan pilih dari opsi berikut di bawah Inferensi skema:
>
> - Biarkan AWS Glue Studio secara otomatis memilih file sampel - AWS Glue Studio akan memilih file sampel di lokasi Amazon S3 sehingga skema dapat disimpulkan. Di bidang File sampel otomatis, Anda dapat melihat file yang dipilih secara otomatis.
> - Pilih file sampel dari Amazon S3 - pilih file Amazon S3 yang akan digunakan dengan mengklik Jelajahi Amazon S3.

7. Klik Skema Infer. Anda kemudian dapat melihat skema output dengan mengklik Skema keluaran tab.

Menggunakan kerangka Delta Lake di sumber data Katalog Data

1. Dari menu Sumber, pilih Katalog AWS Glue Studio Data.
2. Di tab Properti sumber data, pilih database dan tabel.
3. AWS Glue Studiomenampilkan jenis format sebagai Delta Lake dan URL Amazon S3.

> **Note**
>
> Jika sumber Delta Lake Anda belum terdaftar sebagai tabel Katalog AWS Glue Data, Anda memiliki dua opsi:
>
> 1. Buat AWS Glue crawler untuk penyimpanan data Delta Lake. Untuk informasi selengkapnya, lihat Cara menentukan opsi konfigurasi untuk penyimpanan data Delta Lake.
> 2. Gunakan sumber data Amazon S3 untuk memilih sumber data Delta Lake Anda. Lihat Menggunakan kerangka Delta Lake di sumber data Amazon S3 .

Menggunakan format Delta Lake dalam target data

Menggunakan format Delta Lake dalam target data Katalog Data

1. Dari menu Target, pilih Katalog AWS Glue Studio Data.
2. Di tab Properti sumber data, pilih database dan tabel.
3. AWS Glue Studiomenampilkan jenis format sebagai Delta Lake dan URL Amazon S3.

Menggunakan format Delta Lake di sumber data Amazon S3

Masukkan nilai atau pilih dari opsi yang tersedia untuk mengkonfigurasi format Delta Lake.

- Jenis Kompresi - pilih dari salah satu opsi jenis kompresi: Tidak Terkompresi atau Snappy.
- Lokasi Target Amazon S3 - pilih lokasi target Amazon S3 dengan mengklik Jelajahi S3.
- Opsi pembaruan Katalog Data — memperbarui Katalog Data tidak didukung untuk format ini di editor visual Glue Studio.

  - Jangan memutakhirkan Katalog Data: (Default) Pilih opsi ini jika Anda tidak ingin tugas memperbarui Katalog Data, bahkan jika skema berubah atau partisi baru ditambahkan.
  - Untuk memperbarui Katalog Data setelah eksekusi AWS Glue pekerjaan, jalankan atau jadwalkan AWS Glue crawler. Untuk informasi selengkapnya, lihat Cara menentukan opsi konfigurasi untuk penyimpanan data Delta Lake.
- Tombol partisi - Pilih kolom mana yang akan digunakan sebagai kunci partisi dalam output. Untuk menambahkan lebih banyak kunci partisi, pilih Tambahkan kunci partisi.
- Secara opsional, pilih Opsi tambahan untuk memasukkan pasangan kunci-nilai. Misalnya, pasangan kunci-nilai dapat berupa: key: timestampAsOf dan value: 2023-02-24 14:16:18.

Menggunakan kerangka Apache Iceberg di AWS Glue Studio

Menggunakan kerangka Apache Iceberg dalam target data

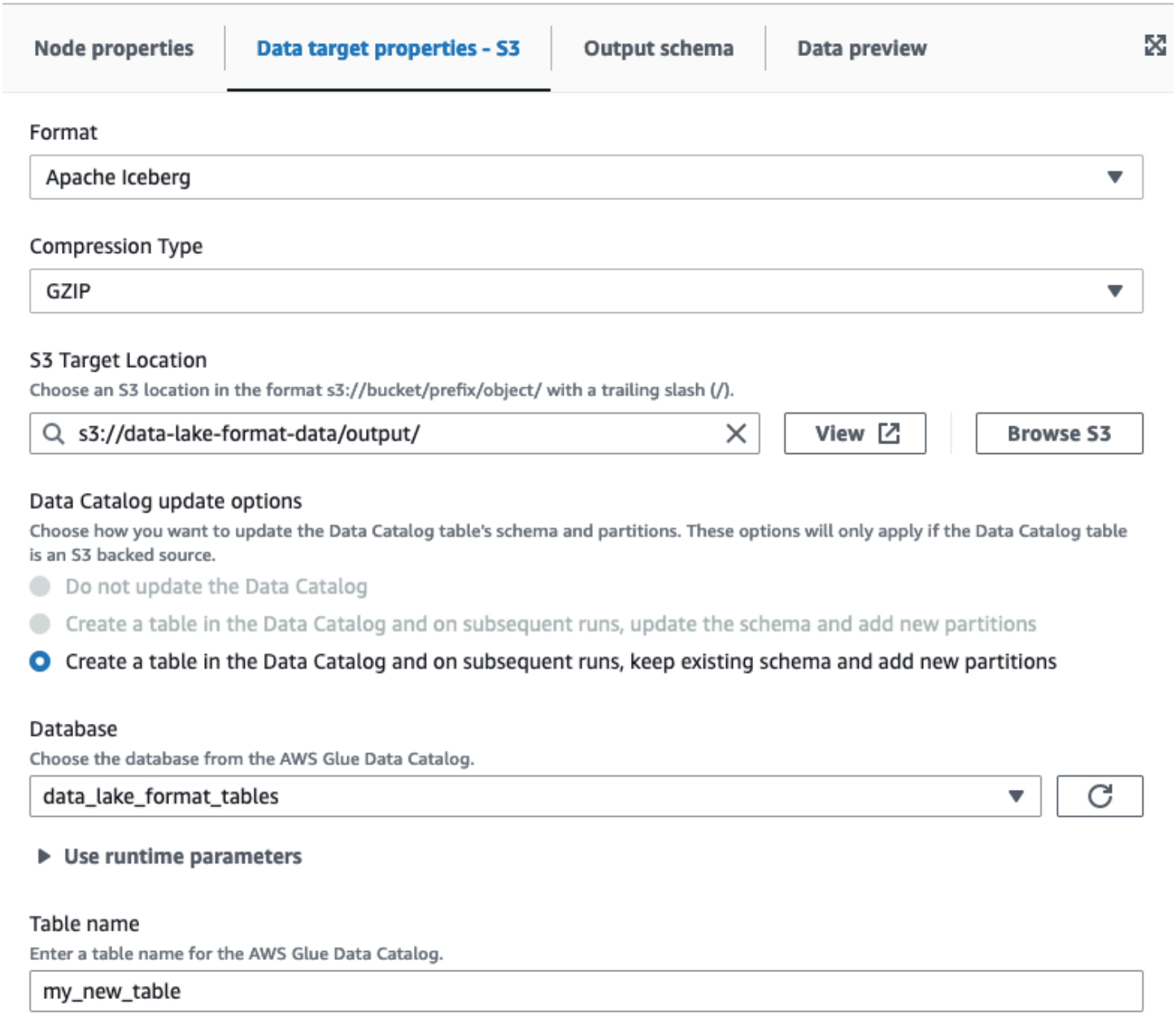
Menggunakan kerangka Apache Iceberg dalam target data Katalog Data

1. Dari menu Target, pilih Katalog AWS Glue Studio Data.
2. Di tab Properti sumber data, pilih database dan tabel.
3. AWS Glue Studiomenampilkan jenis format sebagai Apache Iceberg dan URL Amazon S3.

Menggunakan kerangka Apache Iceberg di target data Amazon S3

Masukkan nilai atau pilih dari opsi yang tersedia untuk mengonfigurasi format Apache Iceberg.

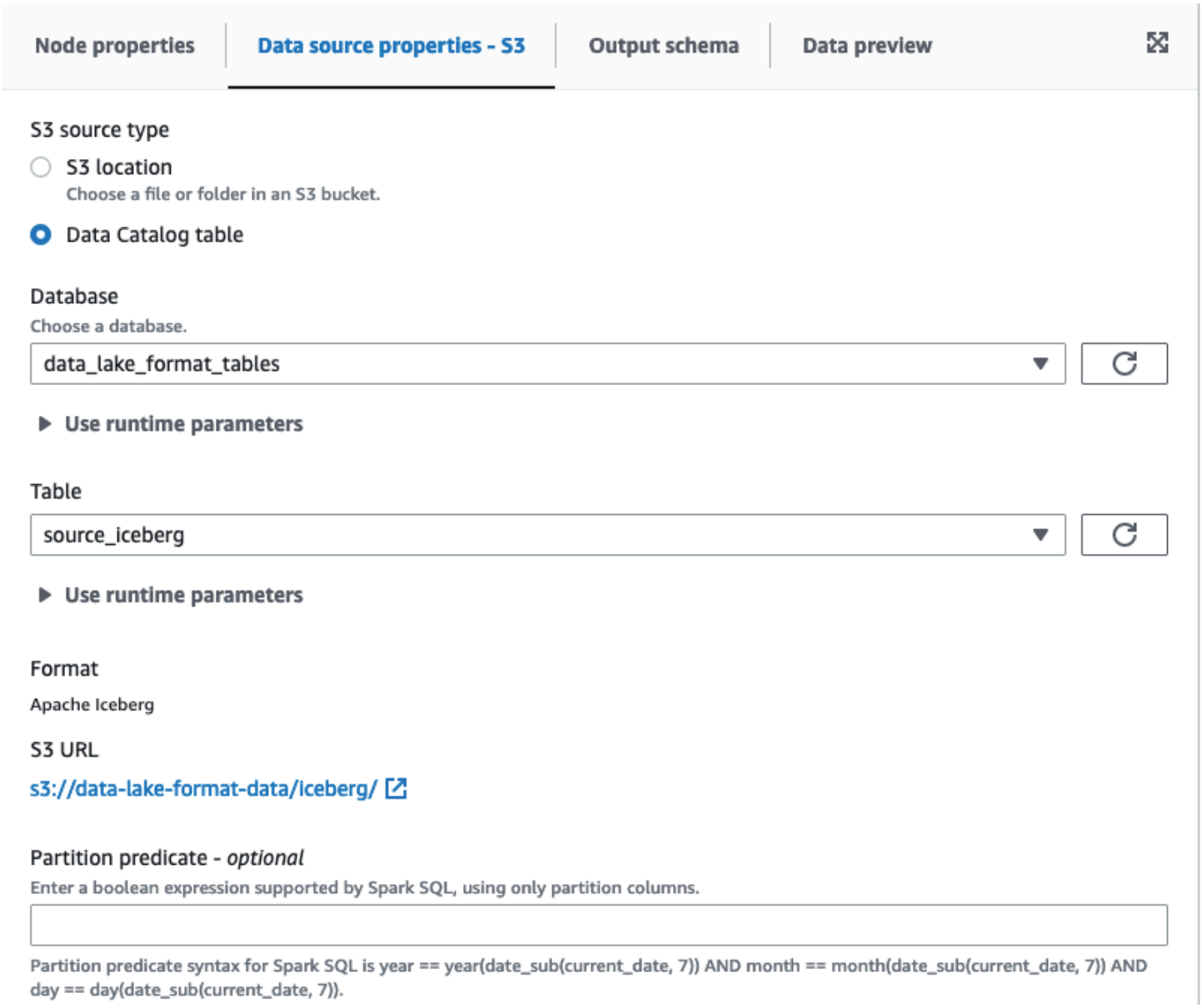
- Format - pilih Apache Iceberg dari menu drop-down.
- Lokasi Target Amazon S3 - pilih lokasi target Amazon S3 dengan mengklik Jelajahi S3.
- Opsi pembaruan Katalog Data - Buat tabel di Katalog Data dan pada proses selanjutnya, pertahankan skema yang ada dan tambahkan partisi baru harus dipilih untuk melanjutkan. Menulis tabel Iceberg baru menggunakan AWS Glue mengharuskan Data Catalog untuk dikonfigurasi sebagai katalog untuk tabel Iceberg. Untuk memperbarui tabel Gunung Es yang ada yang telah terdaftar diData Catalog, pilih Data Catalog sebagai target.
  - Database — Pilih database dariData Catalog.
  - Nama Tabel - Masukkan nilai untuk nama tabel Anda. Nama tabel Apache Iceberg harus dalam semua huruf kecil. Gunakan garis bawah jika diperlukan karena spasi tidak diperbolehkan. Misalnya “data_lake_format_tables”.



Menggunakan kerangka Apache Iceberg di sumber data Amazon S3

Menggunakan kerangka Apache Iceberg di sumber data Katalog Data

1. Dari menu Sumber, pilih Katalog AWS Glue Studio Data.
2. Di tab Properti sumber data, pilih database dan tabel.
3. AWS Glue Studiomenampilkan jenis format sebagai Apache Iceberg dan URL Amazon S3.



Menggunakan kerangka Apache Iceberg di sumber data Amazon S3

Apache Iceberg tidak tersedia sebagai opsi data untuk node sumber Amazon S3 di. AWS Glue Studio

Mengkonfigurasi simpul target data

Target data adalah tempat di mana tugas menulis data yang sudah ditransformasi.

Gambaran umum pilihan target data

Target data Anda (juga disebut sink data) dapat berupa:

- S3 — Tugas menulis data dalam sebuah file di lokasi Amazon S3 yang Anda pilih dan dalam format yang Anda tentukan.

  Jika Anda mengkonfigurasi kolom partisi untuk target data, maka tugas akan menulis set data ke Amazon S3 ke direktori berdasarkan kunci partisi.
- AWS Glue Data Catalog — Tugas menggunakan informasi yang dikaitkan dengan tabel di Katalog Data untuk menulis data output ke sebuah lokasi target.

  Anda dapat membuat tabel secara manual atau dengan crawler. Anda juga dapat menggunakan templat AWS CloudFormation untuk membuat tabel dalam Katalog Data.
- Konektor — Sebuah konektor adalah bagian dari kode yang memudahkan komunikasi antara penyimpanan data Anda dan AWS Glue. Tugas menggunakan konektor dan koneksi terkait untuk menulis data output ke sebuah lokasi target. Anda dapat berlangganan konektor yang ditawarkan di AWS Marketplace atau Anda dapat membuat konektor kustom Anda sendiri. Untuk informasi selengkapnya, lihat Menambahkan konektor ke AWS Glue Studio

Anda dapat memilih untuk memperbarui Katalog Data ketika tugas Anda menulis ke sebuah target data Amazon S3. Alih-alih mengharuskan sebuah crawler untuk memperbarui Katalog Data ketika skema atau partisi berubah, opsi ini memudahkan untuk menjaga tabel Anda selalu diperbarui. Opsi ini menyederhanakan proses membuat data Anda tersedia untuk analitik dengan secara opsional menambahkan tabel baru ke Katalog Data, memperbarui partisi tabel, dan memperbarui skema tabel Anda secara langsung dari tugas.

Mengedit simpul target data

Target data adalah tempat di mana tugas menulis data yang sudah ditransformasi.

Untuk menambah atau mengkonfigurasi sebuah simpul target data dalam diagram tugas Anda

1. (Opsional) Jika Anda perlu menambahkan sebuah simpul target, pilih Target di bilah alat yang ada di bagian atas editor visual, lalu pilih salah satu, S3 atau Katalog Data Glue.

   - Jika Anda memilih S3 untuk target, maka tugas akan menulis set data ke satu atau beberapa file di lokasi Amazon S3 yang Anda tentukan.
   - Jika Anda memilih AWS Glue Data Catalog untuk target, maka tugas akan menulis ke lokasi yang dijelaskan oleh tabel yang dipilih dari Katalog Data.
2. Pilih sebuah simpul target data dalam diagram tugas. Bila Anda memilih sebuah simpul, maka panel detail simpul akan muncul di sisi kanan halaman.

3. Pilih tab Properti simpul, dan kemudian masukkan informasi berikut:

   - Nama: Masukkan nama yang akan dikaitkan dengan simpul dalam diagram tugas.
   - Jenis Simpul: Sebuah nilai harus sudah dipilih, tetapi Anda dapat mengubahnya sesuai kebutuhan.
   - Induk simpul: Induk simpul adalah simpul dalam diagram tugas yang menyediakan output data yang ingin Anda tulis ke lokasi target. Untuk diagram tugas yang sudah diisi sebelumnya, simpul target harus sudah memiliki simpul induk yang dipilih. Jika tidak ada simpul induk yang ditampilkan, maka pilih simpul induk dari daftar.

     Sebuah simpul target memiliki satu simpul induk tunggal.
4. Mengkonfigurasi informasi Properti target data. Untuk informasi selengkapnya, lihat bagian berikut:

   - Menggunakan Amazon S3 untuk target data
   - Menggunakan tabel Katalog Data untuk target data
   - Menggunakan sebuah konektor untuk target data
5. (Opsional) Setelah mengkonfigurasi properti simpul target data, Anda dapat melihat skema output untuk data Anda dengan memilih tab Skema output di panel detail simpul. Pertama kali Anda memilih tab ini untuk setiap simpul dalam tugas Anda, Anda akan diminta untuk memberikan IAM role untuk mengakses data. Jika Anda belum menentukan IAM role pada tab Detail tugas, maka Anda akan diminta untuk memasukkan IAM role di sini.

Menggunakan Amazon S3 untuk target data

Untuk semua sumber data kecuali Amazon S3 dan konektor, tabel harus ada di AWS Glue Data Catalog untuk jenis sumber yang Anda pilih. AWS Glue Studiotidak membuat tabel Katalog Data.

Untuk mengkonfigurasi simpul target data yang menulis ke Amazon S3

1. Pergi ke editor visual untuk sebuah tugas baru atau yang sudah disimpan.
2. Pilih simpul sumber data dalam diagram tugas.
3. Pilih tab Properti sumber data, dan kemudian masukkan informasi berikut:

   - Format: Pilih format dari daftar. Jenis format yang tersedia untuk hasil data adalah:
     - JSON: Notasi JavaScript Objek.
     - CSV: Nilai yang dipisahkan koma.

- Avro: Biner JSON Apache Avro.
- Parquet: Penyimpanan kolumnar Apache Parquet.
- Glue Parquet: Jenis penulis Parquet kustom yang dioptimalkan untuk `DynamicFrames` sebagai format data. Alih-alih mengharuskan skema yang sudah dikomputasi sebelumnya untuk data, ia melakukan komputasi dan modifikasi pada skema secara dinamis.
- ORC: Format Optimized Row Columnar (ORC) Apache.

Untuk mempelajari lebih lanjut tentang opsi format ini, lihat Format Pilihan untuk Input dan Output ETL di AWS Glue dalam Panduan Developer AWS Glue.

- Jenis Kompresi: Anda dapat memilih untuk secara opsional mengkompres data menggunakan format `gzip` atau `bzip2`. Default-nya adalah tidak dikompresi, atau Tidak ada.
- Lokasi Target S3: Bucket Amazon S3 dan lokasi untuk output data. Anda dapat memilih tombol Jelajahi S3 untuk melihat bucket Amazon S3 yang Anda miliki aksesnya dan memilih salah satu sebagai tujuan target.
- Opsi pembaruan katalog data
  - Jangan memutakhirkan Katalog Data: (Default) Pilih opsi ini jika Anda tidak ingin tugas memperbarui Katalog Data, bahkan jika skema berubah atau partisi baru ditambahkan.
  - Membuat tabel di Katalog Data dan eksekusi berikutnya, memperbarui skema dan menambahkan partisi baru: Jika Anda memilih opsi ini, maka tugas akan menciptakan tabel di Katalog Data pada eksekusi pertama tugas. Pada eksekusi tugas berikutnya, tugas memutakhirkan tabel Katalog Data jika skema berubah atau partisi baru ditambahkan.

    Anda juga harus memilih sebuah basis data dari Katalog Data dan memasukkan nama tabel.
  - Membuat tabel di Katalog Data dan eksekusi berikutnya, mempertahankan skema yang ada dan menambahkan partisi baru: Jika Anda memilih opsi ini, maka tugas akan menciptakan tabel di Katalog Data pada eksekusi pertama tugas. Pada eksekusi tugas berikutnya, tugas memutakhirkan tabel Katalog Data hanya jika partisi baru ditambahkan.

    Anda juga harus memilih sebuah basis data dari Katalog Data dan memasukkan nama tabel.
  - Kunci partisi: Pilih kolom mana yang digunakan sebagai kunci partisi dalam output. Untuk menambahkan lebih banyak kunci partisi, pilih Tambahkan kunci partisi.

Menggunakan tabel Katalog Data untuk target data

Untuk semua sumber data kecuali Amazon S3 dan konektor, tabel harus ada di AWS Glue Data Catalog untuk jenis target yang Anda pilih. AWS Glue Studiotidak membuat tabel Katalog Data.

Untuk mengkonfigurasi properti data untuk target yang menggunakan tabel Katalog Data

1. Pergi ke editor visual untuk sebuah tugas baru atau yang sudah disimpan.
2. Pilih sebuah simpul target data dalam diagram tugas.
3. Pilih tab Properti target data, dan kemudian masukkan informasi berikut:

   - Basis data: Pilih basis data yang berisi tabel yang ingin Anda gunakan sebagai target dari daftar. Basis data ini harus sudah ada dalam Katalog Data.
   - Tabel: Pilih tabel yang mendefinisikan skema data output Anda dari daftar. Tabel ini sudah harus ada dalam Katalog Data.

     Sebuah tabel dalam Katalog Data terdiri dari nama-nama kolom, definisi tipe data, informasi partisi, dan metadata lainnya tentang set data target. Tugas Anda menulis ke sebuah lokasi yang dijelaskan oleh tabel ini dalam Katalog Data.

     Untuk informasi selengkapnya tentang membuat tabel dalam Katalog Data, lihat Mendefinisikan Tabel dalam Katalog Data dalam Panduan Developer AWS Glue.
   - Opsi pembaruan katalog data
     - Jangan ubah definisi tabel: (Default) Pilih opsi ini jika Anda tidak ingin tugas memperbarui Katalog Data, bahkan jika skema berubah, atau partisi baru ditambahkan.
     - Memperbarui skema dan menambahkan partisi baru: Jika Anda memilih opsi ini, maka tugas akan memperbarui tabel Katalog Data jika skema berubah atau partisi baru ditambahkan.
     - Pertahankan skema yang ada dan tambahkan partisi baru: Jika Anda memilih opsi ini, maka tugas akan memperbarui tabel Katalog Data hanya untuk menambahkan partisi baru.
     - Kunci partisi: Pilih kolom mana yang digunakan sebagai kunci partisi dalam output. Untuk menambahkan lebih banyak kunci partisi, pilih Tambahkan kunci partisi.

Menggunakan sebuah konektor untuk target data

Jika Anda memilih sebuah konektor untuk Jenis Simpul, ikuti petunjuk di Menulis tugas dengan konektor kustom untuk menyelesaikan konfigurasi properti target data.

Mengedit atau mengunggah sebuah skrip tugas

Gunakan editor AWS Glue Studio visual untuk mengedit skrip pekerjaan atau mengunggah skrip Anda sendiri.

Anda dapat menggunakan editor visual untuk mengedit node pekerjaan hanya jika pekerjaan dibuat denganAWS Glue Studio. Jika pekerjaan dibuat menggunakan AWS Glue konsol, melalui perintah API, atau dengan antarmuka baris perintah (CLI), Anda dapat menggunakan editor skrip AWS Glue Studio untuk mengedit skrip pekerjaan, parameter, dan jadwal. Anda juga dapat mengedit skrip untuk pekerjaan yang dibuat AWS Glue Studio dengan mengonversi pekerjaan ke mode khusus skrip.

Untuk mengedit skrip tugas atau mengunggah skrip Anda sendiri

1. Jika membuat pekerjaan baru, pada halaman Jobs, pilih opsi editor skrip Spark untuk membuat pekerjaan Spark atau pilih editor skrip Python Shell untuk membuat pekerjaan shell Python. Anda dapat menulis skrip baru, atau mengunggah skrip yang ada. Jika Anda memilih editor skrip Spark, Anda dapat menulis atau mengunggah skrip Scala atau Python. Jika Anda memilih editor skrip Python Shell, Anda hanya dapat menulis atau mengunggah skrip Python.

   Setelah memilih opsi untuk membuat pekerjaan baru, di bagian Opsi yang muncul, Anda dapat memilih untuk memulai dengan skrip pemula (Buat skrip baru dengan kode boilerplate), atau Anda dapat mengunggah file lokal untuk digunakan sebagai skrip pekerjaan.

   Jika Anda memilih editor skrip Spark, Anda dapat mengunggah file skrip Python atau Scala. Skrip Scala harus memiliki ekstensi file `.scala`. Skrip Python harus diakui sebagai tipe file Python. Jika Anda memilih editor skrip Python Shell, Anda hanya dapat mengunggah file skrip Python.

   Setelah selesai membuat pilihan, pilih Buat untuk membuat pekerjaan dan buka editor visual.
2. Pergi ke editor tugas visual untuk tugas baru atau yang sudah disimpan, dan kemudian pilih tab Skrip.
3. Jika Anda tidak membuat pekerjaan baru menggunakan salah satu opsi editor skrip, dan Anda belum pernah mengedit skrip untuk pekerjaan yang ada, tab Script menampilkan judul Script (Terkunci). Ini artinya editor skrip dalam mode baca-saja. Pilih edit skrip untuk membuka kunci skrip untuk mengedit.

   Untuk membuat skrip dapat diedit, AWS Glue Studio konversi pekerjaan Anda dari pekerjaan visual ke pekerjaan skrip saja. Jika Anda membuka kunci skrip untuk diedit, Anda tidak dapat menggunakan editor visual lagi untuk pekerjaan ini setelah Anda menyimpannya.

Di jendela konfirmasi, pilih Konfirmasi untuk melanjutkan atau Batalkan untuk menjaga agar tugas tetap tersedia untuk pengeditan visual.

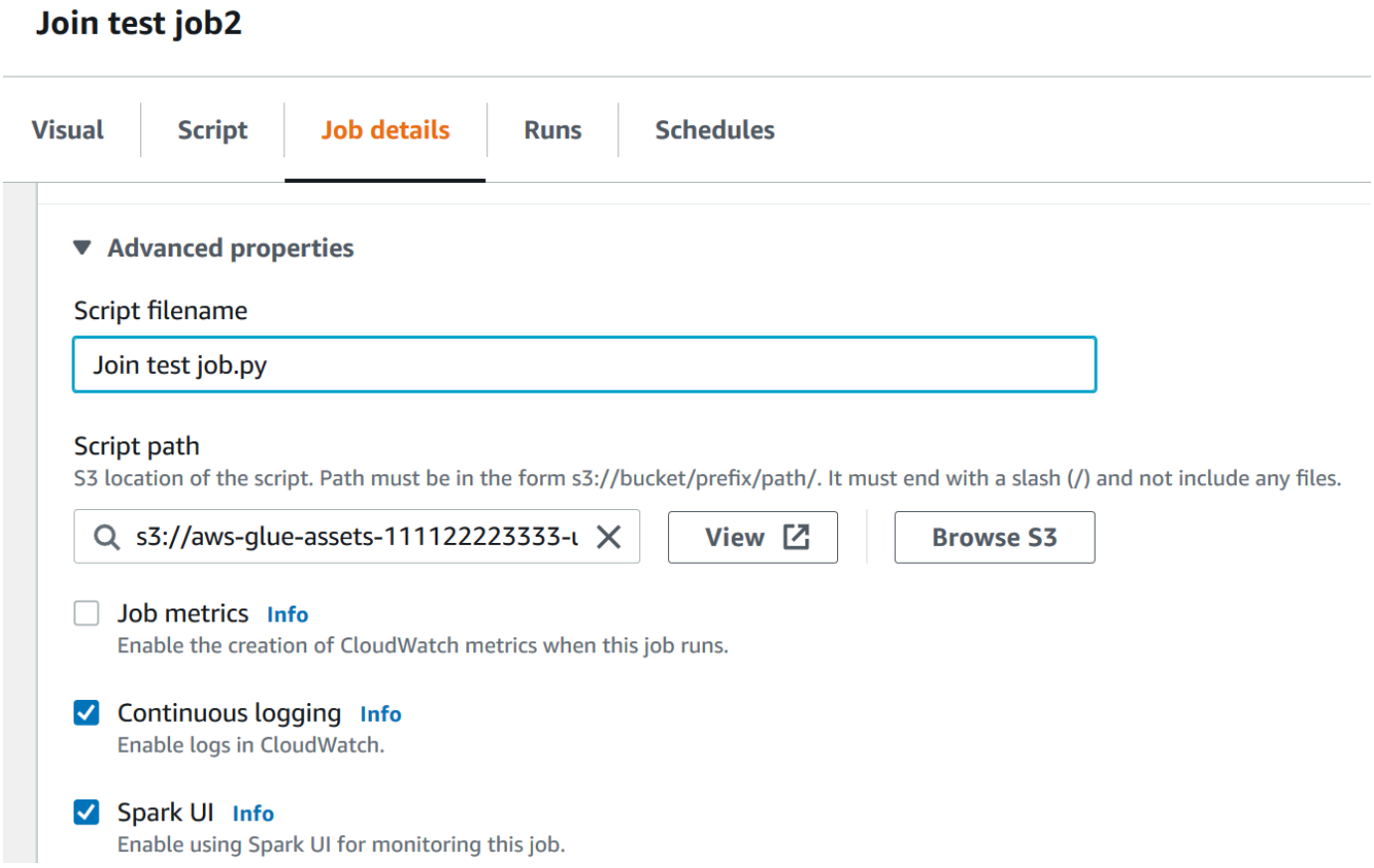
Jika Anda memilih Konfirmasi, tab Visual tidak lagi muncul di editor. Anda dapat menggunakan AWS Glue Studio untuk memodifikasi skrip menggunakan editor skrip, memodifikasi detail pekerjaan atau jadwal, atau melihat pekerjaan berjalan.

> **Note**
>
> Sampai Anda menyimpan pekerjaan, konversi ke pekerjaan skrip saja tidak permanen. Jika Anda menyegarkan halaman web konsol, atau menutup pekerjaan sebelum menyimpannya dan membukanya kembali di editor visual, Anda masih dapat mengedit masing-masing node di editor visual.

4. Mengedit skrip sesuai kebutuhan.

   Setelah selesai mengedit skrip, pilih Simpan untuk menyimpan tugas dan mengubah tugas secara permanen dari visual menjadi skrip-saja.

5. (Opsional) Anda dapat mengunduh skrip dari AWS Glue Studio konsol dengan memilih tombol Unduh pada tab Skrip. Bila Anda memilih tombol ini, jendela peramban baru akan terbuka, menampilkan skrip dari lokasinya di Amazon S3. Parameter Nama file skrip dan Path skrip dalam tab Detail tugas dari tugas tersebut menentukan nama dan lokasi file skrip di Amazon S3.



Ketika Anda menyimpan tugas, AWS Glue menyimpan skrip tugas di lokasi yang ditentukan oleh bidang ini. Jika Anda memodifikasi file skrip di lokasi ini dalam Amazon S3, AWS Glue Studio akan memuat skrip yang dimodifikasi saat berikutnya Anda mengedit pekerjaan.

Membuat dan mengedit skrip Scala di AWS Glue Studio

Ketika Anda memilih editor skrip untuk membuat pekerjaan, secara default, bahasa pemrograman pekerjaan diatur kePython 3. Jika Anda memilih untuk menulis skrip baru daripada mengunggah skrip, AWS Glue Studio mulailah skrip baru dengan teks boilerplate yang ditulis dengan Python. Jika Anda ingin menulis skrip Scala sebagai gantinya, Anda harus terlebih dahulu mengkonfigurasi editor skrip untuk menggunakan Scala.

> **Note**
>
> Jika Anda memilih Scala sebagai bahasa pemrograman untuk pekerjaan itu dan menggunakan editor visual untuk merancang pekerjaan Anda, skrip pekerjaan yang dihasilkan ditulis dalam Scala, dan tidak diperlukan tindakan lebih lanjut.

Untuk menulis skrip Scala baru di AWS Glue Studio

1. Buat pekerjaan baru dengan memilih opsi editor skrip Spark.
2. Di bawah Opsi, pilih Buat skrip baru dengan kode boilerplate.
3. Pilih tab Job details dan atur Language ke `Scala` (bukan`Python 3`).

> **Note**
>
> Properti Type untuk pekerjaan secara otomatis diatur ke `Spark` ketika Anda memilih opsi editor skrip Spark untuk membuat pekerjaan.

4. Pilih tab Script.
5. Hapus teks boilerplate Python. Anda dapat menggantinya dengan teks boilerplate Scala berikut.

```
import com.amazonaws.services.glue.{DynamicRecord, GlueContext}
import org.apache.spark.SparkContext
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job

object MyScript {
  def main(args: Array[String]): Unit = {
    val sc: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(sc)

    }
}
```

6. Tulis skrip pekerjaan Scala Anda di editor. Tambahkan `import` pernyataan tambahan sesuai kebutuhan.

Membuat dan mengedit pekerjaan shell Python di AWS Glue Studio

Saat Anda memilih editor skrip shell Python untuk membuat pekerjaan, Anda dapat mengunggah skrip Python yang ada, atau menulis yang baru. Jika Anda memilih untuk menulis skrip baru, kode boilerplate ditambahkan ke skrip pekerjaan Python baru.

Untuk membuat pekerjaan shell Python baru

Lihat instruksi di[Memulai pekerjaan di AWS Glue Studio](#).

Properti pekerjaan yang didukung untuk pekerjaan shell Python tidak sama dengan yang didukung untuk pekerjaan Spark. Daftar berikut menjelaskan perubahan parameter pekerjaan yang tersedia untuk pekerjaan shell Python pada tab Job details.

- Properti Type untuk pekerjaan secara otomatis diatur ke `Python Shell` dan tidak dapat diubah.
- Alih-alih Bahasa, ada properti versi Python untuk pekerjaan itu. Saat ini, pekerjaan shell Python dibuat menggunakan AWS Glue Studio Python 3.6.
- Properti versi Glue tidak tersedia, karena tidak berlaku untuk pekerjaan shell Python.
- Alih-alih jenis Pekerja dan Jumlah pekerja, properti unit pemrosesan Data ditampilkan sebagai gantinya. Properti pekerjaan ini menentukan berapa banyak unit pemrosesan data (DPU) yang dikonsumsi oleh shell Python saat menjalankan pekerjaan.
- Properti bookmark Job tidak tersedia, karena tidak didukung untuk pekerjaan shell Python.
- Di bawah properti Advanced, properti berikut tidak tersedia untuk pekerjaan shell Python.
  - Metrik Job
  - Pencatatan terus menerus
  - Jalur log UI Spark dan Spark UI
  - Jalur stoples dependen, di bawah judul Perpustakaan

Mengubah simpul induk untuk sebuah simpul dalam diagram tugas

Anda dapat mengubah induk simpul untuk memindahkan simpul dalam diagram tugas atau untuk mengubah sumber data untuk sebuah simpul.

Untuk mengubah simpul induk

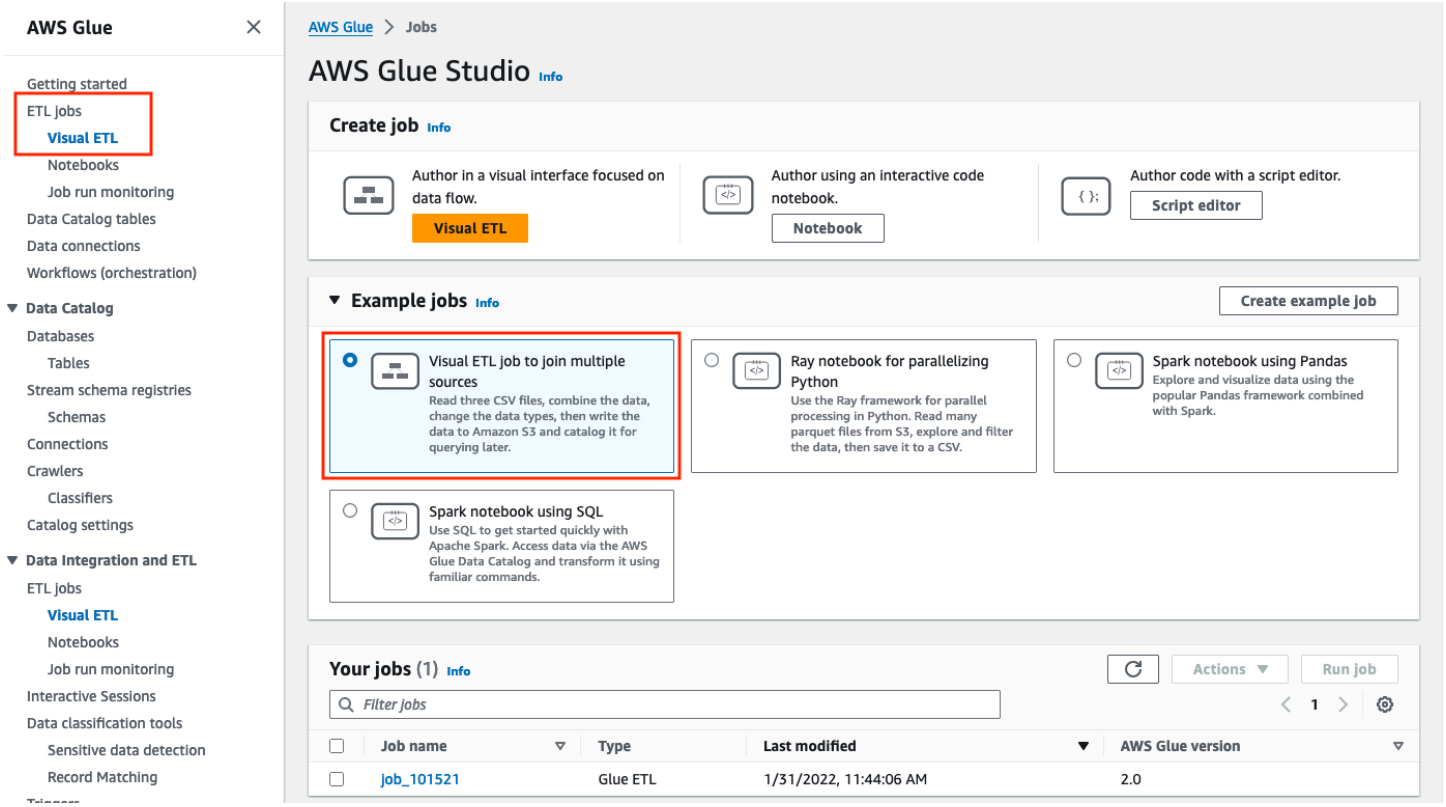
1. Pilih simpul dalam diagram tugas yang ingin Anda ubah.
2. Pada panel detail simpul, pada tab Properti simpul, pada judul Induk simpul, hapus induk simpul saat ini untuk simpul tersebut.
3. Pilih simpul induk baru dari daftar.
4. Memodifikasi properti lain dari simpul sesuai keperluan untuk mencocokkan simpul induk yang baru dipilih.

Jika Anda memodifikasi sebuah simpul secara tidak sengaja, Anda dapat menggunakan tombol Batalkan pada toolbar untuk membalikkan tindakan.

Menghapus simpul dari diagram tugas

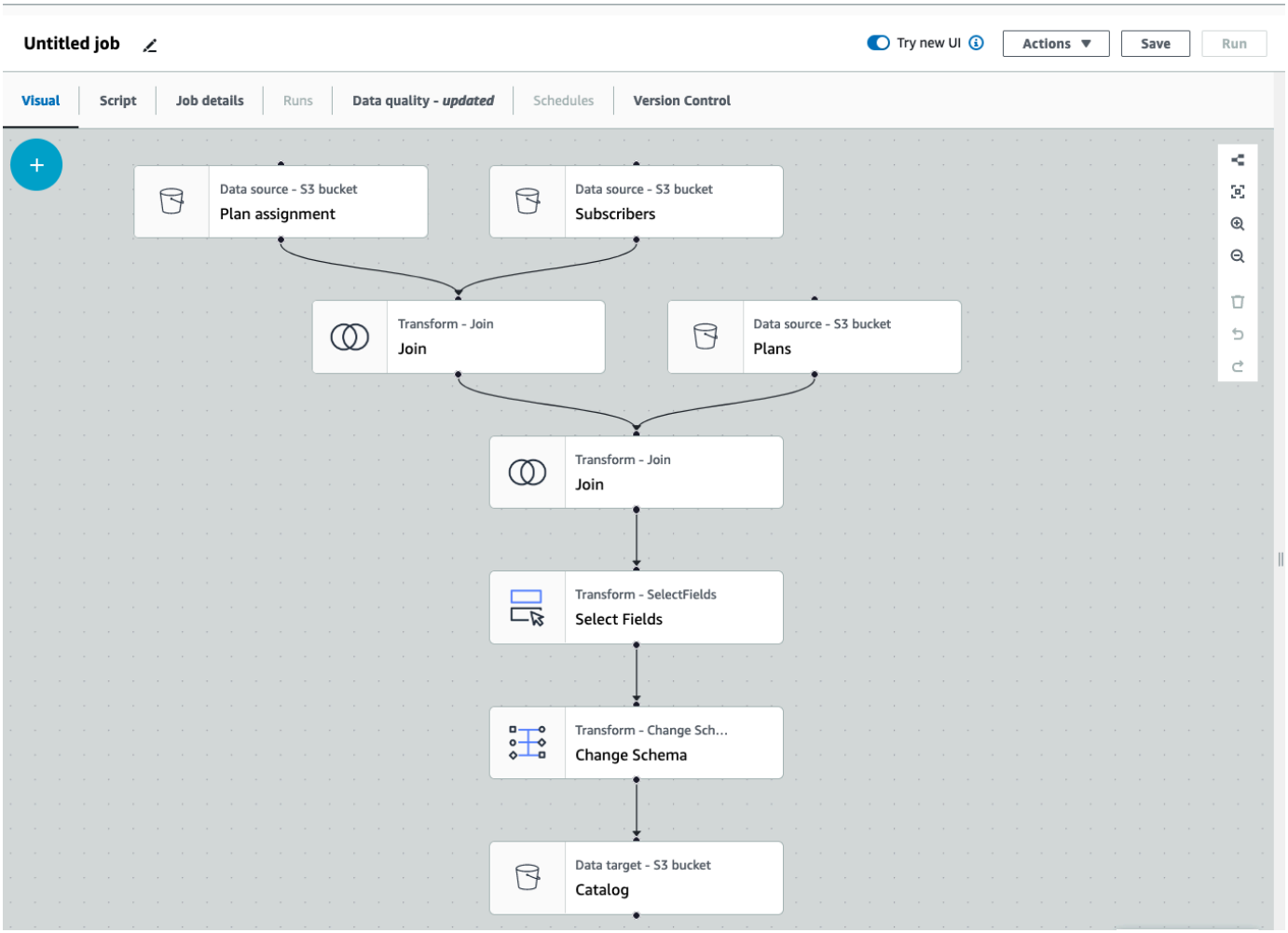
Saat bekerja dengan pekerjaan Visual ETL, Anda dapat menghapus node dari kanvas tanpa harus menambahkan kembali atau merestrukturisasi node apa pun yang terhubung ke node yang dihapus.

Pada contoh di bawah ini, Anda dapat mengikuti dengan memilih pekerjaan ETL > Visual ETL, lalu di Contoh pekerjaan, memilih pekerjaan Visual ETL untuk bergabung dengan beberapa sumber. Pilih Buat contoh pekerjaan untuk membuat pekerjaan dan ikuti langkah-langkah di bawah ini.



Untuk menghapus simpul dari kanvas

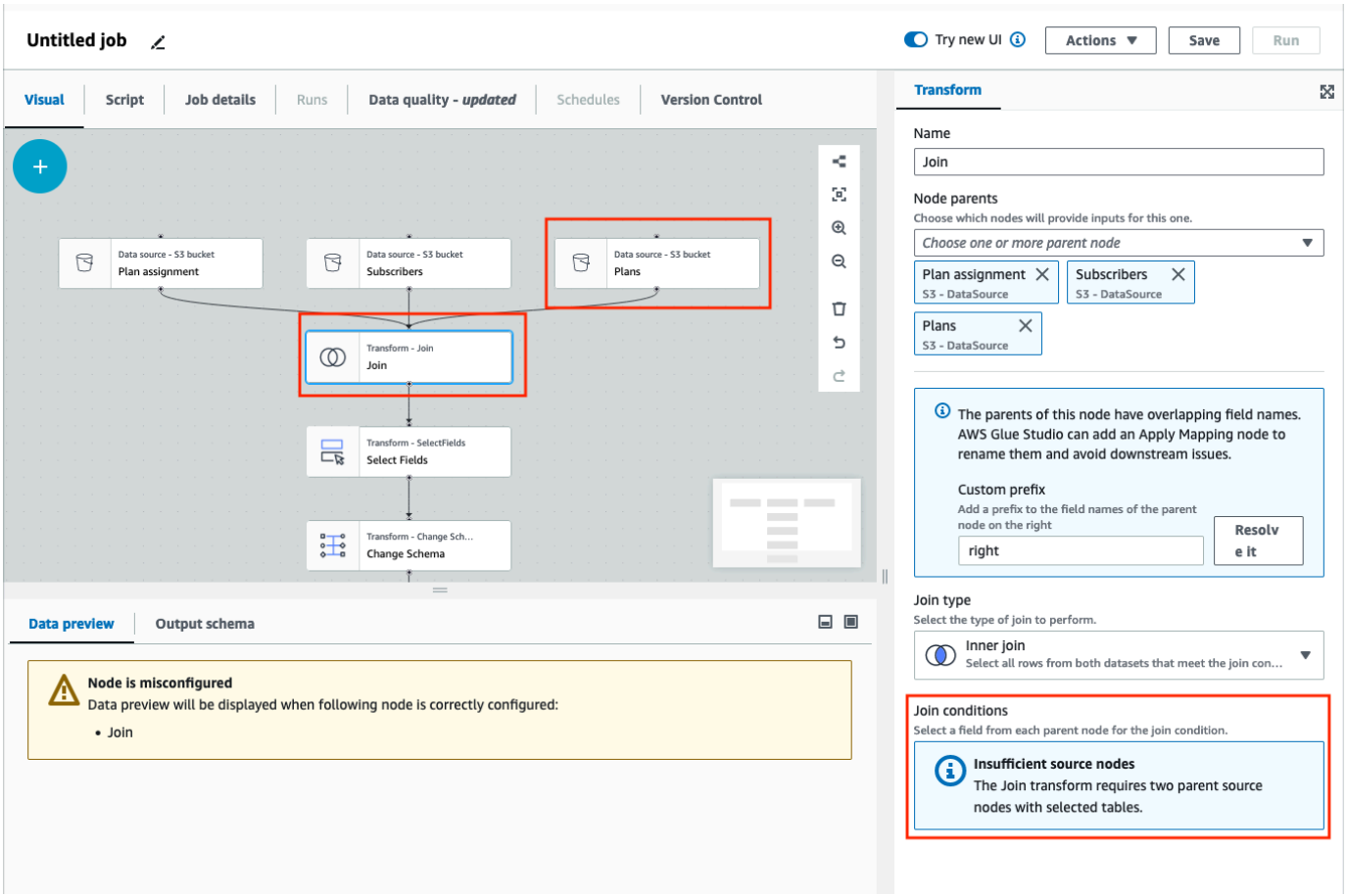
1. Dari AWS Glue konsol, pilih Visual ETL dari menu navigasi dan pilih pekerjaan yang ada. Kanvas pekerjaan menampilkan contoh pekerjaan seperti yang digambarkan di bawah ini.



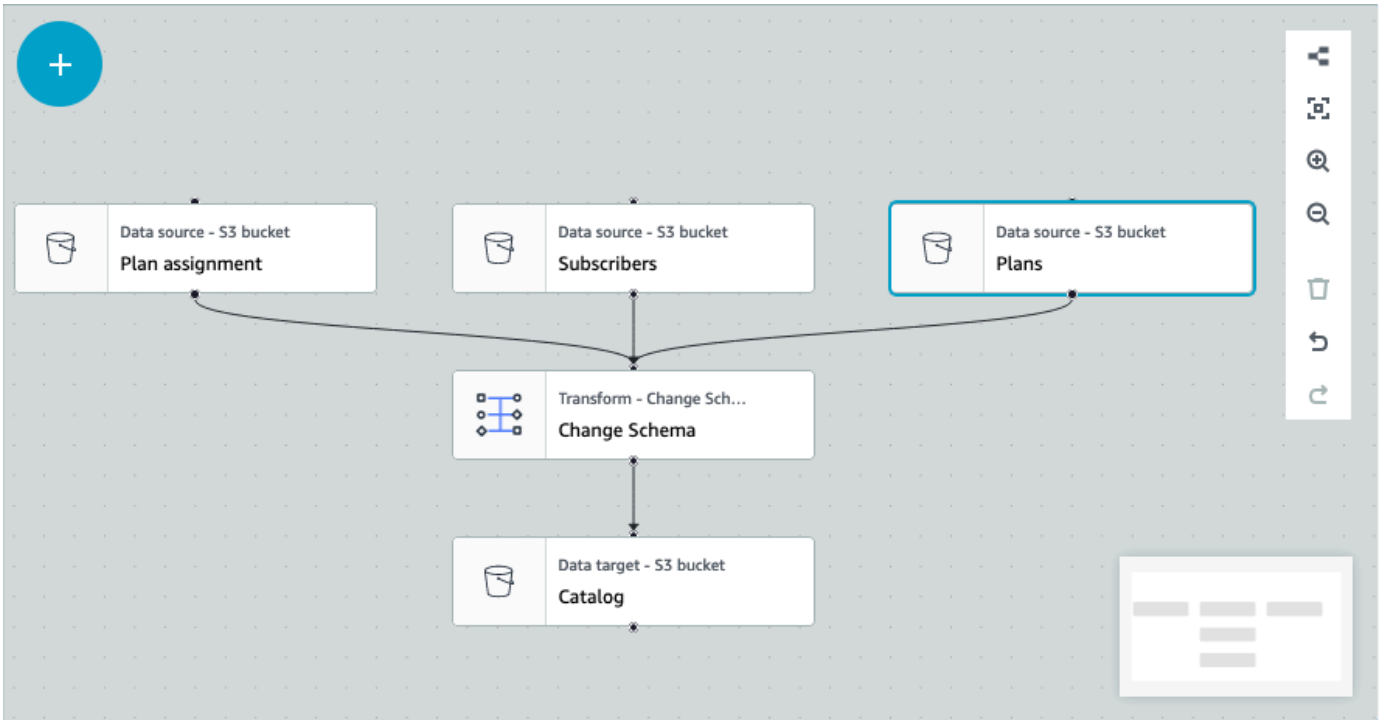
2. Pilih simpul yang ingin Anda hapus. Kanvas akan memperbesar ke node. Di bilah alat di sisi kanan kanvas, pilih ikon Sampah. Ini akan menghapus node dan setiap node yang terhubung ke node akan bergerak untuk mengambil tempatnya dalam alur kerja. Dalam contoh ini, node Join pertama dihapus dari kanvas.

   Jika Anda menghapus node dalam alur kerja, AWS Glue akan mengatur ulang node sehingga mereka diatur dengan cara yang tidak menghasilkan alur kerja yang tidak valid. Anda mungkin masih perlu memperbaiki konfigurasi node.

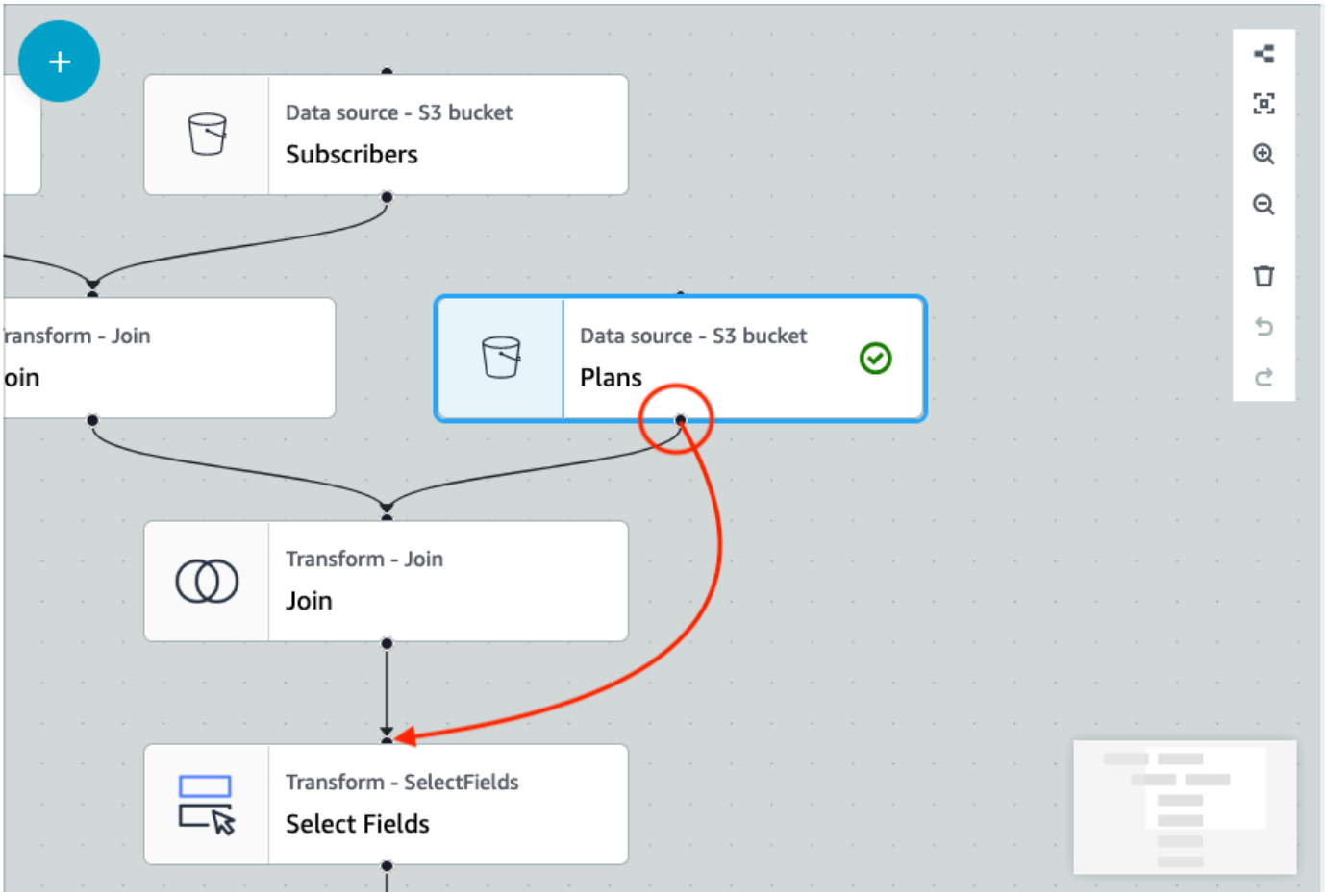
   Dalam contoh, node Join di bawah node Subscribers telah dihapus. Akibatnya, node sumber Rencana telah dipindahkan ke tingkat atas dan masih terhubung ke node Join anak. Node Join sekarang memerlukan konfigurasi tambahan karena Join membutuhkan dua node sumber induk dengan tabel yang dipilih. Tab Transform di sebelah kanan kanvas menampilkan persyaratan yang hilang dalam kondisi Gabung.



3. Hapus node Join kedua dan Select Fields node. Ketika node telah dihapus, alur kerja akan terlihat seperti contoh di bawah ini.



4. Untuk memodifikasi koneksi node, klik pada pegangan node dan seret koneksi ke node baru. Ini akan memungkinkan Anda untuk menghapus node dan mengatur ulang node dalam aliran logis. Dalam contoh, koneksi baru sedang dibuat dengan mengklik pegangan pada node Rencana dan menyeret koneksi ke node Join seperti yang digambarkan oleh panah merah.



5. Jika Anda perlu membatalkan tindakan apa pun, pilih ikon Undo langsung di bawah ikon Sampah di bilah alat di sisi kanan kanvas.

Menambahkan parameter sumber dan target ke node AWS Glue Data Catalog

AWS Glue Studiomemungkinkan Anda untuk membuat parameter pekerjaan visual. Karena nama tabel katalog di lingkungan produksi dan pengembangan mungkin berbeda, Anda dapat menentukan dan memilih parameter runtime untuk database dan tabel yang akan berjalan saat pekerjaan Anda berjalan.

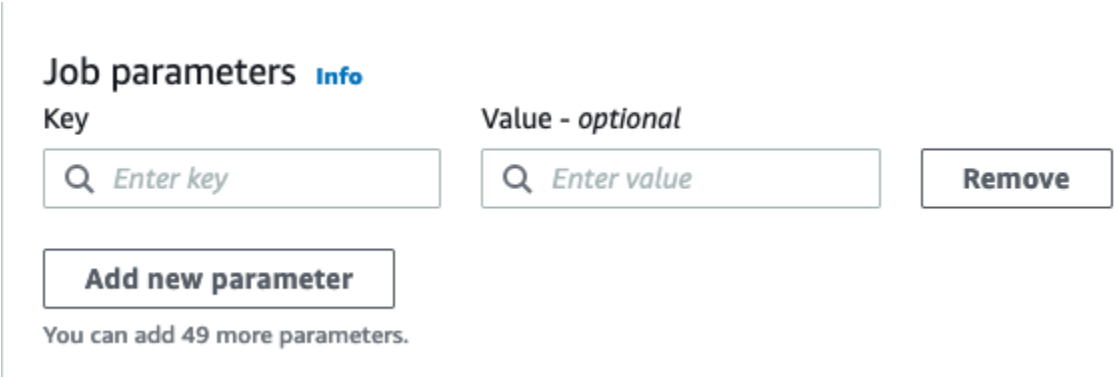
Parameterisasi Job memungkinkan Anda untuk membuat parameter sumber dan target, dan menyimpan parameter tersebut ke pekerjaan saat menggunakan node Data Catalog. AWS Glue Saat Anda menentukan sumber dan target sebagai parameter, Anda mengaktifkan kegunaan kembali pekerjaan, terutama saat menggunakan pekerjaan yang sama di berbagai lingkungan. Ini berguna saat mempromosikan kode di seluruh lingkungan penerapan dengan menghemat waktu dan tenaga

dalam mengelola sumber dan target Anda. Selain itu, parameter kustom yang Anda tentukan akan mengganti argumen default apa pun untuk menjalankan AWS Glue pekerjaan tertentu.

Untuk menambahkan parameter sumber dan target

Apakah Anda menggunakan node AWS Glue Data Catalog sebagai sumber atau target, Anda dapat menentukan parameter runtime di bagian Advanced properties pada tab Job details.

1. Pilih node AWS Glue Data Catalog sebagai node sumber atau node target.
2. Pilih tab Detail tugas.
3. Pilih Properti lanjutan.
4. Di bagian Parameter Job, masukkan nilai kunci. Misalnya, `--db.source` akan menjadi parameter untuk sumber database. Anda dapat memasukkan nama apa pun untuk kunci tersebut, selama nama kunci diikuti oleh 'dasbor dasbor'.



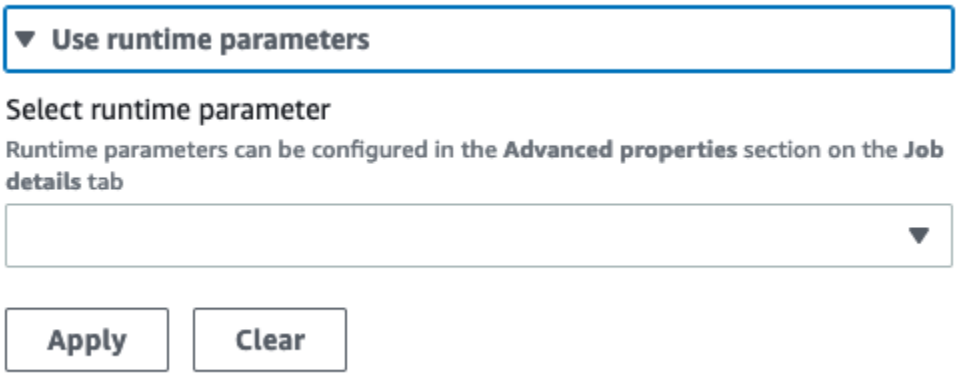
5. Masukkan nilainya. Misalnya, `databasename` akan menjadi nilai untuk database yang diparameterisasi.
6. Pilih Tambahkan parameter baru jika Anda ingin menambahkan lebih banyak parameter. Max 50 parameter diperbolehkan. Setelah pasangan nilai kunci telah ditentukan, Anda dapat menggunakan parameter di node AWS Glue Data Catalog.

Untuk memilih parameter runtime

> **Note**
>
> Proses untuk memilih parameter runtime untuk database dan tabel adalah sama apakah node AWS Glue Data Catalog adalah sumber atau target.

1. Pilih node AWS Glue Data Catalog sebagai node sumber atau node target.

2. Dalam tab Properti sumber data - Katalog Data, di bawah Database, pilih Gunakan parameter runtime.



3. Pilih parameter dari menu drop-down. Misalnya, ketika Anda memilih parameter yang Anda tentukan untuk database sumber, database akan secara otomatis terisi di menu drop-down database ketika Anda memilih Terapkan.
4. Di bagian Tabel, pilih parameter yang sudah Anda definisikan sebagai tabel sumber. Ketika Anda memilih Terapkan, tabel secara otomatis diisi sebagai tabel yang akan digunakan.
5. Ketika Anda menyimpan dan menjalankan pekerjaan, AWS Glue Studio akan referensi parameter yang dipilih selama menjalankan pekerjaan.

Menggunakan sistem kontrol versi Git di AWS Glue

> **Note**
>
> Notebook saat ini tidak didukung untuk kontrol versi diAWS Glue Studio. Namun, kontrol versi untuk skrip AWS Glue pekerjaan dan pekerjaan ETL visual didukung.

Jika Anda memiliki repositori jarak jauh dan ingin mengelola AWS Glue pekerjaan Anda menggunakan repositori Anda, Anda dapat menggunakan AWS Glue Studio atau AWS CLI untuk menyinkronkan perubahan ke repositori dan pekerjaan Anda di. AWS Glue Ketika Anda menyinkronkan perubahan dengan cara ini, Anda mendorong pekerjaan dari AWS Glue Studio ke repositori Anda, atau menarik dari repositori ke. AWS Glue Studio

Dengan integrasi Git diAWS Glue Studio, Anda dapat:

- Integrasi dengan sistem kontrol versi Git, seperti AWS CodeCommit, GitHub, GitLab, dan Bitbucket

- Edit AWS Glue pekerjaan AWS Glue Studio apakah Anda menggunakan pekerjaan visual atau pekerjaan skrip dan sinkronkan ke repositori
- Parameterisasi sumber dan target dalam pekerjaan
- Tarik pekerjaan dari repositori dan edit di AWS Glue Studio
- Uji pekerjaan dengan menarik dari cabang dan/atau mendorong ke cabang menggunakan alur kerja multi-cabang di AWS Glue Studio
- Unduh file dari repositori dan unggah pekerjaan AWS Glue Studio untuk pembuatan pekerjaan lintas akun
- Gunakan alat otomatisasi pilihan Anda (misalnya, Jenkins, AWS CodeDeploy, dll.)

Video ini menunjukkan bagaimana Anda dapat mengintegrasikan AWS Glue dengan Git dan membangun pipeline kode yang berkelanjutan dan kolaboratif.

Izin IAM

Pastikan pekerjaan memiliki salah satu izin IAM berikut. Untuk informasi selengkapnya tentang cara mengatur izin IAM, lihat Menyiapkan izin IAM untuk. AWS Glue Studio

- `AWSGlueServiceRole`
- `AWSGlueConsoleFullAccess`

Minimal, tindakan berikut diperlukan untuk integrasi Git:

- `glue:UpdateJobFromSourceControl`— untuk dapat memperbarui AWS Glue dengan pekerjaan yang ada di sistem kontrol versi
- `glue:UpdateSourceControlFromJob`— untuk dapat memperbarui sistem kontrol versi dengan pekerjaan yang disimpan di AWS Glue
- `s3:GetObject`— untuk dapat mengambil skrip untuk pekerjaan sambil mendorong ke sistem kontrol versi
- `s3:PutObject`— untuk dapat memperbarui skrip saat menarik pekerjaan dari sistem kontrol sumber

Prasyarat

Untuk mendorong pekerjaan ke repositori kontrol sumber, Anda perlu:

- repositori yang telah dibuat oleh administrator Anda
- cabang di repositori
- token akses pribadi (untuk Bitbucket, ini adalah Token Akses Repositori)
- nama pengguna pemilik repositori
- atur izin di repositori untuk memungkinkan membaca dan menulis AWS Glue Studio ke repositori
  - GitLab— atur cakupan token ke api, read_repository, dan write_repository
  - Bitbucket - atur izin ke:
    - Keanggotaan ruang kerja - baca, tulis
    - Proyek - tulis, baca admin
    - Repositori - baca, tulis, admin, hapus

> **Note**
>
> Saat menggunakan AWS CodeCommit, token akses pribadi dan pemilik repositori tidak diperlukan. Lihat Memulai dengan Git dan AWS CodeCommit.

Menggunakan pekerjaan dari repositori kontrol sumber Anda di AWS Glue Studio

Untuk menarik pekerjaan dari repositori kontrol sumber Anda yang tidak adaAWS Glue Studio, dan untuk menggunakan pekerjaan ituAWS Glue Studio, prasyarat akan tergantung pada jenis pekerjaan.

Untuk pekerjaan visual:

- Anda memerlukan folder dan file JSON dari definisi pekerjaan yang cocok dengan nama pekerjaan

  Misalnya, lihat definisi pekerjaan di bawah ini. Cabang di repositori Anda harus berisi jalur `my-visual-job/my-visual-job.json` di mana folder dan file JSON cocok dengan nama pekerjaan

```
{
  "name" : "my-visual-job",
  "description" : "",
  "role" : "arn:aws:iam::aws_account_id:role/Rolename",
  "command" : {
    "name" : "glueetl",
    "scriptLocation" : "s3://foldername/scripts/my-visual-job.py",
```

```
    "pythonVersion" : "3"
  },
  "codeGenConfigurationNodes" : "{\"node-nodeID\":{\"S3CsvSource\":
{\"AdditionalOptions\":{\"EnableSamplePath\":false,\"SamplePath\":\"s3://notebook-
test-input/netflix_titles.csv\"},\"Escaper\":\"\",\"Exclusions\":[],\"Name\":\"Amazon
 S3\",\"OptimizePerformance\":false,\"OutputSchemas\":[{\"Columns\":[{\"Name\":
\"show_id\",\"Type\":\"string\"},{\"Name\":\"type\",\"Type\":\"string\"},{\"Name\":
\"title\",\"Type\":\"choice\"},{\"Name\":\"director\",\"Type\":\"string\"},{\"Name\":
\"cast\",\"Type\":\"string\"},{\"Name\":\"country\",\"Type\":\"string\"},{\"Name\":
\"date_added\",\"Type\":\"string\"},{\"Name\":\"release_year\",\"Type\":\"bigint\"},
{\"Name\":\"rating\",\"Type\":\"string\"},{\"Name\":\"duration\",\"Type\":\"string
\"},{\"Name\":\"listed_in\",\"Type\":\"string\"},{\"Name\":\"description\",\"Type
\":\"string\"}]}],\"Paths\":[\"s3://dalamgir-notebook-test-input/netflix_titles.csv
\"],\"QuoteChar\":\"quote\",\"Recurse\":true,\"Separator\":\"comma\",\"WithHeader
\":true}}}"
}
```

Untuk pekerjaan skrip:

- Anda memerlukan folder, file JSON dari definisi pekerjaan, dan skrip
- folder dan file JSON harus cocok dengan nama pekerjaan. Nama skrip harus sesuai `scriptLocation` dengan definisi pekerjaan bersama dengan ekstensi file

  Misalnya, dalam definisi pekerjaan di bawah ini, cabang di repositori Anda harus berisi jalur `my-script-job/my-script-job.json` dan. `my-script-job/my-script-job.py` Nama skrip harus cocok dengan nama di `scriptLocation` termasuk ekstensi skrip

```
{
  "name" : "my-script-job",
  "description" : "",
  "role" : "arn:aws:iam::aws_account_id:role/Rolename",
  "command" : {
    "name" : "glueetl",
    "scriptLocation" : "s3://foldername/scripts/my-script-job.py",
    "pythonVersion" : "3"
  }
}
```

Batasan

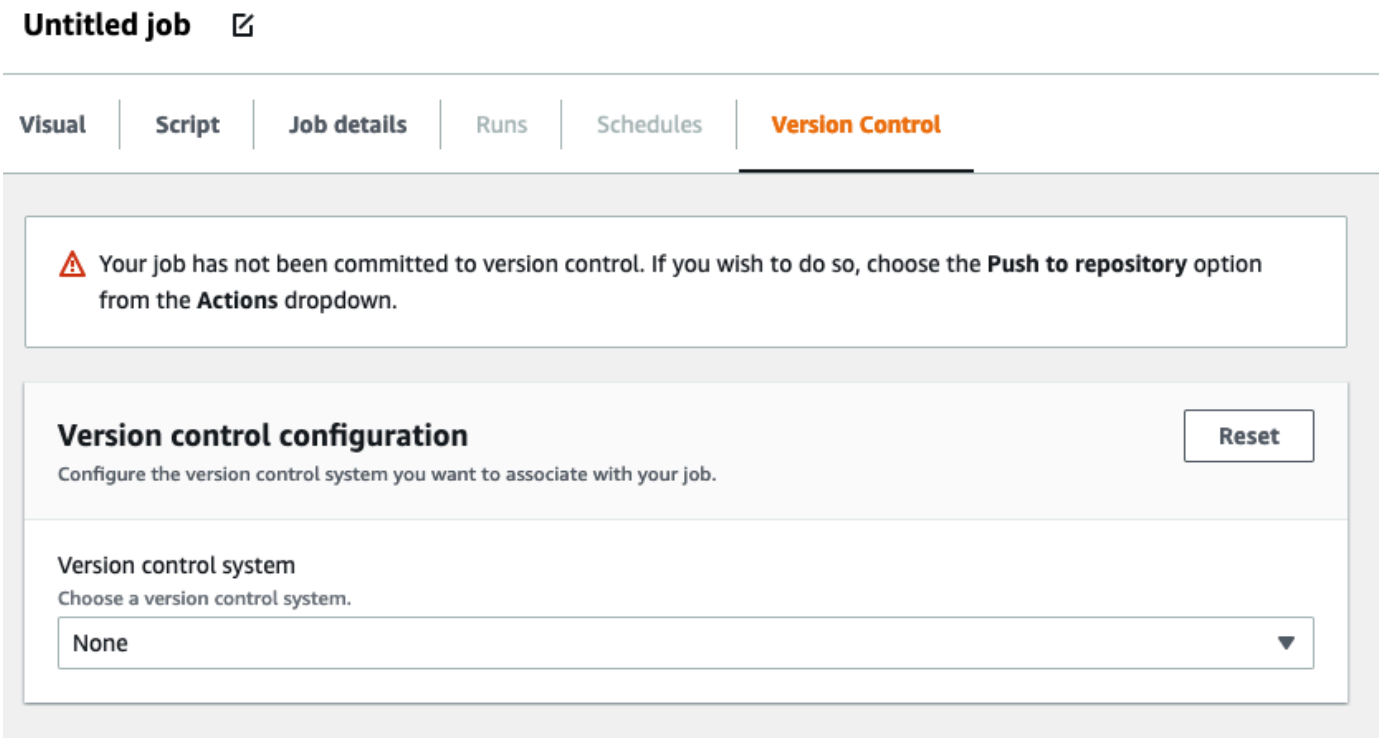
- AWS Gluesaat ini tidak mendukung mendorong/menarik dari -Groups. GitLab

Menghubungkan repositori kontrol versi dengan AWS Glue

Anda dapat memasukkan detail repositori kontrol versi Anda dan mengelolanya di tab Kontrol Versi di editor AWS Glue Studio pekerjaan. Untuk berintegrasi dengan repositori Git Anda, Anda harus terhubung ke repositori Anda setiap kali Anda masuk. AWS Glue Studio

Untuk menghubungkan sistem kontrol versi Git:

1. MasukAWS Glue Studio, mulai pekerjaan baru dan pilih tab Kontrol Versi.



2. Dalam sistem kontrol Versi, pilih Layanan Git dari opsi yang tersedia dengan mengklik menu tarik-turun.

   - AWS CodeCommit
   - GitHub
   - GitLab
   - Bitbucket

3. Bergantung pada sistem kontrol versi Git yang Anda pilih, Anda akan memiliki bidang yang berbeda untuk diselesaikan.

   Untuk AWS CodeCommit:

   Selesaikan konfigurasi repositori dengan memilih repositori dan cabang untuk pekerjaan Anda:

   - Repositori — jika Anda telah mengatur repositori di AWS CodeCommit, pilih repositori dari menu drop-down. Repositori Anda akan secara otomatis terisi dalam daftar
   - Cabang — pilih cabang dari menu drop-down
   - Folder — opsional - masukkan nama folder untuk menyimpan pekerjaan Anda. Jika dibiarkan kosong, folder dibuat secara otomatis. Nama folder default ke nama pekerjaan

   Untuk GitHub:

   Selesaikan GitHub konfigurasi dengan menyelesaikan bidang:

   - Token akses pribadi — ini adalah token yang disediakan oleh GitHub repositori. Untuk informasi selengkapnya tentang token akses pribadi, lihat GitHub Dokumen
   - Pemilik repositori — ini adalah pemilik repositori. GitHub

   Selesaikan konfigurasi repositori dengan memilih repositori dan cabang dari. GitHub

   - Repositori — jika Anda telah mengatur repositori di GitHub, pilih repositori dari menu drop-down. Repositori Anda akan secara otomatis terisi dalam daftar
   - Cabang — pilih cabang dari menu drop-down
   - Folder — opsional - masukkan nama folder untuk menyimpan pekerjaan Anda. Jika dibiarkan kosong, folder dibuat secara otomatis. Nama folder default ke nama pekerjaan

   Untuk GitLab:

   > **Note**
   >
   > AWS Gluesaat ini tidak mendukung mendorong/menarik dari -Groups. GitLab

- Token akses pribadi — ini adalah token yang disediakan oleh GitLab repositori. Untuk informasi selengkapnya tentang token akses pribadi, lihat Token akses GitLab pribadi
- Pemilik repositori — ini adalah pemilik repositori. GitLab

Selesaikan konfigurasi repositori dengan memilih repositori dan cabang dari. GitLab

- Repositori — jika Anda telah mengatur repositori di GitLab, pilih repositori dari menu drop-down. Repositori Anda akan secara otomatis terisi dalam daftar
- Cabang — pilih cabang dari menu drop-down
- Folder — opsional - masukkan nama folder untuk menyimpan pekerjaan Anda. Jika dibiarkan kosong, folder dibuat secara otomatis. Nama folder default ke nama pekerjaan

Untuk Bitbucket:

- Kata sandi aplikasi — Bitbucket menggunakan kata sandi Aplikasi dan bukan Token Akses Repositori. Untuk informasi selengkapnya tentang Sandi aplikasi, lihat Sandi aplikasi.
- Pemilik repositori — ini adalah pemilik repositori Bitbucket. Di Bitbucket, pemiliknya adalah pencipta repositori.

Lengkapi konfigurasi repositori dengan memilih ruang kerja, repositori, cabang, dan folder dari Bitbucket.

- Workspace — jika Anda memiliki ruang kerja yang diatur di Bitbucket, pilih ruang kerja dari menu drop-down. Ruang kerja Anda terisi secara otomatis
- Repositori — jika Anda telah mengatur repositori di Bitbucket, pilih repositori dari menu drop-down. Repositori Anda diisi secara otomatis
- Cabang — pilih cabang dari menu drop-down. Cabang Anda terisi secara otomatis
- Folder — opsional - masukkan nama folder untuk menyimpan pekerjaan Anda. Jika dibiarkan kosong, folder secara otomatis dibuat dengan nama pekerjaan.

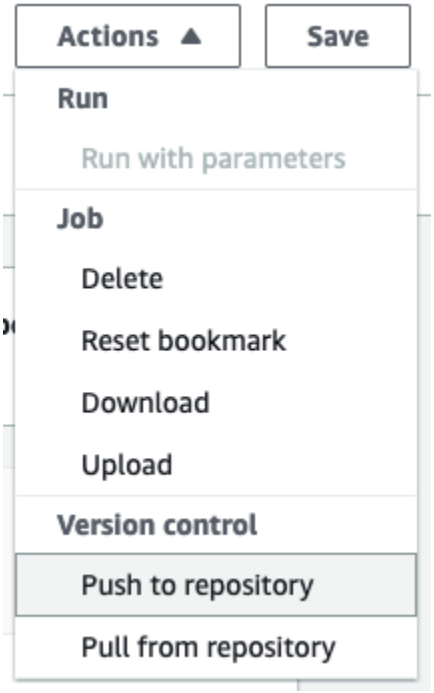
4. Pilih Simpan di bagian atas AWS Glue Studio pekerjaan

Mendorong AWS Glue pekerjaan ke repositori sumber

Setelah Anda memasukkan detail sistem kontrol versi Anda, Anda dapat mengedit pekerjaan AWS Glue Studio dan mendorong pekerjaan ke repositori sumber Anda. Jika Anda tidak terbiasa dengan konsep Git seperti mendorong dan menarik, lihat tutorial ini tentang Memulai dengan Git dan AWS CodeCommit.

Untuk mendorong pekerjaan Anda ke repositori, Anda harus memasukkan detail sistem kontrol versi Anda dan menyimpan pekerjaan Anda.

1. Dalam AWS Glue Studio pekerjaan, pilih Tindakan. Ini akan membuka opsi menu tambahan.



2. Pilih Push to repository.

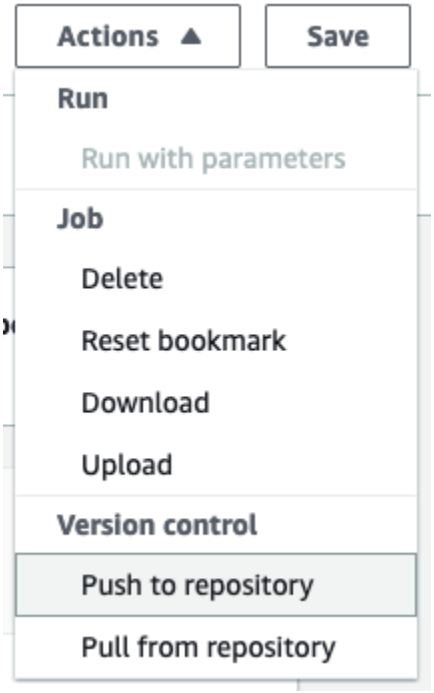
   Tindakan ini akan menyelamatkan pekerjaan. Saat Anda mendorong ke repositori, AWS Glue Studio dorong perubahan terakhir yang disimpan. Jika pekerjaan di repositori diubah oleh Anda atau pengguna lain dan tidak sinkron dengan pekerjaan diAWS Glue Studio, pekerjaan di repositori akan ditimpa dengan pekerjaan yang disimpan AWS Glue Studio saat Anda mendorong pekerjaan dari. AWS Glue Studio

3. Pilih Konfirmasi untuk menyelesaikan tindakan. Ini membuat komit baru di repositori. Jika Anda menggunakan AWS CodeCommit, pesan konfirmasi akan menampilkan tautan ke komit terbaru AWS CodeCommit.

Menarik AWS Glue pekerjaan dari repositori sumber

Setelah Anda memasukkan detail repositori Git Anda ke dalam tab kontrol Versi, Anda juga dapat menarik pekerjaan dari repositori Anda dan mengeditnya. AWS Glue Studio

1. Dalam AWS Glue Studio pekerjaan, pilih Tindakan. Ini akan membuka opsi menu tambahan.



2. Pilih Tarik dari repositori.
3. Pilih Konfirmasi. Ini mengambil komit terbaru dari repositori dan memperbarui pekerjaan Anda. AWS Glue Studio
4. Edit pekerjaan Anda diAWS Glue Studio. Jika Anda membuat perubahan, Anda dapat menyinkronkan pekerjaan Anda ke repositori Anda dengan memilih Push to repository dari menu drop-down Actions.

Menulis kode dengan notebook AWS Glue Studio

Insinyur data dapat membuat AWS Glue pekerjaan lebih cepat dan lebih mudah daripada sebelum menggunakan antarmuka notebook interaktif AWS Glue Studio atau sesi interaktif diAWS Glue.

Topik

- Ikhtisar menggunakan notebook
- Membuat pekerjaan ETL menggunakan notebook di AWS Glue Studio

Ikhtisar menggunakan notebook

AWS Glue Studiomemungkinkan Anda untuk menulis pekerjaan secara interaktif di antarmuka notebook berdasarkan Jupyter Notebooks. Melalui notebook diAWS Glue Studio, Anda dapat mengedit skrip pekerjaan dan melihat output tanpa harus menjalankan pekerjaan penuh, dan Anda dapat mengedit kode integrasi data dan melihat output tanpa harus menjalankan pekerjaan penuh, dan Anda dapat menambahkan penurunan harga dan menyimpan buku catatan sebagai file.ipynb dan skrip pekerjaan. Anda dapat memulai notebook tanpa menginstal perangkat lunak secara lokal atau mengelola server. Ketika Anda puas dengan kode Anda, AWS Glue Studio dapat mengkonversi notebook Anda ke pekerjaan Glue dengan mengklik tombol.

Beberapa manfaat menggunakan notebook meliputi:

- Tidak ada cluster untuk menyediakan atau mengelola
- Tidak ada cluster idle yang harus dibayar
- Tidak diperlukan konfigurasi di muka
- Tidak diperlukan pemasangan notebook Jupyter
- Runtime/platform yang sama dengan ETL AWS Glue

Ketika Anda memulai buku catatanAWS Glue Studio, semua langkah konfigurasi dilakukan untuk Anda sehingga Anda dapat menjelajahi data Anda dan mulai mengembangkan skrip pekerjaan Anda setelah hanya beberapa detik. AWS Glue Studiomengkonfigurasi notebook Jupyter dengan kernel Jupyter. AWS Glue Anda tidak perlu mengonfigurasi VPC, koneksi jaringan, atau titik akhir pengembangan untuk menggunakan buku catatan ini.

Untuk membuat pekerjaan menggunakan antarmuka notebook:

- konfigurasikan izin IAM yang diperlukan.
- memulai sesi buku catatan untuk membuat pekerjaan
- tulis kode di sel di buku catatan
- jalankan dan uji kode untuk melihat output

- selamatkan pekerjaan

Setelah buku catatan Anda disimpan, buku catatan Anda adalah AWS Glue pekerjaan penuh. Anda dapat mengelola semua aspek pekerjaan, seperti menjadwalkan pekerjaan berjalan, menetapkan parameter pekerjaan, dan melihat riwayat menjalankan pekerjaan tepat di samping buku catatan Anda.

Membuat pekerjaan ETL menggunakan notebook di AWS Glue Studio

Untuk mulai menggunakan notebook di konsol AWS Glue Studio

1. Lampirkan AWS Identity and Access Management kebijakan ke AWS Glue Studio pengguna dan buat peran IAM untuk pekerjaan dan buku catatan ETL Anda.
2. Konfigurasikan keamanan IAM tambahan untuk notebook, seperti yang dijelaskan dalam. Memberikan izin untuk peran IAM
3. Buka AWS Glue Studio konsol di https://console.aws.amazon.com/gluestudio/.

   > **Note**
   >
   > Periksa apakah browser Anda tidak memblokir cookie pihak ketiga. Browser apa pun yang memblokir cookie pihak ketiga baik secara default atau sebagai pengaturan yang diaktifkan pengguna akan mencegah notebook diluncurkan. Untuk informasi selengkapnya tentang mengelola cookie, lihat:

   - Chrome
   - Firefox
   - Safari
4. Pilih link Jobs di menu navigasi sisi kiri.
5. Pilih buku catatan Jupyter dan kemudian pilih Buat untuk memulai sesi buku catatan baru.
6. Pada halaman Buat pekerjaan di buku catatan Jupyter, berikan nama pekerjaan, dan pilih peran IAM yang akan digunakan. Pilih Buat tugas.

   Setelah periode waktu yang singkat, editor notebook muncul.
7. Setelah Anda menambahkan kode, Anda harus menjalankan sel untuk memulai sesi. Ada beberapa cara untuk mengeksekusi sel:

- Tekan tombol putar.
- Gunakan pintasan keyboard:
  - Di macOS, Command + Enter untuk menjalankan sel.
  - Pada Windows, Shift+Enter untuk menjalankan sel.

Untuk informasi tentang menulis kode menggunakan antarmuka notebook Jupyter, lihat Dokumentasi Pengguna Notebook Jupyter.

8. Untuk menguji skrip Anda, jalankan seluruh skrip, atau sel individual. Output perintah apa pun akan ditampilkan di area di bawah sel.
9. Setelah Anda selesai mengembangkan notebook Anda, Anda dapat menyimpan pekerjaan dan kemudian menjalankannya. Anda dapat menemukan skrip di tab Script. Sihir apa pun yang Anda tambahkan ke buku catatan akan dilucuti dan tidak akan disimpan sebagai bagian dari skrip pekerjaan yang dihasilkanAWS Glue. AWS Glue Studioakan otomatis menambahkan a `job.commit()` ke akhir skrip yang Anda hasilkan dari isi buku catatan.

   Untuk informasi selengkapnya tentang menjalankan pekerjaan, lihatMemulai eksekusi tugas.

Komponen editor notebook

Antarmuka editor notebook memiliki bagian utama berikut.

- Antarmuka notebook (panel utama) dan bilah alat
- Tab pengeditan pekerjaan

Editor buku catatan

Editor AWS Glue Studio notebook didasarkan pada Aplikasi Notebook Jupyter. Antarmuka AWS Glue Studio notebook mirip dengan yang disediakan oleh Juypter Notebooks, yang dijelaskan di bagian Antarmuka pengguna Notebook. Notebook yang digunakan oleh sesi interaktif adalah Notebook Jupyter.

Meskipun AWS Glue Studio notebook ini mirip dengan Notebook Juptyer, notebook ini berbeda dalam beberapa cara utama:

- saat ini, AWS Glue Studio notebook tidak dapat menginstal ekstensi

- Anda tidak dapat menggunakan banyak tab; ada hubungan 1:1 antara pekerjaan dan buku catatan
- AWS Glue Studionotebook tidak memiliki menu file atas yang sama yang ada di Jupyter Notebooks
- saat ini, AWS Glue Studio notebook hanya berjalan dengan AWS Glue kernel. Perhatikan bahwa Anda tidak dapat memperbarui kernel sendiri.

AWS Glue Studiotab pengeditan pekerjaan

Tab yang Anda gunakan untuk berinteraksi dengan pekerjaan ETL ada di bagian atas halaman buku catatan. Mereka mirip dengan tab yang muncul di editor pekerjaan visualAWS Glue Studio, dan mereka melakukan tindakan yang sama.

- Notebook — Gunakan tab ini untuk melihat skrip pekerjaan menggunakan antarmuka notebook.
- Rincian pekerjaan - Konfigurasikan lingkungan dan properti untuk menjalankan pekerjaan.
- Berjalan - Lihat informasi tentang proses sebelumnya dari pekerjaan ini.
- Jadwal — Konfigurasikan jadwal untuk menjalankan pekerjaan Anda pada waktu tertentu.

Menyimpan buku catatan dan skrip pekerjaan Anda

Anda dapat menyimpan buku catatan dan skrip pekerjaan yang Anda buat kapan saja. Cukup pilih tombol Simpan di sudut kanan atas, sama seperti jika Anda menggunakan editor visual atau skrip.

Saat Anda memilih Simpan, file notebook disimpan di lokasi default:

- Secara default, skrip pekerjaan disimpan ke lokasi Amazon S3 yang ditunjukkan di tab Rincian Pekerjaan, di bawah Properti lanjutan, di jalur Skrip properti Detail pekerjaan. Skrip Job disimpan dalam subfolder bernama. `Scripts`
- Secara default, file notebook (`.ipynb`) disimpan ke lokasi Amazon S3 yang ditunjukkan di tab Rincian Pekerjaan, di bawah Properti lanjutan, di jalur Skrip rincian Pekerjaan. File notebook disimpan dalam subfolder bernama`Notebooks`.

> **Note**
>
> Saat Anda menyimpan pekerjaan, skrip pekerjaan hanya berisi sel kode dari buku catatan. Sel Markdown dan sihir tidak termasuk dalam skrip pekerjaan. Namun, `.ipynb` file tersebut akan berisi penurunan harga dan sihir apa pun.

Setelah Anda menyimpan pekerjaan, Anda kemudian dapat menjalankan pekerjaan menggunakan skrip yang Anda buat di notebook.

Mengelola sesi buku catatan

Notebook di AWS Glue Studio didasarkan pada fitur sesi interaktif. AWS Glue Ada biaya untuk menggunakan sesi interaktif. Untuk membantu mengelola biaya, Anda dapat memantau sesi yang dibuat untuk akun Anda, dan mengonfigurasi pengaturan default untuk semua sesi.

Mengubah batas waktu default untuk semua sesi buku catatan

Secara default, AWS Glue Studio notebook yang disediakan habis setelah 12 jam jika notebook diluncurkan dan tidak ada sel yang dieksekusi. Tidak ada biaya yang terkait dengannya dan batas waktu tidak dapat dikonfigurasi.

Setelah Anda menjalankan sel, ini akan memulai sesi interaktif. Sesi ini memiliki batas waktu default 48 jam. Batas waktu ini dapat dikonfigurasi dengan meneruskan `%idle_timeout` sihir sebelum mengeksekusi sel.

Untuk mengubah batas waktu sesi default untuk buku catatan di AWS Glue Studio

1. Di buku catatan, masukkan `%idle_timeout` keajaiban dalam sel dan tentukan nilai batas waktu dalam hitungan menit.
2. Misalnya: `%idle_timeout 15` akan mengubah batas waktu default menjadi 15 menit. Jika sesi tidak digunakan dalam 15 menit, sesi secara otomatis dihentikan.

Menginstal modul Python tambahan

Jika Anda ingin menginstal modul tambahan ke sesi Anda menggunakan pip, Anda dapat melakukannya dengan menggunakan `%additional_python_modules` untuk menambahkannya ke sesi Anda:

```
%additional_python_modules awswrangler, s3://mybucket/mymodule.whl
```

Semua argumen ke additional_python_modules diteruskan ke `pip3 install -m <>`

Untuk daftar modul Python yang tersedia, lihat Menggunakan pustaka Python dengan. AWS Glue

Mengubah AWS Glue Konfigurasi

Anda dapat menggunakan sihir untuk mengontrol nilai konfigurasi AWS Glue pekerjaan. Jika Anda ingin mengubah nilai konfigurasi pekerjaan, Anda harus menggunakan sihir yang tepat di notebook. Lihat Magics didukung oleh sesi AWS Glue interaktif untuk Jupyter.

> **Note**
>
> Properti pengganti untuk sesi berjalan tidak lagi tersedia. Untuk mengubah konfigurasi sesi, Anda dapat menghentikan sesi, mengatur konfigurasi baru dan kemudian memulai sesi baru.

AWS Gluemendukung berbagai jenis pekerja. Anda dapat mengatur tipe pekerja dengan`%worker_type`. Misalnya: `%worker_type G.2X` . Defaultnya adalah G.1X.

Anda juga dapat menentukan Jumlah pekerja dengan`%number_of_workers`. Misalnya, untuk menentukan 40 pekerja:`%number_of_workers 40`.

Untuk informasi selengkapnya lihat Mendefinisikan Properti Job

Hentikan sesi buku catatan

Untuk menghentikan sesi notebook, gunakan sihir`%stop_session`.

Jika Anda menavigasi jauh dari notebook di AWS konsol, Anda akan menerima pesan peringatan di mana Anda dapat memilih untuk menghentikan sesi.

Menggunakan CodeWhisperer dengan AWS Glue Studio notebooks

AWS Glue Studiomemungkinkan Anda untuk menulis pekerjaan secara interaktif di antarmuka notebook berdasarkan Jupyter Notebooks. Menggunakan CodeWhisperer meningkatkan pengalaman penulisan dalam AWS Glue Studio notebook.

CodeWhisperer Ekstensi Amazon mendukung penulisan kode dengan menghasilkan rekomendasi kode dan menyarankan perbaikan yang terkait dengan masalah kode.

Apa itu Amazon CodeWhisperer?

Amazon CodeWhisperer adalah layanan yang didukung oleh pembelajaran mesin yang membantu meningkatkan produktivitas pengembang. CodeWhisperer mencapai ini dengan menghasilkan

rekomendasi kode berdasarkan komentar pengembang dalam bahasa alami dan kode mereka di IDE. Selama pratinjau, Amazon CodeWhisperer tersedia untuk Java, Python JavaScript, C # dan TypeScript bahasa pemrograman. Layanan ini terintegrasi dengan JupyterLab, Amazon SageMaker Studio, instance Amazon SageMaker notebook, dan lingkungan pengembangan terintegrasi (IDE) lainnya.

Untuk informasi selengkapnya, lihat Pengaturan CodeWhisperer dengan AWS Glue Studio.

AWS Gluestatus job run di konsol

Anda dapat melihat status pekerjaan AWS Glue ekstrak, transformasi, dan beban (ETL) saat sedang berjalan atau setelah berhenti. Anda dapat melihat status menggunakan AWS Glue konsol. Untuk informasi selengkapnya tentang status menjalankan pekerjaan, lihatthe section called “Status Job run”.

Mengakses dasbor pemantauan tugas

Anda mengakses dasbor pemantauan pekerjaan dengan memilih tautan Pemantauan di panel AWS Glue navigasi.

Gambaran umum tentang dasbor pemantauan tugas

Dasbor pemantauan tugas menyediakan ringkasan keseluruhan dari eksekusi tugas, dengan total untuk tugas dengan status Berjalan, Dibatalkan, Berhasil, atau Gagal. Ubin tambahan memberikan tingkat keberhasilan eksekusi tugas secara keseluruhan, perkiraan penggunaan DPU untuk tugas, rincian status tugas yang dihitung berdasarkan jenis tugas, jenis pekerja, dan berdasarkan hari.

Grafik di ubin bersifat interaktif. Anda dapat memilih blok apa pun dalam grafik untuk menjalankan filter yang hanya menampilkan tugas tersebut di tabel Eksekusi tugas di bagian bawah halaman.

Anda dapat mengubah rentang tanggal untuk informasi yang ditampilkan di halaman ini dengan menggunakan pemilih Rentang tanggal. Bila Anda mengubah rentang tanggal, ubin informasi akan menyesuaikan untuk menampilkan nilai untuk jumlah hari yang ditentukan sebelum tanggal saat ini. Anda juga dapat menggunakan rentang tanggal tertentu jika Anda memilih Kustom dari pemilih rentang tanggal.

Tampilan eksekusi tugas

> **Note**
>
> Riwayat Job run dapat diakses selama 90 hari untuk alur kerja dan pekerjaan Anda.

Daftar sumber daya Eksekusi tugas menunjukkan tugas untuk rentang tanggal yang ditentukan dan filter.

Anda dapat mem-filter tugas berdasarkan pada kriteria tambahan, seperti status, jenis pekerja, jenis tugas, dan nama tugas. Di kotak filter di bagian atas tabel, Anda dapat memasukkan teks untuk digunakan sebagai filter. Hasil tabel diperbarui dengan baris yang berisi teks yang cocok dengan saat Anda memasukkan teks.

Anda dapat melihat sebuah subset dari tugas dengan memilih elemen dari grafik pada dasbor pemantauan tugas. Misalnya, jika Anda memilih jumlah tugas yang sedang berjalan di ubin Ringkasan eksekusi tugas, maka Eksekusi tugas akan menampilkan hanya tugas yang saat ini memiliki status `Running` saja. Jika Anda memilih salah satu batang di bagan batang Perincian jenis pekerja, maka hanya tugas yang berjalan dengan jenis pekerja dan status yang cocok saja yang ditampilkan dalam daftar Eksekusi tugas.

Daftar sumber daya Eksekusi tugas menampilkan detail untuk eksekusi tugas. Anda dapat mengurutkan baris dalam tabel dengan memilih judul kolom. Tabel berisi informasi berikut:

Properti	Deskripsi
Nama tugas	Nama pekerjaannya.
Tipe	Jenis lingkungan tugas: • ETL Glue: Berjalan di lingkungan Apache Spark terkelola AWS Glue. • Streaming Glue: Berjalan di lingkungan Apache Spark dan melakukan ETL pada aliran data. • Python shell: Menjalankan skrip Python sebagai shell.

Properti	Deskripsi
Waktu mulai	Tanggal dan waktu saat eksekusi tugas ini dimulai.
Waktu akhir	Tanggal dan waktu saat eksekusi tugas ini selesai.
Status eksekusi	Status eksekusi tugas saat ini. Nilai dapat berupa: • `STARTING` • `RUNNING` • `STOPPING` • `STOPPED` • `SUCCEEDED` • `FAILED` • `TIMEOUT`
Waktu aktif	Jumlah waktu eksekusi tugas menggunakan sumber daya.
Kapasitas	Jumlah unit pemrosesan data (DPU) AWS Glue yang dialokasikan untuk eksekusi tugas ini. Untuk informasi selengkapnya tentang perencanaan kapasitas, lihat Memantau Perencanaan Kapasitas DPU di Panduan Developer AWS Glue.

Properti	Deskripsi
Jenis pekerja	Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan ketika tugas dieksekusi. Nilai bisa`G.1X`,`G.2X`, `G.4X` atau`G.8X`. • **`G.1X`** — Ketika Anda memilih jenis ini, maka Anda juga memberikan nilai untuk Jumlah pekerja. Setiap pekerja memetakan ke 1 DPU (4 vCPU, memori 16 GB) dengan disk 84GB (sekitar 34GB gratis). Kami merekomendasikan jenis pekerja ini untuk tugas yang membutuhkan banyak memori. Ini adalah tipe Pekerja default untuk AWS Glue versi 2.0 atau pekerjaan yang lebih baru. • **`G.2X`** — Ketika Anda memilih jenis ini, maka Anda juga memberikan nilai untuk Jumlah pekerja. Setiap pekerja memetakan ke 2 DPU (8 vCPU, memori 32 GB) dengan disk 128GB (sekitar 77GB gratis). Kami merekomendasikan jenis pekerja ini untuk pekerjaan intensif memori dan pekerjaan yang menjalankan transformasi pembelajaran mesin. • **`G.4X`** — Ketika Anda memilih jenis ini, maka Anda juga memberikan nilai untuk Jumlah pekerja. Setiap pekerja memetakan ke 4 DPU (16 vCPU, memori 64 GB) dengan disk 256GB (sekitar 235GB gratis). Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru di

Properti	Deskripsi
	AWS Wilayah berikut: AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (Oregon), Asia Pasifik (Singapura), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Kanada (Tengah), Eropa (Frankfurt), Eropa (Irlandia), dan Eropa (Stockholm). • **`G.8X`** — Ketika Anda memilih jenis ini, maka Anda juga memberikan nilai untuk Jumlah pekerja. Setiap pekerja memetakan ke 8 DPU (32 vCPU, memori 128 GB) dengan disk 512GB (sekitar 487GB gratis). Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru, di AWS Wilayah yang sama seperti yang didukung untuk tipe `G.4X` pekerja.
Jam DPU	Perkiraan jumlah DPU yang digunakan untuk menjalankan eksekusi tugas. Sebuah DPU adalah ukuran relatif dari kekuatan pemrosesan. DPU digunakan untuk menentukan biaya menjalankan tugas Anda. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

Anda dapat memilih eksekusi tugas dalam daftar dan melihat informasi tambahan. Pilih eksekusi tugas, dan lakukan salah satu hal berikut ini:

- Pilih menu Tindakan dan opsi Lihat tugas untuk melihat tugas di editor visual.
- Pilih menu Tindakan dan opsi Hentikan eksekusi untuk menghentikan eksekusi tugas saat ini.
- Pilih tombol Lihat CloudWatch log untuk melihat log jalankan pekerjaan untuk pekerjaan itu.
- Pilih Lihat detail untuk melihat halaman detail pekerjaan yang dijalankan.

Melihat log eksekusi tugas

Anda dapat melihat log tugas dengan berbagai cara:

- Pada halaman Monitoring, di tabel Job running, pilih job run, lalu pilih View CloudWatch logs.
- Dalam editor tugas visual, pada tab Eksekusi untuk sebuah tugas, pilih hyperlink untuk melihat log:
  - Log — Tautan ke log tugas Apache Spark yang ditulis ketika pencatatan log terus menerus diaktifkan untuk sebuah eksekusi tugas. Saat Anda memilih tautan ini, Anda akan dibawa ke Amazon CloudWatch log di grup `/aws-glue/jobs/logs-v2` log. Secara default, log tersebut mengecualikan pesan heartbeat Apache Hadoop BEARN yang tak berguna dan pesan driver atau log pelaksana Apache Spark. Untuk informasi selengkapnya tentang pencatatan log berkelanjutan, lihat Pencatatan Log Berkelanjutan untuk Tugas AWS Glue di Panduan Developer AWS Glue.
  - Log kesalahan — Tautan ke log yang ditulis ke `stderr` untuk eksekusi tugas ini. Bila Anda memilih tautan ini, tautan ini akan membawa Anda ke log Amazon CloudWatch di grup log `/aws-glue/jobs/error`. Anda dapat menggunakan log ini untuk melihat detail tentang kesalahan yang ditemui selama eksekusi tugas.
  - Log output — Tautan ke log yang ditulis ke `stdout` untuk eksekusi tugas ini. Bila Anda memilih tautan ini, tautan ini akan membawa Anda ke log Amazon CloudWatch di grup log `/aws-glue/jobs/output`. Anda dapat menggunakan log ini untuk melihat semua detail tentang tabel yang dibuat di AWS Glue Data Catalog dan kesalahan apa pun yang ditemui.

Melihat detail sebuah eksekusi tugas

Anda dapat memilih tugas di daftar Eksekusi tugas di halaman Pemantauan, dan kemudian memilih Lihat detail eksekusi untuk melihat informasi detail untuk eksekusi tugas tersebut.

Informasi yang ditampilkan pada halaman detail eksekusi tugas meliputi:

Properti	Deskripsi
Nama tugas	Nama pekerjaannya.
Status Eksekusi	Status eksekusi tugas saat ini. Nilai dapat berupa: • `STARTING`

Properti	Deskripsi
	• RUNNING • STOPPING • STOPPED • SUCCEEDED • FAILED • TIMEOUT
Versi Glue	AWS GlueVersi yang digunakan oleh job run.
Upaya terbaru	Jumlah percobaan ulang otomatis untuk pekerjaan ini dijalankan.
Waktu mulai	Tanggal dan waktu saat eksekusi tugas ini dimulai.
Waktu akhir	Tanggal dan waktu saat eksekusi tugas ini selesai.
Waktu pemulaian	Jumlah waktu yang dihabiskan untuk mempersiapkan diri untuk menjalankan pekerjaan.
Waktu eksekusi	Jumlah waktu yang dihabiskan untuk menjalank an skrip pekerjaan.
Nama pemicu	Nama pemicu yang terkait dengan pekerjaan.
Terakhir diubah pada	Tanggal ketika pekerjaan terakhir diubah.
Konfigurasi keamanan	Konfigurasi keamanan untuk pekerjaan tersebut, yang mencakup enkripsi Amazon S3, enkripsi, dan pengaturan CloudWatch enkripsi bookmark pekerjaan.
Waktu habis	Nilai ambang batas waktu kerja berjalan.

Properti	Deskripsi
Kapasitas yang dialokasikan	Jumlah unit pemrosesan data (DPU) AWS Glue yang dialokasikan untuk eksekusi tugas ini. Untuk informasi selengkapnya tentang perencanaan kapasitas, lihat Memantau Perencanaan Kapasitas DPU di Panduan Developer AWS Glue.
Kapasitas maksimum	Kapasitas maksimum yang tersedia untuk eksekusi tugas.
Jumlah pekerja	Jumlah pekerja yang digunakan untuk menjalankan pekerjaan.

Properti	Deskripsi
Jenis pekerja	Jenis pekerja yang telah ditetapkan sebelumnya yang diperuntukkan untuk eksekusi tugas. Nilai bisa `G.1X` atau`G.2X`. • **`G.1X`** — Ketika Anda memilih jenis ini, maka Anda juga memberikan nilai untuk Jumlah pekerja. Setiap pekerja memetakan ke 1 DPU (4 vCPU, memori 16 GB, disk 64 GB), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk tugas yang membutuhkan banyak memori. Ini adalah tipe Pekerja default untuk AWS Glue versi 2.0 atau pekerjaan yang lebih baru. • **`G.2X`** — Ketika Anda memilih jenis ini, maka Anda juga memberikan nilai untuk Jumlah pekerja. Setiap pekerja memetakan ke 2 DPU (8 vCPU, memori 32 GB, disk 128 GB), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan intensif memori dan pekerjaan yang menjalankan transformasi pembelajaran mesin.
Log	Sebuah link ke log pekerjaan untuk logging berkelanjutan (`/aws-glue/jobs/logs-v2`).
Log Output	Sebuah link ke file log output pekerjaan (`/aws-glue/jobs/output`).
Log kesalahan	Sebuah link ke file log kesalahan pekerjaan (`/aws-glue/jobs/error`).

Anda juga dapat melihat item tambahan berikut, yang tersedia saat Anda melihat informasi untuk menjalankan pekerjaan terbaru. Untuk informasi selengkapnya, lihat the section called “Melihat informasi untuk eksekusi tugas terbaru”.

- Argumen masukan
- Log terus menerus
- Metrik — Anda dapat melihat visualisasi metrik dasar. Untuk informasi selengkapnya tentang metrik yang disertakan, lihatthe section called “Melihat Amazon CloudWatch metrik untuk menjalankan pekerjaan Spark”.
- Spark UI - Anda dapat memvisualisasikan log Spark untuk pekerjaan Anda di UI Spark. Untuk informasi selengkapnya tentang penggunaan UI Web Spark, lihatthe section called “Pemantauan dengan Spark UI”. Aktifkan fitur ini dengan mengikuti prosedur dithe section called “Mengaktifkan UI Spark untuk pekerjaan”.

Melihat Amazon CloudWatch metrik untuk menjalankan pekerjaan Spark

Pada halaman detail untuk menjalankan pekerjaan, di bawah bagian Run details, Anda dapat melihat metrik pekerjaan. AWS Glue Studiomengirimkan metrik pekerjaan Amazon CloudWatch untuk setiap pekerjaan yang dijalankan.

AWS Glue melaporkan metrik ke Amazon CloudWatch setiap 30 detik. Metrik AWS Glue merupakan nilai delta dari nilai yang dilaporkan sebelumnya. Jika sesuai, dasbor metrik meng-agregat (jumlah) nilai 30 detik untuk mendapatkan nilai untuk seluruh menit terakhir. Namun, metrik Apache Spark yang AWS Glue diteruskan ke umumnya Amazon CloudWatch merupakan nilai absolut yang mewakili keadaan saat ini pada saat dilaporkan.

> **Note**
>
> Anda harus mengonfigurasi akun Anda untuk mengakses Amazon CloudWatch,.

Metrik ini memberikan informasi tentang eksekusi tugas Anda, seperti:

- Gerakan data ETL — Jumlah byte yang dibaca dari atau ditulis ke Amazon S3.
- Profil Memori: Tumpukan yang digunakan — Jumlah byte memori yang digunakan oleh tumpukan mesin virtual Java (JVM).

- Profil Memori: Penggunaan tumpukan — Fraksi memori (skala: 0–1), ditampilkan dalam persentase, yang digunakan oleh tumpukan JVM.
- Beban CPU — Fraksi beban sistem CPU yang digunakan (skala: 0–1), ditampilkan dalam persentase.

Melihat Amazon CloudWatch metrik untuk menjalankan pekerjaan Ray

Pada halaman detail untuk menjalankan pekerjaan, di bawah bagian Run details, Anda dapat melihat metrik pekerjaan. AWS Glue Studiomengirimkan metrik pekerjaan Amazon CloudWatch untuk setiap pekerjaan yang dijalankan.

AWS Glue melaporkan metrik ke Amazon CloudWatch setiap 30 detik. Metrik AWS Glue merupakan nilai delta dari nilai yang dilaporkan sebelumnya. Jika sesuai, dasbor metrik meng-agregat (jumlah) nilai 30 detik untuk mendapatkan nilai untuk seluruh menit terakhir. Namun, metrik Apache Spark yang AWS Glue diteruskan ke umumnya Amazon CloudWatch merupakan nilai absolut yang mewakili keadaan saat ini pada saat dilaporkan.

> **Note**
>
> Anda harus mengonfigurasi akun Anda untuk mengakses Amazon CloudWatch, seperti yang dijelaskan dalam.

Dalam pekerjaan Ray, Anda dapat melihat grafik metrik agregat berikut. Dengan ini, Anda dapat membangun profil cluster dan tugas Anda, dan dapat mengakses informasi terperinci tentang setiap node. Data deret waktu yang mendukung grafik ini tersedia CloudWatch untuk analisis lebih lanjut.

Profil Tugas: Status Tugas

Menunjukkan jumlah tugas Ray dalam sistem. Setiap siklus hidup tugas diberikan deret waktunya sendiri.

Profil Tugas: Nama Tugas

Menunjukkan jumlah tugas Ray dalam sistem. Hanya tugas yang tertunda dan aktif yang ditampilkan. Setiap jenis tugas (dengan nama) diberikan deret waktunya sendiri.

Profil Cluster: CPU yang digunakan

Menunjukkan jumlah core CPU yang digunakan. Setiap node diberikan deret waktunya sendiri. Node diidentifikasi oleh alamat IP, yang bersifat sementara dan hanya digunakan untuk identifikasi.

Profil Cluster: Penggunaan memori penyimpanan objek

Menunjukkan penggunaan memori oleh cache objek Ray. Setiap lokasi memori (memori fisik, cache pada disk, dan tumpah di Amazon S3) diberikan deret waktunya sendiri. Toko objek mengelola penyimpanan data di semua node di cluster. Untuk informasi selengkapnya, lihat Objek dalam dokumentasi Ray.

Profil Cluster: Jumlah simpul

Menunjukkan jumlah node yang disediakan untuk cluster.

Detail Node: Penggunaan CPU

Menunjukkan pemanfaatan CPU pada setiap node sebagai persentase. Setiap seri menunjukkan persentase agregat penggunaan CPU di semua core pada node.

Detail Node: Penggunaan memori

Menunjukkan penggunaan memori pada setiap node dalam GB. Setiap seri menunjukkan memori yang dikumpulkan antara semua proses pada node, termasuk tugas Ray dan proses penyimpanan Plasma. Ini tidak akan mencerminkan objek yang disimpan ke disk atau tumpah ke Amazon S3.

Detail Node: Penggunaan disk

Menunjukkan penggunaan disk pada setiap node dalam GB.

Detail Node: Kecepatan I/O Disk

Menampilkan disk I/O pada setiap node dalam KB/s.

Detail Node: Throughput I/O Jaringan

Menunjukkan jaringan I/O pada setiap node dalam KB/s.

Detail Node: Penggunaan CPU oleh komponen Ray

Menunjukkan penggunaan CPU dalam pecahan inti. Setiap komponen sinar pada setiap node diberikan deret waktunya sendiri.

Detail Node: Penggunaan memori oleh komponen Ray

Menunjukkan penggunaan memori di GiB. Setiap komponen sinar pada setiap node diberikan deret waktunya sendiri.

Mendeteksi dan memproses data sensitif

Transformasi PII Detect mengidentifikasi Informasi Identifikasi Pribadi (PII) di sumber data Anda. Anda memilih entitas PII untuk diidentifikasi, bagaimana Anda ingin data dipindai, dan apa yang harus dilakukan dengan entitas PII yang telah diidentifikasi oleh transformasi Detect PII.

Transformasi Detect PII menyediakan kemampuan untuk mendeteksi, menutupi, atau menghapus entitas yang Anda tentukan, atau yang telah ditentukan sebelumnya oleh. AWS Ini memungkinkan Anda untuk meningkatkan kepatuhan dan mengurangi tanggung jawab. Misalnya, Anda mungkin ingin memastikan bahwa tidak ada informasi identitas pribadi dalam data Anda yang dapat dibaca dan ingin menutupi nomor jaminan sosial dengan string tetap (seperti xxx-xx-xxxx), nomor telepon, atau alamat.

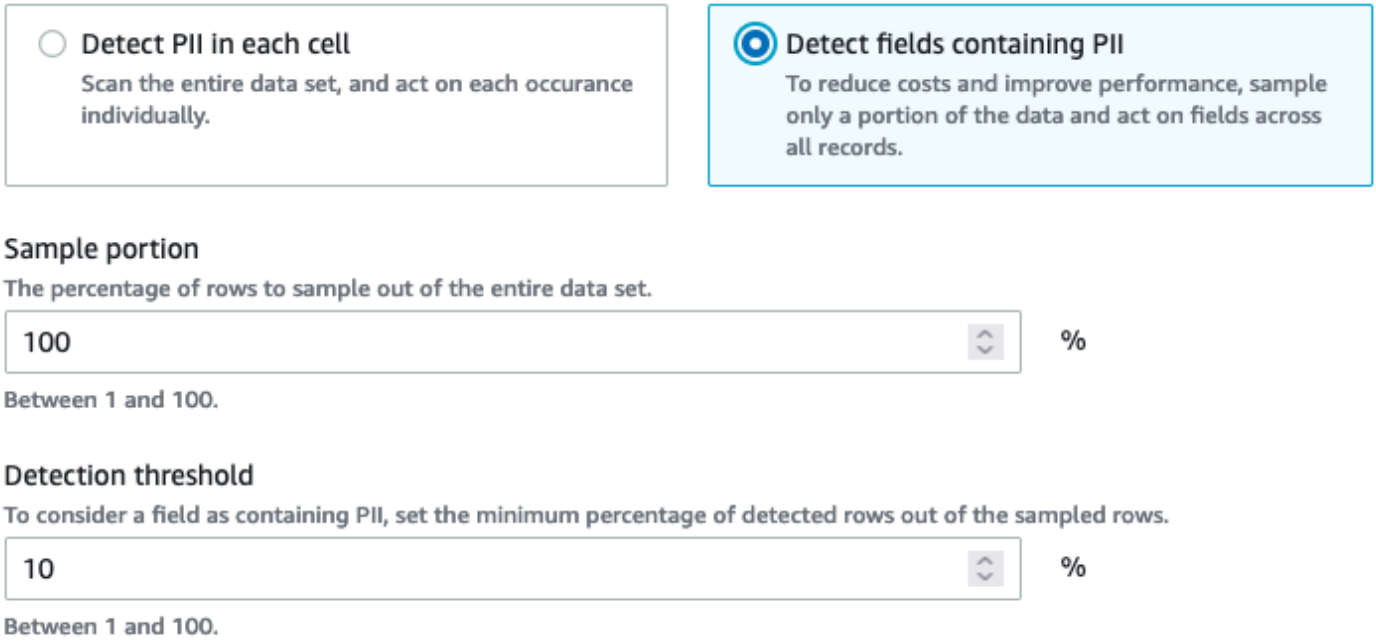
Untuk bekerja dengan data sensitif di luarAWS Glue Studio, lihat Menggunakan Deteksi Data Sensitif di luar AWS Glue Studio

Topik

- Memilih bagaimana Anda ingin data dipindai
- Memilih entitas PII untuk dideteksi
- Menentukan tingkat sensitivitas deteksi
- Memilih apa yang harus dilakukan dengan data PII yang diidentifikasi
- Menambahkan penggantian aksi berbutir halus

Memilih bagaimana Anda ingin data dipindai

Saat Anda memindai kumpulan data Anda untuk data sensitif seperti informasi identitas pribadi (PII), Anda dapat memilih untuk mendeteksi PII di setiap baris atau mendeteksi kolom yang berisi data PII.



Saat Anda memilih Deteksi PII di setiap sel, Anda memilih untuk memindai semua baris di sumber data. Ini adalah pemindaian komprehensif untuk memastikan bahwa entitas PII diidentifikasi.

Saat Anda memilih Mendeteksi bidang yang berisi PII, Anda memilih untuk memindai sampel baris untuk entitas PII. Ini adalah cara untuk menjaga biaya dan sumber daya tetap rendah sambil juga mengidentifikasi bidang tempat entitas PII ditemukan.

Ketika Anda memilih untuk mendeteksi bidang yang berisi PII, Anda dapat mengurangi biaya dan meningkatkan kinerja dengan mengambil sampel sebagian baris. Memilih opsi ini akan memungkinkan Anda untuk menentukan opsi tambahan:

- Bagian sampel: Ini memungkinkan Anda untuk menentukan persentase baris untuk sampel. Misalnya, jika Anda memasukkan '50', Anda menentukan bahwa Anda menginginkan 50 persen baris yang dipindai untuk entitas PII.
- Ambang deteksi: Ini memungkinkan Anda menentukan persentase baris yang berisi entitas PII agar seluruh kolom diidentifikasi memiliki entitas PII. Misalnya, jika Anda memasukkan '10', Anda menentukan bahwa jumlah entitas PII, Telepon AS, dalam baris yang dipindai harus 10 persen atau lebih besar agar bidang tersebut diidentifikasi memiliki entitas PII, Telepon AS. Jika persentase baris yang berisi entitas PII kurang dari 10 persen, bidang tersebut tidak akan diberi label memiliki entitas PII, Telepon AS, di dalamnya.

Memilih entitas PII untuk dideteksi

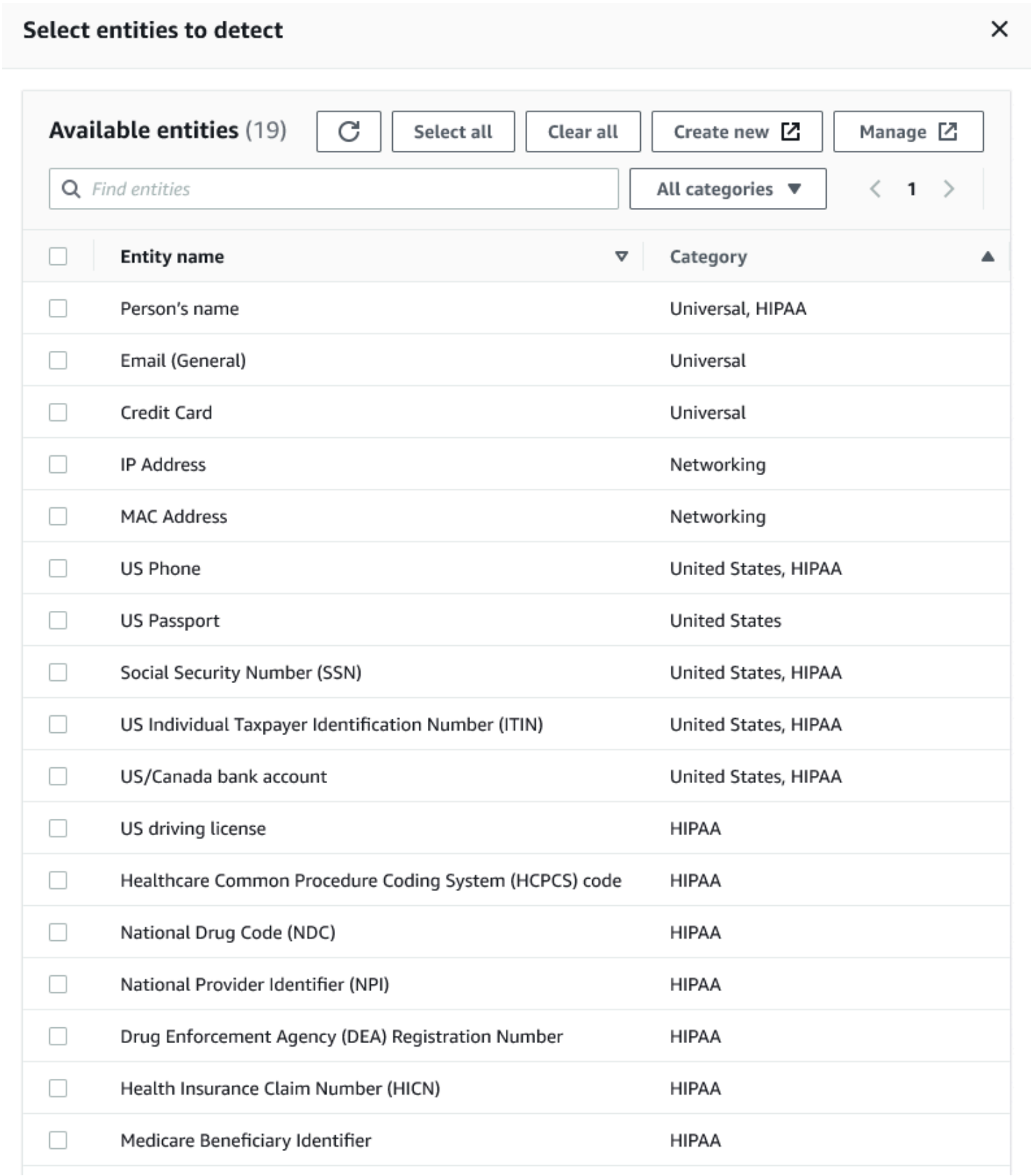
Jika Anda memilih Deteksi PII di setiap sel, Anda dapat memilih salah satu dari tiga opsi:

- Semua pola PII yang tersedia - ini termasuk AWS entitas.
- Pilih kategori - ketika Anda memilih kategori, pola PII akan secara otomatis menyertakan pola dalam kategori yang Anda pilih.
- Pilih pola tertentu - Hanya pola yang Anda pilih yang akan terdeteksi.

Untuk daftar lengkap tipe data sensitif terkelola, lihat Tipe data terkelola.

Pilih dari semua pola PII yang tersedia

Jika Anda memilih Semua pola PII yang tersedia, pilih entitas yang telah ditentukan sebelumnya oleh. AWS Anda dapat memilih satu, lebih dari satu, atau semua entitas.


Select entities to detect
Available entities (19)
Select all
Clear all
Create new
Manage
Find entities
All categories
1
Entity name
Category
Person's name
Universal, HIPAA
Email (General)
Universal
Credit Card
Universal
IP Address
Networking
MAC Address
Networking
US Phone
United States, HIPAA
US Passport
United States
Social Security Number (SSN)
United States, HIPAA
US Individual Taxpayer Identification Number (ITIN)
United States, HIPAA
US/Canada bank account
United States, HIPAA
US driving license
HIPAA
Healthcare Common Procedure Coding System (HCPCS) code
HIPAA
National Drug Code (NDC)
HIPAA
National Provider Identifier (NPI)
HIPAA
Drug Enforcement Agency (DEA) Registration Number
HIPAA
Health Insurance Claim Number (HICN)
HIPAA
Medicare Beneficiary Identifier
HIPAA

Pilih kategori

Jika Anda memilih Pilih kategori sebagai pola PII untuk dideteksi, Anda dapat memilih dari opsi di menu tarik-turun. Perhatikan bahwa beberapa entitas dapat termasuk dalam lebih dari satu kategori. Misalnya, nama Orang adalah entitas yang termasuk dalam kategori Universal dan HIPAA.

- Universal (contoh: Email, Kartu Kredit)
- HIPAA (contoh: Surat Izin Mengemudi AS, Kode Sistem Pengkodean Prosedur Umum Kesehatan (HCPCS))
- Jaringan (contoh: Alamat IP, Alamat MAC)
- Argentina
- Australia
- Austria
- Belgium
- Bosnia
- Bulgaria
- Kanada
- Chili
- Kolombia
- Croatia
- Cyprus
- Ceko
- Denmark
- Estonia
- Finland
- France
- Germany
- Greece
- Hungary
- Ireland
- Korea
- Jepang

- Meksiko
- Netherlands
- Selandia Baru
- Norwegia
- Portugal
- Romania
- Singapura
- Slovakia
- Slovenia
- Spain
- Sweden
- Swiss
- Turki
- Ukraina
- Amerika Serikat
- Britania Raya
- Venezuela

Pilih pola tertentu

Jika Anda memilih Pilih pola tertentu sebagai pola PII yang akan dideteksi, Anda dapat mencari atau menelusuri dari daftar pola yang telah Anda buat, atau membuat pola entitas deteksi baru.

Langkah-langkah di bawah ini menjelaskan cara membuat pola kustom baru untuk mendeteksi data sensitif. Anda akan membuat pola kustom dengan memasukkan nama untuk pola kustom, menambahkan ekspresi reguler, dan secara opsional, menentukan kata-kata konteks.

1. Untuk membuat pola baru, klik tombol Create new.

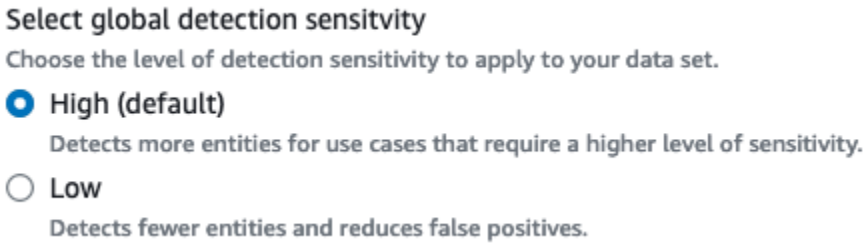
2. Di halaman Create detection entity, masukkan nama entitas dan ekspresi reguler. Ekspresi reguler (Regex) adalah apa yang AWS Glue akan digunakan untuk mencocokkan entitas.
3. Klik Validasi. Jika validasi berhasil, Anda akan melihat pesan konfirmasi yang menyatakan bahwa string adalah ekspresi reguler yang valid. Jika validasi tidak berhasil, Anda akan melihat pesan yang menyatakan bahwa string tidak sesuai dengan pemformatan yang tepat dan literal karakter, operator, atau konstruksi yang diterima.
4. Anda dapat memilih untuk menambahkan kata-kata Konteks selain ekspresi reguler. Kata-kata konteks dapat meningkatkan kemungkinan kecocokan. Ini dapat berguna dalam kasus di mana nama bidang tidak deskriptif entitas. Misalnya, nomor jaminan sosial dapat diberi nama 'SSN' atau 'SS'. Menambahkan kata-kata konteks ini dapat membantu mencocokkan entitas.
5. Klik Buat untuk membuat entitas deteksi. Entitas apa pun yang dibuat terlihat di AWS Glue Studio konsol. Klik entitas Deteksi di menu navigasi sebelah kiri.

   Anda dapat mengedit, menghapus, atau membuat entitas deteksi dari halaman entitas Deteksi. Anda juga dapat mencari pola menggunakan bidang pencarian.

Menentukan tingkat sensitivitas deteksi

Anda dapat mengatur tingkat sensitivitas saat menggunakan mendeteksi data sensitif.

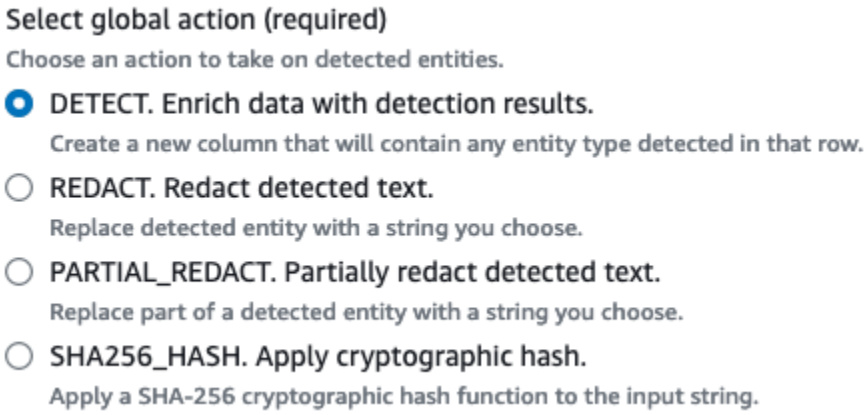
- Tinggi — (Default) Mendeteksi lebih banyak entitas untuk kasus penggunaan yang memerlukan tingkat sensitivitas yang lebih tinggi. Semua AWS Glue pekerjaan yang dibuat setelah November 2023 secara otomatis ikut serta dalam pengaturan ini.
- Rendah - Mendeteksi lebih sedikit entitas dan mengurangi positif palsu.



Memilih apa yang harus dilakukan dengan data PII yang diidentifikasi

Jika Anda memilih untuk mendeteksi PII di seluruh sumber data, Anda dapat memilih tindakan global untuk diterapkan:

- Perkaya data dengan hasil deteksi: Jika Anda memilih Deteksi PII di setiap sel, Anda dapat menyimpan entitas yang terdeteksi ke dalam kolom baru.
- Menyunting teks yang terdeteksi: Anda dapat mengganti nilai PII yang terdeteksi dengan string yang Anda tentukan di bidang input teks Mengganti opsional. Jika tidak ada string yang ditentukan, entitas PII yang terdeteksi diganti dengan '*******'.
- Menyunting sebagian teks yang terdeteksi: Anda dapat mengganti bagian dari nilai PII yang terdeteksi dengan string yang Anda pilih. Ada dua opsi yang mungkin: membiarkan ujungnya terbuka kedok atau menutupi dengan memberikan pola regex eksplisit. Fitur ini tidak tersedia di AWS Glue 2.0.
- Terapkan hash kriptografi: Anda dapat meneruskan nilai PII yang terdeteksi ke fungsi hash kriptografi SHA-256 dan mengganti nilainya dengan output fungsi.



Perbedaan antara AWS Glue versi 2.0 dan 3.0+

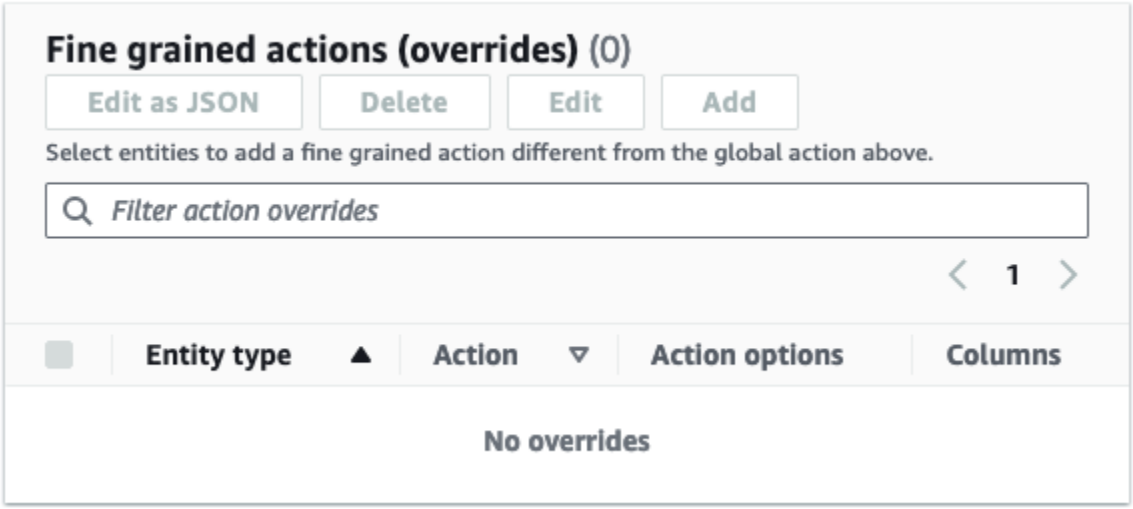
AWS Glue2.0 pekerjaan akan mengembalikan yang baru DataFrame dengan informasi PII yang terdeteksi untuk setiap kolom di kolom tambahan. Setiap redaksi atau pekerjaan hash terlihat dalam AWS Glue skrip di tab visual.

AWS GluePekerjaan 3.0 dan 4.0 akan mengembalikan yang baru DataFrame dengan kolom tambahan yang sama ini. Sebuah kunci baru untuk “ActionUsed” hadir dan dapat menjadi salah satu dari`DETECT`,, `REDACT`PARTIAL_REDACT, atau. `SHA256_HASH` Jika tindakan masking dipilih, DataFrame akan mengembalikan data dengan data sensitif yang disamarkan.

Menambahkan penggantian aksi berbutir halus

Deteksi tambahan dan pengaturan tindakan dapat ditambahkan ke tabel penggantian tindakan berbutir halus. Ini memungkinkan Anda untuk:

- Sertakan atau kecualikan kolom tertentu dari deteksi - Skema yang disimpulkan pada sumber data akan mengisi tabel dengan kolom yang tersedia.
- Tentukan setelan spesifik yang lebih halus daripada menggunakan tindakan global — Misalnya, Anda dapat menentukan setelan teks redaksi yang berbeda untuk jenis entitas yang berbeda.
- Tentukan tindakan yang berbeda dari tindakan global — Jika tindakan yang berbeda ingin diterapkan pada tipe data sensitif yang berbeda, itu dapat dilakukan di sini. Perhatikan bahwa dua edit-in-place tindakan berbeda (redaksi dan hashing) tidak dapat digunakan pada kolom yang sama, tetapi deteksi selalu dapat digunakan.



Mengelola pekerjaan ETL dengan AWS Glue Studio

Anda dapat menggunakan antarmuka grafis sederhana AWS Glue Studio untuk mengelola pekerjaan ETL Anda. Dengan menggunakan menu navigasi, pilih Tugas untuk melihat halaman Tugas. Di halaman ini, Anda dapat melihat semua pekerjaan yang telah Anda buat baik dengan AWS Glue Studio atau AWS Glue konsol. Anda dapat melihat, mengelola, dan menjalankan tugas Anda di halaman ini.

Anda juga dapat melakukan tugas-tugas berikut di halaman ini:

- Memulai eksekusi tugas
- Jadwalkan eksekusi tugas
- Mengelola jadwal tugas
- Menghentikan eksekusi tugas

- Melihat tugas Anda
- Melihat informasi untuk eksekusi tugas terbaru
- Lihat skrip tugas
- Mengubah properti tugas
- Simpan tugas
- Klon tugas
- Menghapus tugas

Memulai eksekusi tugas

DiAWS Glue Studio, Anda dapat menjalankan pekerjaan Anda sesuai permintaan. Sebuah tugas dapat berjalan beberapa kali, dan setiap kali Anda menjalankan tugas, AWS Glue mengumpulkan informasi tentang aktivitas dan performa tugas tersebut. Informasi ini disebut sebagai eksekusi tugas dan diidentifikasi oleh ID eksekusi tugas.

Anda dapat memulai pekerjaan dengan cara-cara berikut diAWS Glue Studio:

- Pada halaman Tugas, pilih tugas yang ingin Anda mulai, kemudian pilih tombol Jalankan tugas.
- Jika Anda melihat sebuah tugas di editor visual dan tugas tersebut telah disimpan, maka Anda dapat memilih opsi Jalankan untuk memulai eksekusi tugas.

Untuk informasi selengkapnya tentang eksekusi tugas, lihat Bekerja dengan Tugas pada Konsol AWS Glue di Panduan Developer AWS Glue.

Jadwalkan eksekusi tugas

DiAWS Glue Studio, Anda dapat membuat jadwal agar pekerjaan Anda berjalan pada waktu tertentu. Anda dapat menentukan batasan-batasan, seperti berapa kali tugas berjalan, pada hari apa tugas tersebut berjalan, dan jam berapa berjalan. Batasan-batasan ini didasarkan pada `cron` dan memiliki keterbatasan yang sama seperti `cron`. Sebagai contoh, jika Anda memilih untuk menjalankan tugas Anda pada hari ke 31 setiap bulan, ingatlah bahwa ada bulan yang tidak terdiri dari 31 hari. Untuk informasi selengkapnya tentang `cron`, lihat Ekspresi Cron dalam Panduan Developer AWS Glue.

Untuk menjalankan tugas sesuai jadwal

1. Buat sebuah jadwal tugas menggunakan salah satu metode berikut:

- Pada halaman Tugas, pilih tugas yang ingin Anda buatkan jadwal, pilih Tindakan, lalu pilih Jadwalkan tugas.
- Jika Anda melihat sebuah tugas di editor visual dan tugas tersebut telah disimpan, pilih tab Jadwal. Kemudian pilih Buat Jadwal.

2. Di halaman Jadwalkan eksekusi tugas, masukkan informasi berikut:

- Nama: Masukkan nama jadwal tugas Anda.
- Frekuensi: Masukkan frekuensi untuk jadwal tugas. Anda dapat memilih yang berikut ini:
  - Jaman: Tugas akan berjalan setiap jam, dimulai pada menit tertentu. Anda dapat menentukan Menit dari jam di mana tugas tersebut harus berjalan. Secara default, ketika Anda memilih per jam, tugas berjalan pada awal jam (menit 0).
  - Harian: Tugas akan berjalan setiap hari, dimulai pada suatu waktu. Anda dapat menentukan Menit dari jam di mana tugas tersebut harus berjalan dan Jam mulai untuk tugas. Jam ditentukan dengan menggunakan jam 23-jam, di mana Anda menggunakan angka 13 sampai 23 untuk jam sore hari. Nilai default untuk menit dan jam adalah 0, yang berarti bahwa jika Anda memilih Per Hari, maka tugas secara default akan berjalan pada tengah malam.
  - Mingguan: Tugas akan berjalan setiap minggu pada satu atau beberapa hari. Selain pengaturan yang sama sebagaimana yang dijelaskan sebelumnya untuk Per Hari, Anda dapat memilih hari dalam seminggu di mana tugas akan berjalan. Anda dapat memilih satu hari atau beberapa hari.
  - Bulanan: Tugas akan berjalan setiap bulan pada hari tertentu. Selain pengaturan yang sama yang dijelaskan sebelumnya untuk Harian, Anda dapat memilih hari bulan di mana tugas akan berjalan. Tentukan hari dengan nilai numerik dari 1 sampai 31. Jika Anda memilih hari yang tidak ada dalam satu bulan, misalnya tanggal ^{30} Februari, maka tugas tidak akan berjalan di bulan itu.
  - Kustom: Masukkan ekspresi untuk jadwal tugas Anda menggunakan sintaksis `cron`. Ekspresi cron memungkinkan Anda membuat jadwal yang lebih rumit, seperti hari terakhir setiap bulan (bukan hari tertentu dalam sebulan) atau setiap bulan ketiga pada tanggal ^{7} dan ^{21} dalam sebulan.

    Lihat Ekspresi Cron di Panduan Developer AWS Glue
- Deskripsi: Anda dapat secara opsional memasukkan deskripsi untuk jadwal tugas Anda. Jika Anda berencana untuk menggunakan jadwal yang sama untuk beberapa tugas, dengan memiliki deskripsi akan membuatnya menjadi lebih mudah untuk menentukan jadwal tugas.

3. Pilih Membuat jadwal untuk menyimpan jadwal tugas.
4. Setelah Anda membuat jadwal, pesan sukses akan muncul di bagian atas halaman konsol tersebut. Anda dapat memilih Detail tugas pada banner ini untuk melihat detail tugas. Ini akan membuka halaman editor tugas visual, dengan tab Jadwal yang dipilih.

Mengelola jadwal tugas

Setelah Anda membuat jadwal untuk sebuah tugas, Anda dapat membuka tugas tersebut di editor visual dan memilih tab Jadwal untuk mengelola jadwalnya.

Pada tab Jadwal editor visual, Anda dapat melakukan tugas-tugas berikut:

- Membuat sebuah jadwal baru.

  Pilih Membuat jadwal kerja, lalu masukkan informasi untuk jadwal Anda seperti yang dijelaskan di the section called “Jadwalkan eksekusi tugas”.
- Edit jadwal yang ada.

  Pilih jadwal yang ingin diedit, lalu pilih Tindakan diikuti dengan Edit jadwal. Ketika Anda memilih untuk mengedit jadwal yang ada, Frekuensi menunjukkan sebagai Kustom, dan jadwal ditampilkan sebagai ekspresi `cron`. Anda dapat mengubah ekspresi `cron` tersebut, atau menentukan sebuah jadwal baru menggunakan tombol Frekuensi. Setelah Anda selesai melakukan perubahan, pilih Perbarui jadwal.
- Menjeda jadwal aktif.

  Pilih sebuah jadwal yang aktif, lalu pilih Tindakan diikuti dengan Jeda jadwal. Jadwal akan langsung dinonaktifkan. Pilih tombol refresh (reload) untuk melihat status jadwal tugas yang sudah diperbarui.
- Lanjutkan jadwal yang dijeda.

  Pilih sebuah jadwal yang dinonaktifkan, lalu pilih Tindakan diikuti dengan Lanjutkan jadwal. Jadwal akan langsung diaktifkan. Pilih tombol refresh (reload) untuk melihat status jadwal tugas yang sudah diperbarui.
- Menghapus sebuah jadwal.

  Pilih sebuah jadwal yang ingin Anda hapus, kemudian pilih Tindakan diikuti dengan Hapus jadwal. Jadwal akan langsung dihapus. Pilih tombol refresh (reload) untuk melihat daftar jadwal tugas yang

sudah diperbarui. Jadwal akan menunjukkan status Menghapus sehingga jadwal benar-benar telah dihapus.

Menghentikan eksekusi tugas

Anda dapat menghentikan sebuah tugas sebelum eksekusi tugas selesai. Anda dapat memilih opsi ini jika Anda tahu bahwa tugas tidak dikonfigurasi dengan benar, atau jika tugas membutuhkan waktu terlalu lama untuk diselesaikan.

Pada halaman Pemantauan, di daftar Eksekusi tugas, pilih tugas yang ingin Anda hentikan, pilih Tindakan, lalu pilih Hentikan eksekusi.

Melihat tugas Anda

Anda dapat melihat semua tugas Anda di halaman Tugas. Anda dapat mengakses halaman ini dengan memilih Tugas di panel navigasi.

Pada halaman Tugas, Anda dapat melihat semua tugas yang dibuat di akun Anda. Daftar Tugas Anda menampilkan nama tugas, jenisnya, status eksekusi terakhir dari tugas tersebut, dan tanggal di mana tugas dibuat dan terakhir diubah. Anda dapat memilih nama tugas untuk melihat informasi detail untuk tugas tersebut.

Anda juga dapat menggunakan dasbor Pemantauan untuk melihat semua tugas Anda. Anda dapat mengakses dasbor tersebut dengan memilih Pemantauan di panel navigasi.

Menyesuaikan tampilan tugas

Anda dapat menyesuaikan bagaimana tugas ditampilkan dalam bagian Tugas Anda dari halaman Tugas. Selain itu, Anda dapat memasukkan teks di bidang teks pencarian untuk hanya menampilkan tugas dengan nama yang berisi teks tersebut.

Jika Anda memilih ikon pengaturan

di bagian Pekerjaan Anda, Anda dapat menyesuaikan cara AWS Glue Studio menampilkan informasi dalam tabel. Anda dapat memilih untuk mengurung baris teks di layar, mengubah jumlah tugas yang ditampilkan pada halaman, dan menentukan kolom mana yang akan ditampilkan.

Melihat informasi untuk eksekusi tugas terbaru

Tugas dapat berjalan beberapa kali karena ada data baru yang ditambahkan di lokasi sumber. Setiap kali tugas berjalan, ID unik ditetapkan pada eksekusi tugas, dan informasi tentang eksekusi tugas dikumpulkan. Anda dapat melihat informasi ini dengan menggunakan cara berikut:

- Pilih tab Eksekusi pada editor visual untuk melihat informasi eksekusi tugas untuk tugas yang saat ini ditampilkan.

  Pada tab Eksekusi (halaman Eksekusi tugas terbaru), ada kartu sebuah untuk setiap eksekusi tugas. Informasi yang ditampilkan pada tab Eksekusi termasuk:

  - ID eksekusi tugas
  - Jumlah percobaan untuk menjalankan tugas ini
  - Status dari eksekusi tugas
  - Waktu mulai dan akhir eksekusi tugas
  - Waktu aktif eksekusi tugas
  - Tautan ke file berkas log tugas
  - Tautan ke file berkas log kesalahan tugas
  - Kesalahan yang dikembalikan untuk tugas yang gagal
- Anda dapat memilih pekerjaan yang dijalankan untuk melihat informasi tambahan tentang pekerjaan tersebut, termasuk yang berikut ini:
  - Argumen masukan
  - Log terus menerus
  - Metrik — Anda dapat melihat visualisasi metrik dasar. Untuk informasi selengkapnya tentang metrik yang disertakan, lihatthe section called “Melihat Amazon CloudWatch metrik untuk menjalankan pekerjaan Spark”.
  - Spark UI - Anda dapat memvisualisasikan log Spark untuk pekerjaan Anda di UI Spark. Untuk informasi selengkapnya tentang penggunaan UI Web Spark, lihatthe section called “Pemantauan dengan Spark UI”. Aktifkan fitur ini dengan mengikuti prosedur dithe section called “Mengaktifkan UI Spark untuk pekerjaan”.

Anda dapat memilih Lihat detail untuk melihat informasi serupa di halaman detail menjalankan pekerjaan. Atau, Anda dapat menavigasi ke halaman rincian pekerjaan dijalankan melalui halaman Monitoring. Di panel navigasi, pilih Pemantauan. Gulir ke bawah ke bagian daftar Eksekusi tugas.

Pilih tugas dan kemudian pilih Tampilkan detail eksekusi. Kandungan dijelaskan dalam Melihat detail sebuah eksekusi tugas.

Untuk informasi selengkapnya tentang log tugas, lihat Melihat log eksekusi tugas.

Lihat skrip tugas

Setelah Anda memberikan informasi untuk semua node dalam pekerjaanAWS Glue Studio, buat skrip yang digunakan oleh pekerjaan untuk membaca data dari sumber, mengubah data, dan menulis data di lokasi target. Jika Anda menyimpan tugas tersebut, maka Anda dapat melihat skrip ini kapan saja.

Untuk melihat skrip yang dihasilkan untuk tugas Anda

1. Pilih Tugas di panel navigasi.
2. Pada halaman Tugas, di daftar Tugas Anda, pilih nama tugas yang ingin Anda tinjau. Atau, Anda dapat memilih sebuah tugas dalam daftar tersebut, pilih menu Tindakan, dan kemudian pilih Edit tugas.
3. Pada halaman editor visual, pilih tab Skrip di bagian atas untuk melihat skrip tugas.

   Jika Anda ingin mengedit skrip tugas, lihat AWS Glue panduan pemrograman.

Mengubah properti tugas

Simpul dalam diagram tugas menentukan tindakan yang dilakukan oleh tugas, tetapi ada beberapa properti yang juga dapat Anda konfigurasi untuk tugas tersebut. Properti ini menentukan lingkungan di mana tugas itu berjalan, sumber daya yang digunakan, pengaturan ambang batas, pengaturan keamanan, dan banyak lagi.

Untuk menyesuaikan lingkungan eksekusi tugas

1. Pilih Tugas di panel navigasi.
2. Pada halaman Tugas, di daftar Tugas Anda, pilih nama tugas yang ingin Anda tinjau.
3. Pada halaman editor visual, pilih tab Detail Tugas di bagian atas panel pengeditan tugas.
4. Ubah properti tugas, sesuai kebutuhan.

   Untuk informasi selengkapnya tentang properti tugas, lihat Mendefinisikan Properti Tugas di Panduan Developer AWS Glue.
5. Perluas bagian Properti lanjutan jika Anda perlu menentukan properti tugas tambahan ini:

- Nama berkas skrip — Nama file yang menyimpan skrip tugas di Amazon S3.
- Path skrip — Lokasi Amazon S3 di mana skrip tugas disimpan.
- Metrik Job — (Tidak tersedia untuk pekerjaan shell Python) Mengaktifkan pembuatan Amazon CloudWatch metrik saat pekerjaan ini berjalan.
- Pencatatan berkelanjutan — (Tidak tersedia untuk pekerjaan shell Python) Mengaktifkan logging berkelanjutan CloudWatch, sehingga log tersedia untuk dilihat sebelum pekerjaan selesai.
- Jalur log UI Spark dan Spark UI — (Tidak tersedia untuk pekerjaan shell Python) Mengaktifkan penggunaan Spark UI untuk memantau pekerjaan ini dan menentukan lokasi untuk log UI Spark.
- Konkurensi maksimum — Mengatur jumlah maksimum eksekusi bersamaan yang diperbolehkan untuk tugas ini.
- Path sementara — Lokasi direktori kerja di Amazon S3 di mana hasil antara sementara ditulis ketika AWS Glue menjalankan skrip tugas.
- Ambang batas notifikasi penundaan (menit) — Menentukan ambang batas penundaan untuk tugas. Jika pekerjaan berjalan lebih lama dari yang ditentukan oleh ambang batas, maka AWS Glue kirimkan pemberitahuan penundaan untuk pekerjaan tersebut CloudWatch.
- Konfigurasi keamanan dan Enkripsi sisi server — Gunakan bidang ini untuk memilih opsi enkripsi untuk tugas tersebut.
- Gunakan Katalog Data Glue sebagai metastore Hive — Pilih opsi ini jika Anda ingin menggunakan opsi AWS Glue Data Catalog ini sebagai alternatif untuk Apache Hive Metastore.
- Koneksi jaringan tambahan — Untuk sumber data di VPC, Anda dapat menentukan koneksi tipe `Network` untuk memastikan bahwa tugas Anda mengakses data Anda melalui VPC.
- Jalur pustaka Python, Jalur jar dependen (Tidak tersedia untuk pekerjaan shell Python), atau Jalur file yang direferensikan - Gunakan bidang ini untuk menentukan lokasi file tambahan yang digunakan oleh pekerjaan saat menjalankan skrip.
- Parameter Tugas — Anda dapat menambahkan satu set pasangan nilai-kunci yang diberikan sebagai parameter bernama untuk skrip tugas. Dalam panggilan Python ke AWS Glue APIs, cara terbaik adalah memberikan parameter secara eksplisit berdasarkan nama. Untuk informasi selengkapnya tentang menggunakan parameter di skrip tugas, lihat Memberikan dan Mengakses Parameter Python di AWS Glue di Panduan Developer AWS Glue.

- Tag — Anda dapat menambahkan tanda ke tugas tersebut untuk membantu Anda mengorganisasi dan mengidentifikasinya.

6. Setelah Anda selesai memodifikasi properti tugas, simpan tugas tersebut.

Simpan file shuffle Spark di Amazon S3

Beberapa pekerjaan ETL memerlukan membaca dan menggabungkan informasi dari beberapa partisi, misalnya, saat menggunakan transformasi gabungan. Operasi ini disebut sebagai shuffling. Selama shuffle, data ditulis ke disk dan ditransfer ke seluruh jaringan. Dengan AWS Glue versi 3.0, Anda dapat mengonfigurasi Amazon S3 sebagai lokasi penyimpanan untuk file-file ini. AWS Gluemenyediakan pengelola acak yang menulis dan membaca file acak ke dan dari Amazon S3. Menulis dan membaca file shuffle dari Amazon S3 lebih lambat (sebesar 5% -20%) dibandingkan dengan disk lokal (atau Amazon EBS yang sangat dioptimalkan untuk Amazon EC2). Namun, Amazon S3 menyediakan kapasitas penyimpanan tak terbatas, jadi Anda tidak perlu khawatir tentang kesalahan "`No space left on device`" saat menjalankan pekerjaan Anda.

Untuk mengonfigurasi pekerjaan Anda untuk menggunakan Amazon S3 untuk file acak

1. Pada halaman Pekerjaan, dalam daftar Pekerjaan Anda, pilih nama pekerjaan yang ingin Anda ubah.
2. Pada halaman editor visual, pilih tab Detail Tugas di bagian atas panel pengeditan tugas.

   Gulir ke bawah ke bagian Parameter Job.
3. Tentukan pasangan kunci-nilai berikut.

   - `--write-shuffle-files-to-s3` — `true`

     Ini adalah parameter utama yang mengonfigurasi pengelola acak AWS Glue untuk menggunakan bucket Amazon S3 untuk menulis dan membaca data acak. Secara default, parameter ini memiliki nilai`false`.
   - (Opsional) `--write-shuffle-spills-to-s3` - `true`

     Parameter ini memungkinkan Anda untuk membongkar file tumpahan ke bucket Amazon S3, yang memberikan ketahanan tambahan untuk pekerjaan Spark Anda di. AWS Glue Ini hanya diperlukan untuk beban kerja besar yang menumpahkan banyak data ke disk. Secara default, parameter ini memiliki nilai`false`.

- (Opsional) `--conf spark.shuffle.glue.s3ShuffleBucket - S3://<shuffle-bucket>`

  Parameter ini menentukan bucket Amazon S3 yang akan digunakan saat menulis file shuffle. Jika Anda tidak mengatur parameter ini, lokasi adalah `shuffle-data` folder di lokasi yang ditentukan untuk Temporary path (`--TempDir`).

  > **Note**
  >
  > Pastikan lokasi bucket shuffle sama dengan Wilayah AWS tempat pekerjaan berjalan. Selain itu, layanan shuffle tidak membersihkan file setelah pekerjaan selesai berjalan, jadi Anda harus mengonfigurasi kebijakan siklus hidup penyimpanan Amazon S3 di lokasi bucket acak. Untuk informasi selengkapnya, lihat Mengelola siklus hidup penyimpanan di Panduan Pengguna Amazon S3.

Simpan tugas

Sebuah callout Tugas belum disimpan berwarna merah ditampilkan di sebelah kiri tombol Simpan sampai Anda menyimpan tugas.

Untuk menyimpan tugas Anda

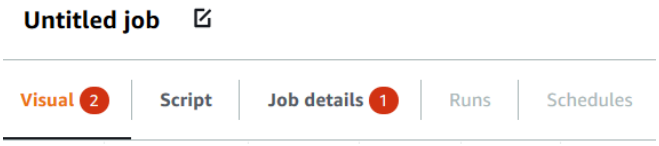
1. Berikan semua informasi yang diperlukan dalam tab Visual dan Detail Tugas.
2. Pilih tombol Simpan.

   Setelah Anda menyimpan tugas, callout 'tidak disimpan' akan berubah untuk menampilkan waktu dan tanggal tugas terakhir disimpan.

Jika Anda keluar AWS Glue Studio sebelum menyimpan pekerjaan Anda, saat berikutnya Anda masukAWS Glue Studio, pemberitahuan akan muncul. Notifikasi tersebut menunjukkan bahwa ada tugas yang belum disimpan, dan menanyakan apakah Anda ingin memulihkannya. Jika Anda memilih untuk memulihkan tugas tersebut, Anda dapat melanjutkan untuk mengeditnya.

Memecahkan masalah kesalahan saat menyimpan tugas

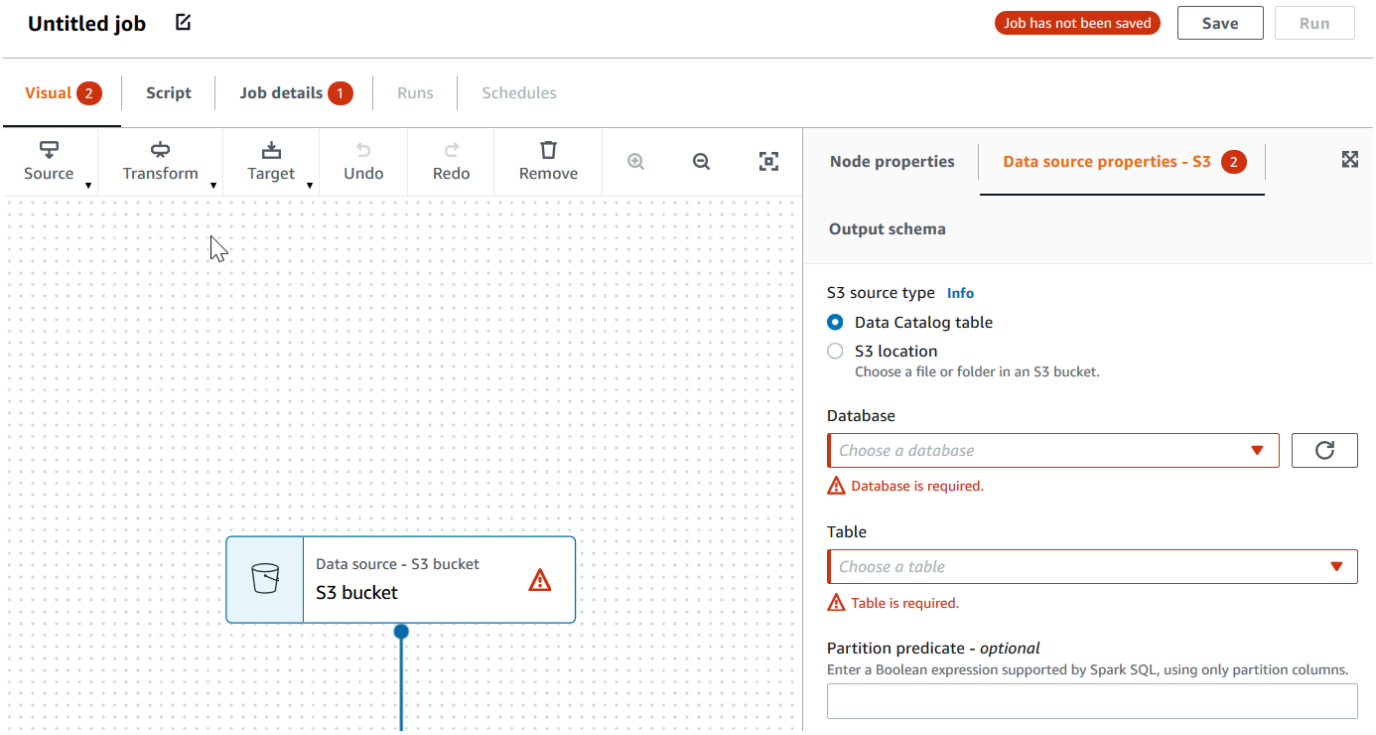
Jika Anda memilih tombol Simpan, tetapi tugas Anda tidak memiliki beberapa informasi yang diperlukan, dan kemudian muncul keterangan merah pada tab di mana informasi tersebut hilang. Nomor dalam callout menunjukkan berapa banyak bidang hilang yang terdeteksi.



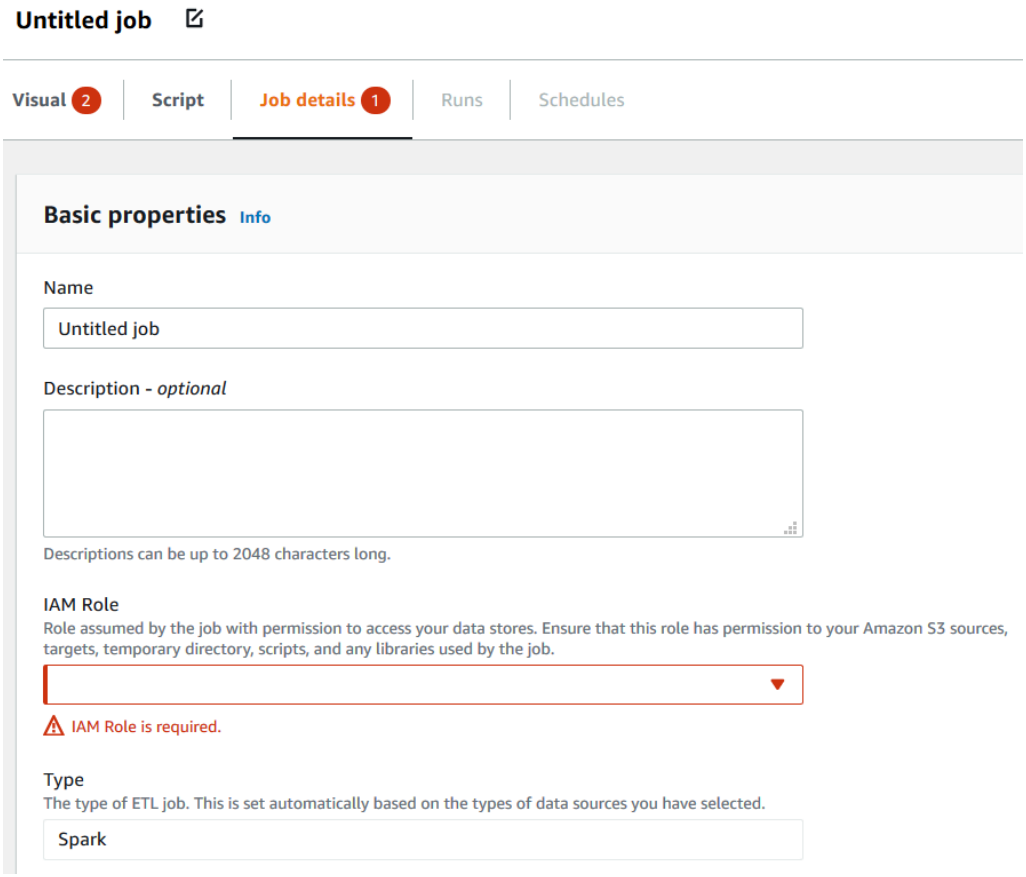
- Jika sebuah simpul di editor visual tidak dikonfigurasi dengan benar, maka tab Visual akan menunjukkan callout warna merah, dan simpul dengan kesalahan akan menampilkan simbol peringatan

  ⚠

  1. Pilih simpul. Di panel detail simpul, callout warna merah muncul di tab di mana ada informasi yang hilang atau salah.
  2. Pilih tab di panel detail simpul yang menunjukkan callout warna merah, dan kemudian cari bidang yang bermasalah, yang disorot. Pesan kesalahan di bawah bidang memberikan informasi tambahan tentang masalah yang terjadi.



- Jika ada masalah pada properti tugas, tab Detail tugas akan menunjukkan callout warna merah. Pilih tab itu dan cari bidang yang bermasalah, yang disorot. Pesan kesalahan di bawah bidang tersebut memberikan informasi tambahan tentang masalah yang terjadi.



Klon tugas

Anda dapat menggunakan tindakan Klon tugas untuk menyalin tugas yang sudah ada ke sebuah tugas baru.

Untuk membuat sebuah tugas baru dengan menyalin tugas yang sudah ada

1. Pada halaman Tugas, di daftar Tugas Anda, pilih tugas yang ingin Anda duplikasi.
2. Dari menu Tindakan, pilih Klon tugas.
3. Masukkan nama untuk tugas baru tersebut. Anda kemudian dapat menyimpan atau mengedit tugas tersebut.

Menghapus tugas

Anda dapat menghapus tugas yang tidak lagi diperlukan. Anda dapat menghapus satu atau beberapa tugas dalam satu operasi.

Untuk menghapus pekerjaan dari AWS Glue Studio

1. Pada halaman Tugas, di daftar Tugas Anda, pilih tugas yang ingin Anda hapus.
2. Dari menu Tindakan, pilih Hapus tugas.
3. Verifikasi bahwa Anda ingin menghapus tugas tersebut dengan memasukkan **`delete`**.

Anda juga dapat menghapus tugas yang tersimpan saat melihat tab Detail tugas untuk tugas tersebut di editor visual.

Bekerja dengan pekerjaan di AWS Glue

Bagian berikut memberikan informasi tentang pekerjaan ETL dan Ray di AWS Glue.

Topik

Versi AWS Glue

Anda dapat mengonfigurasi parameter AWS Glue versi saat menambahkan atau memperbarui pekerjaan. AWS GlueVersi ini menentukan versi Apache Spark dan Python yang mendukung. AWS Glue Versi Python menunjukkan versi yang didukung untuk pekerjaan jenis Spark. Tabel berikut mencantumkan versi AWS Glue yang tersedia, versi Spark dan Python yang sesuai, dan perubahan fungsi lainnya.

Versi AWS Glue

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
AWS Glue4.0	Versi lingkungan percikan • Spark 3.3.0 • Python 3.10	Java 8	AWS Glue4.0 adalah versi terbaru dariAWS Glue. Ada beberapa pengoptimalan dan peningkatan yang dibangun ke dalam AWS Glue rilis ini, seperti:

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas

			• Banyak peningkatan fungsionalitas Spark dari Spark 3.1 ke Spark 3.3:
			  • Beberapa peningkatan fungsionalitas saat dipasangkan dengan Panda. Untuk informasi selengkapnya, lihat Apa yang Baru di Spark 3.3.
			  • Pengoptim alan tambahan dikembangkan di Amazon EMR.
			  • Tingkatkan ke Sistem File EMR (EMRFS) 2.53.
			• Migrasi Log4j 2 dari Log4j 1.x
			• Beberapa pembaruan modul Python dari AWS Glue 3.0, seperti versi upgrade dari Boto.
			• Upgrade beberapa konektor, termasuk konektor Amazon

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
			Redshift default. Lihat Lampiran C: Peningkatan konektor. • Upgrade beberapa driver JDBC. Lihat Lampiran B: Peningkatan driver JDBC. • Diperbarui dengan konektor Amazon Redshift baru dan driver JDBC. • Dukungan asli untuk kerangka kerja danau data terbuka dengan Apache Hudi, Delta Lake, dan Apache Iceberg. • Dukungan asli untuk Plugin Penyimpanan Cloud Shuffle berbasis Amazon S3 (plugin Apache Spark) untuk menggunakan Amazon S3 untuk pengocokan dan kapasitas

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
			penyimpanan elastis. Batasan Berikut ini adalah batasan dengan AWS Glue 4.0: • AWS Gluepembelajaran mesin dan transformasi informasi identifikasi pribadi (PII) belum tersedia di 4.0. AWS Glue Untuk informasi selengkapnya tentang migrasi ke AWS Glue versi 4.0, lihatMigrasi AWS Glue untuk pekerjaan Spark ke versi 4.0 AWS Glue.

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
	Versi lingkungan Ray • Sinar 2.4.0 Python 3.9	N/A	Membangun dan menjalankan aplikasi Python terdistribusi dengan AWS Glue untuk Ray. • Mendukung distribusi data Ray-2.4.0 () `ray[data]` dengan Python 3.9. Untuk informasi lebih lanjut tentang rilis Ray ini, lihat Ray-2.4.0 di repositori Ray. GitHub • Mendukung pemasangan pustaka Python tambahan ke lingkungan runtime. `Ray2.4` Untuk informasi selengkapnya, lihat the section called “Modul Python tambahan untuk pekerjaan Ray”. • Mengintegrasikan log dan metrik dari pekerjaan Ray

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
			dengan Amazon. CloudWatch Untuk informasi selengkapnya, lihat the section called "Memecahkan masalah kesalahan Ray" dan the section called "Metrik pekerjaan Ray". • Mengagregat dan memvisualisasikan metrik untuk pekerjaan Ray di AWS Glue Studio, di setiap halaman menjalankan pekerjaan. • Mendukung distribusi file ke setiap direktori kerja di seluruh cluster Anda, menumpahkan objek dari penyimpanan objek Ray ke Amazon S3, dan mengontrol jumlah minimum node pekerja yang dialokasikan untuk pekerjaan

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
			Ray Anda. Untuk informasi selengkapnya, lihat the section called “Parameter pekerjaan ray”. Keterbatasan pada pekerjaan Ray di AWS Glue 4.0 • AWS Glue sesi interaktif untuk Ray tetap dalam pratinjau untuk rilis ini. • AWS Glue untuk integrasi Ray dengan Amazon VPC saat ini tidak tersedia. Sumber daya dalam VPC tidak AWS akan dapat diakses tanpa rute umum. Untuk informasi selengkapnya tentang penggunaan AWS Glue dengan Amazon VPC, lihat. the section called “Titik

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
			akhir VPC (AWS PrivateLink)” • AWS Glue untuk Ray tersedia di AS Timur (Virginia N.), AS Timur (Ohio), AS Barat (Oregon), Asia Pasifik (Tokyo), dan Eropa (Irlandia).

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
AWS Glue3.0	• Spark 3.1.1 • Python 3.7	Java 8	Selain upgrade mesin Spark ke 3.0, ada pengoptimalan dan peningkatan yang dibangun ke dalam AWS Glue rilis ini, seperti: • Membangun Perpustakaan AWS Glue ETL terhadap Spark 3.0, yang merupakan rilis utama untuk Spark. • Pekerjaan streaming didukung pada AWS Glue 3.0. • Termasuk optimasi runtime AWS Glue Spark baru untuk kinerja dan keandalan: • Pemrosesan kolumnar dalam memori yang lebih cepat berdasarkan Apache Arrow untuk membaca data CSV.

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
			• Eksekusi berbasis SIM untuk pembacaan vektor dengan data CSV. • Peningkatan Spark juga mencakup pengoptimalan tambahan yang dikembangkan di Amazon EMR. • EMRFS yang ditingkatkan dari 2,38 menjadi 2,46 memungkinkan fitur baru dan perbaikan bug untuk akses Amazon S3. • Memutakhirkan beberapa dependensi yang diperlukan untuk versi Spark baru. Lihat Lampiran A: peningkatan ketergantungan penting. • Driver JDBC yang ditingkatkan untuk sumber data kami

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
			yang didukung secara native. Lihat Lampiran B: Peningkatan driver JDBC. Batasan Berikut ini adalah batasan dengan AWS Glue 3.0: • AWS GlueTransformasi pembelajaran mesin belum tersedia di AWS Glue 3.0. • Beberapa konektor Spark khusus tidak berfungsi dengan AWS Glue 3.0 jika bergantung pada Spark 2.4 dan tidak memiliki kompatibilitas dengan Spark 3.1. Untuk informasi selengkapnya tentang migrasi ke AWS Glue versi 3.0, lihatMigrasi AWS Glue untuk

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
			pekerjaan Spark ke versi 3.0 AWS Glue.

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
AWS Glue2.0 (usang, akhir dukungan)	• Spark versi 2.4.3 • Python 3.7	N/A	Selain fitur yang disediakan dalam AWS Glue versi 1.0, AWS Glue versi 2.0 juga menyediakan: • Peningkatan infrastruktur untuk menjalankan tugas ETL Apache Spark di AWS Glue dengan waktu pemulaian yang berkurang. • Pencatatan default sekarang real time, dengan aliran terpisah untuk driver dan pelaksana, serta output dan kesalahan. • Support untuk menentukan modul Python tambahan atau versi yang berbeda pada tingkat tugas.

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
			Note AWS Glueversi 2.0 berbeda dari AWS Glue versi 1.0 untuk beberapa dependensi dan versi karena perubahan arsitektur yang mendasarinya. Validasi AWS Glue pekerjaan Anda sebelum bermigrasi di seluruh rilis AWS Glue versi utama. Untuk informasi selengkapnya tentang fitur dan batasan AWS Glue versi 2.0, lihatMenjalankan pekerjaan Spark ETL

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
			dengan waktu startup yang berkurang.

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
AWS Glue1.0 (usang, akhir dukungan)	• Spark versi 2.4.3 • Python 2.7 • Python 3.6	N/A	Anda dapat menyimpan bookmark tugas untuk format Parket dan ORC di tugas ETL AWS Glue (menggunakan AWS Glue versi 1.0). Sebelumnya, Anda hanya dapat menandai format sumber Amazon S3 umum seperti JSON, CSV, Apache Avro, dan XMLdalam pekerjaan ETL. AWS Glue Ketika menetapkan opsi format untuk input dan output ETL, Anda dapat menentukan untuk menggunakan Apache Avro pembaca/penulis format 1.8 untuk men-support penulisan dan pembacaan Avro jenis logis (menggunakan AWS Glue versi 1.0). Sebelumnya, hanya Avro pembaca/penulis

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
			format versi 1.7 yang didukung. Jenis koneksi DynamoDB mendukung opsi penulis (AWS Gluemenggunakan versi 1.0). Batasan Berikut ini adalah batasan dengan AWS Glue 1.0: • AWS GlueVersi 0.9 dan 1.0 tidak tersedia di Asia Pasifik (Jakarta) (`ap-southeast-3`), Timur Tengah (UEA) (`me-central-1`), atau Wilayah baru lainnya di masa mendatang.

AWS Glue versi	Versi lingkungan runtime yang didukung	Versi Java yang didukung	Perubahan fungsionalitas
AWS Glue0.9 (usang, akhir dukungan)	• Spark versi 2.2.1 • Python 2.7	N/A	Pekerjaan yang dibuat tanpa menentukan AWS Glue versi default ke AWS Glue 0.9. Batasan Berikut ini adalah batasan dengan AWS Glue 0,9: • AWS GlueVersi 0.9 dan 1.0 tidak tersedia di Asia Pasifik (Jakarta) (`ap-southeast-3`), Timur Tengah (UEA) (`me-central-1`), atau Wilayah baru lainnya di masa mendatang.

Menjalankan pekerjaan Spark ETL dengan waktu startup yang berkurang

AWS Glueversi 2.0 dan yang lebih baru menyediakan infrastruktur yang ditingkatkan untuk menjalankan pekerjaan Apache Spark ETL (ekstrak, transformasi, dan muat) AWS Glue dengan waktu startup yang berkurang. Dengan berkurangnya waktu tunggu, insinyur data dapat lebih produktif dan meningkatkan interaktivitas mereka dengan AWS Glue. Varians yang berkurang dalam waktu mulai tugas dapat membantu Anda memenuhi atau melampaui SLA Anda dalam membuat data tersedia untuk analitik.

Untuk menggunakan fitur ini dengan pekerjaan AWS Glue ETL Anda, pilih **`2.0`** atau versi yang lebih baru untuk `Glue version` saat membuat pekerjaan Anda.

Topik

Fitur baru yang didukung

Bagian ini menjelaskan fitur baru yang didukung dengan AWS Glue versi 2.0 dan yang lebih baru.

Support untuk menentukan modul Python tambahan di tingkat pekerjaan

AWS Glueversi 2.0 dan yang lebih baru juga memungkinkan Anda menyediakan modul Python tambahan atau versi berbeda di tingkat pekerjaan. Anda dapat menggunakan `--additional-python-modules` dengan daftar modul Python yang dipisahkan koma untuk menambahkan modul baru atau mengubah versi dari modul yang ada.

Misalnya untuk memperbarui atau menambahkan modul `scikit-learn` baru menggunakan nilai/kunci berikut: `"--additional-python-modules", "scikit-learn==0.21.3"`.

Selain itu, dalam opsi `--additional-python-modules` Anda dapat menentukan path Amazon S3 ke modul roda Python. Misalnya:

```
--additional-python-modules s3://aws-glue-native-spark/tests/j4.2/ephem-3.7.7.1-cp37-cp37m-linux_x86_64.whl,s3://aws-glue-native-spark/tests/j4.2/fbprophet-0.6-py3-none-any.whl,scikit-learn==0.21.3
```

AWS Glue menggunakan Python Package Installer (pip3) untuk menginstal modul tambahan. Anda dapat melewati opsi tambahan yang ditentukan oleh `python-modules-installer-option` ke pip3 untuk memasang modul. Ketidakcocokan atau batasan apa pun dari pip3 akan berlaku.

Modul Python sudah disediakan di versi 2.0 AWS Glue

AWS Glue versi 2.0 mendukung modul python berikut secara siap pakai:

- setuptools—45.2.0

- subprocess32—3.5.4
- ptvsd—4.3.2
- pydevd—1.9.0
- PyMySQL—0.9.3
- docutils—0.15.2
- jmespath—0.9.4
- six—1.14.0
- python_dateutil—2.8.1
- urllib3—1.25.8
- botocore—1.15.4
- s3transfer—0.3.3
- boto3—1.12.4
- certifi—2019.11.28
- chardet—3.0.4
- idna—2.9
- requests—2.23.0
- pyparsing—2.4.6
- enum34—1.1.9
- pytz—2019.3
- numpy—1.18.1
- cycler—0.10.0
- kiwisolver—1.1.0
- scipy—1.4.1
- pandas—1.0.1
- pyarrow—0.16.0
- matplotlib—3.1.3
- pyhocon—0.3.54
- mpmath—1.1.0
- sympy—1.5.1

- patsy—0.5.1
- statsmodels—0.11.1
- fsspec—0.6.2
- s3fs—0.4.0
- Cython—0.29.15
- joblib—0.14.1
- pmdarima—1.5.3
- scikit-learn—0.22.1
- tbats—1.0.9

Perilaku logging

AWS Glueversi 2.0 dan yang lebih baru mendukung perilaku logging default yang berbeda. Perbedaannya meliputi:

- Pencatatan log terjadi secara realtime.
- Ada pengaliran terpisah untuk driver dan pelaksana.
- Untuk setiap driver dan pelaksana ada dua pengaliran, yanitu pengaliran output dan pengaliran kesalahan.

Aliran driver dan eksekutor

Pengaliran driver diidentifikasi berdasarkan ID eksekusi tugas. Pengaliran pelaksana diidentifikasi berdasarkan <*id eksekusi tugas*>_<*id pelaksana tugas*>. Misalnya:

- `"logStreamName": "jr_8255308b426fff1b4e09e00e0bd5612b1b4ec848d7884cebe61ed33a31789..._g-f65f617bd31d54bd94482af755b6cdf464542..."`

Aliran keluaran dan kesalahan

Pengaliran output memiliki output standar (stdout) dari kode Anda. Pengaliran kesalahan memiliki pesan log dari kode/perpustakaan Anda.

- Pengaliran log:

- Pengaliran log driver memiliki <*jr*>, dimana <*jr*> adalah ID eksekusi tugas.
- Pengaliran log pelaksana memiliki <*jr*>_<*g*>, dimana <*g*> adalah ID tugas untuk pelaksana. Anda dapat mencari ID tugas pelaksana di log kesalahan driver.

Grup log default untuk AWS Glue versi 2.0 adalah sebagai berikut:

- `/aws-glue/jobs/logs/output` untuk output
- `/aws-glue/jobs/logs/error` untuk kesalahan

Ketika konfigurasi keamanan disediakan, nama grup log berubah menjadi:

- `/aws-glue/jobs/<security configuration>-role/<Role Name>/output`
- `/aws-glue/jobs/<security configuration>-role/<Role Name>/error`

Pada konsol, tautan Log mengarahkan ke grup log output dan tautan Kesalahan mengarahkan ke grup log kesalahan. Ketika pencatatan log terus menerus diaktifkan, tautan Log mengarahkan ke grup log terus menerus, dan tautan Output mengarahkan ke grup log output.

Aturan pencatatan

> **Note**
>
> Nama grup log default untuk pencatatan log terus-menerus adalah `/aws-glue/jobs/logs-v2`.

Di AWS Glue versi 2.0 dan yang lebih baru, logging berkelanjutan memiliki perilaku yang sama seperti di AWS Glue versi 1.0:

- Grup log default: `/aws-glue/jobs/logs-v2`
- Pengaliran log driver: <*jr*>-driver
- Pengaliran log pelaksana: <*jr*>-<*ID pelaksana*>

  Nama grup log dapat diubah dengan mengatur `--continuous-log-logGroupName`

  Nama pengaliran log dapat menggunakan prefiks dengan mengatur `--continous-log-logStreamPrefix`

Fitur tidak didukung

Fitur AWS Glue berikut tidak didukung:

- Titik akhir pengembangan
- AWS Glueversi 2.0 dan yang lebih baru tidak berjalan di Apache YARN, jadi pengaturan YARN tidak berlaku
- AWS Glueversi 2.0 dan yang lebih baru tidak memiliki Hadoop Distributed File System (HDFS)
- AWS Glueversi 2.0 dan yang lebih baru tidak menggunakan alokasi dinamis, oleh karena itu ExecutorAllocationManager metrik tidak tersedia
- Untuk AWS Glue versi 2.0 atau pekerjaan yang lebih baru, Anda menentukan jumlah pekerja dan jenis pekerja, tetapi tidak menentukan`maxCapacity`.
- AWS Glueversi 2.0 dan yang lebih baru tidak mendukung di `s3n` luar kotak. Kami merekomendasikan penggunaan `s3` atau `s3a`. Jika tugas perlu menggunakan `s3n` untuk alasan apapun, Anda dapat memberikan argumen tambahan berikut:

```
--conf spark.hadoop.fs.s3n.impl=com.amazon.ws.emr.hadoop.fs.EmrFileSystem
```

Migrasi AWS Glue untuk pekerjaan Spark ke versi 3.0 AWS Glue

Topik ini menjelaskan perubahan antara AWS Glue versi 0.9, 1.0, 2.0 dan 3.0 untuk memungkinkan Anda memigrasikan aplikasi Spark dan pekerjaan ETL ke 3.0. AWS Glue

Untuk menggunakan fitur ini dengan tugas ETL AWS Glue Anda, pilih **`3.0`** untuk `Glue version` saat membuat tugas Anda.

Topik

- Lampiran B: Peningkatan driver JDBC

Fitur baru yang didukung

Bagian ini menjelaskan fitur dan keunggulan baru AWS Glue versi 3.0.

- Ini didasarkan pada Apache Spark 3.1.1, yang memiliki optimasi dari Spark open-source dan dikembangkan oleh layanan AWS Glue EMR dan seperti eksekusi kueri adaptif, pembaca vektor, dan pengocokan dan penggabungan partisi yang dioptimalkan.
- Driver JDBC yang ditingkatkan untuk semua sumber asli Glue termasuk MySQL, Microsoft SQL Server, Oracle, PostgreSQL, MongoDB, dan pustaka dan dependensi Spark yang ditingkatkan yang dibawa oleh Spark 3.1.1.
- Akses Amazon S3 yang dioptimalkan dengan EMRFS yang ditingkatkan dan mengaktifkan penghasil keluaran Amazon S3 yang dioptimalkan secara default.
- Akses Katalog Data yang Dioptimalkan dengan indeks partisi, predikat push down, daftar partisi, dan klien metastore Hive yang ditingkatkan.
- Integrasi dengan Lake Formation untuk tabel katalog yang diatur dengan penyaringan tingkat sel dan transaksi data lake.
- Pengalaman Spark UI yang ditingkatkan dengan Spark 3.1.1 dengan metrik memori eksekutor Spark baru dan metrik streaming terstruktur Spark.
- Mengurangi latensi startup meningkatkan waktu penyelesaian pekerjaan secara keseluruhan dan interaktivitas, mirip AWS Glue dengan 2.0.
- Pekerjaan Spark ditagih dalam kenaikan 1 detik dengan durasi penagihan minimum 10x lebih rendah—dari minimum 10 menit hingga minimum 1 menit, mirip dengan 2.0. AWS Glue

Tindakan untuk bermigrasi ke 3.0 AWS Glue

Untuk pekerjaan yang ada, ubah `Glue version` dari versi sebelumnya ke `Glue 3.0` dalam konfigurasi pekerjaan.

- Di konsol, pilih `Spark 3.1, Python 3 (Glue Version 3.0) or Spark 3.1, Scala 2 (Glue Version 3.0)` di`Glue version`.
- Di AWS Glue Studio, pilih `Glue 3.0 - Supports spark 3.1, Scala 2, Python 3` di`Glue version`.
- Di API, pilih `GlueVersion` parameter **`3.0`** di `UpdateJob`API.

Untuk pekerjaan baru, pilih `Glue 3.0` kapan Anda membuat pekerjaan.

- Di konsol, pilih `Spark 3.1, Python 3 (Glue Version 3.0) or Spark 3.1, Scala 2 (Glue Version 3.0)` di`Glue version`.
- Di AWS Glue Studio, pilih `Glue 3.0 - Supports spark 3.1, Scala 2, Python 3` di`Glue version`.
- Di API, pilih `GlueVersion` parameter **`3.0`** di `CreateJob`API.

Untuk melihat log peristiwa Spark AWS Glue 3.0, luncurkan server riwayat Spark yang ditingkatkan untuk Glue 3.0 menggunakan CloudFormation atau Docker.

Daftar cek migrasi

Tinjau daftar periksa ini untuk migrasi.

- Apakah pekerjaan Anda tergantung pada HDFS? Jika ya, coba ganti HDFS dengan S3.
  - Cari jalur sistem file yang dimulai dengan `hdfs://` atau `/` sebagai jalur DFS dalam kode skrip pekerjaan.
  - Periksa apakah sistem file default Anda tidak dikonfigurasi dengan HDFS. Jika dikonfigurasi secara eksplisit, Anda perlu menghapus konfigurasi. `fs.defaultFS`
  - Periksa apakah pekerjaan Anda berisi `dfs.*` parameter apa pun. Jika berisi, Anda perlu memverifikasi tidak apa-apa untuk menonaktifkan parameter.
- Apakah pekerjaan Anda bergantung pada YARN? Jika ya, verifikasi dampaknya dengan memeriksa apakah pekerjaan Anda berisi parameter berikut. Jika berisi, Anda perlu memverifikasi tidak apa-apa untuk menonaktifkan parameter.
  - `spark.yarn.*`

    Misalnya:

    ```
    spark.yarn.executor.memoryOverhead
    spark.yarn.driver.memoryOverhead
    spark.yarn.scheduler.reporterThread.maxFailures
    ```

  - `yarn.*`

    Misalnya:

    ```
    yarn.scheduler.maximum-allocation-mb
    ```

```
yarn.nodemanager.resource.memory-mb
```

- Apakah pekerjaan Anda bergantung pada Spark 2.2.1 atau Spark 2.4.3? Jika ya, verifikasi dampaknya dengan memeriksa apakah pekerjaan Anda menggunakan fitur yang diubah di Spark 3.1.1.
  - `https://spark.apache.org/docs/latest/sql-migration-guide.html#upgrading-from-spark-sql-22-to-23`

    Misalnya `percentile_approx` fungsi, atau `SparkSession` dengan `SparkSession.builder.getOrCreate()` ketika ada yang sudah ada`SparkContext`.
  - `https://spark.apache.org/docs/latest/sql-migration-guide.html#upgrading-from-spark-sql-23-to-24`

    Misalnya `array_contains` fungsi, atau`CURRENT_DATE`, `CURRENT_TIMESTAMP` fungsi dengan`spark.sql.caseSensitive=true`.
- Apakah stoples ekstra pekerjaan Anda bertentangan di Glue 3.0?
  - Dari AWS Glue 0.9/1.0: Stoples tambahan yang disediakan dalam pekerjaan AWS Glue 0.9/1.0 yang ada dapat menimbulkan konflik classpath karena peningkatan atau dependensi baru yang tersedia di Glue 3.0. Anda dapat menghindari konflik classpath di AWS Glue 3.0 dengan parameter `--user-jars-first` AWS Glue pekerjaan atau dengan menaungi dependensi Anda.
  - Dari AWS Glue 2.0: Anda masih dapat menghindari konflik classpath di AWS Glue 3.0 dengan parameter `--user-jars-first` AWS Glue pekerjaan atau dengan menaungi dependensi Anda.
- Apakah pekerjaan Anda bergantung pada Scala 2.11?
  - AWS Glue3.0 menggunakan Scala 2.12 sehingga Anda perlu membangun kembali perpustakaan Anda dengan Scala 2.12 jika perpustakaan Anda bergantung pada Scala 2.11.
- Apakah pustaka Python eksternal pekerjaan Anda bergantung pada Python 2.7/3.6?
  - Gunakan `--additional-python-modules` parameter alih-alih mengatur file telur/roda/zip di jalur pustaka Python.
  - Perbarui pustaka dependen dari Python 2.7/3.6 ke Python 3.7 karena Spark 3.1.1 menghapus dukungan Python 2.7.

Migrasi dari AWS Glue 0,9 ke 3,0 AWS Glue

Perhatikan perubahan berikut saat bermigrasi:

- AWS Glue0.9 menggunakan sumber terbuka Spark 2.2.1 dan AWS Glue 3.0 menggunakan Spark 3.1.1 yang dioptimalkan EMR.
  - Beberapa perubahan Spark saja mungkin memerlukan revisi skrip Anda untuk memastikan fitur yang dihapus tidak direferensikan.
  - Misalnya, Spark 3.1.1 tidak mengaktifkan UDF yang tidak diketik Scala-tetapi Spark 2.2 mengizinkannya.
- Semua pekerjaan di AWS Glue 3.0 akan dieksekusi dengan waktu startup yang ditingkatkan secara signifikan. Pekerjaan Spark akan ditagih dalam kenaikan 1 detik dengan durasi penagihan minimum 10x lebih rendah karena latensi startup akan meningkat dari maksimum 10 menit menjadi maksimum 1 menit.
- Perilaku logging telah berubah sejak AWS Glue 2.0.
- Beberapa pembaruan ketergantungan, disorot dalamLampiran A: peningkatan ketergantungan penting.
- Scala juga diperbarui ke 2.12 dari 2.11, dan Scala 2.12 tidak kompatibel dengan Scala 2.11.
- Python 3.7 juga merupakan versi default yang digunakan untuk skrip Python, karena 0.9 AWS Glue hanya menggunakan Python 2.
  - Python 2.7 tidak didukung dengan Spark 3.1.1.
  - Mekanisme baru untuk menginstal modul Python tambahan tersedia.
- AWS Glue3.0 tidak berjalan di Apache YARN, jadi pengaturan YARN tidak berlaku.
- AWS Glue3.0 tidak memiliki Hadoop Distributed File System (HDFS).
- Setiap stoples tambahan yang disediakan dalam AWS Glue 0.9 pekerjaan yang ada dapat membawa dependensi yang bertentangan karena ada peningkatan di beberapa dependensi di 3.0 dari 0.9. Anda dapat menghindari konflik classpath di AWS Glue 3.0 dengan parameter `--user-jars-first` AWS Glue pekerjaan.
- AWS Glue3.0 belum mendukung alokasi dinamis, oleh karena itu ExecutorAllocationManager metrik tidak tersedia.
- Dalam pekerjaan AWS Glue versi 3.0, Anda menentukan jumlah pekerja dan jenis pekerja, tetapi tidak menentukan`maxCapacity`.
- AWS Glue3.0 belum mendukung transformasi pembelajaran mesin.
- AWS Glue3.0 belum mendukung titik akhir pengembangan.

Lihat dokumentasi migrasi Spark:

- lihat Upgrade dari Spark SQL 2.2 ke 2.3
- lihat Upgrade dari Spark SQL 2.3 ke 2.4
- lihat Memutakhirkan dari Spark SQL 2.4 ke 3.0
- lihat Memutakhirkan dari Spark SQL 3.0 ke 3.1
- lihat Perubahan perilaku Datetime diharapkan sejak Spark 3.0.

Migrasi dari AWS Glue 1.0 ke 3.0 AWS Glue

Perhatikan perubahan berikut saat bermigrasi:

- AWS Glue1.0 menggunakan sumber terbuka Spark 2.4 dan AWS Glue 3.0 menggunakan Spark 3.1.1 yang dioptimalkan EMR.
  - Beberapa perubahan Spark saja mungkin memerlukan revisi skrip Anda untuk memastikan fitur yang dihapus tidak direferensikan.
  - Misalnya, Spark 3.1.1 tidak mengaktifkan UDF yang tidak diketik Scala-tetapi Spark 2.4 mengizinkannya.
- Semua pekerjaan di AWS Glue 3.0 akan dieksekusi dengan waktu startup yang ditingkatkan secara signifikan. Pekerjaan Spark akan ditagih dalam kenaikan 1 detik dengan durasi penagihan minimum 10x lebih rendah karena latensi startup akan meningkat dari maksimum 10 menit menjadi maksimum 1 menit.
- Perilaku logging telah berubah sejak AWS Glue 2.0.
- Beberapa pembaruan ketergantungan, disorot
- Scala juga diperbarui ke 2.12 dari 2.11, dan Scala 2.12 tidak kompatibel dengan Scala 2.11.
- Python 3.7 juga merupakan versi default yang digunakan untuk skrip Python, karena 0.9 AWS Glue hanya menggunakan Python 2.
  - Python 2.7 tidak didukung dengan Spark 3.1.1.
  - Mekanisme baru untuk menginstal modul Python tambahan tersedia.
- AWS Glue3.0 tidak berjalan di Apache YARN, jadi pengaturan YARN tidak berlaku.
- AWS Glue3.0 tidak memiliki Hadoop Distributed File System (HDFS).
- Setiap stoples tambahan yang disediakan dalam pekerjaan AWS Glue 1.0 yang ada dapat membawa dependensi yang bertentangan karena ada peningkatan di beberapa dependensi di 3.0 dari 1.0. Anda dapat menghindari konflik classpath di AWS Glue 3.0 dengan parameter `--user-jars-first` AWS Glue pekerjaan.

- AWS Glue3.0 belum mendukung alokasi dinamis, oleh karena itu ExecutorAllocationManager metrik tidak tersedia.
- Dalam pekerjaan AWS Glue versi 3.0, Anda menentukan jumlah pekerja dan jenis pekerja, tetapi tidak menentukan`maxCapacity`.
- AWS Glue3.0 belum mendukung transformasi pembelajaran mesin.
- AWS Glue3.0 belum mendukung titik akhir pengembangan.

Lihat dokumentasi migrasi Spark:

- lihat Memutakhirkan dari Spark SQL 2.4 ke 3.0
- lihat Perubahan perilaku Datetime diharapkan sejak Spark 3.0.

Migrasi dari AWS Glue 2.0 ke AWS Glue 3.0

Perhatikan perubahan berikut saat bermigrasi:

- Semua parameter pekerjaan yang ada dan fitur utama yang ada di AWS Glue 2.0 akan ada di AWS Glue 3.0.
  - Komitter yang dioptimalkan EMRFS S3 untuk menulis data Parket ke Amazon S3 diaktifkan secara default di 3.0. AWS Glue Namun, Anda masih dapat menonaktifkannya dengan menyetel `--enable-s3-parquet-optimized-committer` kefalse.
- AWS Glue2.0 menggunakan sumber terbuka Spark 2.4 dan AWS Glue 3.0 menggunakan Spark 3.1.1 yang dioptimalkan EMR.
  - Beberapa perubahan Spark saja mungkin memerlukan revisi skrip Anda untuk memastikan fitur yang dihapus tidak direferensikan.
  - Misalnya, Spark 3.1.1 tidak mengaktifkan UDF yang tidak diketik Scala-tetapi Spark 2.4 mengizinkannya.
- AWS Glue3.0 juga dilengkapi pembaruan untuk EMRFS, driver JDBC yang diperbarui, dan inklusi pengoptimalan tambahan ke Spark sendiri yang disediakan oleh. AWS Glue
- Semua pekerjaan di AWS Glue 3.0 akan dieksekusi dengan waktu startup yang ditingkatkan secara signifikan. Pekerjaan Spark akan ditagih dalam kenaikan 1 detik dengan durasi penagihan minimum 10x lebih rendah karena latensi startup akan meningkat dari maksimum 10 menit menjadi maksimum 1 menit.
- Python 2.7 tidak didukung dengan Spark 3.1.1.

- Beberapa pembaruan ketergantungan, disorot dalamLampiran A: peningkatan ketergantungan penting.
- Scala juga diperbarui ke 2.12 dari 2.11, dan Scala 2.12 tidak kompatibel dengan Scala 2.11.
- Setiap stoples tambahan yang disediakan dalam pekerjaan AWS Glue 2.0 yang ada dapat membawa dependensi yang bertentangan karena ada peningkatan di beberapa dependensi di 3.0 dari 2.0. Anda dapat menghindari konflik classpath di AWS Glue 3.0 dengan parameter `--user-jars-first` AWS Glue pekerjaan.
- AWS Glue3.0 memiliki paralelisme tugas Spark yang berbeda untuk konfigurasi driver/pelaksana dibandingkan dengan AWS Glue 2.0 dan meningkatkan kinerja dan memanfaatkan sumber daya yang tersedia dengan lebih baik. Keduanya `spark.driver.cores` dan `spark.executor.cores` dikonfigurasi ke jumlah core pada AWS Glue 3.0 (4 pada standar dan `G.1X` pekerja, dan 8 pada `G.2X` pekerja). Konfigurasi ini tidak mengubah jenis pekerja atau perangkat keras untuk AWS Glue pekerjaan itu. Anda dapat menggunakan konfigurasi ini untuk menghitung jumlah partisi atau split agar sesuai dengan paralelisme tugas Spark di aplikasi Spark Anda.

  Secara umum, pekerjaan akan melihat kinerja yang serupa atau lebih baik dibandingkan dengan AWS Glue 2.0. Jika pekerjaan berjalan lebih lambat, Anda dapat meningkatkan paralelisme tugas dengan meneruskan argumen pekerjaan berikut:

  - key: `--executor-cores` value: < *jumlah tugas yang diinginkan yang dapat berjalan secara paralel>*
  - Nilai tidak boleh melebihi 2x jumlah vCPU pada tipe pekerja, yaitu 8 on, 16 on, 32 `G.1X` `G.4X` on dan `G.2X` 64 on. `G.8X` Anda harus berhati-hati saat memperbarui konfigurasi ini karena dapat memengaruhi kinerja pekerjaan karena peningkatan paralelisme menyebabkan memori dan tekanan disk, serta dapat menghambat sistem sumber dan target.
- AWS Glue3.0 menggunakan Spark 3.1, yang mengubah perilaku memuat/menyimpan stempel waktu dari/ke file parket. Untuk detail selengkapnya, lihat Memutakhirkan dari Spark SQL 3.0 ke 3.1.

  Kami merekomendasikan untuk mengatur parameter berikut saat membaca/menulis data parket yang berisi kolom stempel waktu. Menyetel parameter tersebut dapat menyelesaikan masalah ketidakcocokan kalender yang terjadi selama peningkatan Spark 2 ke Spark 3, untuk AWS Glue Dynamic Frame dan Spark Data Frame. Gunakan opsi CORRECTED untuk membaca nilai datetime apa adanya; dan opsi LEGACY untuk rebase nilai datetime sehubungan dengan perbedaan kalender selama membaca.

```
- Key: --conf
- Value: spark.sql.legacy.parquet.int96RebaseModeInRead=[CORRECTED|LEGACY] --
conf spark.sql.legacy.parquet.int96RebaseModeInWrite=[CORRECTED|LEGACY] --conf
 spark.sql.legacy.parquet.datetimeRebaseModeInRead=[CORRECTED|LEGACY]
```

Lihat dokumentasi migrasi Spark:

- lihat Memutakhirkan dari Spark SQL 2.4 ke 3.0
- lihat Perubahan perilaku Datetime diharapkan sejak Spark 3.0.

Lampiran A: peningkatan ketergantungan penting

Berikut ini adalah peningkatan ketergantungan:

Dependensi	Versi dalam AWS Glue 0.9	Versi dalam AWS Glue 1.0	Versi dalam AWS Glue 2.0	Versi dalam AWS Glue 3.0
Spark	2.2.1	2.4.3	2.4.3	3.1.1-amzn-0
Hadoop	2.7.3-amzn-6	2.8.5-amzn-1	2.8.5-amzn-5	3.2.1-amzn-3
Skala	2.11	2.11	2.11	2.12
Jackson	2.7.x	2.7.x	2.7.x	2.10.x
Hive	1.2	1.2	1.2	2.3.7-amzn-4
EMRFS	2.20.0	2.30.0	2.38.0	2.46.0
JSON4	3.2.x	3.5.x	3.5.x	3.6.6
Panah	T/A	0.10.0	0.10.0	2.0.0
AWS GlueKlien katalog	T/A	T/A	1.10.0	3.0.0

Lampiran B: Peningkatan driver JDBC

Berikut ini adalah upgrade driver JDBC:

Driver	Versi driver JDBC di versi sebelumnya AWS Glue	Versi driver JDBC di 3.0 AWS Glue
MySQL	5.1	8.0.23
Microsoft SQL Server	6.1.0	7.0.0
Database Oracle	11.2	21.1
PostgreSQL	42.1.0	42.2.18
MongoDB	2.0.0	4.0.0

Migrasi AWS Glue untuk pekerjaan Spark ke versi 4.0 AWS Glue

Topik ini menjelaskan perubahan antara AWS Glue versi 0.9, 1.0, 2.0, dan 3.0 untuk memungkinkan Anda memigrasikan aplikasi Spark dan pekerjaan ETL ke 4.0. AWS Glue Ini juga menjelaskan fitur di AWS Glue 4.0 dan keuntungan menggunakannya.

Untuk menggunakan fitur ini dengan tugas ETL AWS Glue Anda, pilih **`4.0`** untuk `Glue version` saat membuat tugas Anda.

Topik

- Lampiran B: Peningkatan driver JDBC
- Lampiran C: Peningkatan konektor

Fitur baru yang didukung

Bagian ini menjelaskan fitur dan keunggulan baru AWS Glue versi 4.0.

- Ini didasarkan pada Apache Spark 3.3.0, tetapi mencakup pengoptimalan di, AWS Glue dan Amazon EMR, seperti proses kueri adaptif, pembaca vektor, dan pengocokan dan penggabungan partisi yang dioptimalkan.
- Driver JDBC yang ditingkatkan untuk semua sumber AWS Glue asli termasuk MySQL, Microsoft SQL Server, Oracle, PostgreSQL, MongoDB, dan pustaka dan dependensi Spark yang ditingkatkan yang dibawa oleh Spark 3.3.0.
- Diperbarui dengan konektor Amazon Redshift baru dan driver JDBC.
- Akses Amazon S3 yang dioptimalkan dengan Sistem File EMR (EMRFS) yang ditingkatkan dan mengaktifkan penghasil keluaran Amazon S3 yang dioptimalkan, secara default.
- Akses Katalog Data yang Dioptimalkan dengan indeks partisi, predikat pushdown, daftar partisi, dan klien metastore Hive yang ditingkatkan.
- Integrasi dengan Lake Formation untuk tabel katalog yang diatur dengan penyaringan tingkat sel dan transaksi data lake.
- Mengurangi latensi startup untuk meningkatkan waktu penyelesaian pekerjaan secara keseluruhan dan interaktivitas.
- Pekerjaan Spark ditagih dalam kenaikan 1 detik dengan durasi penagihan minimum 10x lebih rendah—dari minimum 10 menit hingga minimum 1 menit.
- Dukungan asli untuk kerangka kerja danau data terbuka dengan Apache Hudi, Delta Lake, dan Apache Iceberg.
- Dukungan asli untuk Plugin Penyimpanan Cloud Shuffle berbasis Amazon S3 (plugin Apache Spark) untuk menggunakan Amazon S3 untuk pengocokan dan kapasitas penyimpanan elastis.

Peningkatan utama dari Spark 3.1.1 ke Spark 3.3.0

Perhatikan penyempurnaan berikut:

- Pemfilteran runtime tingkat baris (SPARK-32268).
- Penyempurnaan ANSI (SPARK-38860).

- Perbaikan pesan kesalahan (SPARK-38781).
- Support tipe kompleks untuk pembaca vektor Parket (SPARK-34863).
- Dukungan metadata file tersembunyi untuk Spark SQL (SPARK-37273).
- Menyediakan profiler untuk Python/Pandas UDFs (SPARK-37443).
- Perkenalkan Trigger. AvailableNow untuk menjalankan kueri streaming seperti Trigger.Once dalam beberapa batch (SPARK-36533).
- Kemampuan pushdown Datasource V2 yang lebih komprehensif (SPARK-38788).
- Bermigrasi dari log4j 1 ke log4j 2 (SPARK-37814).

Perubahan penting lainnya

Perhatikan perubahan berikut:

- Melanggar perubahan
  - Jatuhkan referensi ke dukungan Python 3.6 di dokumen dan Python/docs (SPARK-36977).
  - Hapus peretasan tupel bernama dengan mengganti acar bawaan ke cloudpickle (SPARK-32079).
  - Bump versi panda minimum ke 1.0.5 (SPARK-37465).

Tindakan untuk bermigrasi ke 4.0 AWS Glue

Untuk pekerjaan yang ada, ubah `Glue version` dari versi sebelumnya ke `Glue 4.0` dalam konfigurasi pekerjaan.

- Di AWS Glue Studio, pilih `Glue 4.0 - Supports Spark 3.3, Scala 2, Python 3` di`Glue version`.
- Di API, pilih `GlueVersion` parameter **`4.0`** dalam operasi `UpdateJob`API.

Untuk pekerjaan baru, pilih `Glue 4.0` kapan Anda membuat pekerjaan.

- Di konsol, pilih `Spark 3.3, Python 3 (Glue Version 4.0) or Spark 3.3, Scala 2 (Glue Version 3.0)` di`Glue version`.
- Di AWS Glue Studio, pilih `Glue 4.0 - Supports Spark 3.3, Scala 2, Python 3` di`Glue version`.
- Di API, pilih `GlueVersion` parameter **`4.0`** dalam operasi `CreateJob`API.

Untuk melihat log peristiwa Spark AWS Glue 4.0 yang berasal dari AWS Glue 2.0 atau sebelumnya, luncurkan server riwayat Spark yang ditingkatkan untuk AWS Glue 4.0 menggunakan AWS CloudFormation atau Docker.

Daftar periksa migrasi

Tinjau daftar periksa ini untuk migrasi:

> **Note**
>
> Untuk item daftar periksa yang terkait dengan AWS Glue 3.0, lihat. Daftar cek migrasi

- Apakah pustaka Python eksternal pekerjaan Anda bergantung pada Python 2.7/3.6?
  - Perbarui pustaka dependen dari Python 2.7/3.6 ke Python 3.10 karena Spark 3.3.0 sepenuhnya menghapus dukungan Python 2.7 dan 3.6.

Migrasi dari AWS Glue 3.0 ke AWS Glue 4.0

Perhatikan perubahan berikut saat bermigrasi:

- Semua parameter pekerjaan yang ada dan fitur utama yang ada di AWS Glue 3.0 akan ada di AWS Glue 4.0.
- AWS Glue3.0 menggunakan Spark 3.1.1 yang dioptimalkan oleh Amazon EMR, dan AWS Glue 4.0 menggunakan Spark 3.3.0 yang dioptimalkan oleh Amazon EMR.

  Beberapa perubahan Spark saja mungkin memerlukan revisi skrip Anda untuk memastikan bahwa fitur yang dihapus tidak direferensikan.
- AWS Glue4.0 juga dilengkapi pembaruan untuk EMRFS dan Hadoop. Untuk versi tertentu, lihatLampiran A: Peningkatan ketergantungan penting.
- AWSSDK yang disediakan dalam pekerjaan ETL sekarang ditingkatkan dari 1,11 menjadi 1,12.
- Semua pekerjaan Python akan menggunakan Python versi 3.10. Sebelumnya, Python 3.7 digunakan di 3.0. AWS Glue

  Akibatnya, beberapa pymodules yang dibawa out-of-the-box oleh AWS Glue ditingkatkan.
- Log4j telah ditingkatkan ke Log4j2.
  - Untuk informasi tentang jalur migrasi Log4j2, lihat dokumentasi Log4j.

  - Anda harus mengganti nama file log4j.properties kustom sebagai file log4j2.properties sebagai gantinya, dengan properti log4j2 yang sesuai.
- Untuk memigrasikan konektor tertentu, lihatKonektor dan migrasi driver JDBC untuk 4.0 AWS Glue.
- AWSEncryption SDK ditingkatkan dari 1.x ke 2.x. AWS Gluepekerjaan yang menggunakan konfigurasi AWS Glue keamanan dan pekerjaan yang bergantung pada dependensi SDK AWS Enkripsi yang disediakan dalam runtime terpengaruh. Lihat petunjuk untuk migrasi AWS Glue pekerjaan.

  Anda dapat dengan aman memutakhirkan pekerjaan AWS Glue 2.0/3.0 ke pekerjaan AWS Glue 4.0 karena AWS Glue 2.0/3.0 sudah berisi versi jembatan SDK AWS Enkripsi.

Lihat dokumentasi migrasi Spark:

- Upgrade dari Spark SQL 3.1 ke 3.2
- Upgrade dari Spark SQL 3.2 ke 3.3

Migrasi dari AWS Glue 2.0 ke 4.0 AWS Glue

Perhatikan perubahan berikut saat bermigrasi:

> **Note**
>
> Untuk langkah-langkah migrasi yang terkait dengan AWS Glue 3.0, lihatMigrasi dari AWS Glue 3.0 ke AWS Glue 4.0.

- Semua parameter pekerjaan yang ada dan fitur utama yang ada di AWS Glue 2.0 akan ada di AWS Glue 4.0.
- Komitter yang dioptimalkan EMRFS S3 untuk menulis data Parket ke Amazon S3 diaktifkan secara default sejak 3.0. AWS Glue Namun, Anda masih dapat menonaktifkannya dengan menyetel `--enable-s3-parquet-optimized-committer` `kefalse`.
- AWS Glue2.0 menggunakan sumber terbuka Spark 2.4 dan AWS Glue 4.0 menggunakan Spark 3.3.0 yang dioptimalkan oleh Amazon EMR.
  - Beberapa perubahan Spark saja mungkin memerlukan revisi skrip Anda untuk memastikan bahwa fitur yang dihapus tidak direferensikan.

  - Misalnya, Spark 3.3.0 tidak mengaktifkan UDF yang tidak diketik SCALA, tetapi Spark 2.4 mengizinkannya.
- AWSSDK yang disediakan dalam pekerjaan ETL sekarang ditingkatkan dari 1,11 menjadi 1,12.
- AWS Glue4.0 juga dilengkapi pembaruan untuk EMRFS, driver JDBC yang diperbarui, dan inklusi pengoptimalan tambahan ke Spark sendiri yang disediakan oleh. AWS Glue
- Scala diperbarui ke 2.12 dari 2.11, dan Scala 2.12 tidak kompatibel dengan Scala 2.11.
- Python 3.10 adalah versi default yang digunakan untuk skrip Python, karena 2.0 AWS Glue hanya menggunakan Python 3.7 dan 2.7.
  - Python 2.7 tidak didukung dengan Spark 3.3.0. Pekerjaan apa pun yang meminta Python 2 dalam konfigurasi pekerjaan akan gagal dengan file. IllegalArgumentException
  - Mekanisme baru untuk menginstal modul Python tambahan tersedia sejak AWS Glue 2.0.
- Beberapa pembaruan ketergantungan, disorot dalamLampiran A: Peningkatan ketergantungan penting.
- File JAR tambahan apa pun yang disediakan dalam pekerjaan AWS Glue 2.0 yang ada mungkin membawa dependensi yang bertentangan karena ada peningkatan di beberapa dependensi di 4.0 dari 2.0. Anda dapat menghindari konflik classpath di AWS Glue 4.0 dengan parameter `--user-jars-first` AWS Glue pekerjaan.
- AWS Glue4.0 menggunakan Spark 3.3. Dimulai dengan Spark 3.1, terjadi perubahan perilaku memuat/menyimpan stempel waktu dari/ke file parket. Untuk detail selengkapnya, lihat Memutakhirkan dari Spark SQL 3.0 ke 3.1.

  Kami merekomendasikan untuk mengatur parameter berikut saat membaca/menulis data parket yang berisi kolom stempel waktu. Menyetel parameter tersebut dapat menyelesaikan masalah ketidakcocokan kalender yang terjadi selama peningkatan Spark 2 ke Spark 3, untuk AWS Glue Dynamic Frame dan Spark Data Frame. Gunakan opsi CORRECTED untuk membaca nilai datetime apa adanya; dan opsi LEGACY untuk rebase nilai datetime sehubungan dengan perbedaan kalender selama membaca.

  ```
  - Key: --conf
  - Value: spark.sql.legacy.parquet.int96RebaseModeInRead=[CORRECTED|LEGACY] --
  conf spark.sql.legacy.parquet.int96RebaseModeInWrite=[CORRECTED|LEGACY] --conf
   spark.sql.legacy.parquet.datetimeRebaseModeInRead=[CORRECTED|LEGACY]
  ```
- Untuk memigrasikan konektor tertentu, lihatKonektor dan migrasi driver JDBC untuk 4.0 AWS Glue.
- AWSEncryption SDK ditingkatkan dari 1.x ke 2.x. AWS Gluepekerjaan yang menggunakan konfigurasi AWS Glue keamanan dan pekerjaan yang bergantung pada dependensi SDK AWS

Enkripsi yang disediakan dalam runtime terpengaruh. Lihat petunjuk ini untuk migrasi AWS Glue pekerjaan:

- Anda dapat dengan aman memutakhirkan pekerjaan AWS Glue 2.0 ke pekerjaan AWS Glue 4.0 karena AWS Glue 2.0 sudah berisi versi jembatan SDK AWS Enkripsi.

Lihat dokumentasi migrasi Spark:

- Upgrade dari Spark SQL 2.4 ke 3.0
- Upgrade dari Spark SQL 3.1 ke 3.2
- Upgrade dari Spark SQL 3.2 ke 3.3
- Perubahan perilaku Datetime diharapkan sejak Spark 3.0.

Migrasi dari AWS Glue 1.0 ke 4.0 AWS Glue

Perhatikan perubahan berikut saat bermigrasi:

- AWS Glue1.0 menggunakan sumber terbuka Spark 2.4 dan AWS Glue 4.0 menggunakan Spark 3.3.0 yang dioptimalkan oleh Amazon EMR.
  - Beberapa perubahan Spark saja mungkin memerlukan revisi skrip Anda untuk memastikan bahwa fitur yang dihapus tidak direferensikan.
  - Misalnya, Spark 3.3.0 tidak mengaktifkan UDF yang tidak diketik SCALA, tetapi Spark 2.4 mengizinkannya.
- Semua pekerjaan di AWS Glue 4.0 akan dijalankan dengan waktu startup yang ditingkatkan secara signifikan. Pekerjaan Spark akan ditagih dalam kenaikan 1 detik dengan durasi penagihan minimum 10x lebih rendah karena latensi startup akan meningkat dari maksimum 10 menit menjadi maksimum 1 menit.
- Perilaku logging telah berubah secara signifikan di AWS Glue 4.0, Spark 3.3.0 memiliki persyaratan minimum Log4j2.
- Beberapa pembaruan ketergantungan, disorot dalam lampiran.
- Scala juga diperbarui ke 2.12 dari 2.11, dan Scala 2.12 tidak kompatibel dengan Scala 2.11.
- Python 3.10 juga merupakan versi default yang digunakan untuk skrip Python, karena 0.9 AWS Glue hanya menggunakan Python 2.

  Python 2.7 tidak didukung dengan Spark 3.3.0. Pekerjaan apa pun yang meminta Python 2 dalam konfigurasi pekerjaan akan gagal dengan file. IllegalArgumentException

- Mekanisme baru untuk menginstal modul Python tambahan melalui pip tersedia sejak 2.0. AWS Glue Untuk informasi selengkapnya, lihat Menginstal modul Python tambahan dengan pip di 2.0+. AWS Glue
- AWS Glue4.0 tidak berjalan di Apache YARN, jadi pengaturan YARN tidak berlaku.
- AWS Glue4.0 tidak memiliki Hadoop Distributed File System (HDFS).
- File JAR tambahan apa pun yang disediakan dalam pekerjaan AWS Glue 1.0 yang ada mungkin membawa dependensi yang bertentangan karena ada peningkatan di beberapa dependensi di 4.0 dari 1.0. Kami mengaktifkan AWS Glue 4.0 dengan parameter `--user-jars-first` AWS Glue pekerjaan secara default, untuk menghindari masalah ini.
- AWS Glue4.0 mendukung penskalaan otomatis. Oleh karena itu, ExecutorAllocationManager metrik akan tersedia saat penskalaan otomatis diaktifkan.
- Dalam pekerjaan AWS Glue versi 4.0, Anda menentukan jumlah pekerja dan jenis pekerja, tetapi tidak menentukan`maxCapacity`.
- AWS Glue4.0 belum mendukung transformasi pembelajaran mesin.
- Untuk memigrasikan konektor tertentu, lihatKonektor dan migrasi driver JDBC untuk 4.0 AWS Glue.
- AWSEncryption SDK ditingkatkan dari 1.x ke 2.x. AWS Gluepekerjaan yang menggunakan konfigurasi AWS Glue keamanan dan pekerjaan yang bergantung pada dependensi SDK AWS Enkripsi yang disediakan dalam runtime terpengaruh. Lihat petunjuk ini untuk migrasi AWS Glue pekerjaan.
  - Anda tidak dapat memigrasikan pekerjaan AWS Glue 0.9/1.0 ke pekerjaan AWS Glue 4.0 secara langsung. Ini karena ketika memutakhirkan langsung ke versi 2.x atau yang lebih baru dan mengaktifkan semua fitur baru dengan segera, SDK AWS Enkripsi tidak akan dapat mendekripsi ciphertext yang dienkripsi di bawah versi SDK Enkripsi sebelumnya. AWS
  - Untuk memutakhirkan dengan aman, pertama-tama kami sarankan Anda bermigrasi ke pekerjaan AWS Glue 2.0/3.0 yang berisi versi jembatan SDK AWS Enkripsi. Jalankan pekerjaan sekali untuk menggunakan versi jembatan AWS Encryption SDK.
  - Setelah selesai, Anda dapat dengan aman memigrasikan pekerjaan AWS Glue 2.0/3.0 ke 4.0. AWS Glue

Lihat dokumentasi migrasi Spark:

- Upgrade dari Spark SQL 2.4 ke 3.0
- Upgrade dari Spark SQL 3.0 ke 3.1
- Upgrade dari Spark SQL 3.1 ke 3.2

- Upgrade dari Spark SQL 3.2 ke 3.3
- Perubahan perilaku Datetime diharapkan sejak Spark 3.0.

Migrasi dari AWS Glue 0,9 ke 4,0 AWS Glue

Perhatikan perubahan berikut saat bermigrasi:

- AWS Glue0.9 menggunakan Spark 2.2.1 dan AWS Glue 4.0 open-source menggunakan Spark 3.3.0 yang dioptimalkan oleh Amazon EMR.
  - Beberapa perubahan Spark saja mungkin memerlukan revisi skrip Anda untuk memastikan bahwa fitur yang dihapus tidak direferensikan.
  - Misalnya, Spark 3.3.0 tidak mengaktifkan UDF yang tidak diketik Scala, tetapi Spark 2.2 mengizinkannya.
- Semua pekerjaan di AWS Glue 4.0 akan dijalankan dengan waktu startup yang ditingkatkan secara signifikan. Pekerjaan Spark akan ditagih dalam kenaikan 1 detik dengan durasi penagihan minimum 10x lebih rendah karena latensi startup akan meningkat dari maksimum 10 menit menjadi maksimum 1 menit.
- Perilaku logging telah berubah secara signifikan sejak AWS Glue 4.0, Spark 3.3.0 memiliki persyaratan minimum Log4j2 seperti yang disebutkan di sini (https://spark.apache.org/docs/latest/ .html# -32-to-33). core-migration-guide upgrading-from-core
- Beberapa pembaruan ketergantungan, disorot dalam lampiran.
- Scala juga diperbarui ke 2.12 dari 2.11, dan Scala 2.12 tidak kompatibel dengan Scala 2.11.
- Python 3.10 juga merupakan versi default yang digunakan untuk skrip Python, karena 0.9 AWS Glue hanya menggunakan Python 2.
  - Python 2.7 tidak didukung dengan Spark 3.3.0. Pekerjaan apa pun yang meminta Python 2 dalam konfigurasi pekerjaan akan gagal dengan file. IllegalArgumentException
  - Mekanisme baru untuk menginstal modul Python tambahan melalui pip tersedia.
- AWS Glue4.0 tidak berjalan di Apache YARN, jadi pengaturan YARN tidak berlaku.
- AWS Glue4.0 tidak memiliki Hadoop Distributed File System (HDFS).
- File JAR tambahan apa pun yang disediakan dalam AWS Glue 0.9 pekerjaan yang ada mungkin membawa dependensi yang bertentangan karena ada peningkatan di beberapa dependensi di 3.0 dari 0.9. Anda dapat menghindari konflik classpath di AWS Glue 3.0 dengan parameter `--user-jars-first` AWS Glue pekerjaan.

- AWS Glue4.0 mendukung penskalaan otomatis. Oleh karena itu, ExecutorAllocationManager metrik akan tersedia saat penskalaan otomatis diaktifkan.
- Dalam pekerjaan AWS Glue versi 4.0, Anda menentukan jumlah pekerja dan jenis pekerja, tetapi tidak menentukan`maxCapacity`.
- AWS Glue4.0 belum mendukung transformasi pembelajaran mesin.
- Untuk memigrasikan konektor tertentu, lihatKonektor dan migrasi driver JDBC untuk 4.0 AWS Glue.
- AWSEncryption SDK ditingkatkan dari 1.x ke 2.x. AWS Gluepekerjaan yang menggunakan konfigurasi AWS Glue keamanan dan pekerjaan yang bergantung pada dependensi SDK AWS Enkripsi yang disediakan dalam runtime akan terpengaruh. Lihat petunjuk ini untuk migrasi AWS Glue pekerjaan.
  - Anda tidak dapat memigrasikan pekerjaan AWS Glue 0.9/1.0 ke pekerjaan AWS Glue 4.0 secara langsung. Ini karena ketika memutakhirkan langsung ke versi 2.x atau yang lebih baru dan mengaktifkan semua fitur baru dengan segera, SDK AWS Enkripsi tidak akan dapat mendekripsi ciphertext yang dienkripsi di bawah versi SDK Enkripsi sebelumnya. AWS
  - Untuk memutakhirkan dengan aman, pertama-tama kami sarankan Anda bermigrasi ke pekerjaan AWS Glue 2.0/3.0 yang berisi versi jembatan SDK AWS Enkripsi. Jalankan pekerjaan sekali untuk menggunakan versi jembatan AWS Encryption SDK.
  - Setelah selesai, Anda dapat dengan aman memigrasikan pekerjaan AWS Glue 2.0/3.0 ke 4.0. AWS Glue

Lihat dokumentasi migrasi Spark:

- Upgrade dari Spark SQL 2.2 ke 2.3
- Upgrade dari Spark SQL 2.3 ke 2.4
- Upgrade dari Spark SQL 2.4 ke 3.0
- Upgrade dari Spark SQL 3.0 ke 3.1
- Upgrade dari Spark SQL 3.1 ke 3.2
- Upgrade dari Spark SQL 3.2 ke 3.3
- Perubahan perilaku Datetime diharapkan sejak Spark 3.0.

Konektor dan migrasi driver JDBC untuk 4.0 AWS Glue

Untuk versi JDBC dan konektor data lake yang ditingkatkan, lihat:

- Lampiran B: Peningkatan driver JDBC
- Lampiran C: Peningkatan konektor

Hudi

- Peningkatan dukungan Spark SQL:
  - Melalui `Call Procedure` perintah, ada dukungan tambahan untuk upgrade, downgrade, bootstrap, clean, dan repair. `Create/Drop/Show/Refresh Index`sintaks dimungkinkan di Spark SQL.
  - Kesenjangan kinerja telah ditutup antara penggunaan melalui Spark DataSource sebagai lawan dari Spark SQL. Datasource menulis di masa lalu dulunya lebih cepat dari SQL.
  - Semua generator kunci bawaan mengimplementasikan operasi API khusus SPARK yang lebih berkinerja tinggi.
  - Mengganti transformasi UDF dalam `insert` operasi massal dengan transformasi RDD untuk mengurangi biaya penggunaan. SerDe
  - Spark SQL dengan Hudi membutuhkan `primaryKey` untuk ditentukan oleh `tblproperites` atau opsi dalam pernyataan SQL. Untuk operasi pembaruan dan penghapusan, `preCombineField` diperlukan juga.
- Setiap tabel Hudi yang dibuat sebelum versi 0.10.0 tanpa `primaryKey` perlu dibuat ulang dengan `primaryKey` bidang sejak versi 0.10.0.

PostgreSQL

- Beberapa kerentanan (CVE) telah diatasi.
- Java 8 didukung secara native.
- Jika pekerjaan menggunakan Array Array, dengan pengecualian array byte, skenario ini dapat diperlakukan sebagai array multidimensi.

MongoDB

- Konektor MongoDB saat ini mendukung Spark versi 3.1 atau yang lebih baru dan MongoDB versi 4.0 atau yang lebih baru.

- Karena peningkatan konektor, beberapa nama properti berubah. Misalnya, nama properti URI diubah menjadi`connection.uri`. Untuk informasi lebih lanjut tentang opsi saat ini, lihat blog MongoDB Spark Connector.
- Menggunakan MongoDB 4.0 yang dihosting oleh Amazon DocumentDB memiliki beberapa perbedaan fungsional. Untuk informasi lebih lanjut, lihat topik-topik ini:
  - Perbedaan Fungsional: Amazon DocumentDB dan MongoDB
  - Didukung MongoDB API, Operasi, dan Tipe Data.
- Opsi "partisi" dibatasi untuk`ShardedPartitioner`,`PaginateIntoPartitionsPartitioner`, dan. `SinglePartitionPartitioner` Itu tidak dapat menggunakan default `SamplePartitioner` dan `PaginateBySizePartitioner` untuk Amazon DocumentDB karena operator panggung tidak mendukung API MongoDB. Untuk informasi selengkapnya, lihat API, Operasi, dan Jenis Data MongoDB yang Didukung.

Danau Delta

- Delta Lake sekarang mendukung perjalanan waktu di SQL untuk menanyakan data lama dengan mudah. Dengan pembaruan ini, perjalanan waktu sekarang tersedia baik di Spark SQL maupun melalui API. DataFrame Support telah ditambahkan untuk versi TIMESTAMP saat ini di SQL.
- Spark 3.3 memperkenalkan Trigger. AvailableNowuntuk menjalankan kueri streaming sebagai setara dengan `Trigger.Once` untuk kueri batch. Dukungan ini juga tersedia saat menggunakan tabel Delta sebagai sumber streaming.
- Support untuk SHOW COLUMNS untuk mengembalikan daftar kolom dalam tabel.
- Support untuk DESKRIPTION DETAIL di Scala dan DeltaTable Python API. Ini mengambil informasi rinci tentang tabel Delta menggunakan DeltaTable API atau Spark SQL.
- Support untuk mengembalikan metrik operasi dari perintah SQL Delete, Merge, dan Update. Sebelumnya perintah SQL ini mengembalikan kosong DataFrame, sekarang mereka mengembalikan DataFrame dengan metrik yang berguna tentang operasi yang dilakukan.
- Optimalkan peningkatan kinerja:
  - Atur opsi konfigurasi `spark.databricks.delta.optimize.repartition.enabled=true` untuk digunakan `repartition(1)` alih-alih `coalesce(1)` dalam perintah Optimalkan untuk kinerja yang lebih baik saat memadatkan banyak file kecil.
  - Peningkatan kinerja dengan menggunakan pendekatan berbasis antrian untuk memparalelkan pekerjaan pemadatan.
- Perubahan penting lainnya:

- Support untuk menggunakan variabel dalam perintah VACUUM dan OPTIMIZE SQL.
- Perbaikan untuk CONVERT TO DELTA dengan tabel katalog termasuk:
  - Isi otomatis skema partisi dari katalog saat tidak disediakan.
  - Gunakan informasi partisi dari katalog untuk menemukan file data yang akan dikomit alih-alih melakukan pemindaian direktori lengkap. Alih-alih melakukan semua file data dalam direktori tabel, hanya file data di bawah direktori partisi aktif yang akan dilakukan.
- Support for Change Data Feed (CDF) batch membaca pada tabel yang diaktifkan pemetaan kolom saat DROP COLUMN dan RENAME COLUMN belum digunakan. Untuk informasi lebih lanjut, lihat dokumentasi Delta Lake.
- Tingkatkan performa perintah Update dengan mengaktifkan pemangkasan skema di lintasan pertama.

Gunung Es Apache

- Menambahkan beberapa peningkatan kinerja untuk perencanaan pemindaian dan kueri Spark.
- Menambahkan klien katalog REST umum yang menggunakan komit berbasis perubahan untuk menyelesaikan konflik komit di sisi layanan.
- `AS 0F`sintaks untuk kueri perjalanan waktu SQL didukung.
- Ditambahkan merge-on-read dukungan untuk MERGE dan UPDATE query.
- Ditambahkan dukungan untuk menulis ulang partisi menggunakan Z-order.
- Menambahkan spesifikasi dan implementasi untuk Puffin, format untuk statistik besar dan gumpalan indeks, seperti sketsa Theta atau filter mekar.
- Menambahkan antarmuka baru untuk mengkonsumsi data secara bertahap (baik pemindaian append dan changelog).
- Menambahkan dukungan untuk operasi massal dan pembacaan berkisar ke antarmuka FileIO.
- Menambahkan lebih banyak tabel metadata untuk menampilkan file hapus di pohon metadata.
- Perilaku drop table berubah. Di Iceberg 0.13.1, menjalankan `DROP TABLE` menghapus tabel dari katalog dan menghapus isi tabel juga. Di Iceberg 1.0.0, `DROP TABLE` hanya menghapus tabel dari katalog. Untuk menghapus isi tabel gunakan`DROP TABLE PURGE`.
- Pembacaan vektor parket diaktifkan secara default di Iceberg 1.0.0. Jika Anda ingin menonaktifkan pembacaan vektor, setel ke. `read.parquet.vectorization.enabled` `false`

Oracle

Perubahan kecil.

MySQL

Perubahan kecil.

Amazon Redshift

AWS Glue4.0 memiliki konektor Amazon Redshift baru dengan driver JDBC baru. Untuk informasi tentang penyempurnaan dan cara bermigrasi dari AWS Glue versi sebelumnya, lihat. the section called “Koneksi Redshift”

Lampiran A: Peningkatan ketergantungan penting

Berikut ini adalah peningkatan ketergantungan:

Dependensi	Versi dalam AWS Glue 4.0	Versi dalam AWS Glue 3.0	Versi dalam AWS Glue 2.0	Versi dalam AWS Glue 1.0
Spark	3.3.0-amzn-1	3.1.1-amzn-0	2.4.3	2.4.3
Hadoop	3.3.3-amzn-0	3.2.1-amzn-3	2.8.5-amzn-5	2.8.5-amzn-1
Skala	2.12	2.12	2.11	2.11
Jackson	2.13.3	2.10.x	2.7.x	2.7.x
Hive	2.3.9-amzn-2	2.3.7-amzn-4	1.2	1.2
EMRFS	2.54.0	2.46.0	2.38.0	2.30.0
JSON4	3.7.0-M11	3.6.6	3.5.x	3.5.x
Panah	7.0.0	2.0.0	0.10.0	0.10.0
AWS GlueKlien Katalog Data	3.7.0	3.0.0	1.10.0	T/A
Python	3.10	3.7	2.7 & 3.6	2.7 & 3.6

Dependensi	Versi dalam AWS Glue 4.0	Versi dalam AWS Glue 3.0	Versi dalam AWS Glue 2.0	Versi dalam AWS Glue 1.0
Boto	1.26	1.18	1.12	T/A

Lampiran B: Peningkatan driver JDBC

Berikut ini adalah upgrade driver JDBC:

Driver	Versi driver JDBC di versi sebelumnya AWS Glue	Versi driver JDBC di 3.0 AWS Glue	Versi driver JDBC di 4.0 AWS Glue
MySQL	5.1	8.0.23	8.0.23
Microsoft SQL Server	6.1.0	7.0.0	9.4.0
Database Oracle	11.2	21.1	21.7
PostgreSQL	42.1.0	42.2.18	42.3.6
MongoDB	2.0.0	4.0.0	4.7.2
Amazon Redshift	redshift-jdbc41-1.2.12.1017	redshift-jdbc41-1.2.12.1017	redshift-jdbc42-2.1.0.16

Lampiran C: Peningkatan konektor

Berikut ini adalah upgrade konektor:

Driver	Versi konektor di AWS Glue 3.0	Versi konektor di AWS Glue 4.0
MongoDB	3.0.0	10.0.4
Hudi	0.10.1	0.12.1
Danau Delta	1.0.0	2.1.0

Driver	Versi konektor di AWS Glue 3.0	Versi konektor di AWS Glue 4.0
Gunung es	0.13.1	1.0.0
DynamoDB	1.11	1.12

Migrasi AWS Glue untuk Ray dari pratinjau ke lingkungan runtime Ray2.4

> **Warning**
>
> Saat Anda menyimpan pekerjaan AWS Glue untuk Ray (pratinjau)AWS Glue Studio, itu akan secara otomatis ditingkatkan ke `Ray2.4` runtime. Jika Anda mengalami masalah kompatibilitas dengan skrip Anda, silakan hubungi dukungan.

Anda harus memigrasikan AWS Glue pekerjaan yang dibuat selama AWS Glue for Ray (pratinjau) ke AWS Glue for Ray. Ini akan melibatkan beberapa perubahan bersamaan pada konfigurasi pekerjaan Anda.

- Di `Runtime` lapangan, berikan nilai `Ray2.4` runtime. Ini akan meningkatkan versi Ray yang mendasarinya dari 2.0.0 ke 2.4.0.
- Pustaka Python tertentu yang disertakan secara default dalam pratinjau tidak lagi disediakan. Jika pekerjaan Anda memanfaatkan AWS SDK untuk panda (awswrangler), dask, modin, atau pymars, Anda harus menyertakan ini sebagai pustaka tambahan. Untuk informasi selengkapnya tentang menyertakan pustaka Python tambahan, lihat. the section called “Modul Python tambahan untuk pekerjaan Ray”
- Jika Anda menggunakan `--additional-python-modules` parameter, parameter yang digunakan untuk mendukung alur kerja ini telah dipecah menjadi `--pip-install` dan`--s3-py-modules`. Untuk informasi selengkapnya tentang parameter ini, lihat the section called “Modul Python tambahan untuk pekerjaan Ray”.
- Jika Anda menggunakan `--auto-scaling-ray-min-workers` parameter, itu telah diganti namanya`--min-workers`.

AWS Gluekebijakan dukungan versi

AWS Glueadalah layanan integrasi data tanpa server yang memudahkan untuk menemukan, menyiapkan, dan menggabungkan data untuk analitik, pembelajaran mesin, dan pengembangan aplikasi. AWS GluePekerjaan berisi logika bisnis yang melakukan pekerjaan integrasi dataAWS Glue. Ada tiga jenis pekerjaan diAWS Glue: Spark (batch dan streaming), Ray dan Python shell. Saat Anda menentukan pekerjaan Anda, Anda menentukan AWS Glue versi, yang mengonfigurasi versi di lingkungan runtime Spark, Ray, atau Python yang mendasarinya. Misalnya: pekerjaan Spark AWS Glue versi 2.0 mendukung Spark 2.4.3 dan Python 3.7.

Kebijakan Support

Terkadang AWS Glue menghentikan dukungan untuk AWS Glue versi lama. Namun, pekerjaan yang berjalan pada versi usang tidak lagi memenuhi syarat untuk mendapatkan dukungan teknis. AWS Gluetidak akan lagi menerapkan patch keamanan atau pembaruan lainnya ke versi yang tidak digunakan lagi. AWS Gluejuga tidak akan menghormati SLA ketika pekerjaan dijalankan pada versi yang tidak digunakan lagi.

Ketika akhir dukungan terjadi untuk AWS Glue versi 2.0 atau yang lebih baru, Anda tidak akan dapat membuat pekerjaan, tetapi hanya mengedit atau menjalankan pekerjaan.

AWS GlueVersi berikut telah mencapai atau dijadwalkan untuk akhir dukungan. Akhir dukungan dimulai pada tengah malam (zona waktu Pasifik) pada tanggal yang ditentukan.

Jenis	Versi Glue	Akhir dari dukungan
Spark	Spark 2.2, Scala 2 (Glue versi 0.9)	6/1/2022
Spark	Spark 2.2, Python 2 (Glue versi 0.9)	6/1/2022
Spark	Spark 2.4, Python 2 (Glue versi 1.0)	6/1/2022
Spark	Spark 2.4, Python 3 (Glue versi 1.0)	9/30/2022

Jenis	Versi Glue	Akhir dari dukungan
Spark	Spark 2.4, Scala 2 (Glue versi 1.0)	9/30/2022
Spark	Glue versi 2.0	1/31/2024
Jenis	Versi Python	Akhir dari dukungan
Cangkang Python	Python 2 (Glue Versi 1.0)	6/1/2022
Jenis	Versi notebook	Akhir dari dukungan
Titik akhir pengembangan	Notebook Zeppelin	9/30/2022

AWSsangat menyarankan agar Anda memigrasikan pekerjaan Anda ke versi yang didukung.

Untuk informasi tentang memigrasi pekerjaan Spark Anda ke AWS Glue versi terbaru, lihat Memigrasi AWS Glue pekerjaan ke AWS Glue versi 4.0.

Untuk memigrasikan pekerjaan shell Python Anda ke versi terbaru: AWS Glue

- Di konsol, pilih`Python 3 (Glue Version 4.0)`.
- Di CreateJob/UpdateJobAPI, atur `GlueVersion` parameter ke`2.0`, dan `PythonVersion` ke `3` di bawah `Command` parameter. `GlueVersion`Konfigurasi tidak memengaruhi perilaku pekerjaan shell Python, jadi tidak ada keuntungan untuk menambah. `GlueVersion`
- Anda perlu membuat skrip pekerjaan Anda kompatibel dengan Python 3.

> **Note**
>
> Seluruh AWS Wilayah yang diluncurkan sebelum peluncuran Wilayah Jakarta, Indonesia (ap-tenggara 3) pada Agustus 2022 memiliki daftar izin pelanggan yang diizinkan menjalankan job run AWS Glue versi 0.9/1.0. Di Wilayah lama ini, Anda dapat membuat pekerjaan dengan nilai nol dan itu akan default ke versi 0.9/1.0 tergantung pada Wilayah. Untuk AWS Wilayah yang diluncurkan nanti, Anda harus secara eksplisit menyetel AWS Glue versi di API. AWS Gluetidak lagi menerima parameter null. Jika Anda melewati 0.9 atau 1.0 dalam parameter, Anda menemukan kesalahan “Glue Version 0.9 (or) 1.0 is not supported.”

Bekerja dengan pekerjaan Spark di AWS Glue

Memberikan informasi tentang AWS Glue untuk pekerjaan Spark ETL.

Topik

AWS Glueparameter pekerjaan

Saat membuat pekerjaan AWS Glue, Anda menetapkan beberapa bidang standar, seperti `Role` dan`WorkerType`. Anda dapat memberikan informasi konfigurasi tambahan melalui `Argument` bidang (Parameter Pekerjaan di konsol). Di bidang ini, Anda dapat memberikan pekerjaan AWS Glue dengan argumen (parameter) yang tercantum dalam topik ini. Untuk informasi selengkapnya tentang AWS Glue Job API, lihatthe section called “Tugas”.

Mengatur parameter pekerjaan

Anda dapat mengonfigurasi pekerjaan melalui konsol di tab Detail pekerjaan, di bawah judul Parameter Pekerjaan. Anda juga dapat mengonfigurasi pekerjaan AWS CLI melalui pengaturan `DefaultArguments` atau `NonOverridableArguments` pekerjaan, atau pengaturan `Arguments` pada pekerjaan. Argumen yang ditetapkan pada pekerjaan akan diteruskan setiap kali pekerjaan dijalankan, sedangkan argumen yang ditetapkan pada job run hanya akan diteruskan untuk menjalankan individu tersebut.

Misalnya, berikut ini adalah sintaks untuk menjalankan pekerjaan menggunakan `--arguments` untuk menetapkan parameter pekerjaan.

```
$ aws glue start-job-run --job-name "CSV to CSV" --arguments='--scriptLocation="s3://
my_glue/libraries/test_lib.py"'
```

Mengakses parameter pekerjaan

Saat menulis skrip AWS Glue, Anda mungkin ingin mengakses nilai parameter pekerjaan untuk mengubah perilaku kode Anda sendiri. Kami menyediakan metode pembantu untuk melakukannya di perpustakaan kami. Metode ini menyelesaikan nilai parameter job run yang mengesampingkan nilai parameter pekerjaan. Saat menyelesaikan parameter yang ditetapkan di beberapa tempat, job `NonOverridableArguments` akan menggantikan job run`Arguments`, yang akan menggantikan job. `DefaultArguments`

Dengan Python:

Dalam pekerjaan Python, kami menyediakan fungsi bernama. `getResolvedParameters` Untuk informasi selengkapnya, lihat the section called “getResolvedOptions”. Parameter Job tersedia dalam `sys.argv` variabel.

Dalam Scala:

Dalam pekerjaan Scala, kami menyediakan objek bernama`GlueArgParser`. Untuk informasi selengkapnya, lihat the section called “GlueArgParser”. Parameter Job tersedia dalam `sysArgs` variabel.

Referensi parameter Job

AWS Gluemengenali nama argumen berikut yang dapat Anda gunakan untuk menyiapkan lingkungan skrip untuk pekerjaan dan pekerjaan Anda:

`--additional-python-modules`

Daftar dibatasi koma yang mewakili satu set paket Python yang akan diinstal. Anda dapat menginstal paket dari PyPI atau menyediakan distribusi khusus. Entri paket PyPI akan dalam format*`package`*==*`version`*, dengan nama PyPI dan versi paket target Anda. Entri distribusi kustom adalah jalur S3 ke distribusi.

Entri menggunakan pencocokan versi Python untuk mencocokkan paket dan versi. Ini berarti Anda harus menggunakan dua tanda yang sama, seperti==. Ada operator pencocokan versi lain, untuk informasi lebih lanjut lihat PEP 440.

Untuk meneruskan opsi instalasi modul kepip3, gunakan --python-modules-installer-option parameter.

`--auto-scale-within-microbatch`

Nilai default adalah false. Parameter ini hanya dapat digunakan untuk pekerjaan streaming AWS Glue, yang memproses data streaming dalam serangkaian batch mikro, dan penskalaan otomatis harus diaktifkan. Saat menyetel nilai ini ke false, nilai ini menghitung rata-rata pergerakan eksponensial durasi batch untuk batch mikro yang diselesaikan dan membandingkan nilai ini dengan ukuran jendela untuk menentukan apakah akan meningkatkan atau menurunkan jumlah pelaksana. Penskalaan hanya terjadi ketika batch mikro selesai. Saat menyetel nilai ini ke true, selama batch mikro, nilai ini meningkat ketika jumlah tugas Spark tetap sama selama 30 detik, atau pemrosesan batch saat ini lebih besar dari ukuran jendela. Jumlah eksekutor akan turun jika eksekutor telah menganggur selama lebih dari 60 detik, atau rata-rata pergerakan eksponensial durasi batch rendah.

`--class`

Kelas Scala yang berfungsi sebagai titik masuk untuk skrip Scala Anda. Ini hanya berlaku jika `--job-language` Anda diatur ke `scala`.

`--continuous-log-conversionPattern`

Menentukan pola log konversi kustom untuk pekerjaan yang diaktifkan untuk logging berkelanjutan. Pola konversi hanya berlaku untuk log driver dan log pelaksana saja. Itu tidak mempengaruhi bilah kemajuan AWS Glue.

`--continuous-log-logGroup`

Menentukan nama grup CloudWatch log Amazon kustom untuk pekerjaan yang diaktifkan untuk logging berkelanjutan.

`--continuous-log-logStreamPrefix`

Menentukan awalan aliran CloudWatch log kustom untuk pekerjaan diaktifkan untuk logging berkelanjutan.

`--customer-driver-env-vars` dan **`--customer-executor-env-vars`**

Parameter ini mengatur variabel lingkungan pada sistem operasi masing-masing untuk setiap pekerja (driver atau pelaksana). Anda dapat menggunakan parameter ini saat membangun platform dan kerangka kerja khusus di atas AWS Glue, untuk memungkinkan pengguna Anda menulis pekerjaan di atasnya. Mengaktifkan dua flag ini akan memungkinkan Anda untuk mengatur variabel lingkungan yang berbeda pada driver dan eksekutor masing-masing tanpa harus menyuntikkan logika yang sama dalam skrip pekerjaan itu sendiri.

Contoh penggunaan

Berikut ini adalah contoh penggunaan parameter ini:

```
"–customer-driver-env-vars", "CUSTOMER_KEY1=VAL1,CUSTOMER_KEY2=\"val2,val2 val2\"",
"–customer-executor-env-vars", "CUSTOMER_KEY3=VAL3,KEY4=VAL4"
```

Menyetel ini dalam argumen job run sama dengan menjalankan perintah berikut:

Di pengemudi:

- ekspor CUSTOMER_KEY1 = VAL1
- ekspor customer_key2 = “val2, val2 val2”

Dalam pelaksana:

- ekspor CUSTOMER_KEY3 = VAL3

Kemudian, dalam skrip pekerjaan itu sendiri, Anda dapat mengambil variabel lingkungan menggunakan `os.environ.get("CUSTOMER_KEY1")` atau`System.getenv("CUSTOMER_KEY1")`.

Sintaks yang ditegakkan

Perhatikan standar berikut saat mendefinisikan variabel lingkungan:

- Setiap kunci harus memiliki`CUSTOMER_ prefix`.

  Misalnya: for`"CUSTOMER_KEY3=VAL3,KEY4=VAL4"`, `KEY4=VAL4` akan diabaikan dan tidak disetel.
- Setiap pasangan kunci dan nilai harus digambarkan dengan koma tunggal.

  Misalnya: `"CUSTOMER_KEY3=VAL3,CUSTOMER_KEY4=VAL4"`
- Jika “nilai” memiliki spasi atau koma, maka itu harus didefinisikan dalam kutipan.

  Misalnya: `CUSTOMER_KEY2=\"val2,val2 val2\"`

Sintaks ini secara dekat memodelkan standar pengaturan variabel lingkungan bash.

`--datalake-formats`

Didukung di AWS Glue 3.0 dan versi yang lebih baru.

Menentukan kerangka data lake untuk digunakan. AWS Glue menambahkan file JAR yang diperlukan untuk kerangka kerja yang Anda tentukan ke dalam. `classpath` Untuk informasi selengkapnya, lihat Menggunakan kerangka kerja data lake dengan pekerjaan AWS Glue ETL.

Anda dapat menentukan satu atau lebih dari nilai-nilai berikut, dipisahkan dengan koma:

- `hudi`
- `delta`
- `iceberg`

Misalnya, berikan argumen berikut untuk menentukan ketiga kerangka kerja.

```
'--datalake-formats': 'hudi,delta,iceberg'
```

`--disable-proxy-v2`

Nonaktifkan proxy layanan untuk mengizinkan panggilan AWS layanan ke Amazon S3, CloudWatch, dan AWS Glue berasal dari skrip Anda melalui VPC Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi AWS panggilan untuk melalui VPC Anda. Untuk menonaktifkan proxy layanan, atur nilai paramater ini ke`true`.

`--enable-auto-scaling`

Mengaktifkan penskalaan otomatis dan penagihan per pekerja saat Anda menetapkan nilainya. `true`

`--enable-continuous-cloudwatch-log`

Mengaktifkan pencatatan berkelanjutan secara real-time untuk AWS Glue pekerjaan. Anda dapat melihat log pekerjaan Apache Spark real-time. CloudWatch

`--enable-continuous-log-filter`

Menentukan filter standar (`true`) atau tidak ada filter (`false`) saat Anda membuat atau mengedit pekerjaan diaktifkan untuk logging berkelanjutan. Memilih opsi filter standar akan membuang driver/pelaksana Apache Spark dan pesan log heartbeat Apache Hadoop YARN yang tidak berguna. Memilih tanpa filter akan memberikan semua pesan log.

`--enable-glue-datacatalog`

Memungkinkan Anda menggunakan Katalog Data AWS Glue sebagai metastore Apache Spark Hive. Untuk mengaktifkan fitur ini, atur nilainya ke`true`.

`--enable-job-insights`

Mengaktifkan pemantauan analisis kesalahan tambahan dengan wawasan AWS Glue job run. Lihat perinciannya di the section called “Pemantauan dengan wawasan menjalankan AWS Glue pekerjaan”. Secara default, nilai disetel ke `true` dan wawasan job run diaktifkan.

Opsi ini tersedia untuk AWS Glue versi 2.0 dan 3.0.

`--enable-metrics`

Mengaktifkan pengumpulan metrik untuk pembuatan profil pekerjaan untuk menjalankan pekerjaan ini. Metrik ini tersedia di AWS Glue konsol dan CloudWatch konsol Amazon. Nilai parameter ini tidak relevan. Untuk mengaktifkan fitur ini, Anda dapat memberikan parameter ini dengan nilai apa pun, tetapi `true` disarankan untuk kejelasan. Untuk menonaktifkan fitur ini, hapus parameter ini dari konfigurasi pekerjaan Anda.

`--enable-observability-metrics`

Mengaktifkan satu set metrik Observability untuk menghasilkan wawasan tentang apa yang terjadi di dalam setiap pekerjaan yang dijalankan di halaman Job Runs Monitoring di bawah AWS Glue konsol dan konsol. Amazon CloudWatch Untuk mengaktifkan fitur ini, atur nilai parameter ini ke true. Untuk menonaktifkan fitur ini, atur ke `false` atau hapus parameter ini dari konfigurasi pekerjaan Anda.

`--enable-rename-algorithm-v2`

Menetapkan EMRFS mengubah nama algoritma versi ke versi 2. Ketika tugas Spark menggunakan mode menimpa partisi dinamis, ada kemungkinan bahwa sebuah partisi duplikat yang dibuat. Misalnya, Anda dapat berakhir dengan partisi duplikat seperti `s3://bucket/table/location/p1=1/p1=1`. Di sini, P1 adalah partisi yang sedang ditimpa. Mengubah nama algoritme versi 2 akan memperbaiki masalah ini.

Opsi ini hanya tersedia di AWS Glue versi 1.0.

`--enable-s3-parquet-optimized-committer`

Mengaktifkan committer yang dioptimalkan EMRFS S3 untuk menulis data Parket ke Amazon S3. Anda dapat menyediakan pasangan parameter/nilai melalui AWS Glue konsol saat membuat atau memperbarui pekerjaan. AWS Glue Mengatur nilai ke **`true`** akan mengaktifkan committer. Secara default, bendera dihidupkan di AWS Glue 3.0 dan dimatikan di AWS Glue 2.0.

Untuk informasi selengkapnya, lihat Menggunakan Committer yang Dioptimalkan-S3 EMRFS.

--enable-spark-ui

Saat disetel ke`true`, aktifkan fitur untuk menggunakan Spark UI untuk memantau dan men-debug pekerjaan AWS Glue ETL.

--executor-cores

Jumlah tugas percikan yang dapat berjalan secara paralel. Opsi ini didukung pada AWS Glue 3.0+. Nilai tidak boleh melebihi 2x jumlah vCPU pada tipe pekerja, yaitu 8 on, 16 on, 32 `G.1X` `G.4X` on dan `G.2X` 64 on. `G.8X` Anda harus berhati-hati saat memperbarui konfigurasi ini karena dapat memengaruhi kinerja pekerjaan karena peningkatan paralelisme tugas menyebabkan memori, tekanan disk, serta dapat menghambat sistem sumber dan target (misalnya: ini akan menyebabkan lebih banyak koneksi bersamaan di Amazon RDS).

--extra-files

Jalur Amazon S3 ke file tambahan, seperti file konfigurasi yang AWS Glue menyalin ke direktori kerja skrip Anda sebelum menjalankannya. Beberapa nilai harus path lengkap yang dipisahkan dengan sebuah koma (`,`). Hanya mendukung file individu, tidak mendukung path direktori. Opsi ini tidak didukung untuk jenis pekerjaan Python Shell.

--extra-jars

Jalur Amazon S3 ke `.jar` file Java tambahan yang AWS Glue ditambahkan ke classpath Java sebelum menjalankan skrip Anda. Beberapa nilai harus path lengkap yang dipisahkan dengan sebuah koma (`,`).

--extra-py-files

Jalur Amazon S3 ke modul Python tambahan yang ditambahkan AWS Glue ke jalur Python sebelum menjalankan skrip Anda. Beberapa nilai harus path lengkap yang dipisahkan dengan sebuah koma (`,`). Hanya mendukung file individu, tidak mendukung path direktori.

--job-bookmark-option

Mengontrol perilaku bookmark pekerjaan. Nilai opsi berikut dapat diatur.

--job-bookmark-nilai opsi	Deskripsi
`job-bookmark-enable`	Melacak data yang diproses sebelumnya. Ketika sebuah tugas berjalan, memproses data baru sejak pos pemeriksaan terakhir.

--job-bookmark-nilai opsi	Deskripsi
`job-bookmark-disable`	Selalu memproses seluruh set data. Anda bertanggung jawab untuk mengelola output dari eksekusi tugas sebelumnya.
`job-bookmark-pause`	Memproses data tambahan sejak eksekusi terakhir yang berhasil atau data dalam kisaran yang diidentifikasi oleh subopsi berikutnya, tanpa memperbarui status bookmark terakhir. Anda bertanggung jawab untuk mengelola output dari eksekusi tugas sebelumnya. Dua subopsi tersebut adalah sebagai berikut: • **`job-bookmark-from`** `<from-value>` adalah ID eksekusi yang merepresentasikan semua input yang diproses sampai eksekusi terakhir yang berhasil sebelum dan termasuk ID eksekusi yang ditentukan. Masukan yang sesuai diabaikan. • **`job-bookmark-to`** `<to-value>` adalah ID eksekusi yang merepresentasikan semua input yang diproses sampai eksekusi terakhir yang berhasil sebelum dan termasuk ID eksekusi yang ditentukan. Input yang sesuai tidak termasuk input yang diidentifikasi oleh `<from-value>` diproses oleh tugas. Setiap masukan berikutnya nanti selain dari input ini juga dikecualikan untuk diproses. Status bookmark tugas tidak diperbarui ketika rangkaian opsi ini ditentukan. Subopsi adalah opsional. Namun, bila digunakan, kedua subopsi tersebut harus disediakan.

Misalnya, untuk mengaktifkan bookmark tugas, berikan argumen berikut.

```
'--job-bookmark-option': 'job-bookmark-enable'
```

`--job-language`

Bahasa pemrograman skrip. Nilai ini harus berupa `scala` atau `python`. Jika parameter ini tidak ada, maka nilai default-nya adalah `python`.

`--python-modules-installer-option`

String plaintext yang mendefinisikan opsi yang akan diteruskan `pip3` saat menginstal modul dengan. --additional-python-modules Berikan opsi seperti yang Anda lakukan di baris perintah, dipisahkan oleh spasi dan diawali dengan tanda hubung. Untuk informasi lebih lanjut tentang penggunaan, lihatthe section called “Menginstal modul Python tambahan dengan pip di 2.0+ AWS Glue”.

> **Note**
>
> Opsi ini tidak didukung untuk pekerjaan AWS Glue saat Anda menggunakan Python 3.9.

`--scriptLocation`

Lokasi Amazon Simple Storage Service (Amazon S3) tempat skrip ETL Anda berada (dalam formulir). `s3://path/to/my/script.py` Parameter ini menimpa lokasi skrip yang ditetapkan dalam objek `JobCommand`.

`--spark-event-logs-path`

Menentukan jalur Amazon S3. Saat menggunakan fitur pemantauan UI Spark, AWS Glue siram log peristiwa Spark ke jalur Amazon S3 ini setiap 30 detik ke bucket yang dapat digunakan sebagai direktori sementara untuk menyimpan peristiwa Spark UI.

`--TempDir`

Menentukan jalur Amazon S3 ke bucket yang dapat digunakan sebagai direktori sementara untuk pekerjaan itu.

Misalnya, untuk menetapkan sebuah direktori sementara, berikan argumen berikut.

```
'--TempDir': 's3-path-to-directory'
```

> **Note**
>
> AWS Gluemembuat ember sementara untuk pekerjaan jika ember belum ada di Wilayah. Bucket ini mungkin mengizinkan akses publik. Anda dapat memodifikasi bucket di

Amazon S3 untuk menyetel blok akses publik, atau menghapus bucket nanti setelah semua pekerjaan di Wilayah tersebut selesai.

`--use-postgres-driver`

Saat menyetel nilai ini`true`, ini memprioritaskan driver Postgres JDBC di jalur kelas untuk menghindari konflik dengan driver Amazon Redshift JDBC. Opsi ini hanya tersedia di AWS Glue versi 2.0.

`--user-jars-first`

Saat menyetel nilai ini`true`, ini memprioritaskan file JAR tambahan pelanggan di classpath. Opsi ini hanya tersedia di AWS Glue versi 2.0 atau yang lebih baru.

`--conf`

Mengontrol parameter konfigurasi Spark. Ini untuk kasus penggunaan lanjutan.

`--encryption-type`

Parameter warisan. Perilaku yang sesuai harus dikonfigurasi menggunakan konfigurasi keamanan. untuk informasi selengkapnya tentang konfigurasi keamanan, lihat. the section called “Mengenkripsi data yang ditulis oleh AWS Glue”

AWS Gluemenggunakan argumen berikut secara internal dan Anda tidak boleh menggunakannya:

- `--debug`— Internal keAWS Glue. Jangan diatur.
- `--mode`— Internal keAWS Glue. Jangan diatur.
- `--JOB_NAME`— Internal keAWS Glue. Jangan diatur.
- `--endpoint`— Internal keAWS Glue. Jangan diatur.

AWS Gluemendukung bootstrap lingkungan dengan `site` modul Python yang digunakan `sitecustomize` untuk melakukan penyesuaian khusus situs. Bootstrapping fungsi initilisasi Anda sendiri direkomendasikan untuk kasus penggunaan lanjutan saja dan didukung atas dasar upaya terbaik pada 4.0. AWS Glue

Awalan variabel lingkungan,`GLUE_CUSTOMER`, dicadangkan untuk penggunaan pelanggan.

AWS Glue Spark dan pekerjaan PySpark

Bagian berikut memberikan informasi tentang AWS Glue Spark dan PySpark pekerjaan.

Topik

Menambahkan Spark dan PySpark pekerjaan di AWS Glue

Bagian berikut memberikan informasi tentang menambahkan Spark dan PySpark pekerjaan diAWS Glue.

Topik

Mengkonfigurasi properti pekerjaan untuk pekerjaan Spark di AWS Glue

Sebuah tugas AWS Glue merangkum skrip yang terhubung ke sumber data Anda, memprosesnya, dan kemudian menuliskannya ke target data Anda. Biasanya, sebuah tugas menjalankan skrip extract, transform, and load (ETL). Tugas juga dapat menjalankan skrip Python tujuan umum (tugas shell Python.) pemicu AWS Glue dapat memulai tugas berdasarkan jadwal atau peristiwa, atau sesuai permintaan. Anda dapat memantau eksekusi tugas untuk memahami metrik waktu aktif seperti status penyelesaian, durasi, dan waktu mulai.

Anda dapat menggunakan skrip yang dihasilkan AWS Glue atau Anda dapat memberikan milik Anda sendiri. Dengan skema sumber dan lokasi target atau skema, pembuat AWS Glue kode dapat secara otomatis membuat skrip Apache Spark API (). PySpark Anda dapat menggunakan skrip ini sebagai titik awal dan mengedit skrip tersebut untuk memenuhi tujuan Anda.

AWS Glue dapat menulis file output dalam beberapa format data, termasuk JSON, CSV, ORC (Optimized Row Columnar), Apache Parquet, dan Apache Avro. Untuk beberapa format data, format-format kompresi umum dapat ditulis.

Ada tiga jenis tugas di AWS Glue: Spark, Streaming ETL, dan Python shell.

- Pekerjaan Spark dijalankan di lingkungan Apache Spark yang dikelola oleh. AWS Glue Tugas ini memproses data dalam batch.
- Sebuah tugas ETL streaming mirip dengan tugas Spark, kecuali bahwa ia melakukan ETL pada aliran data. Menggunakan kerangka kerja Apache Spark Structured Streaming. Beberapa fitur tugas Spark tidak tersedia untuk streaming tugas ETL.
- Sebuah tugas shell Python menjalankan skrip Python sebagai shell dan mendukung versi Python yang bergantung pada versi AWS Glue yang Anda gunakan. Anda dapat menggunakan tugas ini untuk menjadwalkan dan menjalankan tugas-tugas yang tidak memerlukan lingkungan Apache Spark.

Mendefinisikan properti pekerjaan untuk pekerjaan Spark

Bila Anda menentukan tugas Anda di konsol AWS Glue, Anda memberikan nilai untuk properti untuk mengontrol lingkungan waktu aktif AWS Glue.

Daftar berikut menjelaskan properti-properti tugas Spark. Untuk properti dari tugas shell Python, lihat Mendefinisikan properti pekerjaan untuk pekerjaan shell Python. Untuk properti dari tugas ETL streaming, lihat the section called “Mendefinisikan properti pekerjaan untuk pekerjaan ETL streaming”.

Properti tercantum dalam urutan tampilannya di tab Tambahkan tugas pada konsol AWS Glue.

Nama

Berikan string UTF-8 dengan panjang maksimum 255 karakter.

Deskripsi

Berikan deskripsi opsional hingga 2048 karakter.

IAM Role

Tentukan IAM role yang digunakan untuk otorisasi ke sumber daya yang digunakan untuk menjalankan tugas dan mengakses penyimpanan data. Untuk informasi selengkapnya tentang izin menjalankan tugas di AWS Glue, lihat Manajemen identitas dan akses untuk AWS Glue.

Tipe

Jenis pekerjaan ETL. Ini diatur secara otomatis berdasarkan jenis sumber data yang Anda pilih.

- Spark menjalankan skrip Apache Spark ETL dengan perintah pekerjaan. `glueetl`
- Spark Streaming menjalankan skrip ETL streaming Apache Spark dengan perintah pekerjaan. `gluestreaming` Untuk informasi selengkapnya, lihat the section called “Lowongan kerja Streaming ETL”.
- Shell Python menjalankan skrip Python dengan perintah job. `pythonshell` Untuk informasi selengkapnya, lihat Mengonfigurasi properti pekerjaan untuk pekerjaan shell Python di AWS Glue.

Versi AWS Glue

Versi AWS Glue menentukan versi Apache Spark dan Python yang tersedia untuk tugas, seperti yang ditentukan dalam tabel berikut.

Versi AWS Glue	Versi Spark dan Python yang didukung
4.0	• Spark 3.3.0 • Python 3.10
3.0	• Spark 3.1.1 • Python 3.7
2.0	• Spark versi 2.4.3 • Python 3.7
1.0	• Spark versi 2.4.3 • Python 2.7 • Python 3.6
0,9	• Spark versi 2.2.1 • Python 2.7

Jenis pekerja

Jenis-jenis pekerja berikut tersedia:

Sumber daya yang tersedia pada AWS Glue pekerja diukur dalam DPU. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori.

- **`G.1X`** — Ketika Anda memilih jenis ini, maka Anda juga memberikan nilai untuk Jumlah pekerja. Setiap pekerja memetakan ke 1 DPU (4 vCPU, memori 16 GB) dengan disk 84GB (sekitar 34GB gratis). Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang skalabel dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
- **`G.2X`** — Ketika Anda memilih jenis ini, maka Anda juga memberikan nilai untuk Jumlah pekerja. Setiap pekerja memetakan ke 2 DPU (8 vCPU, memori 32 GB) dengan disk 128GB (sekitar 77GB gratis). Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang skalabel dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
- **`G.4X`** — Ketika Anda memilih jenis ini, maka Anda juga memberikan nilai untuk Jumlah pekerja. Setiap pekerja memetakan ke 4 DPU (16 vCPU, memori 64 GB) dengan disk 256GB (sekitar 235GB gratis). Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru di AWS Wilayah berikut: AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (Oregon), Asia Pasifik (Singapura), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Kanada (Tengah), Eropa (Frankfurt), Eropa (Irlandia), dan Eropa (Stockholm).
- **`G.8X`** — Ketika Anda memilih jenis ini, maka Anda juga memberikan nilai untuk Jumlah pekerja. Setiap pekerja memetakan ke 8 DPU (32 vCPU, memori 128 GB) dengan disk 512GB (sekitar 487GB gratis). Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru, di AWS Wilayah yang sama seperti yang didukung untuk tipe `G.4X` pekerja.
- **`G.025X`** — Ketika Anda memilih jenis ini, maka Anda juga memberikan nilai untuk Jumlah pekerja. Setiap pekerja memetakan ke 0, 25 DPU (2 vCPU, memori 4 GB) dengan disk 84GB (sekitar 34GB gratis). Kami merekomendasikan jenis pekerja ini untuk pekerjaan streaming volume rendah. Jenis pekerja ini hanya tersedia untuk pekerjaan streaming AWS Glue versi 3.0.

Anda akan dikenakan tarif per jam berdasarkan jumlah DPU yang digunakan untuk menjalankan tugas ETL Anda. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

Untuk tugas AWS Glue versi 1.0 atau tugas sebelumnya, ketika Anda mengkonfigurasi tugas dengan menggunakan konsol dan menentukan Jenis pekerja dengan jenis Standard, maka Kapasitas maksimal diatur dan Jumlah pekerja menjadi nilai Kapasitas maksimal - 1. Jika Anda menggunakan AWS Command Line Interface (AWS CLI) atau AWS SDK, Anda dapat

menentukan parameter Kapasitas maks, atau Anda dapat menentukan jenis Pekerja dan Jumlah pekerja.

Untuk AWS Glue versi 2.0 atau pekerjaan yang lebih baru, Anda tidak dapat menentukan Kapasitas maksimum. Sebaliknya, Anda harus menentukan Jenis pekerja dan Jumlah pekerja.

Bahasa

Kode dalam skrip ETL mendefinisikan logika tugas Anda. Skrip dapat dikodekan dengan Python atau Scala. Anda dapat memilih apakah skrip yang dijalankan tugas dihasilkan oleh AWS Glue atau disediakan oleh Anda. Anda menyediakan nama skrip dan lokasi di Amazon Simple Storage Service (Amazon S3). Konfirmasi bahwa tidak ada file dengan nama yang sama sebagai direktori skrip pada path. Untuk mem-pelajari selengkapnya tentang menulis skrip, lihat AWS Glue panduan pemrograman.

Jumlah pekerja yang diminta

Untuk sebagian besar jenis pekerja, Anda harus menentukan jumlah pekerja yang dialokasikan saat pekerjaan berjalan.

Bookmark Tugas

Tentukan cara AWS Glue memproses informasi status saat tugas berjalan. Anda dapat membuatnya mengingat data yang diproses sebelumnya, memperbarui informasi status, atau mengabaikan informasi status. Untuk informasi selengkapnya, lihat the section called “Melacak data yang diproses menggunakan bookmark pekerjaan”.

Eksekusi fleksibel

Saat mengonfigurasi pekerjaan menggunakan AWS Studio atau API, Anda dapat menentukan kelas eksekusi pekerjaan standar atau fleksibel. Pekerjaan Anda mungkin memiliki berbagai tingkat prioritas dan sensitivitas waktu. Kelas eksekusi standar sangat ideal untuk beban kerja yang sensitif terhadap waktu yang membutuhkan startup pekerjaan cepat dan sumber daya khusus.

Kelas eksekusi fleksibel cocok untuk pekerjaan yang tidak mendesak seperti pekerjaan pra-produksi, pengujian, dan pemuatan data satu kali. Jalankan pekerjaan yang fleksibel didukung untuk pekerjaan yang menggunakan AWS Glue versi 3.0 atau yang lebih baru dan `G.1X` atau jenis `G.2X` pekerja.

Flex job run ditagih berdasarkan jumlah pekerja yang berjalan kapan saja. Jumlah pekerja dapat ditambahkan atau dihapus untuk menjalankan pekerjaan yang fleksibel. Alih-alih menagih sebagai

perhitungan `Max Capacity` sederhana*`Execution Time`, setiap pekerja akan berkontribusi untuk waktu yang dijalankan selama pekerjaan dijalankan. Tagihan adalah jumlah dari (`Number of DPUs per worker*time each worker ran`).

Untuk informasi selengkapnya, lihat panel bantuan di AWS Studio, atau Tugas dan Tugas berjalan.

Jumlah percobaan

Tentukan jumlah percobaan, dari 0 sampai 10, yang membuat AWS Glue secara otomatis me-restart tugas jika gagal. Tugas yang mencapai batas habis waktu tidak dimulai ulang.

Tugas habis waktu

Atur waktu eksekusi maksimal dalam satuan menit. Default-nya adalah 2880 menit (48 jam) untuk tugas batch. Ketika waktu eksekusi tugas melebihi batas ini, maka status eksekusi tugas berubah menjadi `TIMEOUT`.

Pekerjaan streaming harus memiliki nilai batas waktu kurang dari 7 hari atau 10080 menit. Ketika nilai dibiarkan kosong, pekerjaan akan dimulai ulang setelah 7 hari berdasarkan jika Anda belum menyiapkan jendela pemeliharaan. Jika Anda telah menyiapkan jendela pemeliharaan, itu akan dimulai ulang selama jendela pemeliharaan setelah 7 hari.

> **Praktik terbaik untuk batas waktu kerja**
>
> Pekerjaan ditagih berdasarkan waktu eksekusi. Untuk menghindari tagihan yang tidak terduga, konfigurasikan nilai batas waktu yang sesuai untuk waktu eksekusi yang diharapkan dari pekerjaan Anda.

Properti Lanjutan

Nama berkas skrip

Nama skrip unik untuk pekerjaan Anda. Tidak dapat diberi nama Pekerjaan Tanpa Judul.

Jalur skrip

Lokasi skrip Amazon S3. Jalannya harus dalam bentuk `s3://bucket/prefix/path/`. Itu harus diakhiri dengan garis miring (/) dan tidak menyertakan file apa pun.

Metrik Tugas

Aktifkan atau matikan pembuatan CloudWatch metrik Amazon saat pekerjaan ini berjalan. Untuk melihat data pemrofilan, Anda harus mengaktifkan opsi ini. Untuk informasi

selengkapnya tentang cara mengaktifkan dan memvisualisasikan metrik, lihat Pemantauan dan debugging Job.

Metrik observabilitas pekerjaan

Aktifkan pembuatan CloudWatch metrik observabilitas tambahan saat pekerjaan ini berjalan. Untuk informasi selengkapnya, lihat the section called “Pemantauan dengan metrik AWS Glue Observabilitas”.

Pencatatan log berkelanjutan

Aktifkan pencatatan berkelanjutan ke Amazon CloudWatch. Jika opsi ini tidak diaktifkan, maka log hanya tersedia setelah tugas selesai saja. Untuk informasi selengkapnya, lihat the section called “Pencatatan terus menerus untuk AWS Glue pekerjaan”.

Spark UI

Hidupkan penggunaan Spark UI untuk memantau tugas ini. Untuk informasi selengkapnya, lihat Mengaktifkan UI web Apache Spark untuk pekerjaan AWS Glue.

Jalur log UI Spark

Jalur untuk menulis log saat Spark UI diaktifkan.

Konfigurasi logging dan pemantauan Spark UI

Pilih salah satu opsi berikut:

- Standar: tulis log menggunakan ID AWS Glue job run sebagai nama file. Aktifkan pemantauan UI Spark di AWS Glue konsol.
- Legacy: tulis log menggunakan 'spark-application- {timestamp} 'sebagai nama file. Jangan nyalakan pemantauan UI Spark.
- Standar dan warisan: tulis log ke lokasi standar dan lama. Aktifkan pemantauan UI Spark di AWS Glue konsol.

Konkurensi maksimum

Mengatur jumlah maksimal eksekusi bersamaan yang diperbolehkan untuk tugas ini. Default-nya adalah 1. Kesalahan dikembalikan ketika ambang batas ini tercapai. Nilai maksimal yang dapat Anda tentukan dikendalikan oleh kuota layanan. Sebagai contoh, jika eksekusi tugas sebelumnya masih berjalan ketika sebuah instans baru dimulai, maka Anda mungkin ingin mengembalikan kesalahan untuk mencegah dua instans dari tugas yang sama agar tidak berjalan secara bersamaan.

Jalur sementara

Menyediakan lokasi sebuah direktori kerja di Amazon S3 di mana hasil menengah sementara ditulis ketika AWS Glue menjalankan skrip. Konfirmasi bahwa tidak ada file dengan nama yang sama sebagai direktori sementara pada path. Direktori ini digunakan saat AWS Glue membaca dan menulis ke Amazon Redshift dan dengan transformasi AWS Glue tertentu.

> **Note**
>
> AWS Glue membuat sebuah bucket sementara untuk tugas jika sebuah bucket belum ada di suatu wilayah. Bucket ini mungkin mengizinkan akses publik. Anda dapat memodifikasi bucket di Amazon S3 untuk mengatur blok akses publik, atau menghapus bucket nanti setelah semua tugas di wilayah tersebut telah selesai.

Ambang batas pemberitahuan tunda (menit)

Menetapkan ambang batas (dalam menit) sebelum sebuah notifikasi penundaan dikirim. Anda dapat mengatur ambang batas ini untuk mengirim notifikasi ketika eksekusi tugas `RUNNING`, `STARTING`, atau `STOPPING` memerlukan waktu lebih dari jumlah menit yang diharapkan.

Konfigurasi keamanan

Pilih sebuah konfigurasi keamanan dari daftar. Sebuah konfigurasi keamanan menentukan bagaimana data pada target Amazon S3 dienkripsi: tidak ada enkripsi, enkripsi sisi server dengan kunci terkelola AWS KMS(SSE-KMS), atau kunci enkripsi yang dikelola Amazon S3 (SSE-S3).

enkripsi di sisi server

Jika Anda memilih opsi ini, maka ketika tugas ETL menulis ke Amazon S3, data dienkripsi secara at rest dengan menggunakan enkripsi SSE-S3. Baik target data Amazon S3 Anda dan data yang ditulis ke direktori sementara Amazon S3 , keduanya dienkripsi. Opsi ini diberikan sebagai parameter tugas. Untuk informasi selengkapnya, lihat Melindungi Data Menggunakan Enkripsi Sisi Server dengan Kunci Enkripsi Terkelola Amazon S3 (SSE-S3) di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

> **Important**
>
> Opsi ini diabaikan jika sebuah konfigurasi keamanan ditentukan.

Gunakan katalog data Glue sebagai metastore Hive

Pilih untuk menggunakan Katalog Data AWS Glue sebagai metastore Hive. IAM role yang digunakan untuk tugas harus memiliki izin `glue:CreateDatabase`. Sebuah basis data dengan nama “default” dibuat dalam Katalog Data jika tidak ada.

Koneksi

Pilih konfigurasi VPC untuk mengakses sumber data Amazon S3 yang terletak di cloud pribadi virtual (VPC) Anda. Anda dapat membuat dan mengelola koneksi Jaringan di AWS Glue. Untuk informasi selengkapnya, lihat Menghubungkan ke data.

Perpustakaan

Jalur pustaka Python, jalur JAR Dependen, dan jalur file yang direferensikan

Tentukan opsi ini jika skrip Anda memerlukannya. Anda dapat menentukan path Amazon S3 yang dipisahkan koma untuk pilihan ini ketika Anda menentukan tugas. Anda dapat mengganti path ini ketika Anda menjalankan tugas. Untuk informasi selengkapnya, lihat Menyediakan skrip kustom Anda sendiri.

Parameter Tugas

Satu set pasangan nilai-kunci yang diberikan sebagai parameter bernama untuk skrip. Ini adalah nilai default yang digunakan ketika skrip dijalankan, tetapi Anda dapat menimpanya di pemicu atau ketika Anda menjalankan tugas. Anda harus memberikan prefiks pada nama kunci dengan `--`; misalnya: `--myKey`. Anda meneruskan parameter pekerjaan sebagai peta saat menggunakan AWS Command Line Interface.

Sebagai contoh, lihat parameter Python di Melewati dan mengakses parameter Python di AWS Glue.

Tanda

Tandai tugas Anda dengan Kunci tag dan opsional Nilai tag. Setelah kunci tag dibuat, mereka hanya bisa dibaca. Gunakan tag ke sumber daya Anda untuk membantu mengatur dan mengidentifikasi sumber daya tersebut. Untuk informasi selengkapnya, lihat AWS tag di AWS Glue.

Pembatasan untuk pekerjaan yang mengakses tabel terkelola Lake Formation

Ingatlah catatan dan batasan berikut saat membuat pekerjaan yang membaca dari atau menulis ke tabel yang dikelola oleh AWS Lake Formation:

- Fitur berikut tidak didukung dalam pekerjaan yang mengakses tabel dengan filter tingkat sel:
  - Bookmark Job dan eksekusi terbatas
  - Predikat push-down
  - Predikat partisi katalog sisi server
  - enableUpdateCatalog

Mengedit skrip Spark di konsol AWS Glue

Skrip berisi kode yang mengekstrak data dari sumber, mengubahnya, dan memuatnya menjadi target. AWS Glue menjalankan skrip ketika memulai pekerjaan.

Skrip ETL AWS Glue dapat dikodekan dengan Python atau Scala. Skrip Python menggunakan bahasa yang merupakan perpanjangan dari dialek PySpark Python untuk mengekstrak, mengubah, dan memuat (ETL) pekerjaan. Skrip berisi konstruksi diperpanjang untuk menangani transformasi ETL. Ketika Anda secara otomatis membuat logika kode sumber untuk tugas Anda, sebuah skrip dibuat. Anda dapat mengedit skrip ini, atau Anda dapat memberikan skrip Anda sendiri untuk memproses tugas ETL Anda.

Untuk informasi tentang mendefinisikan dan mengedit skripAWS Glue, lihat. AWS Glue panduan pemrograman

Pustaka atau file tambahan

Jika skrip Anda memerlukan perpustakaan atau file tambahan, Anda dapat menentukannya sebagai berikut:

Path perpustakaan Python

Path Amazon Simple Storage Service (Amazon S3) yang dipisahkan koma ke perpustakaan Python yang diperlukan oleh skrip.

> **Note**
>
> Hanya perpustakaan Python murni yang dapat digunakan. Perpustakaan yang mengandalkan ekstensi C, seperti Perpustakaan Analisis Data Python, yakni pandas, saat ini tidak didukung.

Path jar dependen

Path Amazon S3 yang dipisahkan koma untuk file JAR yang diperlukan oleh skrip.

> **Note**
>
> Saat ini, hanya perpustakaan murni Java atau Scala (2.11) yang dapat digunakan.

Path file yang direferensikan

Path Amazon S3 yang dipisahkan koma untuk file tambahan (misalnya, file konfigurasi) yang diperlukan oleh skrip.

Lowongan Kerja (legacy)

Sebuah skrip berisi kode yang melakukan tugas extract, transform, and load (ETL). Anda dapat memberikan skrip Anda sendiri, atau AWS Glue dapat membuat skrip dengan bimbingan dari Anda. Untuk informasi selengkapnya mengenai cara membuat skrip Anda sendiri, lihat Menyediakan skrip kustom Anda sendiri.

Anda dapat mengedit skrip di konsol AWS Glue. Saat mengedit sebuah skrip, Anda dapat menambahkan sumber, target, dan transformasi.

Untuk mengedit sebuah skrip

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/. Lalu pilih tab Tugas.
2. Pilih tugas dalam daftar, lalu pilih Tindakan, Edit skrip untuk membuka editor skrip.

   Anda juga dapat mengakses editor skrip dari halaman detail tugas. Pilih tab Skrip, lalu pilih Edit skrip.

Editor skrip

Editor skrip AWS Glue memungkinkan Anda memasukkan, memodifikasi, dan menghapus sumber, target, dan transformasi dalam skrip Anda. Editor skrip menampilkan skrip dan diagram untuk membantu Anda memvisualisasikan aliran data.

Untuk membuat diagram untuk skrip tersebut, pilih Buat diagram. AWS Glue menggunakan baris anotasi dalam skrip yang diawali dengan ## untuk membuat diagram. Untuk merepresentasikan skrip Anda dengan benar dalam diagram, Anda harus menyimpan parameter dalam anotasi dan parameter dalam kode Apache Spark secara sinkron.

Editor skrip memungkinkan Anda menambahkan templat kode di mana pun kursor Anda diposisikan dalam skrip. Di bagian atas editor, pilih dari opsi berikut:

- Untuk menambahkan tabel sumber ke skrip, pilih Sumber.
- Untuk menambahkan tabel target ke skrip, pilih Target.
- Untuk menambahkan lokasi target ke skrip, pilih Lokasi target.
- Untuk menambahkan transformasi ke skrip, pilih Transformasi. Untuk informasi tentang fungsi yang dipanggil dalam skrip Anda, lihat Program skrip AWS Glue ETL di PySpark.
- Untuk menambahkan transformasi Spigot ke skrip, pilih Spigot.

Dalam kode yang dimasukkan, ubah `parameters` yang ada di anotasi dan kode Apache Spark. Sebagai contoh, jika Anda menambahkan transformasi Spigot, maka Anda harus memverifikasi bahwa `path` sudah diganti, baik di baris anotasi `@args` dan baris kode `output`.

Tab Log menunjukkan log yang dikaitkan dengan tugas Anda saat ia berjalan. 1.000 baris terbaru ditampilkan.

Tab Skema menunjukkan skema sumber dan target yang dipilih, bila ada dalam Katalog Data.

Melacak data yang diproses menggunakan bookmark pekerjaan

AWS Glue melacak data yang telah diproses selama eksekusi tugas ETL sebelumnya dengan mempertahankan informasi status dari eksekusi tugas. Informasi status yang dipertahankan ini disebut bookmark tugas. Bookmark tugas membantu AWS Glue mempertahankan informasi status dan mencegah pengolahan ulang data lama. Dengan bookmark tugas, Anda dapat memproses data baru saat menjalankan ulang berdasarkan pada interval yang dijadwalkan. Sebuah bookmark tugas terdiri dari status untuk berbagai elemen tugas, seperti sumber, transformasi, dan target. Misalnya, tugas ETL Anda mungkin membaca partisi baru dalam file Amazon S3. AWS Glue melacak partisi mana yang telah berhasil diproses oleh tugas untuk mencegah pemrosesan duplikat dan data duplikat di penyimpanan data target tugas.

Bookmark tugas diimplementasikan untuk sumber data JDBC, transformasi Relationalize, dan beberapa sumber Amazon Simple Storage Service (Amazon S3). Tabel berikut mencantumkan format sumber Amazon S3 yang didukung AWS Glue untuk bookmark tugas.

Versi AWS Glue	Format sumber Amazon S3
Versi 0.9	JSON, CSV, Apache Avro, XML
Versi 1.0 dan yang lebih baru	JSON, CSV, Apache Avro, XML, Parquet, ORC

Untuk informasi tentang versi AWS Glue, lihat Mendefinisikan properti pekerjaan untuk pekerjaan Spark.

Fitur bookmark pekerjaan memiliki fungsi tambahan saat diakses melalui AWS Glue skrip. Saat menelusuri skrip yang Anda buat, Anda mungkin melihat konteks transformasi, yang terkait dengan fitur ini. Untuk informasi selengkapnya, lihat the section called “Menggunakan bookmark pekerjaan”.

Topik

- Menggunakan bookmark pekerjaan di AWS Glue
- Rincian operasional fitur bookmark pekerjaan

Menggunakan bookmark pekerjaan di AWS Glue

Opsi bookmark tugas diberikan sebagai parameter ketika tugas dimulai. Tabel berikut menjelaskan opsi untuk pengaturan bookmark tugas pada konsol AWS Glue.

Bookmark tugas	Deskripsi
Aktifkan	Menyebabkan tugas memperbarui status setelah eksekusi untuk melacak data yang sebelumnya telah diproses. Jika tugas Anda memiliki sumber dengan support bookmark tugas, maka tugas akan melacak data yang diproses, dan ketika tugas berjalan, ia memproses data baru sejak pos pemeriksaan terakhir.
Nonaktifkan	Bookmark tugas tidak digunakan, dan tugas selalu memproses seluruh set data. Anda bertanggung jawab untuk mengelola output dari eksekusi tugas sebelumnya. Ini adalah default.

Bookmark tugas	Deskripsi
Jeda	Memproses data tambahan sejak eksekusi terakhir yang berhasil atau data dalam kisaran yang diidentifikasi oleh sub-opsi berikutnya, tanpa memperbarui status bookmark terakhir. Anda bertanggung jawab untuk mengelola output dari eksekusi tugas sebelumnya. Dua sub-opsi-nya adalah: • job-bookmark-from<from-value>adalah ID run yang mewakili semua input yang diproses hingga proses terakhir yang berhasil sebelumnya dan termasuk ID run yang ditentukan. Masukan yang sesuai diabaikan. • job-bookmark-to<to-value>adalah ID run yang mewakili semua input yang diproses hingga proses terakhir yang berhasil sebelumnya dan termasuk ID run yang ditentukan. Input yang sesuai tidak termasuk input yang diidentifikasi oleh <from-value> diproses oleh tugas. Setiap masukan berikutnya nanti selain dari input ini juga dikecualikan untuk diproses. Status bookmark tugas tidak diperbarui ketika rangkaian opsi ini ditentukan. Sub-opsi bersifat opsional, namun ketika digunakan, kedua sub-opsi tersebut perlu disediakan.

Untuk detail tentang parameter yang diberikan ke sebuah tugas pada baris perintah, dan secara khusus untuk bookmark tugas, lihat AWS Glueparameter pekerjaan.

Untuk sumber input Amazon S3, bookmark tugas AWS Glue memeriksa waktu modifikasi terakhir atas objek untuk memverifikasi objek mana yang perlu diproses ulang. Jika data sumber masukan Anda telah dimodifkasi sejak eksekusi tugas terakhir Anda, maka file akan diproses kembali ketika Anda menjalankan tugas itu lagi.

Untuk sumber JDBC, aturan-aturan berikut berlaku:

- Untuk setiap tabel, AWS Glue menggunakan satu atau beberapa kolom sebagai kunci bookmark untuk menentukan data baru dan data yang telah diproses. Kunci bookmark bergabung untuk membentuk satu kunci gabungan tunggal.

- AWS Gluesecara default menggunakan kunci primer sebagai kunci bookmark, asalkan secara berurutan meningkat atau menurun (tanpa celah).
- Anda dapat menentukan kolom yang akan digunakan sebagai tombol bookmark dalam AWS Glue skrip Anda. Untuk informasi selengkapnya tentang menggunakan bookmark Job dalam AWS Glue skrip, lihat. the section called “Menggunakan bookmark pekerjaan”
- AWS Gluetidak mendukung penggunaan kolom dengan nama peka huruf besar/kecil sebagai kunci bookmark pekerjaan.

Anda dapat memundurkan bookmark tugas Anda untuk tugas ETL Spark AWS Glue Anda untuk setiap eksekusi tugas sebelumnya. Anda dapat men-support skenario pengumpulan data historis dengan lebih baik dengan memundurkan bookmark tugas Anda ke setiap eksekusi tugas sebelumnya, sehingga eksekusi tugas berikutnya hanya mengolah ulang data dari eksekusi tugas yang diberi bookmark.

Jika Anda berniat untuk memproses ulang semua data menggunakan tugas yang sama, setel ulang bookmark tugas. Untuk menyetel ulang status bookmark tugas, gunakan konsol AWS Glue, operasi API ResetJobBookmark tindakan (Python: reset_job_bookmark), atau AWS CLI. Sebagai contoh, masukkan perintah berikut menggunakan AWS CLI:

```
aws glue reset-job-bookmark --job-name my-job-name
```

Saat Anda memundurkan atau mengatur ulang bookmark, AWS Glue tidak membersihkan file target karena mungkin ada beberapa target dan target tidak dilacak dengan bookmark tugas. Hanya file sumber yang dilacak dengan bookmark tugas. Anda dapat membuat target output yang berbeda ketika memundurkan dan mengolah ulang file sumber untuk menghindari duplikat data dalam output Anda.

AWS Glue melacak bookmark tugas berdasarkan tugas. Jika Anda menghapus sebuah tugas, maka bookmark tugas akan dihapus.

Dalam beberapa kasus, Anda mungkin telah mengaktifkan bookmark tugas ETL AWS Glue tapi tugas ETL Anda mengolah ulang data yang sudah diproses dalam eksekusi sebelumnya. Untuk informasi selengkapnya tentang mengatasi penyebab umum kesalahan ini, lihat Memecahkan masalah kesalahan untuk Spark AWS Glue.

Rincian operasional fitur bookmark pekerjaan

Bagian ini menjelaskan lebih lanjut tentang detail operasional penggunaan bookmark tugas.

Bookmark tugas menyimpan status untuk sebuah tugas. Setiap instans dari status tersebut di beri kunci berdasarkan nama tugas dan nomor versi. Ketika sebuah skrip memanggil `job.init`, maka ia mengambil status dan selalu mendapatkan versi terbaru. Dalam sebuah status, ada beberapa elemen status, yang spesifik untuk setiap sumber, transformasi, dan instans sink dalam skrip tersebut. Unsur-unsur status diidentifikasi oleh konteks transformasi yang dilampirkan pada elemen yang sesuai (sumber, transformasi, atau sink) dalam skrip. Unsur-unsur status disimpan secara atom saat `job.commit` dipanggil dari skrip pengguna. Skrip mendapat nama tugas dan pilihan kontrol untuk bookmark tugas dari argumen.

Unsur-unsur status dalam bookmark tugas adalah sumber, transformasi, atau data spesifik-sink. Sebagai contoh, misalkan Anda ingin membaca data tambahan dari lokasi Amazon S3 yang terus ditulis oleh tugas atau proses hulu. Dalam hal ini, skrip harus menentukan apa yang telah diproses sejauh ini. Implementasi bookmark tugas untuk sumber Amazon S3 menyimpan informasi sehingga ketika tugas berjalan lagi, ia dapat mem-filter hanya objek baru saja menggunakan informasi tersimpan dan melakukan komputasi ulang atas status untuk eksekusi tugas berikutnya. Sebuah stempel waktu digunakan untuk mem-filter file baru.

Selain elemen status, bookmark tugas memiliki nomor eksekusi, sebuah nomor percobaan, dan nomor versi. Nomor eksekusi melacak eksekusi tugas, dan nomor percobaan mencatat percobaan yang dilakukan untuk eksekusi tugas. Nomor eksekusi tugas adalah nomor yang meningkat secara monoton yang bertambah untuk setiap eksekusi yang berhasil. Nomor percobaan melacak upaya untuk setiap eksekusi, dan hanya bertambah ketika ada eksekusi setelah ada percobaan yang gagal. Nomor versi meningkat secara monoton dan melacak pembaruan ke bookmark tugas.

Dalam database AWS Glue layanan, status bookmark untuk semua transformasi disimpan bersama sebagai pasangan nilai kunci:

```
{
  "job_name" : ...,
  "run_id": ...,
  "run_number": ..,
  "attempt_number": ...
  "states": {
    "transformation_ctx1" : {
      bookmark_state1
    },
```

```
    "transformation_ctx2" : {
      bookmark_state2
    }
  }
}
```

Praktik terbaik

Berikut ini adalah praktik terbaik untuk menggunakan bookmark pekerjaan.

- Jangan mengubah properti sumber data dengan bookmark diaktifkan. Misalnya, ada sumber data0 yang menunjuk ke jalur input Amazon S3 A, dan pekerjaan telah membaca dari sumber yang telah berjalan selama beberapa putaran dengan bookmark diaktifkan. Jika Anda mengubah jalur input datasource0 ke Amazon S3 path B tanpa mengubah`transformation_ctx`, AWS Glue pekerjaan akan menggunakan status bookmark lama yang disimpan. Itu akan mengakibatkan file hilang atau melewatkan di jalur input B seperti yang AWS Glue akan mengasumsikan bahwa file-file tersebut telah diproses dalam proses sebelumnya.
- Gunakan tabel katalog dengan bookmark untuk manajemen partisi yang lebih baik. Bookmark berfungsi baik untuk sumber data dari Katalog Data atau dari opsi. Namun, sulit untuk menghapus/menambahkan partisi baru dengan pendekatan from options. Menggunakan tabel katalog dengan crawler dapat memberikan otomatisasi yang lebih baik untuk melacak partisi yang baru ditambahkan dan memberi Anda fleksibilitas untuk memilih partisi tertentu dengan predikat pushdown.
- Gunakan lister file AWS Glue Amazon S3 untuk kumpulan data besar. Bookmark akan mencantumkan semua file di bawah setiap partisi input dan melakukan filering, jadi jika ada terlalu banyak file di bawah satu partisi bookmark dapat berjalan ke driver OOM. Gunakan lister file AWS Glue Amazon S3 untuk menghindari daftar semua file dalam memori sekaligus.

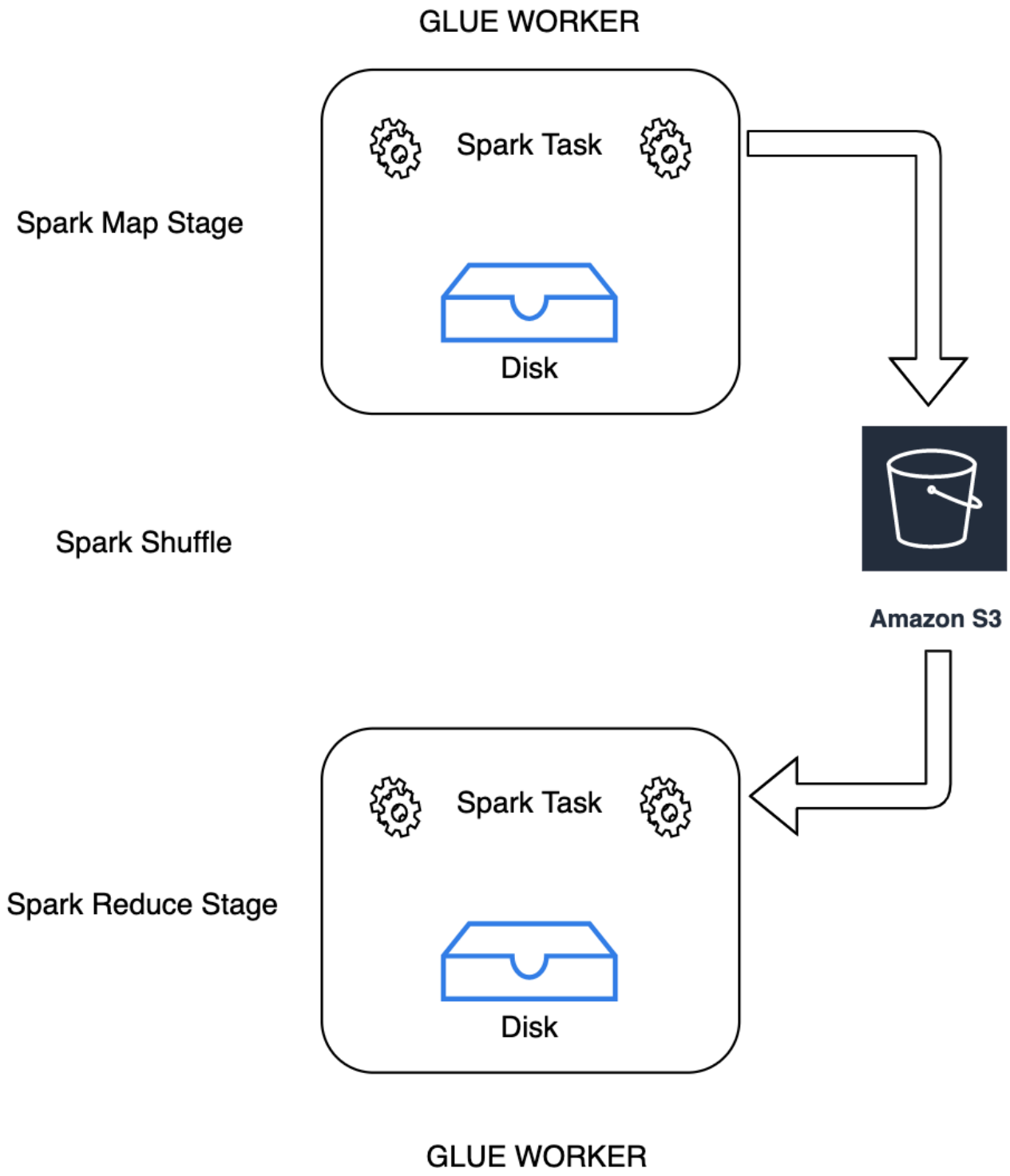
AWS GluePlugin Spark shuffle dengan Amazon S3

Shuffling adalah langkah penting dalam pekerjaan Spark setiap kali data disusun ulang di antara partisi. Hal ini diperlukan karena transformasi yang luas seperti`join`,, `groupByKey``reduceByKey`, dan `repartition` memerlukan informasi dari partisi lain untuk menyelesaikan pemrosesan. Spark mengumpulkan data yang diperlukan dari setiap partisi dan menggabungkannya menjadi partisi baru. Selama shuffle, data ditulis ke disk dan ditransfer ke seluruh jaringan. Akibatnya, operasi shuffle terikat pada kapasitas disk lokal. Spark melempar `MetadataFetchFailedException` kesalahan `No space left on device` atau ketika tidak ada cukup ruang disk yang tersisa pada eksekutor dan tidak ada pemulihan.

> **Note**
>
> AWS GluePlugin Spark shuffle dengan Amazon S3 hanya didukung AWS Glue untuk pekerjaan ETL.

Solusi

DenganAWS Glue, Anda sekarang dapat menggunakan Amazon S3 untuk menyimpan data shuffle Spark. Amazon S3 adalah layanan penyimpanan objek yang menawarkan skalabilitas, ketersediaan data, keamanan, dan kinerja terdepan di industri. Solusi ini memisahkan komputasi dan penyimpanan untuk pekerjaan Spark Anda, dan memberikan elastisitas lengkap dan penyimpanan shuffle berbiaya rendah, memungkinkan Anda menjalankan beban kerja paling intensif acak dengan andal.



Kami memperkenalkan Plugin Penyimpanan Cloud Shuffle baru untuk Apache Spark untuk menggunakan Amazon S3. Anda dapat mengaktifkan pengocokan Amazon S3 untuk menjalankan AWS Glue pekerjaan dengan andal tanpa kegagalan jika diketahui terikat oleh kapasitas disk lokal untuk operasi pengocokan besar. Dalam beberapa kasus, pengocokan ke Amazon S3 sedikit lebih

lambat daripada disk lokal (atau EBS) jika Anda memiliki sejumlah besar partisi kecil atau file acak yang ditulis ke Amazon S3.

Prasyarat untuk menggunakan Plugin Cloud Shuffle Storage

Untuk menggunakan Plugin Cloud Shuffle Storage dengan pekerjaan AWS Glue ETL, Anda memerlukan yang berikut ini:

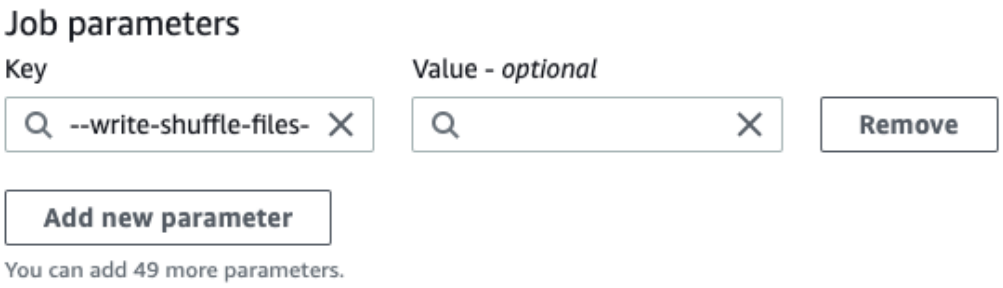
- Bucket Amazon S3 yang terletak di wilayah yang sama dengan pekerjaan Anda, untuk menyimpan data acak dan tumpah perantara. Awalan penyimpanan shuffle Amazon S3 dapat ditentukan dengan`--conf spark.shuffle.glue.s3ShuffleBucket=s3://`*`shuffle-bucket`*`/`*`prefix`*`/`, seperti pada contoh berikut:

```
--conf spark.shuffle.glue.s3ShuffleBucket=s3://glue-shuffle-123456789-us-east-1/glue-shuffle-data/
```

- Tetapkan kebijakan siklus hidup penyimpanan Amazon S3 pada awalan (sepertiglue-shuffle-data) karena pengelola acak tidak membersihkan file setelah pekerjaan selesai. Data shuffle dan tumpah perantara harus dihapus setelah pekerjaan selesai. Pengguna dapat menetapkan kebijakan siklus hidup singkat pada awalan. Petunjuk untuk menyiapkan kebijakan siklus hidup Amazon S3 tersedia di Menyetel konfigurasi siklus hidup pada bucket di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Menggunakan AWS Glue Spark shuffle manager dari konsol AWS

Untuk menyiapkan pengelola acak AWS Glue Spark menggunakan AWS Glue konsol atau AWS Glue Studio saat mengonfigurasi pekerjaan: pilih parameter pekerjaan -- write-shuffle-files-to -s3 untuk mengaktifkan pengocokan Amazon S3 untuk pekerjaan tersebut.



Menggunakan plugin AWS Glue Spark shuffle

Parameter pekerjaan berikut menyala dan menyetel manajer AWS Glue acak. Parameter ini adalah bendera, jadi nilai apa pun yang diberikan tidak dipertimbangkan.

- `--write-shuffle-files-to-s3`— Bendera utama, yang memungkinkan manajer shuffle AWS Glue Spark menggunakan bucket Amazon S3 untuk menulis dan membaca data shuffle. Ketika bendera tidak ditentukan, manajer acak tidak digunakan.
- `--write-shuffle-spills-to-s3`— (Didukung hanya pada AWS Glue versi 2.0). Bendera opsional yang memungkinkan Anda membongkar file tumpahan ke bucket Amazon S3, yang memberikan ketahanan tambahan pada pekerjaan Spark Anda. Ini hanya diperlukan untuk beban kerja besar yang menumpahkan banyak data ke disk. Ketika bendera tidak ditentukan, tidak ada file tumpahan perantara yang ditulis.
- `--conf spark.shuffle.glue.s3ShuffleBucket=s3://<shuffle-bucket>`— Bendera opsional lain yang menentukan bucket Amazon S3 tempat Anda menulis file shuffle. Secara default, `--TempDir` /shuffle-data. AWS Glue3.0+ mendukung penulisan file shuffle ke beberapa bucket dengan menentukan bucket dengan pembatas koma, seperti pada. `--conf spark.shuffle.glue.s3ShuffleBucket=s3://`*`shuffle-bucket-1`*`/`*`prefix`*`,s3://`*`shuffle-bucket-2`*`/`*`prefix`*`/` Menggunakan beberapa ember meningkatkan kinerja.

Anda perlu menyediakan pengaturan konfigurasi keamanan untuk mengaktifkan enkripsi saat istirahat untuk data acak. Untuk informasi selengkapnya tentang konfigurasi keamanan, lihatthe section called “Menyiapkan enkripsi”. AWS Gluemendukung semua konfigurasi terkait shuffle lainnya yang disediakan oleh Spark.

Binari perangkat lunak untuk plugin Cloud Shuffle Storage

Anda juga dapat mengunduh binari perangkat lunak Cloud Shuffle Storage Plugin untuk Apache Spark di bawah lisensi Apache 2.0 dan menjalankannya di lingkungan Spark apa pun. Plugin baru ini dilengkapi dengan dukungan out-of-the kotak untuk Amazon S3, dan juga dapat dengan mudah dikonfigurasi untuk menggunakan bentuk penyimpanan cloud lainnya seperti Google Cloud Storage dan Microsoft Azure Blob Storage. Untuk informasi selengkapnya, lihat Plugin Cloud Shuffle Storage untuk Apache Spark.

Catatan dan batasan

Berikut ini adalah catatan atau batasan untuk manajer AWS Glue acak:

- AWS Gluepengelola acak tidak secara otomatis menghapus file data acak (sementara) yang disimpan di bucket Amazon S3 Anda setelah pekerjaan selesai. Untuk memastikan perlindungan data, ikuti petunjuk Prasyarat untuk menggunakan Plugin Cloud Shuffle Storage sebelum mengaktifkan Plugin Cloud Shuffle Storage.
- Anda dapat menggunakan fitur ini jika data Anda miring.

Plugin Penyimpanan Cloud Shuffle untuk Apache Spark

Plugin Cloud Shuffle Storage adalah plugin Apache Spark yang kompatibel dengan `ShuffleDataIOAPI` yang memungkinkan penyimpanan data shuffle pada sistem penyimpanan cloud (seperti Amazon S3). Ini membantu Anda untuk menambah atau mengganti kapasitas penyimpanan disk lokal untuk operasi shuffle besar, biasanya dipicu oleh transformasi seperti`join`,, `groupByKey` dan `repartition` dalam aplikasi Spark And`areduceByKey`, sehingga mengurangi kegagalan umum atau dislokasi harga/kinerja pekerjaan analitik data tanpa server Anda dan pipeline.

AWS Glue

AWS Glueversi 3.0 dan 4.0 dilengkapi dengan plugin yang sudah diinstal sebelumnya dan siap untuk mengaktifkan pengocokan ke Amazon S3 tanpa langkah tambahan. Untuk informasi selengkapnya, lihat Plugin AWS Glue Spark shuffle dengan Amazon S3 untuk mengaktifkan fitur untuk aplikasi Spark Anda.

Lingkungan Spark lainnya

Plugin ini memerlukan konfigurasi Spark berikut untuk disetel di lingkungan Spark lainnya:

- `--conf spark.shuffle.sort.io.plugin.class=com.amazonaws.spark.shuffle.io.cloud.Choppe`
  Ini menginformasikan Spark untuk menggunakan plugin ini untuk Shuffle IO.
- `--conf spark.shuffle.storage.path=s3://`*`bucket-name`*`/`*`shuffle-file-dir`*: Jalur tempat file shuffle Anda akan disimpan.

> **Note**
>
> Plugin menimpa satu kelas inti Spark. Akibatnya, toples plugin perlu dimuat sebelum stoples Spark. Anda dapat melakukan ini menggunakan `userClassPathFirst` di lingkungan YARN on-prem jika plugin digunakan di luar. AWS Glue

Bundling plugin dengan aplikasi Spark Anda

Anda dapat menggabungkan plugin dengan aplikasi Spark dan distribusi Spark Anda (versi 3.1 ke atas) dengan menambahkan ketergantungan plugin di Maven Anda `pom.xml` sambil mengembangkan aplikasi Spark Anda secara lokal. Untuk informasi lebih lanjut tentang plugin dan versi Spark, lihat Versi plugin.

```
<repositories>
   ...
    <repository>
        <id>aws-glue-etl-artifacts</id>
        <url>https://aws-glue-etl-artifacts.s3.amazonaws.com/release/ </url>
    </repository>
</repositories>
...
<dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>chopper-plugin</artifactId>
    <version>3.1-amzn-LATEST</version>
</dependency>
```

Anda dapat mengunduh binari dari artefak AWS Glue Maven secara langsung dan memasukkannya ke dalam aplikasi Spark Anda sebagai berikut.

```
#!/bin/bash
sudo wget -v https://aws-glue-etl-artifacts.s3.amazonaws.com/release/com/amazonaws/
chopper-plugin/3.1-amzn-LATEST/chopper-plugin-3.1-amzn-LATEST.jar -P /usr/lib/spark/
jars/
```

Contoh spark-submit

```
spark-submit --deploy-mode cluster \
--conf spark.shuffle.storage.s3.path=s3://<ShuffleBucket>/<shuffle-dir> \
--conf spark.driver.extraClassPath=<Path to plugin jar> \
--conf spark.executor.extraClassPath=<Path to plugin jar> \
--class <your test class name> s3://<ShuffleBucket>/<Your application jar> \
```

Konfigurasi opsional

Ini adalah nilai konfigurasi opsional yang mengontrol perilaku shuffle Amazon S3.

- `spark.shuffle.storage.s3.enableServerSideEncryption`: Aktifkan/nonaktifkan S3 SSE untuk mengacak dan menumpahkan file. Nilai defaultnya adalah `true`.
- `spark.shuffle.storage.s3.serverSideEncryption.algorithm`: Algoritma SSE yang akan digunakan. Nilai defaultnya adalah `AES256`.
- `spark.shuffle.storage.s3.serverSideEncryption.kms.key`: ARN kunci KMS saat SSE aws:kms diaktifkan.

Seiring dengan konfigurasi ini, Anda mungkin perlu mengatur konfigurasi seperti `spark.hadoop.fs.s3.enableServerSideEncryption` dan konfigurasi khusus lingkungan lainnya untuk memastikan enkripsi yang sesuai diterapkan untuk kasus penggunaan Anda.

Versi plugin

Plugin ini didukung untuk versi Spark yang terkait dengan setiap AWS Glue versi. Tabel berikut menunjukkan AWS Glue versi, versi Spark dan versi plugin terkait dengan lokasi Amazon S3 untuk biner perangkat lunak plugin.

Versi AWS Glue	Versi Spark	Versi plugin	Lokasi Amazon S3
3.0	3.1	3.1-AMZN-Terbaru	s3://aws-glue-etl-artifacts/rilis/com/amazonaws/chopper-plugin/3.1-amzn-0/chopper-plugin-3.1-amzn-terbaru.jar
4.0	3.3	3.3-AMZN-Terbaru	s3://aws-glue-etl-artifacts/rilis/com/amazonaws/chopper-plugin/3.3-AMZN-0/chopper-plugin-3.3-AMZN-terbaru.jar

Lisensi

Biner perangkat lunak untuk plugin ini dilisensikan di bawah Lisensi Apache-2.0.

Lowongan kerja Monitoring AWS Glue Spark

Topik

Metrik Spark tersedia di AWS Glue Studio

Tab Metrik menampilkan metrik yang dikumpulkan saat sebuah eksekusi tugas dan pembuatan profil diaktifkan. Grafik berikut ditunjukkan dalam pekerjaan Spark:

- Pergerakan Data ETL
- Profil Memori: Driver dan Pelaksana

Pilih Lihat metrik tambahan untuk menampilkan grafik berikut:

- Pergerakan Data ETL
- Profil Memori: Driver dan Pelaksana
- Data yang Diacak di Seluruh Pelaksana
- Beban CPU: Driver dan Pelaksana
- Eksekusi Tugas: Pelaksana Aktif, Tahapan Selesai & Pelaksana Maksimal yang Dibutuhkan

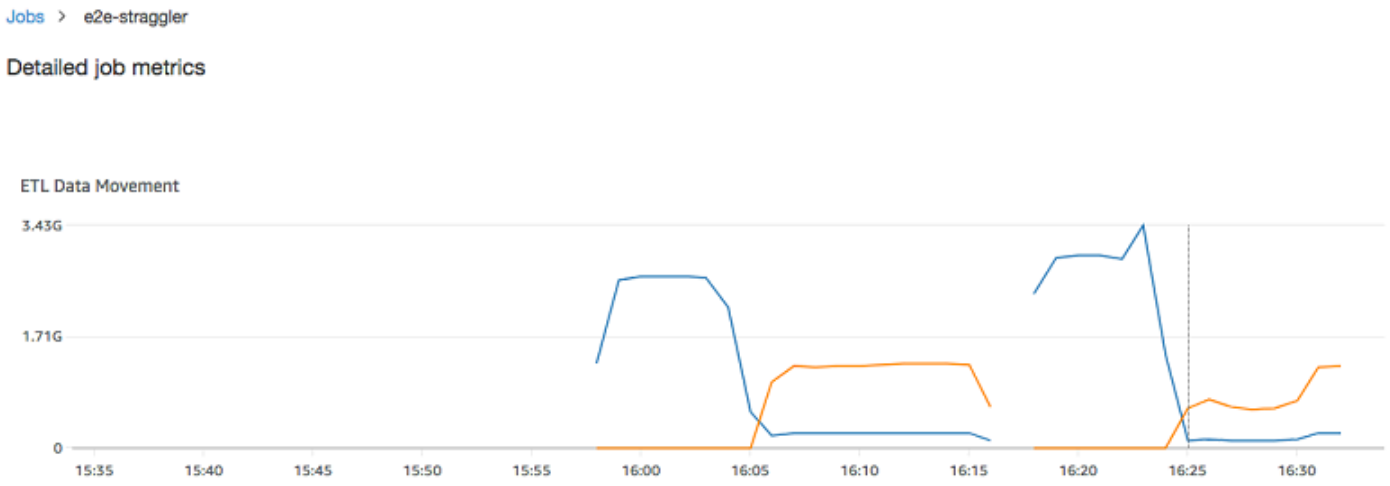
Data untuk grafik ini didorong ke CloudWatch metrik jika pekerjaan dikonfigurasi untuk mengumpulkan metrik. Untuk informasi selengkapnya tentang cara mengaktifkan metrik dan menafsirkan grafik, lihat Pemantauan dan debugging Job.

Example Grafik pergerakan data ETL

Grafik Pergerakan Data ETL menunjukkan metrik berikut:

- Jumlah byte yang dibaca dari Amazon S3 oleh semua pelaksana —`glue.ALL.s3.filesystem.read_bytes`

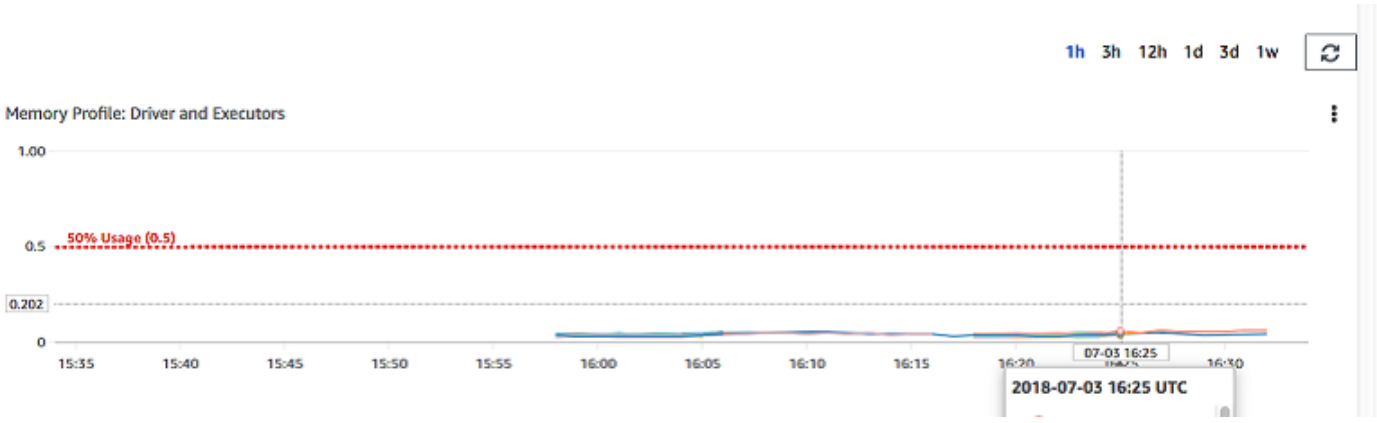
- Jumlah byte yang ditulis ke Amazon S3 oleh semua pelaksana —glue.ALL.s3.filesystem.write_bytes



Example Grafik profil memori

Grafik Profil Memori menampilkan metrik berikut ini:

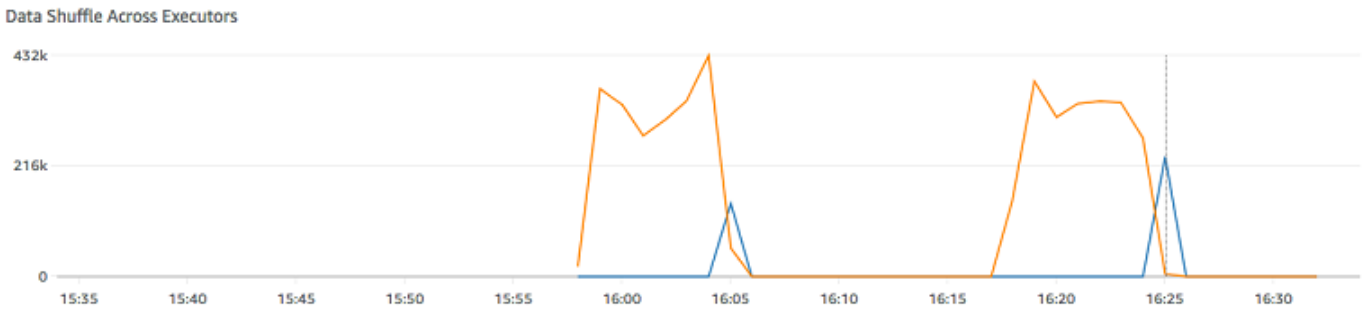
- Fraksi memori yang digunakan oleh tumpukan JVM untuk driver ini (skala: 0–1) oleh driver, yakni pelaksana diidentifikasi oleh executorId, atau semua pelaksana—
  - glue.driver.jvm.heap.usage
  - glue.*executorId*.jvm.heap.usage
  - glue.ALL.jvm.heap.usage



Example Pengocokan data di seluruh grafik pelaksana

Grafik Data yang Diacak di Seluruh Pelaksana menunjukkan metrik berikut:

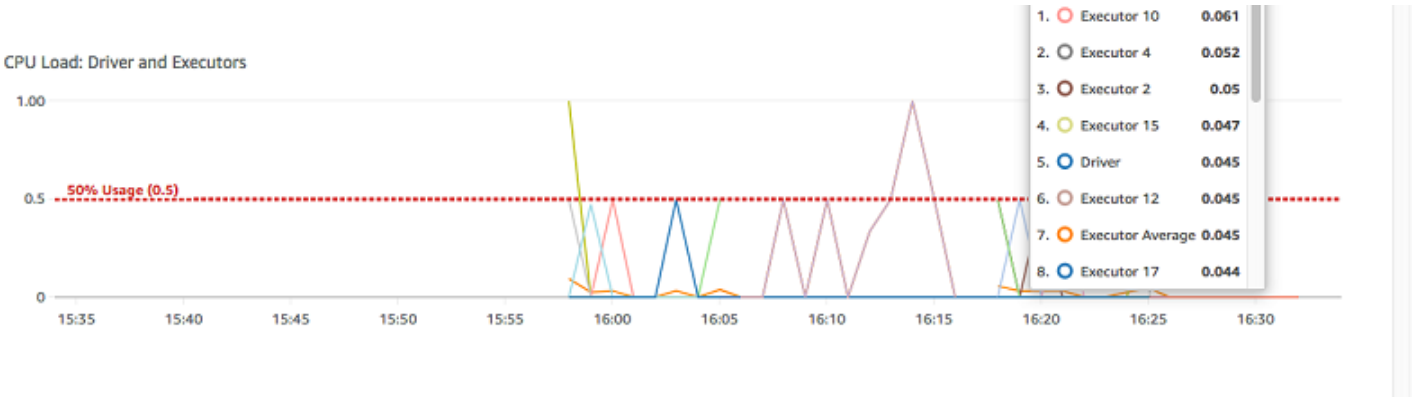
- Jumlah byte yang dibaca oleh semua pelaksana untuk mengacak data antara mereka —`glue.driver.aggregate.shuffleLocalBytesRead`
- Jumlah byte yang ditulis oleh semua pelaksana untuk mengacak data antara mereka —`glue.driver.aggregate.shuffleBytesWritten`



Example Grafik beban CPU

Grafik Beban CPU menunjukkan metrik berikut:

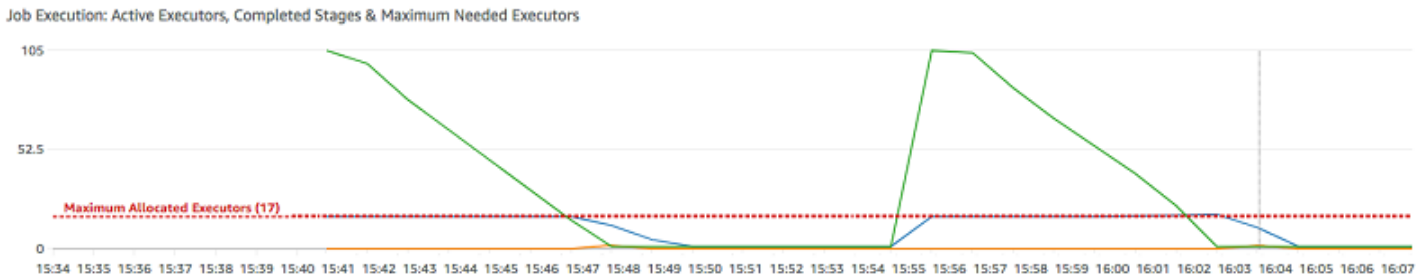
- Fraksi beban sistem CPU yang digunakan (skala: 0–1) oleh driver, yakni pelaksana yang diidentifikasi oleh executorId, atau semua pelaksana—
  - `glue.driver.system.cpuSystemLoad`
  - `glue.executorId.system.cpuSystemLoad`
  - `glue.ALL.system.cpuSystemLoad`



Example Grafik eksekusi Job

Grafik Eksekusi Tugas menampilkan metrik berikut:

- Jumlah pelaksana yang berjalan aktif —glue.driver.ExecutorAllocationManager.executors.numberAllExecutors
- Jumlah tahap yang telah selesai—glue.aggregate.numCompletedStages
- Jumlah maksimal pelaksana yang dibutuhkan —glue.driver.ExecutorAllocationManager.executors.numberMaxNeededExecutors



Memantau pekerjaan menggunakan UI web Apache Spark

Anda dapat menggunakan Apache Spark Web UI untuk memantau dan melakukan debug tugas ETL AWS Glue yang berjalan pada sistem tugas AWS Glue, dan juga aplikasi Spark yang berjalan di titik akhir pengembangan AWS Glue. Spark UI memungkinkan Anda untuk memeriksa hal berikut untuk setiap tugas:

- Lini waktu peristiwa dari setiap tahap Spark
- Grafik asiklik terarah (DAG) dari tugas tersebut
- Rencana fisik dan logis untuk kueri SparkSQL
- Variabel lingkungan Spark yang mendasari untuk setiap tugas

Untuk informasi selengkapnya tentang penggunaan UI Web Spark, lihat UI Web di dokumentasi Spark. Untuk panduan tentang cara menafsirkan hasil Spark UI guna meningkatkan kinerja pekerjaan Anda, lihat Praktik terbaik untuk penyetelan AWS Glue kinerja untuk pekerjaan Apache Spark di Panduan Preskriptif. AWS

Anda dapat melihat UI Spark di AWS Glue konsol. Ini tersedia ketika AWS Glue pekerjaan berjalan pada versi AWS Glue 3.0 atau yang lebih baru dengan log yang dihasilkan dalam format Standar

(bukan warisan), yang merupakan default untuk pekerjaan yang lebih baru. Jika Anda memiliki file log lebih besar dari 0,5 GB, Anda dapat mengaktifkan dukungan log bergulir untuk pekerjaan berjalan pada versi AWS Glue 4.0 atau yang lebih baru untuk menyederhanakan pengarsipan log, analisis, dan pemecahan masalah.

Anda dapat mengaktifkan UI Spark dengan menggunakan AWS Glue konsol atau AWS Command Line Interface (AWS CLI). Saat Anda mengaktifkan UI Spark, pekerjaan AWS Glue ETL, dan aplikasi Spark pada titik akhir AWS Glue pengembangan dapat mencadangkan log peristiwa Spark ke lokasi yang Anda tentukan di Amazon Simple Storage Service (Amazon S3). Anda dapat menggunakan log peristiwa yang dicadangkan di Amazon S3 dengan Spark UI, baik secara real time saat pekerjaan beroperasi dan setelah pekerjaan selesai. Meskipun log tetap ada di Amazon S3, UI Spark di AWS Glue konsol dapat melihatnya.

Izin

Untuk menggunakan UI Spark di AWS Glue konsol, Anda dapat menggunakan `UseGlueStudio` atau menambahkan semua API layanan individual. Semua API diperlukan untuk menggunakan UI Spark sepenuhnya, namun pengguna dapat mengakses fitur SparkUI dengan menambahkan API layanannya di izin IAM mereka untuk akses berbutir halus.

`RequestLogParsing`adalah yang paling penting karena melakukan penguraian log. API yang tersisa adalah untuk membaca data yang diuraikan masing-masing. Misalnya, `GetStages` menyediakan akses ke data tentang semua tahapan pekerjaan Spark.

Daftar API layanan Spark UI yang dipetakan `UseGlueStudio` ada di bawah ini dalam kebijakan sampel. Kebijakan di bawah ini menyediakan akses untuk hanya menggunakan fitur UI Spark. Untuk menambahkan lebih banyak izin seperti Amazon S3 dan IAM, lihat Membuat Kebijakan IAM Kustom untuk. AWS Glue Studio

Daftar API layanan Spark UI yang dipetakan `UseGlueStudio` ada di bawah ini dalam kebijakan sampel. Saat menggunakan API layanan Spark UI, gunakan namespace berikut:. `glue:<ServiceAPI>`

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowGlueStudioSparkUI",
   "Effect": "Allow",
            "Action": [
                "glue:RequestLogParsing",
```

```
                "glue:GetLogParsingStatus",
                "glue:GetEnvironment",
                "glue:GetJobs",
                "glue:GetJob",
                "glue:GetStage",
                "glue:GetStages",
                "glue:GetStageFiles",
                "glue:BatchGetStageFiles",
                "glue:GetStageAttempt",
                "glue:GetStageAttemptTaskList",
                "glue:GetStageAttemptTaskSummary",
                "glue:GetExecutors",
                "glue:GetExecutorsThreads",
                "glue:GetStorage",
                "glue:GetStorageUnit",
                "glue:GetQueries",
                "glue:GetQuery"
            ],
    "Resource": [
     "*"
    ]
   }
  ]
}
```

Batasan

- Spark UI di AWS Glue konsol tidak tersedia untuk menjalankan pekerjaan yang terjadi sebelum 20 November 2023 karena berada dalam format log lama.
- Spark UI di AWS Glue konsol mendukung rolling log untuk AWS Glue 4.0, seperti yang dihasilkan secara default dalam pekerjaan streaming. Jumlah maksimum dari semua file peristiwa log gulung yang dihasilkan adalah 2 GB. Untuk AWS Glue pekerjaan tanpa dukungan log yang digulung, ukuran file peristiwa log maksimum yang didukung untuk SparkUI adalah 0,5 GB.
- UI Spark Tanpa Server tidak tersedia untuk log peristiwa Spark yang disimpan di bucket Amazon S3 yang hanya dapat diakses oleh VPC Anda.

Contoh: Apache Spark web UI

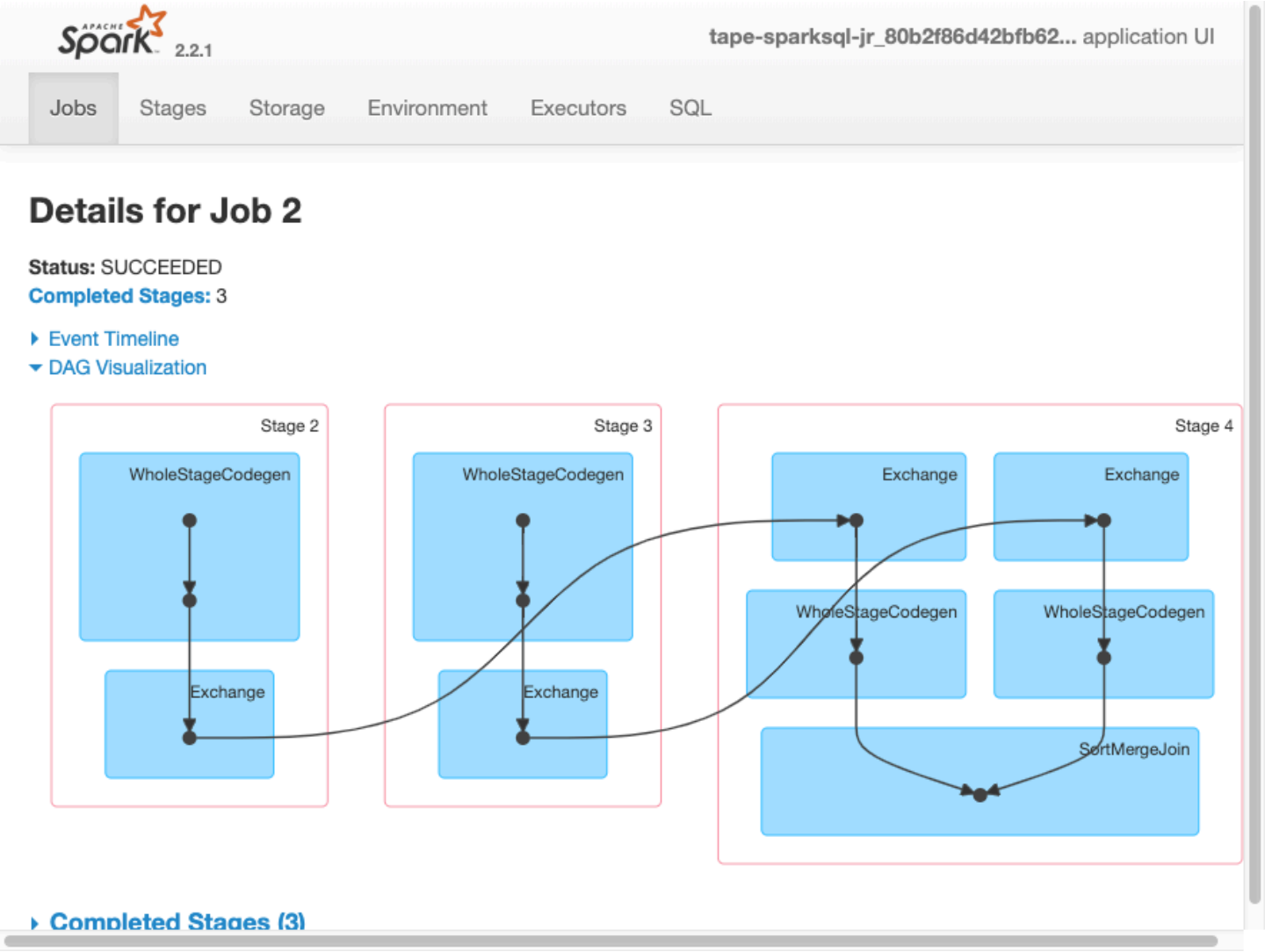
Contoh ini menunjukkan cara menggunakan UI Spark untuk memahami kinerja pekerjaan Anda. Tangkapan layar menunjukkan UI web Spark seperti yang disediakan oleh server riwayat Spark

yang dikelola sendiri. Spark UI di AWS Glue konsol memberikan tampilan serupa. Untuk informasi selengkapnya tentang penggunaan UI Web Spark, lihat UI Web di dokumentasi Spark.

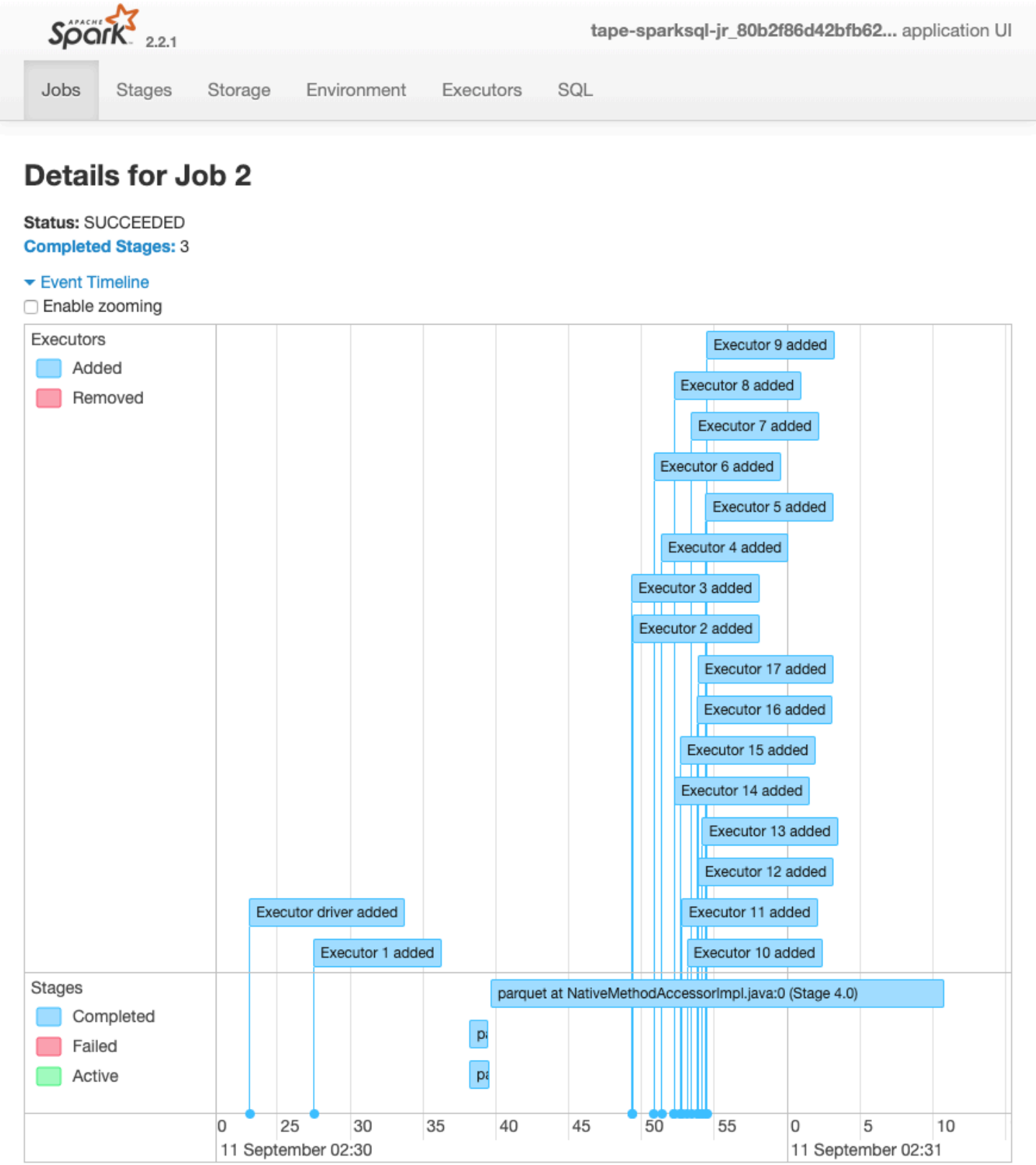
Berikut ini adalah contoh aplikasi Spark yang membaca dari dua sumber data, melakukan transformasi gabungan, dan menuliskannya ke Amazon S3 dalam format Parket.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from pyspark.sql.functions import count, when, expr, col, sum, isnull
from pyspark.sql.functions import countDistinct
from awsglue.dynamicframe import DynamicFrame

args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

job = Job(glueContext)
job.init(args['JOB_NAME'])

df_persons = spark.read.json("s3://awsglue-datasets/examples/us-legislators/all/
persons.json")
df_memberships = spark.read.json("s3://awsglue-datasets/examples/us-legislators/all/
memberships.json")

df_joined = df_persons.join(df_memberships, df_persons.id == df_memberships.person_id,
 'fullouter')
df_joined.write.parquet("s3://aws-glue-demo-sparkui/output/")

job.commit()
```

Visualisasi DAG berikut menunjukkan tahapan yang berbeda dalam tugas Spark ini.

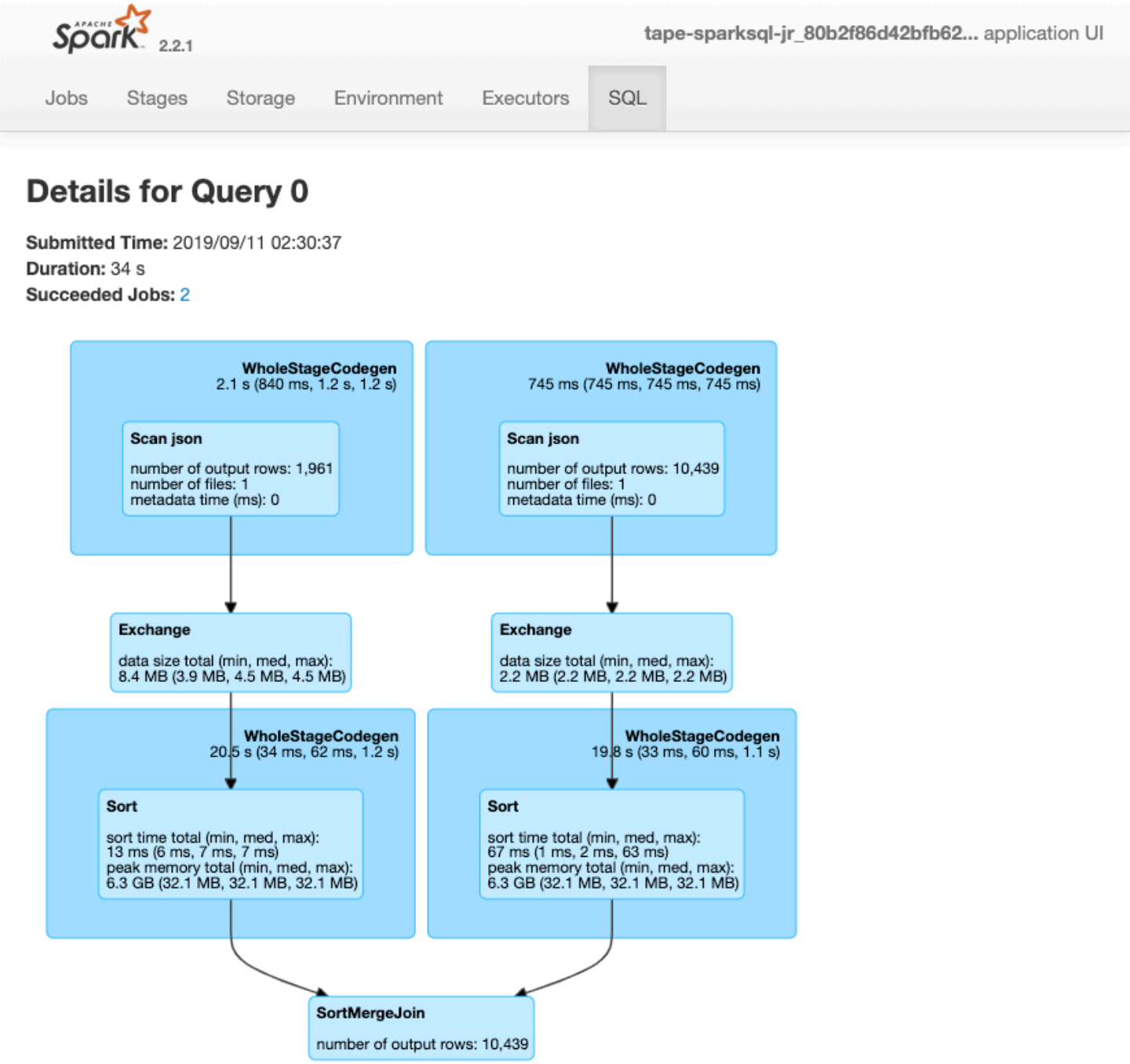


Lini waktu peristiwa tugas berikut menunjukkan awal, eksekusi, dan pengakhiran pelaksana Spark yang berbeda.


Spark 2.2.1
tape-sparksql-jr_80b2f86d42bfb62... application UI
Jobs
Stages
Storage
Environment
Executors
SQL
Details for Job 2
Status: SUCCEEDED
Completed Stages: 3
Event Timeline
Enable zooming
Executors
Added
Removed
Executor 9 added
Executor 8 added
Executor 7 added
Executor 6 added
Executor 5 added
Executor 4 added
Executor 3 added
Executor 2 added
Executor 17 added
Executor 16 added
Executor 15 added
Executor 14 added
Executor 13 added
Executor 12 added
Executor 11 added
Executor 10 added
Executor driver added
Executor 1 added
Stages
Completed
Failed
Active
parquet at NativeMethodAccessorImpl.java:0 (Stage 4.0)
0
25
30
35
40
45
50
55
0
5
10
11 September 02:30
11 September 02:31
DAG Visualization
Completed Stages (3)

Layar berikut menunjukkan detail dari rencana kueri SparkSQL:

- Rencana logis diurai
- Rencana logis yang dianalisis
- Rencana logis yang dioptimalkan
- Rencana fisik untuk eksekusi


Spark 2.2.1
tape-sparksql-jr_80b2f86d42bfb62... application UI
Jobs
Stages
Storage
Environment
Executors
SQL
Details for Query 0
Submitted Time: 2019/09/11 02:30:37
Duration: 34 s
Succeeded Jobs: 2
WholeStageCodegen
2.1 s (840 ms, 1.2 s, 1.2 s)
Scan json
number of output rows: 1,961
number of files: 1
metadata time (ms): 0
WholeStageCodegen
745 ms (745 ms, 745 ms, 745 ms)
Scan json
number of output rows: 10,439
number of files: 1
metadata time (ms): 0
Exchange
data size total (min, med, max):
8.4 MB (3.9 MB, 4.5 MB, 4.5 MB)
Exchange
data size total (min, med, max):
2.2 MB (2.2 MB, 2.2 MB, 2.2 MB)
WholeStageCodegen
20.5 s (34 ms, 62 ms, 1.2 s)
Sort
sort time total (min, med, max):
13 ms (6 ms, 7 ms, 7 ms)
peak memory total (min, med, max):
6.3 GB (32.1 MB, 32.1 MB, 32.1 MB)
WholeStageCodegen
19.8 s (33 ms, 60 ms, 1.1 s)
Sort
sort time total (min, med, max):
67 ms (1 ms, 2 ms, 63 ms)
peak memory total (min, med, max):
6.3 GB (32.1 MB, 32.1 MB, 32.1 MB)
SortMergeJoin
number of output rows: 10,439
Details
== Parsed Logical Plan ==
Join FullOuter, (id#14 = person_id#50)
:-
Relation[birth_date#8,contact_details#9,death_date#10,family_name#11,gender#12,given_name#13,id#14,identifiers#15
,image#16,images#17,links#18,name#19,other_names#20,sort_name#21] json
+-
Relation[area_id#45,end_date#46,legislative_period_id#47,on_behalf_of_id#48,organization_id#49,person_id#50,role#
51,start_date#52] json
== Analyzed Logical Plan ==
birth_date: string, contact_details: array<struct<type:string,value:string>>, death_date: string, family_name:
string, gender: string, given_name: string, id: string, identifiers:
array<struct<identifier:string,scheme:string>>, image: string, images: array<struct<url:string>>, links:
array<struct<note:string,url:string>>, name: string, other_names:
array<struct<lang:string,name:string,note:string>>, sort_name: string, area_id: string, end_date: string,
legislative_period_id: string, on_behalf_of_id: string, organization_id: string, person_id: string, role: string,
start_date: string
Join FullOuter, (id#14 = person_id#50)

Topik

- Mengaktifkan UI web Apache Spark untuk pekerjaan AWS Glue
- Meluncurkan server sejarah Spark

Mengaktifkan UI web Apache Spark untuk pekerjaan AWS Glue

Anda dapat menggunakan Apache Spark Web UI untuk memantau dan melakukan debug tugas ETL AWS Glue yang berjalan pada sistem tugas AWS Glue. Anda dapat mengkonfigurasi Spark UI dengan menggunakan konsol AWS Glue atau AWS Command Line Interface (AWS CLI).

Setiap 30 detik, AWS Glue buat cadangan log peristiwa Spark ke jalur Amazon S3 yang Anda tentukan.

Topik

- Mengkonfigurasi UI Spark (konsol)
- Mengkonfigurasi Spark UI (AWS CLI)
- Mengonfigurasi UI Spark untuk sesi menggunakan Notebook
- Aktifkan log bergulir

Mengkonfigurasi UI Spark (konsol)

Ikuti langkah-langkah ini untuk mengonfigurasi UI Spark dengan menggunakan. AWS Management Console Saat membuat AWS Glue pekerjaan, Spark UI diaktifkan secara default.

Untuk mengaktifkan UI Spark saat Anda membuat atau mengedit pekerjaan

1. Masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pilih Tugas.
3. Pilih Tambah pekerjaan, atau pilih yang sudah ada.
4. Di detail Job, buka properti Advanced.
5. Di bawah tab Spark UI, pilih Tulis log UI Spark ke Amazon S3.
6. Tentukan path Amazon S3 untuk menyimpan log peristiwa Spark untuk tugas itu. Perhatikan bahwa jika Anda menggunakan konfigurasi keamanan dalam pekerjaan, enkripsi juga berlaku untuk file log UI Spark. Untuk informasi selengkapnya, lihat Mengenkripsi data yang ditulis oleh AWS Glue.

7. Di bawah konfigurasi logging dan pemantauan Spark UI:

   - Pilih Standar jika Anda membuat log untuk dilihat di AWS Glue konsol.
   - Pilih Legacy jika Anda membuat log untuk dilihat di server riwayat Spark.
   - Anda juga dapat memilih untuk menghasilkan keduanya.

Mengkonfigurasi Spark UI (AWS CLI)

Untuk menghasilkan log untuk dilihat dengan Spark UI, di AWS Glue konsol, gunakan AWS CLI untuk meneruskan parameter pekerjaan berikut ke AWS Glue pekerjaan. Untuk informasi selengkapnya, lihat the section called “Parameter Tugas”.

```
'--enable-spark-ui': 'true',
'--spark-event-logs-path': 's3://s3-event-log-path'
```

Untuk mendistribusikan log ke lokasi lawasannya, setel `--enable-spark-ui-legacy-path` parameter ke`"true"`. Jika Anda tidak ingin menghasilkan log di kedua format, hapus `--enable-spark-ui` parameternya.

Mengonfigurasi UI Spark untuk sesi menggunakan Notebook

> **Warning**
>
> AWS Glue sesi interaktif saat ini tidak mendukung Spark UI di konsol. Konfigurasikan server riwayat Spark.

Jika Anda menggunakan AWS Glue buku catatan, siapkan konfigurasi SparkUI sebelum memulai sesi. Untuk melakukan ini, gunakan sihir `%%configure` sel:

```
%%configure { “--enable-spark-ui”: “true”, “--spark-event-logs-path”: “s3://path” }
```

Aktifkan log bergulir

Mengaktifkan SparkUI dan bergulir file peristiwa log untuk AWS Glue pekerjaan memberikan beberapa manfaat:

- File Acara Log Bergulir - Dengan file peristiwa log bergulir diaktifkan, AWS Glue menghasilkan file log terpisah untuk setiap langkah pelaksanaan pekerjaan, sehingga lebih mudah untuk mengidentifikasi dan memecahkan masalah khusus untuk tahap atau transformasi tertentu.
- Manajemen Log yang Lebih Baik - File acara log bergulir membantu mengelola file log dengan lebih efisien. Alih-alih memiliki satu file log yang berpotensi besar, log dibagi menjadi file yang lebih kecil dan lebih mudah dikelola berdasarkan tahap eksekusi pekerjaan. Ini dapat menyederhanakan pengarsipan log, analisis, dan pemecahan masalah.
- Peningkatan Toleransi Kesalahan - Jika AWS Glue pekerjaan gagal atau terganggu, file acara log bergulir dapat memberikan informasi berharga tentang tahap sukses terakhir, sehingga lebih mudah untuk melanjutkan pekerjaan dari titik itu daripada memulai dari awal.
- Optimalisasi Biaya - Dengan mengaktifkan file peristiwa log bergulir, Anda dapat menghemat biaya penyimpanan yang terkait dengan file log. Alih-alih menyimpan satu file log yang berpotensi besar, Anda menyimpan file log yang lebih kecil dan lebih mudah dikelola, yang bisa lebih hemat biaya, terutama untuk pekerjaan yang berjalan lama atau kompleks.

Di lingkungan baru, pengguna dapat secara eksplisit mengaktifkan log bergulir melalui:

```
'–conf': 'spark.eventLog.rolling.enabled=true'
```

atau

```
'–conf': 'spark.eventLog.rolling.enabled=true –conf
spark.eventLog.rolling.maxFileSize=128m'
```

Saat log bergulir diaktifkan, `spark.eventLog.rolling.maxFileSize` tentukan ukuran maksimum file log peristiwa sebelum berguling. Nilai default parameter opsional ini jika tidak ditentukan adalah 128 MB. Minimal adalah 10 MB.

Jumlah maksimum dari semua file peristiwa log gulung yang dihasilkan adalah 2 GB. Untuk AWS Glue pekerjaan tanpa dukungan log bergulir, ukuran file peristiwa log maksimum yang didukung untuk SparkUI adalah 0,5 GB.

Anda dapat mematikan log bergulir untuk pekerjaan streaming dengan meneruskan konfigurasi tambahan. Perhatikan bahwa file log yang sangat besar mungkin mahal untuk dipelihara.

Untuk mematikan log bergulir, berikan konfigurasi berikut:

```
'--spark-ui-event-logs-path': 'true',
'--conf': 'spark.eventLog.rolling.enabled=false'
```

Meluncurkan server sejarah Spark

Anda dapat menggunakan server riwayat Spark untuk memvisualisasikan log Spark pada infrastruktur Anda sendiri. Anda dapat melihat visualisasi yang sama di AWS Glue konsol untuk AWS Glue pekerjaan berjalan pada versi AWS Glue 4.0 atau yang lebih baru dengan log yang dihasilkan dalam format Standar (bukan warisan). Untuk informasi selengkapnya, lihat the section called “Pemantauan dengan Spark UI”.

Anda dapat meluncurkan server riwayat Spark menggunakan templat AWS CloudFormation yang meng-host server pada instans EC2, atau meluncurkan menggunakan Docker secara lokal.

Topik

Meluncurkan server riwayat Spark dan melihat UI Spark menggunakan AWS CloudFormation

Anda dapat menggunakan templat AWS CloudFormation untuk memulai server riwayat Apache Spark dan melihat Spark Web UI. Templat ini adalah contoh yang harus Anda modifikasi untuk memenuhi kebutuhan Anda.

Untuk memulai server riwayat Spark dan melihat Spark UI menggunakan AWS CloudFormation

1. Pilih salah satu tombol Luncurkan Tumpukan dalam tabel berikut. Ini akan meluncurkan tumpukan pada konsol AWS CloudFormation.

Wilayah	Luncurkan
AS Timur (Ohio)	
US East (N. Virginia)	
US West (N. California)	

Wilayah	Luncurkan
US West (Oregon)	Launch Stack
Africa (Cape Town)	Launch Stack
Asia Pacific (Hong Kong)	Launch Stack
Asia Pacific (Mumbai)	Launch Stack
Asia Pacific (Osaka)	Launch Stack
Asia Pacific (Seoul)	Launch Stack
Asia Pacific (Singapore)	Launch Stack
Asia Pacific (Sydney)	Launch Stack
Asia Pacific (Tokyo)	Launch Stack
Canada (Central)	Launch Stack
Europe (Frankfurt)	Launch Stack
Europe (Ireland)	Launch Stack
Europe (London)	Launch Stack
Europe (Milan)	Launch Stack
Europe (Paris)	Launch Stack

Wilayah	Luncurkan
Europe (Stockholm)	
Middle East (Bahrain)	
South America (São Paulo)	

2. Pada halaman Tentukan templat, pilih Selanjutnya.
3. Di halaman Tentukan detail tumpukan, masukkan Nama tumpukan. Masukkan informasi tambahan di bawah Parameter.

   a. Konfigurasi Spark UI

      Berikan informasi berikut ini:

      - Rentang alamat IP — Rentang alamat IP yang dapat digunakan untuk melihat Spark UI. Jika Anda ingin membatasi akses dari rentang alamat IP tertentu, maka Anda harus menggunakan nilai kustom.
      - Port server riwayat — Port untuk Spark UI. Anda dapat menggunakan nilai default.
      - Direktori log peristiwa — Pilih lokasi di mana log peristiwa Spark disimpan dari tugas AWS Glue atau titik akhir pengembangan. Anda harus menggunakan **`s3a://`** untuk skema path log peristiwa.
      - Lokasi paket Spark — Anda dapat menggunakan nilai default.
      - Path keystore — Path keystore SSL/TLS untuk HTTPS. Jika Anda ingin menggunakan file keystore kustom, Anda dapat menentukan path S3 di sini `s3://path_to_your_keystore_file`. Jika Anda membiarkan parameter ini kosong, maka keystore berbasis sertifikat self-signed akan dihasilkan dan digunakan.
      - Kata sandi keystore - Masukkan kata sandi keystore SSL/TLS untuk HTTPS.

   b. Konfigurasi instans EC2

      Berikan informasi berikut ini:

      - Tipe instans — Jenis instans Amazon EC2 yang menjadi host server riwayat Spark. Karena templat ini meluncurkan instans Amazon EC2 di akun Anda, Amazon EC2 akan dikenakan biaya di akun Anda secara terpisah.

- ID AMI terbaru — ID AMI dari Amazon Linux 2 untuk instans server riwayat Spark. Anda dapat menggunakan nilai default.
- ID VPC — ID virtual private cloud (VPC) untuk instans server riwayat Spark. Anda dapat menggunakan salah satu VPC yang tersedia di akun Anda. Menggunakan VPC default dengan ACL Jaringan default tidak dianjurkan. Untuk informasi selengkapnya, lihat VPC Default dan Subnet Default dan Membuat VPC di Panduan Pengguna Amazon VPC.
- ID Subnet — ID untuk instans server riwayat Spark. Anda dapat menggunakan salah satu subnet di VPC Anda. Anda harus dapat mencapai jaringan dari klien Anda ke subnet tersebut. Jika Anda ingin mengakses melalui internet, maka Anda harus menggunakan subnet publik yang memiliki gateway internet di tabel rute.

c. Pilih Selanjutnya.

4. Pada halaman Configure stack options, untuk menggunakan kredensyal pengguna saat ini untuk menentukan cara CloudFormation membuat, memodifikasi, atau menghapus sumber daya dalam tumpukan, pilih Berikutnya. Anda juga dapat menentukan peran di bagian Izin yang akan digunakan, bukan izin pengguna saat ini, lalu pilih Berikutnya.
5. Pada halaman Tinjau, tinjau templat.

   Pilih Saya mengakui bahwa AWS CloudFormation dapat membuat sumber daya IAM, dan kemudian pilih Buat tumpukan.
6. Tunggu tumpukan yang akan dibuat.
7. Buka tab Output.

   a. Salin URL SparkUiPublicUrljika Anda menggunakan subnet publik.

   b. Salin URL SparkUiPrivateUrljika Anda menggunakan subnet pribadi.
8. Buka peramban web, dan tempel di URL. Hal ini memungkinkan Anda mengakses server dengan menggunakan HTTPS pada port yang ditentukan. Peramban Anda mungkin tidak mengenali sertifikat server, di mana Anda harus mengganti perlindungannya dan terus melanjutkan.

Meluncurkan server riwayat Spark dan melihat UI Spark menggunakan Docker

Jika Anda lebih suka akses lokal (tidak memiliki instans EC2 untuk server riwayat Apache Spark), Anda juga dapat menggunakan Docker untuk memulai server riwayat Apache Spark dan melihat Spark UI secara lokal. Dockerfile ini adalah contoh yang Anda harus modifikasi untuk memenuhi kebutuhan Anda.

Prasyarat

Untuk informasi tentang cara menginstal Docker di laptop Anda lihat Komunitas Docker Engine.

Untuk memulai server riwayat Spark dan melihat Spark UI secara lokal menggunakan Docker

1. Unduh file dari GitHub.

   Unduh Dockerfile dan `pom.xml` dari contoh AWS Gluekode.

2. Tentukan apakah Anda ingin menggunakan kredensyal pengguna atau kredensyal pengguna gabungan untuk mengakses. AWS

   - Untuk menggunakan kredensyal pengguna saat ini untuk mengaksesAWS, dapatkan nilai yang akan digunakan untuk `AWS_ACCESS_KEY_ID` dan `AWS_SECRET_ACCESS_KEY` dalam perintah. `docker run` Untuk informasi lebih lanjut, lihat Mengelola access key untuk pengguna IAM dalam Panduan Pengguna IAM.
   - Untuk menggunakan pengguna federasi SAMP 2.0 untuk mengaksesAWS, dapatkan nilai untuk, `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` dan. `AWS_SESSION_TOKEN` Untuk informasi selengkapnya, lihat Meminta kredensyal keamanan sementara

3. Tentukan lokasi direktori log peristiwa Anda, untuk digunakan dalam `docker run` perintah.

4. Buat gambar Docker menggunakan file di direktori lokal, menggunakan nama `glue/sparkui`, dan tag`latest`.

   ```
   $ docker build -t glue/sparkui:latest .
   ```

5. Buat dan mulai wadah docker.

   Dalam perintah berikut, gunakan nilai yang diperoleh sebelumnya pada langkah 2 dan 3.

   a. Untuk membuat wadah docker menggunakan kredensyal pengguna Anda, gunakan perintah yang mirip dengan yang berikut

   ```
   docker run -itd -e SPARK_HISTORY_OPTS="$SPARK_HISTORY_OPTS -
   Dspark.history.fs.logDirectory=s3a://path_to_eventlog
    -Dspark.hadoop.fs.s3a.access.key=AWS_ACCESS_KEY_ID -
   Dspark.hadoop.fs.s3a.secret.key=AWS_SECRET_ACCESS_KEY"
    -p 18080:18080 glue/sparkui:latest "/opt/spark/bin/spark-class
    org.apache.spark.deploy.history.HistoryServer"
   ```

b. Untuk membuat wadah docker menggunakan kredensyal sementara, gunakan `org.apache.hadoop.fs.s3a.TemporaryAWSCredentialsProvider` sebagai penyedia, dan berikan nilai kredensi yang diperoleh pada langkah 2. Untuk informasi selengkapnya, lihat Menggunakan Kredensyal Sesi dengan Sementara AWSCredentialsProvider di dokumentasi Hadoop: Integrasi dengan Amazon Web Services.

```
docker run -itd -e SPARK_HISTORY_OPTS="$SPARK_HISTORY_OPTS -
Dspark.history.fs.logDirectory=s3a://path_to_eventlog
 -Dspark.hadoop.fs.s3a.access.key=AWS_ACCESS_KEY_ID -
Dspark.hadoop.fs.s3a.secret.key=AWS_SECRET_ACCESS_KEY
 -Dspark.hadoop.fs.s3a.session.token=AWS_SESSION_TOKEN
 -
Dspark.hadoop.fs.s3a.aws.credentials.provider=org.apache.hadoop.fs.s3a.TemporaryAWSCred
 -p 18080:18080 glue/sparkui:latest "/opt/spark/bin/spark-class
 org.apache.spark.deploy.history.HistoryServer"
```

> **Note**
>
> Parameter konfigurasi ini berasal dari Hadoop-AWSModul. Anda mungkin perlu menambahkan konfigurasi spesifik berdasarkan kasus penggunaan Anda. Misalnya: pengguna di wilayah terisolasi perlu mengkonfigurasi file `spark.hadoop.fs.s3a.endpoint`.

6. Buka `http://localhost:18080` di browser Anda untuk melihat UI Spark secara lokal.

Pemantauan dengan wawasan menjalankan AWS Glue pekerjaan

AWS Gluejob run insights adalah fitur AWS Glue yang menyederhanakan debugging dan pengoptimalan pekerjaan untuk pekerjaan Anda. AWS Glue AWS Gluemenyediakan Spark UI, serta CloudWatch log dan metrik untuk memantau pekerjaan Anda AWS Glue . Dengan fitur ini, Anda mendapatkan informasi ini tentang eksekusi AWS Glue pekerjaan Anda:

- Nomor baris skrip AWS Glue pekerjaan Anda yang mengalami kegagalan.
- Tindakan percikan yang dieksekusi terakhir dalam rencana kueri Spark tepat sebelum kegagalan pekerjaan Anda.
- Peristiwa pengecualian percikan terkait dengan kegagalan yang disajikan dalam aliran log yang diurutkan waktu.

- Analisis akar penyebab dan tindakan yang disarankan (seperti menyetel skrip Anda) untuk memperbaiki masalah.
- Peristiwa Spark umum (pesan log yang berkaitan dengan tindakan Spark) dengan tindakan yang direkomendasikan yang membahas akar penyebabnya.

Semua wawasan ini tersedia untuk Anda menggunakan dua aliran log baru di CloudWatch log untuk pekerjaan AndaAWS Glue.

Persyaratan

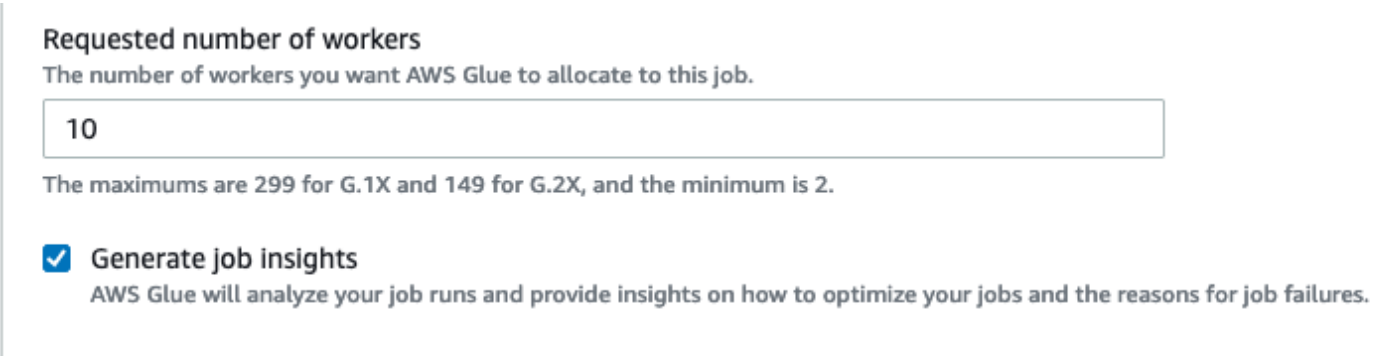
Fitur AWS Glue job run insights tersedia untuk AWS Glue versi 2.0, 3.0, dan 4.0. Anda dapat mengikuti panduan migrasi untuk pekerjaan yang ada untuk memutakhirkannya dari AWS Glue versi yang lebih lama.

Mengaktifkan wawasan menjalankan pekerjaan untuk pekerjaan ETL AWS Glue

Anda dapat mengaktifkan wawasan menjalankan pekerjaan melalui AWS Glue Studio atau CLI.

AWS Glue Studio

Saat membuat pekerjaan melaluiAWS Glue Studio, Anda dapat mengaktifkan atau menonaktifkan wawasan menjalankan pekerjaan di bawah tab Rincian Pekerjaan. Centang apakah kotak Hasilkan wawasan pekerjaan dipilih.



Baris perintah

Jika membuat pekerjaan melalui CLI, Anda dapat memulai pekerjaan dengan satu parameter pekerjaan baru:. `--enable-job-insights = true`

Secara default, aliran log wawasan job run dibuat di bawah grup log default yang sama yang digunakan oleh logging AWS Glue berkelanjutan, yaitu,. `/aws-glue/jobs/logs-v2/` Anda

dapat mengatur nama grup log kustom, filter log, dan konfigurasi grup log menggunakan kumpulan argumen yang sama untuk pencatatan berkelanjutan. Untuk informasi selengkapnya, lihat Mengaktifkan Pencatatan Berkelanjutan untuk AWS Glue Pekerjaan.

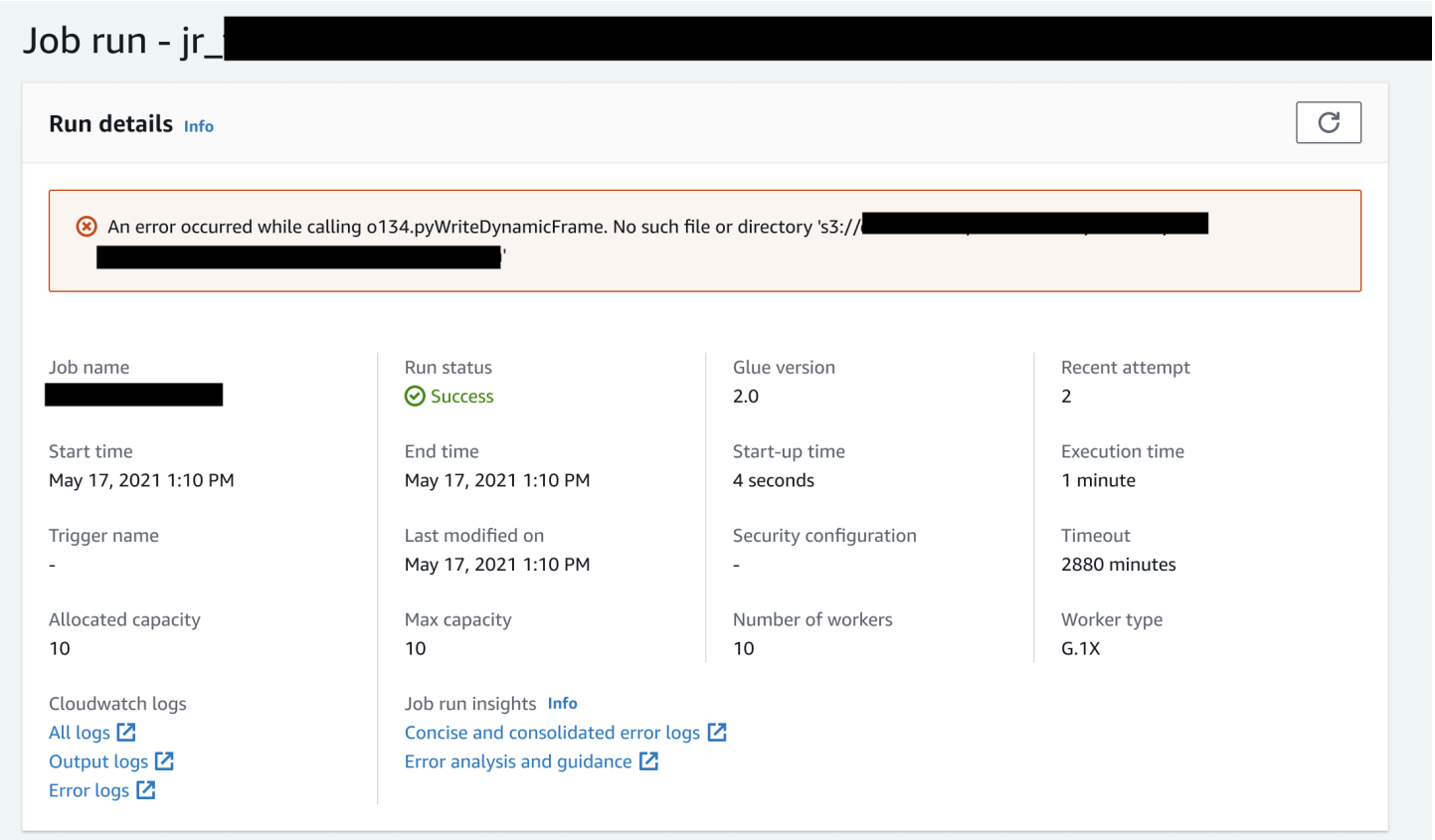
Mengakses aliran log wawasan job run di CloudWatch

Dengan fitur job run insights diaktifkan, mungkin ada dua aliran log yang dibuat saat pekerjaan dijalankan gagal. Ketika pekerjaan selesai dengan sukses, tidak satu pun dari aliran yang dihasilkan.

1. Aliran log analisis pengecualian:`<job-run-id>-job-insights-rca-driver`. Aliran ini menyediakan yang berikut:
   - Nomor baris skrip AWS Glue pekerjaan Anda yang menyebabkan kegagalan.
   - Tindakan percikan yang dieksekusi terakhir dalam rencana kueri Spark (DAG).
   - Peristiwa singkat yang diurutkan waktu dari driver Spark dan pelaksana yang terkait dengan pengecualian. Anda dapat menemukan detail seperti pesan kesalahan lengkap, tugas Spark yang gagal, dan ID pelaksana yang membantu Anda fokus pada aliran log pelaksana tertentu untuk penyelidikan lebih dalam jika diperlukan.
2. Aliran wawasan berbasis aturan:
   - Analisis akar penyebab dan rekomendasi tentang cara memperbaiki kesalahan (seperti menggunakan parameter pekerjaan tertentu untuk mengoptimalkan kinerja).
   - Peristiwa Spark yang relevan berfungsi sebagai dasar untuk analisis akar penyebab dan tindakan yang direkomendasikan.

> **Note**
>
> Aliran pertama hanya akan ada jika peristiwa Spark pengecualian tersedia untuk menjalankan pekerjaan yang gagal, dan aliran kedua hanya akan ada jika ada wawasan yang tersedia untuk menjalankan pekerjaan yang gagal. Misalnya, jika pekerjaan Anda berhasil diselesaikan, tidak satu pun aliran akan dihasilkan; jika pekerjaan Anda gagal tetapi tidak ada aturan yang ditentukan layanan yang dapat cocok dengan skenario kegagalan Anda, maka hanya aliran pertama yang akan dihasilkan.

Jika pekerjaan dibuat dariAWS Glue Studio, tautan ke aliran di atas juga tersedia di bawah tab rincian pekerjaan (Wawasan Jalankan pekerjaan) sebagai “Log kesalahan ringkas dan terkonsolidasi” dan “Analisis dan panduan kesalahan”.



Contoh untuk wawasan menjalankan AWS Glue pekerjaan

Di bagian ini kami menyajikan contoh bagaimana fitur job run insights dapat membantu Anda menyelesaikan masalah dalam pekerjaan yang gagal. Dalam contoh ini, pengguna lupa mengimpor modul yang diperlukan (tensorflow) dalam AWS Glue pekerjaan untuk menganalisis dan membangun model pembelajaran mesin pada data mereka.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from pyspark.sql.types import *
from pyspark.sql.functions import udf,col

args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
```

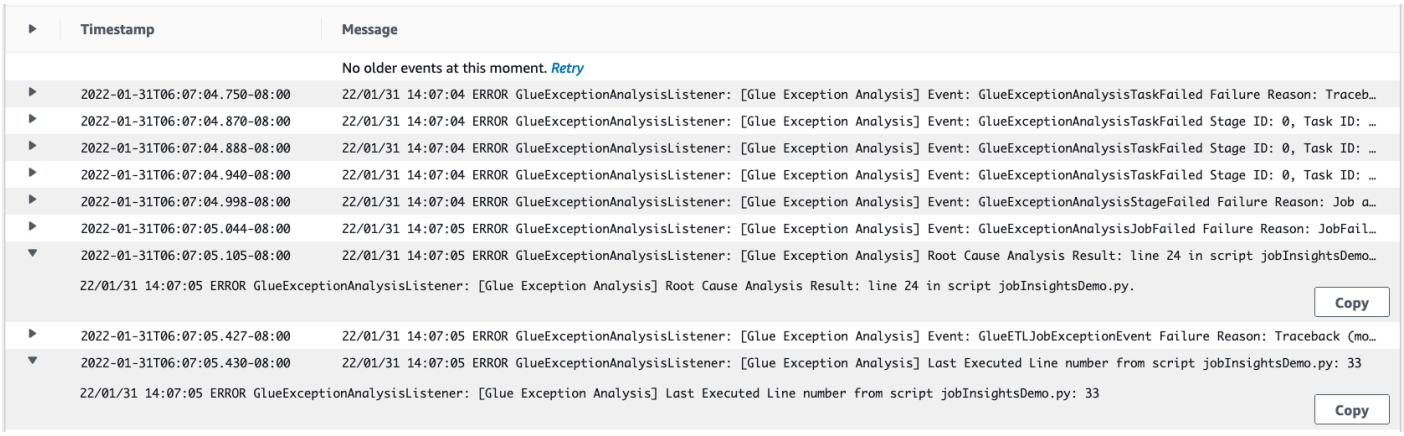
```
job = Job(glueContext)
job.init(args['JOB_NAME'], args)

data_set_1 = [1, 2, 3, 4]
data_set_2 = [5, 6, 7, 8]

scoresDf = spark.createDataFrame(data_set_1, IntegerType())

def data_multiplier_func(factor, data_vector):
    import tensorflow as tf
    with tf.compat.v1.Session() as sess:
        x1 = tf.constant(factor)
        x2 = tf.constant(data_vector)
        result = tf.multiply(x1, x2)
        return sess.run(result).tolist()

data_multiplier_udf = udf(lambda x:data_multiplier_func(x, data_set_2),
 ArrayType(IntegerType(),False))
factoredDf = scoresDf.withColumn("final_value", data_multiplier_udf(col("value")))
print(factoredDf.collect())
```

Tanpa fitur job run insights, karena pekerjaan gagal, Anda hanya melihat pesan ini dilemparkan oleh Spark:

`An error occurred while calling o111.collectToPython. Traceback (most recent call last):`

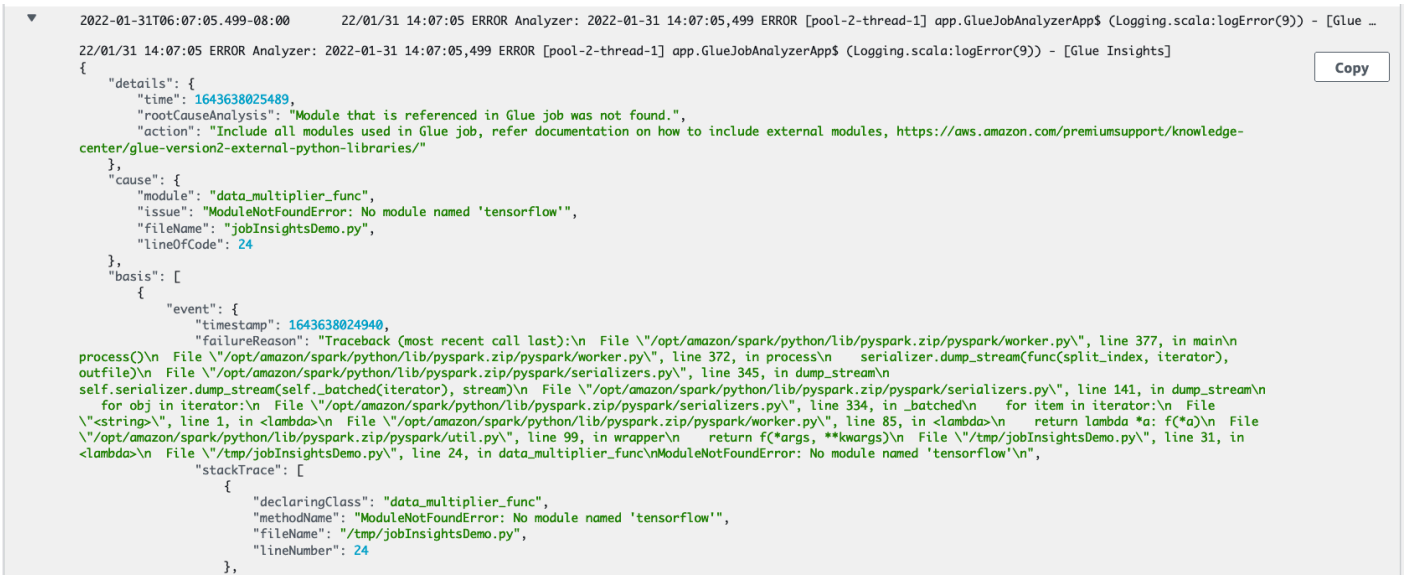
Pesannya ambigu dan membatasi pengalaman debugging Anda. Dalam hal ini, fitur ini memberi Anda wawasan tambahan dalam dua aliran CloudWatch log:

1. Aliran `job-insights-rca-driver` log:
   - Peristiwa pengecualian: Aliran log ini memberi Anda peristiwa pengecualian Spark yang terkait dengan kegagalan yang dikumpulkan dari driver Spark dan pekerja terdistribusi yang berbeda. Peristiwa ini membantu Anda memahami propagasi pengecualian yang diatur waktu karena kode yang salah dijalankan di seluruh tugas Spark, pelaksana, dan tahapan yang didistribusikan ke seluruh pekerja. AWS Glue
   - Nomor baris: Aliran log ini mengidentifikasi baris 21, yang membuat panggilan untuk mengimpor modul Python yang hilang yang menyebabkan kegagalan; itu juga mengidentifikasi baris 24, panggilan ke Spark `collect()` Action, sebagai baris terakhir yang dieksekusi dalam skrip Anda.



2. Aliran `job-insights-rule-driver` log:

   - Akar penyebab dan rekomendasi: Selain nomor baris dan nomor baris terakhir yang dieksekusi untuk kesalahan dalam skrip Anda, aliran log ini menunjukkan analisis akar penyebab dan rekomendasi bagi Anda untuk mengikuti AWS Glue dokumen dan mengatur parameter pekerjaan yang diperlukan untuk menggunakan modul Python tambahan dalam pekerjaan AndaAWS Glue.
   - Peristiwa dasar: Aliran log ini juga menunjukkan peristiwa pengecualian Spark yang dievaluasi dengan aturan yang ditentukan layanan untuk menyimpulkan akar penyebab dan memberikan rekomendasi.



Pemantauan CloudWatch dengan Amazon

Anda dapat memantau AWS Glue menggunakan Amazon CloudWatch, yang mengumpulkan dan memproses data mentah dari AWS Glue menjadi metrik yang dapat dibaca. near-real-time Statistik ini

dicatat selama jangka waktu dua minggu, sehingga Anda dapat mengakses informasi historis untuk perspektif yang lebih baik tentang performa aplikasi web atau layanan Anda. Secara default, data AWS Glue metrik dikirim ke CloudWatch otomatis. Untuk informasi selengkapnya, lihat Apa yang dimaksud dengan Amazon CloudWatch? di Panduan CloudWatch Pengguna Amazon, danMetrik AWS Glue.

Penebangan terus menerus

AWS Glue juga mendukung pencatatan log berkelanjutan waktu nyata untuk tugas AWS Glue. Ketika pencatatan log terus-menerus diaktifkan untuk tugas, Anda dapat melihat log waktu nyata pada AWS Glue konsol atau dasbor CloudWatch konsol. Untuk informasi selengkapnya, lihat Pencatatan terus menerus untuk AWS Glue pekerjaan.

Metrik observabilitas

Saat metrik observabilitas Job diaktifkan, Amazon CloudWatch metrik tambahan akan dihasilkan saat job dijalankan. Gunakan metrik AWS Glue Observability untuk menghasilkan wawasan tentang apa yang terjadi di dalam diri Anda AWS Glue untuk meningkatkan triaging dan analisis masalah.

Topik

- Pemantauan AWS Glue menggunakan CloudWatch metrik Amazon
- Menyiapkan CloudWatch alarm Amazon di profil AWS Glue pekerjaan
- Pencatatan terus menerus untuk AWS Glue pekerjaan
- Pemantauan dengan metrik AWS Glue Observabilitas

Pemantauan AWS Glue menggunakan CloudWatch metrik Amazon

Anda dapat membuat profil dan memantau operasi AWS Glue menggunakan pembuat profil tugas AWS Glue. Ini mengumpulkan dan memproses data mentah dari AWS Glue pekerjaan menjadi metrik yang dapat dibaca, mendekati waktu nyata yang disimpan di Amazon. CloudWatch Statistik ini disimpan dan dikumpulkan CloudWatch sehingga Anda dapat mengakses informasi historis untuk perspektif yang lebih baik tentang kinerja aplikasi Anda.

> **Note**
>
> Anda mungkin dikenakan biaya tambahan saat mengaktifkan metrik pekerjaan dan metrik CloudWatch khusus dibuat. Untuk informasi selengkapnya, lihat CloudWatch harga Amazon.

AWS Glueikhtisar metrik

Saat Anda berinteraksiAWS Glue, itu mengirimkan metrik ke CloudWatch. Anda dapat melihat metrik ini menggunakan AWS Glue konsol (metode pilihan), dasbor CloudWatch konsol, atau AWS Command Line Interface (AWS CLI).

Untuk melihat metrik menggunakan dasbor konsol AWS Glue

Anda dapat melihat gambaran umum atau grafik detail metrik untuk tugas, atau grafik terperinci untuk eksekusi tugas.

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pilih Job run monitoring.
3. Di Job run pilih Tindakan untuk menghentikan pekerjaan yang sedang berjalan, melihat pekerjaan, atau memundurkan bookmark pekerjaan.
4. Pilih pekerjaan, lalu pilih Lihat rincian jalankan untuk melihat informasi tambahan tentang pekerjaan yang dijalankan.

Untuk melihat metrik menggunakan dasbor CloudWatch konsol

Metrik dikelompokkan terlebih dahulu berdasarkan namespace layanan, kemudian berdasarkan berbagai kombinasi dimensi dalam setiap namespace.

1. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Di panel navigasi, pilih Metrik.
3. Pilih namespace Glue.

Untuk melihat metrik menggunakan konsol AWS CLI

- Pada jendela perintah, gunakan perintah berikut.

```
aws cloudwatch list-metrics --namespace Glue
```

AWS Gluemelaporkan metrik CloudWatch setiap 30 detik, dan dasbor CloudWatch metrik dikonfigurasi untuk menampilkannya setiap menit. Metrik AWS Glue merupakan nilai delta dari nilai

yang dilaporkan sebelumnya. Jika sesuai, dasbor metrik meng-agregat (jumlah) nilai 30 detik untuk mendapatkan nilai untuk seluruh menit terakhir.

AWS Glueperilaku metrik untuk pekerjaan Spark

AWS Gluemetrik diaktifkan pada inisialisasi `GlueContext` dalam skrip dan umumnya diperbarui hanya pada akhir tugas Apache Spark. Mereka mewakili nilai-nilai agregat di semua tugas Spark yang sudah selesai sejauh ini.

Namun, metrik Spark yang AWS Glue diteruskan ke umumnya CloudWatch merupakan nilai absolut yang mewakili keadaan saat ini pada saat dilaporkan. AWS Gluemelaporkannya CloudWatch setiap 30 detik, dan dasbor metrik umumnya menunjukkan rata-rata di seluruh titik data yang diterima dalam 1 menit terakhir.

Nama metrik AWS Glue semuanya didahului oleh salah satu jenis prefiks berikut:

- `glue.driver.` — Metrik yang namanya dimulai dengan prefiks ini mewakili metrik AWS Glue yang dikumpulkan dari semua pelaksana di driver Spark, atau metrik Spark yang sesuai dengan driver Spark.
- `glue.executorId.` — executorId adalah nomor pelaksana Spark tertentu. Nomor itu sesuai dengan pelaksana yang tercantum dalam log.
- `glue.ALL.` — Metrik yang namanya dimulai dengan prefiks ini menjumlahkan nilai dari semua pelaksana Spark.

Metrik AWS Glue

AWS Glueprofil dan mengirimkan metrik berikut ke CloudWatch setiap 30 detik, dan Dasbor AWS Glue Metrik melaporkannya sekali dalam satu menit:

Metrik	Deskripsi
`glue.driver.aggregate.bytesRead`	Jumlah byte yang dibaca dari semua sumber data oleh semua tugas Spark yang diselesaikan yang berjalan di semua pelaksana. Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (hitung).

Metrik	Deskripsi
	Statistik yang valid: SUM. Metrik ini adalah nilai delta dari nilai terakhir yang dilaporkan, jadi pada Dasbor Metrik AWS Glue, statistik SUM digunakan untuk agregasi. Unit: Bita Dapat digunakan untuk memantau: • Byte dibaca. • Kemajuan tugas. • Sumber data JDBC. • Masalah Bookmark Tugas. • Varians di seluruh Eksekusi Tugas. Metrik ini dapat digunakan dengan cara yang sama seperti metrik `glue.ALL.s3.filesystem.read_bytes` , dengan perbedaan bahwa metrik ini diperbarui pada akhir tugas Spark dan menangkap sumber data non-S3 juga.

Metrik	Deskripsi
`glue.driver.aggregate.elapsedTime`	Waktu berlalu ETL dinyatakan dalam milidetik (tidak termasuk waktu bootstrap tugas). Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (hitung). Statistik yang valid: SUM. Metrik ini adalah nilai delta dari nilai terakhir yang dilaporkan, jadi pada Dasbor Metrik AWS Glue, statistik SUM digunakan untuk agregasi. Unit: Milidetik Dapat digunakan untuk menentukan berapa lama waktu rata-rata yang dibutuhkan untuk menjalankan sebuah eksekusi tugas. Beberapa cara menggunakan data: • Mengatur alarm untuk simpul dengan performa buruk. • Mengukur varians di seluruh tugas berjalan.

Metrik	Deskripsi
`glue.driver.aggregate.numCompletedStages`	Jumlah tahap yang diselesaikan dalam tugas. Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (hitung). Statistik yang valid: SUM. Metrik ini adalah nilai delta dari nilai terakhir yang dilaporkan, jadi pada Dasbor Metrik AWS Glue, statistik SUM digunakan untuk agregasi. Unit: Jumlah Dapat digunakan untuk memantau: • Kemajuan tugas. • Lini waktu per tahap eksekusi tugas, bila berkorelasi dengan metrik lainnya. Beberapa cara menggunakan data: • Mengidentifikasi tahapan yang banyak permintaannya dalam pelaksanaan tugas. • Mengatur alarm untuk lonjakan yang berkorelasi (tahap yang banyak permintaannya) di seluruh eksekusi tugas.

Metrik	Deskripsi
`glue.driver.aggregate.numCompletedTasks`	Jumlah tugas yang telah selesai dalam tugas tersebut. Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (hitung). Statistik yang valid: SUM. Metrik ini adalah nilai delta dari nilai terakhir yang dilaporkan, jadi pada Dasbor Metrik AWS Glue, statistik SUM digunakan untuk agregasi. Unit: Jumlah Dapat digunakan untuk memantau: • Kemajuan tugas. • Paralelisme dalam sebuah tahap.

Metrik	Deskripsi
`glue.driver.aggregate.numFailedTasks`	Jumlah tugas yang gagal. Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (hitung). Statistik yang valid: SUM. Metrik ini adalah nilai delta dari nilai terakhir yang dilaporkan, jadi pada Dasbor Metrik AWS Glue, statistik SUM digunakan untuk agregasi. Unit: Jumlah Dapat digunakan untuk memantau: • Kelainan data yang menyebabkan tugas dari Tugas gagal. • Kelainan klaster yang menyebabkan tugas dari Tugas gagal. • Kelainan skrip yang menyebabkan tugas dari Tugas gagal. Data dapat digunakan untuk mengatur alarm untuk peningkatan kegagalan yang mungkin menunjukkan kelainan pada data, klaster atau skrip.

Metrik	Deskripsi
`glue.driver.aggregate.numKilledTasks`	Jumlah tugas yang dihentikan. Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (hitung). Statistik yang valid: SUM. Metrik ini adalah nilai delta dari nilai terakhir yang dilaporkan, jadi pada Dasbor Metrik AWS Glue, statistik SUM digunakan untuk agregasi. Unit: Jumlah Dapat digunakan untuk memantau: • Kelainan dalam Data Skew yang mengakibatkan pengecualian (OOM) yang menghentikan tugas. • Kelainan skrip yang mengakibatkan pengecualian (OOM) yang menghentikan tugas. Beberapa cara menggunakan data: • Mengatur alarm untuk peningkatan kegagalan yang menunjukkan kelainan data. • Mengatur alarm untuk peningkatan kegagalan yang menunjukkan kelainan klaster. • Mengatur alarm untuk peningkatan kegagalan yang menunjukkan kelainan skrip.

Metrik	Deskripsi
`glue.driver.aggregate.recordsRead`	Jumlah catatan yang dibaca dari semua sumber data oleh semua tugas Spark yang telah diselesaikan yang berjalan di semua pelaksana.. Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (hitung). Statistik yang valid: SUM. Metrik ini adalah nilai delta dari nilai terakhir yang dilaporkan, jadi pada Dasbor Metrik AWS Glue, statistik SUM digunakan untuk agregasi. Unit: Jumlah Dapat digunakan untuk memantau: • Catatan dibaca. • Kemajuan tugas. • Sumber data JDBC. • Masalah Bookmark Tugas. • Skew di Eksekusi Tugas selama berhari-hari. Metrik ini dapat digunakan dengan cara yang serupa seperti metrik `glue.ALL.s3.filesystem.read_bytes` , dengan perbedaan bahwa metrik ini diperbarui pada akhir tugas Spark.

Metrik	Deskripsi
`glue.driver.aggregate.shuffleBytesWritten`	Jumlah byte yang ditulis oleh semua pelaksana untuk mengacak data di antara mereka sejak laporan sebelumnya (dijumlahkan oleh Dasbor Metrik AWS Glue sebagai jumlah byte yang ditulis untuk tujuan ini selama menit sebelumnya). Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (hitung). Statistik yang valid: SUM. Metrik ini adalah nilai delta dari nilai terakhir yang dilaporkan, jadi pada Dasbor Metrik AWS Glue, statistik SUM digunakan untuk agregasi. Unit: Bita Dapat digunakan untuk memantau: Acakan data dalam tugas (gabungan besar, groupBy, partisi ulang, menyatu). Beberapa cara menggunakan data: • Melakukan partisi ulang atau dekompresi file input besar sebelum diproses lebih lanjut. • Melakukan partisi ulang data secara lebih seragam untuk menghindari hot key. • Melakukan pra-filter data sebelum operasi menggabungkan atau GroupBy.

Metrik	Deskripsi
`glue.driver.aggregate.shuffleLocalBytesRead`	Jumlah byte yang dibaca oleh semua pelaksana untuk mengacak data di antara mereka sejak laporan sebelumnya (dijumlahkan oleh Dasbor Metrik AWS Glue sebagai jumlah byte yang dibaca untuk tujuan ini selama menit sebelumnya). Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (hitung). Statistik yang valid: SUM. Metrik ini adalah nilai delta dari nilai terakhir yang dilaporkan, jadi pada Dasbor Metrik AWS Glue, statistik SUM digunakan untuk agregasi. Unit: Bita Dapat digunakan untuk memantau: Acakan data dalam tugas (gabungan besar, groupBy, partisi ulang, menyatu). Beberapa cara menggunakan data: • Melakukan partisi ulang atau dekompresi file input besar sebelum diproses lebih lanjut. • Melakukan partisi ulang data secara lebih seragam menggunakan hot key. • Melakukan pra-filter data sebelum operasi menggabungkan atau GroupBy.

Metrik	Deskripsi
`glue.driver.BlockManager.disk.diskSpaceUsed_MB`	Jumlah megabyte ruang disk yang digunakan di semua pelaksana. Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (gauge). Statistik yang valid: Rata-rata. Ini adalah sebuah metrik Spark, yang dilaporkan sebagai nilai absolut. Unit: Megabyte Dapat digunakan untuk memantau: • Ruang disk yang digunakan untuk blok yang mewakili partisi RDD yang di-cache. • Ruang disk yang digunakan untuk blok yang mewakili output acak sedang. • Ruang disk yang digunakan untuk blok yang mewakili siaran. Beberapa cara menggunakan data: • Mengidentifikasi kegagalan tugas karena peningkatan penggunaan disk. • Mengidentifikasi partisi besar yang mengakibatkan tumpahan atau acakan. • Meningkatkan kapasitas DPU yang disediakan untuk memperbaiki masalah ini.

Metrik	Deskripsi
`glue.driver.ExecutorAllocationManager.executors.numberAllExecutors`	Jumlah aktif pelaksana tugas yang berjalan. Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (gauge). Statistik yang valid: Rata-rata. Ini adalah sebuah metrik Spark, yang dilaporkan sebagai nilai absolut. Unit: Jumlah Dapat digunakan untuk memantau: • Aktivitas tugas. • Pelaksana yang mempunyai performa buruk (dengan beberapa pelaksana berjalan saja) • Paralelisme tingkat pelaksana saat ini. Beberapa cara menggunakan data: • Partisi ulang atau dekompresi file input besar terlebih dahulu jika klaster di kurang-dimanfaatkan. • Mengidentifikasi tahap atau penundaan eksekusi tugas karena skenario adanya simpul dengan performa buruk. • • Bandingkan dengan numberMaxNeeded Executors untuk memahami backlog untuk penyediaan lebih banyak DPU.

Metrik	Deskripsi
`glue.driver.ExecutorAllocationManager.executors.numberMaxNeededExecutors`	Jumlah maksimum (aktif berjalan dan tertunda) pelaksana tugas yang diperlukan untuk memenuhi beban saat ini. Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (gauge). Statistik yang Valid: Maksimum. Ini adalah sebuah metrik Spark, yang dilaporkan sebagai nilai absolut. Unit: Jumlah Dapat digunakan untuk memantau: • Aktivitas tugas. • Paralelisme tingkat pelaksana saat ini dan backlog tugas tertunda belum dijadwalkan karena pelaksana tidak tersedia karena kapasitas DPU atau pelaksana yang dihentikan/gagal. Beberapa cara menggunakan data: • Mengidentifikasi penundaan/backlog dari antrean penjadwalan. • Mengidentifikasi tahap atau penundaan eksekusi tugas karena skenario adanya simpul dengan performa buruk. • Bandingkan dengan numberAllExecutors untuk memahami backlog untuk penyediaan lebih banyak DPU. • Meningkatkan kapasitas DPU yang disediakan untuk memperbaiki penundaan backlog pelaksana.

Metrik	Deskripsi

`glue.driver.jvm.heap.usage`
 	
`glue.executorId.jvm.heap.usage`
 	
`glue.ALL.jvm.heap.usage`
	Fraksi memori digunakan oleh tumpukan JVM untuk driver ini (skala: 0-1) untuk driver, pelaksana diidentif ikasi oleh executorId, atau SEMUA pelaksana.
	
	Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (gauge).
	
	Statistik yang valid: Rata-rata. Ini adalah sebuah metrik Spark, yang dilaporkan sebagai nilai absolut.
	
	Unit: Persentase
	
	Dapat digunakan untuk memantau:
	
	• out-of-memory Kondisi driver (OOM) menggunak an`glue.driver.jvm.heap.usage` .
	• out-of-memory Kondisi pelaksana (OOM) menggunakan`glue.ALL.jvm.heap.usage` .
	
	Beberapa cara menggunakan data:
	
	• Mengidentifikasi konsumsi memori id eksekutor dan tahapan.
	• Mengidentifikasi id pelaksana dan tahapan yang memiliki performa buruk.
	• Identifikasi out-of-memory kondisi pengemudi (OOM).
	• Identifikasi out-of-memory kondisi pelaksana (OOM) dan dapatkan ID pelaksana yang sesuai sehingga bisa mendapatkan jejak tumpukan dari log pelaksana.
	• Identifikasi file atau partisi yang mungkin memiliki kemiringan data yang mengakibatkan penyimpang atau kondisi (OOM). out-of-memory

Metrik	Deskripsi

`glue.driver.jvm.heap.used`
 	
`glue.executorId.jvm.heap.used`
 	
`glue.ALL.jvm.heap.used`
	Jumlah byte memori yang digunakan oleh timbunan JVM untuk driver, pelaksana yang diidentifikasi oleh executorId, atau SEMUA pelaksana.
	
	Dimensi yang valid: `JobName` (nama AWS Glue Job), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (gauge).
	
	Statistik yang valid: Rata-rata. Ini adalah sebuah metrik Spark, yang dilaporkan sebagai nilai absolut.
	
	Unit: Bita
	
	Dapat digunakan untuk memantau:
	
	• out-of-memory Kondisi pengemudi (OOM).
	• out-of-memory Kondisi pelaksana (OOM).
	
	Beberapa cara menggunakan data:
	
	• Mengidentifikasi konsumsi memori id eksekutor dan tahapan.
	• Mengidentifikasi id pelaksana dan tahapan yang memiliki performa buruk.
	• Identifikasi out-of-memory kondisi pengemudi (OOM).
	• Identifikasi out-of-memory kondisi pelaksana (OOM) dan dapatkan ID pelaksana yang sesuai sehingga bisa mendapatkan jejak tumpukan dari log pelaksana.
	• Identifikasi file atau partisi yang mungkin memiliki kemiringan data yang mengakibatkan penyimpang atau kondisi (OOM). out-of-memory

Metrik	Deskripsi

`glue.driver.s3.filesystem.read_bytes`
 	
`glue.executorId.s3.filesystem.read_bytes`
 	
`glue.ALL.s3.filesystem.read_bytes`
	Jumlah byte yang dibaca dari Amazon S3 oleh driver, pelaksana yang diidentifkasi oleh executorId, atau SEMUA pelaksana sejak laporan sebelumnya (yang diagregat oleh Dasbor Metrik AWS Glue sebagai jumlah byte yang dibaca selama menit sebelumnya).
	
	Dimensi yang valid: `JobName`, `JobRunId`, dan `Type`(meteran).
	
	Statistik yang valid: SUM. Metrik ini adalah nilai delta dari nilai terakhir yang dilaporkan, jadi pada Dasbor Metrik AWS Glue, statistik SUM digunakan untuk agregasi. Daerah di bawah kurva pada Dasbor Metrik AWS Glue dapat digunakan untuk membandingkan byte dibaca oleh dua eksekusi tugas yang berbeda secara visual.
	
	Unit: Byte.
	
	Dapat digunakan untuk memantau:
	
	• Pergerakan data ETL.
	• Kemajuan tugas.
	• Masalah bookmark tugas (data yang diproses, diproses ulang, dan dilewati).
	• Perbandingan baca untuk tingkat penyerapan dari sumber data eksternal.
	• Varians di seluruh eksekusi tugas.
	
	Data yang dihasilkan dapat digunakan untuk:
	
	• Perencanaan kapasitas DPU.
	• Mengatur alarm untuk lonjakan besar atau penurunan data dibaca untuk eksekusi tugas dan tahap tugas.

Metrik	Deskripsi

`glue.driver.s3.filesystem.write_bytes`
 	
`glue.executorId.s3.filesystem.write_bytes`
 	
`glue.ALL.s3.filesystem.write_bytes`
	Jumlah byte yang ditulis ke Amazon S3 oleh driver, pelaksana yang diidentifkasi oleh executorId, atau SEMUA pelaksana sejak laporan sebelumnya (yang diagregat oleh Dasbor Metrik AWS Glue sebagai jumlah byte yang ditulis selama menit sebelumnya).
	
	Dimensi yang valid: `JobName`, `JobRunId`, dan `Type`(meteran).
	
	Statistik yang valid: SUM. Metrik ini adalah nilai delta dari nilai terakhir yang dilaporkan, jadi pada Dasbor Metrik AWS Glue, statistik SUM digunakan untuk agregasi. Daerah di bawah kurva pada Dasbor Metrik AWS Glue dapat digunakan untuk membandingkan byte ditulis oleh dua eksekusi tugas yang berbeda secara visual.
	
	Unit: Bita
	
	Dapat digunakan untuk memantau:
	
	• Pergerakan data ETL.
	• Kemajuan tugas.
	• Masalah bookmark tugas (data yang diproses, diproses ulang, dan dilewati).
	• Perbandingan baca untuk tingkat penyerapan dari sumber data eksternal.
	• Varians di seluruh eksekusi tugas.
	
	Beberapa cara menggunakan data:
	
	• Perencanaan kapasitas DPU.
	• Mengatur alarm untuk lonjakan besar atau penurunan data dibaca untuk eksekusi tugas dan tahap tugas.

Metrik	Deskripsi
`glue.driver.streaming.numRecords`	Jumlah catatan yang diterima dalam batch mikro. Metrik ini hanya tersedia untuk pekerjaan AWS Glue streaming dengan AWS Glue versi 2.0 ke atas. Dimensi yang valid: `JobName` (nama AWS Glue pekerjaan), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (hitung). Statistik yang Valid: Jumlah, Maksimum, Minimum, Rata-rata, Persentil Unit: Jumlah Dapat digunakan untuk memantau: • Catatan dibaca. • Kemajuan tugas.
`glue.driver.streaming.batchProcessingTimeInMs`	Waktu yang dibutuhkan untuk memproses batch dalam milidetik. Metrik ini hanya tersedia untuk pekerjaan AWS Glue streaming dengan AWS Glue versi 2.0 ke atas. Dimensi yang valid: `JobName` (nama AWS Glue pekerjaan), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (hitung). Statistik yang Valid: Jumlah, Maksimum, Minimum, Rata-rata, Persentil Unit: Jumlah Dapat digunakan untuk memantau: • Kemajuan tugas. • Kinerja skrip.

Metrik	Deskripsi

`glue.driver.system.cpuSystemLoad`
 	
`glue.executorId.system.cpuSystemLoad`
 	
`glue.ALL.system.cpuSystemLoad`
	Fraksi beban sistem CPU yang digunakan (skala: 0-1) oleh driver, yakni pelaksana yang diidentifikasi oleh executorId, atau SEMUA pelaksana.
	
	Dimensi yang valid: `JobName` (nama AWS Glue pekerjaan), `JobRunId` (JobRun ID. atau`ALL`), dan `Type` (pengukur).
	
	Statistik yang valid: Rata-rata. Ini adalah sebuah metrik yang dilaporkan sebagai nilai absolut.
	
	Unit: Persentase
	
	Dapat digunakan untuk memantau:
	
	• Beban CPU driver.
	• Beban CPU pelaksana.
	• Mendeteksi pelaksana atau tahapan terikat-CPU atau terikat-IO dalam sebuah Tugas.
	
	Beberapa cara menggunakan data:
	
	• Perencanaan kapasitas DPU bersama dengan Metrik IO (Byte Baca/Byte Acak, Paralelisme Tugas) dan jumlah metrik pelaksana maksimum yang dibutuhkan.
	• Mengidentifikasi rasio terikat-CPU/IO. Hal ini memungkinkan untuk melakukan pemartisian ulang dan meningkatkan kapasitas yang disediakan untuk tugas yang berjalan dalam waktu lama dengan set data yang bisa dipecah yang memiliki pemanfaatan CPU yang lebih rendah.

Dimensi untuk AWS Glue Metrik

Metrik AWS Glue menggunakan namespace AWS Glue dan menyediakan metrik untuk dimensi berikut:

Dimensi	Deskripsi
`JobName`	Dimensi ini mem-filter untuk metrik dari semua eksekusi tugas dari tugas AWS Glue tertentu.
`JobRunId`	Dimensi ini menyaring metrik AWS Glue pekerjaan tertentu yang dijalankan oleh JobRun ID, atau`ALL`.
`Type`	Dimensi ini mem-filter untuk metrik dengan `count` (jumlah agregat) atau `gauge` (nilai pada satu titik waktu).

Untuk informasi selengkapnya, lihat Panduan CloudWatch Pengguna Amazon.

Menyiapkan CloudWatch alarm Amazon di profil AWS Glue pekerjaan

AWS GlueMetrik juga tersedia di Amazon CloudWatch. Anda dapat mengatur alarm pada metrik AWS Glue untuk tugas terjadwal.

Beberapa skenario umum untuk menyiapkan alarm adalah sebagai berikut:

- Tugas kehabisan memori (OOM): Mengatur alarm ketika penggunaan memori melebihi penggunaan rata-rata normal untuk driver atau pelaksana untuk sebuah tugas AWS Glue.
- Pelaksana yang memiliki performa buruk: Mengatur alarm ketika jumlah pelaksana jatuh di bawah ambang batas tertentu selama durasi waktu yang besar dalam sebuah tugas AWS Glue.
- Data backlog atau pemrosesan ulang: Bandingkan metrik dari masing-masing pekerjaan dalam alur kerja menggunakan ekspresi matematika. CloudWatch Anda kemudian dapat memicu alarm pada nilai ekspresi yang dihasilkan (seperti rasio byte yang ditulis oleh tugas dan byte dibaca oleh tugas berikut).

Untuk petunjuk mendetail tentang pengaturan alarm, lihat Membuat atau Mengedit CloudWatch Alarm di Panduan Pengguna CloudWatch Acara Amazon.

Untuk memantau dan men-debug skenario menggunakan CloudWatch, lihatPemantauan dan debugging Job.

Pencatatan terus menerus untuk AWS Glue pekerjaan

AWS Glue menyediakan pencatatan log secara waktu nyata dan berkelanjutan untuk tugas AWS Glue. Anda dapat melihat log pekerjaan Apache Spark real-time di Amazon CloudWatch, termasuk log driver, log pelaksana, dan bilah kemajuan pekerjaan Apache Spark. Melihat log waktu nyata memberi Anda perspektif yang lebih baik tentang tugas yang sedang berjalan.

Ketika Anda memulai AWS Glue pekerjaan, ia mengirimkan informasi pencatatan waktu nyata ke CloudWatch (setiap 5 detik dan sebelum setiap penghentian pelaksana) setelah aplikasi Spark mulai berjalan. Anda dapat melihat log di AWS Glue konsol atau dasbor CloudWatch konsol.

Fitur pencatatan log berkelanjutan mencakup kemampuan berikut:

- Pencatatan log berkelanjutan
- Pencatat log skrip kustom untuk mencatat log pesan spesifik-aplikasi
- Sebuah bar kemajuan konsol untuk melacak status berjalan dari tugas AWS Glue saat ini

Untuk informasi tentang bagaimana pencatatan log berkelanjutan didukung di AWS Glue Versi 2.0, lihat Menjalankan Tugas ETL Spark dengan Waktu Pemulaian yang Dikurangi.

Anda dapat membatasi akses ke grup CloudWatch log atau aliran untuk peran IAM untuk membaca log. Untuk detail selengkapnya tentang pembatasan akses, lihat Menggunakan kebijakan berbasis identitas (kebijakan IAM) untuk CloudWatch Log dalam dokumentasi. CloudWatch

> **Note**
>
> Anda mungkin dikenakan biaya tambahan ketika Anda mengaktifkan pencatatan berkelanjutan dan peristiwa CloudWatch log tambahan dibuat. Untuk informasi selengkapnya, lihat CloudWatch harga Amazon.

Topik

- Mengaktifkan pencatatan berkelanjutan untuk pekerjaan AWS Glue
- Melihat logging berkelanjutan untuk AWS Glue pekerjaan

Mengaktifkan pencatatan berkelanjutan untuk pekerjaan AWS Glue

Anda dapat mengaktifkan logging terus menerus menggunakan AWS Glue konsol atau melalui AWS Command Line Interface (AWS CLI).

Anda dapat mengaktifkan pencatatan berkelanjutan saat membuat pekerjaan baru, mengedit pekerjaan yang ada, atau mengaktifkannya melalui AWS CLI.

Anda juga dapat menentukan opsi konfigurasi kustom seperti nama grup Amazon CloudWatch log, awalan aliran CloudWatch log sebelum ID driver/executor ID AWS Glue job run, dan pola konversi log untuk pesan log. Konfigurasi ini membantu Anda mengatur log agregat dalam grup CloudWatch log kustom dengan kebijakan kedaluwarsa yang berbeda, dan menganalisisnya lebih lanjut dengan awalan aliran log kustom dan pola konversi.

Topik

Menggunakan AWS Management Console

Ikuti langkah-langkah ini untuk menggunakan konsol tersebut guna mengaktifkan pencatatan log berkelanjutan saat membuat atau mengedit tugas AWS Glue.

Untuk membuat sebuah tugas AWS Glue dengan pencatatan log berkelanjutan

1. Masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pilih pekerjaan ETL.
3. Pilih Visual ETL.
4. Di tab Job details, perluas bagian Advanced properties.
5. Di bawah Pencatatan berkelanjutan pilih Aktifkan log masuk CloudWatch.

Untuk mengaktifkan pencatatan log berkelanjutan untuk tugas AWS Glue

1. Buka AWS Glue konsol di https://console.aws.amazon.com/glue/.

2. Di panel navigasi, pilih Tugas.
3. Pilih sebuah tugas yang ada dari daftar Tugas.
4. Pilih Tindakan, Edit tugas.
5. Di tab Job details, perluas bagian Advanced properties.
6. Di bawah Pencatatan berkelanjutan pilih Aktifkan log masuk CloudWatch.

Menggunakan AWS CLI

Untuk mengaktifkan pencatatan log berkelanjutan, Anda berikan parameter tugas untuk tugas AWS Glue. Lewati parameter pekerjaan khusus berikut yang mirip dengan parameter AWS Glue pekerjaan lainnya. Untuk informasi selengkapnya, lihat AWS Glueparameter pekerjaan.

```
'--enable-continuous-cloudwatch-log': 'true'
```

Anda dapat menentukan nama grup CloudWatch log Amazon kustom. Jika tidak ditentukan, nama grup log default-nya adalah `/aws-glue/jobs/logs-v2/`.

```
'--continuous-log-logGroup': 'custom_log_group_name'
```

Anda dapat menentukan awalan aliran CloudWatch log Amazon kustom. Jika tidak ditentukan, prefiks pengaliran log default-nya adalah ID eksekusi tugas.

```
'--continuous-log-logStreamPrefix': 'custom_log_stream_prefix'
```

Anda dapat menentukan sebuah pola konversi pencatatan log berkelanjutan kustom. Jika tidak ditentukan, maka pola konversi default-nya adalah `%d{yy/MM/dd HH:mm:ss} %p %c{1}: %m %n`. Perhatikan bahwa pola konversi hanya berlaku untuk log driver dan log pelaksana. Itu tidak mempengaruhi bilah kemajuan AWS Glue.

```
'--continuous-log-conversionPattern': 'custom_log_conversion_pattern'
```

Mencatat pesan khusus aplikasi menggunakan pencatat skrip kustom

Anda dapat menggunakan pencatat AWS Glue untuk mencatat log pesan spesifik-aplikasi dalam skrip yang dikirim secara waktu nyata untuk pengaliran log driver.

Contoh berikut menunjukkan skrip Python.

```
from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)
logger = glueContext.get_logger()
logger.info("info message")
logger.warn("warn message")
logger.error("error message")
```

Contoh berikut menunjukkan skrip Scala.

```
import com.amazonaws.services.glue.log.GlueLogger

object GlueApp {
  def main(sysArgs: Array[String]) {
    val logger = new GlueLogger
    logger.info("info message")
    logger.warn("warn message")
    logger.error("error message")
  }
}
```

Mengaktifkan bilah kemajuan untuk menunjukkan kemajuan pekerjaan

AWS Glue menyediakan bilah kemajuan waktu nyata pada pengaliran log `JOB_RUN_ID-progress-bar` untuk memeriksa status eksekusi tugas AWS Glue. Saat ini hanya mendukung tugas yang menginisialisasi `glueContext`. Jika Anda menjalankan tugas Spark murni tanpa menginisialisasi `glueContext`, maka bilah kemajuan AWS Glue tidak muncul.

Bilah kemajuan tersebut menunjukkan pembaruan kemajuan berikut setiap 5 detik.

```
Stage Number (Stage Name): > (numCompletedTasks + numActiveTasks) /
 totalNumOfTasksInThisStage]
```

Konfigurasi keamanan dengan pencatatan terus menerus

Jika konfigurasi keamanan diaktifkan untuk CloudWatch log, AWS Glue akan membuat grup log bernama sebagai berikut untuk log berkelanjutan:

```
<Log-Group-Name>-<Security-Configuration-Name>
```

Grup log default dan kustom adalah sebagai berikut:

- Grup pencatatan log berkelanjutan default adalah `/aws-glue/jobs/logs-v2-<Security-Configuration-Name>`
- Grup pencatatan log berkelanjutan kustom adalah `<custom-log-group-name>-<Security-Configuration-Name>`

Anda perlu menambahkan izin peran IAM Anda, jika Anda mengaktifkan konfigurasi keamanan dengan CloudWatch Log. `logs:AssociateKmsKey` Jika izin tersebut tidak disertakan, maka pencatatan log berkelanjutan akan dinonaktifkan. Selain itu, untuk mengonfigurasi enkripsi untuk CloudWatch Log, ikuti petunjuk di Enkripsi Data Log di CloudWatch Log Menggunakan AWS Key Management Service di Panduan Pengguna Amazon CloudWatch Logs.

Untuk informasi selengkapnya tentang cara membuat konfigurasi keamanan, lihat Bekerja dengan konfigurasi keamanan di konsol AWS Glue.

Melihat logging berkelanjutan untuk AWS Glue pekerjaan

Anda dapat melihat log waktu nyata menggunakan AWS Glue konsol atau CloudWatch konsol Amazon.

Untuk melihat log waktu nyata menggunakan dasbor konsol AWS Glue

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pilih Tugas.
3. Menambahkan atau memulai tugas yang sudah ada. Pilih Tindakan, Jalankan tugas.

   Ketika Anda mulai menjalankan tugas, Anda akan membuka halaman yang berisi informasi tentang tugas yang sedang berjalan:

   - Tab Log menunjukkan log aplikasi agregat yang lebih tua.
   - Tab Pencatatan log berkelanjutan menampilkan bilah kemajuan waktu nyata ketika tugas sedang berjalan dengan `glueContext` yang diinisialisasi.
   - Tab Pencatatan log berkelanjutan juga berisi Log driver, yang menangkap log driver Apache Spark secara waktu nyata, dan log aplikasi dari skrip yang di-log dengan menggunakan aplikasi pencatat log AWS Glue saat tugas berjalan.

4. Untuk tugas lama, Anda juga dapat melihat log waktu nyata pada tampilan Riwayat Tugas dengan memilih Log. Tindakan ini membawa Anda ke CloudWatch konsol yang menampilkan semua aliran log driver, eksekutor, dan bilah kemajuan Spark untuk menjalankan pekerjaan tersebut.

Untuk melihat log waktu nyata menggunakan dasbor CloudWatch konsol

1. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
2. Di panel navigasi, pilih Log.
3. Pilih grup log /aws-glue/jobs/logs-v2/.
4. Di kotak Filter, tempel ID eksekusi tugas.

   Anda dapat melihat log driver, log pelaksana, dan bilah kemajuan (jika menggunakan Filter standar).

Pemantauan dengan metrik AWS Glue Observabilitas

> **Note**
>
> AWS GlueMetrik observabilitas tersedia di AWS Glue versi 4.0 dan yang lebih baru.

Gunakan metrik AWS Glue Observability untuk menghasilkan wawasan tentang apa yang terjadi di dalam pekerjaan Apache Spark Anda AWS Glue untuk meningkatkan triaging dan analisis masalah. Metrik observabilitas divisualisasikan melalui Amazon CloudWatch dasbor dan dapat digunakan untuk membantu melakukan analisis akar penyebab kesalahan dan untuk mendiagnosis kemacetan kinerja. Anda dapat mengurangi waktu yang dihabiskan masalah debugging dalam skala besar sehingga Anda dapat fokus pada penyelesaian masalah lebih cepat dan lebih efektif.

AWS GlueObservabilitas menyediakan Amazon CloudWatch metrik yang dikategorikan dalam empat kelompok berikut:

- Keandalan (yaitu, Kelas Kesalahan) — dengan mudah mengidentifikasi alasan kegagalan yang paling umum pada rentang waktu tertentu yang mungkin ingin Anda atasi.
- Kinerja (yaitu, Skewness) - mengidentifikasi hambatan kinerja dan menerapkan teknik penyetelan. Misalnya, ketika Anda mengalami penurunan kinerja karena kemiringan pekerjaan, Anda mungkin ingin mengaktifkan Eksekusi Kueri Adaptif Spark dan menyempurnakan ambang gabungan miring.

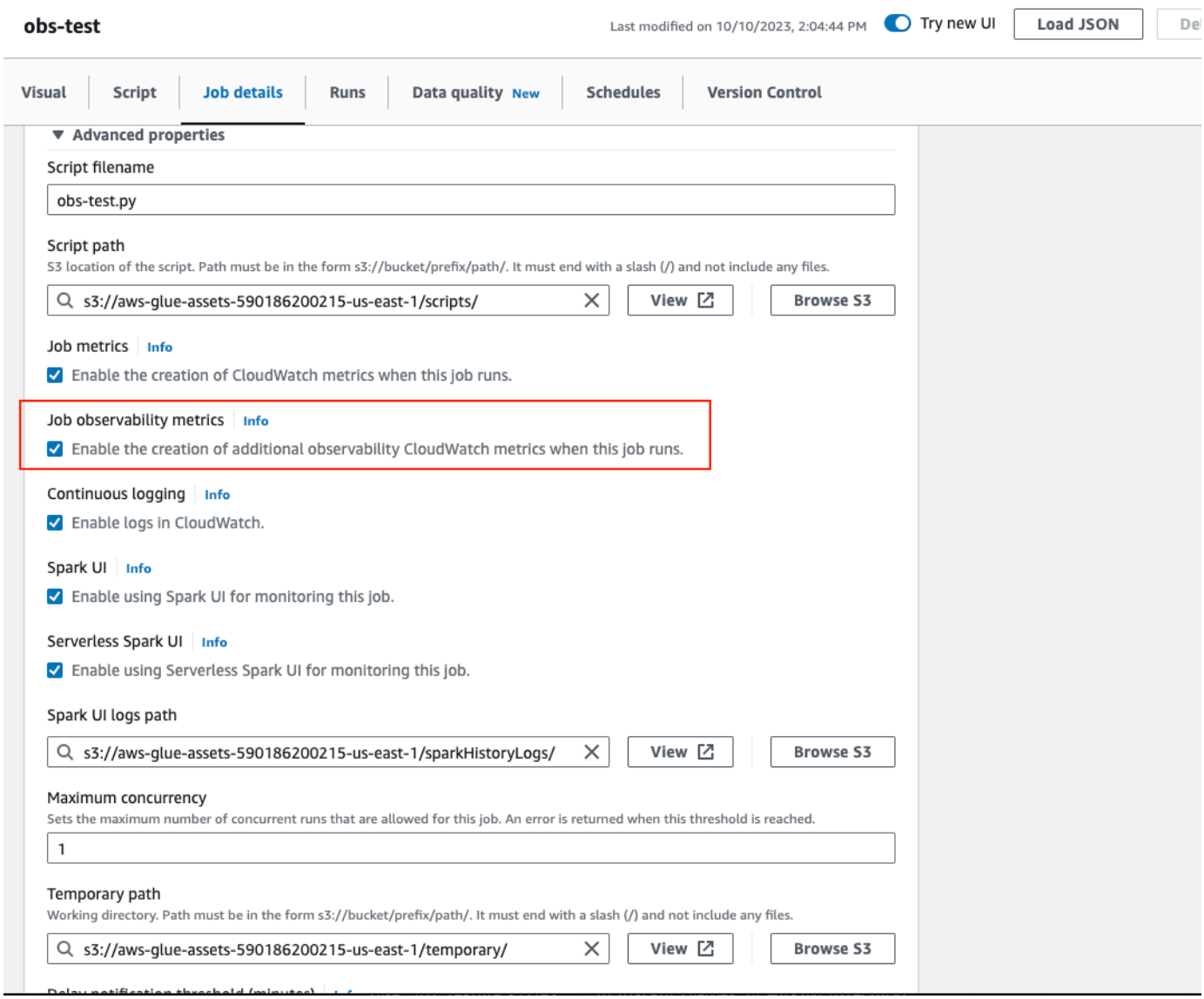
- Throughput (yaitu, per sumber/sink throughput) — memantau tren pembacaan dan penulisan data. Anda juga dapat mengonfigurasi Amazon CloudWatch alarm untuk anomali.
- Pemanfaatan Sumber Daya (yaitu, pekerja, memori dan pemanfaatan disk) — secara efisien menemukan pekerjaan dengan pemanfaatan kapasitas rendah. Anda mungkin ingin mengaktifkan AWS Glue auto-scaling untuk pekerjaan tersebut.

Memulai dengan metrik AWS Glue Observability

> **Note**
>
> Metrik baru diaktifkan secara default di AWS Glue Studio konsol.

Untuk mengonfigurasi metrik observabilitas di: AWS Glue Studio

1. Masuk ke AWS Glue konsol dan pilih pekerjaan ETL dari menu konsol.
2. Pilih pekerjaan dengan mengklik nama pekerjaan di bagian Pekerjaan Anda.
3. Pilih tab Detail tugas.
4. Gulir ke bawah dan pilih Advanced properties, lalu Job observability metrics.



Untuk mengaktifkan metrik AWS Glue Observability menggunakan: AWS CLI

- Tambahkan ke `--default-arguments` peta kunci-nilai berikut dalam file JSON masukan:

```
--enable-observability-metrics, true
```

Menggunakan AWS Glue observabilitas

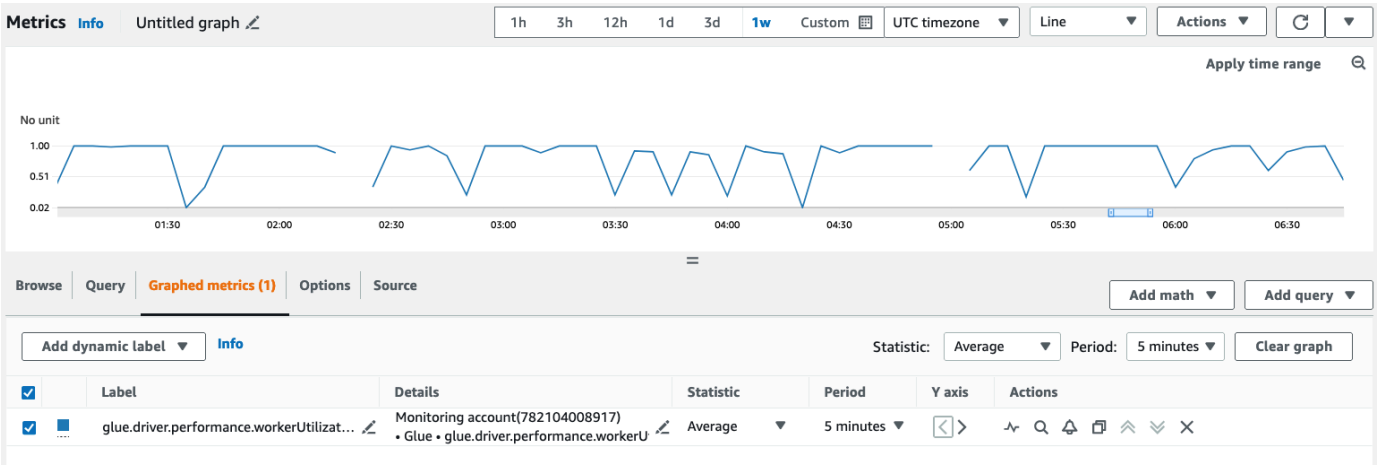
Karena metrik AWS Glue observabilitas disediakan Amazon CloudWatch, Anda dapat menggunakan Amazon CloudWatch konsol, SDK AWS CLI, atau API untuk menanyakan titik data metrik

observabilitas. Lihat Menggunakan Glue Observability untuk memantau pemanfaatan sumber daya guna mengurangi biaya untuk contoh kasus penggunaan saat menggunakan metrik AWS Glue observabilitas.

Menggunakan AWS Glue observabilitas di konsol Amazon CloudWatch

Untuk menanyakan dan memvisualisasikan metrik di konsol: Amazon CloudWatch

1. Buka Amazon CloudWatch konsol dan pilih Semua metrik.
2. Di bawah ruang nama khusus, pilih. AWS Glue
3. Pilih Metrik Observabilitas Pekerjaan, Metrik Observabilitas Per Sumber, atau Metrik Observabilitas Per Wastafel.
4. Cari nama metrik tertentu, nama pekerjaan, ID jalankan pekerjaan, dan pilih.
5. Di bawah tab Metrik grafik, konfigurasikan statistik pilihan Anda, periode, dan opsi lainnya.



Untuk menanyakan metrik Observabilitas menggunakan AWS CLI:

1. Buat file JSON definisi metrik dan ganti `your-Glue-job-name` dan `your-Glue-job-run-id` dengan milik Anda.

```
$ cat multiplequeries.json
[
    {
        "Id": "avgWorkerUtil_0",
        "MetricStat": {
            "Metric": {
                "Namespace": "Glue",
                "MetricName": "glue.driver.workerUtilization",
```

```
                "Dimensions": [
                    {
                        "Name": "JobName",
                        "Value": "<your-Glue-job-name-A>"
                    },
                    {
                        "Name": "JobRunId",
                        "Value": "<your-Glue-job-run-id-A>"
                    },
                    {
                        "Name": "Type",
                        "Value": "gauge"
                    },
                    {
                        "Name": "ObservabilityGroup",
                        "Value": "resource_utilization"
                    }
                ]
            },
            "Period": 1800,
            "Stat": "Minimum",
            "Unit": "None"
        }
    },
    {
        "Id": "avgWorkerUtil_1",
        "MetricStat": {
            "Metric": {
                "Namespace": "Glue",
                "MetricName": "glue.driver.workerUtilization",
                "Dimensions": [
                    {
                        "Name": "JobName",
                        "Value": "<your-Glue-job-name-B>"
                    },
                    {
                        "Name": "JobRunId",
                        "Value": "<your-Glue-job-run-id-B>"
                    },
                    {
                        "Name": "Type",
                        "Value": "gauge"
                    },
                    {
```

```
                        "Name": "ObservabilityGroup",
                        "Value": "resource_utilization"
                    }
                ]
            },
            "Period": 1800,
            "Stat": "Minimum",
            "Unit": "None"
        }
    }
]
```

2. Jalankan perintah `get-metric-data`:

```
$ aws cloudwatch get-metric-data --metric-data-queries file: //multiplequeries.json
 \
     --start-time '2023-10-28T18: 20' \
     --end-time '2023-10-28T19: 10'  \
     --region us-east-1
{
    "MetricDataResults": [
        {
            "Id": "avgWorkerUtil_0",
            "Label": "<your-label-for-A>",
            "Timestamps": [
                "2023-10-28T18:20:00+00:00"
            ],
            "Values": [
                0.06718750000000001
            ],
            "StatusCode": "Complete"
        },
        {
            "Id": "avgWorkerUtil_1",
            "Label": "<your-label-for-B>",
            "Timestamps": [
                "2023-10-28T18:50:00+00:00"
            ],
            "Values": [
                0.5959183673469387
            ],
            "StatusCode": "Complete"
```

```
        }
    ],
    "Messages": []
}
```

Metrik observabilitas

AWS GlueProfil observabilitas dan mengirimkan metrik berikut ke Amazon CloudWatch setiap 30 detik, dan beberapa metrik ini dapat terlihat di Halaman Pemantauan AWS Glue Studio Job Runs.

Metrik	Deskripsi	Kategori
lem.driver.skewness.stage	Kategori Metrik: job_performance Spark stage eksekusi Skewness: metrik ini menangkap kecondongan eksekusi, yang mungkin disebabkan oleh kemiringan data input atau oleh transformasi (mis., Skewed join). Nilai metrik ini jatuh ke dalam kisaran [0, tak terhingga [, di mana 0 berarti rasio waktu eksekusi tugas maksimum terhadap median, di antara semua tugas dalam tahap kurang dari faktor kemiringan tahap tertentu. Faktor kemiringan tahap default adalah `5` dan ditimpa melalui spark conf: spark.metrics.conf.driver.source.glue.jobPerformance.SkewnessFactor	job_kinerja

Metrik	Deskripsi	Kategori
	Nilai kemiringan tahap 1 berarti rasio dua kali faktor kemiringan tahap. Nilai stage skewnewss diperbarui setiap 30 detik untuk mencerminkan kemiringan saat ini. Nilai pada akhir tahap mencerminkan kemiringan tahap akhir. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (job_performance) Statistik yang Valid: Rata-rata, Maksimum, Minimum, Persentil Unit: Jumlah	

Metrik	Deskripsi	Kategori
lem.driver.skewness.job	Kategori Metrik: job_perfo rmance Job skewness adalah rata-rata tertimbang dari kemiringa n tahap pekerjaan. Rata-rata tertimbang memberi bobot lebih pada tahapan yang membutuhkan waktu lebih lama untuk dieksekusi. Ini untuk menghindari kasus sudut ketika tahap yang sangat miring sebenarnya berjalan untuk waktu yang sangat singkat relatif terhadap tahap lain (dan dengan demikian kemiringannya tidak signifikan untuk perforamnce pekerjaan secara keseluruh an dan tidak sepadan dengan upaya untuk mencoba mengatasi kemiringannya). Metrik ini diperbarui setelah menyelesaikan setiap tahap, dan dengan demikian nilai terakhir mencermin kan kemiringan pekerjaan keseluruhan yang sebenarnya. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan	job_kinerja

Metrik	Deskripsi	Kategori
	ObservabilityGroup (job_performance) Statistik yang Valid: Rata-rata, Maksimum, Minimum, Persentil Unit: Jumlah	
lem.berhasil.semua	Kategori Metrik: kesalahan Jumlah total pekerjaan yang berhasil berjalan, untuk melengkapi gambaran kategori kegagalan Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (count), dan ObservabilityGroup (error) Statistik yang Valid: SUM Unit: Jumlah	kesalahan

Metrik	Deskripsi	Kategori
lem.error.all	Kategori Metrik: kesalahan Jumlah total kesalahan menjalankan pekerjaan, untuk melengkapi gambaran kategori kegagalan Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (count), dan ObservabilityGroup (error) Statistik yang Valid: SUM Unit: Jumlah	kesalahan

Metrik	Deskripsi	Kategori
lem.error. [kategori kesalahan]	Kategori Metrik: kesalahan Ini sebenarnya adalah satu set metrik, yang diperbarui hanya ketika pekerjaan berjalan gagal. Kategorisasi kesalahan membantu dengan triaging dan debugging. Ketika pekerjaan berjalan gagal, kesalahan yang menyebabkan kegagalan dikategorikan dan metrik kategori kesalahan yang sesuai disetel ke 1. Ini membantu untuk melakukan analisis kegagalan dari waktu ke waktu, serta atas semua analisis kesalahan pekerjaan untuk mengidentifikasi kategori kegagalan yang paling umum untuk mulai mengatasinya. AWS Gluememiliki 28 kategori kesalahan, termasuk kategori kesalahan OUT_OF_MEMORY (driver dan eksekutor), PERMISSION, SYNTAX dan THROTTLING. Kategori kesalahan juga mencakup kategori kesalahan KOMPILASI, PELUNCURAN, dan TIMEOUT. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau	kesalahan

Metrik	Deskripsi	Kategori
	ALL), Type (count), dan ObservabilityGroup (error) Statistik yang Valid: SUM Unit: Jumlah	
Glue.driver.workerUtilization	Kategori Metrik: resource_ utilization Persentase pekerja yang dialokasikan yang benar-benar digunakan. Jika tidak bagus, penskalaan otomatis dapat membantu. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata, Maksimum, Minimum, Persentil Unit: Persentase	resource_utilization

Metrik	Deskripsi	Kategori
lem.driver.memory.heap. [tersedia | digunakan]	Kategori Metrik: resource_ utilization Memori heap driver yang tersedia/digunakan selama pekerjaan dijalankan. Ini membantu untuk memahami tren penggunaan memori, terutama dari waktu ke waktu, yang dapat membantu menghindari potensi kegagalan, selain men-debug kegagalan terkait memori. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Unit: Bita	resource_utilization

Metrik	Deskripsi	Kategori
lem.driver.memory.heap.used .percentage	Kategori Metrik: resource_ utilization Pengemudi menggunakan memori heap (%) selama pekerjaan dijalankan. Ini membantu untuk memahami tren penggunaan memori, terutama dari waktu ke waktu, yang dapat membantu menghindari potensi kegagalan, selain men-debug kegagalan terkait memori. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Unit: Persentase	resource_utilization

Metrik	Deskripsi	Kategori
lem.driver.memory.non-heap. [tersedia | digunakan]	Kategori Metrik: resource_ utilization Pengemudi yang tersedia/ menggunakan memori non-heap selama pekerjaan dijalankan. Ini membantu untuk memahami tren penggunaan memori, terutama dari waktu ke waktu, yang dapat membantu menghinda ri potensi kegagalan, selain men-debug kegagalan terkait memori. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Unit: Bita	resource_utilization

Metrik	Deskripsi	Kategori
lem.driver.memory.non-heap.used.percentage	Kategori Metrik: resource_utilization Pengemudi menggunakan memori non-heap (%) selama pekerjaan dijalankan. Ini membantu untuk memahami tren penggunaan memori, terutama dari waktu ke waktu, yang dapat membantu menghindari potensi kegagalan, selain men-debug kegagalan terkait memori. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource_utilization) Statistik yang Valid: Rata-rata Unit: Persentase	resource_utilization

Metrik	Deskripsi	Kategori
lem.driver.memory.total. [tersedia | digunakan]	Kategori Metrik: resource_ utilization Pengemudi tersedia/ menggunakan memori total selama pekerjaan dijalanka n. Ini membantu untuk memahami tren penggunaa n memori, terutama dari waktu ke waktu, yang dapat membantu menghindari potensi kegagalan, selain men-debug kegagalan terkait memori. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Unit: Bita	resource_utilization

Metrik	Deskripsi	Kategori
lem.driver.memory.total.used.percentage	Kategori Metrik: resource_utilization Pengemudi menggunakan (%) total memori selama menjalankan pekerjaan . Ini membantu untuk memahami tren penggunaan memori, terutama dari waktu ke waktu, yang dapat membantu menghindari potensi kegagalan, selain men-debug kegagalan terkait memori. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource_utilization) Statistik yang Valid: Rata-rata Unit: Persentase	resource_utilization

Metrik	Deskripsi	Kategori
Lek.all.memory.heap. [tersedia | digunakan]	Kategori Metrik: resource_ utilization Memori heap eksekutor yang tersedia/digunakan. SEMUA berarti semua pelaksana. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Unit: Bita	resource_utilization
Glue.all.memory.heap.used.p ercentage	Kategori Metrik: resource_ utilization Memori heap (%) yang digunakan para eksekutor . SEMUA berarti semua pelaksana. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Unit: Persentase	resource_utilization

Metrik	Deskripsi	Kategori
Lek.all.memory.non-heap. [tersedia | digunakan]	Kategori Metrik: resource_ utilization Memori non-heap eksekutor yang tersedia/digunakan . SEMUA berarti semua pelaksana. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Unit: Bita	resource_utilization
Glue.all.memory.non-heap.us ed.percentage	Kategori Metrik: resource_ utilization Eksekutor menggunakan (%) memori non-heap. SEMUA berarti semua pelaksana. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Unit: Persentase	resource_utilization

Metrik	Deskripsi	Kategori
Lek.all.memory.total. [tersedia | digunakan]	Kategori Metrik: resource_ utilization Memori total yang tersedia/ digunakan eksekutor. SEMUA berarti semua pelaksana. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Unit: Bita	resource_utilization
Glue.all.memory.total.used. percentage	Kategori Metrik: resource_ utilization Eksekutor menggunakan (%) total memori. SEMUA berarti semua pelaksana. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Unit: Persentase	resource_utilization

Metrik	Deskripsi	Kategori
lem.driver.disk. [tersedia_GB | digunakan_GB]	Kategori Metrik: resource_ utilization Ruang disk yang tersedia/ digunakan pengemudi selama menjalankan pekerjaan. Ini membantu untuk memahami tren penggunaan disk, terutama dari waktu ke waktu, yang dapat membantu menghindari potensi kegagalan, selain debugging tidak cukup kegagalan terkait ruang disk. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Satuan: Gigabytes	resource_utilization

Metrik	Deskripsi	Kategori
lem.driver.disk.used.percentage]	Kategori Metrik: resource_utilization Ruang disk yang tersedia/digunakan pengemudi selama menjalankan pekerjaan. Ini membantu untuk memahami tren penggunaan disk, terutama dari waktu ke waktu, yang dapat membantu menghindari potensi kegagalan, selain debugging tidak cukup kegagalan terkait ruang disk. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource_utilization) Statistik yang Valid: Rata-rata Unit: Persentase	resource_utilization

Metrik	Deskripsi	Kategori
Lek.all.disk. [tersedia_GB | digunakan_GB]	Kategori Metrik: resource_ utilization Ruang disk yang tersedia/ digunakan para eksekutor . SEMUA berarti semua pelaksana. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Satuan: Gigabytes	resource_utilization
Glue.all.disk.used.percentage	Kategori Metrik: resource_ utilization Ruang disk yang tersedia/ digunakan/digunakan (%) eksekutor. SEMUA berarti semua pelaksana. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), dan ObservabilityGroup (resource _utilization) Statistik yang Valid: Rata-rata Unit: Persentase	resource_utilization

Metrik	Deskripsi	Kategori
Lek.driver.bytesRead	Kategori Metrik: throughput Jumlah byte yang dibaca per sumber input dalam pekerjaan ini dijalankan, serta untuk SEMUA sumber. Ini membantu memahami volume data dan perubahan nya dari waktu ke waktu, yang membantu mengatasi masalah seperti kemiringan data. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), (resource _utilization), dan Source ObservabilityGroup (lokasi data sumber) Statistik yang Valid: Rata-rata Unit: Bita	throughput

Metrik	Deskripsi	Kategori
lem.driver. [RecordsRead | FilesRead]	Kategori Metrik: throughput Jumlah rekaman/file yang dibaca per sumber input dalam pekerjaan ini dijalanka n, serta untuk SEMUA sumber. Ini membantu memahami volume data dan perubahannya dari waktu ke waktu, yang membantu mengatasi masalah seperti kemiringan data. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), (resource _utilization), dan Source ObservabilityGroup (lokasi data sumber) Statistik yang Valid: Rata-rata Unit: Jumlah	throughput

Metrik	Deskripsi	Kategori
lem.driver.partitionsRead	Kategori Metrik: throughput Jumlah partisi yang dibaca per sumber input Amazon S3 dalam pekerjaan ini dijalanka n, serta untuk SEMUA sumber. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), (resource _utilization), dan Source ObservabilityGroup (lokasi data sumber) Statistik yang Valid: Rata-rata Unit: Jumlah	throughput

Metrik	Deskripsi	Kategori
Lek.driver.bytesWrittten	Kategori Metrik: throughput Jumlah byte yang ditulis per output tenggelam dalam pekerjaan ini, serta untuk SEMUA sink. Ini membantu memahami volume data dan bagaimana perkembangannya dari waktu ke waktu, yang membantu mengatasi masalah seperti kemiringan pemrosesan. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), ObservabilityGroup (resource_utilization), dan Sink (lokasi data sink) Statistik yang Valid: Rata-rata Unit: Bita	throughput

Metrik	Deskripsi	Kategori
lem.driver. [RecordsWritten | FilesWritten]	Kategori Metrik: throughput Jumlah catatan/file yang ditulis per output tenggelam dalam pekerjaan ini, serta untuk SEMUA sink. Ini membantu memahami volume data dan bagaimana perkembangannya dari waktu ke waktu, yang membantu mengatasi masalah seperti kemiringan pemrosesan. Dimensi yang valid: JobName (nama AWS Glue Job), JobRunId (JobRun ID. atau ALL), Type (gauge), ObservabilityGroup (resource_utilization), dan Sink (lokasi data sink) Statistik yang Valid: Rata-rata Unit: Jumlah	throughput

Kategori kesalahan

Kategori kesalahan	Deskripsi
COMPILATION_ERROR	Kesalahan muncul selama kompilasi kode Scala.
CONNECTION_ERROR	Kesalahan muncul saat menghubungkan ke layanan/layanan host/database jarak jauh, dll.

Kategori kesalahan	Deskripsi
DISK_NO_SPACE_ERROR	Kesalahan muncul ketika tidak ada ruang tersisa di disk pada driver/executor.
OUT_OF_MEMORY_ERROR	Kesalahan muncul ketika tidak ada ruang yang tersisa di memori pada driver/executor.
IMPORT_ERROR	Kesalahan muncul saat dependensi impor.
INVALID_ARGUMENT_ERROR	Kesalahan muncul ketika argumen input tidak valid/ilegal.
PERMISSION_ERROR	Kesalahan muncul ketika tidak memiliki izin untuk layanan, data, dll.
RESOURCE_NOT_FOUND_ERROR	Kesalahan muncul ketika data, lokasi, dll tidak keluar.
QUERY_ERROR	Kesalahan muncul dari eksekusi kueri Spark SQL.
SYNTAX_ERROR	Kesalahan muncul ketika ada kesalahan sintaks dalam skrip.
THROTTLING_ERROR	Kesalahan muncul saat menekan batasan konkurensi layanan atau mengeksekusi batas kuota layanan.
DATA_LAKE_FRAMEWORK_ERROR	Kesalahan muncul dari kerangka data lake yang AWS Glue didukung asli seperti Hudi, Iceberg, dll.
UNSUPPORTED_OPERATION_ERROR	Kesalahan muncul saat melakukan operasi yang tidak didukung.
RESOURCES_ALREADY_EXISTS_ERROR	Kesalahan muncul ketika sumber daya yang akan dibuat atau ditambahkan sudah ada.

Kategori kesalahan	Deskripsi
GLUE_INTERNAL_SERVICE_ERROR	Kesalahan muncul ketika ada masalah layanan AWS Glue internal.
GLUE_OPERATION_TIMEOUT_ERROR	Kesalahan muncul saat AWS Glue operasi habis waktu.
GLUE_VALIDATION_ERROR	Kesalahan muncul ketika nilai yang diperluka n tidak dapat divalidasi untuk AWS Glue pekerjaan.
GLUE_JOB_BOOKMARK_VERSION_M ISMATCH_ERROR	Kesalahan muncul ketika pekerjaan yang sama menunjukkan keranjang sumber yang sama dan menulis ke tujuan yang sama/berbeda secara bersamaan (konkurensi> 1)
LAUNCH_ERROR	Kesalahan muncul selama fase peluncuran AWS Glue pekerjaan.
DYNAMODB_ERROR	Kesalahan umum muncul dari Amazon DynamoDB layanan.
GLUE_ERROR	Kesalahan umum muncul dari AWS Glue layanan.
LAKEFORMATION_ERROR	Kesalahan umum muncul dari AWS Lake Formation layanan.
REDSHIFT_ERROR	Kesalahan umum muncul dari Amazon Redshift layanan.
S3_KESALAHAN	Kesalahan Umum muncul dari layanan Amazon S3.
SYSTEM_EXIT_ERROR	Kesalahan keluar sistem generik.
TIMEOUT_ERROR	Kesalahan umum muncul ketika pekerjaan gagal karena waktu operasi habis.

Kategori kesalahan	Deskripsi
UNCLASSIFIED_SPARK_ERROR	Kesalahan umum muncul dari Spark.
UNCLASSIFIED_ERROR	Kategori kesalahan default.

Batasan

> **Note**
>
> `glueContext`harus diinisialisasi untuk mempublikasikan metrik.

Di Dimensi Sumber, nilainya adalah jalur Amazon S3 atau nama tabel, tergantung pada jenis sumbernya. Selain itu, jika sumbernya adalah JDBC dan opsi kueri digunakan, string kueri diatur dalam dimensi sumber. Jika nilainya lebih panjang dari 500 karakter, itu dipangkas dalam 500 karakter.Berikut ini adalah batasan dalam nilai:

- Karakter non-ASCII akan dihapus.
- <non-ASCII input>Jika nama sumber tidak mengandung karakter ASCII, itu dikonversi ke.

Keterbatasan dan pertimbangan untuk metrik throughput

- DataFrame dan DataFrame berbasis DynamicFrame (misalnya JDBC, membaca dari parket di Amazon S3) didukung, namun, berbasis RDD DynamicFrame (misalnya membaca csv, json di Amazon S3, dll.) tidak didukung. Secara teknis, semua membaca dan menulis yang terlihat di Spark UI didukung.
- `recordsRead`Metrik akan dipancarkan jika sumber data adalah tabel katalog dan formatnya adalah JSON, CSV, teks, atau Iceberg.
- `glue.driver.throughput.recordsWritten`,`glue.driver.throughput.bytesWritten`, dan `glue.driver.throughput.filesWritten` metrik tidak tersedia dalam tabel JDBC dan Iceberg.
- Metrik mungkin tertunda. Jika pekerjaan selesai dalam waktu sekitar satu menit, mungkin tidak ada metrik throughput di Metrik. Amazon CloudWatch

Pemantauan dan debugging Job

Anda dapat mengumpulkan metrik tentang AWS Glue pekerjaan dan memvisualisasikannya di CloudWatch konsol Amazon AWS Glue dan Amazon untuk mengidentifikasi dan memperbaiki masalah. Membuat profil tugas AWS Glue memerlukan langkah-langkah berikut:

1. Aktifkan metrik:

   a. Mengaktifkan opsi Metrik Tugas dalam definisi tugas. Anda dapat mengaktifkan pemrofilan di konsol AWS Glue atau sebagai sebuah parameter untuk tugas. Untuk informasi selengkapnya, lihat Mendefinisikan properti pekerjaan untuk pekerjaan Spark atau AWS Glueparameter pekerjaan.

   b. Aktifkan opsi metrik AWS Glue Observabilitas dalam definisi pekerjaan. Anda dapat mengaktifkan Observabilitas di AWS Glue konsol atau sebagai parameter untuk pekerjaan. Untuk informasi selengkapnya, lihat Pemantauan dengan metrik AWS Glue Observabilitas.

2. Mengonfirmasi bahwa skrip tugas menginisialisasi sebuah `GlueContext`. Sebagai contoh, potongan skrip berikut menginisialisasi `GlueContext` dan menunjukkan di mana kode yang diprofilkan ditempatkan dalam skrip tersebut. Format umum ini digunakan dalam skenario debugging yang mengikuti.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
import time

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)

...
...
code-to-profile
```

```
...
...


job.commit()
```

3. Jalankan tugas.
4. Visualisasikan metrik:
   a. Visualisasikan metrik pekerjaan di AWS Glue konsol dan identifikasi metrik abnormal untuk driver atau pelaksana.
   b. Periksa metrik observabilitas di halaman pemantauan Job run, halaman detail menjalankan pekerjaan, atau di Amazon. CloudWatch Untuk informasi selengkapnya, lihat Pemantauan dengan metrik AWS Glue Observabilitas.
5. Persempit akar masalah dengan menggunakan metrik yang diidentifikasi.
6. Opsional, mengonfirmasi akar masalah menggunakan pengaliran log dari driver atau pelaksana tugas yang diidentifikasi.

Gunakan kasus untuk AWS Glue metrik observabilitas

- Debugging pengecualian OOM dan kelainan pekerjaan
- Mendebug tahapan yang menuntut dan tugas yang menyimpang
- Memantau kemajuan beberapa pekerjaan
- Pemantauan perencanaan kapasitas DPU
- Menggunakan AWS Glue Observabilitas untuk memantau pemanfaatan sumber daya untuk mengurangi biaya

Debugging pengecualian OOM dan kelainan pekerjaan

Anda dapat men-debug pengecualian out-of-memory (OOM) dan kelainan pekerjaan di. AWS Glue Bagian berikut menjelaskan skenario untuk debugging out-of-memory pengecualian driver Apache Spark atau pelaksana Spark.

- Mendebug pengecualian OOM driver
- Mendebug pengecualian OOM eksekutor

Mendebug pengecualian OOM driver

Dalam skenario ini, sebuah tugas Spark membaca sejumlah besar file dalam ukuran kecil dari Amazon Simple Storage Service (Amazon S3). Ia mengkonversi file tersebut ke format Apache Parquet dan kemudian menulisnya ke Amazon S3. Driver Spark sedang kehabisan memori. Data input Amazon S3 memiliki lebih dari 1 juta file dalam partisi Amazon S3 yang berbeda.

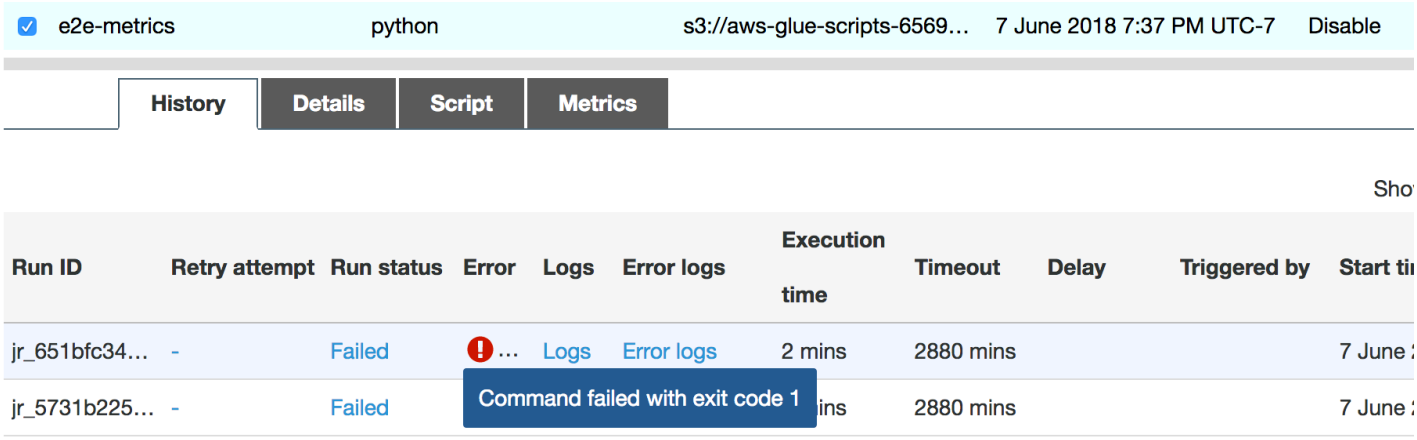
Kode profilannya adalah sebagai berikut:

```
data = spark.read.format("json").option("inferSchema", False).load("s3://input_path")
data.write.format("parquet").save(output_path)
```

Visualisasikan metrik yang diprofilkan di konsol AWS Glue

Grafik berikut menunjukkan penggunaan memori dalam satuan persentase untuk driver dan pelaksana. Penggunaan ini diplot sebagai satu titik data yang dirata-ratakan atas nilai-nilai yang dilaporkan di menit terakhir. Anda dapat melihat di profil memori dari tugas bahwa memori driver melewati ambang batas aman penggunaan 50 persen dengan cepat. Di sisi lain, penggunaan memori rata-rata di semua pelaksana masih kurang dari 4 persen. Hal ini jelas menunjukkan kelainan yang terjadi pada pelaksana eksekusi driver dalam tugas Spark ini.

Eksekusi tugas segera gagal, dan kesalahan berikut muncul di tab Riwayat pada Konsol AWS Glue: Perintah Gagal dengan Kode Keluar 1. String kesalahan ini berarti bahwa tugas gagal karena adanya sebuah kesalahan sistemik—yang dalam hal ini adalah bahwa driver kehabisan memori.



Di konsol, pilih tautan Kesalahan log pada tab Sejarah untuk mengonfirmasi temuan tentang driver OOM dari CloudWatch Log. Cari "**`Error`**" di log kesalahan tugas untuk mengonfirmasi bahwa hal itu memang pengecualian OOM yang membuat tugas gagal:

```
# java.lang.OutOfMemoryError: Java heap space
# -XX:OnOutOfMemoryError="kill -9 %p"
# Executing /bin/sh -c "kill -9 12039"...
```

Pada tab Riwayat untuk tugas, pilih Log. Anda dapat menemukan jejak eksekusi driver berikut di CloudWatch Log di awal pekerjaan. Driver Spark mencoba untuk mencantumkan semua file di semua direktori, membangun sebuah `InMemoryFileIndex`, dan meluncurkan satu tugas setiap file. Hal ini pada gilirannya akan mengakibatkan driver Spark harus mempertahankan sejumlah besar status di memori untuk melacak semua tugas. Ia meng-cache daftar lengkap dari sejumlah besar file untuk indeks dalam memori, yang mengakibatkan driver mengalami OOM.

Perbaiki pemrosesan beberapa file menggunakan pengelompokan

Anda dapat memperbaiki pemrosesan beberapa file dengan menggunakan fitur pengelompokan dalam grup di AWS Glue. Pengelompokan dalam grup secara otomatis diaktifkan ketika Anda menggunakan bingkai dinamis dan ketika set data masukan memiliki sejumlah besar file (lebih dari 50.000). Pengelompokan dalam grup memungkinkan Anda untuk menggabungkan beberapa file bersama-sama ke dalam sebuah grup, dan memungkinkan tugas untuk memproses seluruh grup tersebut, alih-alih satu file. Akibatnya, driver Spark menyimpan jauh lebih sedikit status di memori

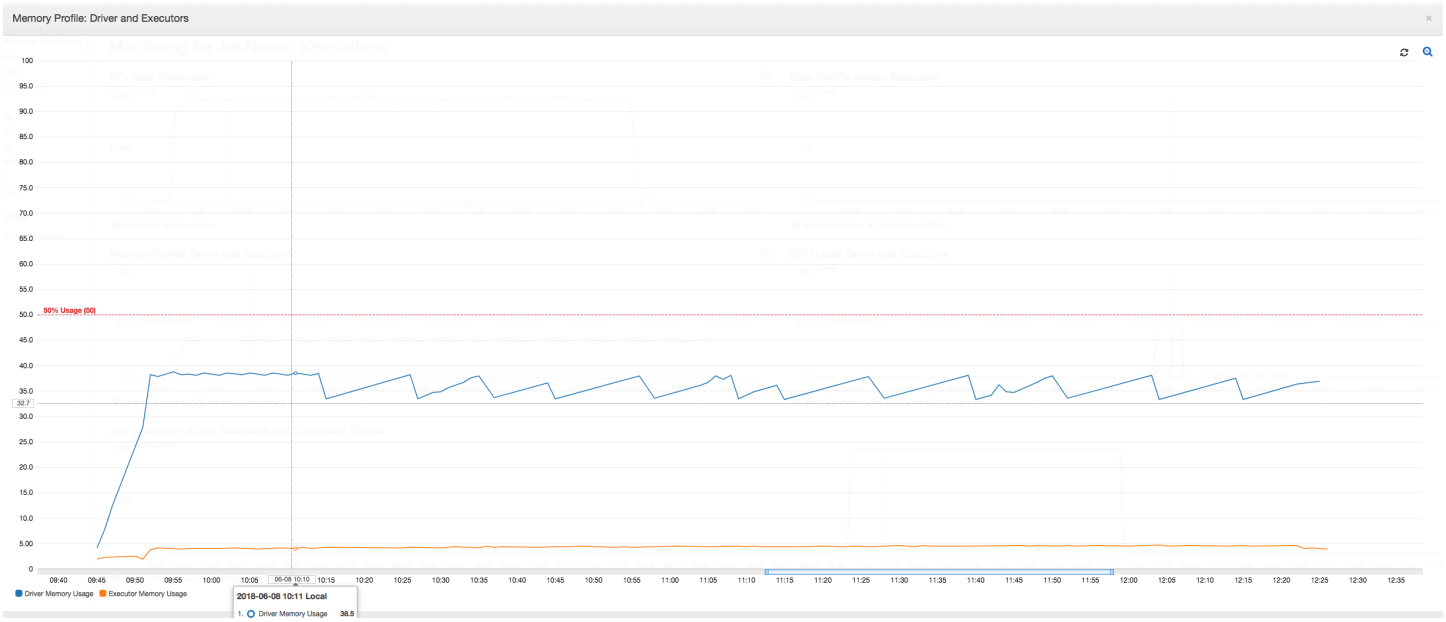
untuk melacak lebih sedikit tugas. Untuk informasi lebih lanjut tentang pengaktifan pengelompokan data dalam grup secara manual, lihat Membaca file input dalam kelompok yang lebih besar.

Untuk memeriksa profil memori tugas AWS Glue, lakukan pemrofilan pada kode berikut dengan pengelompokan grup diaktifkan:

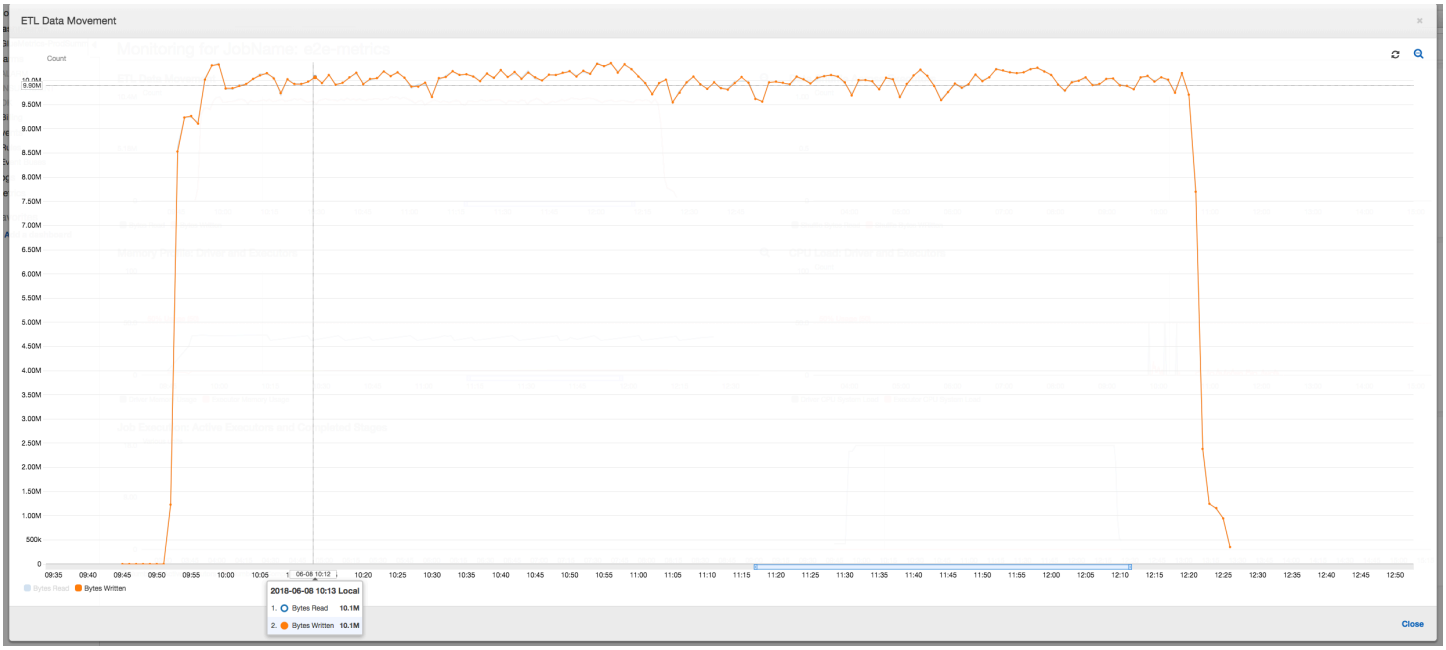
```
df = glueContext.create_dynamic_frame_from_options("s3", {'paths': ["s3://input_path"],
 "recurse":True, 'groupFiles': 'inPartition'}, format="json")
datasink = glueContext.write_dynamic_frame.from_options(frame = df, connection_type
 = "s3", connection_options = {"path": output_path}, format = "parquet",
 transformation_ctx = "datasink")
```

Anda dapat memantau profil memori dan pergerakan data ETL di profil tugas AWS Glue.

Driver berjalan di bawah ambang batas penggunaan memori 50 persen di sepanjang durasi tugas AWS Glue. Pelaksana mengalirkan data dari Amazon S3, memprosesnya, dan menuliskan data tersebut ke Amazon S3. Akibatnya, mereka menggunakan memori kurang dari 5 persen pada setiap titik waktu.



Profil pergerakan data di bawah ini menunjukkan jumlah byte Amazon S3 yang di-baca dan di-tulis pada menit terakhir oleh semua pelaksana saat tugas berlangsung. Keduanya mengikuti pola yang sama karena data dialirkan di semua pelaksana. Tugas tersebut menyelesaikan proses satu juta file dalam waktu kurang dari tiga jam.



Mendebug pengecualian OOM eksekutor

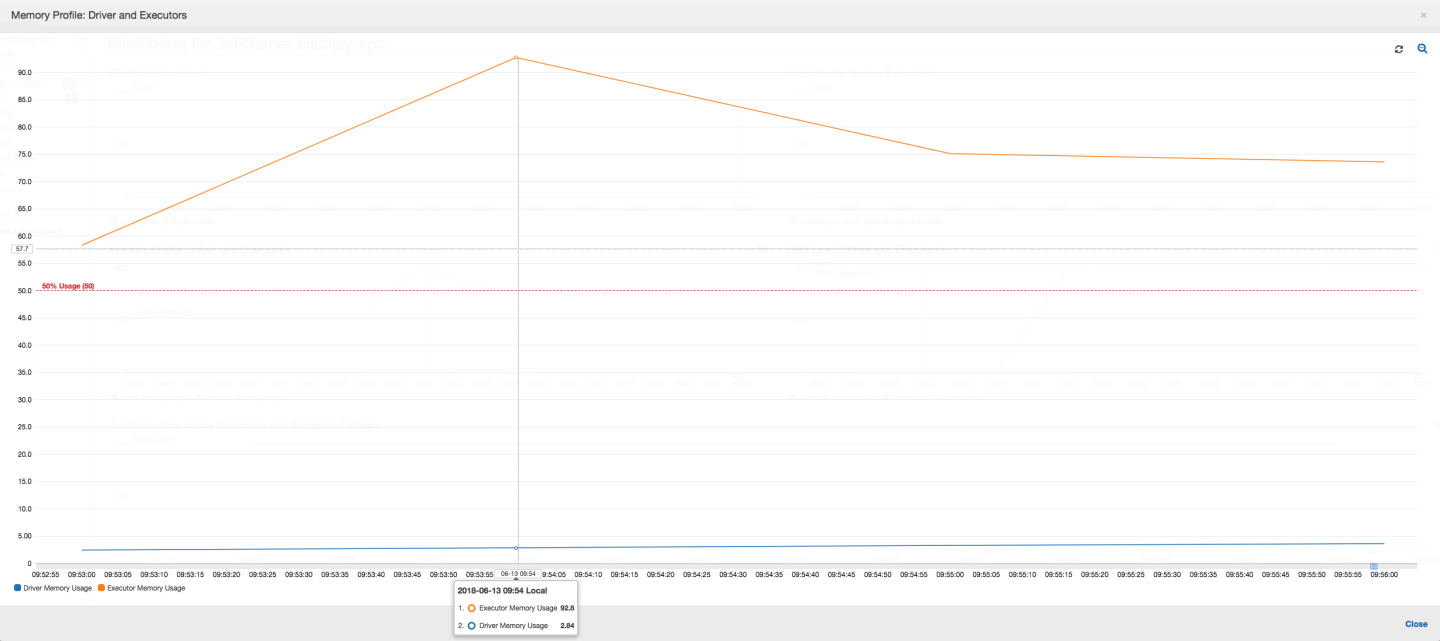
Dalam skenario ini, Anda dapat belajar bagaimana melakukan debug pada pengecualian OOM yang dapat terjadi dalam pelaksana Apache Spark. Kode berikut menggunakan pembaca Spark MySQL untuk membaca tabel besar yang mempunyai sekitar 34 juta baris ke dataframe Spark. Ia kemudian tulis tabel tersebut ke Amazon S3 dalam format Parquet. Anda dapat memberikan properti koneksi dan menggunakan konfigurasi Spark default untuk membaca tabel tersebut.

```
val connectionProperties = new Properties()
connectionProperties.put("user", user)
connectionProperties.put("password", password)
connectionProperties.put("Driver", "com.mysql.jdbc.Driver")
val sparkSession = glueContext.sparkSession
val dfSpark = sparkSession.read.jdbc(url, tableName, connectionProperties)
dfSpark.write.format("parquet").save(output_path)
```

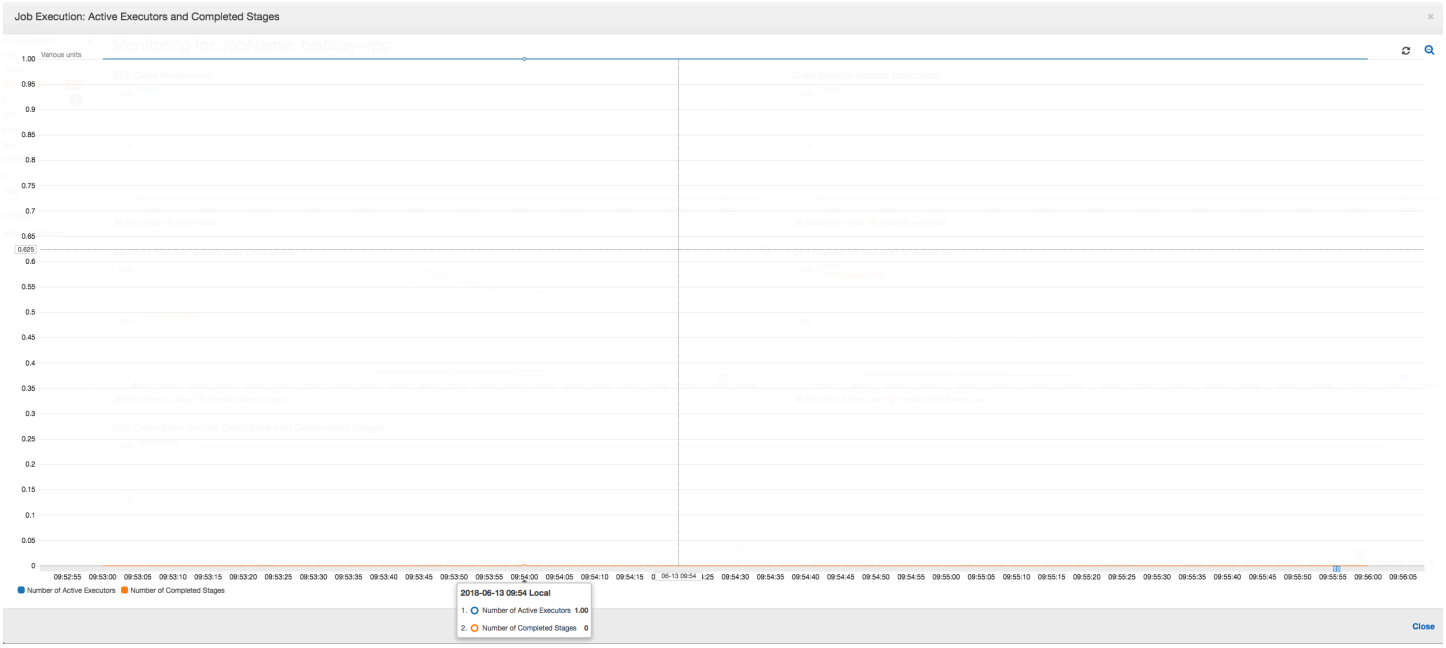
Visualisasikan metrik yang diprofilkan di konsol AWS Glue

Jika kemiringan grafik penggunaan memori positif dan melewati 50 persen, maka jika tugas gagal sebelum metrik berikutnya dipancarkan, maka kelelahan memori menjadi kandidat yang tepat sebagai penyebabnya. Grafik berikut menunjukkan bahwa dalam satu menit eksekusi, penggunaan memori rata-rata di semua pelaksana melonjak dengan cepat di atas 50 persen. Penggunaan

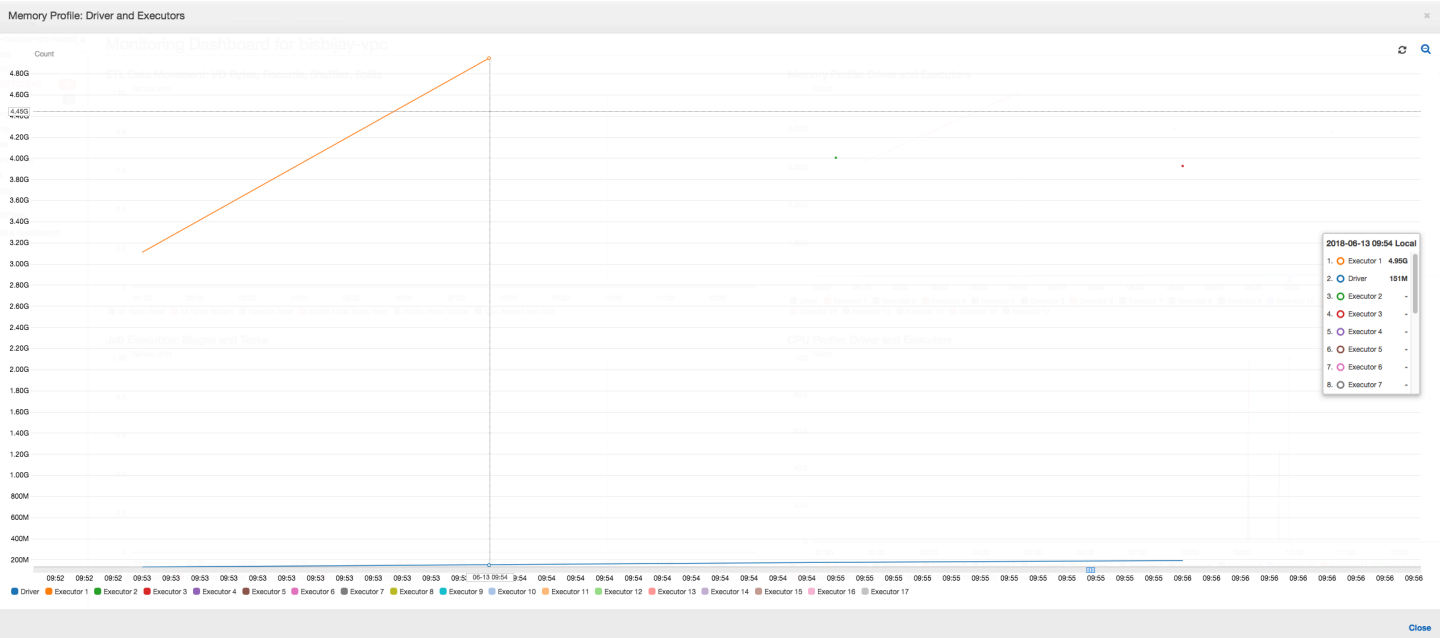
mencapai hingga 92 persen dan kontainer yang menjalankan pelaksana dihentikan oleh Apache Hadoop YARN.



Seperti yang ditunjukkan pada grafik berikut, selalu ada satu pelaksana tunggal yang berjalan sampai tugas gagal. Hal ini karena pelaksana baru diluncurkan untuk menggantikan pelaksana yang dihentikan. Pembacaan sumber data JDBC tidak diparalelisasi secara default karena akan membutuhkan pemartisian tabel pada kolom dan membuka beberapa koneksi. Akibatnya, hanya satu pelaksana yang membaca dalam tabel yang lengkap secara berurutan.



Sebagaimana ditunjukkan dalam grafik berikut, Spark mencoba untuk meluncurkan tugas baru empat kali sebelum gagal tugas. Anda dapat melihat profil memori dari tiga pelaksana. Setiap pelaksana dengan cepat menggunakan semua memorinya. Pelaksana keempat kehabisan memori, dan tugas gagal. Akibatnya, metriknya tidak langsung dilaporkan.



Anda dapat mengonfirmasi dari string kesalahan pada konsol AWS Glue bahwa tugas gagal karena pengecualian OOM, seperti yang ditunjukkan dalam gambar berikut.

Log keluaran Job: Untuk mengonfirmasi lebih lanjut temuan Anda tentang pengecualian OOM eksekutor, lihat CloudWatch Log. Saat Anda mencari **`Error`**, Anda menemukan empat pelaksana yang dihentikan di sekitar jendela waktu yang sama seperti yang ditunjukkan pada dasbor metrik. Semua diakhiri oleh YARN karena mereka melebihi batas memori mereka.

Pelaksana 1

```
18/06/13 16:54:29 WARN YarnAllocator: Container killed by YARN for exceeding
 memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:54:29 WARN YarnSchedulerBackend$YarnSchedulerEndpoint: Container killed
 by YARN for exceeding memory limits. 5.5 GB of 5.5 GB physical memory used. Consider
 boosting spark.yarn.executor.memoryOverhead.
```

```
18/06/13 16:54:29 ERROR YarnClusterScheduler: Lost executor 1 on
 ip-10-1-2-175.ec2.internal: Container killed by YARN for exceeding
 memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:54:29 WARN TaskSetManager: Lost task 0.0 in stage 0.0 (TID 0,
 ip-10-1-2-175.ec2.internal, executor 1): ExecutorLostFailure (executor 1
 exited caused by one of the running tasks) Reason: Container killed by YARN for
 exceeding memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

Pelaksana 2

```
18/06/13 16:55:35 WARN YarnAllocator: Container killed by YARN for exceeding
 memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:55:35 WARN YarnSchedulerBackend$YarnSchedulerEndpoint: Container killed
 by YARN for exceeding memory limits. 5.8 GB of 5.5 GB physical memory used. Consider
 boosting spark.yarn.executor.memoryOverhead.
18/06/13 16:55:35 ERROR YarnClusterScheduler: Lost executor 2 on
 ip-10-1-2-16.ec2.internal: Container killed by YARN for exceeding
 memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:55:35 WARN TaskSetManager: Lost task 0.1 in stage 0.0 (TID 1,
 ip-10-1-2-16.ec2.internal, executor 2): ExecutorLostFailure (executor 2 exited
 caused by one of the running tasks) Reason: Container killed by YARN for
 exceeding memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

Pelaksana 3

```
18/06/13 16:56:37 WARN YarnAllocator: Container killed by YARN for exceeding
 memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:56:37 WARN YarnSchedulerBackend$YarnSchedulerEndpoint: Container killed
 by YARN for exceeding memory limits. 5.8 GB of 5.5 GB physical memory used. Consider
 boosting spark.yarn.executor.memoryOverhead.
18/06/13 16:56:37 ERROR YarnClusterScheduler: Lost executor 3 on
 ip-10-1-2-189.ec2.internal: Container killed by YARN for exceeding
 memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:56:37 WARN TaskSetManager: Lost task 0.2 in stage 0.0 (TID 2,
 ip-10-1-2-189.ec2.internal, executor 3): ExecutorLostFailure (executor 3
 exited caused by one of the running tasks) Reason: Container killed by YARN for
```

```
 exceeding memory limits. 5.8 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

Pelaksana 4

```
18/06/13 16:57:18 WARN YarnAllocator: Container killed by YARN for exceeding
 memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:57:18 WARN YarnSchedulerBackend$YarnSchedulerEndpoint: Container killed
 by YARN for exceeding memory limits. 5.5 GB of 5.5 GB physical memory used. Consider
 boosting spark.yarn.executor.memoryOverhead.
18/06/13 16:57:18 ERROR YarnClusterScheduler: Lost executor 4 on
 ip-10-1-2-96.ec2.internal: Container killed by YARN for exceeding
 memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
18/06/13 16:57:18 WARN TaskSetManager: Lost task 0.3 in stage 0.0 (TID 3,
 ip-10-1-2-96.ec2.internal, executor 4): ExecutorLostFailure (executor 4 exited
 caused by one of the running tasks) Reason: Container killed by YARN for
 exceeding memory limits. 5.5 GB of 5.5 GB physical memory used. Consider boosting
 spark.yarn.executor.memoryOverhead.
```

Perbaiki pengaturan ukuran ambil menggunakan frame AWS Glue dinamis

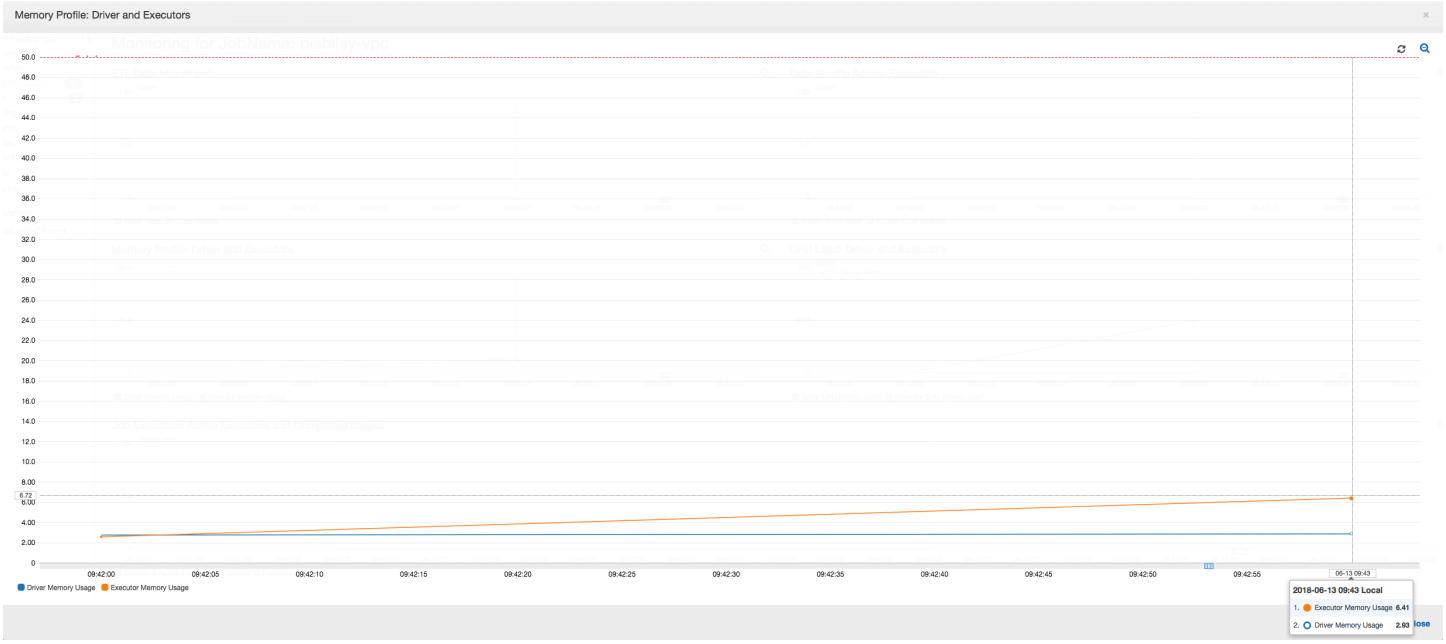
Pelaksana kehabisan memori saat membaca tabel JDBC karena konfigurasi default untuk ukuran pengambilan JDBC Spark adalah nol. Ini berarti bahwa driver JDBC pada pelaksana Spark mencoba untuk mengambil 34 juta baris dari basis data bersama-sama dan meng-cache mereka, meskipun pengaliran Spark melalui baris-baris tersebut, sekali dalam satu waktu. Dengan Spark, Anda dapat menghindari skenario ini dengan menetapkan parameter ukuran pengambilan ke nilai default non-nol.

Anda juga dapat memperbaiki masalah ini dengan menggunakan bingkai dinamis AWS Glue sebagai gantinya. Secara default, bingkai dinamis menggunakan ukuran pengambilan sebesar 1.000 baris yang merupakan nilai yang biasanya cukup. Akibatnya, pelaksana tidak mengambil lebih dari 7 persen dari total memorinya. Tugas AWS Glue selesai dalam waktu kurang dari dua menit dengan hanya satu pelaksana tunggal. Selain menggunakan bingkai dinamis AWS Glue merupakan pendekatan yang direkomendasikan, ia juga memungkinkan kita untuk mengatur ukuran pengambilan dengan menggunakan properti `fetchsize` Apache Spark. Lihat Spark SQL, DataFrames dan Panduan Datasets.

```
val (url, database, tableName) = {
 ("jdbc_url", "db_name", "table_name")
```

```
 }
val source = glueContext.getSource(format, sourceJson)
val df = source.getDynamicFrame
glueContext.write_dynamic_frame.from_options(frame = df, connection_type = "s3",
 connection_options = {"path": output_path}, format = "parquet", transformation_ctx =
 "datasink")
```

Metrik profilan normal: Memori pelaksana dengan bingkai dinamis AWS Glue tidak pernah melebihi ambang batas aman, seperti yang ditunjukkan pada gambar berikut. Ia mengalirkan dalam baris-baris dari basis data dan meng-cache hanya 1.000 baris dalam driver JDBC pada setiap titik waktu. Pengecualian kehabisan memori tidak terjadi.



Mendebug tahapan yang menuntut dan tugas yang menyimpang

Anda dapat menggunakan pemrofilan tugas AWS Glue untuk mengidentifikasi tahapan yang menuntut dan tugas dengan performa buruk dalam tugas extract, transform, and load (ETL) Anda. Sebuah tugas dengan performa buruk memakan waktu lebih lama dari tugas-tugas lainnya dalam tahap sebuah tugas AWS Glue. Akibatnya, tahapan tersebut membutuhkan waktu lebih lama untuk selesai, yang juga menunda total waktu eksekusi tugas.

Menggabungkan file input kecil menjadi file output yang lebih besar

Sebuah tugas dengan performa buruk dapat terjadi ketika ada distribusi pekerjaan yang tidak seragam di seluruh tugas yang berbeda, atau adanya data condong yang mengakibatkan satu tugas memproses lebih banyak data.

Anda dapat melakukan pemrofilan pada kode berikut—pola umum di Apache Spark—untuk menggabungkan sejumlah besar file kecil ke dalam file output yang lebih besar. Untuk contoh ini, set data inputnya adalah file JSON Gzip terkompresi sebesar 32 GB. Set data output mempunyai file JSON tidak terkompresi kira-kira sebesar 190 GB.

Kode profilannya adalah sebagai berikut:

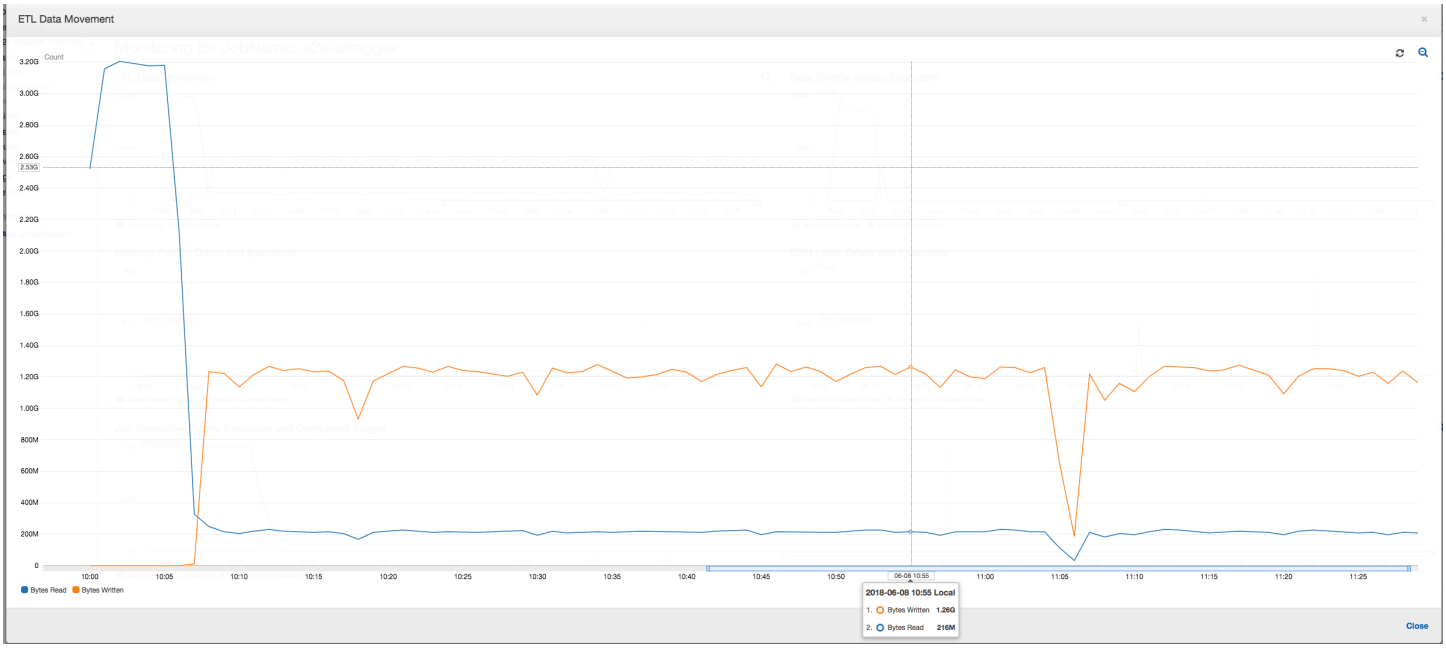
```
datasource0 = spark.read.format("json").load("s3://input_path")
df = datasource0.coalesce(1)
df.write.format("json").save(output_path)
```

Visualisasikan metrik yang diprofilkan di konsol AWS Glue

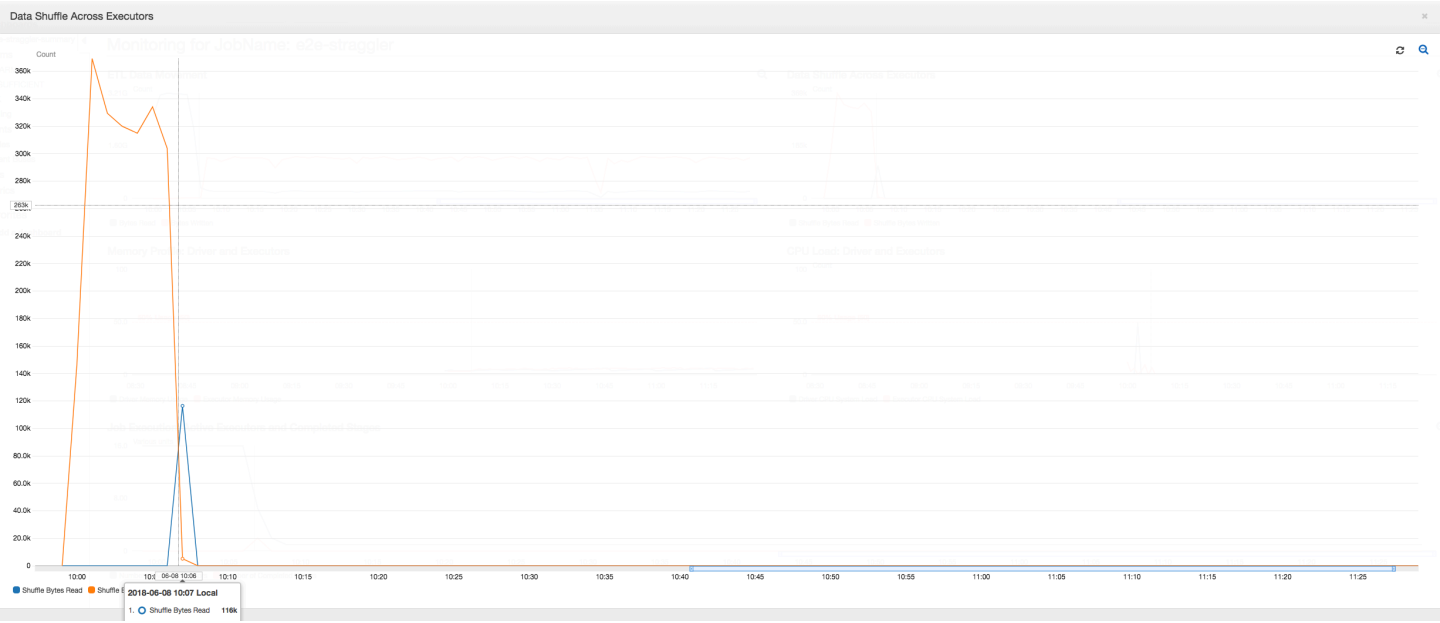
Anda dapat melakukan pemrofilan pada tugas Anda untuk memeriksa empat rangkaian metrik yang berbeda:

- Pergerakan data ETL
- Data yang diacak di seluruh pelaksana
- Eksekusi tugas
- Profil memori

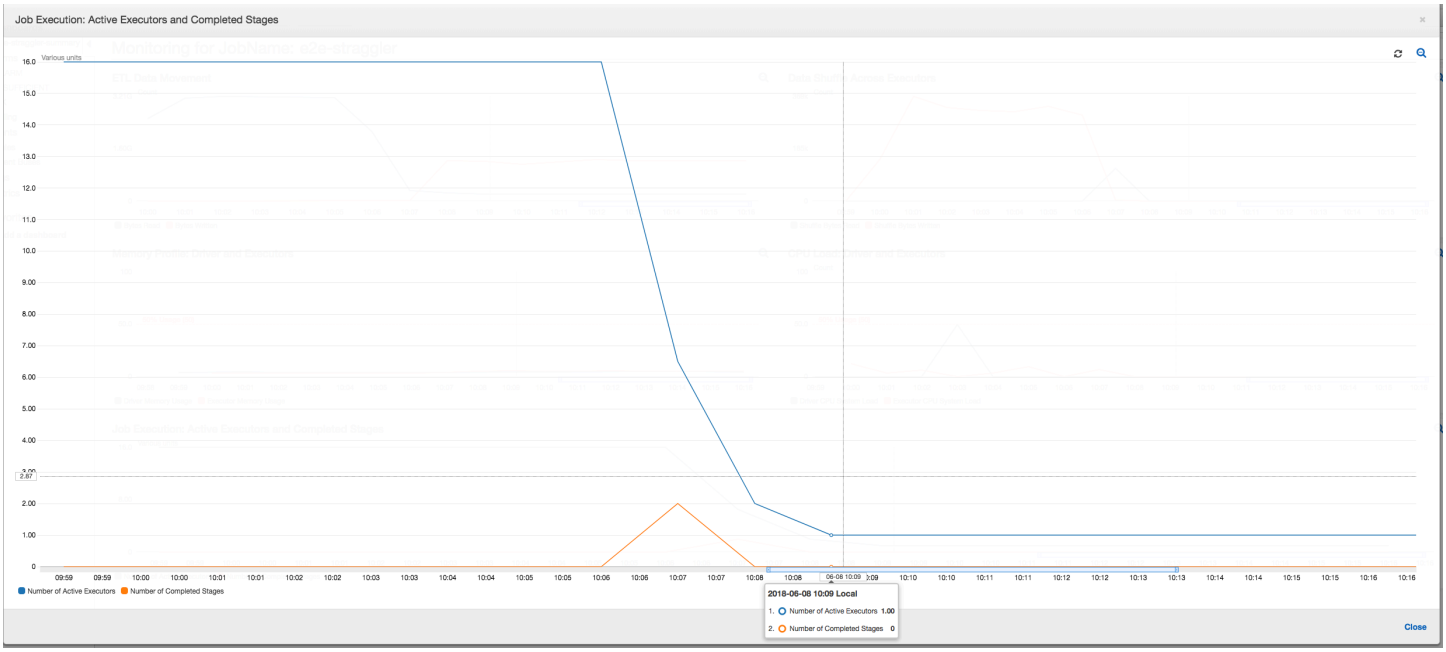
Pergerakan data ETL: Dalam profil Pergerakan data ETL, byte di-baca cukup cepat oleh semua pelaksana di tahap pertama yang selesai dalam waktu enam menit pertama. Namun, total waktu eksekusi tugas sekitar satu jam, sebagian besar terdiri dari eksekusi tulis data.



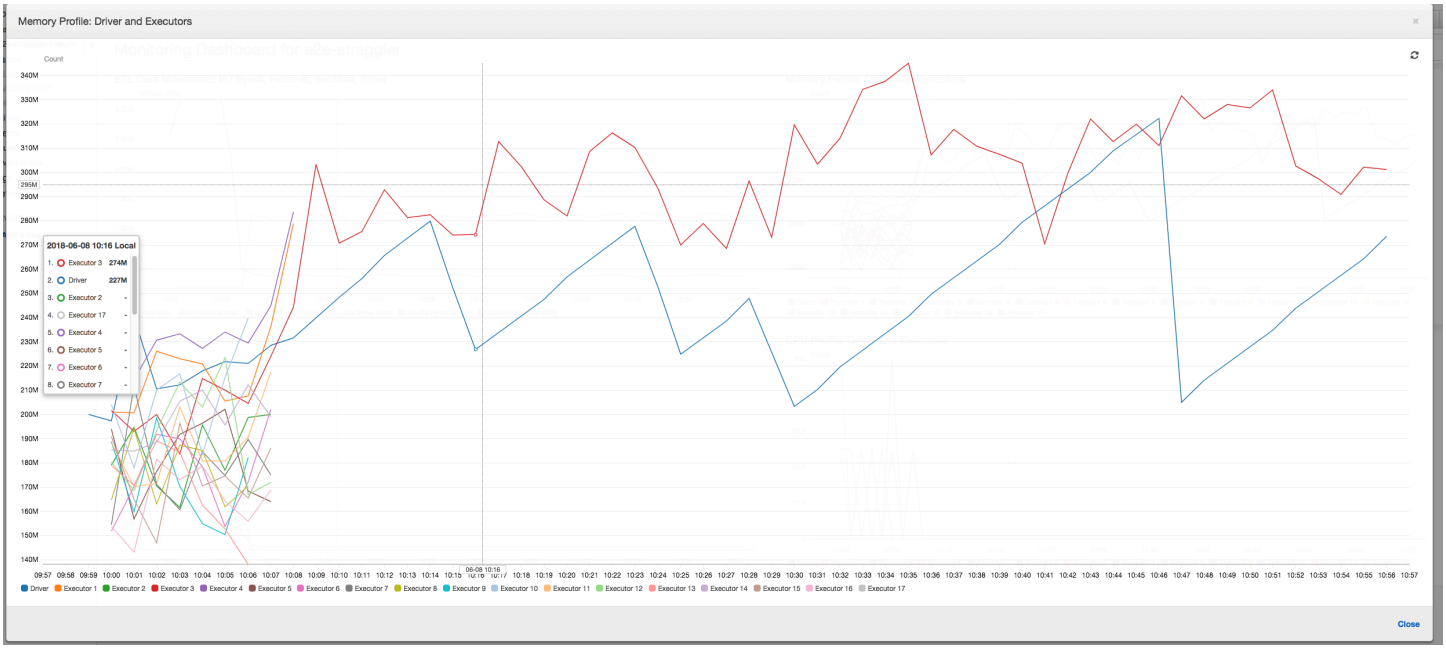
Data yang diacak di seluruh pelaksana: Jumlah byte yang di-baca dan di-tulis selama pengacakan juga menunjukkan lonjakan sebelum Tahap 2 berakhir, seperti yang ditunjukkan oleh metrik Eksekusi Tugas dan Pengacakan Data. Setelah pengacakan data dilakukan di semua pelaksana, baca dan tulis melanjutkan dari pelaksana nomor 3 saja.



Eksekusi Tugas: Seperti ditunjukkan pada grafik di bawah ini, semua pelaksana lainnya menganggur dan akhirnya dilepaskan pada waktu 10:09. Pada saat itu, jumlah total pelaksana menurun menjadi hanya satu saja. Hal ini jelas menunjukkan bahwa pelaksana nomor 3 terdiri dari tugas dengan performa buruk yang mengambil waktu eksekusi paling lama dan memberikan kontribusi untuk sebagian besar waktu eksekusi tugas.



Profil memori: Setelah dua tahap pertama, hanya pelaksana nomor 3 yang secara aktif menggunakan memori untuk memproses data. Pelaksana yang tersisa hanya menganggur atau telah dilepaskan tak lama setelah dua tahap pertama selesai.



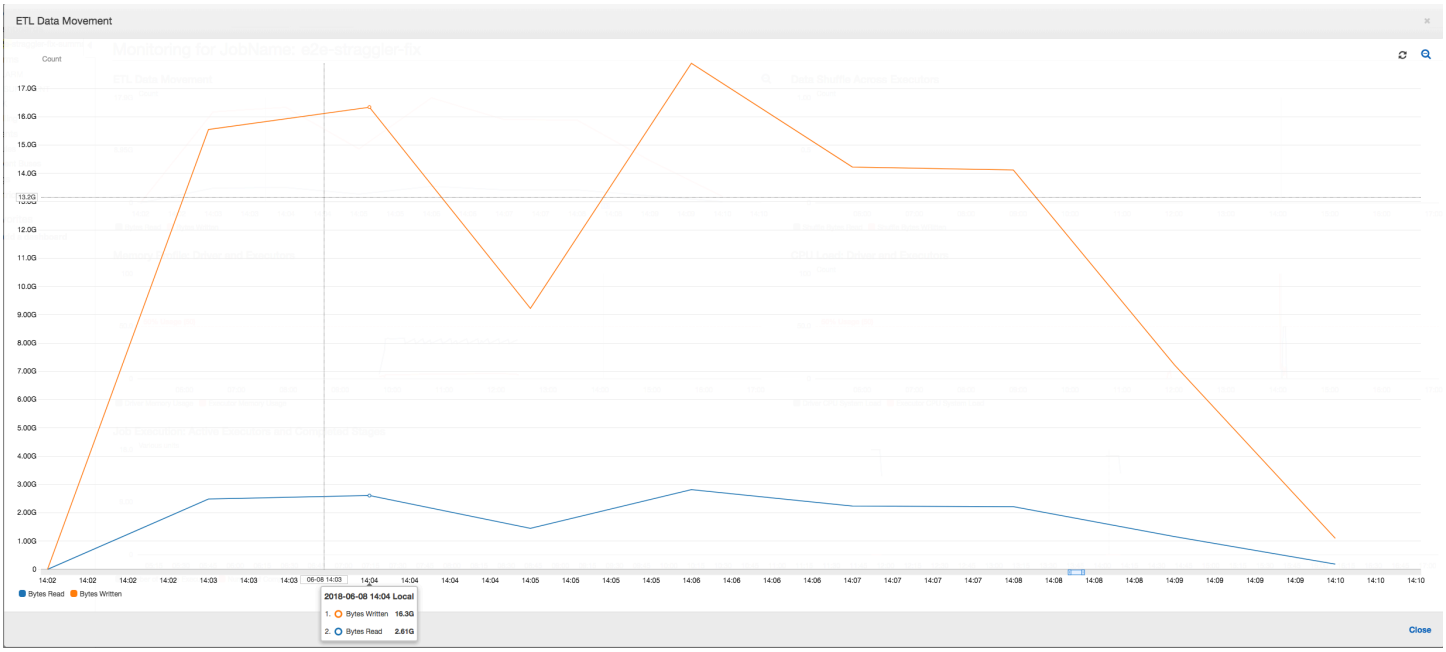
Perbaiki pelaksana yang tersesat menggunakan pengelompokan

Anda dapat menghindari terjadinya pelaksana dengan performa buruk dengan menggunakan fitur pengelompokan dalam grup di AWS Glue. Gunakan pengelompokan dalam grup untuk mendistribusikan data secara seragam di semua pelaksana dan menyatukan file-file ke dalam file yang lebih besar dengan menggunakan semua pelaksana yang tersedia pada klaster. Untuk informasi selengkapnya, lihat Membaca file input dalam kelompok yang lebih besar.

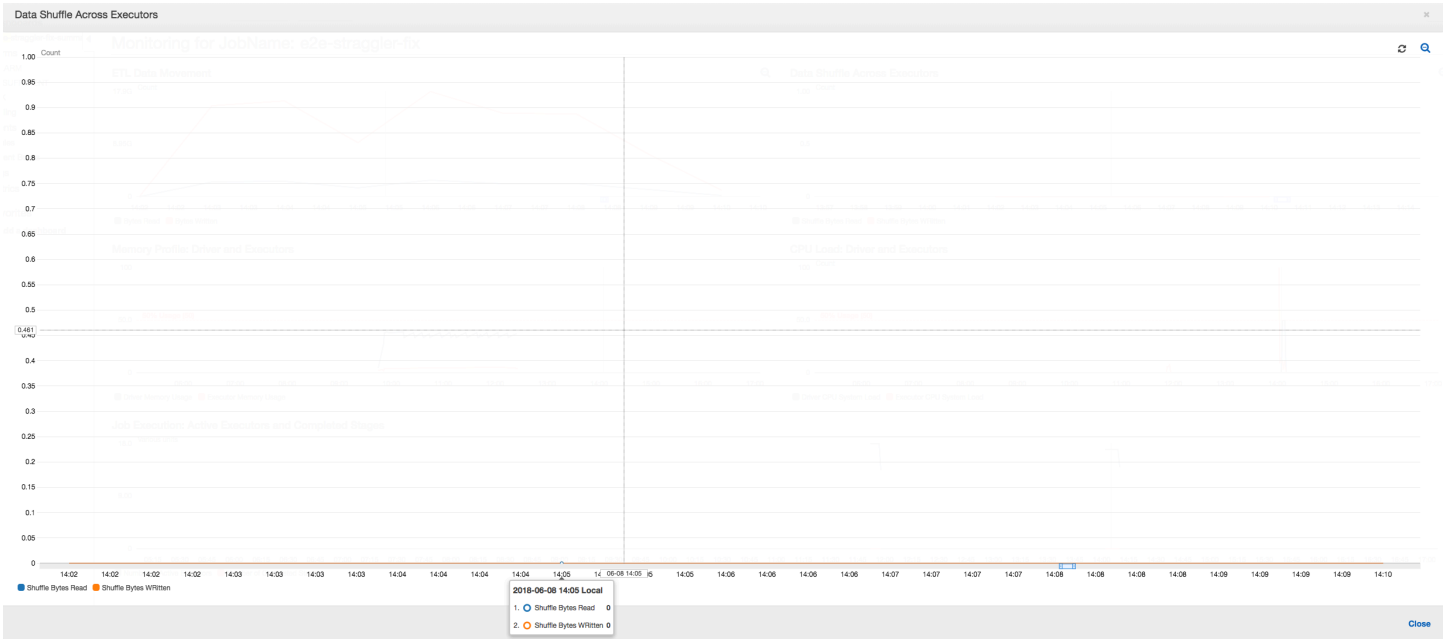
Untuk memeriksa pergerakan data ETL di tugas AWS Glue, lakukan pemrofilan pada kode berikut dengan pengelompokan grup yang diaktifkan:

```
df = glueContext.create_dynamic_frame_from_options("s3", {'paths': ["s3://input_path"],
 "recurse":True, 'groupFiles': 'inPartition'}, format="json")
datasink = glueContext.write_dynamic_frame.from_options(frame = df, connection_type =
 "s3", connection_options = {"path": output_path}, format = "json", transformation_ctx
 = "datasink4")
```

Pergerakan data ETL: Data tulis sekarang dialirkan secara paralel dengan data baca di sepanjang waktu eksekusi tugas. Akibatnya, tugas selesai dalam delapan menit—jauh lebih cepat dari sebelumnya.

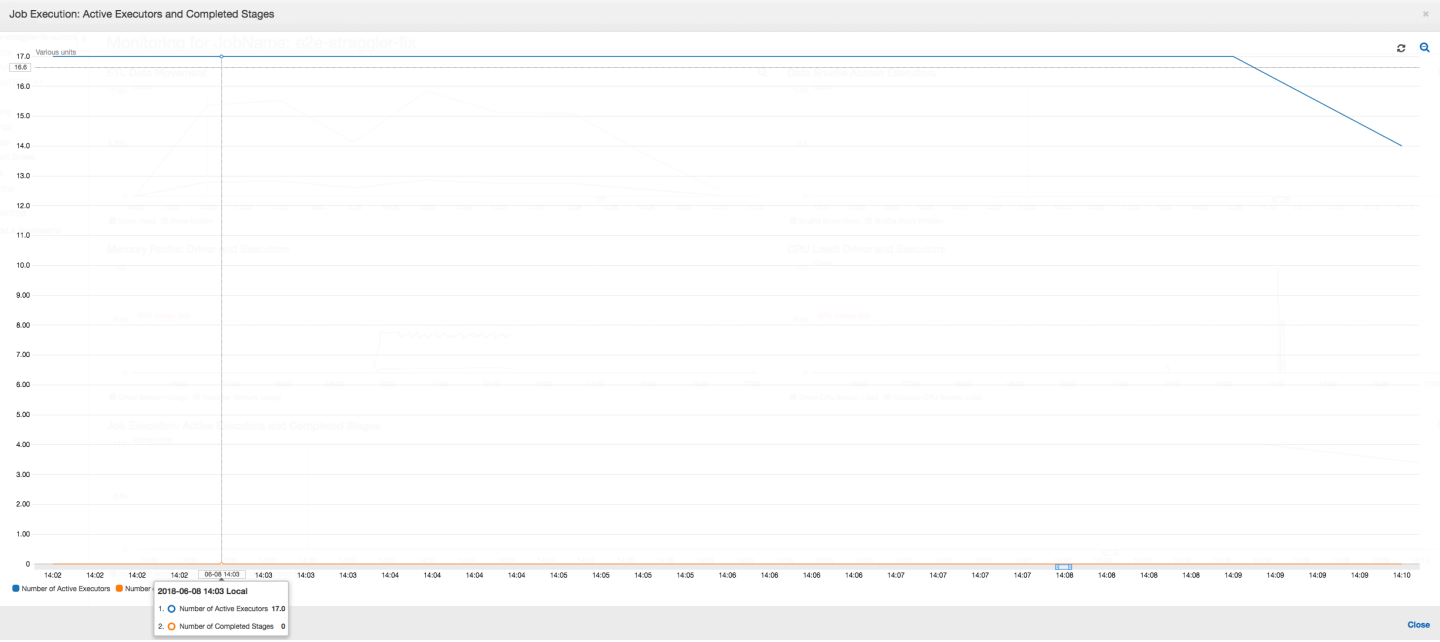


Data yang diacak di seluruh pelaksana: Karena file input digabungkan selama baca menggunakan fitur pengelompokan dalam grup, tidak ada pengacakan data yang mahal setelah pembacaan data.

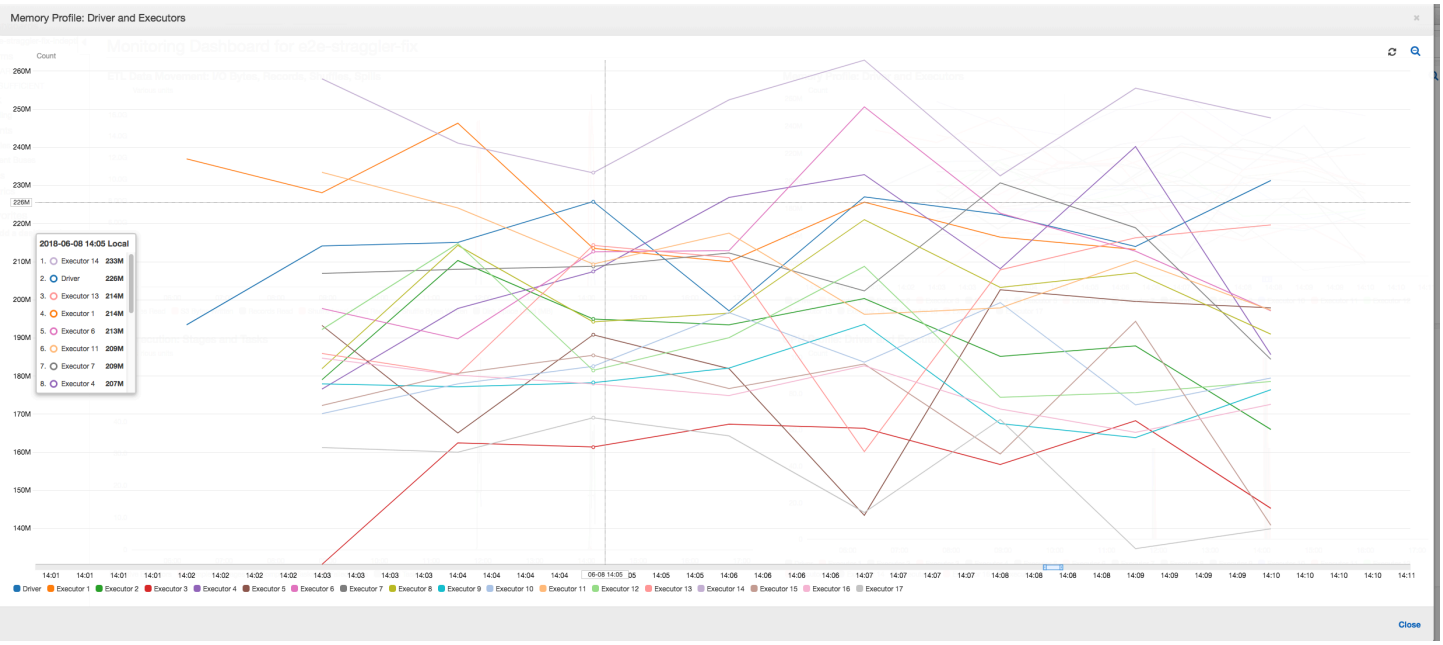


Eksekusi tugas: Metrik eksekusi tugas menunjukkan bahwa jumlah pelaksana aktif yang menjalankan dan memproses data masih tetap cukup konstan. Tidak ada performa buruk yang terjadi dalam tugas

itu. Semua pelaksana aktif dan tidak dilepaskan sampai tugas selesai. Karena tidak ada pengacakan menengah pada data di seluruh pelaksana, maka hanya ada satu tahap dalam tugas tersebut.



Profil memori: Metrik menunjukkan konsumsi memori aktif di semua pelaksana—mengonfirmasi kembali bahwa ada aktivitas di semua pelaksana. Karena data dialirkan dan ditulis secara paralel, maka total jejak memori dari semua pelaksana kira-kira seragam dan jauh di bawah ambang batas aman untuk semua pelaksana.



Memantau kemajuan beberapa pekerjaan

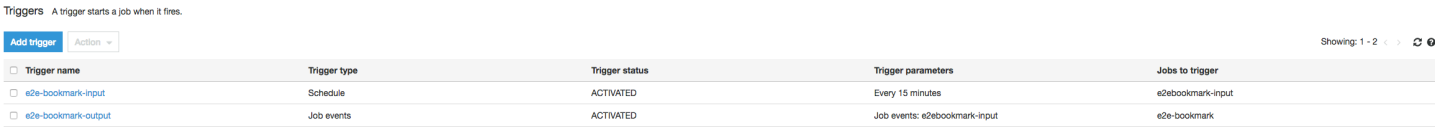
Anda dapat melakukan pemrofilan pada beberapa tugas AWS Glue secara bersama-sama dan memantau pengaliran data di antara mereka. Ini adalah sebuah pola alur kerja yang umum, dan memerlukan pemantauan untuk kemajuan tugas individu, backlog pemrosesan data, pemrosesan ulang data, dan bookmark tugas.

Topik

- Kode diprofilkan
- Visualisasikan metrik yang diprofilkan di konsol AWS Glue
- Perbaiki pemrosesan file

Kode diprofilkan

Dalam alur kerja ini, Anda memiliki dua tugas: Tugas Input dan Tugas Output. Tugas Input dijadwalkan untuk berjalan setiap 30 menit dengan menggunakan pemicu periodik. Tugas Output dijadwalkan untuk berjalan setelah setiap eksekusi Tugas Input yang berhasil. Tugas-tugas terjadwal ini dikendalikan dengan menggunakan pemicu tugas.



Tugas Input: Tugas ini membaca data dari lokasi Amazon Simple Storage Service (Amazon S3), mengubahnya menggunakan `ApplyMapping`, dan menuliskannya ke lokasi Amazon S3 yang bertahap. Kode berikut adalah kode profilan untuk tugas Input:

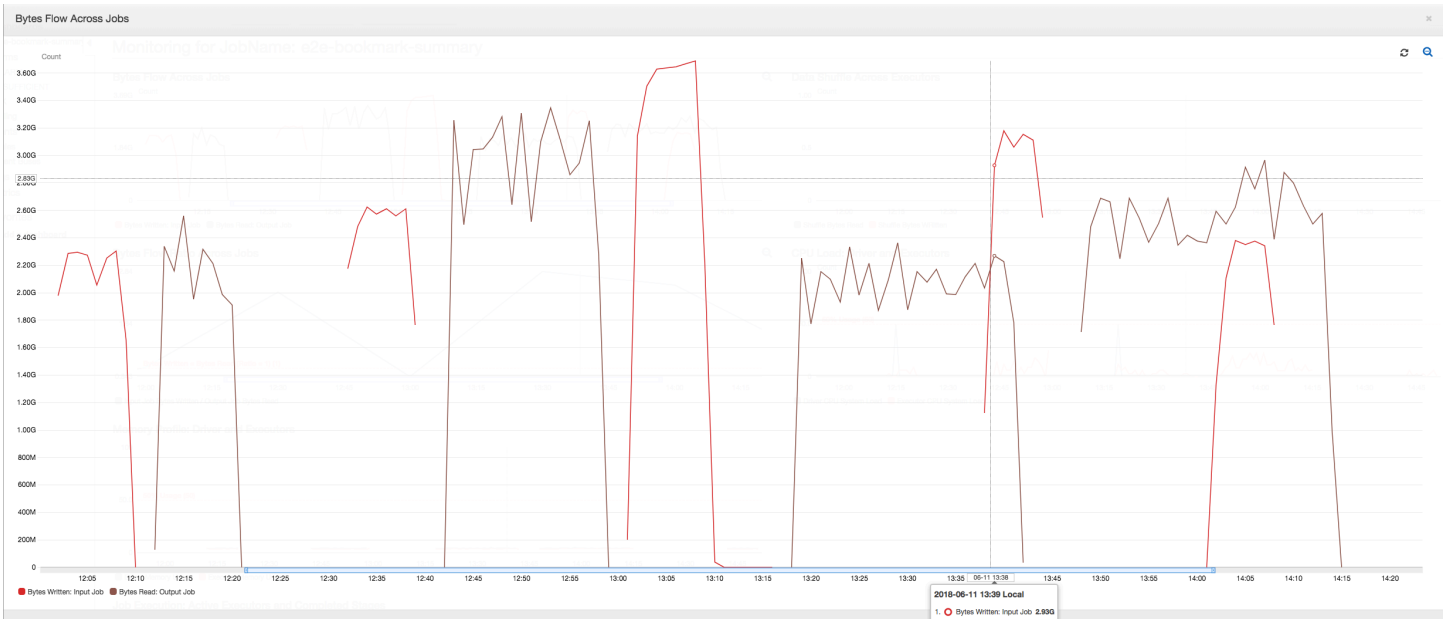
```
datasource0 = glueContext.create_dynamic_frame.from_options(connection_type="s3",
 connection_options = {"paths": ["s3://input_path"],
 "useS3ListImplementation":True,"recurse":True}, format="json")
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [map_spec])
datasink2 = glueContext.write_dynamic_frame.from_options(frame = applymapping1,
 connection_type = "s3", connection_options = {"path": staging_path, "compression":
 "gzip"}, format = "json")
```

Tugas Output: Tugas ini membaca output dari tugas Input dari lokasi pentahapan di Amazon S3, melakukan transformasi lagi padanya, dan menuliskannya ke tujuan:

```
datasource0 = glueContext.create_dynamic_frame.from_options(connection_type="s3",
 connection_options = {"paths": [staging_path],
 "useS3ListImplementation":True,"recurse":True}, format="json")
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [map_spec])
datasink2 = glueContext.write_dynamic_frame.from_options(frame = applymapping1,
 connection_type = "s3", connection_options = {"path": output_path}, format = "json")
```

Visualisasikan metrik yang diprofilkan di konsol AWS Glue

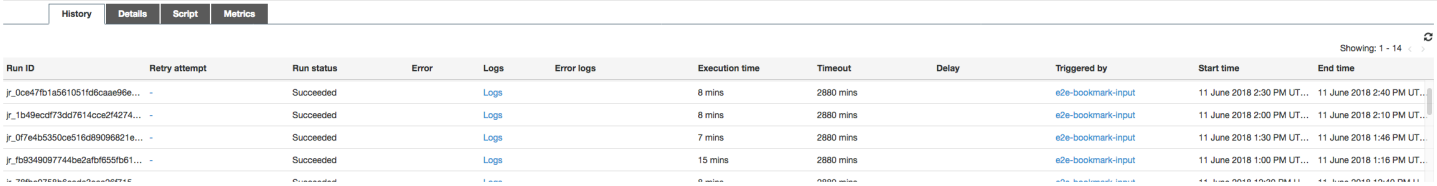
Dasbor berikut melapiskan metrik tertulis byte Amazon S3 dari tugas Input ke metrik pembacaan byte Amazon S3 pada lini waktu yang sama untuk tugas Output. Lini waktu tersebut menunjukkan eksekusi tugas yang berbeda dari tugas Input dan Output. Tugas Input (ditampilkan dalam warna merah) dimulai setiap 30 menit. Tugas Output (ditampilkan dalam warna coklat) dimulai pada saat Tugas Input selesai, dengan Konkurensi Maksimal 1.



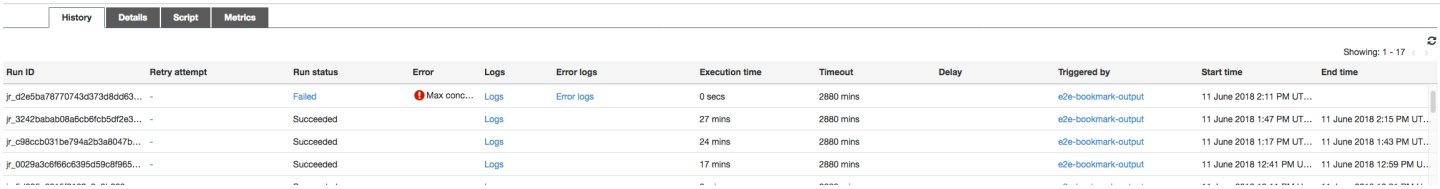
Dalam contoh ini, bookmark tugas tidak diaktifkan. Tidak ada konteks transformasi yang digunakan untuk mengaktifkan bookmark tugas dalam kode skrip tersebut.

Riwayat Tugas: Tugas Input dan Output menjalani beberapa eksekusi, seperti yang ditunjukkan pada tab Riwayat, mulai dari 12:00 siang.

Tugas Input pada konsol AWS Glue terlihat seperti ini:

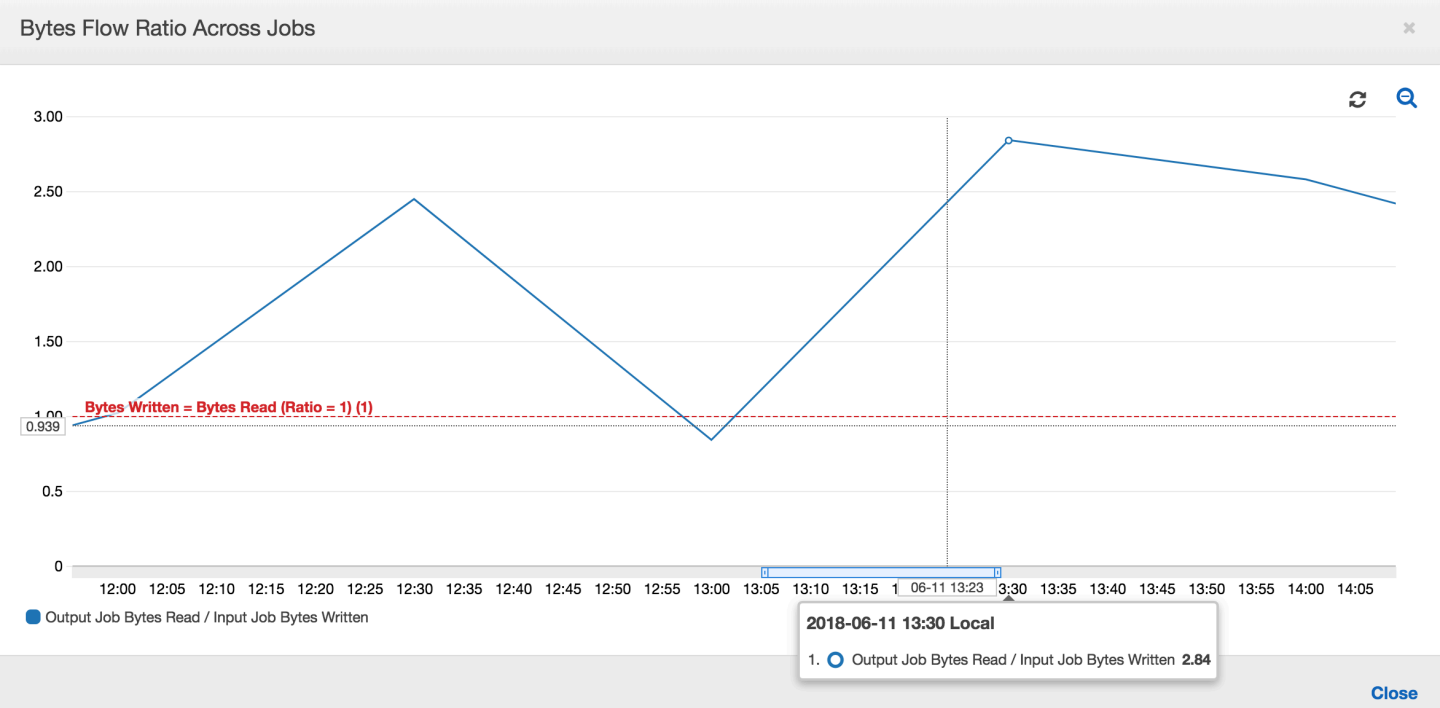


Gambar berikut menunjukkan tugas Output:



Eksekusi tugas pertama: Seperti yang ditunjukkan dalam grafik Byte Data yang Di-baca dan Di-tulis di bawah ini, eksekusi tugas pertama dari tugas Input dan Output antara jam 12:00 dan 12:30 menunjukkan sekitar area yang sama di bawah kurva. Area-area tersebut mewakili byte Amazon S3 yang ditulis oleh tugas Input dan byte Amazon S3 yang dibaca oleh tugas Output. Data ini juga dikonfirmasi oleh rasio byte Amazon S3 yang ditulis (dijumlahkan lebih dari 30 menit – frekuensi pemicu tugas untuk tugas Input). Titik data untuk rasio untuk eksekusi tugas Input yang dimulai pada 12:00 siang juga 1.

Grafik berikut menunjukkan rasio pengaliran data di semua eksekusi tugas:



Eksekusi tugas kedua: Dalam eksekusi tugas kedua, ada perbedaan yang jelas dalam jumlah byte yang dibaca oleh tugas Output dibandingkan dengan jumlah byte yang ditulis oleh tugas Input. (Bandingkan area di bawah kurva di dua eksekusi tugas untuk tugas Output, atau bandingkan area dalam eksekusi tugas kedua dari tugas Input dan Output.) Rasio byte yang dibaca dan ditulis menunjukkan bahwa Tugas Output membaca sekitar 2,5x data yang ditulis oleh tugas Input dalam rentang kedua 30 menit dari jam 12:30 hingga 13:00. Hal ini karena Tugas Output memproses

kembali output eksekusi tugas pertama dari tugas Input karena bookmark tugas tidak diaktifkan. Rasio di atas 1 menunjukkan bahwa ada backlog data tambahan yang diproses oleh tugas Output.

Eksekusi tugas ketiga: Tugas Input cukup konsisten dalam hal jumlah byte yang ditulis (lihat area di bawah kurva merah). Namun, eksekusi tugas ketiga dari tugas Input berjalan lebih lama dari yang diharapkan (lihat ekor panjang kurva merah). Akibatnya, eksekusi tugas ketiga dari tugas Output dimulai terlambat. Eksekusi tugas ketiga hanya memproses sebagian kecil dari data yang terakumulasi di lokasi pentahapan di sisa 30 menit antara pukul 13:00 dan 13:30. Rasio aliran byte menunjukkan bahwa ia hanya memproses 0,83 data yang ditulis oleh eksekusi tugas ketiga dari tugas Input (lihat rasio di pukul 13:00).

Tumpang tindih tugas Input dan Output: Eksekusi tugas keempat dari tugas Input dimulai pada pukul 13:30 sesuai jadwal, sebelum eksekusi tugas ketiga dari tugas Output selesai. Ada tumpang tindih parsial antara dua eksekusi tugas ini. Namun demikian, eksekusi tugas ketiga dari tugas Output menangkap hanya file yang terdaftar di lokasi pentahapan Amazon S3 ketika dimulai sekitar pukul 13:17. Hal ini terdiri dari semua output data dari eksekusi tugas pertama dari tugas Input. Rasio aktual pada pukul 13:30 adalah sekitar 2.75. Eksekusi tugas ketiga dari tugas Output memproses sekitar 2.75x data yang ditulis oleh eksekusi tugas keempat dari tugas Input pada pukul 13:30 sampai 14:00.

Seperti yang ditunjukkan gambar ini, tugas Output mengolah ulang data dari lokasi pentahapan dari semua eksekusi tugas sebelumnya dari tugas Input. Akibatnya, eksekusi tugas keempat untuk tugas Output menjadi eksekusi tugas terpanjang dan tumpang tindih dengan seluruh eksekusi tugas kelima dari Input tugas.

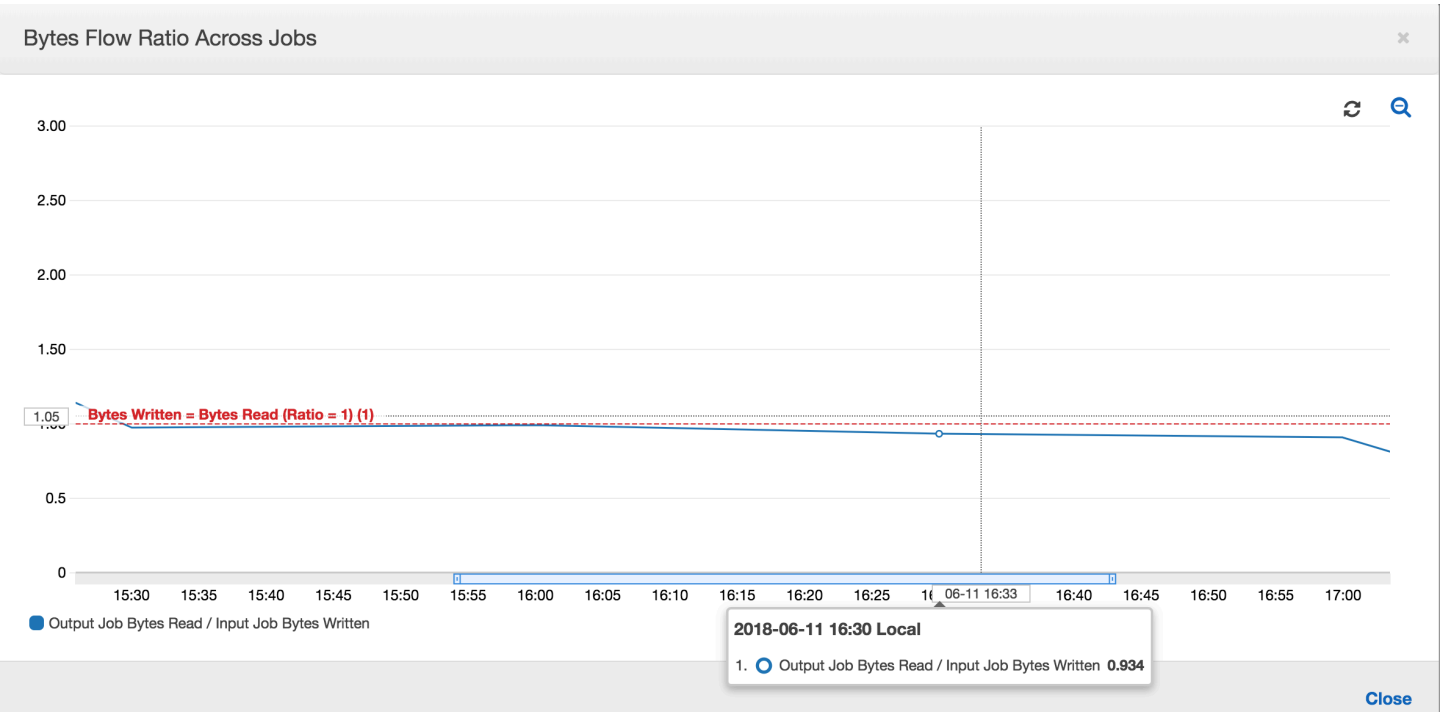
Perbaiki pemrosesan file

Anda harus memastikan bahwa tugas Output memproses hanya file yang belum diproses oleh eksekusi tugas sebelumnya dari tugas Output. Untuk melakukan ini, aktifkan bookmark tugas dan atur konteks transformasi dalam tugas Output, sebagai berikut:

```
datasource0 = glueContext.create_dynamic_frame.from_options(connection_type="s3",
 connection_options = {"paths": [staging_path],
 "useS3ListImplementation":True,"recurse":True}, format="json", transformation_ctx =
 "bookmark_ctx")
```

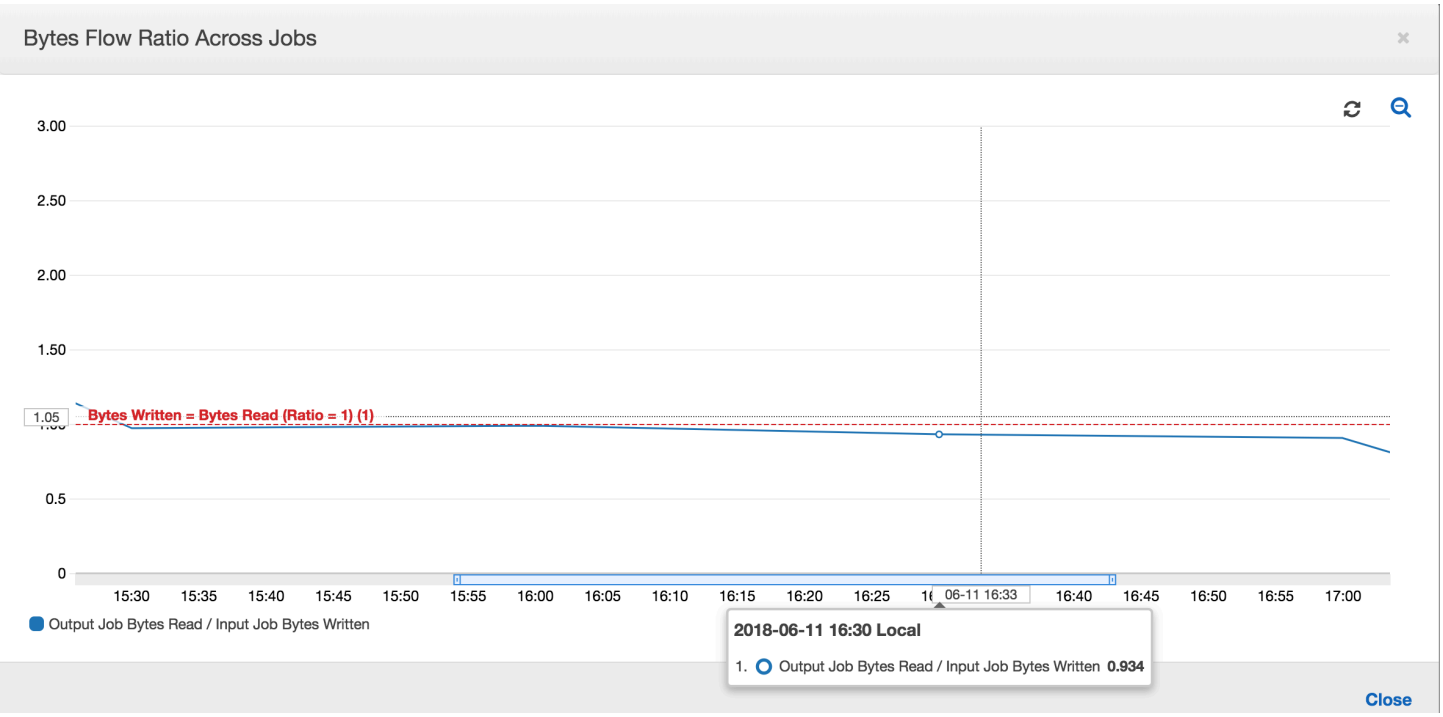
Dengan bookmark tugas yang diaktifkan, tugas Output tidak akan memproses ulang data di lokasi pentahapan dari semua eksekusi tugas sebelumnya dari tugas Input. Pada gambar berikut, menunjukkan data yang dibaca dan ditulis, area yang ada di bawah kurva coklat cukup konsisten dan mirip dengan kurva merah.

Rasio aliran byte juga kira-kira masih tetap mendekati 1 karena tidak ada data tambahan yang diproses.



Sebuah eksekusi tugas untuk tugas Output dimulai dan menangkap file di lokasi pentahapan sebelum eksekusi tugas Input berikutnya mulai menempatkan lebih banyak data ke lokasi

pentahapan. Selama hal tersebut berjalan seperti itu, ia hanya memproses file yang ditangkap dari eksekusi tugas Input sebelumnya, dan rasio tetap dekat dengan 1.



Misalkan tugas Input memakan waktu lebih lama dari yang diharapkan, maka sebagai akibatnya, tugas Output menangkap file di lokasi pentahapan dari dua eksekusi tugas Input. Rasio ini kemudian lebih tinggi dari 1 untuk eksekusi tugas output itu. Namun demikian, eksekusi tugas Output berikutnya tidak memproses file yang sudah diproses oleh eksekusi tugas dari tugas Output.

Pemantauan perencanaan kapasitas DPU

Anda dapat menggunakan metrik tugas di AWS Glue untuk memperkirakan jumlah unit pemrosesan data (DPU) yang dapat digunakan untuk menskalakan keluar sebuah tugas AWS Glue.

> **Note**
>
> Halaman ini hanya berlaku untuk AWS Glue versi 0.9 dan 1.0. Versi selanjutnya AWS Glue berisi fitur hemat biaya yang memperkenalkan pertimbangan tambahan saat perencanaan kapasitas.

Topik

- Kode diprofilkan
- Visualisasikan metrik yang diprofilkan di konsol AWS Glue

- Tentukan kapasitas DPU yang optimal

Kode diprofilkan

Skrip berikut membaca partisi Amazon Simple Storage Service (Amazon S3) yang berisi 428 file JSON gzip. Skrip tersebut menerapkan pemetaan untuk mengubah nama bidang, dan mengkonversi dan menuliskannya mereka ke Amazon S3 dalam format Apache Parquet. Anda menyediakan 10 DPU sesuai default dan menjalankan tugas ini.

```
datasource0 = glueContext.create_dynamic_frame.from_options(connection_type="s3",
 connection_options = {"paths": [input_path],
 "useS3ListImplementation":True,"recurse":True}, format="json")
applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [(map_spec])
datasink2 = glueContext.write_dynamic_frame.from_options(frame = applymapping1,
 connection_type = "s3", connection_options = {"path": output_path}, format =
 "parquet")
```

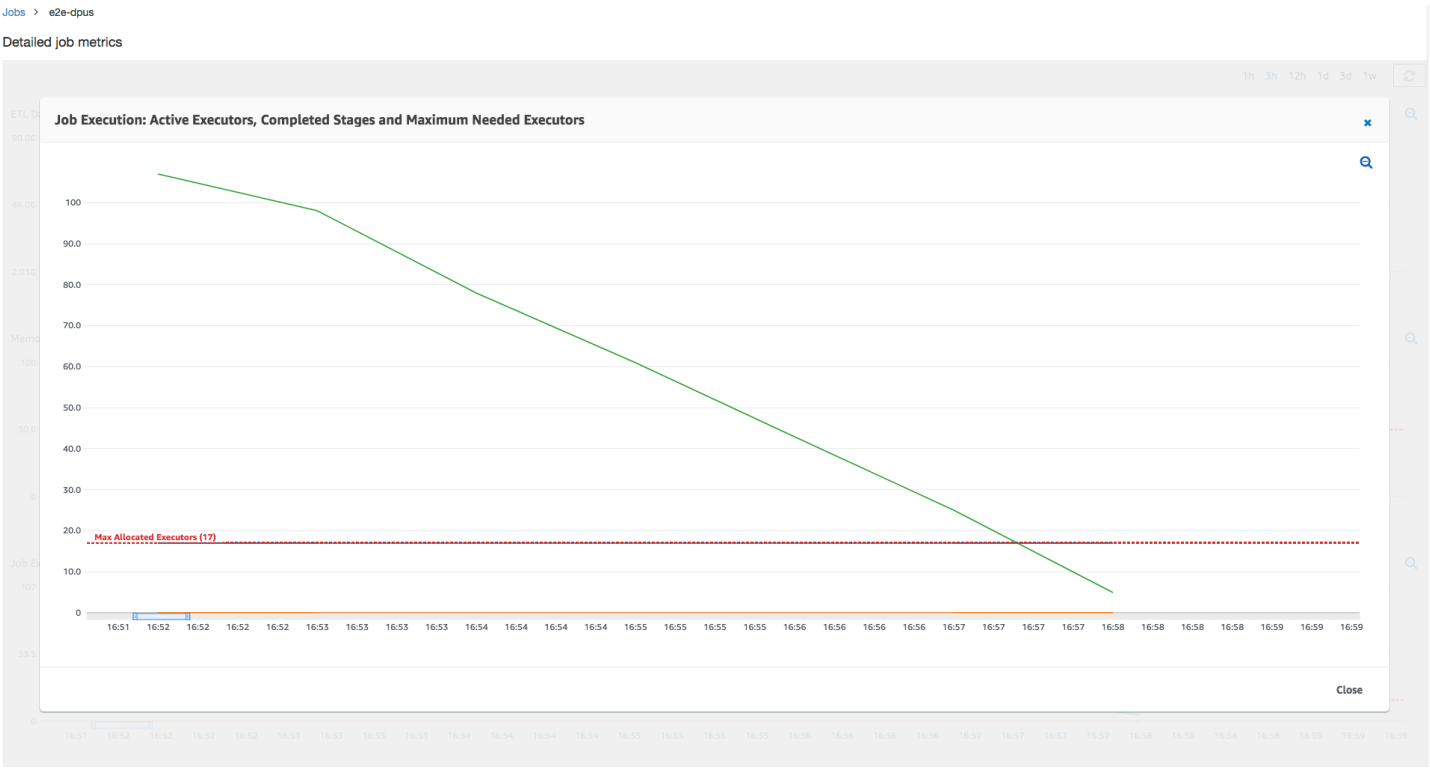
Visualisasikan metrik yang diprofilkan di konsol AWS Glue

Job run 1: Dalam job run ini kami menunjukkan cara mencari apakah ada DPU yang kurang disediakan di cluster. Fungsi eksekusi tugas di AWS Glue menampilkan total jumlah pelaksana yang berjalan aktif, jumlah tahap yang telah selesai, dan jumlah maksimal pelaksana yang dibutuhkan.

Jumlah pelaksana maksimum yang dibutuhkan dihitung dengan menambahkan jumlah total tugas berjalan dan tugas yang tertunda, dan membaginya dengan tugas per pelaksana. Hasil ini adalah ukuran jumlah total pelaksana yang diperlukan untuk memenuhi beban saat ini.

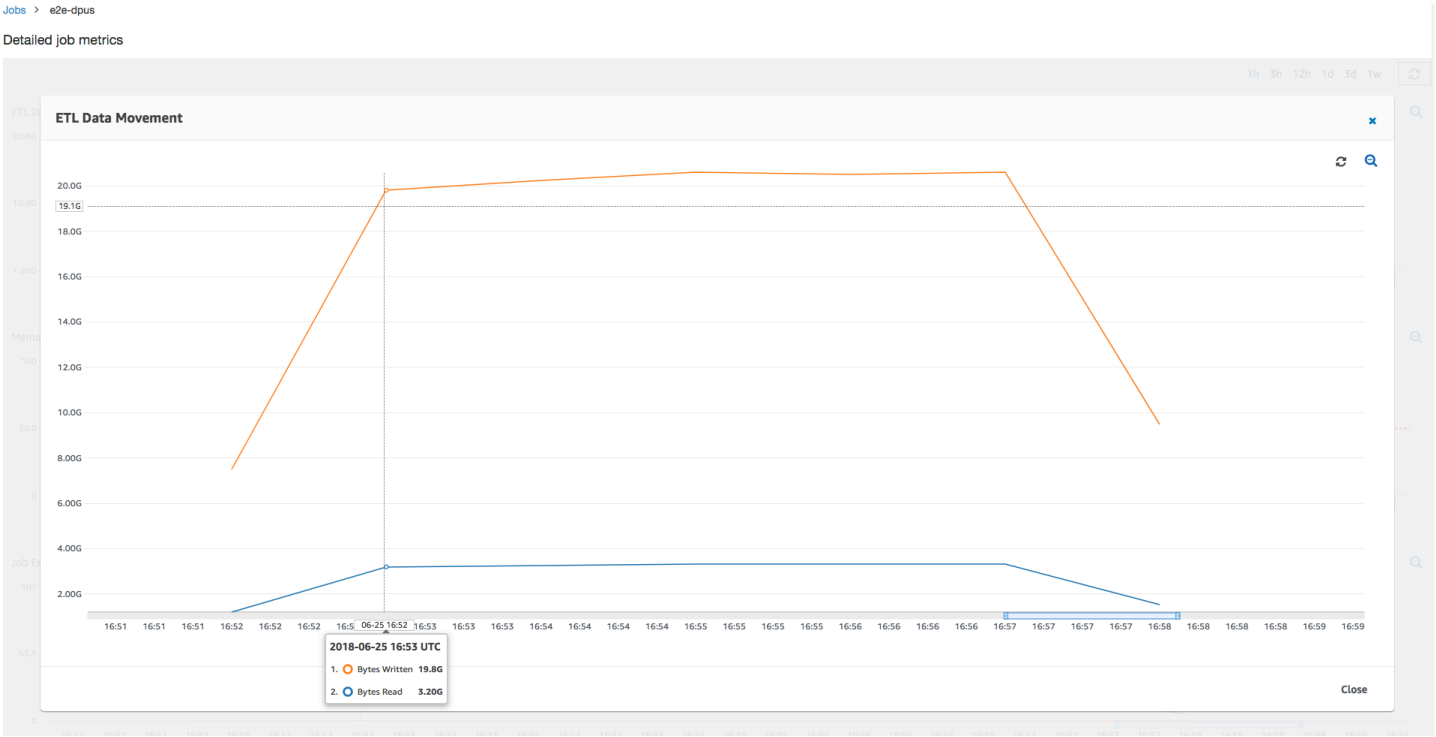
Sebaliknya, jumlah pelaksana yang berjalan aktif mengukur berapa banyak pelaksana yang menjalankan tugas Apache Spark secara aktif. Saat tugas berlangsung, pelaksana maksimum yang diperlukan dapat mengubah dan biasanya turun menjelang akhir tugas saat antrean tugas tertunda berkurang.

Garis merah horizontal pada grafik berikut menunjukkan jumlah pelaksana maksimum yang dialokasikan, yang bergantung pada jumlah DPU yang Anda alokasikan untuk tugas tersebut. Dalam kasus ini, Anda mengalokasikan 10 DPU untuk eksekusi tugas tersebut. Satu DPU dicadangkan untuk pengelolaan. Sembilan DPU menjalankan masing-masing dua pelaksana dan satu pelaksana dicadangkan untuk driver Spark. Driver Spark berjalan dalam aplikasi utama. Jadi, jumlah maksimum pelaksana yang dialokasikan adalah 2*9 - 1 = 17 pelaksana.



Sebagaimana ditunjukkan dalam grafik, jumlah maksimum pelaksana yang diperlukan mulai pada 107 pada awal tugas, sedangkan jumlah pelaksana aktif tetap 17. Jumlah ini sama dengan jumlah maksimum pelaksana yang dialokasikan dengan 10 DPU. Rasio antara jumlah maksimum pelaksana yang dibutuhkan dan jumlah maksimum pelaksana yang dialokasikan (menambahkan 1 ke kedua driver Spark) memberi Anda faktor kurang-penyediaan: 108/18 = 6x. Anda dapat menyediakan 6 (rasio kurang-penyediaan) *9 (kapasitas DPU saat ini - 1) + 1 DPU = 55 DPU untuk menskalakan keluar tugas untuk menjalankannya dengan paralelisme maksimum dan selesai dengan waktu lebih cepat.

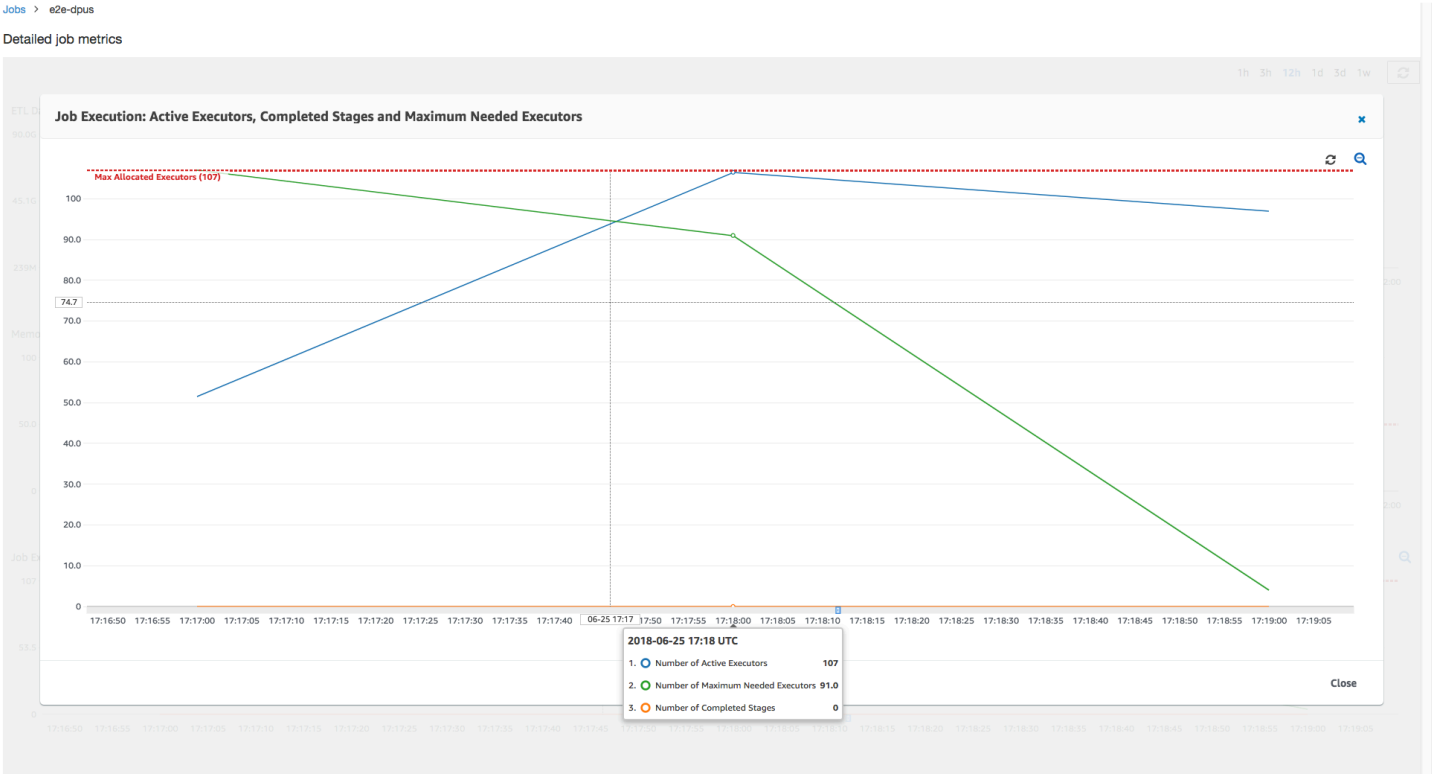
Konsol AWS Glue menampilkan metrik tugas detail sebagai garis statis yang mewakili jumlah asli dari jumlah maksimum pelaksana yang dialokasikan. Konsol menghitung pelaksana maksimum yang dialokasikan dari definisi tugas untuk metrik. Sebaliknya, untuk metrik eksekusi tugas detail, konsol menghitung jumlah maksimum pelaksana dialokasikan dari konfigurasi eksekusi tugas, khususnya DPU yang dialokasikan untuk eksekusi tugas. Untuk melihat metrik masing-masing eksekusi tugas, pilih sebuah eksekusi tugas dan pilih Lihat metrik eksekusi.



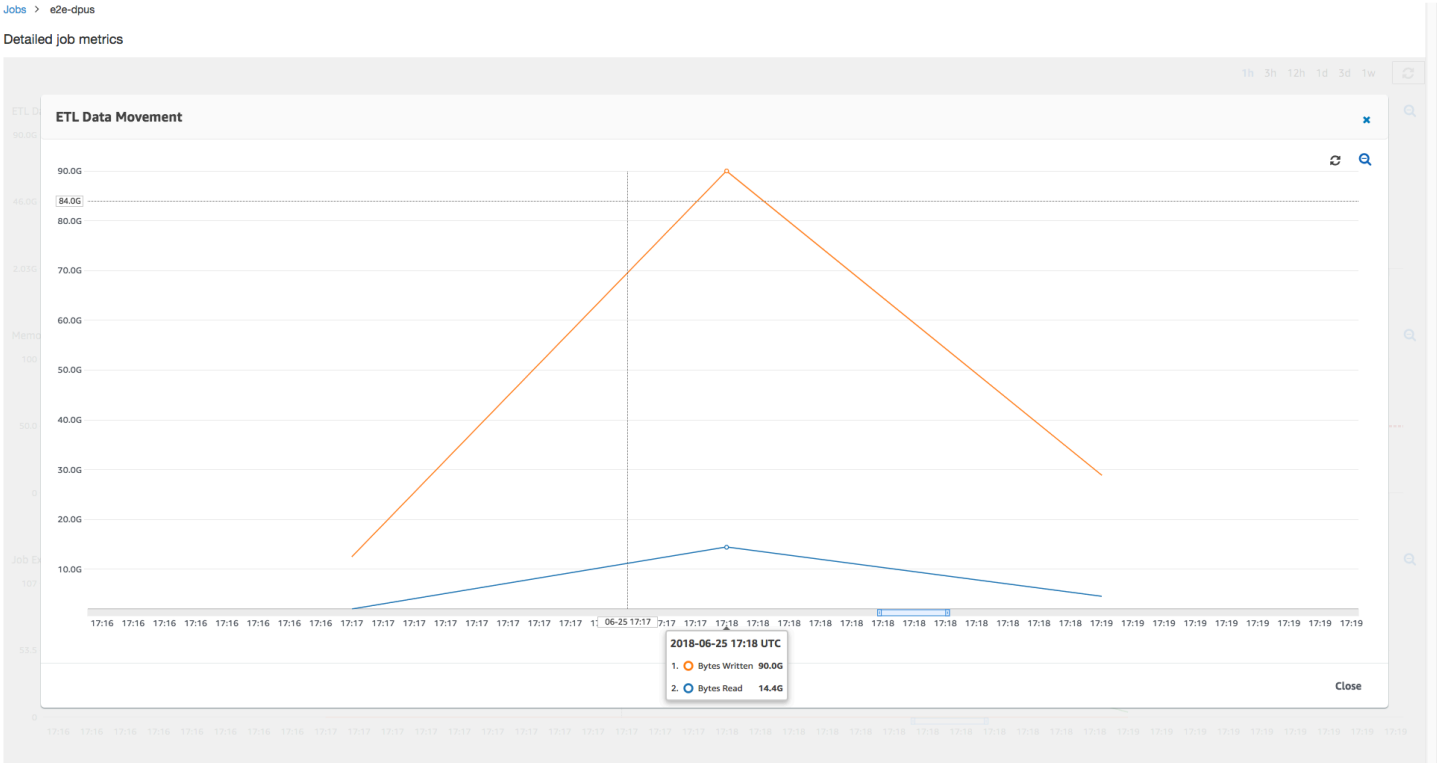
Melihat byte Amazon S3 yang di-baca dan di-tulis, perhatikan bahwa tugas menghabiskan waktu enam menit streaming data dari Amazon S3 dan menuliskannya secara paralel. Semua inti pada DPU yang dialokasikan membaca dan menulis ke Amazon S3. Jumlah maksimum pelaksana yang dibutuhkan menjadi 107 juga cocok dengan jumlah file dalam input Amazon S3 path—yakni 428. Setiap pelaksana dapat meluncurkan empat tugas Spark untuk memproses empat file input (JSON gzip).

Tentukan kapasitas DPU yang optimal

Berdasarkan hasil dari eksekusi tugas sebelumnya, Anda dapat meningkatkan jumlah total DPU yang dialokasikan menjadi 55, dan melihat bagaimana performa tugas tersebut. Tugas selesai dalam waktu kurang dari tiga menit—setengah dari waktu yang dibutuhkan sebelumnya. Penskalaan keluar tugas tidak linear dalam hal ini karena eksekusi tugas merupakan eksekusi tugas singkat. Pekerjaan dengan tugas yang berumur panjang atau sejumlah besar tugas (sejumlah besar pelaksana maksimum yang dibutuhkan) mendapat manfaat dari percepatan kinerja skala close-to-linear DPU.



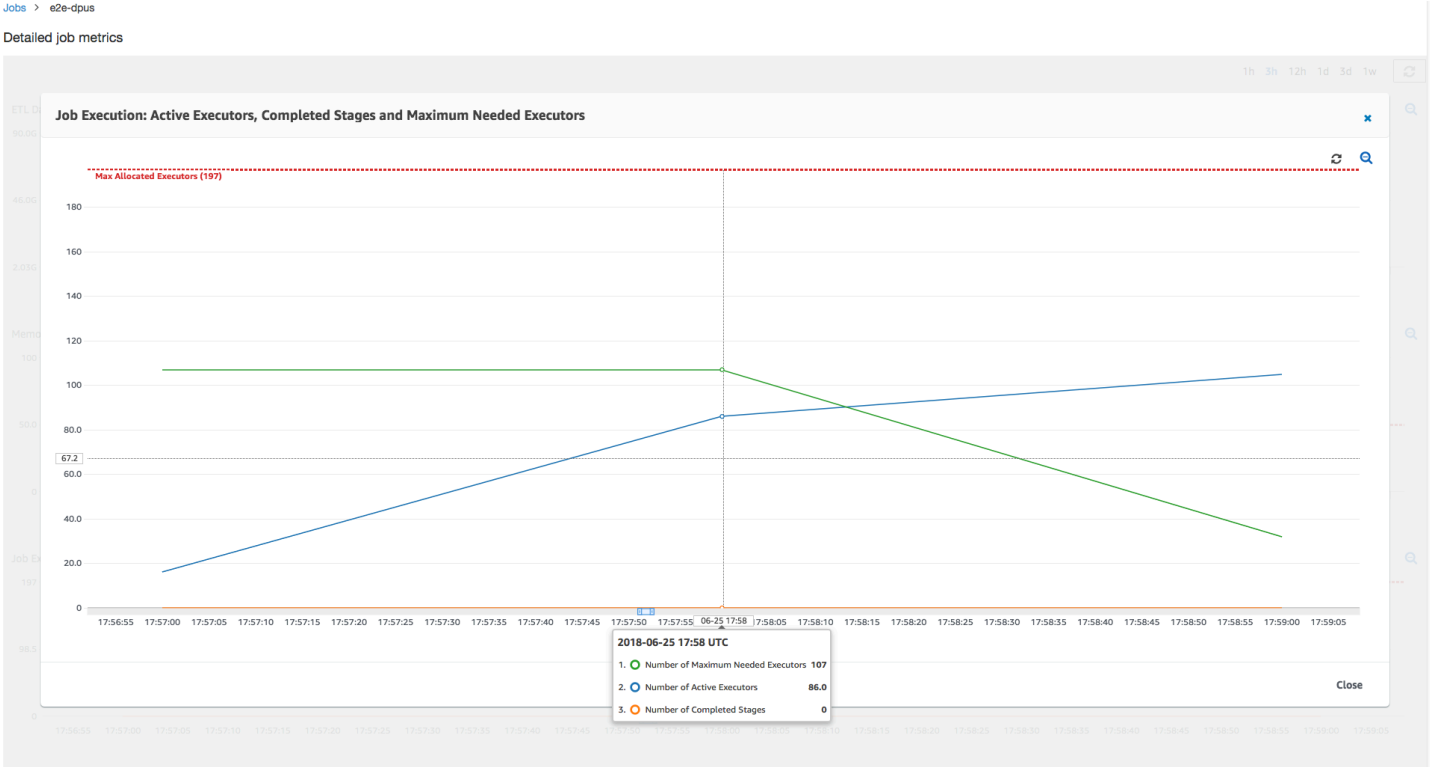
Sebagaimana yang ditunjukkan gambar di atas, jumlah pelaksana aktif mencapai jumlah pelaksana maksimum yang dialokasikan—yakni, 107 pelaksana. Demikian pula, jumlah pelaksana maksimum yang dibutuhkan tidak pernah berada di atas jumlah maksimum pelaksana yang dialokasikan. Jumlah maksimum pelaksana yang dibutuhkan dihitung dari jumlah tugas yang aktif berjalan dan tertunda, sehingga mungkin lebih kecil dari jumlah pelaksana aktif. Hal ini karena dapat ada pelaksana yang sebagian atau seluruhnya menganggur dalam waktu singkat dan belum dinonaktifkan.



Eksekusi tugas ini menggunakan 6x pelaksana lebih untuk membaca dan menulis dari Amazon S3 secara paralel. Akibatnya, eksekusi tugas ini menggunakan lebih banyak bandwidth Amazon S3 untuk membaca dan menulis, dan selesai lebih cepat.

Identifikasi DPU yang dilebih-lebihkan

Selanjutnya, Anda dapat menentukan apakah penskalaan keluar tugas dengan 100 DPU (99 * 2 = 198 pelaksana) membantu untuk melakukan penskalaan keluar lebih lanjut. Seperti yang ditunjukkan dalam grafik berikut, tugas masih membutuhkan waktu tiga menit untuk selesai. Demikian pula, tugas tidak menskalakan keluar melampaui 107 pelaksana (55 konfigurasi DPU), dan tersisa 91 pelaksana di-sediakan-berlebih dan tidak digunakan sama sekali. Hal ini menunjukkan bahwa meningkatkan jumlah DPU mungkin tidak selalu meningkatkan performa, sebagaimana terbukti dari pelaksana maksimum yang dibutuhkan.



Bandingkan perbedaan waktu

Tiga eksekusi tugas ditunjukkan dalam tabel berikut menunjukkan waktu eksekusi tugas untuk 10 DPU, 55 DPU, dan 100 DPU. Anda dapat menemukan bahwa kapasitas DPU untuk meningkatkan waktu pelaksanaan tugas menggunakan perkiraan yang Anda buat dengan memantau tugas pertama.

ID Tugas	Jumlah DPU	Waktu eksekusi
jr_c894524c8ef5048a4d9...	10	6 menit.
jr_1a466cf2575e7ffe6856...	55	3 menit.
jr_34fa1ed4c6aa9ff0a814...	100	3 menit.

Lowongan kerja Streaming ETL di AWS Glue

Anda dapat membuat tugas extract, transform, and load (ETL) yang berjalan terus menerus, mengkonsumsi data dari sumber streaming seperti Amazon Kinesis Data Streams, Apache Kafka, dan Amazon Managed Streaming for Apache Kafka (Amazon MSK). Tugas tersebut membersihkan

dan mengubah data, dan kemudian memuat hasil ke danau data Amazon S3 atau penyimpanan data JDBC.

Selain itu, Anda dapat menghasilkan data untuk Amazon Kinesis Data Streams. Fitur ini hanya tersedia saat menulis AWS Glue skrip. Untuk informasi selengkapnya, lihat the section called "Koneksi Kinesis".

Secara default, AWS Glue memproses dan menulis data dalam jendela 100-detik. Hal ini memungkinkan data diproses secara efisien dan memungkinkan agregasi untuk dilakukan pada data yang tiba lebih lambat dari yang diharapkan. Anda dapat memodifikasi ukuran jendela ini untuk meningkatkan ketepatan waktu atau akurasi agregasi. Tugas streaming AWS Glue menggunakan pos pemeriksaan alih-alih bookmark tugas untuk melacak data yang telah dibaca.

> **Note**
>
> AWS Glue mengenakan biaya per jam untuk tugas ETL streaming saat mereka berjalan.

Video ini membahas tantangan biaya streaming ETL, dan fitur penghematan biaya di. AWS Glue

Membuat tugas ETL streaming melibatkan langkah-langkah berikut:

1. Untuk sumber streaming Apache Kafka, buatlah sebuah koneksi AWS Glue ke sumber Kafka atau klaster Amazon MSK.
2. Secara manual, buat tabel Katalog Data untuk sumber streaming.
3. Buatlah sebuah tugas ETL untuk sumber data streaming. Tentukan properti tugas spesifik-streaming, dan sediakan skrip Anda sendiri atau, secara opsional, modifikasi skrip yang dihasilkan.

Untuk informasi selengkapnya, lihat Streaming ETL di AWS Glue.

Saat membuat tugas ETL streaming untuk Amazon Kinesis Data Streams, Anda tidak perlu membuat koneksi AWS Glue. Namun demikian, jika ada koneksi yang dilampirkan pada tugas ETL streaming AWS Glue yang memiliki Kinesis Data Streams sebagai sumber, maka titik akhir virtual private cloud (VPC) untuk Kinesis diperlukan. Untuk informasi selengkapnya, lihat Membuat titik akhir antarmuka dalam Panduan Pengguna Amazon VPC. Saat menentukan pengaliran Amazon Kinesis Data Streams di akun lain, Anda harus menyiapkan peran dan kebijakan untuk memungkinkan akses lintas akun. Untuk informasi selengkapnya, lihat Contoh: Baca Dari Pengaliran Kinesis di Akun Berbeda.

AWS Gluestreaming pekerjaan ETL dapat mendeteksi data terkompresi secara otomatis, mendekompresi data streaming secara transparan, melakukan transformasi biasa pada sumber input, dan memuat ke penyimpanan output.

AWS Gluemendukung dekompresi otomatis untuk jenis kompresi berikut yang diberikan format input:

Jenis kompresi	Berkas Avro	Tanggal Avro	JSON	CSV	Grok
BZIP2	Ya	Ya	Ya	Ya	Ya
GZIP	Tidak	Ya	Ya	Ya	Ya
SNAPPY	Ya (Snappy mentah)	Ya (berbingkai Snappy)	Ya (berbingkai Snappy)	Ya (berbingkai Snappy)	Ya (berbingkai Snappy)
XZ	Ya	Ya	Ya	Ya	Ya
ZSTD	Ya	Tidak	Tidak	Tidak	Tidak
DEFLATE	Ya	Ya	Ya	Ya	Ya

Topik

- Membuat AWS Glue koneksi untuk aliran data Apache Kafka
- Membuat tabel Katalog Data untuk sumber streaming
- Catatan dan batasan untuk sumber streaming Avro
- Menerapkan pola grok ke sumber streaming
- Mendefinisikan properti pekerjaan untuk pekerjaan ETL streaming
- Streaming catatan dan batasan ETL

Membuat AWS Glue koneksi untuk aliran data Apache Kafka

Untuk membaca dari pengaliran Apache Kafka, Anda harus membuat sebuah koneksi AWS Glue.

Untuk membuat sebuah koneksi AWS Glue untuk sumber Kafka (konsol)

1. Buka AWS Glue konsol di https://console.aws.amazon.com/glue/.

2. Pada panel navigasi, di Katalog data, pilih Koneksi.
3. Pilih Tambahkan koneksi, dan pada halaman Siapkan properti koneksi Anda, masukkan nama koneksi.

> **Note**
>
> Untuk informasi selengkapnya tentang menentukan properti koneksi, lihat properti AWS Glue koneksi. .

4. Untuk Jenis koneksi, pilih Kafka.
5. Untuk URL server bootstrap Kafka, masukkan host dan nomor port untuk broker bootstrap untuk klaster Amazon MSK atau klaster Apache Kafka. Gunakan hanya titik akhir Keamanan Lapisan Pengangkutan (TLS) untuk membangun koneksi awal ke klaster Kafka. Tidak mendukung titik akhir teks biasa.

   Berikut ini adalah contoh daftar pasangan nama host dan nomor port untuk klaster Amazon MSK.

   ```
   myserver1.kafka.us-east-1.amazonaws.com:9094,myserver2.kafka.us-
   east-1.amazonaws.com:9094,
   myserver3.kafka.us-east-1.amazonaws.com:9094
   ```

   Untuk informasi selengkapnya tentang mendapatkan informasi broker bootstrap, lihat Mendapatkan Bootstrap Broker untuk Klaster Amazon MSK dalam Panduan Developer Amazon Managed Streaming for Apache Kafka.
6. Jika Anda ingin koneksi yang aman ke sumber data Kafka, pilih Memerlukan koneksi SSL, dan untuk Lokasi sertifikat CA privat Kafka, masukkan path Amazon S3 yang valid ke sebuah sertifikat SSL khusus.

   Untuk koneksi SSL ke Kafka dikelola sendiri, sertifikat khusus tersebut bersifat wajib. Ini bersifat opsional untuk Amazon MSK.

   Untuk informasi selengkapnya tentang menentukan sebuah sertifikat khusus untuk Kafka, lihat the section called “Properti koneksi SSL”.
7. Gunakan AWS Glue Studio atau AWS CLI untuk menentukan metode otentikasi klien Kafka. Untuk mengakses AWS Glue Studio pilih AWS Gluedari menu ETL di panel navigasi kiri.

Untuk informasi selengkapnya tentang metode otentikasi klien Kafka, lihat. AWS GlueProperti koneksi Kafka untuk otentikasi klien

8. Secara opsional, masukkan deskripsi, dan kemudian pilih Selanjutnya.
9. Untuk klaster Amazon MSK, tentukan virtual private cloud (VPC), subnet, dan grup keamanan dari klaster tersebut. Informasi VPC bersifat opsional untuk Kafka yang dikelola sendiri.
10. Pilih Selanjutnya untuk meninjau semua properti koneksi, dan kemudian pilih Selesai.

Untuk informasi selengkapnya tentang koneksi AWS Glue, lihat Menghubungkan ke data.

AWS GlueProperti koneksi Kafka untuk otentikasi klien

Otentikasi SASL/GSSAPI (Kerberos)

Memilih metode otentikasi ini akan memungkinkan Anda untuk menentukan properti Kerberos.

Kerberos Keytab

Pilih lokasi file tab tombol. Keytab menyimpan kunci jangka panjang untuk satu atau lebih prinsipal. Untuk informasi selengkapnya, lihat Dokumentasi MIT Kerberos: Keytab.

Kerberos berkas krb5.conf

Pilih file krb5.conf. Ini berisi ranah default (jaringan logis, mirip dengan domain, yang mendefinisikan sekelompok sistem di bawah KDC yang sama) dan lokasi server KDC. Untuk informasi lebih lanjut, lihat Dokumentasi MIT Kerberos: krb5.conf.

Kepala sekolah Kerberos dan nama layanan Kerberos

Masukkan nama kepala dan layanan Kerberos. Untuk informasi lebih lanjut, lihat Dokumentasi MIT Kerberos: Kepala sekolah Kerberos.

Otentikasi SASL/SCRAM-SHA-512

Memilih metode otentikasi ini akan memungkinkan Anda untuk menentukan kredensyal otentikasi.

AWS Secrets Manager

Cari token Anda di kotak Pencarian dengan mengetikkan nama atau ARN.

Nama pengguna dan kata sandi penyedia secara langsung

Cari token Anda di kotak Pencarian dengan mengetikkan nama atau ARN.

Otentikasi klien SSL

Memilih metode otentikasi ini memungkinkan Anda untuk memilih lokasi keystore klien Kafka dengan menjelajahi Amazon S3. Secara opsional, Anda dapat memasukkan kata sandi keystore klien Kafka dan kata sandi kunci klien Kafka.

Autentikasi IAM

Metode otentikasi ini tidak memerlukan spesifikasi tambahan dan hanya berlaku jika sumber Streaming adalah MSK Kafka.

Otentikasi SASL/PLAIN

Memilih metode otentikasi ini memungkinkan Anda untuk menentukan kredensyal otentikasi.

Membuat tabel Katalog Data untuk sumber streaming

Tabel Katalog Data yang menentukan properti aliran data sumber, termasuk skema data dapat dibuat secara manual untuk sumber streaming. Tabel ini digunakan sebagai sumber data untuk tugas ETL streaming.

Jika Anda tidak tahu skema data dalam aliran data sumber, maka Anda dapat membuat tabel tanpa skema. Kemudian ketika Anda membuat tugas ETL streaming, Anda dapat mengaktifkan fungsi deteksi skema AWS Glue. AWS Glue akan menentukan skema dari data streaming.

Gunakan AWS Gluekonsol, AWS Command Line Interface (AWS CLI), atau AWS Glue API untuk membuat tabel. Untuk informasi selengkapnya tentang cara membuat sebuah tabel secara manual dengan konsol AWS Glue, lihat the section called “Membuat tabel”.

> **Note**
>
> Anda tidak dapat menggunakan AWS Lake Formation konsol untuk membuat tabel; Anda harus menggunakan AWS Glue konsol.

Juga mempertimbangkan informasi berikut untuk sumber streaming dalam format Avro atau untuk data log yang dapat Anda terapkan pola Grok untuknya.

- the section called “Catatan dan batasan untuk sumber streaming Avro”
- the section called “Menerapkan pola grok ke sumber streaming”

Topik

- Sumber data Kinesis
- Sumber data Kafka
- AWS GlueSumber tabel Schema Registry

Sumber data Kinesis

Saat membuat tabel, atur properti ETL streaming berikut (konsol).

Jenis Sumber

Kinesis

Untuk sebuah sumber Kinesis pada akun yang sama:

Wilayah

AWS Wilayah tempat layanan Amazon Kinesis Data Streams berada. Nama pengaliran Wilayah dan Kinesis bersama-sama diterjemahkan ke Stream ARN.

Contoh: https://kinesis.us-east-1.amazonaws.com

Nama pengaliran Kinesis

Nama pengaliran seperti yang dijelaskan dalam Membuat Pengaliran dalam Panduan Developer Amazon Kinesis Data Streams.

Untuk sumber Kinesis di akun lain, lihat contoh ini untuk menyiapkan peran dan kebijakan untuk memungkinkan akses lintas akun. Konfigurasikan pengaturan ini:

ARN Streaming

ARN dari pengaliran data Kinesis dimana konsumen terdaftar. Untuk informasi selengkapnya, lihat Amazon Resource Names (ARN) dan AWS Service Namespaces di. Referensi Umum AWS

ARN Peran yang Diambil

Amazon Resource Name (ARN) dari peran yang diambil.

Nama sesi (opsional)

Sebuah pengenal untuk sesi peran yang diambil.

Gunakan nama sesi peran untuk mengidentifikasi sebuah sesi secara unik ketika peran yang sama diambil oleh prinsipal utama yang berbeda atau untuk alasan yang berbeda. Dalam skenario lintas akun, nama sesi peran dapat dilihat, dan dapat dicatat oleh akun yang memiliki peran tersebut. Nama sesi peran juga digunakan dalam ARN dari prinsipal utama peran yang diambil. Ini berarti bahwa permintaan API lintas akun berikutnya yang menggunakan kredensyal keamanan sementara akan mengekspos nama sesi peran ke akun eksternal di log mereka. AWS CloudTrail

Mengkonfigurasi properti ETL streaming untuk Amazon Kinesis Data Streams (API AWS Glue atau AWS CLI)

- Untuk menyiapkan properti ETL streaming untuk sebuah sumber Kinesis di akun yang sama, tentukan parameter `streamName` dan `endpointUrl` dalam struktur `StorageDescriptor` dari operasi API `CreateTable` atau fungsi perintah CLI `create_table`.

```
"StorageDescriptor": {
 "Parameters": {
  "typeOfData": "kinesis",
  "streamName": "sample-stream",
  "endpointUrl": "https://kinesis.us-east-1.amazonaws.com"
 }
 ...
}
```

  Atau, tentukan `streamARN`.

  Example

```
"StorageDescriptor": {
 "Parameters": {
  "typeOfData": "kinesis",
  "streamARN": "arn:aws:kinesis:us-east-1:123456789:stream/sample-stream"
 }
 ...
}
```

- Untuk menyiapkan properti ETL streaming untuk sebuah sumber Kinesis di akun lain, tentukan parameter `streamARN`, `awsSTSRoleARN` dan `awsSTSSessionName` (opsional) dalam struktur `StorageDescriptor` dari operasi API `CreateTable` atau fungsi perintah CLI `create_table`.

```
"StorageDescriptor": {
 "Parameters": {
  "typeOfData": "kinesis",
  "streamARN": "arn:aws:kinesis:us-east-1:123456789:stream/sample-stream",
  "awsSTSRoleARN": "arn:aws:iam::123456789:role/sample-assume-role-arn",
  "awsSTSSessionName": "optional-session"
 }
 ...
}
```

Sumber data Kafka

Saat membuat tabel, atur properti ETL streaming berikut (konsol).

Jenis Sumber

Kafka

Untuk sebuah sumber Kafka:

Nama topik

Nama topik sebagaimana ditentukan dalam Kafka.

Koneksi

Sebuah koneksi AWS Glue yang me-referensi sumber Kafka, seperti yang dijelaskan dalam the section called “Membuat koneksi untuk aliran data Kafka”.

AWS GlueSumber tabel Schema Registry

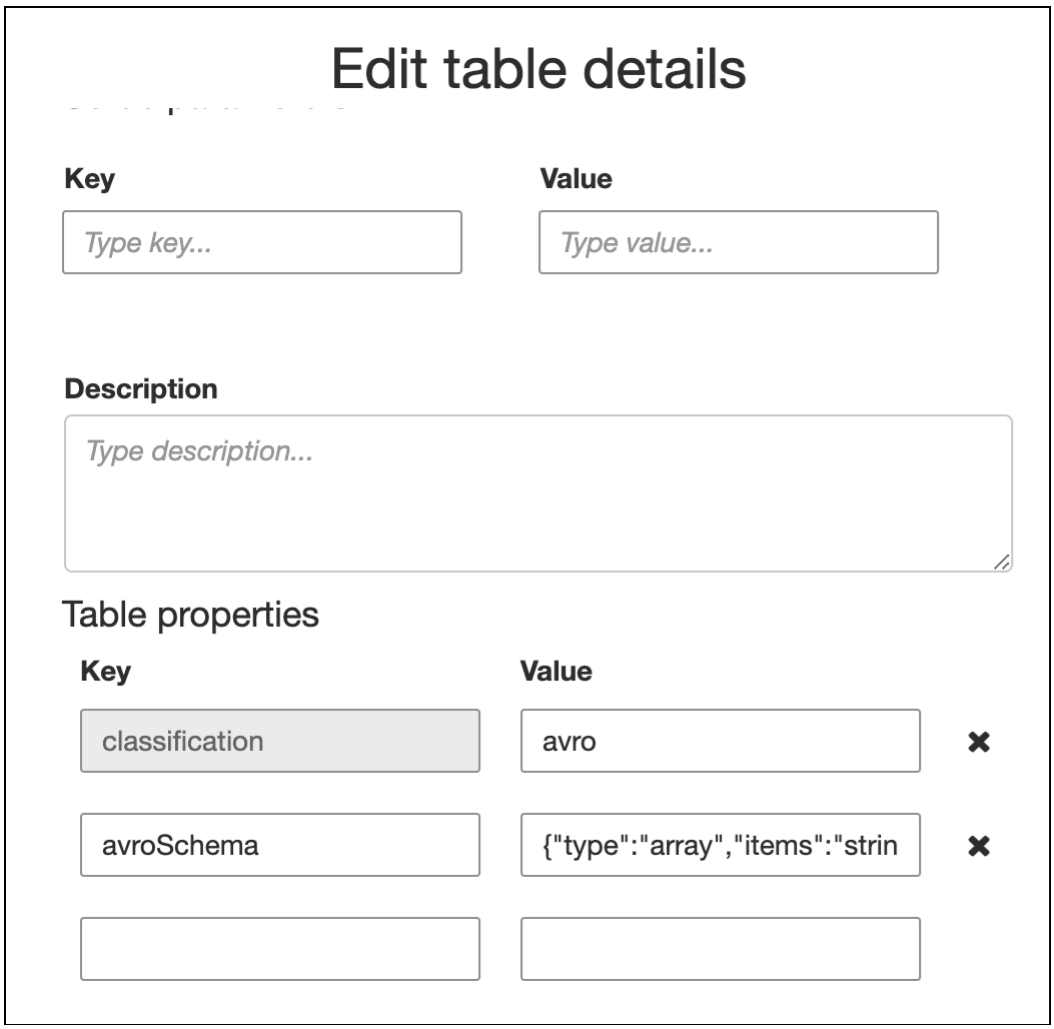
Untuk menggunakan AWS Glue Schema Registry untuk pekerjaan streaming, ikuti petunjuk di Kasus penggunaan: AWS Glue Data Catalog untuk membuat atau memperbarui tabel Schema Registry.

Saat ini, AWS Glue Streaming hanya mendukung format Glue Schema Registry Avro dengan inferensi skema disetel ke. `false`

Catatan dan batasan untuk sumber streaming Avro

Catatan dan pembatasan berikut berlaku untuk sumber streaming dalam format Avro:

- Ketika skema deteksi diaktifkan, skema Avro tersebut harus disertakan dalam muatan. Saat dimatikan, muatan hanya berisi data saja.
- Beberapa tipe data Avro tidak didukung dalam bingkai dinamis. Anda tidak dapat menentukan tipe data ini ketika mendefinisikan skema di halaman Menentukan skema di penuntun membuat tabel di konsol AWS Glue. Selama deteksi skema, jenis yang tidak didukung dalam skema Avro dikonversi ke jenis yang didukung sebagai berikut:
  - `EnumType => StringType`
  - `FixedType => BinaryType`
  - `UnionType => StructType`
- Jika Anda menentukan skema tabel menggunakan halaman Menentukan skema di konsol, maka jenis elemen akar tersirat untuk skema adalah `record`. Jika Anda ingin jenis elemen akar selain `record`, misalnya `array` atau `map`, Anda tidak dapat menentukan skema menggunakan halaman Menentukan skema. Sebaliknya Anda harus melewatkan halaman itu dan menentukan skema baik sebagai sebuah properti tabel atau dalam skrip ETL.
  - Untuk menentukan skema di properti tabel, selesaikan penuntun membuat tabel, edit detail tabel, dan tambahkan pasangan nilai-kunci baru pada Properti tabel. Gunakan kunci `avroSchema`, dan masukkan objek skema JSON untuk nilai, seperti yang ditunjukkan pada tangkapan layar berikut.



- Untuk menentukan skema dalam skrip ETL, modifikasi pernyataan tugas `datasource0` dan tambahkan kunci `avroSchema` ke argumen `additional_options`, seperti yang ditunjukkan dalam contoh Python dan Scala berikut.

  Python

  ```
  SCHEMA_STRING = '{"type":"array","items":"string"}'
  datasource0 = glueContext.create_data_frame.from_catalog(database =
   "database", table_name = "table_name", transformation_ctx = "datasource0",
   additional_options = {"startingPosition": "TRIM_HORIZON", "inferSchema":
   "false", "avroSchema": SCHEMA_STRING})
  ```

  Scala

  ```
  val SCHEMA_STRING = """{"type":"array","items":"string"}"""
  val datasource0 = glueContext.getCatalogSource(database = "database", tableName
   = "table_name", redshiftTmpDir = "", transformationContext = "datasource0",
  ```

```
additionalOptions = JsonOptions(s"""{"startingPosition": "TRIM_HORIZON",
"inferSchema": "false", "avroSchema":"$SCHEMA_STRING"}""")).getDataFrame()
```

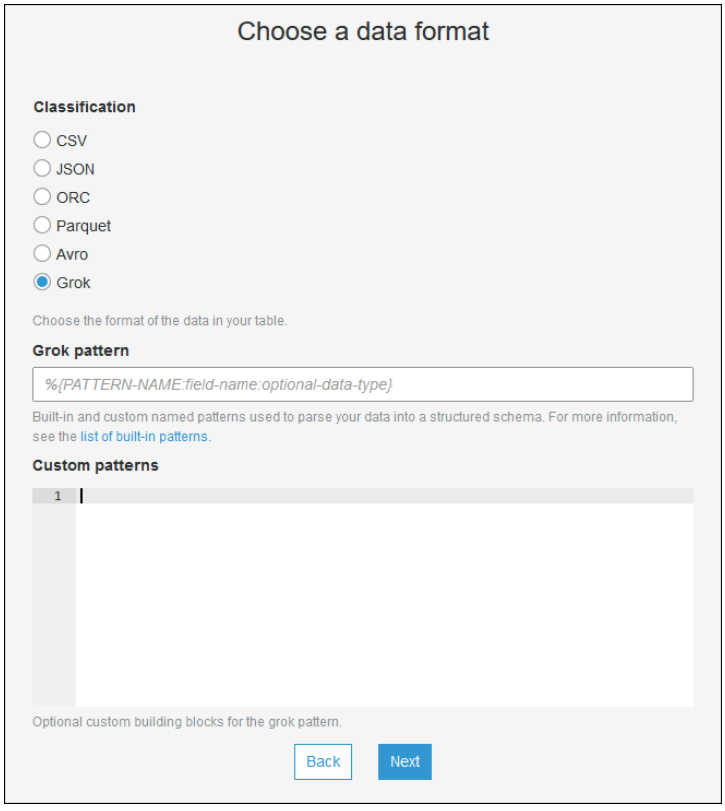
Menerapkan pola grok ke sumber streaming

Anda dapat membuat sebuah tugas ETL streaming untuk sumber data log dan menggunakan pola Grok untuk mengkonversi log menjadi data terstruktur. Tugas ETL tersebut kemudian memproses data sebagai sebuah sumber data terstruktur. Anda menentukan pola Grok untuk diterapkan saat Anda membuat tabel Katalog Data untuk sumber streaming.

Untuk informasi tentang pola Grok dan nilai-nilai string pola kustom, lihat Menulis pengklasifikasi kustom grok.

Untuk menambahkan pola grok ke tabel Katalog Data (konsol)

- Gunakan penuntun membuat tabel, dan buatlah tabel dengan parameter yang ditentukan dalam the section called “Membuat tabel Katalog Data untuk sumber streaming”. Tentukan format data sebagai Grok, isi Pola grok, dan secara opsional, Anda bisa memilih untuk menambahkan pola penyesuaian pada Pola kustom (opsional).



  Tekan Enter setelah setiap pola kustom.

Untuk menambahkan pola grok ke tabel Katalog Data (AWS GlueAPI atau AWS CLI)

- Tambahkan parameter `GrokPattern` dan, secara opsional, parameter `CustomPatterns` ke operasi API `CreateTable` atau fungsi perintah CLI `create_table`.

```
 "Parameters": {
...
    "grokPattern": "string",
    "grokCustomPatterns": "string",
...
},
```

Ekspresikan `grokCustomPatterns` sebagai string dan gunakan "\n" sebagai pemisah antara pola.

Berikut ini adalah contoh cara menentukan parameter ini.

Example

```
"parameters": {
...
    "grokPattern": "%{USERNAME:username} %{DIGIT:digit:int}",
    "grokCustomPatterns": "digit \d",
...
}
```

Mendefinisikan properti pekerjaan untuk pekerjaan ETL streaming

Bila Anda menentukan tugas ETL streaming di konsol AWS Glue, berikan properti spesifik-pengaliran berikut ini. Untuk deskripsi tentang properti tugas tambahan, lihat Mendefinisikan properti pekerjaan untuk pekerjaan Spark.

Peran IAM

Tentukan peran AWS Identity and Access Management (IAM) yang digunakan untuk otorisasi sumber daya yang digunakan untuk menjalankan pekerjaan, mengakses sumber streaming, dan mengakses penyimpanan data target.

Untuk akses ke Amazon Kinesis Data Streams, `AmazonKinesisFullAccess` AWS lampirkan kebijakan terkelola ke peran, atau lampirkan kebijakan IAM serupa yang mengizinkan akses yang

lebih halus. Untuk kebijakan sampel, lihat Mengontrol Akses ke Amazon Kinesis Data Streams Menggunakan IAM.

Untuk informasi selengkapnya tentang izin menjalankan tugas di AWS Glue, lihat Manajemen identitas dan akses untuk AWS Glue.

Tipe

Pilih Spark streaming.

Versi AWS Glue

Versi AWS Glue menentukan versi Apache Spark, dan Python atau Scala, yang tersedia untuk tugas. Pilih pilihan yang menentukan versi Python atau Scala yang tersedia untuk pekerjaan itu. AWS Glue Versi 2.0 dengan support Python 3 adalah default untuk streaming tugas ETL.

Periode pemeliharaan

Menentukan jendela di mana pekerjaan streaming dapat dimulai ulang. Lihat the section called “Jendela pemeliharaan”.

Tugas habis waktu

Opsional, masukkan durasi dalam menit. Nilai defaultnya kosong.

- Pekerjaan streaming harus memiliki nilai batas waktu kurang dari 7 hari atau 10080 menit.
- Ketika nilai dibiarkan kosong, pekerjaan akan dimulai ulang setelah 7 hari, jika Anda belum menyiapkan jendela pemeliharaan. Jika Anda telah menyiapkan jendela pemeliharaan, pekerjaan akan dimulai kembali selama jendela pemeliharaan setelah 7 hari.

Sumber data

Tentukan tabel yang Anda buat di the section called “Membuat tabel Katalog Data untuk sumber streaming”.

Target data

Lakukan salah satu hal berikut ini:

- Pilih Buat tabel di target data Anda dan tentukan properti target data berikut.

  Penyimpanan data

  Pilih Amazon S3 atau JDBC.

  format

  Pilih format apa saja. Semua didukung untuk streaming.

- Pilih Gunakan tabel dalam katalog data dan perbarui target data, dan pilih sebuah tabel untuk menyimpan data JDBC.

Definisi skema output

Lakukan salah satu hal berikut ini:

- Pilih Secara otomatis mendeteksi skema dari setiap catatan untuk mengaktifkan deteksi skema. AWS Glue menentukan skema dari data streaming.
- Pilih Tentukan skema output untuk semua catatan untuk menggunakan transformasi Terapkan Pemetaan untuk menentukan skema output.

Skrip

Opsional, berikan skrip Anda sendiri atau modifikasi skrip yang dihasilkan untuk melakukan operasi yang didukung oleh mesin Apache Spark Structured Streaming. Untuk informasi tentang operasi yang tersedia, lihat Operasi pada streaming DataFrames /Datasets.

Streaming catatan dan batasan ETL

Ingatlah catatan dan pembatasan berikut:

- Dekompresi otomatis untuk AWS Glue streaming pekerjaan ETL hanya tersedia untuk jenis kompresi yang didukung. Perhatikan juga hal berikut:
  - Framed Snappy mengacu pada format pembingkaian resmi untuk Snappy.
  - Deflate didukung di Glue versi 3.0, bukan Glue versi 2.0.
- Bila menggunakan deteksi skema, Anda tidak dapat melakukan penggabungan data streaming.
- AWS Gluestreaming pekerjaan ETL tidak mendukung tipe data Union untuk AWS Glue Schema Registry dengan format Avro.
- Skrip ETL Anda dapat menggunakan AWS Glue transformasi bawaan dan transformasi asli ke Apache Spark Structured Streaming. Untuk informasi selengkapnya, lihat Operasi pada streaming DataFrames /Datasets di situs web Apache Spark atau. AWS Glue PySpark mengubah referensi
- Tugas ETL streaming AWS Glue menggunakan pos pemeriksaan untuk melacak data yang telah dibaca. Oleh karena itu, tugas yang dihentikan dan dimulai-ulang memilih tempat ia pergi di pengaliran. Jika Anda ingin memproses ulang data, maka Anda dapat menghapus folder pos pemeriksaan yang direferensikan dalam skrip.
- Bookmark Tugas tidak didukung.

- Untuk menggunakan fitur fan-out yang disempurnakan dari Kinesis Data Streams dalam pekerjaan Anda, konsultasikan. the section called “Menggunakan fan-out yang disempurnakan dalam pekerjaan streaming Kinesis”
- Jika Anda menggunakan tabel Katalog Data yang dibuat dari AWS Glue Schema Registry, ketika versi skema baru tersedia, untuk mencerminkan skema baru, Anda perlu melakukan hal berikut:
  1. Hentikan pekerjaan yang terkait dengan tabel.
  2. Perbarui skema untuk tabel Katalog Data.
  3. Mulai ulang pekerjaan yang terkait dengan tabel.

Rekam pencocokan dengan AWS Lake Formation FindMatches

> **Note**
>
> Pencocokan rekaman saat ini tidak tersedia di Wilayah berikut di AWS Glue konsol: Timur Tengah (UEA), Eropa (Spanyol) (`eu-south-2`), dan Eropa (Zurich) (). `eu-central-2`

AWS Lake Formation menyediakan kemampuan machine learning untuk membuat transformasi kustom untuk membersihkan data Anda. Saat ini ada satu transformasi yang tersedia bernama FindMatches. FindMatchesTransformasi memungkinkan Anda mengidentifikasi duplikat atau pencocokan catatan dalam kumpulan data Anda, bahkan ketika catatan tidak memiliki pengidentifikasi unik umum dan tidak ada bidang yang sama persis. Ini tidak memerlukan penulisan kode apa pun atau mengetahui cara kerja pembelajaran mesin. FindMatches dapat berguna dalam berbagai masalah, seperti:

- Mencocokkan pelanggan: Menautkan catatan pelanggan di berbagai basis data pelanggan, bahkan ketika banyak bidang pelanggan tidak cocok persis di seluruh basis data (misalnya ejaan nama yang berbeda, perbedaan alamat, data yang hilang atau tidak akurat, dll).
- Pencocokan produk: Mencocokkan produk dalam katalog Anda dengan sumber produk lain, seperti katalog produk dengan katalog pesaing, di mana entri disusun secara berbeda.
- Meningkatkan deteksi penipuan: Mengidentifikasi akun pelanggan duplikat, menentukan kapan akun yang baru dibuat (atau mungkin) cocok untuk pengguna penipuan yang diketahui sebelumnya.

- Masalah pencocokan lainnya: Alamat pencocokan, film, daftar bagian, dll. Secara umum, jika seorang manusia dapat melihat baris database Anda dan menentukan bahwa mereka cocok, ada kemungkinan besar bahwa FindMatches transformasi dapat membantu Anda.

Anda dapat membuat transformasi ini saat membuat sebuah tugas. Transformasi yang Anda buat didasarkan pada skema penyimpanan data sumber dan contoh data dari kumpulan data sumber yang Anda beri label (kami menyebut proses ini "mengajar" transformasi). Catatan yang Anda beri label harus ada dalam kumpulan data sumber. Dalam proses ini kami membuat file yang Anda beri label dan kemudian mengunggah kembali yang akan dipelajari transformasi dengan cara tertentu. Setelah Anda mengajarkan transformasi Anda, Anda dapat memanggilnya dari AWS Glue pekerjaan berbasis Spark Anda (PySpark atau Scala Spark) dan menggunakannya dalam skrip lain dengan penyimpanan data sumber yang kompatibel.

Setelah transformasi dibuat, ia disimpan dalam AWS Glue. Pada konsol AWS Glue, Anda dapat mengelola transformasi yang Anda buat. Di panel navigasi di bawah Integrasi Data dan ETL, Alat klasifikasi data > Pencocokan Rekam, Anda dapat mengedit dan terus mengajarkan transformasi pembelajaran mesin Anda. Untuk informasi selengkapnya tentang mengelola transformasi pada konsol tersebut, lihat Bekerja dengan transformasi pembelajaran mesin di konsol AWS Glue.

> **Note**
>
> AWS Glue FindMatches pekerjaan versi 2.0 menggunakan bucket Amazon S3 `aws-glue-temp-<accountID>-<region>` untuk menyimpan file sementara saat transformasi memproses data. Anda dapat menghapus data ini setelah proses selesai, baik secara manual atau dengan menyetel aturan Siklus Hidup Amazon S3.

Jenis transformasi pembelajaran mesin

Anda dapat membuat transformasi machine learning untuk membersihkan data Anda. Anda dapat memanggil transformasi ini dari skrip ETL Anda. Data Anda berpindah dari transformasi ke transformasi dalam struktur data yang disebut a DynamicFrame, yang merupakan ekstensi ke Apache Spark SQL. `DataFrame` `DynamicFrame` tersebut berisi data Anda, dan Anda me-referensi skemanya untuk memproses data Anda.

Jenis transformasi machine learning berikut sudah tersedia:

Menemukan kecocokan

Menemukan duplikat catatan dalam data sumber. Anda mengajarkan transformasi machine learning ini dengan melabelkan set data contoh untuk menunjukkan baris yang cocok. Transformasi machine learning tersebut akan mempelajari baris mana yang seharusnya cocok dengan semakin banyak Anda mengajarinya dengan data contoh berlabel. Tergantung cara Anda mengkonfigurasi transformasi, outputnya adalah salah satu dari berikut ini:

- Salinan tabel masukan dan kolom `match_id` yang telah diisi dengan nilai-nilai yang menunjukkan sekumpulan catatan yang cocok. Kolom `match_id` adalah pengenal sebarang. Setiap catatan yang memiliki `match_id` yang sama telah diidentifikasi sebagai kecocokan satu sama lain. Catatan dengan `match_id` berbeda tidak cocok.
- Salinan tabel masukan dengan duplikat baris dihapus. Jika beberapa duplikat ditemukan, maka catatan dengan kunci primer terendah yang disimpan.

Temukan kecocokan tambahan

Transformasi Find Matches juga dapat dikonfigurasi untuk menemukan kecocokan di seluruh frame yang ada dan inkremental dan mengembalikan sebagai output kolom yang berisi ID unik per grup kecocokan.

Untuk informasi lebih lanjut, lihat: Menemukan kecocokan tambahan

Menggunakan FindMatches transformasi

Anda dapat menggunakan transformasi `FindMatches` untuk menemukan catatan duplikat dalam data sumber. Sebuah file pelabelan dihasilkan atau disediakan untuk membantu mengajarkan transformasi.

> **Note**
>
> Saat ini, `FindMatches` transformasi yang menggunakan kunci enkripsi kustom tidak didukung di Wilayah berikut:
>
> - Asia Pasifik (Osaka) - `ap-northeast-3`

Untuk memulai FindMatches transformasi, Anda dapat mengikuti langkah-langkah di bawah ini. Untuk contoh yang lebih maju dan terperinci, lihat AWSBig Data Blog: Harmonize data using AWS Glue dan AWS Lake Formation FindMatches ML untuk membangun tampilan pelanggan 360.

Memulai menggunakan transformasi Find Matches

Ikuti langkah-langkah ini untuk memulai transformasi `FindMatches`:

1. Buat tabel di AWS Glue Data Catalog untuk data sumber yang harus dibersihkan. Untuk informasi tentang cara membuat crawler, lihat Bekerja dengan Crawler di Konsol AWS Glue.

   Jika data sumber adalah file berbasis teks seperti file nilai yang dipisahkan oleh koma (CSV), pertimbangkan hal berikut:

   - Simpan file CSV catatan input Anda dan file pelabelan di folder terpisah. Jika tidak, crawler AWS Glue mungkin akan menganggap mereka sebagai beberapa bagian dari tabel yang sama dan membuat beberapa tabel di Katalog Data dengan tidak semestinya.
   - Kecuali file CSV Anda menyertakan karakter ASCII saja, pastikan bahwa encoding UTF-8 tanpa BOM (byte order mark) digunakan untuk file CSV. Microsoft Excel sering kali menambahkan BOM di awal file CSV UTF-8. Untuk menghapusnya, buka file CSV di editor teks, dan simpan ulang file sebagai UTF-8 tanpa BOM.
2. Pada konsol AWS Glue, buat sebuah tugas, dan pilih transformasi Temukan kecocokan.

   > **Important**
   >
   > Tabel sumber data yang Anda pilih untuk tugas tersebut tidak boleh memiliki lebih dari 100 kolom.
3. Sampaikan pada AWS Glue untuk menghasilkan file pelabelan dengan memilih Buat file pelabelan. AWS Glue akan mengambil sediaan pertama di grup catatan serupa untuk setiap `labeling_set_id` sehingga Anda dapat meninjau pengelompokan tersebut. Anda memberikan label pada kecocokan yang ada di kolom `label`.
   - Jika Anda sudah memiliki file pelabelan, yaitu contoh catatan yang menunjukkan baris yang cocok, maka unggah file tersebut ke Amazon Simple Storage Service (Amazon S3). Untuk informasi tentang format file pelabelan, lihat Format file pelabelan. Lanjutkan ke langkah 4.
4. Unduh file pelabelan dan beri label file tersebut seperti yang diterangkan dalam bagian Pelabelan.
5. Unggah file yang telah diberi label dan telah dikoreksi. AWS Glue menjalankan tugas untuk mengajarkan transformasi bagaimana menemukan kecocokan.

   Halaman daftar Transformasi Machine Learning, pilih tab Riwayat. Halaman ini menunjukkan kapan AWS Glue melakukan tugas-tugas berikut:

   - Impor label

- Label ekspor
- Hasilkan label
- Perkirakan kualitas

6. Untuk membuat transformasi yang lebih baik, Anda dapat secara berulang mengunduh, label, dan mengunggah file yang telah diberi label. Dalam eksekusi awal, lebih banyak catatan mungkin tidak cocok. Tapi AWS Glue akan belajar saat Anda terus mengajarinya dengan memverifikasi file pelabelan.
7. Evaluasi dan setel transformasi Anda dengan mengevaluasi performa dan hasil penemuan kecocokan. Untuk informasi selengkapnya, lihat Pembelajaran mesin tuning berubah di AWS Glue.

Pelabelan

Saat `FindMatches` menghasilkan file pelabelan, catatan dipilih dari tabel sumber Anda. Berdasarkan pelatihan sebelumnya, `FindMatches` mengidentifikasi catatan paling bernilai untuk dipelajari.

Tindakan pelabelan adalah mengedit file pelabelan (sebaiknya gunakan spreadsheet seperti Microsoft Excel) dan menambahkan pengidentifikasi, atau label, ke kolom `label` yang mengidentifikasi kecocokan dan ketidakcocokan catatan. Penting bagi Anda untuk memiliki definisi yang jelas dan konsisten atas kecocokan dalam data sumber Anda. `FindMatches` akan belajar dari catatan mana yang Anda tentukan sebagai kecocokan (atau tidak) dan menggunakan keputusan Anda untuk mempelajari cara menemukan catatan duplikat.

Ketika file pelabelan dibuat oleh `FindMatches`, sekitar 100 catatan yang dihasilkan. 100 catatan ini biasanya dibagi menjadi 10 set pelabelan, dimana setiap set pelabelan diidentifikasi berdasarkan `labeling_set_id` yang dihasilkan oleh `FindMatches`. Setiap set pelabelan harus dilihat sebagai tugas pelabelan terpisah independen dari set pelabelan lainnya. Tugas Anda adalah mengidentifikasi kecocokan dan ketidakcocokan catatan dalam setiap set pelabelan.

Kiat untuk mengedit file pelabelan dalam spreadsheet

Saat mengedit file pelabelan dalam sebuah aplikasi spreadsheet, pertimbangkan hal berikut:

- File mungkin tidak terbuka dengan bidang kolom yang diperluas sepenuhnya. Anda mungkin harus memperluas kolom `labeling_set_id` dan `label` untuk melihat konten dalam sel-sel tersebut.
- Jika kolom kunci primer adalah angka, seperti tipe data `long`, maka spreadsheet mungkin akan menafsirkannya sebagai angka dan mengubah nilai-nya. Nilai kunci ini harus diperlakukan sebagai teks. Untuk memperbaiki masalah ini, format semua sel di kolom kunci primer sebagai Data teks.

Format file pelabelan

File pelabelan yang dihasilkan oleh AWS Glue untuk mengajarkan transformasi `FindMatches` Anda menggunakan format berikut. Jika anda membuat file anda sendiri untuk AWS Glue, file tersebut harus mengikuti format ini juga:

- File ini adalah file nilai yang dipisahkan koma (CSV).
- File ini harus dikodekan dalam `UTF-8`. Jika Anda mengedit file menggunakan Microsoft Windows, file mungkin akan dikodekan dengan `cp1252`.
- File ini harus berada di lokasi Amazon S3 untuk diberikan ke AWS Glue.
- Gunakan baris dalam jumlah sedang untuk setiap tugas pelabelan. Disarankan 10–20 baris per tugas, meskipun 2–30 baris per tugas dapat diterima. Tugas yang lebih besar dari 50 baris tidak dianjurkan dan dapat menyebabkan hasil yang buruk atau kegagalan pada sistem.
- Jika Anda memiliki data yang sudah diberi label yang terdiri dari pasangan catatan berlabel "cocok" atau "tidak cocok", itu tidak apa-apa. Pasangan yang sudah diberi label ini dapat direpresentasikan sebagai set pelabelan ukuran 2. Dalam kasus ini, beri label kedua catatan dengan, misalnya, huruf "A" jika mereka cocok, tetapi beri label catatan satu sebagai "A" dan lainnya sebagai "B" jika mereka tidak cocok.

  > **Note**
  >
  > Karena memiliki kolom tambahan, file pelabelan memiliki skema yang berbeda dari file yang berisi data sumber Anda. Tempatkan file pelabelan di folder yang berbeda dari file CSV input transformasi sehingga crawler AWS Glue tidak menganggapnya ketika membuat tabel dalam Katalog Data. Jika tidak, tabel yang dibuat oleh crawler AWS Glue mungkin tidak mewakili data Anda dengan benar.

- Dua kolom pertama (`labeling_set_id`, `label`) diperlukan oleh AWS Glue. Kolom yang tersisa harus cocok dengan skema data yang akan diproses.
- Untuk setiap `labeling_set_id`, Anda mengidentifikasi semua catatan yang cocok dengan menggunakan label yang sama. Sebuah label adalah string unik yang ditempatkan di kolom `label`. Sebaiknya gunakan label yang berisi karakter sederhana, seperti A, B, C, dan sebagainya. Label tersebut peka huruf besar kecil dan dimasukkan dalam kolom `label`.
- Baris yang berisi `labeling_set_id` yang sama dan label yang sama dipahami untuk diberi label sebagai kecocokan.

- Baris yang berisi `labeling_set_id` yang sama dan label yang berbeda dipahami untuk diberi label sebagai bukan kecocokan
- Baris yang berisi `labeling_set_id` berbeda dipahami untuk tidak menyampaikan informasi yang mendukung atau bertentangan dengan kecocokan.

  Berikut ini adalah contoh pelabelan data:

labeling_set_id	label	first_name	last_name	Birthday
ABC123	A	John	Doe	04/01/1980
ABC123	B	Jane	Smith	04/03/1980
ABC123	A	Johnny	Doe	04/01/1980
ABC123	A	Jon	Doe	04/01/1980
DEF345	A	Richard	Jones	12/11/1992
DEF345	A	Rich	Jones	11/12/1992
DEF345	B	Sarah	Jones	12/11/1992
DEF345	C	Richie	Jones Jr.	05/06/2017
DEF345	B	Sarah	Jones-Walker	12/11/1992
GHI678	A	Robert	Miller	1/3/1999
GHI678	A	Bob	Miller	1/3/1999
XYZABC	A	William	Robinson	2/5/2001
XYZABC	B	Andrew	Robinson	2/5/1971

- Dalam contoh di atas kita mengidentifikasi John/Johnny/Jon Doe sebagai sebuah kecocokan dan kami mengajarkan sistem bahwa catatan-catatan ini tidak cocok Jane Smith. Secara terpisah, kami mengajarkan sistem bahwa Richard dan Rich Jones adalah orang yang sama, tetapi catatan-catatan tersebut tidak cocok dengan Sarah Jones/Jones-Walker dan Richie Jones Jr.

- Seperti yang Anda lihat, ruang lingkup label terbatas pada `labeling_set_id`. Jadi label tidak menyeberangi batas `labeling_set_id`. Sebagai contoh, label "A" dalam `labeling_set_id` 1 tidak memiliki hubungan dengan label "A" di `labeling_set_id` 2.
- Jika sebuah catatan tidak memiliki kecocokan dalam satu set pelabelan, maka tetapkan label unik padanya. Sebagai contoh, Jane Smith tidak cocok dengan catatan apa pun dalam set pelabelan ABC123, sehingga ia menjadi satu-satunya catatan dalam set pelabelan dengan label B.
- Set pelabelan "GHI678" menunjukkan bahwa satu set pelabelan hanya dapat terdiri dari dua catatan yang diberi label yang sama untuk menunjukkan bahwa set label tersebut cocok. Demikian pula, "XYZABC" menunjukkan dua catatan yang diberikan label yang berbeda untuk menunjukkan bahwa catatan-catatan tersebut tidak cocok.
- Perhatikan bahwa kadang-kadang set pelabelan mungkin berisi ketidakcocokan (yaitu, Anda memberikan setiap catatan dalam set pelabelan dengan label yang berbeda) atau set pelabelan mungkin semua "sama" (Anda memberi mereka semua dengan label yang sama). Hal ini tidak apa-apa selama set pelabelan Anda secara kolektif berisi contoh catatan yang "sama" dan tidak "sama" dengan kriteria Anda.

> **Important**
>
> Mengonfirmasi bahwa IAM role yang Anda berikan ke AWS Glue memiliki akses ke bucket Amazon S3 yang berisi file pelabelan. Dengan konvensi, kebijakan AWS Glue memberikan izin ke bucket Amazon S3 atau folder yang namanya memiliki prefiks aws-glue-. Jika file pelabelan Anda berada di lokasi yang berbeda, tambahkan izin ke lokasi tersebut dalam IAM role.

Pembelajaran mesin tuning berubah di AWS Glue

Anda dapat menyetel transformasi machine learning Anda di AWS Glue untuk meningkatkan hasil tugas pembersihan data Anda untuk memenuhi tujuan Anda. Untuk meningkatkan transformasi Anda, Anda dapat mengajarinya dengan menghasilkan set pelabelan, menambahkan label, dan kemudian mengulangi langkah-langkah ini beberapa kali sampai Anda mendapatkan hasil yang diinginkan. Anda juga dapat menyetel dengan mengubah beberapa parameter machine learning.

Untuk informasi selengkapnya tentang transformasi machine learning, lihat Rekam pencocokan dengan AWS Lake Formation FindMatches.

Topik

- Pengukuran pembelajaran mesin
- Memutuskan antara presisi dan mengingat
- Memutuskan Antara Akurasi dan Biaya
- Memperkirakan kualitas pertandingan menggunakan skor kepercayaan pertandingan
- Mengajarkan transformasi Find Matches

Pengukuran pembelajaran mesin

Untuk memahami pengukuran yang digunakan untuk menyetel transformasi machine learning Anda, Anda harus terbiasa dengan terminologi berikut:

Benar positif (TP)

: Sebuah kecocokan dalam data yang ditemukan dengan benar oleh transformasi, kadang-kadang disebut temuan.

Benar negatif (TN)

: Sebuah ketidakcocokan dalam data yang dengan benar ditolak oleh transformasi.

Positif palsu (FP)

: Sebuah ketidakcocokan dalam data yang keliru diklasifikasikan sebagai kecocokan, kadang-kadang disebut alarm palsu.

Negatif palsu (FN)

: Sebuah kecocokan dalam data yang tidak ditemukan oleh transformasi, kadang-kadang disebut meleset.

Untuk informasi lebih lanjut tentang terminologi yang digunakan dalam machine learning, lihat matriks kebingungan di Wikipedia.

Untuk menyetel transformasi machine learning Anda, Anda dapat mengubah nilai pengukuran berikut di Properti lanjutan dari transformasi tersebut.

- Precision mengukur seberapa baik transformasi menemukan benar positif di antara jumlah total catatan yang diidentifikasi sebagai positif (benar positif dan positif palsu). Untuk informasi selengkapnya, lihat Precision dan recall di Wikipedia.
- Recall mengukur seberapa baik transformasi menemukan benar positif dari total catatan dalam data sumber. Untuk informasi selengkapnya, lihat Precision dan recall di Wikipedia.

- Akurasi mengukur seberapa baik transformasi menemukan benar positif dan benar negatif. Peningkatan akurasi memerlukan lebih banyak sumber daya dan biaya mesin. Tetapi, itu juga menghasilkan peningkatan recall. Untuk informasi selengkapnya, lihat Accuracy dan precision di Wikipedia.
- Biaya mengukur berapa banyak sumber daya komputasi (dan uang) yang digunakan untuk menjalankan transformasi.

Memutuskan antara presisi dan mengingat

Setiap transformasi `FindMatches` berisi parameter `precision-recall`. Anda menggunakan parameter tersebut untuk menentukan salah satu hal berikut ini:

- Jika Anda lebih peduli tentang transformasi yang keliru melaporkan bahwa dua catatan cocok ketika mereka sebenarnya tidak cocok, maka Anda harus menekankan pada precision.
- Jika Anda lebih peduli tentang transformasi yang gagal untuk mendeteksi catatan yang sebenarnya cocok, maka Anda harus menekankan pada recall.

Anda dapat melakukan trade-off ini di konsol AWS Glue atau menggunakan operasi API machine learning AWS Glue.

Kapan harus mendukung presisi

Dukung precision jika Anda lebih peduli tentang risiko jika `FindMatches` menghasilkan sepasang catatan yang cocok ketika mereka sebenarnya tidak cocok. Untuk mendukung precision, pilih nilai trade-off dari precision-recall yang lebih tinggi. Dengan nilai yang lebih tinggi, transformasi `FindMatches` akan membutuhkan lebih banyak bukti untuk memutuskan bahwa sepasang catatan harus dicocokkan. Transformasi disetel untuk cenderung ke arah yang mengatakan bahwa catatan tidak cocok.

Misalnya, bayangkan bahwa Anda menggunakan `FindMatches` untuk mendeteksi item duplikat dalam katalog video, dan Anda memberikan nilai precision-recall yang lebih tinggi untuk transformasi tersebut. Jika transformasi Anda salah mendeteksi bahwa Star Wars: A New Hope adalah sama dengan Star Wars: The Empire Strikes Back, maka seorang pelanggan yang ingin A New Hope mungkin ditampilkan The Empire Strikes Back. Ini akan menjadi pengalaman pelanggan yang buruk.

Namun, jika transformasi tersebut gagal untuk mendeteksi bahwa Star Wars: A New Hope dan Star Wars: Episode IV—A New Hope adalah item yang sama, maka pelanggan mungkin bingung pada

awalnya tapi mungkin akhirnya mengenali mereka sebagai video yang sama. Hal itu akan menjadi kesalahan, tapi tidak seburuk skenario sebelumnya.

Kapan harus mengingat kembali

Dukung recall jika Anda lebih peduli tentang risiko yang diakibatkan transformasi `FindMatches` jika ia gagal untuk mendeteksi sepasang catatan yang sebenarnya cocok. Untuk mendukung recall, pilih nilai trade-off dari precision-recall menjadi lebih rendah. Dengan nilai yang lebih rendah, transformasi `FindMatches` akan membutuhkan lebih sedikit bukti untuk memutuskan bahwa sepasang catatan harus dicocokkan. Transformasi disetel untuk cenderung ke arah yang mengatakan bahwa catatan cocok.

Sebagai contoh, hal ini mungkin menjadi prioritas untuk organisasi keamanan. Misalkan Anda mencocokkan pelanggan terhadap daftar penipu yang sudah dikenal, dan penting untuk menentukan apakah seorang pelanggan adalah seorang penipu. Anda menggunakan `FindMatches` untuk mencocokkan daftar penipu dengan daftar pelanggan. Setiap kali `FindMatches` mendeteksi kecocokan antara dua daftar tersebut, maka auditor manusia ditugaskan untuk memverifikasi bahwa orang tersebut, pada kenyataannya, adalah seorang penipu. Organisasi Anda mungkin lebih memilih untuk memilih recall dari pada precision. Dengan kata lain, Anda lebih suka membuat auditor meninjau dan menolak secara manual beberapa kasus ketika pelanggan tersebut bukan seorang penipu daripada gagal untuk mengidentifikasi bahwa seorang pelanggan, pada kenyataannya, ada dalam daftar penipu.

Bagaimana mendukung presisi dan ingatan

Cara terbaik untuk meningkatkan precision dan recall adalah dengan memberi labeli pada lebih banyak data. Saat Anda melabeli lebih banyak data, maka keakuratan transformasi `FindMatches` secara keseluruhan meningkat, sehingga hal itu juga meningkatkan precision dan recall. Namun demikian, bahkan dengan transformasi yang paling akurat sekalipun, selalu ada area abu-abu di mana Anda perlu bereksperimen dengan mendukung precision atau recall, atau memilih nilai di tengah.

Memutuskan Antara Akurasi dan Biaya

Setiap transformasi `FindMatches` berisi sebuah parameter `accuracy-cost`. Anda dapat menggunakan parameter tersebut untuk menentukan salah satu hal berikut ini:

- Jika Anda lebih peduli dengan transformasi yang secara akurat melaporkan bahwa dua catatan cocok, maka Anda harus menekankan pada accuracy.

- Jika Anda lebih peduli dengan biaya atau kecepatan menjalankan transformasi, maka Anda harus menekankan pada biaya lebih rendah.

Anda dapat melakukan trade-off ini di konsol AWS Glue atau menggunakan operasi API machine learning AWS Glue.

Kapan harus mendukung akurasi

Dukung accuracy jika Anda lebih peduli tentang risiko yang diakibatkan `find matches` jika hasilnya tidak akan berisi kecocokan. Untuk mendukung accuracy, pilih nilai trade-off accuracy-cost yang lebih tinggi. Dengan nilai yang lebih tinggi, transformasi `FindMatches` memerlukan lebih banyak waktu untuk melakukan pencarian yang lebih menyeluruh untuk melakukan pencocokan catatan dengan benar. Perhatikan bahwa parameter ini tidak mengurangi kemungkinan keliru memanggil pasangan catatan yang tidak cocok sebagai pasangan yang cocok. Transformasi disetel untuk bias ke arah menghabiskan lebih banyak waktu untuk menemukan kecocokan.

Kapan harus mendukung biaya

Dukung cost jika Anda lebih peduli tentang biaya menjalankan transformasi `find matches` dan kurang peduli tentang berapa banyak kecocokan yang ditemukan. Untuk mendukung cost, pilih nilai trade-off dari accuracy-cost yang lebih rendah. Dengan nilai yang lebih rendah, transformasi `FindMatches` memerlukan lebih sedikit sumber daya untuk dijalankan. Transformasi disetel untuk bias ke arah menemukan kecocokan lebih sedikit. Jika hasilnya dapat diterima ketika mendukung biaya yang lebih rendah, gunakan pengaturan ini.

Bagaimana mendukung akurasi dan biaya yang lebih rendah

Dibutuhkan lebih banyak waktu mesin untuk memeriksa lebih banyak pasangan catatan untuk menentukan apakah mereka mungkin pasangan yang cocok. Jika Anda ingin mengurangi biaya tanpa mengurangi kualitas, berikut adalah beberapa langkah yang dapat Anda lakukan:

- Hilangkan catatan di sumber data yang tidak Anda pedulikan dalam pencocokan.
- Hilangkan kolom dari sumber data yang Anda yakin tidak berguna untuk membuat keputusan kecocokan/ketidakcocokan. Cara yang baik untuk memutuskan ini adalah dengan menghilangkan kolom yang menurut Anda tidak mempengaruhi keputusan Anda sendiri tentang apakah satu set catatan adalah “sama.”

Memperkirakan kualitas pertandingan menggunakan skor kepercayaan pertandingan

Skor kepercayaan kecocokan memberikan perkiraan kualitas kecocokan yang ditemukan oleh FindMatches untuk membedakan antara catatan yang cocok di mana model pembelajaran mesin sangat percaya diri, tidak pasti, atau tidak mungkin. Skor kepercayaan pertandingan akan berada di antara 0 dan 1, di mana skor yang lebih tinggi berarti kesamaan yang lebih tinggi. Memeriksa skor kepercayaan kecocokan memungkinkan Anda membedakan antara kelompok kecocokan di mana sistem sangat percaya diri (yang mungkin Anda putuskan untuk digabungkan), kelompok yang sistemnya tidak pasti (yang mungkin Anda putuskan untuk ditinjau oleh manusia), dan cluster yang dianggap tidak mungkin (yang mungkin Anda putuskan untuk ditolak).

Anda mungkin ingin menyesuaikan data latihan Anda dalam situasi di mana Anda melihat skor kepercayaan pertandingan yang tinggi, tetapi tentukan tidak ada pertandingan, atau di mana Anda melihat skor rendah tetapi tentukan ada, pada kenyataannya, pertandingan.

Skor kepercayaan sangat berguna ketika ada kumpulan data industri berukuran besar, di mana tidak mungkin untuk meninjau setiap keputusan. FindMatches

Skor kepercayaan pertandingan tersedia dalam AWS Glue versi 2.0 atau yang lebih baru.

Menghasilkan skor kepercayaan pertandingan

Anda dapat menghasilkan skor kepercayaan kecocokan dengan menyetel nilai Boolean `computeMatchConfidenceScores` ke True saat memanggil `FindIncrementalMatches` API `FindMatches` atau.

AWS Gluemenambahkan yang baru `column match_confidence_score` ke output.

Contoh penilaian pertandingan

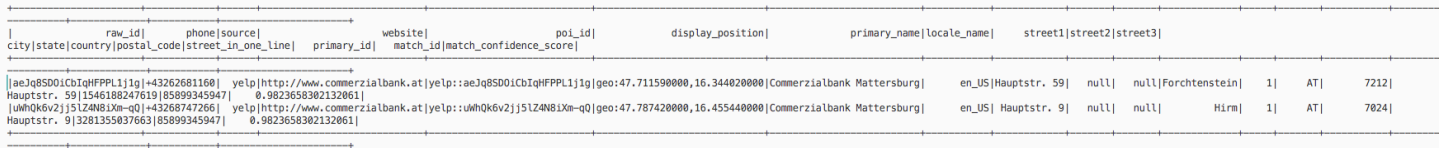
Misalnya, pertimbangkan catatan yang cocok berikut ini:

Skor >= 0,9

Ringkasan catatan yang cocok:

```
  primary_id  |   match_id  | match_confidence_score

3281355037663    85899345947   0.9823658302132061
1546188247619    85899345947   0.9823658302132061
```

Rincian:



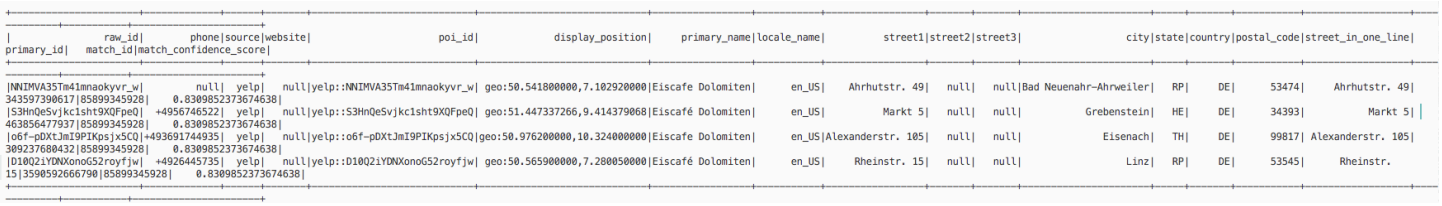
Dari contoh ini, kita dapat melihat bahwa dua catatan sangat mirip dan berbagi`display_position`,`primary_name`, dan`street name`.

Skor >= 0.8 dan skor <0.9

Ringkasan catatan yang cocok:

```
  primary_id  |   match_id  | match_confidence_score

309237680432     85899345928   0.8309852373674638
3590592666790    85899345928   0.8309852373674638
343597390617     85899345928   0.8309852373674638
249108124906     85899345928   0.8309852373674638
463856477937     85899345928   0.8309852373674638
```

Rincian:



Dari contoh ini, kita dapat melihat bahwa catatan ini berbagi hal yang sama`primary_name`, dan`country`.

Skor >= 0.6 dan skor <0.7

Ringkasan catatan yang cocok:

```
  primary_id  |   match_id  | match_confidence_score

2164663519676    85899345930   0.6971099896480333
 317827595278    85899345930   0.6971099896480333
 472446424341    85899345930   0.6971099896480333
3118146262932    85899345930   0.6971099896480333
```

```
214748380804    85899345930    0.6971099896480333
```

Rincian:

```
+--------------------+------------+------+-------+--------------------------+--------------------------+------------+-----------+------------------------------+-------+-------+---------+-----+-------+-----------+------------------------------+
-------------+--------------+----------------------+
|              raw_id|       phone|source|website|                    poi_id|          display_position|primary_name|locale_name|                       street1|street2|street3|     city|state|country|postal_code|             street_in_one_line|
primary_id|   match_id|match_confidence_score|
+--------------------+------------+------+-------+--------------------------+--------------------------+------------+-----------+------------------------------+-------+-------+---------+-----+-------+-----------+------------------------------+
-------------+--------------+----------------------+
|IQT_R8tk4ngTFXHhpyB8ww|+33490963451|  yelp|   null|yelp::IQT_R8tk4ngTFXHhpyB8ww| geo:43.675559000,4.626792000|   Le Vésuve|      en_US|       15 Rue de la Rotonde|   null|   null|    Arles|   13|     FR|      13200|       15 Rue de la Rotonde|
317827595278|85899345930|    0.6971099896480333|
|b8cCAxbvEcug27QMmQYMjQ|        null|  yelp|   null|yelp::b8cCAxbvEcug27QMmQYMjQ| geo:50.631700000,3.067380000|   Le Vésuve|      en_US|30 ave du President Kennedy|   null|   null|    Lille|   59|     FR|      59800|30 ave du President
Kennedy|3118146262932|85899345930|    0.6971099896480333|
|djOC4FZnWXS1wEnFB6vj5g|+33442750804|  yelp|   null|yelp::djOC4FZnWXS1wEnFB6vj5g| geo:43.427710000,5.236950000|   le Vesuve|      en_US|         24 ave Bruxelles|   null|   null|Vitrolles|   13|     FR|      13127|          24 ave
Bruxelles|2164663519676|85899345930|    0.6971099896480333|
|uB59q9Ga561Cljt4wypnkg|+33297251001|  yelp|   null|yelp::uB59q9Ga561Cljt4wypnkg|geo:48.071493200,-2.963742000|   Le Vésuve|      en_US|       49 Rue Gén de Gaulle|   null|   null|  Pontivy|   56|     FR|      56300|       49 Rue Gén de Gaulle|
472446424341|85899345930|    0.6971099896480333|
|3wHOMEhra3DUOUgF_YcoTA|+33164069200|  yelp|   null|yelp::3wHOMEhra3DUOUgF_YcoTA| geo:48.610984000,2.888184000|   Le Vesuve|      en_US|59 Avenue Charles de Gaulle|   null|   null|  Mormant|   77|     FR|      77720|59 Avenue Charles de Gaulle|
214748380804|85899345930|    0.6971099896480333|
+--------------------+------------+------+-------+--------------------------+--------------------------+------------+-----------+------------------------------+-------+-------+---------+-----+-------+-----------+------------------------------+
-------------+--------------+----------------------+
```

Dari contoh ini, kita dapat melihat bahwa catatan ini hanya berbagi hal yang sama`primary_name`.

Untuk informasi selengkapnya, lihat:

- Langkah 5: Tambahkan dan jalankan pekerjaan dengan transformasi pembelajaran mesin Anda
- PySpark: FindMatches kelas
- PySpark: FindIncrementalMatches kelas
- Scala: FindMatches kelas
- Scala: FindIncrementalMatches kelas

Mengajarkan transformasi Find Matches

Setiap transformasi `FindMatches` harus diajari apa yang harus dianggap kecocokan dan apa yang seharusnya tidak dianggap sebagai kecocokan. Anda mengajarkan transformasi Anda dengan menambahkan label ke sebuah file dan mengunggah pilihan Anda ke AWS Glue.

Anda dapat mengatur pelabelan ini pada konsol AWS Glue atau menggunakan operasi API machine learning AWS Glue.

Berapa kali saya harus menambahkan label? Berapa banyak label yang saya butuhkan?

Jawaban atas pertanyaan-pertanyaan ini sebagian besar terserah Anda. Anda harus mengevaluasi apakah `FindMatches` memberikan tingkat akurasi yang Anda butuhkan dan apakah menurut Anda upaya melakukan pelabelan ekstra sepadan untuk Anda. Cara terbaik untuk memutuskan hal ini adalah dengan melihat metrik “Precision,” “Recall,” dan “Area di bawah kurva precision-recall” yang dapat Anda hasilkan saat Anda memilih Estimasi kualitas pada konsol AWS Glue. Setelah Anda memberikan label pada kumpulan tugas lainnya, jalankan kembali metrik ini dan verifikasi apakah telah ada peningkatan. Jika, setelah memberikan label pada beberapa kumpulan tugas tersebut,

Anda tidak melihat peningkatan pada metrik yang Anda fokuskan, maka kualitas transformasi mungkin telah mencapai kualitas tertingginya.

Mengapa label positif dan negatif sejati dibutuhkan?

Transformasi `FindMatches` membutuhkan contoh baik positif dan negatif untuk mempelajari apa yang menurut Anda adalah sebuah kecocokan. Jika Anda melabeli data pelatihan yang dihasilkan oleh `FindMatches` (sebagai contoh, menggunakan opsi Saya tidak memiliki label), maka `FindMatches` mencoba untuk menghasilkan satu set “label set id” untuk Anda. Dalam setiap tugas, Anda memberikan “label” yang sama untuk beberapa catatan dan “label” yang berbeda untuk catatan lain. Dengan kata lain, tugas secara umum adalah tidak semua sama atau semua berbeda (tapi tidak apa-apa jika tugas tertentu adalah semua “sama” atau semua “tidak sama”).

Jika Anda mengajari transformasi `FindMatches` Anda menggunakan opsi Unggah label dari S3, coba untuk menyertakan kedua contoh kecocokan dan ketidakcocokan. Hanya memiliki satu jenis saja bisa diterima. Label ini membantu Anda membangun transformasi `FindMatches` yang lebih akurat, tetapi Anda masih perlu memberikan label pada beberapa catatan yang Anda hasilkan dengan menggunakan opsi Buat file pelabelan.

Bagaimana saya bisa menegakkan bahwa transformasi cocok persis seperti yang saya ajarkan?

Transformasi `FindMatches` belajar dari label yang Anda berikan, sehingga dapat menghasilkan pasangan catatan yang tidak mematuhi label yang disediakan. Untuk menegakkan bahwa `FindMatches` transformasi menghormati label Anda, pilih EnforceProvidedLabelsdi. FindMatchesParameter

Teknik apa yang dapat Anda gunakan ketika transformasi MLmengidentifikasi item sebagai kecocokan yang bukan kecocokan sejati?

Anda dapat menggunakan teknik berikut:

- Meningkatkan `precisionRecallTradeoff` ke nilai yang lebih tinggi. Hal ini pada akhirnya akan menghasilkan lebih sedikit kecocokan, tetapi juga harus memecah klaster besar Anda ketika mencapai nilai yang cukup tinggi.
- Ambil baris output yang sesuai dengan hasil yang salah dan lakukan format ulang padanya sebagai set pelabelan (menghapus kolom `match_id` dan menambahkan kolom `labeling_set_id` dan `label`). Jika perlu, pecah (bagi) menjadi beberapa set pelabelan untuk memastikan bahwa pemberi label dapat mengingat setiap label saat menetapkan label. Kemudian, beri label dengan benar pada set yang cocok dan unggah file label dan tambahkan ke label yang

ada. Hal ini mungkin akan cukup mengajarkan transformasi Anda tentang apa yang dicari untuk memahami polanya.

- (Lanjutan) Akhirnya, lihat data tersebut untuk melihat apakah ada pola yang dapat Anda deteksi yang tidak diperhatikan oleh sistem tersebut. Lakukan pra-proses pada data tersebut dengan menggunakan fungsi AWS Glue standar untuk menormalkan data. Sorot apa yang ingin Anda pelajari dari algoritme dengan memisahkan data yang Anda ketahui sebagai hal penting yang berbeda ke dalam kolom tersendiri. Atau bangun kolom gabungan dari kolom yang data-nya Anda ketahui sebagai data terkait.

Bekerja dengan transformasi pembelajaran mesin di konsol AWS Glue

Anda dapat menggunakan AWS Glue untuk membuat transformasi pembelajaran mesin kustom yang dapat digunakan untuk membersihkan data Anda. Anda dapat menggunakan transformasi ini saat membuat tugas di konsol AWS Glue .

Untuk informasi lebih lanjut tentang cara membuat transformasi machine learning, lihat Rekam pencocokan dengan AWS Lake Formation FindMatches.

Topik

Mengubah properti

Untuk melihat transformasi pembelajaran mesin yang ada, masuk ke AWS Management Console, dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/. Di panel navigasi di bawah Integrasi Data dan ETL, pilih Alat klasifikasi data > Pencocokan Rekam.

Properti untuk setiap transformasi:

Nama transformasi

Nama unik yang Anda berikan pada transformasi saat Anda membuatnya.

ID

Sebuah pengenal unik untuk transformasi.

Jumlah label

Jumlah label dalam file pelabelan yang disediakan untuk membantu mengajarkan transformasi.

Status

Menunjukkan apakah transformasi dalam status Siap atau Perlu pelatihan. Untuk menjalankan transformasi machine learning dengan berhasil dalam sebuah tugas, maka ia harus Siap.

Dibuat

Tanggal transformasi dibuat.

Modifikasi

Tanggal transformasi terakhir diperbarui.

Deskripsi

Deskripsi disediakan untuk transformasi, jika ada.

Versi AWS Glue

Versi yang AWS Glue digunakan.

Jalankan ID

Nama unik yang Anda berikan pada transformasi saat Anda membuatnya.

Jenis tugas

Jenis transformasi machine learning; misalnya Menemukan catatan yang cocok.

Status

Menunjukkan status tugas yang dijalankan. Status yang mungkin meliputi:

- Starting
- Berjalan
- Stopping
- Dihentikan
- Berhasil
- Failed
- Waktu habis

Kesalahan

Jika statusnya Gagal, pesan kesalahan ditampilkan menjelaskan alasan kegagalan.

Menambahkan dan mengedit transformasi pembelajaran mesin

Anda dapat melihat, menghapus, mengatur, dan mengajar, atau menyetel transformasi di konsol AWS Glue. Pilih kotak centang di samping transformasi yang ada dalam daftar, pilih Tindakan, kemudian pilih tindakan yang ingin Anda ambil.

Membuat transformasi ML baru

Untuk menambahkan transformasi pembelajaran mesin baru, pilih Buat transformasi. Ikuti petunjuk di Add job wizard. Untuk informasi selengkapnya, lihat Rekam pencocokan dengan AWS Lake Formation FindMatches.

Langkah 1. Tetapkan properti transformasi.

1. Masukkan nama dan deskripsi (opsional).
2. Secara opsional, atur konfigurasi keamanan. Lihat Menggunakan enkripsi data dengan transformasi pembelajaran mesin.
3. Secara opsional, atur pengaturan eksekusi tugas. Pengaturan eksekusi tugas memungkinkan Anda untuk menyesuaikan bagaimana tugas dijalankan. Pilih jenis Pekerja, jumlah pekerja, batas waktu tugas (dalam menit), jumlah percobaan ulang, dan versi. AWS Glue
4. Secara opsional, atur Tag. Tag adalah label yang dapat Anda tetapkan ke AWS sumber daya. Setiap tanda terdiri dari kunci dan nilai opsional. Tag dapat digunakan untuk mencari dan memfilter sumber daya Anda atau melacak AWS biaya Anda.

Langkah 2. Pilih tabel dan kunci utama.

1. Pilih database dan tabel AWS Glue Katalog.
2. Pilih kunci utama dari tabel yang dipilih. Kolom kunci primer biasanya berisi pengenal unik untuk setiap catatan dalam sumber data.

Langkah 3. Pilih opsi penyetelan.

1. Untuk Recall vs presisi, pilih nilai tuning untuk menyetel transformasi agar mendukung penarikan atau presisi. Secara default, Balanced dipilih, tetapi Anda dapat memilih untuk memilih untuk mengingat atau mendukung presisi, atau memilih Custom dan memasukkan nilai antara 0,0 dan 1,0 (inklusif).
2. Untuk biaya lebih rendah vs akurasi, pilih nilai tuning untuk mendukung biaya atau akurasi yang lebih rendah, atau pilih Custom dan masukkan nilai antara 0,0 dan 1,0 (inklusif).

3. Untuk penegakan Match, pilih Paksa output agar sesuai dengan label jika Anda ingin mengajarkan transformasi ML dengan memaksa output agar sesuai dengan label yang digunakan.

Langkah 4. Tinjau dan buat.

1. Tinjau opsi untuk langkah 1 - 3.
2. Pilih Edit untuk setiap langkah yang perlu dimodifikasi. Pilih Buat transformasi untuk menyelesaikan wizard buat transformasi.

Menggunakan enkripsi data dengan transformasi pembelajaran mesin

Saat menambahkan transformasi machine learning ke AWS Glue, Anda dapat menentukan konfigurasi keamanan yang dikaitkan dengan sumber data atau target data. Jika bucket Amazon S3 digunakan untuk menyimpan, maka data dienkripsi dengan konfigurasi keamanan, tentukan konfigurasi keamanan yang sama saat membuat transformasi.

Anda juga dapat memilih untuk menggunakan enkripsi sisi server dengan AWS KMS (SSE-KMS) untuk mengenkripsi model dan label untuk mencegah orang yang tidak berwenang memeriksanya. Jika Anda memilih opsi ini, Anda diminta untuk memilih AWS KMS key berdasarkan nama, atau Anda dapat memilih Masukkan kunci ARN. Jika Anda memilih untuk memasukkan ARN untuk kunci KMS, maka kolom kedua muncul di mana Anda dapat memasukkan ARN kunci KMS.

> **Note**
>
> Saat ini, transformasi ML yang menggunakan kunci enkripsi kustom tidak didukung di Wilayah berikut:
>
> - Asia Pasifik (Osaka) - `ap-northeast-3`

Melihat detail transformasi

Melihat properti transformasi

Halaman properti Transform menyertakan atribut transformasi Anda. Ia menunjukkan detail tentang definisi transformasi, termasuk yang berikut:

- Nama transformasi menunjukkan nama transformasi.

- Jenis mencantumkan jenis transformasi.
- Status menampilkan apakah transformasi siap untuk digunakan dalam skrip atau tugas.
- Paksa keluaran untuk mencocokkan label menampilkan apakah transformasi memaksa output untuk mencocokkan label yang disediakan oleh pengguna.
- Versi Spark terkait dengan versi AWS Glue yang Anda pilih di Properti eksekusi tugas saat menambahkan transformasi. AWS Glue 1.0 dan Spark 2.4 direkomendasikan untuk sebagian besar pelanggan. Untuk informasi selengkapnya, lihat Versi AWS Glue.

Riwayat, Perkirakan kualitas, dan Tag tab

Detail transformasi termasuk informasi yang Anda tetapkan saat Anda membuat transformasi. Untuk melihat detail transformasi, pilih transformasi di daftar Transformasi machine learning, dan tinjau informasi pada tab berikut:

- Riwayat
- Estimasi kualitas
- Tanda

Riwayat

Tab Riwayat menampilkan riwayat eksekusi tugas transformasi Anda. Beberapa jenis tugas dijalankan untuk mengajarkan transformasi. Untuk masing-masing tugas, metrik eksekusi meliputi yang berikut ini:

- ID Eksekusi adalah sebuah pengenal yang dibuat oleh AWS Glue untuk setiap eksekusi tugas ini.
- Jenis tugas menunjukkan jenis eksekusi tugas.
- Status menunjukkan keberhasilan setiap tugas yang tercantum dengan eksekusi terbaru di bagian atas.
- Kesalahan menunjukkan detail pesan kesalahan jika eksekusi tidak berhasil.
- Waktu mulai menunjukkan tanggal dan waktu (waktu setempat) bahwa tugas dimulai.
- Waktu akhir menunjukkan tanggal dan waktu (waktu setempat) bahwa tugas berakhir.
- Log tautan ke log yang ditulis ke `stdout` untuk eksekusi tugas ini.

  Tautan Log membawa Anda ke Amazon CloudWatch Logs. Di sana Anda dapat melihat detail tentang tabel yang dibuat di AWS Glue Data Catalog dan kesalahan apa pun yang ditemui. Anda

dapat mengelola periode penyimpanan log Anda di CloudWatch konsol. Retensi log default adalah `Never Expire`. Untuk informasi selengkapnya tentang cara mengubah periode penyimpanan, lihat Mengubah Penyimpanan Data Log di CloudWatch Log di Panduan Pengguna CloudWatch Log Amazon.

- File label menunjukkan tautan ke Amazon S3 untuk file pelabelan yang dihasilkan.

Estimasi kualitas

Tab Estimasi Kualitas menunjukkan metrik yang Anda gunakan untuk mengukur kualitas transformasi. Estimasi dihitung dengan membandingkan prediksi kecocokan transformasi menggunakan subset dari data berlabel Anda terhadap label yang telah Anda berikan. Perkiraan ini adalah perkiraan. Anda dapat menjalankan tugas Estimasi kualitas yang dijalankan dari tab ini.

Tab Estimasi Kualitas menampilkan metrik dari eksekusi Estimasi kualitas terakhir termasuk properti berikut:

- Area di bawah kurva Presisi-Recall adalah nomor tunggal memperkirakan batas atas kualitas keseluruhan transformasi. Ia bersifat independen tidak tergantung pada pilihan yang dibuat untuk parameter precision-recall. Nilai yang lebih tinggi menunjukkan bahwa Anda memiliki precision-recall tradeoff yang lebih menarik.
- Precision memperkirakan seberapa sering transformasi benar ketika memprediksi kecocokan.
- Batas atas recall memperkirakan bahwa untuk kecocokan yang sebenarnya, seberapa sering transformasi memprediksi kecocokan.
- F1 memperkirakan akurasi transformasi antara 0 dan 1, di mana 1 adalah akurasi terbaik. Untuk informasi selengkapnya, lihat Skor F1 di Wikipedia.
- Tabel Nilai penting kolom menunjukkan nama kolom dan nilai pentingnya untuk setiap kolom. Nilai penting kolom membantu Anda memahami bagaimana kolom berkontribusi pada model Anda, dengan mengidentifikasi kolom dalam catatan Anda yang paling sering digunakan untuk melakukan pencocokan. Data ini dapat meminta Anda untuk menambah atau mengubah label Anda untuk meningkatkan atau menurunkan nilai penting kolom.

  Nilai penting kolom memberikan skor numerik untuk setiap kolom, dengan angka desimal tidak lebih besar dari 1,0.

Untuk informasi tentang memahami estimasi kualitas dibandingkan kualitas sebenarnya, lihat Estimasi kualitas versus kualitas end-to-end (benar).

Untuk informasi lebih lanjut tentang cara menyetel transformasi Anda, lihat Pembelajaran mesin tuning berubah di AWS Glue.

Estimasi kualitas versus kualitas end-to-end (benar)

AWS Gluememperkirakan kualitas transformasi Anda dengan menyajikan model internal yang dipelajari mesin dengan sejumlah pasang catatan yang Anda berikan label yang cocok tetapi model tersebut belum pernah terlihat sebelumnya. Estimasi kualitas ini adalah fungsi dari kualitas model yang dipelajari mesin (yang dipengaruhi oleh jumlah catatan yang Anda beri label untuk "mengajarkan" transformasi). Ingatan end-to-end, atau true (yang tidak dihitung secara otomatis oleh`ML transform`) juga dipengaruhi oleh mekanisme `ML transform` penyaringan yang mengusulkan berbagai kemungkinan kecocokan dengan model yang dipelajari mesin.

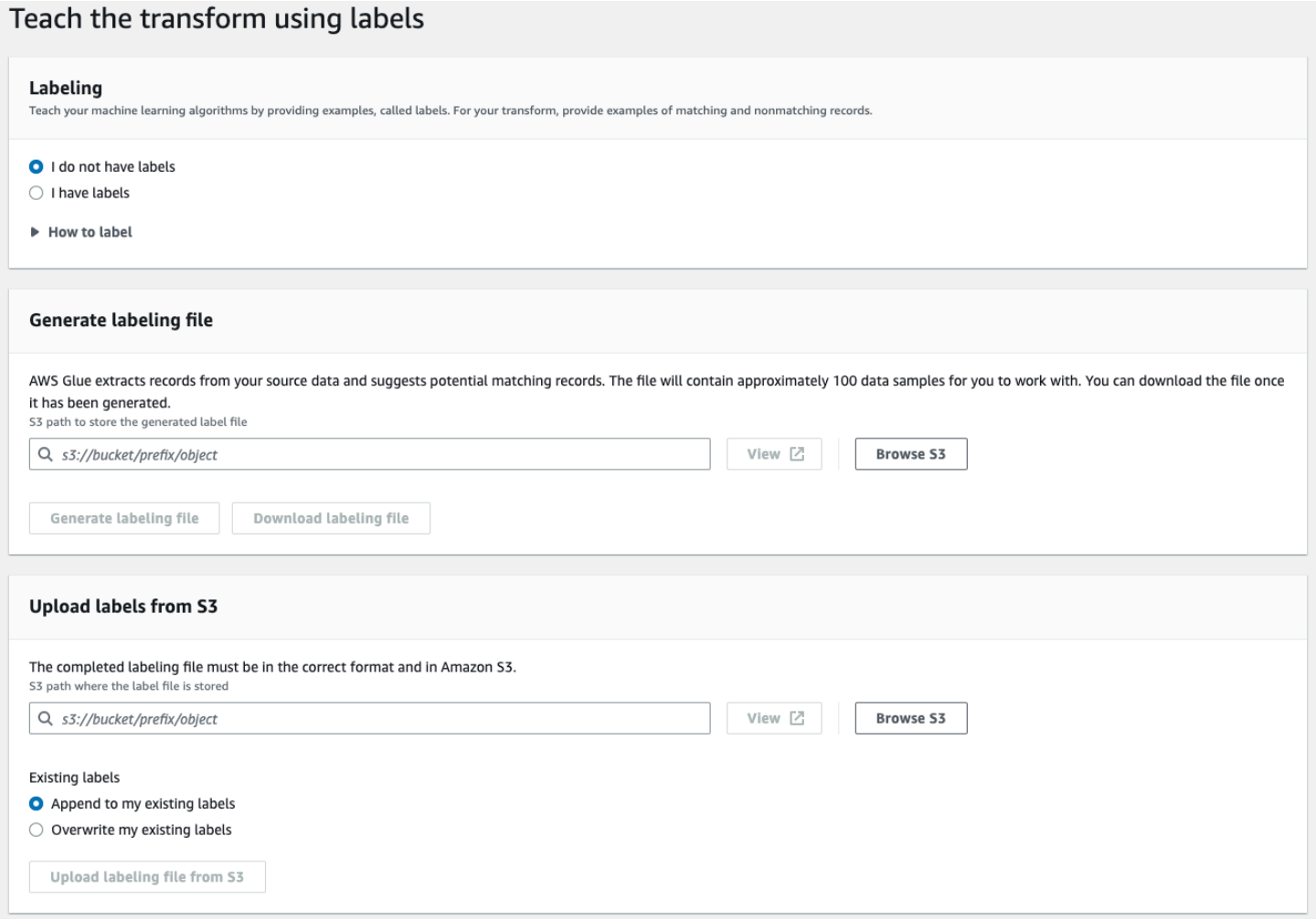
Anda dapat menyetel metode penyaringan ini terutama dengan menentukan nilai tuning Akurasi Biaya Rendah. Karena nilai tuning semakin mendekati Akurasi, sistem melakukan pencarian yang lebih menyeluruh dan mahal untuk pasangan catatan yang mungkin cocok. Lebih banyak pasang catatan diumpankan ke model yang dipelajari mesin Anda, dan ingatan Anda `ML transform` end-to-end atau sebenarnya mendekati metrik penarikan yang diperkirakan. Akibatnya, perubahan end-to-end kualitas pertandingan Anda sebagai akibat dari perubahan tradeoff biaya/akurasi untuk pertandingan Anda biasanya tidak akan tercermin dalam perkiraan kualitas.

Tanda

Tag adalah label yang dapat Anda tetapkan ke AWS sumber daya. Setiap tanda terdiri dari kunci dan nilai opsional. Tag dapat digunakan untuk mencari dan memfilter sumber daya Anda atau melacak AWS biaya Anda.

Ajarkan transformasi menggunakan label

Anda dapat mengajarkan transformasi ML menggunakan label (contoh) dengan memilih Teach transform dari halaman detail transformasi ML. Saat Anda mengajarkan algoritma pembelajaran mesin Anda dengan memberikan contoh (disebut label), Anda dapat memilih label yang ada untuk digunakan, atau membuat file pelabelan.



- Pelabelan — Jika Anda memiliki label, pilih Saya memiliki label. Jika Anda tidak memiliki label, Anda masih dapat melanjutkan dengan langkah berikutnya dalam menghasilkan file pelabelan.
- Hasilkan file pelabelan — AWS Glue ekstrak catatan dari data sumber Anda dan sarankan catatan pencocokan potensial. Anda memilih bucket Amazon S3 untuk menyimpan file label yang dihasilkan. Pilih Hasilkan file pelabelan untuk memulai proses. Setelah selesai, pilih Unduh file pelabelan. File yang diunduh akan memiliki kolom untuk label tempat Anda dapat mengisi label.
- Unggah label dari Amazon S3 — Pilih file pelabelan yang sudah selesai dari bucket Amazon S3 tempat file label disimpan. Kemudian, pilih untuk menambahkan label ke label yang ada atau menimpa label yang ada. Pilih Unggah file pelabelan dari Amazon S3.

Tutorial: Membuat transformasi pembelajaran mesin dengan AWS Glue

Tutorial ini memandu Anda melakukan tindakan untuk membuat dan mengelola transformasi machine learning (ML) menggunakan AWS Glue. Sebelum menggunakan tutorial ini, Anda harus terbiasa

dengan menggunakan konsol AWS Glue untuk menambahkan crawler dan tugas serta mengedit skrip. Anda juga harus terbiasa dengan mencari dan mengunduh file di konsol Amazon Simple Storage Service (Amazon S3).

Dalam contoh ini, Anda membuat sebuah transformasi `FindMatches` untuk menemukan catatan yang cocok, mengajari cara mengidentifikasi catatan yang cocok dan tidak cocok, dan menggunakannya dalam sebuah tugas AWS Glue. Tugas AWS Glue menulis file Amazon S3 baru dengan sebuah kolom tambahan bernama `match_id`.

Sumber data yang digunakan oleh tutorial ini adalah sebuah file bernama `dblp_acm_records.csv`. File ini adalah versi modifikasi dari publikasi akademis (DBLP dan ACM) yang tersedia dari Set data DBLP ACM asli. File `dblp_acm_records.csv` adalah file nilai yang dipisahkan dengan koma (CSV) dalam format UTF-8 tanpa tanda urutan byte (BOM).

File kedua, `dblp_acm_labels.csv`, adalah contoh file pelabelan yang berisi catatan kecocokan dan catatan ketidakcocokan yang digunakan untuk mengajarkan transformasi sebagai bagian dari tutorial ini.

Topik

Langkah 1: Merayapi data sumber

Pertama, lakukan crawling pada file CSV Amazon S3 sumber untuk membuat sebuah tabel metadata yang sesuai di Katalog Data.

> **Important**
>
> Untuk mengarahkan crawler agar membuat tabel untuk file CSV saja, simpan data sumber CSV di folder Amazon S3 yang berbeda dari file lain.

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pilih Crawler, Tambahkan crawler.
3. Ikuti penuntun untuk membuat dan menjalankan crawler bernama `demo-crawl-dblp-acm` dengan keluaran ke basisdata `demo-db-dblp-acm`. Saat menjalankan penuntun, buat basis data `demo-db-dblp-acm` jika itu belum ada. Pilih Amazon S3 termasuk path ke data sampel dalam Wilayah AWS saat ini. Contohnya, untuk `us-east-1`, Amazon S3 mencakup path ke file sumber yakni `s3://ml-transforms-public-datasets-us-east-1/dblp-acm/records/dblp_acm_records.csv`.

   Jika berhasil, crawler menciptakan tabel `dblp_acm_records_csv` dengan kolom berikut: id, judul, penulis, tempat, tahun, dan sumber.

Langkah 2: Tambahkan transformasi pembelajaran mesin

Selanjutnya, tambahkan sebuah transformasi machine learning yang didasarkan pada skema tabel sumber data yang dibuat oleh crawler bernama `demo-crawl-dblp-acm`.

1. Di AWS Glue konsol, di panel navigasi di bawah Integrasi Data dan ETL, pilih Alat klasifikasi data > Pencocokan Rekam, lalu Tambahkan transformasi. Ikuti wizard untuk membuat `Find matches` transformasi dengan properti berikut.
   a. Untuk Nama transformasi, masukkan **`demo-xform-dblp-acm`**. Ini adalah nama transformasi yang digunakan untuk menemukan kecocokan dalam data sumber.
   b. Untuk IAM role, pilih sebuah IAM role yang memiliki izin ke data sumber Amazon S3, file pelabelan, dan operasi API AWS Glue. Untuk informasi selengkapnya, lihat Membuat IAM role untuk AWS Glue dalam Panduan Developer AWS Glue.
   c. Untuk sumber Data, pilih tabel bernama dblp_acm_records_csv dalam database. demo-db-dblp-acm
   d. Untuk Kunci primer, pilih kolom kunci primer untuk tabel, id.
2. Dalam penuntun, pilih Selesai dan kembali ke daftar Transformasi ML.

Langkah 3: Ajarkan transformasi pembelajaran mesin Anda

Selanjutnya, Anda ajari transformasi machine learning Anda dengan menggunakan file pelabelan sampel dalam tutorial ini.

Anda tidak dapat menggunakan transformasi bahasa mesin dalam tugas extract, transform, and load (ETL) sampai statusnya Siap digunakan. Agar transformasi Anda siap, maka Anda harus mengajarinya cara mengidentifikasi catatan kecocokan dan ketidakcocokan dengan memberikan contoh catatan kecocokan dan ketidakcocokan. Untuk mengajarkan transformasi Anda, Anda dapat Buat sebuah file label, menambahkan label, dan kemudian Unggah file label. Dalam tutorial ini, Anda dapat menggunakan contoh file pelabelan bernama `dblp_acm_labels.csv`. Untuk informasi selengkapnya tentang proses pelabelan, lihat Pelabelan.

1. Di AWS Glue konsol, di panel navigasi, pilih Rekam Pencocokan.
2. Pilih transformasi `demo-xform-dblp-acm`, dan kemudian pilih Tindakan, Ajarkan. Ikuti panduan untuk mengajar transformasi `Find matches` Anda.
3. Pada halaman properti transformasi, pilih Saya memiliki label. Pilih path Amazon S3 untuk file pelabelan sampel dalam Wilayah AWS saat ini. Contohnya, untuk `us-east-1`, unggah file pelabelan yang tersedia dari path Amazon S3 `s3://ml-transforms-public-datasets-us-east-1/dblp-acm/labels/dblp_acm_labels.csv` dengan opsi menimpa label yang ada. File pelabelan harus ditempatkan di Amazon S3 dengan Wilayah yang sama dengan konsol AWS Glue.

   Ketika Anda mengunggah file pelabelan, sebuah tugas dimulai di AWS Glue untuk menambah atau menimpa label yang digunakan untuk mengajar transformasi cara memproses sumber data.
4. Di halaman terakhir penuntun, pilih Selesai, dan kembali ke daftar Transformasi ML.

Langkah 4: Perkirakan kualitas transformasi pembelajaran mesin Anda

Selanjutnya, Anda dapat memperkirakan kualitas transformasi machine learning Anda. Kualitasnya tergantung pada berapa banyak pelabelan yang telah Anda lakukan. Untuk informasi selengkapnya tentang cara memperkirakan kualitas, lihat Estimasi kualitas.

1. Di AWS Glue konsol, di panel navigasi di bawah Integrasi Data dan ETL, pilih Alat klasifikasi data > Pencocokan Rekam.
2. Pilih transformasi `demo-xform-dblp-acm`, dan pilih tab Estimasi kualitas. Tab ini menampilkan perkiraan kualitas saat ini, jika tersedia, untuk transformasi tersebut.
3. Pilih Estimasi kualitas untuk memulai sebuah tugas untuk memperkirakan kualitas transformasi. Keakuratan estimasi kualitas tersebut didasarkan pada pelabelan data sumber.

4. Arahkan ke tab Riwayat. Dalam panel ini, eksekusi tugas dicantumkan untuk transformasi, termasuk tugas Estimasi kualitas. Untuk detail lebih lanjut tentang eksekusi, pilih Log. Periksa apakah status eksekusi adalah Berhasil ketika selesai.

Langkah 5: Tambahkan dan jalankan pekerjaan dengan transformasi pembelajaran mesin Anda

Pada langkah ini, Anda menggunakan transformasi machine learning Anda untuk menambah dan menjalankan tugas di AWS Glue. Saat transformasi `demo-xform-dblp-acm` sudah Siap digunakan, Anda dapat menggunakannya dalam tugas ETL.

1. Pada konsol AWS Glue, di panel navigasi, pilih Tugas.
2. Pilih Tambahkan tugas, dan ikuti langkah-langkah yang ada di penuntun untuk membuat tugas ETL Spark dengan skrip yang sudah dihasilkan. Pilih nilai-nilai properti berikut untuk transformasi Anda:

   a. Untuk Nama, pilih contoh pekerjaan dalam tutorial ini, demo-etl-dblp-acm.

   b. Untuk IAM role, pilih IAM role dengan izin ke data sumber Amazon S3, file pelabelan, dan operasi API AWS Glue. Untuk informasi selengkapnya, lihat Membuat IAM role untuk AWS Glue dalam Panduan Developer AWS Glue.

   c. Untuk Bahasa ETL, pilih Scala. Ini adalah bahasa pemrograman dalam skrip ETL.

   d. Untuk nama file Script, pilih demo-etl-dblp-acm. Ini adalah nama file dari skrip Scala (sama dengan nama tugas).

   e. Untuk Sumber data, pilih dblp_acm_records_csv. Sumber data yang Anda pilih harus sesuai dengan skema sumber data transformasi machine learning.

   f. Untuk Jenis transformasi, pilih Temukan catatan yang cocok untuk membuat sebuah tugas dengan menggunakan transformasi machine learning.

   g. Bersihkan Hapus catatan duplikasi. Anda tidak ingin menghapus catatan duplikat karena catatan keluaran yang ditulis memiliki tambahan kolom `match_id` yang ditambahkan.

   h. Untuk Transform, pilih demo-xform-dblp-acm, transformasi pembelajaran mesin yang digunakan oleh pekerjaan.

   i. Untuk Buat tabel di target data Anda, pilih untuk membuat tabel dengan properti berikut:

      - Jenis penyimpanan data — **`Amazon S3`**
      - Format — **`CSV`**
      - Jenis kompresi — **`None`**

- Path target — Path Amazon S3 di mana output dari tugas ditulis (di Wilayah AWS konsol saat ini)

3. Pilih Simpan tugas dan edit skrip untuk menampilkan halaman editor skrip.
4. Edit skrip untuk menambahkan pernyataan untuk menyebabkan output tugas untuk Path target untuk ditulis ke satu file partisi tunggal. Tambahkan pernyataan ini segera setelah pernyataan yang menjalankan transformasi `FindMatches`. Pernyataan tersebut serupa dengan yang berikut ini.

```
val single_partition = findmatches1.repartition(1)
```

   Anda harus mengubah pernyataan `.writeDynamicFrame(findmatches1)` untuk menulis output sebagai `.writeDynamicFrame(single_partion)`.
5. Setelah Anda mengedit skrip, pilih Simpan. Skrip yang sudah dimodifikasi terlihat mirip dengan kode berikut, tapi disesuaikan untuk lingkungan Anda.

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.ml.FindMatches
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    // @params: [JOB_NAME]
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
    // @type: DataSource
    // @args: [database = "demo-db-dblp-acm", table_name = "dblp_acm_records_csv",
 transformation_ctx = "datasource0"]
    // @return: datasource0
    // @inputs: []
    val datasource0 = glueContext.getCatalogSource(database = "demo-db-dblp-acm",
 tableName = "dblp_acm_records_csv", redshiftTmpDir = "", transformationContext =
 "datasource0").getDynamicFrame()
```

```
    // @type: FindMatches
    // @args: [transformId = "tfm-123456789012", emitFusion = false,
 survivorComparisonField = "<primary_id>", transformation_ctx = "findmatches1"]
    // @return: findmatches1
    // @inputs: [frame = datasource0]
    val findmatches1 = FindMatches.apply(frame = datasource0, transformId
 = "tfm-123456789012", transformationContext = "findmatches1",
 computeMatchConfidenceScores = true)


    // Repartition the previous DynamicFrame into a single partition.
    val single_partition = findmatches1.repartition(1)


    // @type: DataSink
    // @args: [connection_type = "s3", connection_options = {"path": "s3://aws-
glue-ml-transforms-data/sal"}, format = "csv", transformation_ctx = "datasink2"]
    // @return: datasink2
    // @inputs: [frame = findmatches1]
    val datasink2 = glueContext.getSinkWithFormat(connectionType =
 "s3", options = JsonOptions("""{"path": "s3://aws-glue-ml-transforms-
data/sal"}"""), transformationContext = "datasink2", format =
 "csv").writeDynamicFrame(single_partition)
    Job.commit()
  }
}
```

6. Pilih Jalankan tugas untuk memulai eksekusi tugas. Periksa status tugas dalam daftar tugas. Setelah tugas selesai, di Transformasi ML, tab Riwayat, ada baris ID eksekusi baru yang ditambahkan dari jenis Tugas ETL.
7. Arahkan ke Tugas, tab Riwayat. Dalam panel ini, eksekusi tugas dicantumkan. Untuk detail lebih lanjut tentang eksekusi, pilih Log. Periksa apakah status eksekusi adalah Berhasil ketika selesai.

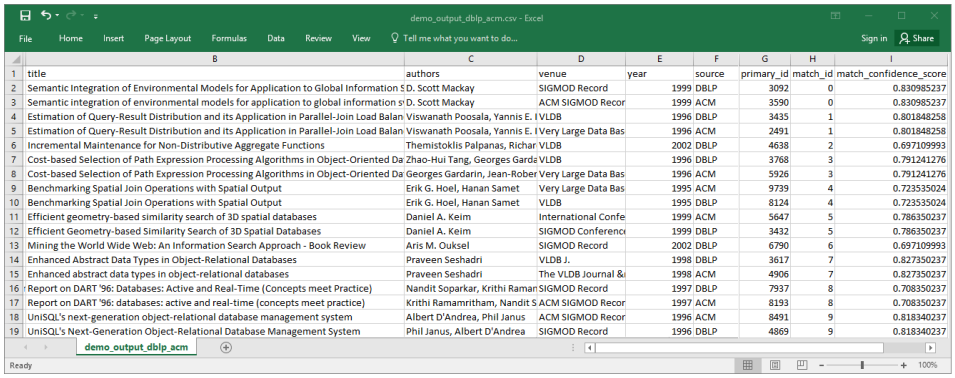
Langkah 6: Verifikasi data keluaran dari Amazon S3

Pada langkah ini, Anda memeriksa output dari eksekusi tugas di bucket Amazon S3 yang Anda pilih ketika Anda menambahkan tugas. Anda dapat mengunduh file output ke komputer lokal Anda dan memverifikasi bahwa catatan kecocokan telah diidentifikasi.

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

2. Unduh file output target dari tugas `demo-etl-dblp-acm`. Buka file dalam aplikasi spreadsheet (Anda mungkin perlu menambahkan ekstensi file `.csv` pada file untuk membukanya dengan benar).

   Gambar berikut menunjukkan kutipan dari output pada Microsoft Excel.



	B	C	D	E	F	G	H	I
1	title	authors	venue	year	source	primary_id	match_id	match_confidence_score
2	Semantic Integration of Environmental Models for Application to Global Information S	D. Scott Mackay	SIGMOD Record	1999	DBLP	3092	0	0.830985237
3	Semantic integration of environmental models for application to global information s	D. Scott Mackay	ACM SIGMOD Recor	1999	ACM	3590	0	0.830985237
4	Estimation of Query-Result Distribution and its Application in Parallel-Join Load Balan	Viswanath Poosala, Yannis E. I	VLDB	1996	DBLP	3435	1	0.801848258
5	Estimation of Query-Result Distribution and its Application in Parallel-Join Load Balan	Viswanath Poosala, Yannis E. I	Very Large Data Bas	1996	ACM	2491	1	0.801848258
6	Incremental Maintenance for Non-Distributive Aggregate Functions	Themistoklis Palpanas, Richar	VLDB	2002	DBLP	4638	2	0.697109993
7	Cost-based Selection of Path Expression Processing Algorithms in Object-Oriented Da	Zhao-Hui Tang, Georges Garda	VLDB	1996	DBLP	3768	3	0.791241276
8	Cost-based Selection of Path Expression Processing Algorithms in Object-Oriented Da	Georges Gardarin, Jean-Rober	Very Large Data Bas	1996	ACM	5926	3	0.791241276
9	Benchmarking Spatial Join Operations with Spatial Output	Erik G. Hoel, Hanan Samet	Very Large Data Bas	1995	ACM	9739	4	0.723535024
10	Benchmarking Spatial Join Operations with Spatial Output	Erik G. Hoel, Hanan Samet	VLDB	1995	DBLP	8124	4	0.723535024
11	Efficient geometry-based similarity search of 3D spatial databases	Daniel A. Keim	International Confe	1999	ACM	5647	5	0.786350237
12	Efficient Geometry-based Similarity Search of 3D Spatial Databases	Daniel A. Keim	SIGMOD Conferenc	1999	DBLP	3432	5	0.786350237
13	Mining the World Wide Web: An Information Search Approach - Book Review	Aris M. Ouksel	SIGMOD Record	2002	DBLP	6790	6	0.697109993
14	Enhanced Abstract Data Types in Object-Relational Databases	Praveen Seshadri	VLDB J.	1998	DBLP	3617	7	0.827350237
15	Enhanced abstract data types in object-relational databases	Praveen Seshadri	The VLDB Journal &	1998	ACM	4906	7	0.827350237
16	Report on DART '96: Databases: Active and Real-Time (Concepts meet Practice)	Nandit Soparkar, Krithi Raman	SIGMOD Record	1997	DBLP	7937	8	0.708350237
17	Report on DART '96: databases: active and real-time (concepts meet practice)	Krithi Ramamritham, Nandit S	ACM SIGMOD Recor	1997	ACM	8193	8	0.708350237
18	UniSQL's next-generation object-relational database management system	Albert D'Andrea, Phil Janus	ACM SIGMOD Recor	1996	ACM	8491	9	0.818340237
19	UniSQL's Next-Generation Object-Relational Database Management System	Phil Janus, Albert D'Andrea	SIGMOD Record	1996	DBLP	4869	9	0.818340237

   Sumber data dan file target keduanya memiliki 4.911 catatan. Namun, transformasi `Find matches` menambahkan kolom lain bernama `match_id` untuk mengidentifikasi catatan kecocokan dalam output. Baris dengan `match_id` yang sama dianggap catatan yang cocok. `match_confidence_score`Ini adalah angka antara 0 dan 1 yang memberikan perkiraan kualitas kecocokan yang ditemukan oleh`Find matches`.

3. Urutkan file output berdasarkan `match_id` untuk dengan mudah melihat catatan mana yang cocok. Bandingkan nilai di kolom lain untuk melihat apakah Anda setuju dengan hasil transformasi `Find matches`. Jika tidak, Anda dapat terus mengajarkan transformasi dengan menambahkan lebih banyak label lainnya.

   Anda juga dapat mengurutkan file berdasarkan kolom lain, seperti `title`, untuk melihat apakah catatan dengan judul serupa memiliki `match_id` yang sama.

Menemukan kecocokan tambahan

Fitur Temukan kecocokan memungkinkan Anda mengidentifikasi rekaman duplikat atau pencocokan dalam kumpulan data Anda, bahkan ketika catatan tidak memiliki pengenal unik yang umum dan tidak ada bidang yang sama persis. Rilis awal Find match mengubah catatan pencocokan yang diidentifikasi dalam satu kumpulan data. Saat menambahkan data baru ke kumpulan data, Anda harus menggabungkannya dengan kumpulan data bersih yang ada dan menjalankan kembali pencocokan dengan kumpulan data gabungan lengkap.

Fitur pencocokan inkremental membuatnya lebih mudah untuk mencocokkan catatan tambahan terhadap kumpulan data yang cocok. Misalkan Anda ingin mencocokkan data prospek dengan kumpulan data pelanggan yang ada. Kemampuan kecocokan tambahan memberi Anda fleksibilitas untuk mencocokkan ratusan ribu prospek baru dengan basis data prospek dan pelanggan yang ada dengan menggabungkan hasilnya ke dalam satu database atau tabel. Dengan hanya mencocokkan antara kumpulan data baru dan yang sudah ada, optimasi kecocokan tambahan find mengurangi waktu komputasi, yang juga mengurangi biaya.

Penggunaan pencocokan inkremental mirip dengan Temukan kecocokan seperti yang dijelaskan dalamTutorial: Membuat transformasi pembelajaran mesin dengan AWS Glue. Topik ini hanya mengidentifikasi perbedaan dengan pencocokan inkremental.

Untuk informasi lebih lanjut, lihat posting blog tentang Pencocokan data tambahan.

Menjalankan pekerjaan pencocokan tambahan

Untuk prosedur berikut, misalkan yang berikut:

- Anda telah meng-crawl dataset yang ada ke dalam tabel first_records. Dataset first_records harus berupa kumpulan data yang cocok, atau output dari pekerjaan yang cocok.
- Anda telah membuat dan melatih transformasi Find match dengan AWS Glue versi 2.0. Ini adalah satu-satunya versi AWS Glue yang mendukung kecocokan tambahan.
- Bahasa ETL adalah Scala. Perhatikan bahwa Python juga didukung.
- Model yang sudah dihasilkan disebut`demo-xform`.

1. Merayapi kumpulan data tambahan ke tabel second_records.
2. Pada konsol AWS Glue, di panel navigasi, pilih Tugas.
3. Pilih Tambahkan tugas, dan ikuti langkah-langkah yang ada di penuntun untuk membuat tugas ETL Spark dengan skrip yang sudah dihasilkan. Pilih nilai-nilai properti berikut untuk transformasi Anda:

   a. Untuk Nama, pilih demo-etl.

   b. Untuk peran IAM, pilih peran IAM dengan izin ke data sumber Amazon S3, file pelabelan, dan operasi API. AWS Glue

   c. Untuk Bahasa ETL, pilih Scala.

   d. Untuk nama file Script, pilih demo-etl. Ini adalah nama file dari skrip Scala.

e. Untuk sumber Data, pilih first_records. Sumber data yang Anda pilih harus sesuai dengan skema sumber data transformasi machine learning.

f. Untuk Jenis transformasi, pilih Temukan catatan yang cocok untuk membuat sebuah tugas dengan menggunakan transformasi machine learning.

g. Pilih opsi pencocokan tambahan, dan untuk Sumber Data pilih tabel bernama second_records.

h. Untuk Transform, pilih demo-xform, transformasi pembelajaran mesin yang digunakan oleh pekerjaan.

i. Pilih Buat tabel di target data Anda atau Gunakan tabel di katalog data dan perbarui target data Anda.

4. Pilih Simpan tugas dan edit skrip untuk menampilkan halaman editor skrip.
5. Pilih Jalankan tugas untuk memulai eksekusi tugas.

Menggunakan FindMatches dalam pekerjaan visual

Untuk menggunakan FindMatchestransform inAWS Glue Studio, Anda dapat menggunakan node Custom Transform yang memanggil FindMatches API. Untuk informasi selengkapnya tentang cara menggunakan transformasi kustom, lihat Membuat transformasi kustom

> **Note**
>
> Saat ini, FindMatches API hanya berfungsi dengan`Glue 2.0`. Untuk menjalankan pekerjaan dengan transformasi Kustom yang memanggil FindMatches API, pastikan AWS Glue versinya ada `Glue 2.0` di tab Detail pekerjaan. Jika versi tidak`Glue 2.0`, pekerjaan akan gagal saat runtime dengan pesan kesalahan berikut: “tidak dapat mengimpor nama '' dari 'FindMatchesawsglueml.transforms'”. AWS Glue

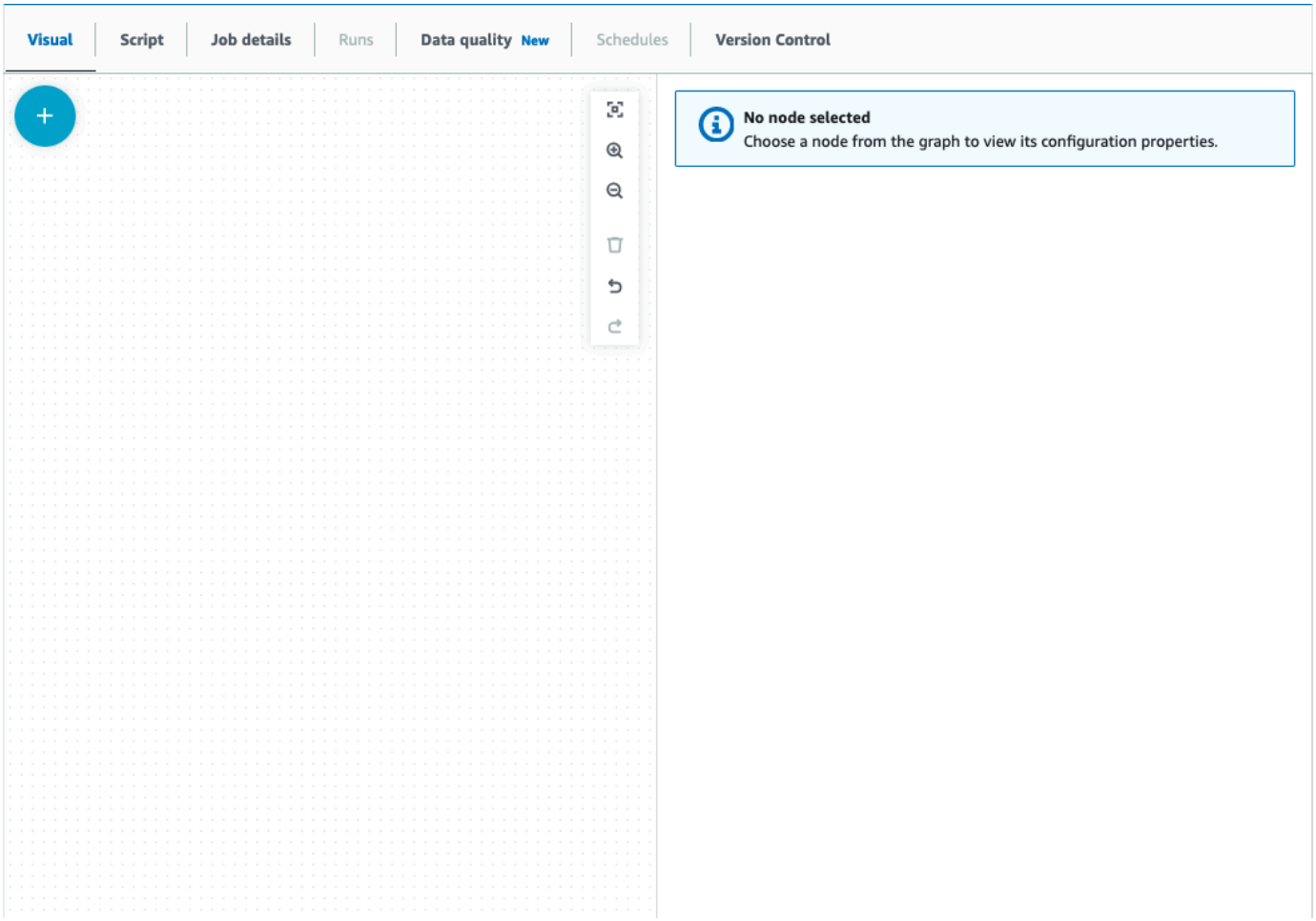
Prasyarat

- Untuk menggunakan transformasi Find Matches, buka AWS Glue Studio konsol dihttps://console.aws.amazon.com/gluestudio/.
- Buat transformasi pembelajaran mesin. Saat dibuat, transformId dihasilkan. Anda akan memerlukan ID ini untuk langkah-langkah di bawah ini. Untuk informasi selengkapnya tentang cara membuat transformasi pembelajaran mesin, lihat Menambahkan dan mengedit transformasi pembelajaran mesin.

Menambahkan FindMatches transformasi

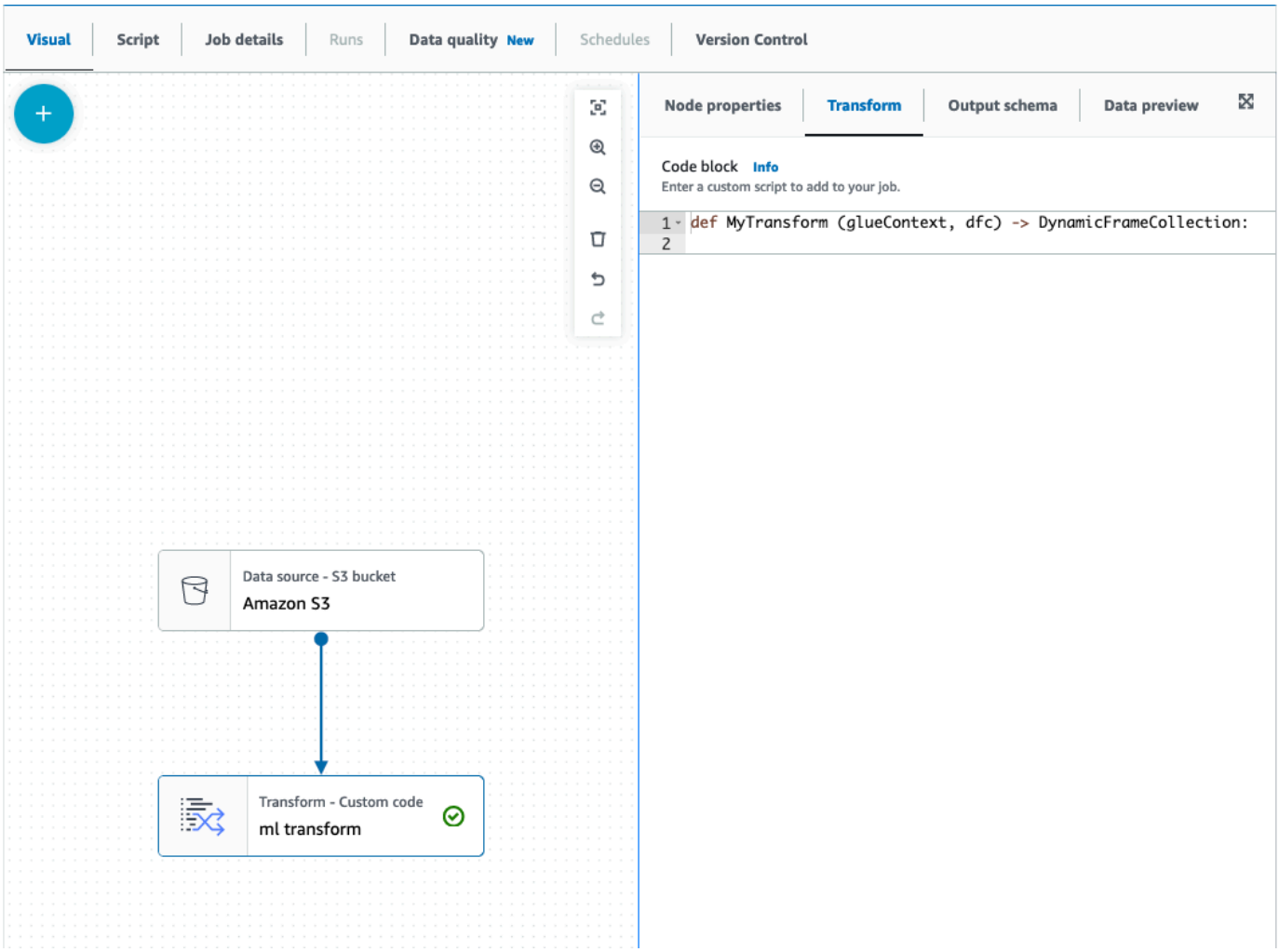
Untuk menambahkan FindMatches transformasi:

1. Di editor AWS Glue Studio pekerjaan, buka panel Sumber daya dengan mengklik simbol silang di sudut kiri atas grafik pekerjaan visual dan pilih Sumber data dengan memilih tab Data. Ini adalah sumber data yang ingin Anda periksa kecocokan.



2. Pilih simpul sumber data, lalu buka panel Resource dengan mengklik simbol silang di sudut kiri atas grafik pekerjaan visual dan cari 'custom transform'. Pilih node Custom Transform untuk menambahkannya ke grafik. Transformasi Kustom ditautkan ke node sumber data. Jika tidak, Anda dapat mengklik node Custom Transform dan memilih tab properti Node, lalu di bawah orang tua Node, pilih sumber data.
3. Klik node Custom Transform di grafik visual, lalu pilih tab Properti Node dan beri nama transformasi kustom. Disarankan agar Anda mengganti nama transformasi sehingga nama transformasi mudah diidentifikasi dalam grafik visual.

4. Pilih tab Transform, di mana Anda dapat mengedit blok kode. Di sinilah kode untuk memanggil FindMatches API dapat ditambahkan.



Blok kode berisi kode yang telah diisi sebelumnya untuk membantu Anda memulai. Timpa kode yang telah diisi sebelumnya dengan templat di bawah ini. Template memiliki placeholder untuk transformId, yang dapat Anda berikan.

```
def MyTransform (glueContext, dfc) -> DynamicFrameCollection:
    dynf = dfc.select(list(dfc.keys())[0])
    from awsglueml.transforms import FindMatches
    findmatches = FindMatches.apply(frame = dynf, transformId = "<your id>")
    return(DynamicFrameCollection({"FindMatches": findmatches}, glueContext))
```

5. Klik node Custom Transform di grafik visual, lalu buka panel Resource dengan mengklik simbol silang di sudut kiri atas grafik pekerjaan visual dan cari 'Select From Collection'. Tidak perlu mengubah pilihan default karena hanya ada satu DynamicFrame di koleksi.
6. Anda dapat terus menambahkan transformasi atau menyimpan hasilnya, yang sekarang diperkaya dengan kolom tambahan find match. Jika Anda ingin mereferensikan kolom-kolom baru tersebut dalam transformasi hilir, Anda perlu menambahkannya ke skema keluaran transformasi. cara termudah untuk melakukannya adalah dengan memilih tab pratinjau data dan kemudian di tab skema pilih “Gunakan skema tinjauan data”.
7. Untuk menyesuaikan FindMatches, Anda dapat menambahkan parameter tambahan untuk diteruskan ke metode 'terapkan'. Lihat FindMatches kelas.

Menambahkan FindMatches transformasi bertahap

Dalam kasus kecocokan inkremental, prosesnya sama dengan Menambahkan FindMatches transformasi dengan perbedaan berikut:

- Alih-alih node induk untuk transformasi kustom, Anda memerlukan dua node induk.
- Node induk pertama harus berupa kumpulan data.
- Node induk kedua harus berupa kumpulan data tambahan.

  Ganti `transformId` dengan Anda `transformId` di blok kode template:

```
def MyTransform (glueContext, dfc) -> DynamicFrameCollection:
    dfs = list(dfc.values())
    dynf = dfs[0]
    inc_dynf = dfs[1]
    from awsglueml.transforms import FindIncrementalMatches
    findmatches = FindIncrementalMatches.apply(existingFrame = dynf, incrementalFrame
 = inc_dynf,
                                    transformId = "<your id>")
    return(DynamicFrameCollection({"FindMatches": findmatches}, glueContext))
```

- Untuk parameter opsional, lihat FindIncrementalMatches kelas.

Migrasi program Apache Spark ke AWS Glue

Apache Spark adalah platform sumber terbuka untuk beban kerja komputasi terdistribusi yang dilakukan pada kumpulan data besar. AWS Gluememanfaatkan kemampuan Spark untuk memberikan pengalaman yang dioptimalkan untuk ETL. Anda dapat memigrasi program Spark AWS Glue untuk memanfaatkan fitur kami. AWS Gluememberikan peningkatan kinerja yang sama seperti yang Anda harapkan dari Apache Spark di Amazon EMR.

Jalankan kode Spark

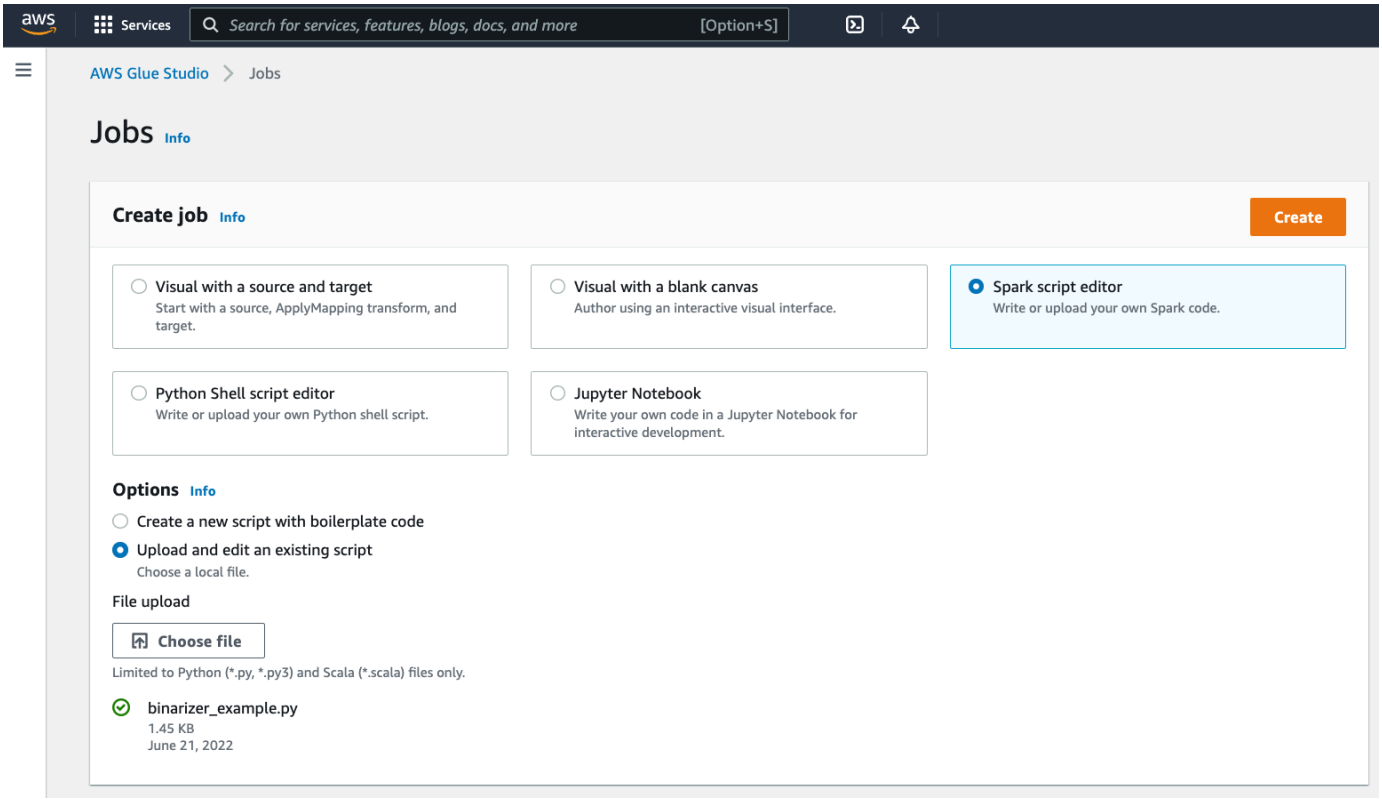
Kode Native Spark dapat dijalankan di AWS Glue lingkungan di luar kotak. Skrip sering dikembangkan dengan mengubah sepotong kode secara iteratif, alur kerja yang cocok untuk Sesi Interaktif. Namun, kode yang ada lebih cocok untuk dijalankan dalam AWS Glue pekerjaan, yang memungkinkan Anda menjadwalkan dan secara konsisten mendapatkan log dan metrik untuk setiap skrip yang dijalankan. Anda dapat mengunggah dan mengedit skrip yang ada melalui konsol.

1. Dapatkan sumber ke skrip Anda. Untuk contoh ini, Anda akan menggunakan contoh skrip dari repositori Apache Spark. Contoh Binarizer
2. Di AWS Glue Console, perluas panel navigasi sisi kiri dan pilih ETL > Jobs

   Di panel Buat pekerjaan, pilih Editor skrip Spark. Bagian Opsi akan muncul. Di bawah Opsi, pilih Unggah dan edit skrip yang ada.

   Bagian unggahan file akan muncul. Di bawah Unggah file, klik Pilih file. Pemilih file sistem Anda akan muncul. Arahkan ke lokasi tempat Anda menyimpan`binarizer_example.py`, pilih dan konfirmasikan pilihan Anda.

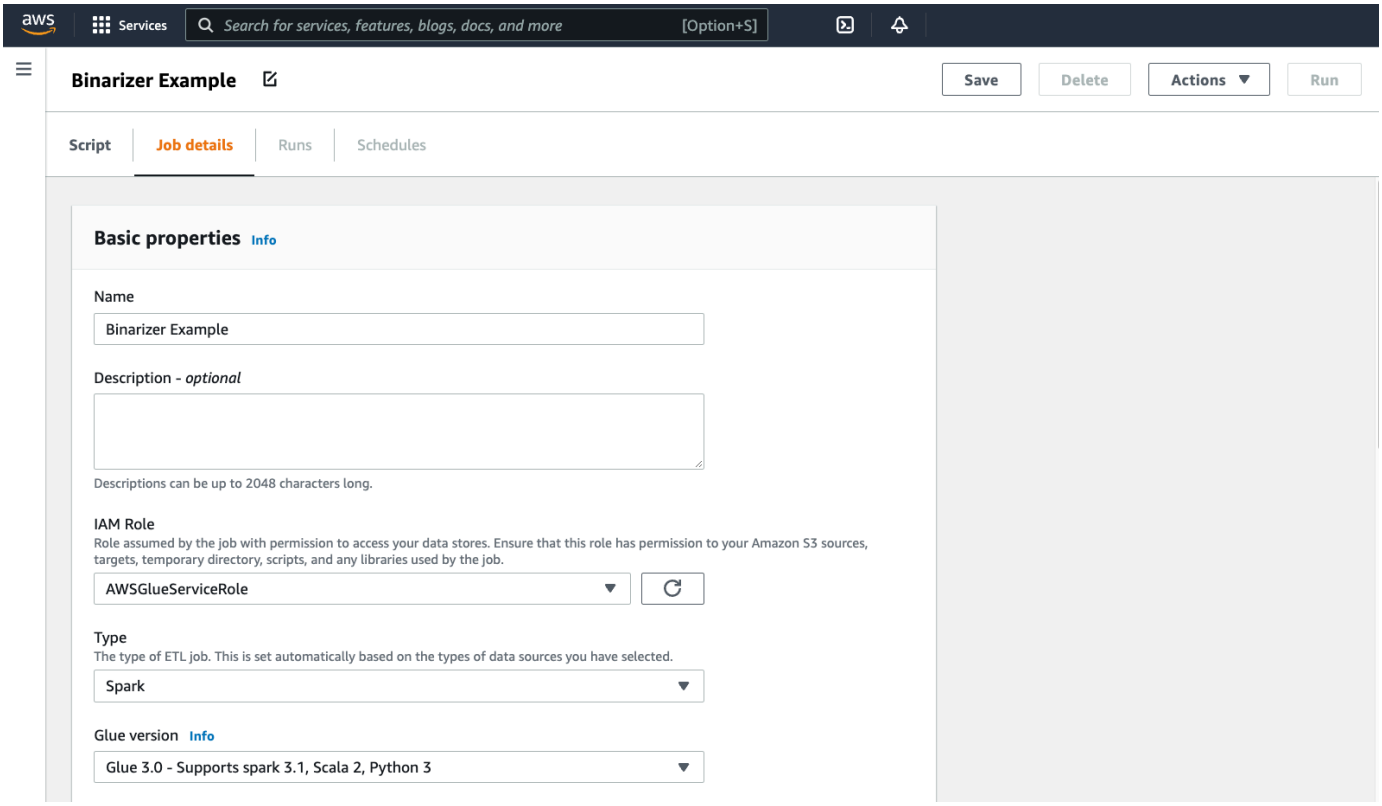
   Tombol Create akan muncul di header untuk panel Create job. Klik itu.



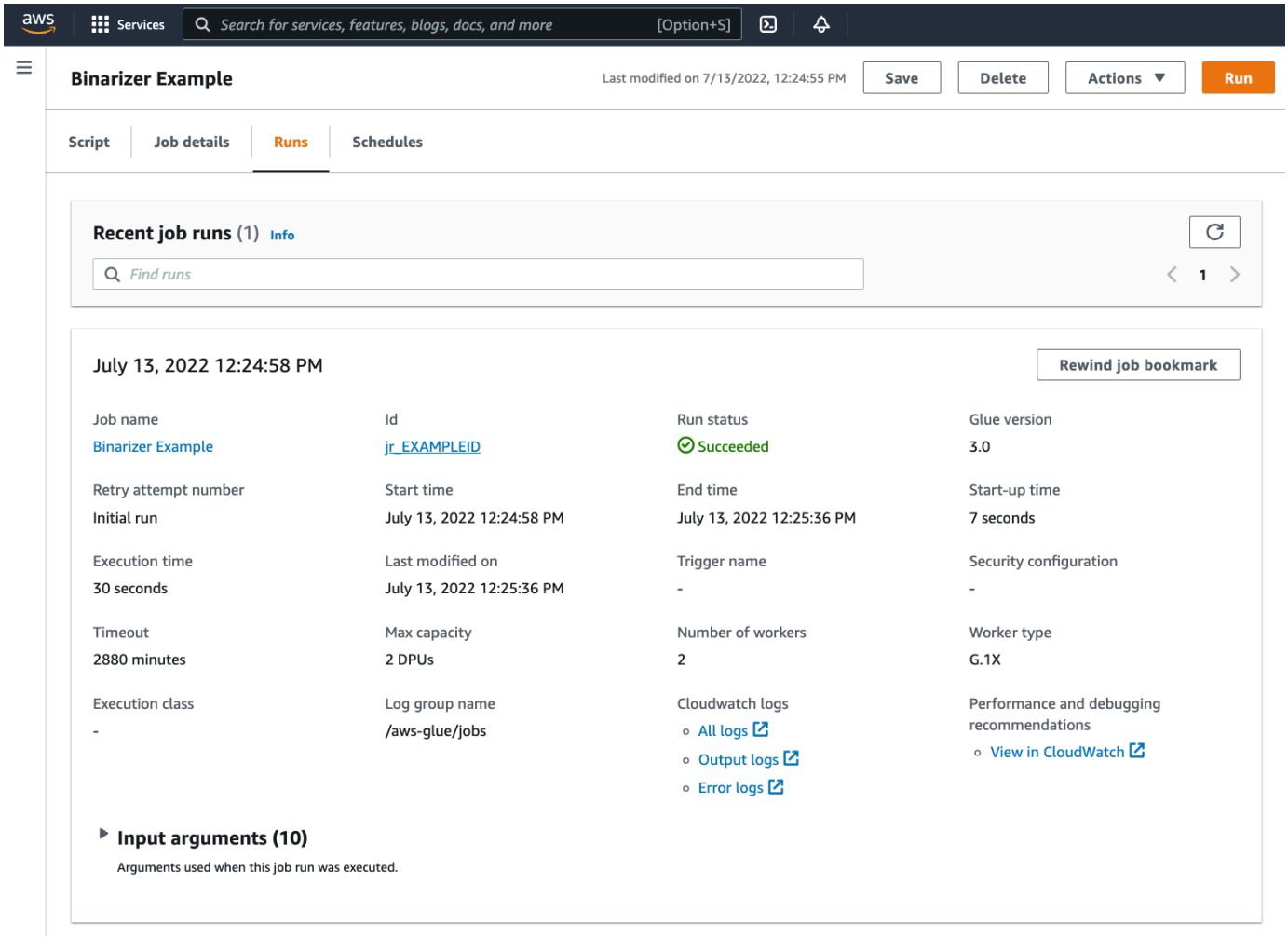
3. Browser Anda akan menavigasi ke editor skrip. Pada header, klik tab Job details. Tetapkan Nama dan Peran IAM. Untuk panduan seputar peran AWS Glue IAM, konsultasikanthe section called “Menyiapkan izin Peran IAM”.

   Opsional - atur jumlah pekerja yang diminta ke 2 dan Jumlah percobaan ulang ke. 1 Opsi ini berharga saat menjalankan pekerjaan produksi, tetapi menolaknya akan merampingkan pengalaman Anda saat menguji fitur.

   Di bilah judul, klik Simpan, lalu Jalankan

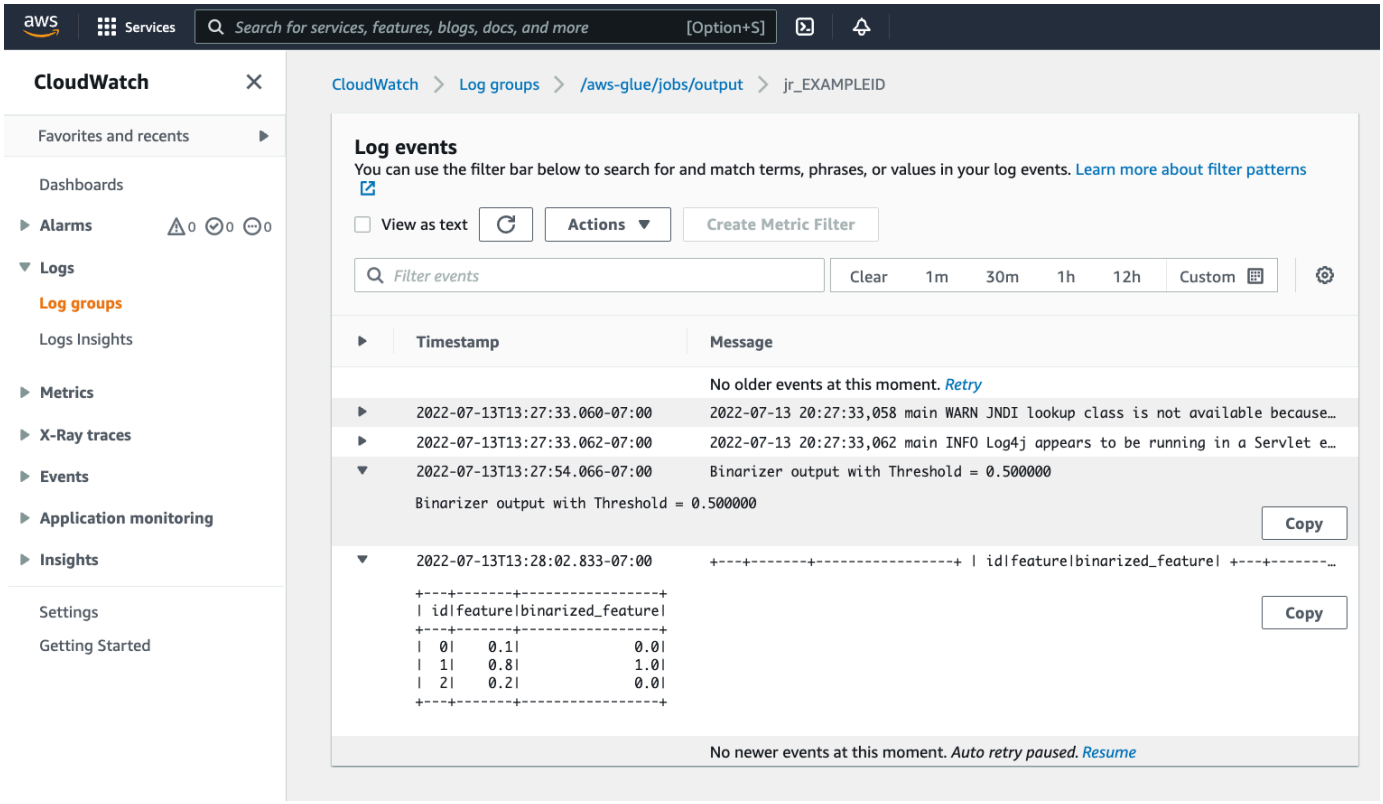


4. Arahkan ke tab Runs. Anda akan melihat panel yang sesuai dengan pekerjaan Anda. Tunggu beberapa menit dan halaman akan disegarkan secara otomatis untuk menampilkan Berhasil di bawah status Jalankan.



5. Anda akan ingin memeriksa output Anda untuk mengonfirmasi bahwa skrip Spark berjalan sebagaimana dimaksud. Skrip contoh Apache Spark ini harus menulis string ke aliran output. Anda dapat menemukannya dengan menavigasi ke log Output di bawah log Cloudwatch di panel agar pekerjaan berhasil dijalankan. Perhatikan id job run, id yang dihasilkan di bawah label Id yang dimulai dengan`jr_`.

   Ini akan membuka CloudWatch konsol, diatur untuk memvisualisasikan konten grup AWS Glue log default`/aws-glue/jobs/output`, disaring ke konten aliran log untuk id jalankan pekerjaan. Setiap pekerja akan membuat aliran log, ditampilkan sebagai baris di bawah aliran Log. Satu pekerja seharusnya menjalankan kode yang diminta. Anda harus membuka semua aliran log untuk mengidentifikasi pekerja yang benar. Setelah Anda menemukan pekerja yang tepat, Anda akan melihat output skrip, seperti yang terlihat pada gambar berikut:



Prosedur umum yang diperlukan untuk memigrasi program Spark

Menilai dukungan versi Spark

AWS Glueversi rilis menentukan versi Apache Spark dan Python yang tersedia untuk pekerjaan itu. AWS Glue Anda dapat menemukan AWS Glue versi kami dan apa yang mereka dukungthe section called “Versi AWS Glue”. Anda mungkin perlu memperbarui program Spark Anda agar kompatibel dengan versi Spark yang lebih baru untuk mengakses fitur tertentu. AWS Glue

Sertakan pustaka pihak ketiga

Banyak program Spark yang ada akan memiliki ketergantungan, baik pada artefak pribadi maupun publik. AWS Gluemendukung dependensi gaya JAR untuk Pekerjaan Scala serta dependensi Roda dan sumber pure-Python untuk pekerjaan Python.

Python - Untuk informasi tentang dependensi Python, lihat the section called “Pustaka Python”

Dependensi Python umum disediakan di AWS Glue lingkungan, termasuk pustaka Pandas yang umum diminta. Dependensi disertakan dalam AWS Glue Versi 2.0+. Untuk informasi selengkapnya tentang modul yang disediakan, lihatthe section called “Modul Python sudah disediakan di AWS

Glue". Jika Anda perlu menyediakan Job dengan versi dependensi yang berbeda yang disertakan secara default, Anda dapat menggunakannya`--additional-python-modules`. Untuk informasi tentang argumen pekerjaan, lihatthe section called "Parameter Tugas".

Anda dapat menyediakan dependensi Python tambahan dengan argumen pekerjaan. `--extra-py-files` Jika Anda memigrasikan pekerjaan dari program Spark, parameter ini adalah opsi yang baik karena secara fungsional setara dengan `--py-files` bendera di PySpark, dan tunduk pada batasan yang sama. Untuk informasi selengkapnya tentang `--extra-py-files` parameter, lihat the section called "Termasuk file Python dengan fitur asli PySpark "

Untuk pekerjaan baru, Anda dapat mengelola dependensi Python dengan argumen pekerjaan. `--additional-python-modules` Menggunakan argumen ini memungkinkan pengalaman manajemen ketergantungan yang lebih menyeluruh. Parameter ini mendukung dependensi gaya Roda, termasuk yang memiliki binding kode asli yang kompatibel dengan Amazon Linux 2.

Scala

Anda dapat menyediakan dependensi Scala tambahan dengan Job `--extra-jars` Argument. Dependensi harus di-host di Amazon S3 dan nilai argumen harus berupa daftar jalur Amazon S3 yang dibatasi koma tanpa spasi. Anda mungkin merasa lebih mudah untuk mengelola konfigurasi Anda dengan menggabungkan ulang dependensi Anda sebelum menghosting dan mengonfigurasinya. AWS Glue Dependensi JAR berisi bytecode Java, yang dapat dihasilkan dari bahasa JVM apa pun. Anda dapat menggunakan bahasa JVM lainnya, seperti Java, untuk menulis dependensi khusus.

Mengelola kredensi sumber data

Program Spark yang ada mungkin datang dengan konfigurasi yang kompleks atau khusus untuk menarik data dari sumber data mereka. Alur autentikasi sumber data umum didukung oleh koneksi. AWS Glue Untuk informasi selengkapnya tentang koneksi AWS Glue, lihat Menghubungkan ke data.

AWS Gluekoneksi memfasilitasi menghubungkan Job Anda ke berbagai jenis penyimpanan data dengan dua cara utama: melalui panggilan metode ke pustaka kami dan mengatur koneksi jaringan Tambahan di AWS konsol. Anda juga dapat memanggil AWS SDK dari dalam pekerjaan Anda untuk mengambil informasi dari koneksi.

Panggilan metode — AWS Glue Koneksi terintegrasi erat dengan Katalog AWS Glue Data, layanan yang memungkinkan Anda mengkurasi informasi tentang kumpulan data Anda, dan metode yang tersedia untuk berinteraksi dengan AWS Glue koneksi mencerminkan hal itu. Jika Anda memiliki

konfigurasi autentikasi yang ingin Anda gunakan kembali, untuk koneksi JDBC, Anda dapat mengakses konfigurasi AWS Glue koneksi Anda melalui metode pada`extract_jdbc_conf`. `GlueContext` Untuk informasi selengkapnya, lihat the section called “extract_jdbc_conf”

Konfigurasi konsol — AWS Glue Pekerjaan menggunakan AWS Glue koneksi terkait untuk mengonfigurasi koneksi ke subnet Amazon VPC. Jika Anda langsung mengelola materi keamanan Anda, Anda mungkin perlu menyediakan `NETWORK` jenis Koneksi jaringan tambahan di AWS konsol untuk mengonfigurasi perutean. Untuk informasi selengkapnya tentang API AWS Glue koneksi, lihat the section called “Koneksi”

Jika program Spark Anda memiliki alur autentikasi khusus atau tidak biasa, Anda mungkin perlu mengelola materi keamanan Anda secara langsung. Jika AWS Glue koneksi sepertinya tidak cocok, Anda dapat meng-host materi keamanan dengan aman di Secrets Manager dan mengaksesnya melalui boto3 atau AWS SDK, yang disediakan dalam pekerjaan.

Konfigurasikan Apache Spark

Migrasi kompleks sering mengubah konfigurasi Spark menjadi beban kerja mereka. Versi modern Apache Spark memungkinkan konfigurasi runtime diatur dengan file. `SparkSession` AWS Glue 3.0+ Pekerjaan disediakan`SparkSession`, yang dapat dimodifikasi untuk mengatur konfigurasi runtime. Konfigurasi Apache Spark. Tuning Spark rumit, dan AWS Glue tidak menjamin dukungan untuk mengatur semua konfigurasi Spark. Jika migrasi Anda memerlukan konfigurasi tingkat Spark yang substansif, hubungi dukungan.

Mengatur konfigurasi kustom

Program Spark yang dimigrasi dapat dirancang untuk mengambil konfigurasi khusus. AWS Gluememungkinkan konfigurasi diatur pada tingkat pekerjaan dan pekerjaan, melalui argumen pekerjaan. Untuk informasi tentang argumen pekerjaan, lihatthe section called “Parameter Tugas”. Anda dapat mengakses argumen pekerjaan dalam konteks pekerjaan melalui perpustakaan kami. AWS Gluemenyediakan fungsi utilitas untuk memberikan tampilan yang konsisten antara argumen yang ditetapkan pada pekerjaan dan argumen yang ditetapkan pada pekerjaan yang dijalankan. Lihat the section called “getResolvedOptions” di Python dan the section called “GlueArgParser” di Scala.

Migrasikan kode Java

Seperti dijelaskan dalamthe section called “Perpustakaan pihak ketiga”, dependensi Anda dapat berisi kelas yang dihasilkan oleh bahasa JVM, seperti Java atau Scala. Dependensi Anda dapat menyertakan metode. `main` Anda dapat menggunakan `main` metode dalam ketergantungan sebagai

titik masuk untuk pekerjaan Scala. AWS Glue Ini memungkinkan Anda untuk menulis `main` metode Anda di Java, atau menggunakan kembali `main` metode yang dikemas dengan standar perpustakaan Anda sendiri.

Untuk menggunakan `main` metode dari dependensi, lakukan hal berikut: Hapus isi panel pengeditan yang menyediakan objek default`GlueApp`. Berikan nama kelas yang sepenuhnya memenuhi syarat dalam ketergantungan sebagai argumen pekerjaan dengan kunci`--class`. Anda kemudian harus dapat memicu Job run.

Anda tidak dapat mengonfigurasi urutan atau struktur argumen yang AWS Glue diteruskan ke `main` metode. Jika kode Anda yang ada perlu membaca konfigurasi yang disetelAWS Glue, ini kemungkinan akan menyebabkan ketidakcocokan dengan kode sebelumnya. Jika Anda menggunakan`getResolvedOptions`, Anda juga tidak akan memiliki tempat yang baik untuk memanggil metode ini. Pertimbangkan untuk menjalankan dependensi Anda langsung dari metode utama yang dihasilkan oleh. AWS Glue Skrip AWS Glue ETL berikut menunjukkan contoh ini.

```
import com.amazonaws.services.glue.util.GlueArgParser

object GlueApp {
  def main(sysArgs: Array[String]) {
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)

    // Invoke static method from JAR. Pass some sample arguments as a String[], one
 defined inline and one taken from the job arguments, using getResolvedOptions
    com.mycompany.myproject.MyClass.myStaticPublicMethod(Array("string parameter1",
 args("JOB_NAME")))

    // Alternatively, invoke a non-static public method.
    (new com.mycompany.myproject.MyClass).someMethod()
  }
}
```

Bekerja dengan pekerjaan Ray di AWS Glue

Bagian ini memberikan informasi tentang penggunaan AWS Glue untuk pekerjaan Ray. Untuk informasi lebih lanjut tentang menulis AWS Glue untuk skrip Ray, lihat the section called “AWS Glue untuk Ray” bagian ini.

Topik

- Memulai dengan AWS Glue untuk Ray

- Lingkungan runtime Ray yang didukung
- Akuntansi untuk pekerja di pekerjaan Ray
- Menggunakan parameter pekerjaan di pekerjaan Ray
- Memantau pekerjaan Ray dengan metrik

Memulai dengan AWS Glue untuk Ray

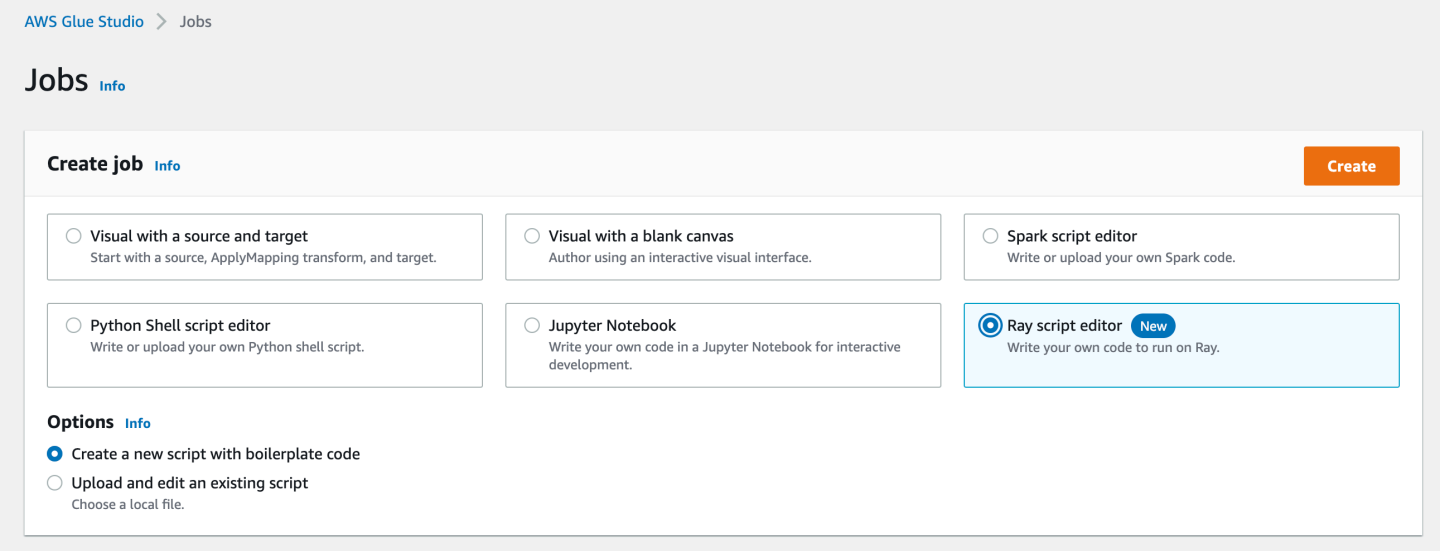
Untuk bekerja dengan AWS Glue Ray, Anda menggunakan AWS Glue pekerjaan yang sama dan sesi interaktif yang Anda gunakan AWS Glue untuk Spark. AWS Gluepekerjaan dirancang untuk menjalankan skrip yang sama pada irama berulang, sementara sesi interaktif dirancang untuk memungkinkan Anda menjalankan cuplikan kode secara berurutan terhadap sumber daya yang disediakan yang sama.

AWS GlueETL dan Ray berbeda di bawahnya, jadi dalam skrip Anda, Anda memiliki akses ke berbagai alat, fitur, dan konfigurasi. Sebagai kerangka komputasi baru yang dikelola olehAWS Glue, Ray memiliki arsitektur yang berbeda dan menggunakan kosakata yang berbeda untuk menggambarkan apa yang dilakukannya. Untuk informasi selengkapnya, lihat Whitepaper Arsitektur dalam dokumentasi Ray.

> **Note**
>
> AWS Glueuntuk Ray tersedia di AS Timur (Virginia N.), AS Timur (Ohio), AS Barat (Oregon), Asia Pasifik (Tokyo), dan Eropa (Irlandia).

Pekerjaan Ray di AWS Glue Studio konsol

Pada halaman Jobs di AWS Glue Studio konsol, Anda dapat memilih opsi baru saat Anda membuat pekerjaan di AWS Glue Studio — Ray script editor. Pilih opsi ini untuk membuat pekerjaan Ray di konsol. Untuk informasi selengkapnya tentang pekerjaan dan cara penggunaannya, lihatMembangun pekerjaan ETL visual dengan AWS Glue Studio.



lowongan kerja Ray di AWS CLI dan SDK

Pekerjaan Ray dalam AWS CLI menggunakan tindakan dan parameter SDK yang sama dengan pekerjaan lainnya. AWS Glueuntuk Ray memperkenalkan nilai-nilai baru untuk parameter tertentu. Untuk informasi selengkapnya di API Pekerjaan, lihatthe section called “Tugas”.

Lingkungan runtime Ray yang didukung

Dalam pekerjaan Spark, `GlueVersion` menentukan versi Apache Spark dan Python yang tersedia dalam pekerjaan untuk Spark. AWS Glue Versi Python menunjukkan versi yang didukung untuk pekerjaan jenis Spark. Ini bukan bagaimana lingkungan runtime Ray dikonfigurasi.

Untuk pekerjaan Ray, Anda harus mengatur `GlueVersion` ke `4.0` atau lebih besar. Namun, versi Ray, Python, dan pustaka tambahan yang tersedia di pekerjaan Ray Anda ditentukan oleh `Runtime` bidang dalam definisi pekerjaan.

Lingkungan `Ray2.4` runtime akan tersedia minimal 6 bulan setelah rilis. Saat Ray berkembang pesat, Anda akan dapat menggabungkan pembaruan dan peningkatan Ray melalui rilis lingkungan runtime masa depan.

Nilai yang valid: `Ray2.4`

Nilai runtime	Versi Ray dan Python
`Ray2.4`(untuk AWS Glue 4.0+)	Sinar 2.4.0

Nilai runtime	Versi Ray dan Python
	Python 3.9

Informasi tambahan

- Untuk catatan rilis yang menyertai AWS Glue rilis Ray, lihatthe section called “Versi AWS Glue”.
- Untuk library Python yang disediakan di lingkungan runtime, lihat. the section called “Modul disediakan dengan pekerjaan Ray”

Akuntansi untuk pekerja di pekerjaan Ray

AWS Gluemenjalankan pekerjaan Ray pada tipe pekerja EC2 berbasis Graviton baru, yang hanya tersedia untuk pekerjaan Ray. Untuk menyediakan pekerja ini dengan tepat untuk beban kerja yang dirancang Ray, kami menyediakan rasio sumber daya komputasi yang berbeda terhadap sumber daya memori dari sebagian besar pekerja. Untuk memperhitungkan sumber daya ini, kami menggunakan unit pemrosesan data yang dioptimalkan memori (M-DPU) daripada unit pemrosesan data standar (DPU).

- Satu M-DPU sesuai dengan 4 vCPU dan memori 32 GB.
- Satu DPU sesuai dengan 4 vCPU dan memori 16 GB. DPU digunakan untuk memperhitungkan sumber daya AWS Glue dengan pekerjaan Spark dan pekerja terkait.

Pekerjaan Ray saat ini memiliki akses ke satu jenis pekerja,`Z.2X`. `Z.2X`Pekerja memetakan ke 2 m-DPU (8 vCPU, memori 64 GB) dan memiliki ruang disk 128 GB. Sebuah `Z.2X` mesin menyediakan 8 pekerja Ray (satu per vCPU).

Jumlah m-DPU yang dapat Anda gunakan secara bersamaan di akun tunduk pada kuota layanan. Untuk informasi selengkapnya tentang batas AWS Glue akun Anda, lihat AWS Gluetitik akhir dan kuota.

Anda menentukan jumlah node pekerja yang tersedia untuk pekerjaan Ray dengan `--number-of-workers (NumberOfWorkers)` dalam definisi pekerjaan. Untuk informasi selengkapnya tentang nilai Ray di Jobs API, lihatthe section called “Tugas”.

Anda selanjutnya dapat menentukan jumlah minimum pekerja yang harus dialokasikan oleh pekerjaan Ray dengan parameter `--min-workers` pekerjaan. Untuk informasi selengkapnya tentang parameter pekerjaan, lihatthe section called “Referensi”.

Menggunakan parameter pekerjaan di pekerjaan Ray

Anda menetapkan argumen untuk pekerjaan AWS Glue Ray dengan cara yang sama seperti Anda menetapkan argumen AWS Glue untuk pekerjaan Spark. Untuk informasi selengkapnya tentang API AWS Glue, lihat the section called “Tugas”. Anda dapat mengkonfigurasi pekerjaan AWS Glue Ray dengan argumen yang berbeda, yang tercantum dalam referensi ini. Anda juga dapat memberikan argumen Anda sendiri.

Anda dapat mengonfigurasi pekerjaan melalui konsol, di tab Detail pekerjaan, di bawah judul Parameter Pekerjaan. Anda juga dapat mengonfigurasi pekerjaan AWS CLI melalui pengaturan `DefaultArguments` pekerjaan, atau pengaturan `Arguments` pada pekerjaan. Argumen default dan parameter pekerjaan tetap dengan pekerjaan melalui beberapa proses.

Misalnya, berikut ini adalah sintaksis untuk menjalankan sebuah tugas menggunakan `--arguments` untuk menetapkan parameter khusus.

```
$ aws glue start-job-run --job-name "CSV to CSV" --arguments='--scriptLocation="s3://
my_glue/libraries/test_lib.py",--test-environment="true"'
```

Setelah Anda mengatur argumen, Anda dapat mengakses parameter pekerjaan dari dalam pekerjaan Ray Anda melalui variabel lingkungan. Ini memberi Anda cara untuk mengonfigurasi pekerjaan Anda untuk setiap proses. Nama variabel lingkungan akan menjadi nama argumen pekerjaan tanpa `--` awalan.

Misalnya, dalam contoh sebelumnya, nama variabel akan menjadi `scriptLocation` dan`test-environment`. Anda kemudian akan mengambil argumen melalui metode yang tersedia di pustaka standar:`test_environment = os.environ.get('test-environment')`. Untuk informasi selengkapnya tentang mengakses variabel lingkungan dengan Python, lihat modul os dalam dokumentasi Python.

Konfigurasikan bagaimana pekerjaan Ray menghasilkan log

Secara default, pekerjaan Ray menghasilkan log dan metrik yang dikirim ke CloudWatch dan Amazon S3. Anda dapat menggunakan `--logging_configuration` parameter untuk mengubah cara log dihasilkan, saat ini Anda dapat menggunakannya untuk menghentikan pekerjaan Ray dari

menghasilkan berbagai jenis log. Parameter ini mengambil objek JSON, yang kuncinya sesuai dengan log/perilaku yang ingin Anda ubah. Ini mendukung kunci berikut:

- `CLOUDWATCH_METRICS`— Mengkonfigurasi rangkaian CloudWatch metrik yang dapat digunakan untuk memvisualisasikan kesehatan kerja. Untuk informasi lebih lanjut tentang metrik, lihat the section called “Metrik pekerjaan Ray”.
- `CLOUDWATCH_LOGS`— Mengkonfigurasi CloudWatch log yang memberikan rincian tingkat aplikasi Ray tentang status pekerjaan yang dijalankan. Untuk informasi lebih lanjut tentang log, lihat the section called “Memecahkan masalah kesalahan Ray”.
- `S3`— Mengkonfigurasi apa yang AWS Glue menulis ke Amazon S3, terutama informasi CloudWatch yang mirip dengan log tetapi sebagai file daripada aliran log.

Untuk menonaktifkan perilaku logging Ray, berikan nilainya`{\"IS_ENABLED\": \"False\"}`. Misalnya, untuk menonaktifkan CloudWatch metrik dan CloudWatch log, berikan konfigurasi berikut:

```
"--logging_configuration": "{\"CLOUDWATCH_METRICS\": {\"IS_ENABLED\": \"False\"},
 \"CLOUDWATCH_LOGS\": {\"IS_ENABLED\": \"False\"}}"
```

Referensi

Pekerjaan Ray mengenali nama argumen berikut yang dapat Anda gunakan untuk mengatur lingkungan skrip untuk pekerjaan dan pekerjaan Ray Anda:

- `--logging_configuration`— Digunakan untuk menghentikan pembuatan berbagai log yang dibuat oleh Ray jobs. Log ini dihasilkan secara default pada semua pekerjaan Ray. Format: Objek JSON yang lolos dari string. Untuk informasi selengkapnya, lihat the section called “Konfigurasikan bagaimana pekerjaan Ray menghasilkan log”.
- `--min-workers`— Jumlah minimum node pekerja yang dialokasikan untuk pekerjaan Ray. Node pekerja dapat menjalankan beberapa replika, satu per CPU virtual. Format: bilangan bulat. Minimal: 0. Maksimum: nilai yang ditentukan `--number-of-workers (NumberOfWorkers)` pada definisi pekerjaan. Untuk informasi selengkapnya tentang akuntansi untuk node pekerja, lihatthe section called “Akuntansi untuk pekerja di pekerjaan Ray”.
- `--object_spilling_config`— AWS Glue untuk dukungan Ray menggunakan Amazon S3 sebagai cara memperluas ruang yang tersedia untuk penyimpanan objek Ray. Untuk mengaktifkan perilaku ini, Anda dapat memberikan Ray objek yang menumpahkan objek konfigurasi JSON dengan parameter ini. Untuk informasi lebih lanjut tentang konfigurasi tumpahan objek Ray, lihat Object Spilling dalam dokumentasi Ray. Format: objek JSON.

AWS Glueuntuk Ray hanya mendukung tumpahan ke disk atau tumpah ke Amazon S3 sekaligus. Anda dapat menyediakan beberapa lokasi untuk tumpahan, selama mereka menghormati batasan ini. Saat tumpah ke Amazon S3, Anda juga perlu menambahkan izin IAM ke pekerjaan Anda untuk bucket ini.

Saat menyediakan objek JSON sebagai konfigurasi dengan CLI, Anda harus menyediakannya sebagai string, dengan objek JSON string-escaped. Misalnya, nilai string untuk tumpah ke satu jalur Amazon S3 akan terlihat seperti:. `"{\"type\": \"smart_open\", \"params\": {\"uri\":\"s3path\"}}"` DalamAWS Glue Studio, berikan parameter ini sebagai objek JSON tanpa pemformatan tambahan.

- `--object_store_memory_head`— Memori yang dialokasikan ke penyimpanan objek Plasma pada simpul kepala Ray. Instance ini menjalankan layanan manajemen klaster, serta replika pekerja. Nilai mewakili persentase memori bebas pada instance setelah awal yang hangat. Anda menggunakan parameter ini untuk menyetel beban kerja intensif memori — default dapat diterima untuk sebagian besar kasus penggunaan. Format: bilangan bulat positif. Minimal: 1. Maksimal: 100.

  Untuk informasi lebih lanjut tentang Plasma, lihat Toko Objek Dalam Memori Plasma di dokumentasi Ray.

- `--object_store_memory_worker`— Memori yang dialokasikan ke penyimpanan objek Plasma pada node pekerja Ray. Contoh ini hanya menjalankan replika pekerja. Nilai mewakili persentase memori bebas pada instance setelah awal yang hangat. Parameter ini digunakan untuk menyetel beban kerja intensif memori — default dapat diterima untuk sebagian besar kasus penggunaan. Format: bilangan bulat positif. Minimal: 1. Maksimal: 100.

  Untuk informasi lebih lanjut tentang Plasma, lihat Toko Objek Dalam Memori Plasma di dokumentasi Ray.

- `--pip-install`— Satu set paket Python yang akan diinstal. Anda dapat menginstal paket dari PyPI menggunakan argumen ini. Format: dibatasi koma daftar.

  Entri paket PyPI ada dalam format`package==version`, dengan nama PyPI dan versi paket target Anda. Entri menggunakan pencocokan versi Python untuk mencocokkan paket dan versi, `==` seperti, bukan yang sama. = Ada yang lain operator pencocokan versi. Untuk informasi lebih lanjut, lihat PEP 440 di situs web Python. Anda juga dapat menyediakan modul khusus dengan`--s3-py-modules`.

- `--s3-py-modules`- Satu set jalur Amazon S3 yang menampung distribusi modul Python. Format: daftar yang dibatasi koma.

  Anda dapat menggunakan ini untuk mendistribusikan modul Anda sendiri ke pekerjaan Ray Anda. Anda juga dapat menyediakan modul dari PyPI dengan. `--pip-install` Tidak seperti AWS Glue ETL, modul kustom tidak diatur melalui pip, tetapi diteruskan ke Ray untuk distribusi. Untuk informasi selengkapnya, lihat the section called “Modul Python tambahan untuk pekerjaan Ray”.
- `--working-dir`— Jalur ke file.zip yang dihosting di Amazon S3 yang berisi file yang akan didistribusikan ke semua node yang menjalankan pekerjaan Ray Anda. Format: string. Untuk informasi selengkapnya, lihat the section called “Menyediakan file untuk pekerjaan Ray Anda”.

Memantau pekerjaan Ray dengan metrik

Anda dapat memantau pekerjaan Ray menggunakan AWS Glue Studio dan Amazon CloudWatch. CloudWatch mengumpulkan dan memproses metrik mentah dari AWS Glue dengan Ray, yang membuatnya tersedia untuk analisis. Metrik ini divisualisasikan di AWS Glue Studio konsol, sehingga Anda dapat memantau pekerjaan Anda saat berjalan.

Untuk gambaran umum tentang cara memantauAWS Glue, lihatthe section called “Menggunakan CloudWatch metrik”. Untuk gambaran umum tentang cara menggunakan CloudWatch metrik yang diterbitkan olehAWS Glue, lihatthe section called “Pemantauan CloudWatch dengan”.

Memantau pekerjaan Ray di AWS Glue konsol

Pada halaman detail pentatan tugas, pada bagian Detail jalankan, Anda dapat melihat grafik agregat yang dibuat sebelumnya yang memvisualisasikan metrik tugas yang tersedia. AWS Glue Studiomengirim metrik tugas ke CloudWatch untuk setiap jalankan tugas. Dengan ini, Anda dapat membangun profil cluster dan tugas Anda, serta mengakses informasi terperinci tentang setiap node.

Untuk informasi selengkapnya tentang grafik metrik yang tersedia, lihat. the section called “Melihat Amazon CloudWatch metrik untuk menjalankan pekerjaan Ray”

Ikhtisar metrik pekerjaan Ray di CloudWatch

Kami menerbitkan metrik Ray saat pemantauan terperinci CloudWatch diaktifkan. Metrik dipublikasikan ke `Glue/Ray` CloudWatch namespace.

- Metrik instans

Kami menerbitkan metrik tentang CPU, memori, dan pemanfaatan disk dari instance yang ditugaskan ke pekerjaan. Metrik ini diidentifikasi oleh fitur seperti`ExecutorId`, `ExecutorType` dan`host`. Metrik ini adalah bagian dari metrik CloudWatch agen Linux standar. Anda dapat menemukan informasi tentang nama metrik dan fitur pada CloudWatch dokumentasi. Untuk informasi selengkapnya, lihat Metrik yang dikumpulkan oleh CloudWatch agen.

- Metrik kluster sinar

  Kami meneruskan metrik dari proses Ray yang menjalankan skrip Anda ke namespace ini, lalu memberikannya yang paling penting untuk Anda. Metrik yang tersedia mungkin berbeda menurut versi Ray. Untuk informasi selengkapnya tentang versi Ray yang menjalankan tugas Anda, lihatthe section called “Versi AWS Glue”.

  Ray mengumpulkan metrik pada tingkat instans. Ini juga menyediakan metrik untuk tugas dan cluster. Untuk informasi selengkapnya tentang strategi metrik dasar Ray, lihat Metrik dalam dokumentasi Ray.

> **Note**
>
> Kami tidak mempublikasikan metrik Ray ke `Glue/Job Metrics/` namespace, yang hanya digunakan untuk AWS Glue pekerjaan ETL.

Mengonfigurasi properti pekerjaan untuk pekerjaan shell Python di AWS Glue

Anda dapat menggunakan tugas shell Python untuk menjalankan skrip Python sebagai shell di AWS Glue. Dengan pekerjaan shell Python, Anda dapat menjalankan skrip yang kompatibel dengan Python 3.6 atau Python 3.9.

Topik

- Batasan
- Mendefinisikan properti pekerjaan untuk pekerjaan shell Python
- Pustaka yang didukung untuk pekerjaan shell Python
- Menyediakan pustaka Python Anda sendiri
- Gunakan AWS CloudFormation dengan pekerjaan shell Python di AWS Glue

Batasan

Perhatikan batasan pekerjaan Python Shell berikut ini:

- Anda tidak dapat menggunakan bookmark tugas dengan tugas shell Python tersebut.
- Anda tidak dapat mengemas pustaka Python apa pun sebagai file `.egg` di Python 3.9+. Sebaliknya, gunakan `.whl`.
- `--extra-files`Opsi ini tidak dapat digunakan, karena pembatasan salinan sementara data S3.

Mendefinisikan properti pekerjaan untuk pekerjaan shell Python

Bagian ini menjelaskan mendefinisikan properti pekerjaan diAWS Glue Studio, atau menggunakan AWS CLI.

AWS Glue Studio

Saat Anda menentukan pekerjaan shell Python AndaAWS Glue Studio, Anda memberikan beberapa properti berikut:

Peran IAM

Tentukan peran AWS Identity and Access Management (IAM) yang digunakan untuk otorisasi sumber daya yang digunakan untuk menjalankan pekerjaan dan mengakses penyimpanan data. Untuk informasi selengkapnya tentang izin menjalankan tugas di AWS Glue, lihat Manajemen identitas dan akses untuk AWS Glue.

Jenis

Pilih Python shell untuk menjalankan skrip Python dengan perintah tugas bernama `pythonshell`.

Versi Python

Pilih versi Python. Defaultnya adalah Python 3.9. Versi yang valid adalah Python 3.6 dan Python 3.9.

Muat pustaka analitik umum (Disarankan)

Pilih opsi ini untuk menyertakan pustaka umum untuk Python 3.9 di shell Python.

Jika pustaka Anda kustom atau bertentangan dengan yang sudah diinstal sebelumnya, Anda dapat memilih untuk tidak menginstal pustaka umum. Namun, Anda dapat menginstal pustaka tambahan selain pustaka umum.

Ketika Anda memilih opsi ini, `library-set` opsi diatur ke`analytics`. Saat Anda membatalkan pilihan ini, `library-set` opsi diatur ke`none`.

Nama file skrip dan jalur Skrip

Kode dalam Skrip mendefinisikan logika prosedural tugas Anda. Anda menyediakan nama skrip dan lokasi di Amazon Simple Storage Service (Amazon S3). Konfirmasi bahwa tidak ada file dengan nama yang sama sebagai direktori skrip pada path. Untuk mem-pelajari selengkapnya tentang menggunakan skrip, lihat AWS Glue panduan pemrograman.

Skrip

Kode dalam Skrip mendefinisikan logika prosedural tugas Anda. Anda dapat membuat kode skrip dengan Python 3.6 atau Python 3.9. Anda dapat mengedit skrip diAWS Glue Studio.

Unit pengolahan data

Jumlah maksimum unit pengolahan data (DPU) AWS Glue yang dapat dialokasikan ketika tugas ini berjalan. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi selengkapnya, lihat harga AWS Glue.

Anda dapat mengatur nilai-nya ke 0,0625 atau 1. Default-nya adalah 0,0625. Dalam kedua kasus, disk lokal untuk contoh akan menjadi 20GB.

CLI

Anda juga dapat membuat pekerjaan shell Python menggunakan AWS CLI, seperti pada contoh berikut.

```
aws glue create-job --name python-job-cli --role Glue_DefaultRole
    --command '{"Name" :  "pythonshell", "PythonVersion": "3.9", "ScriptLocation" :
"s3://DOC-EXAMPLE-BUCKET/scriptname.py"}'
    --max-capacity 0.0625
```

> **Note**
>
> Anda tidak perlu menentukan versi AWS Glue karena parameter `--glue-version` tidak berlaku untuk pekerjaan AWS Glue shell. Versi apa pun yang ditentukan akan diabaikan.

Pekerjaan yang Anda buat dengan AWS CLI default ke Python 3. Versi Python yang valid adalah 3 (sesuai dengan 3.6), dan 3.9. Untuk menentukan Python 3.6, tambahkan tuple ini ke parameter: `--command "PythonVersion":"3"`

Untuk menentukan Python 3.9, tambahkan tuple ini ke parameter: `--command "PythonVersion":"3.9"`

Untuk mengatur kapasitas maksimal yang digunakan oleh tugas shell Python, berikan parameter `--max-capacity`. Untuk tugas shell Python, parameter `--allocated-capacity` tidak dapat digunakan.

Pustaka yang didukung untuk pekerjaan shell Python

Dalam Python shell menggunakan Python 3.9, Anda dapat memilih set perpustakaan untuk menggunakan set perpustakaan pra-paket untuk kebutuhan Anda. Anda dapat menggunakan `library-set` opsi untuk memilih kumpulan perpustakaan. Nilai yang valid adalah`analytics`, dan`none`.

Lingkungan untuk menjalankan tugas shell Python mendukung perpustakaan berikut:

Versi Python	Python 3.6	Python 3.9	
Set perpustakaan	N/A	analitik	tidak satupun
avro		1.11.0	
awscli	116.242	1.23.5	1.23.5
awswrangler		2.15.1	
botocore	1.12.232	1.24.21	1.23.5
boto3	1.9.203	1.21.21	

Versi Python	Python 3.6	Python 3.9	
elasticsearch		8.2.0	
numpy	1.16.2	1.22.3	
panda	0.24.2	1.4.2	
psycopg2		2.9.3	
pyathena		2.5.3	
PyGreSQL	5.0.6		
PyMySQL		1.0.2	
pyodbc		4.0.32	
pyorc		0.6.0	
konektor pergeseran merah		2.0.907	
permintaan	2.22.0	2.27.1	
scikit-belajar	0.20.3	1.0.2	
scipy	1.2.1	1.8.0	
SQLAlchemy		1.4.36	
s3fs		2022.3.0	

Anda dapat menggunakan perpustakaan `NumPy` dalam sebuah tugas shell Python untuk komputasi ilmiah. Untuk informasi lebih lanjut, lihat NumPy. Contoh berikut menunjukkan NumPy skrip yang dapat digunakan dalam pekerjaan shell Python. Skrip mencetak "Hello world" dan hasil dari beberapa perhitungan matematis.

```
import numpy as np
print("Hello world")
```

```
a = np.array([20,30,40,50])
print(a)

b = np.arange( 4 )

print(b)

c = a-b

print(c)

d = b**2

print(d)
```

Menyediakan pustaka Python Anda sendiri

Menggunakan PIP

Python shell menggunakan Python 3.9 memungkinkan Anda menyediakan modul Python tambahan atau versi yang berbeda di tingkat pekerjaan. Anda dapat menggunakan `--additional-python-modules` dengan daftar modul Python yang dipisahkan koma untuk menambahkan modul baru atau mengubah versi dari modul yang ada. Anda tidak dapat menyediakan modul Python khusus yang dihosting di Amazon S3 dengan parameter ini saat menggunakan pekerjaan shell Python.

Misalnya untuk memperbarui atau menambahkan `scikit-learn` modul baru gunakan kunci dan nilai berikut:`"--additional-python-modules", "scikit-learn==0.21.3"`.

AWS Glue menggunakan Python Package Installer (pip3) untuk menginstal modul tambahan. Anda dapat meneruskan opsi pip3 tambahan di dalam nilai. `--additional-python-modules` Misalnya, `"scikit-learn==0.21.3 -i https://pypi.python.org/simple/"`. Ketidakcocokan atau batasan apa pun dari pip3 berlaku.

> **Note**
>
> Untuk menghindari ketidakcocokan di masa mendatang, sebaiknya gunakan library yang dibuat untuk Python 3.9.

Menggunakan file Egg atau Whl

Anda mungkin sudah memiliki satu atau beberapa perpustakaan Python yang dikemas sebagai sebuah file `.egg` atau `.whl`. Jika demikian, Anda dapat menentukannya ke tugas Anda dengan menggunakan AWS Command Line Interface (AWS CLI) pada bendera "`--extra-py-files`", seperti dalam contoh berikut.

```
aws glue create-job --name python-redshift-test-cli --role role --command '{"Name" :
 "pythonshell", "ScriptLocation" : "s3://MyBucket/python/library/redshift_test.py"}'
    --connections Connections=connection-name --default-arguments '{"--extra-py-
files" : ["s3://DOC-EXAMPLE-BUCKET/EGG-FILE", "s3://DOC-EXAMPLE-BUCKET/WHEEL-FILE"]}'
```

Jika Anda tidak benar-benar mengetahui cara membuat sebuah file `.egg` atau `.whl` dari perpustakaan Python, maka gunakan langkah-langkah berikut. Contoh ini berlaku di macOS, Linux, dan Windows Subsystem for Linux (WSL).

Untuk membuat sebuah file .egg atau file .whl Python

1. Buat klaster Amazon Redshift di sebuah virtual private cloud (VPC), dan tambahkan beberapa data ke tabel.
2. Buat AWS Glue koneksi untuk kombinasi VPC- SecurityGroup -Subnet yang Anda gunakan untuk membuat cluster. Menguji apakah koneksi berhasil.
3. Buatlah sebuah direktori bernama `redshift_example`, dan buatlah file bernama `setup.py`. Tempel kode berikut ke `setup.py`.

   ```
   from setuptools import setup

   setup(
       name="redshift_module",
       version="0.1",
       packages=['redshift_module']
   )
   ```

4. Pada direktori `redshift_example`, buat `redshift_module`. Pada direktori `redshift_module`, buat file `__init__.py` dan file `pygresql_redshift_common.py`.
5. Biarkan file `__init__.py` kosong. Di `pygresql_redshift_common.py`, tempel kode berikut. Ganti *`port`*, *`db_name`*, *`pengguna`*, dan *`password_for_user`* dengan detail khusus untuk klaster Amazon Redshift Anda. Ganti *`table_name`* dengan nama tabel di Amazon Redshift.

```
import pg


def get_connection(host):
    rs_conn_string = "host=%s port=%s dbname=%s user=%s password=%s" % (
        host, port, db_name, user, password_for_user)

    rs_conn = pg.connect(dbname=rs_conn_string)
    rs_conn.query("set statement_timeout = 1200000")
    return rs_conn


def query(con):
    statement = "Select * from table_name;"
    res = con.query(statement)
    return res
```

6. Jika Anda belum sampai di langkah itu, ubah ke direktori `redshift_example`.
7. Lakukan salah satu hal berikut ini:

   - Untuk membuat sebuah file `.egg`, jalankan perintah berikut.

     ```
     python setup.py bdist_egg
     ```

   - Untuk membuat sebuah file `.whl`, jalankan perintah berikut.

     ```
     python setup.py bdist_wheel
     ```

8. Menginstal dependensi yang diperlukan untuk perintah sebelumnya.
9. Perintah tersebut membuat file di direktori `dist`:

   - Jika Anda membuat sebuah file egg, namanya adalah `redshift_module-0.1-py2.7.egg`.
   - Jika Anda membuat sebuah file roda, namanya adalah `redshift_module-0.1-py2.7-none-any.whl`.

   Unggah file ini ke Amazon S3.

   Dalam contoh ini, jalur file yang diunggah adalah *`s3://DOC-EXAMPLE-BUCKET/EGG-FILE`* atau *`s3://DOC-EXAMPLE-BUCKET/WHEEL-FILE`*.

10. Buat sebuah file Python untuk digunakan sebagai skrip untuk tugas AWS Glue, dan tambahkan kode berikut ke file tersebut.

```
from redshift_module import pygresql_redshift_common as rs_common

con1 = rs_common.get_connection(redshift_endpoint)
res = rs_common.query(con1)

print "Rows in the table cities are: "

print res
```

11. Unggah file sebelumnya ke Amazon S3. Dalam contoh ini, jalur file yang diunggah adalah *s3://DOC-EXAMPLE-BUCKET/scriptname.py.*

12. Buat sebuah tugas shell Python menggunakan skrip ini. Pada konsol AWS Glue, pada halaman Properti Tugas, tentukan path ke file `.egg/.whl` pada kotak Path perpustakaan Python. Jika Anda memiliki beberapa file `.egg/.whl` dan file Python, berikan daftar dipisahkan koma dalam kotak ini.

    Saat memodifikasi atau mengganti nama `.egg`, nama file harus menggunakan nama default yang dihasilkan oleh perintah "python setup.py bdist_egg" atau harus mematuhi konvensi penamaan modul Python. Untuk informasi selengkapnya, lihat bagian Panduan Gaya untuk Kode Python.

    Menggunakan AWS CLI, membuat pekerjaan dengan perintah, seperti pada contoh berikut.

```
aws glue create-job --name python-redshift-test-cli --role Role --command
 '{"Name" :  "pythonshell", "ScriptLocation" : "s3://DOC-EXAMPLE-BUCKET/
scriptname.py"}'
     --connections Connections="connection-name" --default-arguments '{"--extra-
py-files" : ["s3://DOC-EXAMPLE-BUCKET/EGG-FILE", "s3://DOC-EXAMPLE-BUCKET/WHEEL-
FILE"]}'
```

    Ketika tugas tersebut berjalan, skrip akan mencetak baris yang dibuat dalam tabel *table_name* di klaster Amazon Redshift.

Gunakan AWS CloudFormation dengan pekerjaan shell Python di AWS Glue

Anda dapat menggunakan AWS CloudFormation dengan pekerjaan shell Python di. AWS Glue Berikut adalah contohnya:

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  Python39Job:
    Type: 'AWS::Glue::Job'
    Properties:
      Command:
        Name: pythonshell
        PythonVersion: '3.9'
        ScriptLocation: 's3://bucket/location'
      MaxRetries: 0
      Name: python-39-job
      Role: RoleName
```

Grup Amazon CloudWatch Logs untuk output pekerjaan shell Python adalah. `/aws-glue/python-jobs/output` Untuk kesalahan, lihat grup log `/aws-glue/python-jobs/error`.

AWS Glue Pemantauan

Pemantauan adalah bagian penting dari pemeliharaan keandalan, ketersediaan, dan kinerja AWS Glue dan solusi AWS lain Anda. AWS menyediakan alat pemantauan untuk memantau AWS Glue, melaporkan jika ada yang salah, dan mengambil tindakan secara otomatis jika diperlukan:

Anda dapat menggunakan alat pemantauan otomatis berikut untuk melihat AWS Glue dan melaporkan saat terjadi kesalahan:

- Amazon CloudWatch Events memberikan aliran peristiwa sistem yang mendekati waktu nyata yang menjelaskan perubahan AWS sumber daya. CloudWatch Acara memungkinkan komputasi berbasis peristiwa otomatis. Anda dapat menulis aturan yang mengawasi kejadian tertentu dan memicu tindakan otomatis dalam layanan AWS lainnya saat kejadian ini terjadi. Untuk informasi selengkapnya, lihat Panduan Pengguna CloudWatch Acara Amazon.
- Amazon CloudWatch Logs memungkinkan Anda memantau, menyimpan, dan mengakses file log Anda dari instans Amazon EC2,AWS CloudTrail, dan sumber lainnya. CloudWatch Log

dapat memantau informasi dalam file log dan memberi tahu Anda ketika ambang batas tertentu terpenuhi. Anda juga dapat mengarsipkan data log Anda dalam penyimpanan yang sangat tahan lama. Untuk informasi selengkapnya, lihat Panduan Pengguna Amazon CloudWatch Logs.

- AWS CloudTrail merekam panggilan API dan peristiwa terkait yang dilakukan oleh atau atas nama akun AWS Anda dan mengirimkan berkas log ke bucket Amazon S3 yang Anda tentukan. Anda dapat mengidentifikasi pengguna dan akun mana yang memanggil AWS, alamat IP sumber yang melakukan panggilan, dan kapan panggilan tersebut terjadi. Untuk informasi selengkapnya, lihat Panduan Pengguna AWS CloudTrail.

Selain itu, Anda memiliki akses ke wawasan berikut di AWS Glue konsol untuk membantu Anda men-debug dan pekerjaan profil:

- Spark jobs — Anda dapat melihat visualisasi rangkaian CloudWatch metrik yang dipilih, dan pekerjaan yang lebih baru memiliki akses ke UI Spark. Untuk informasi selengkapnya, lihat the section called “Lowongan kerja Monitoring Spark”.
- Pekerjaan Ray — Anda dapat melihat visualisasi seri CloudWatch metrik yang dipilih. Untuk informasi selengkapnya, lihat the section called “Metrik pekerjaan Ray”.

Topik

- AWS tag di AWS Glue
- Mengotomatisasi AWS Glue dengan Acara CloudWatch
- AWS Gluepemantauan sumber daya
- Mencatat panggilan API AWS Glue dengan AWS CloudTrail

AWS tag di AWS Glue

Untuk membantu Anda mengelola sumber daya AWS Glue, Anda dapat secara opsional menetapkan tag Anda sendiri untuk beberapa jenis sumber daya AWS Glue. Tag adalah label yang Anda tetapkan ke AWS sumber daya. Setiap tanda terdiri dari kunci dan nilai opsional, yang keduanya Anda tentukan. Anda menggunakan tag di AWS Glue untuk mengorganisasi dan mengatur sumber daya Anda. Tag dapat digunakan untuk membuat laporan hitungan biaya dan membatasi akses ke sumber daya. Jika Anda menggunakan AWS Identity and Access Management, Anda dapat mengontrol pengguna mana di AWS akun Anda yang memiliki izin untuk membuat, mengedit, atau menghapus tag. Selain izin untuk memanggil API terkait tag, Anda juga memerlukan izin untuk memanggil API

penandaan pada koneksi, dan `glue:GetConnection` izin untuk memanggil API penandaan pada database. `glue:GetDatabase` Untuk informasi selengkapnya, lihat ABAC dengan AWS Glue.

Di AWS Glue, Anda dapat menandai sumber daya berikut:

- Koneksi
- Basis data
- Crawler
- Sesi interaktif
- Titik akhir pengembangan
- Tugas
- Pemicu
- Alur kerja
- Cetak biru
- Transformasi Machine Learning
- Aturan kualitas data
- Skema aliran
- Pendaftaran skema aliran

> **Note**
>
> Sebagai praktik terbaik, untuk memungkinkan penandaan pada sumber daya AWS Glue, selalu sertakan tindakan `glue:TagResource` dalam kebijakan Anda.

Pertimbangkan hal berikut saat menggunakan tag dengan AWS Glue.

- Maksimum 50 tag didukung untuk setiap entitas.
- Di AWS Glue, Anda menentukan tag sebagai daftar pasangan nilai kunci dalam format `{"string": "string" ...}`
- Bila Anda membuat tag pada sebuah objek, maka kunci tag adalah wajib, sementara nilai tag adalah opsional.
- Kunci dan nilai tag peka huruf besar kecil.

- Kunci tag dan nilai tag tidak boleh berisi prefiks aws. Tidak ada operasi yang diizinkan pada tag tersebut.
- Panjang kunci tag maksimum adalah 128 karakter Unicode dalam UTF-8. Kunci tag tidak boleh kosong atau nol.
- Panjang nilai tag maksimum adalah 256 karakter Unicode dalam UTF-8. Nilai tag bisa kosong atau nol.

Menandai dukungan untuk koneksi AWS Glue

Anda dapat membatasi`CreateConnection`,`UpdateConnection`, `GetConnection` dan, izin `DeleteConnection` tindakan berdasarkan tag sumber daya. Ini memungkinkan Anda untuk menerapkan kontrol akses hak istimewa paling sedikit pada AWS Glue pekerjaan dengan sumber data JDBC yang perlu mengambil informasi koneksi JDBC dari Katalog Data.

Contoh penggunaan

Buat AWS Glue koneksi dengan tag ["connection-category”, “dev-test"].

Tentukan kondisi tag untuk `GetConnection` tindakan dalam kebijakan IAM.

```
{
   "Effect": "Allow",
   "Action": [
     "glue:GetConnection"
     ],
   "Resource": "*",
   "Condition": {
     "ForAnyValue:StringEquals": {
       "aws:ResourceTag/tagKey": "dev-test"
     }
   }
 }
```

Contoh-contoh

Contoh berikut membuat tugas dengan tag yang ditetapkan.

AWS CLI

```
aws glue create-job --name job-test-tags --role MyJobRole --command
 Name=glueetl,ScriptLocation=S3://aws-glue-scripts//prod-job1
```

```
--tags key1=value1,key2=value2
```

AWS CloudFormation JSON

```
{
  "Description": "AWS Glue Job Test Tags",
  "Resources": {
    "MyJobRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  "glue.amazonaws.com"
                ]
              },
              "Action": [
                "sts:AssumeRole"
              ]
            }
          ]
        },
        "Path": "/",
        "Policies": [
          {
            "PolicyName": "root",
            "PolicyDocument": {
              "Version": "2012-10-17",
              "Statement": [
                {
                  "Effect": "Allow",
                  "Action": "*",
                  "Resource": "*"
                }
              ]
            }
          }
        ]
      }
```

```
    },
    "MyJob": {
      "Type": "AWS::Glue::Job",
      "Properties": {
        "Command": {
          "Name": "glueetl",
          "ScriptLocation": "s3://aws-glue-scripts//prod-job1"
        },
        "DefaultArguments": {
          "--job-bookmark-option": "job-bookmark-enable"
        },
        "ExecutionProperty": {
          "MaxConcurrentRuns": 2
        },
        "MaxRetries": 0,
        "Name": "cf-job1",
        "Role": {
           "Ref": "MyJobRole",
           "Tags": {
                "key1": "value1",
                "key2": "value2"
           }
        }
      }
    }
  }
}
```

AWS CloudFormation YAML

```
Description: AWS Glue Job Test Tags
Resources:
  MyJobRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - glue.amazonaws.com
            Action:
```

```
              - sts:AssumeRole
      Path: "/"
      Policies:
        - PolicyName: root
          PolicyDocument:
            Version: '2012-10-17'
            Statement:
              - Effect: Allow
                Action: "*"
                Resource: "*"
  MyJob:
    Type: AWS::Glue::Job
    Properties:
      Command:
        Name: glueetl
        ScriptLocation: s3://aws-glue-scripts//prod-job1
      DefaultArguments:
        "--job-bookmark-option": job-bookmark-enable
      ExecutionProperty:
        MaxConcurrentRuns: 2
      MaxRetries: 0
      Name: cf-job1
      Role:
        Ref: MyJobRole
        Tags:
           key1: value1
           key2: value2
```

Untuk mengetahui informasi lebih lanjut, lihat AWS Strategi Penandaan.

Untuk informasi tentang cara mengontrol akses menggunakan tag, lihat ABAC dengan AWS Glue.

Mengotomatisasi AWS Glue dengan Acara CloudWatch

Anda dapat menggunakan Amazon CloudWatch Events untuk mengotomatiskan AWS layanan Anda dan merespons secara otomatis peristiwa sistem seperti masalah ketersediaan aplikasi atau perubahan sumber daya. Acara dari AWS layanan dikirim ke CloudWatch Acara dalam waktu dekat. Anda dapat menulis aturan sederhana untuk menunjukkan kejadian mana yang sesuai kepentingan Anda, dan tindakan otomatis apa yang diambil ketika suatu kejadian sesuai dengan suatu aturan. Tindakan yang dapat dipicu secara otomatis meliputi hal berikut:

- Mengambil fungsi AWS Lambda

- Meminta Perintah Amazon EC2 Run
- Mengirim peristiwa ke Amazon Kinesis Data Streams
- Mengaktifkan mesin keadaan AWS Step Functions
- Memberi tahu topik Amazon SNS atau antrean Amazon SQS

Beberapa contoh menggunakan CloudWatch Events dengan AWS Glue meliputi yang berikut:

- Mengaktifkan fungsi Lambda ketika tugas ETL berhasil
- Memberi tahu topik Amazon SNS saat tugas ETL gagal

CloudWatch Peristiwa berikut dihasilkan olehAWS Glue.

- Peristiwa untuk `"detail-type":"Glue Job State Change"` yang dihasilkan untuk `SUCCEEDED`, `FAILED`, `TIMEOUT`, dan `STOPPED`.
- Peristiwa untuk `"detail-type":"Glue Job Run Status"` yang dihasilkan untuk `RUNNING`, `STARTING`, dan eksekusi tugas `STOPPING` ketika mereka melampaui ambang batas notifikasi penundaan tugas. Anda harus menyetel properti ambang notifikasi penundaan pekerjaan untuk menerima peristiwa ini.

  Hanya satu peristiwa yang dihasilkan per status menjalankan pekerjaan ketika ambang pemberitahuan penundaan pekerjaan terlampaui.
- Peristiwa untuk `"detail-type":"Glue Crawler State Change"` yang dihasilkan untuk `Started`, `Succeeded`, dan `Failed`.
- Peristiwa untuk `"detail-type":"Glue Data Catalog Database State Change"` yang dihasilkan untuk `CreateDatabase`, `DeleteDatabase`, `CreateTable`, `DeleteTable` dan `BatchDeleteTable`. Misalnya, jika tabel dibuat atau dihapus, pemberitahuan dikirim ke CloudWatch Acara. Perhatikan bahwa Anda tidak dapat menulis program yang tergantung pada urutan keberadaan atau notifikasi peristiwa, karena program tersebut mungkin tidak berurutan atau hilang. Peristiwa dipancarkan atas dasar upaya terbaik. Dalam detail notifikasi tersebut:
  - `typeOfChange` memberi tahu nama operasi API.
  - `databaseName` memberi tahu nama basis data yang terpengaruh.
  - `changedTables` memberi tahu hingga 100 nama tabel yang terpengaruh per notifikasi. Ketika nama tabel panjang, mungkin dibuat dalam beberapa notifikasi.

- Acara untuk `"detail-type":"Glue Data Catalog Table State Change"` dihasilkan untuk`UpdateTable`,`CreatePartition`,`BatchCreatePartition`,`UpdatePartition`,`DeletePartit` `BatchUpdatePartition` dan`BatchDeletePartition`. Misalnya, jika tabel atau partisi diperbarui, pemberitahuan dikirim ke CloudWatch Acara. Perhatikan bahwa Anda tidak dapat menulis program yang tergantung pada urutan keberadaan atau notifikasi peristiwa, karena program tersebut mungkin tidak berurutan atau hilang. Peristiwa dipancarkan atas dasar upaya terbaik. Dalam detail notifikasi tersebut:
  - `typeOfChange` memberi tahu nama operasi API.
  - `databaseName` memberi tahu nama basis data yang berisi sumber daya yang terpengaruh.
  - `tableName` memberi tahu nama tabel yang terpengaruh.
  - `changedPartitions` menentukan hingga 100 partisi yang terpengaruh dalam satu notifikasi. Ketika nama partisi panjang, mungkin dibuat dalam beberapa notifikasi.

    Misalnya jika ada dua tombol partisi, `Year` dan `Month`, maka `"2018,01", "2018,02"` akan memodifikasi partisi adalah `"Year=2018" and "Month=01"` dan partisi adalah `"Year=2018" and "Month=02"`.

```
{
    "version":"0",
    "id":"abcdef00-1234-5678-9abc-def012345678",
    "detail-type":"Glue Data Catalog Table State Change",
    "source":"aws.glue",
    "account":"123456789012",
    "time":"2017-09-07T18:57:21Z",
    "region":"us-west-2",
    "resources":["arn:aws:glue:us-west-2:123456789012:database/default/foo"],
    "detail":{
        "changedPartitions": [
            "2018,01",
            "2018,02"
        ],
        "databaseName": "default",
        "tableName": "foo",
        "typeOfChange": "BatchCreatePartition"
        }
}
```

Untuk informasi selengkapnya, lihat Panduan Pengguna CloudWatch Acara Amazon. Untuk peristiwa khusus pada AWS Glue, lihat Peristiwa AWS Glue.

AWS Gluepemantauan sumber daya

AWS Gluememiliki batasan layanan untuk melindungi pelanggan dari penyediaan berlebihan yang tidak terduga dan dari tindakan jahat yang dimaksudkan untuk meningkatkan tagihan Anda. Batas juga melindungi layanan. Masuk ke konsol Kuota AWS Layanan, pelanggan dapat melihat batas sumber daya mereka saat ini dan meminta peningkatan (jika perlu).

AWS Gluememungkinkan Anda melihat penggunaan sumber daya layanan sebagai persentase di Amazon CloudWatch dan mengonfigurasi CloudWatch alarm di dalamnya untuk memantau penggunaan. Amazon CloudWatch menyediakan pemantauan untuk AWS sumber daya dan aplikasi pelanggan yang berjalan di infrastruktur Amazon. Metrik tidak dipungut biaya untuk Anda. Metrik berikut didukung:

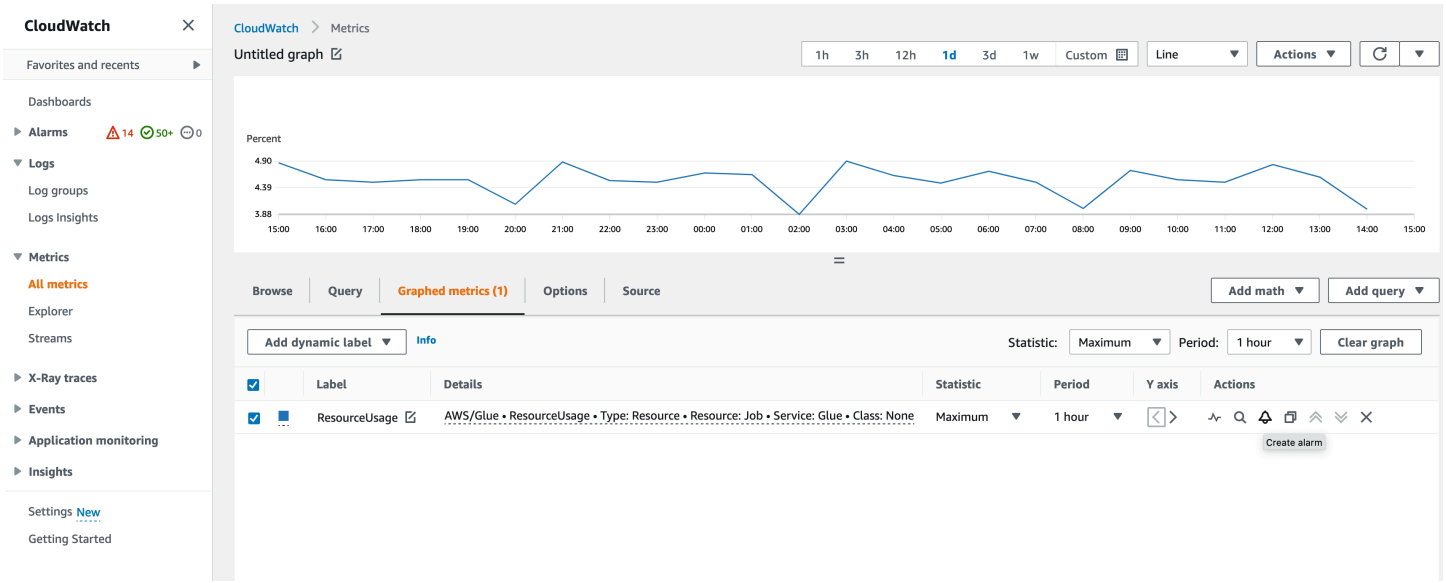
- Jumlah alur kerja per akun
- Jumlah pemicu per akun
- Jumlah pekerjaan per akun
- Jumlah pekerjaan bersamaan yang dijalankan per akun
- Jumlah cetak biru per akun
- Jumlah sesi interaktif per akun

Mengkonfigurasi dan menggunakan metrik sumber daya

Untuk menggunakan fitur ini, Anda dapat pergi ke CloudWatch konsol Amazon untuk melihat metrik dan mengonfigurasi alarm. Metrik berada di bawah namespace AWS /Glue dan merupakan persentase dari jumlah penggunaan sumber daya aktual dibagi dengan kuota sumber daya. CloudWatch Metrik dikirimkan ke akun Anda, yang tidak akan dikenakan biaya untuk Anda. Misalnya, jika Anda memiliki 10 alur kerja yang dibuat, dan kuota layanan Anda memungkinkan Anda memiliki 200 alur kerja maksimum, maka penggunaan Anda adalah 10/200 = 5%, dan dalam grafik, Anda akan melihat titik data 5 sebagai persentase. Untuk lebih spesifik:

```
Namespace: AWS/Glue
Metric name: ResourceUsage
Type: Resource
Resource: Workflow (or Trigger, Job, JobRun, Blueprint, InteractiveSession)
Service: Glue
```

```
Class: None
```



Untuk membuat alarm pada metrik di CloudWatch konsol:

1. Setelah Anda menemukan metrik, buka Metrik grafik.
2. Klik Buat alarm di bawah Tindakan.
3. Konfigurasikan alarm sesuai kebutuhan.

Kami memancarkan metrik setiap kali penggunaan sumber daya Anda mengalami perubahan (seperti kenaikan atau penurunan). Tetapi jika penggunaan sumber daya Anda tidak berubah, kami memancarkan metrik setiap jam, sehingga Anda memiliki grafik kontinu. CloudWatch Untuk menghindari kehilangan titik data, kami tidak menyarankan Anda untuk mengonfigurasi periode kurang dari 1 jam.

Anda juga dapat mengonfigurasi alarm menggunakan AWS CloudFormation seperti pada contoh berikut. Dalam contoh ini, setelah penggunaan sumber daya alur kerja mencapai 80%, itu memicu alarm untuk mengirim pesan ke topik SNS yang ada, di mana Anda dapat berlangganan untuk mendapatkan notifikasi.

```
{
 "Type": "AWS::CloudWatch::Alarm",
 "Properties": {
  "AlarmName": "WorkflowUsageAlarm",
  "ActionsEnabled": true,
  "OKActions": [],
```

```
  "AlarmActions": [
   "arn:aws:sns:af-south-1:085425700061:Default_CloudWatch_Alarms_Topic"
  ],
  "InsufficientDataActions": [],
  "MetricName": "ResourceUsage",
  "Namespace": "AWS/Glue",
  "Statistic": "Maximum",
  "Dimensions": [{
    "Name": "Type",
    "Value": "Resource"
   },
   {
    "Name": "Resource",
    "Value": "Workflow"
   },
   {
    "Name": "Service",
    "Value": "Glue"
   },
   {
    "Name": "Class",
    "Value": "None"
   }
  ],
  "Period": 3600,
  "EvaluationPeriods": 1,
  "DatapointsToAlarm": 1,
  "Threshold": 80,
  "ComparisonOperator": "GreaterThanThreshold",
  "TreatMissingData": "notBreaching"
 }
}
```

Mencatat panggilan API AWS Glue dengan AWS CloudTrail

AWS Glueterintegrasi denganAWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan diAWS Glue. CloudTrail menangkap semua panggilan API untuk AWS Glue sebagai peristiwa. Panggilan yang direkam mencakup panggilan dari AWS Glue konsol dan panggilan kode ke operasi API AWS Glue ini. Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara berkelanjutan ke bucket Amazon S3, termasuk acara untuk. AWS Glue Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru di CloudTrail konsol dalam Riwayat acara. Dengan menggunakan informasi yang

dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuatAWS Glue, alamat IP dari mana permintaan dibuat, siapa yang membuat permintaan, kapan dibuat, dan detail tambahan.

Untuk mempelajari selengkapnya CloudTrail, lihat Panduan AWS CloudTrail Pengguna.

AWS Glueinformasi di CloudTrail

CloudTrail diaktifkan di AWS akun Anda saat Anda membuat akun. Ketika aktivitas terjadi diAWS Glue, aktivitas tersebut dicatat dalam suatu CloudTrail peristiwa bersama dengan peristiwa AWS layanan lainnya dalam riwayat Acara. Anda dapat melihat, mencari, dan mengunduh peristiwa terbaru di akun AWS Anda. Untuk informasi selengkapnya, lihat Melihat Acara dengan Riwayat CloudTrail Acara.

Untuk catatan berkelanjutan tentang peristiwa di akun AWS Anda, termasuk peristiwa untuk AWS Glue, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket Amazon S3. Secara default, saat Anda membuat jejak di dalam konsol tersebut, jejak diterapkan ke semua Wilayah AWS. Jejak mencatat peristiwa dari semua Wilayah di partisi AWS dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat yang berikut:

- Membuat jejak untuk AWS akun Anda
- CloudTrail layanan dan integrasi yang didukung
- Mengonfigurasi Notifikasi Amazon SNS untuk CloudTrail
- Menerima File CloudTrail Log dari Beberapa Wilayah dan Menerima File CloudTrail Log dari Beberapa Akun

Semua AWS Glue tindakan dicatat oleh CloudTrail dan didokumentasikan dalamAWS Glue API. Misalnya, panggilan ke`CreateDatabase`, `CreateTable` dan `CreateScript` tindakan menghasilkan entri dalam file CloudTrail log.

Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut:

- Jika permintaan tersebut dibuat dengan kredensial pengguna root atau IAM.
- Jika permintaan tersebut dibuat dengan kredensial keamanan sementara untuk peran atau pengguna federasi.
- Bahwa permintaan dibuat oleh layanan AWS lain.

Untuk informasi lain, lihat Elemen userIdentity CloudTrail .

Namun, CloudTrail tidak mencatat semua informasi mengenai panggilan. Misalnya, CloudTrail tidak mencatat log dari informasi sensitif tertentu, seperti `ConnectionProperties` yang digunakan dalam permintaan koneksi, dan mencatat `null` alih-alih tanggapan yang dikembalikan oleh API berikut:

```
BatchGetPartition       GetCrawlers          GetJobs          GetTable
CreateScript            GetCrawlerMetrics    GetJobRun        GetTables
GetCatalogImportStatus  GetDatabase          GetJobRuns       GetTableVersions
GetClassifier           GetDatabases         GetMapping       GetTrigger
GetClassifiers          GetDataflowGraph     GetObjects       GetTriggers
GetConnection           GetDevEndpoint       GetPartition     GetUserDefinedFunction
GetConnections          GetDevEndpoints      GetPartitions    GetUserDefinedFunctions
GetCrawler              GetJob               GetPlan
```

Memahami entri file log AWS Glue

Trail adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket Amazon S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, jadi file tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `DeleteCrawler` tindakan.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
    "arn": "arn:aws:iam::123456789012:user/johndoe",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "johndoe"
  },
  "eventTime": "2017-10-11T22:29:49Z",
  "eventSource": "glue.amazonaws.com",
  "eventName": "DeleteCrawler",
  "awsRegion": "us-east-1",
```

```
  "sourceIPAddress": "72.21.198.64",
  "userAgent": "aws-cli/1.11.148 Python/3.6.1 Darwin/16.7.0 botocore/1.7.6",
  "requestParameters": {
    "name": "tes-alpha"
  },
  "responseElements": null,
  "requestID": "b16f4050-aed3-11e7-b0b3-75564a46954f",
  "eventID": "e73dd117-cfd1-47d1-9e2f-d1271cad838c",
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```

Contoh ini menunjukkan entri CloudTrail log yang menunjukkan `CreateConnection` tindakan.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
    "arn": "arn:aws:iam::123456789012:user/johndoe",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "johndoe"
  },
  "eventTime": "2017-10-13T00:19:19Z",
  "eventSource": "glue.amazonaws.com",
  "eventName": "CreateConnection",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "72.21.198.66",
  "userAgent": "aws-cli/1.11.148 Python/3.6.1 Darwin/16.7.0 botocore/1.7.6",
  "requestParameters": {
    "connectionInput": {
      "name": "test-connection-alpha",
      "connectionType": "JDBC",
      "physicalConnectionRequirements": {
        "subnetId": "subnet-323232",
        "availabilityZone": "us-east-1a",
        "securityGroupIdList": [
          "sg-12121212"
        ]
      }
    }
  },
```

```
  "responseElements": null,
  "requestID": "27136ebc-afac-11e7-a7d6-ab217e5c3f19",
  "eventID": "e8b3baeb-c511-4597-880f-c16210c60a4a",
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```

AWS Gluestatus job run

Anda dapat melihat status pekerjaan AWS Glue ekstrak, transformasi, dan beban (ETL) saat sedang berjalan atau setelah berhenti. Anda dapat melihat status menggunakan AWS Glue konsol, AWS Command Line Interface (AWS CLI), atau GetJobRuntindakan di AWS Glue API.

Status job run yang mungkin adalah`STARTING`,`RUNNING`,`STOPPING`,`STOPPED`,`SUCCEEDED`, `FAILEDERROR`, `WAITING` dan`TIMEOUT`.

Tabel berikut mencantumkan status yang menunjukkan penghentian tugas yang tidak normal.

Status eksekusi tugas	Deskripsi
`FAILED`	Tugas yang melebihi eksekusi bersamaan maksimum yang diizinkan, atau yang diakhiri dengan kode keluar yang tidak diketahui.
`ERROR`	Alur kerja, pemicu jadwal, atau pemicu peristiwa mencoba untuk menjalankan tugas yang dihapus.
`TIMEOUT`	Waktu aktif tugas melebihi nilai batas waktu yang ditentukan.

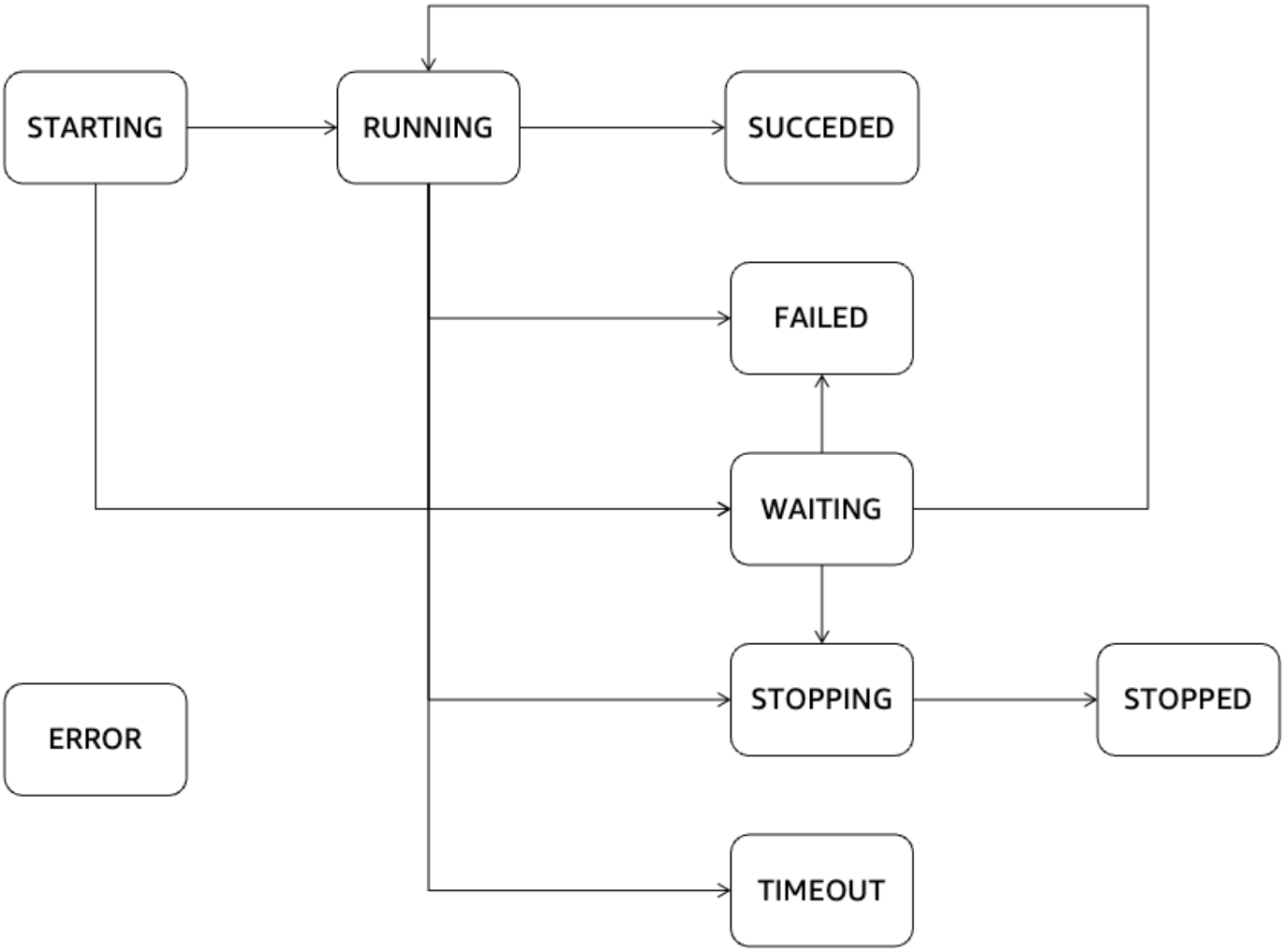
`WAITING`Status menunjukkan bahwa menjalankan pekerjaan sedang menunggu sumber daya. Tabel berikut menjelaskan perilaku menunggu untuk kelas pekerjaan yang berbeda.

Jenis Tugas	Perilaku
Lowongan kerja Spark (Standard)	Pekerjaan yang belum dikonfigurasi untuk mencoba lagi berdasarkan `maxRetries`

Jenis Tugas	Perilaku
	konfigurasi Anda dapat masuk ke status WAITING. Jalankan pekerjaan baru akan berada dalam status MENUNGGU jika layanan tidak dapat memperoleh sumber daya yang cukup untuk memulai proses. Hal ini dapat terjadi karena kuota layanan untuk akun Anda atau batas kapasitas di wilayah Anda menghadapi salah satu kasus kesalahan berikut: • Pekerjaan bersamaan maksimum per akun terlampaui • Pekerjaan bersamaan maksimum per pekerjaan terlampaui (termasuk kuota layanan tingkat akun serta batas yang Anda tentukan pada pekerjaan) `MaxConcurrentRuns` • Komputasi bersamaan maks (penggunaan DPU) terlampaui • Sumber daya tidak tersedia Untuk informasi selengkapnya tentang kuota layanan AWS Glue, lihat titik akhir dan kuota AWS Glue. Waktu AWS Glue akan menunggu sumber daya mungkin berbeda berdasarkan keadaan. Pekerjaan dapat bertransisi antara status non-terminal saat mencoba memperoleh sumber daya. Akhirnya, pekerjaan akan beralih ke `FAILED` jika tidak dapat memperoleh sumber daya. AWS Glue akan mencoba lagi selama maksimal 15 menit atau 10 upaya, mana yang lebih dulu.

Jenis Tugas	Perilaku
Lowongan kerja Spark (Flex)	Jalankan pekerjaan baru akan berada dalam status MENUNGGU jika layanan tidak dapat memperoleh sumber daya yang cukup untuk memulai proses, yang menunda dimulainy a proses. Jalankan akan dalam status MENUNGGU selama maksimal 20 menit (batas waktu dikendalikan oleh layanan). Setelah 15 menit, layanan akan mencoba melakukan start paksa dan tergantung pada kapasitas yang tersedia, proses dapat dimulai atau gagal dengan pesan kesalahan yang sesuai.
Pekerjaan Python shell	Perilaku yang sama dengan pekerjaan standar menggunakan Spark.

Diagram status berikut menguraikan transisi status yang diharapkan melalui siklus hidup pekerjaan Glue. AWS Informasi ini berlaku untuk semua jenis pekerjaan.


STARTING
RUNNING
SUCCEDED
FAILED
WAITING
STOPPING
STOPPED
ERROR
TIMEOUT

AWS Glue Streaming

AWS Glue Streaming, komponen dari AWS Glue, memungkinkan Anda menangani streaming data secara efisien dalam waktu dekat, memberdayakan Anda untuk melakukan tugas-tugas penting seperti konsumsi data, pemrosesan, dan pembelajaran mesin. Menggunakan kerangka Apache Spark Streaming, AWS Glue Streaming menyediakan layanan tanpa server yang dapat menangani streaming data dalam skala besar. AWS Glue menyediakan berbagai pengoptimalan di atas Apache Spark seperti infrastruktur tanpa server, auto-scaling, pengembangan pekerjaan visual, notebook instan untuk pekerjaan streaming dan peningkatan kinerja lainnya.

Kasus penggunaan untuk streaming

Beberapa kasus penggunaan umum untuk AWS Glue Streaming meliputi:

Pemrosesan ear-real-time data N: AWS Glue Streaming memungkinkan organisasi untuk memproses data streaming dalam waktu dekat, memungkinkan mereka memperoleh wawasan dan membuat keputusan tepat waktu berdasarkan informasi terbaru.

Deteksi penipuan: Anda dapat memanfaatkan AWS Glue Streaming untuk analisis real-time data streaming, membuatnya berharga untuk mendeteksi aktivitas penipuan, seperti penipuan kartu kredit, intrusi jaringan, atau penipuan online. Dengan terus memproses dan menganalisis data yang masuk, Anda dapat dengan cepat mengidentifikasi pola atau anomali yang mencurigakan.

Analisis media sosial: AWS Glue Streaming dapat memproses data media sosial real-time, seperti tweet, posting, atau komentar, memungkinkan organisasi untuk memantau tren, analisis sentimen, dan mengelola reputasi merek secara real-time.

Analisis Internet of Things (IoT): AWS Glue Streaming cocok untuk menangani dan menganalisis aliran data berkecepatan tinggi yang dihasilkan oleh perangkat IoT, sensor, dan mesin yang terhubung. Ini memungkinkan pemantauan real-time, deteksi anomali, pemeliharaan prediktif, dan kasus penggunaan analitik IoT lainnya.

Analisis Clickstream: AWS Glue Streaming dapat memproses dan menganalisis data clickstream real-time dari situs web atau aplikasi seluler. Hal ini memungkinkan bisnis untuk mendapatkan wawasan tentang perilaku pengguna, mempersonalisasi pengalaman pengguna, dan mengoptimalkan kampanye pemasaran berdasarkan data clickstream real-time.

Pemantauan dan analisis log: AWS Glue Streaming dapat terus memproses dan menganalisis data log dari server, aplikasi, atau perangkat jaringan secara real-time. Ini membantu dalam mendeteksi anomali, memecahkan masalah, dan memantau kesehatan dan kinerja sistem.

Sistem rekomendasi: AWS Glue Streaming dapat memproses data aktivitas pengguna secara real-time dan memperbarui model rekomendasi secara dinamis. Ini memungkinkan rekomendasi yang dipersonalisasi dan real-time berdasarkan perilaku dan preferensi pengguna.

Ini adalah beberapa contoh dari beragam kasus penggunaan di mana AWS Glue Streaming dapat diterapkan. Integrasinya dengan AWS ekosistem dan layanan terkelola menjadikannya pilihan yang nyaman untuk pemrosesan streaming real-time dan analitik di cloud.

Apa manfaat menggunakan AWS Glue Streaming?

Manfaat menggunakan AWS Glue Streaming adalah sebagai berikut:

- Tanpa Server: AWS Glue Streaming tanpa server, menghilangkan kebutuhan untuk mengelola infrastruktur. Ini mengurangi overhead operasional dan memungkinkan pengguna untuk fokus pada pemrosesan data dan tugas analitik daripada manajemen infrastruktur.
- Autoscaling: AWS Glue Streaming menyediakan kemampuan penskalaan otomatis, secara dinamis menyesuaikan kapasitas pemrosesan berdasarkan beban kerja. Secara otomatis skala keluar atau masuk untuk menangani fluktuasi volume data, memastikan kinerja optimal dan pemanfaatan sumber daya.
- Pengembangan visual: Streaming pengembangan pekerjaan bisa menjadi kompleks. AWS Glue Streaming mengatasi tantangan ini dengan menawarkan AWS Glue Studio, alat penulisan visual. AWS Glue Studio menyederhanakan proses pembuatan alur kerja streaming dan memungkinkan pengembang untuk merancang dan mengelola aplikasi streaming secara visual, mengurangi kurva pembelajaran dan meningkatkan produktivitas.
- Hemat biaya: Sebagai layanan tanpa server, AWS Glue Streaming menawarkan efisiensi biaya dengan menghilangkan kebutuhan untuk penyediaan dan pemeliharaan infrastruktur. Pengguna ditagih berdasarkan sumber daya yang dikonsumsi selama pelaksanaan pekerjaan streaming, memungkinkan pengoptimalan biaya dan penskalaan berdasarkan penggunaan aktual.
- Menangani beban kerja yang kompleks: AWS Glue Streaming dirancang untuk menangani beban kerja streaming yang kompleks. Ini dapat memproses dan menganalisis volume besar data real-time, mendukung transformasi lanjutan, dan berintegrasi dengan AWS layanan lain, memungkinkan jalur data streaming yang canggih dan alur kerja analitik.

- Tanpa penguncian: AWS Glue Streaming memberikan fleksibilitas dan menghindari penguncian vendor. Pengguna dapat memanfaatkan AWS Glue Streaming sebagai bagian dari AWS ekosistem yang lebih luas, mengintegrasikannya dengan AWS layanan lain dengan mulus. Hal ini memungkinkan integrasi yang mudah dengan sumber data, aplikasi, dan layanan yang ada tanpa terikat pada teknologi atau platform tertentu.

Kapan menggunakan AWS Glue Streaming?

Ada banyak opsi dalam hal kasus penggunaan streaming. Kami merekomendasikan AWS Glue streaming dalam skenario berikut.

1. Jika Anda sudah menggunakan AWS Glue atau Spark untuk pemrosesan batch, AWS Glue Streaming adalah pilihan ideal untuk Anda. Ini memberikan transisi yang mulus untuk membangun pekerjaan streaming tanpa perlu mempelajari bahasa atau kerangka kerja baru. Memanfaatkan pengetahuan dan infrastruktur yang ada, AWS Glue Streaming menyederhanakan proses pengembangan pekerjaan dan memungkinkan Anda untuk dengan mudah memperluas kemampuan pemrosesan data Anda ke skenario streaming waktu nyata.
2. Jika Anda memerlukan layanan atau produk terpadu untuk menangani beban kerja batch, streaming, dan berbasis peristiwa, AWS Glue Streaming adalah solusi untuk Anda. Dengan AWS Glue Streaming, Anda dapat mengkonsolidasikan kebutuhan pemrosesan data Anda ke dalam satu kerangka kerja, menghilangkan kompleksitas pengelolaan beberapa sistem. Hal ini memungkinkan pengembangan dan pemeliharaan yang efisien dari beragam alur kerja data sekaligus memastikan konsistensi dan kompatibilitas di berbagai jenis beban kerja.
3. AWS Glue Streaming sangat cocok untuk skenario yang melibatkan volume data streaming yang sangat besar dan transformasi kompleks, seperti bergabung antara aliran atau database relasional. Ini dapat secara efisien memproses dan menganalisis aliran data yang sangat besar, memungkinkan Anda mengatasi beban kerja yang menuntut dengan mudah. Baik itu konsumsi data kecepatan tinggi atau manipulasi data yang rumit, skalabilitas AWS Glue Streaming dan kemampuan pemrosesan lanjutan memastikan kinerja optimal dan hasil yang akurat.
4. Jika Anda lebih suka pendekatan visual untuk membangun pekerjaan streaming, AWS Glue menawarkan AWS Glue Studio, yang dengannya Anda dapat merancang dan mengelola aplikasi streaming Anda secara visual, menyederhanakan proses pengembangan. Antarmuka intuitif ini memungkinkan pengembang untuk membuat, mengonfigurasi, dan memantau alur kerja streaming menggunakan antarmuka visual, mengurangi kurva pembelajaran, dan meningkatkan produktivitas.

5. AWS Glue Streaming adalah pilihan yang sangat baik untuk kasus near-real-time penggunaan di mana ada SLA yang ketat (Perjanjian Tingkat Layanan) lebih dari 10 detik.
6. Jika Anda membangun danau data transaksional menggunakan Apache Iceberg, Apache Hudi, atau Delta Lake, AWS Glue Streaming menyediakan dukungan asli untuk format tabel terbuka ini. Integrasi tanpa batas ini memungkinkan Anda memproses data streaming langsung dari danau data transaksional ini, memastikan konsistensi, integritas, dan kompatibilitas data.
7. Saat perlu menelan data streaming untuk berbagai target data: AWS Glue Streaming menyediakan target asli ke berbagai target data seperti Amazon Redshift, Amazon RDS, Amazon Aurora, Oracle, SQL Server, dan target lainnya.

Sumber data yang didukung

AWS Glue Streaming mendukung sumber data berikut:

- Amazon Kinesis
- Amazon MSK (Managed Streaming untuk Apache Kafka)
- Apache Kafka yang dikelola sendiri

Target data yang didukung

AWS Glue Streaming mendukung berbagai target data seperti:

- Target data yang didukung oleh Katalog AWS Glue Data
- Amazon S3
- Amazon Redshift
- MySQL
- PostgreSQL
- Oracle
- Microsoft SQL Server
- Kepingan salju
- Database apa pun yang dapat dihubungkan menggunakan JDBC
- Apache Iceberg, Delta dan Apache Hudi
- AWS Glue Konektor Marketplace

Tutorial: Bangun beban kerja streaming pertama Anda menggunakan Studio AWS Glue

Dalam tutorial ini, Anda akan belajar cara membuat pekerjaan streaming menggunakan AWS Glue Studio. AWS Glue Studio adalah antarmuka visual untuk menciptakan AWS Glue pekerjaan.

Anda dapat membuat pekerjaan ekstrak, transformasi, dan pemuatan streaming (ETL) yang berjalan terus menerus dan menggunakan data dari sumber streaming di Amazon Kinesis Data Streams, Apache Kafka, dan Amazon Managed Streaming untuk Apache Kafka (Amazon MSK).

Prasyarat

Untuk mengikuti tutorial ini, Anda memerlukan pengguna dengan izin AWS konsol untuk digunakanAWS Glue, Amazon Kinesis, Amazon S3, Amazon Athena,, AWS CloudFormation LambdaAWS, dan Amazon Cognito.

Konsumsi data streaming dari Amazon Kinesis

Topik

- Menghasilkan data tiruan dengan Kinesis Data Generator
- Membuat pekerjaan AWS Glue streaming dengan AWS Glue Studio
- Melakukan transformasi dan menyimpan hasil yang ditransformasikan di Amazon S3

Menghasilkan data tiruan dengan Kinesis Data Generator

Anda dapat secara sintetis menghasilkan data sampel dalam format JSON menggunakan Kinesis Data Generator (KDG). Anda dapat menemukan instruksi dan detail lengkap dalam dokumentasi alat.

1. Untuk memulai, klik

   untuk menjalankan AWS CloudFormation template di AWS lingkungan Anda.

   > **Note**
   >
   > Anda mungkin mengalami kegagalan CloudFormation template karena beberapa sumber daya, seperti pengguna Amazon Cognito untuk Kinesis Data Generator sudah ada di akun Anda. AWS Ini bisa jadi karena Anda sudah mengaturnya dari tutorial atau blog lain. Untuk

mengatasi hal ini, Anda dapat mencoba template di AWS akun baru untuk awal yang baru, atau menjelajahi AWS Wilayah yang berbeda. Opsi ini memungkinkan Anda menjalankan tutorial tanpa bertentangan dengan sumber daya yang ada.

Template menyediakan aliran data Kinesis dan akun Kinesis Data Generator untuk Anda. Ini juga membuat bucket Amazon S3 untuk menyimpan data dan Peran Layanan Glue dengan izin yang diperlukan untuk tutorial ini.

2. Masukkan Nama Pengguna dan Kata Sandi yang akan digunakan KDG untuk mengautentikasi. Perhatikan nama pengguna dan kata sandi untuk penggunaan lebih lanjut.
3. Pilih Berikutnya sampai ke langkah terakhir. Mengakui penciptaan sumber daya IAM. Periksa kesalahan apa pun di bagian atas layar, seperti kata sandi yang tidak memenuhi persyaratan minimum, dan gunakan templat.
4. Arahkan ke tab Output dari tumpukan. Setelah template digunakan, itu akan menampilkan properti KinesisDataGeneratorUrlyang dihasilkan. Klik URL tersebut.
5. Masukkan Nama Pengguna dan Kata Sandi yang Anda catat.
6. Pilih Wilayah yang Anda gunakan dan pilih Kinesis Stream `GlueStreamTest-{AWS::AccountId}`
7. Masukkan template berikut:

```
{
    "ventilatorid": {{random.number(100)}},
    "eventtime": "{{date.now("YYYY-MM-DD HH:mm:ss")}}",
    "serialnumber": "{{random.uuid}}",
    "pressurecontrol": {{random.number(
        {
            "min":5,
            "max":30
        }
    )}},
    "o2stats": {{random.number(
        {
            "min":92,
            "max":98
        }
    )}},
    "minutevolume": {{random.number(
        {
```

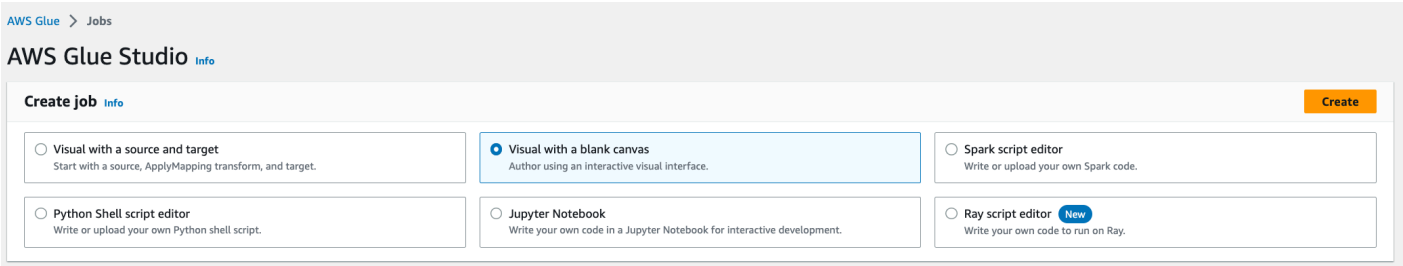
```
            "min":5,
            "max":8
        }
    )}},
    "manufacturer": "{{random.arrayElement(
        ["3M", "GE","Vyaire", "Getinge"]
    )}}"
}
```

Anda sekarang dapat melihat data tiruan dengan template Uji dan menelan data tiruan ke Kinesis dengan data Kirim.

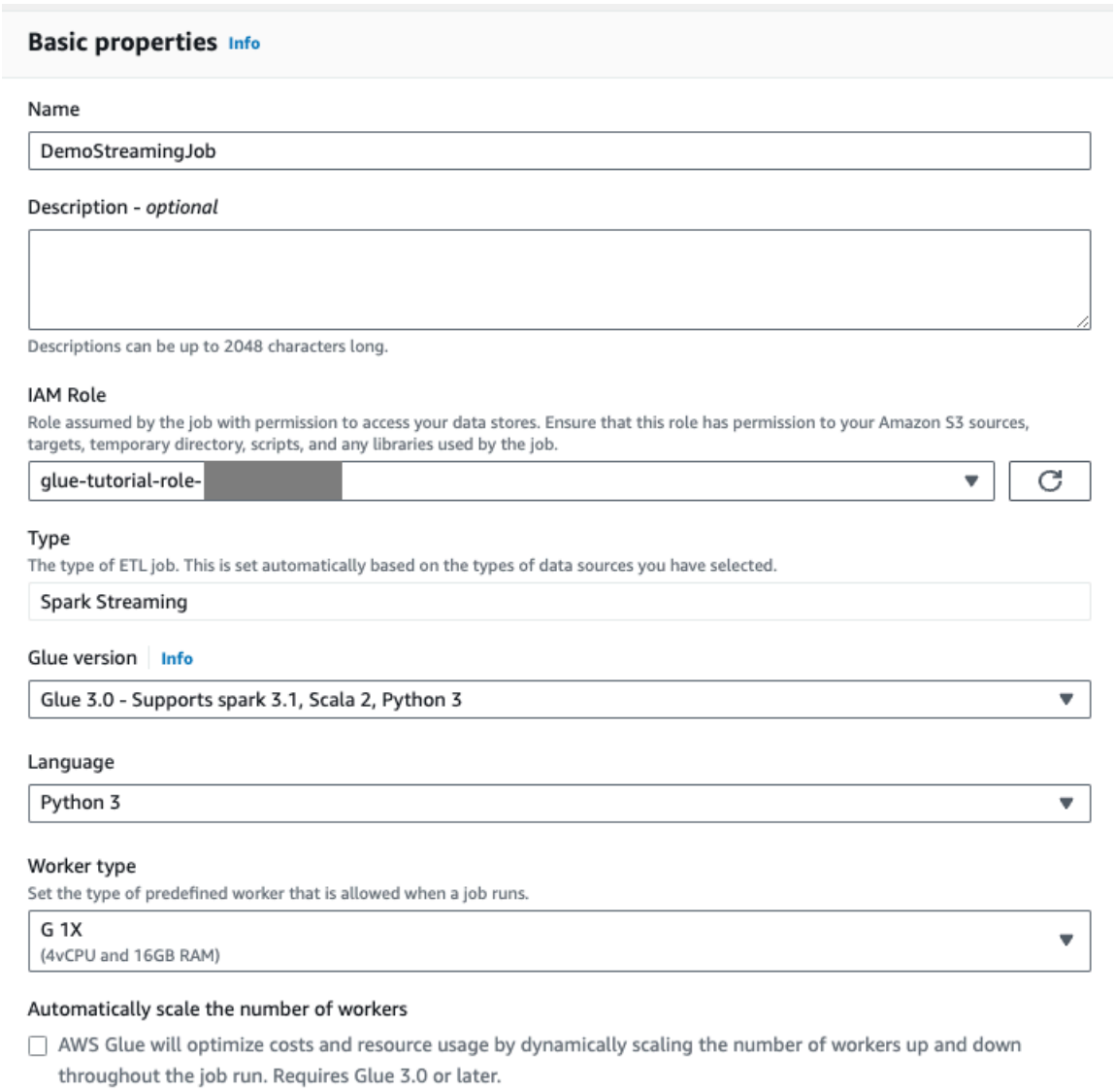
8. Klik Kirim data dan hasilkan 5-10K catatan ke Kinesis.

Membuat pekerjaan AWS Glue streaming dengan AWS Glue Studio

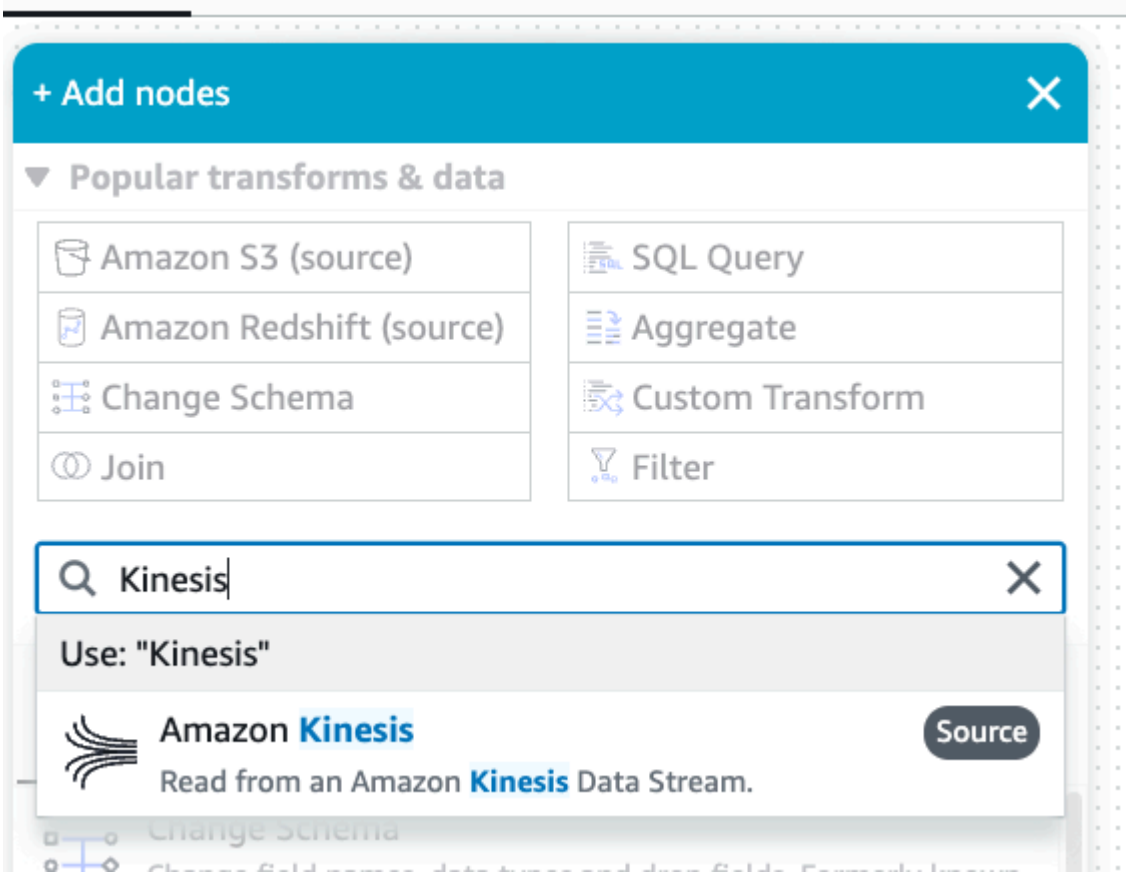
1. Arahkan ke AWS Glue konsol di Wilayah yang sama.
2. Pilih pekerjaan ETL di bawah bilah navigasi sisi kiri di bawah Integrasi Data dan ETL.
3. Buat AWS Glue Job via Visual dengan kanvas kosong.



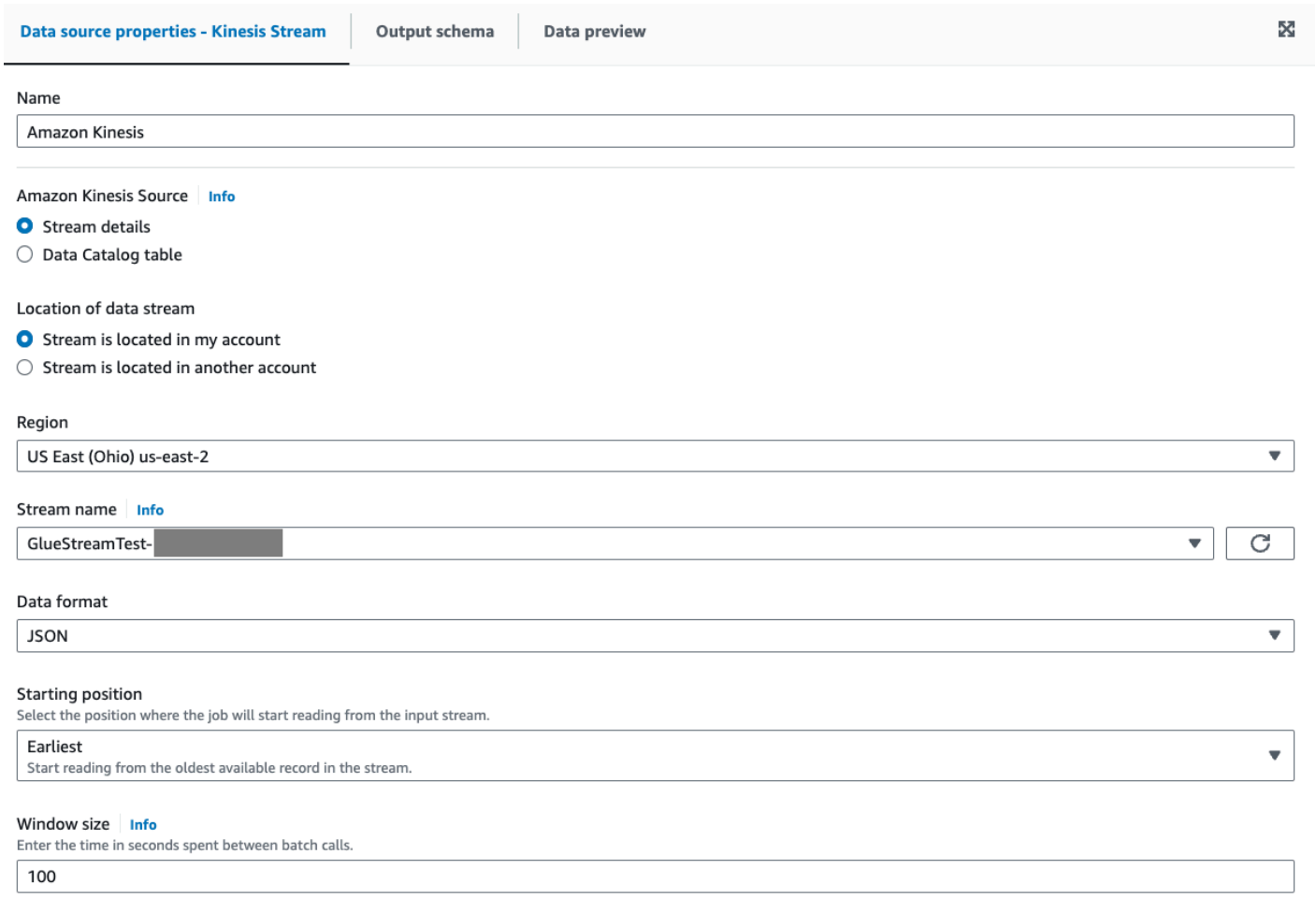
4. Arahkan ke tab Job Details.
5. Untuk nama AWS Glue pekerjaan, masukkan`DemoStreamingJob`.
6. Untuk Peran IAM, pilih peran yang disediakan oleh templat, CloudFormation . `glue-tutorial-role-${AWS::AccountId}`
7. Untuk versi Glue, pilih Glue 3.0. Biarkan semua opsi lain sebagai default.



8. Arahkan ke tab Visual.
9. Klik pada ikon plus. Masukkan Kinesis di bilah pencarian. Pilih sumber data Amazon Kinesis.



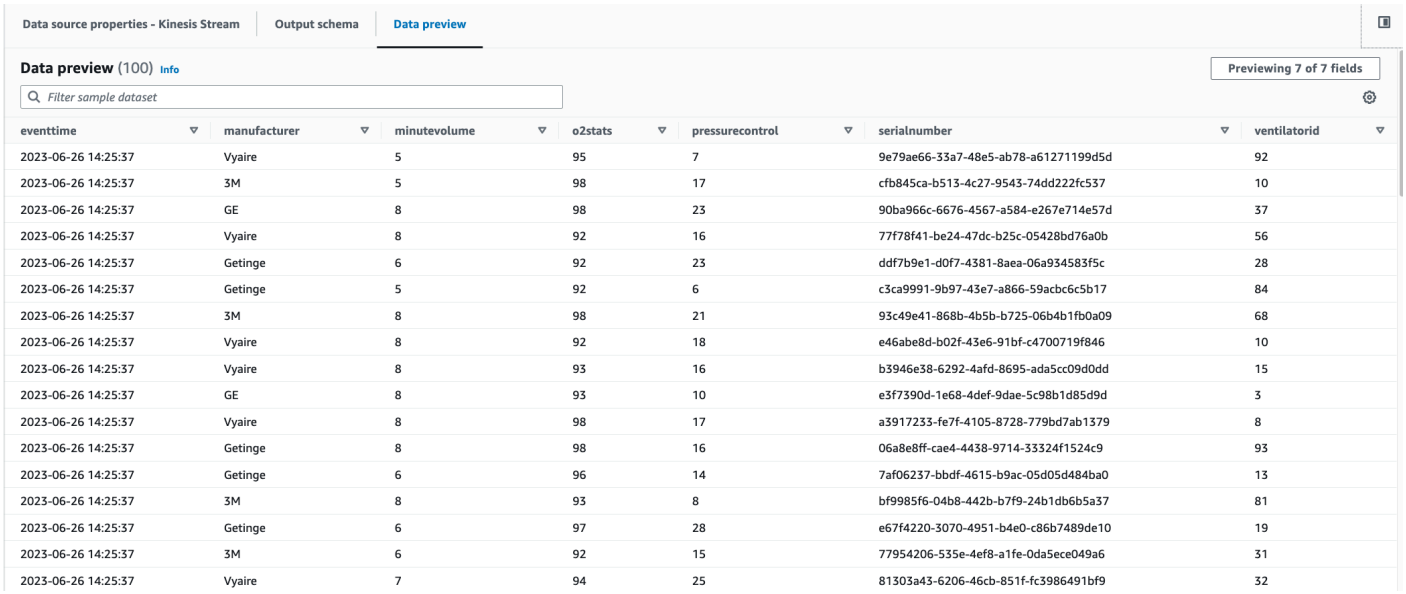
10. Pilih Streaming detail untuk Sumber Amazon Kinesis di bawah tab Properti sumber data - Aliran Kinesis.
11. Pilih Stream terletak di akun saya untuk Lokasi aliran data.
12. Pilih Wilayah yang Anda gunakan.
13. Pilih `GlueStreamTest-{AWS::AccountId}` aliran.
14. Simpan semua pengaturan lainnya sebagai default.



15. Arahkan ke tab Pratinjau data.
16. Klik Mulai sesi pratinjau data, yang menampilkan pratinjau data tiruan yang dihasilkan oleh KDG. Pilih Peran Layanan Glue yang sebelumnya Anda buat untuk pekerjaan AWS Glue Streaming.

    Dibutuhkan 30-60 detik agar data pratinjau muncul. Jika ditampilkan Tidak ada data untuk ditampilkan, klik ikon roda gigi dan ubah Jumlah baris yang akan dijadikan sampel`100`.

    Anda dapat melihat data sampel seperti di bawah ini:



Data source properties - Kinesis Stream | Output schema | Data preview

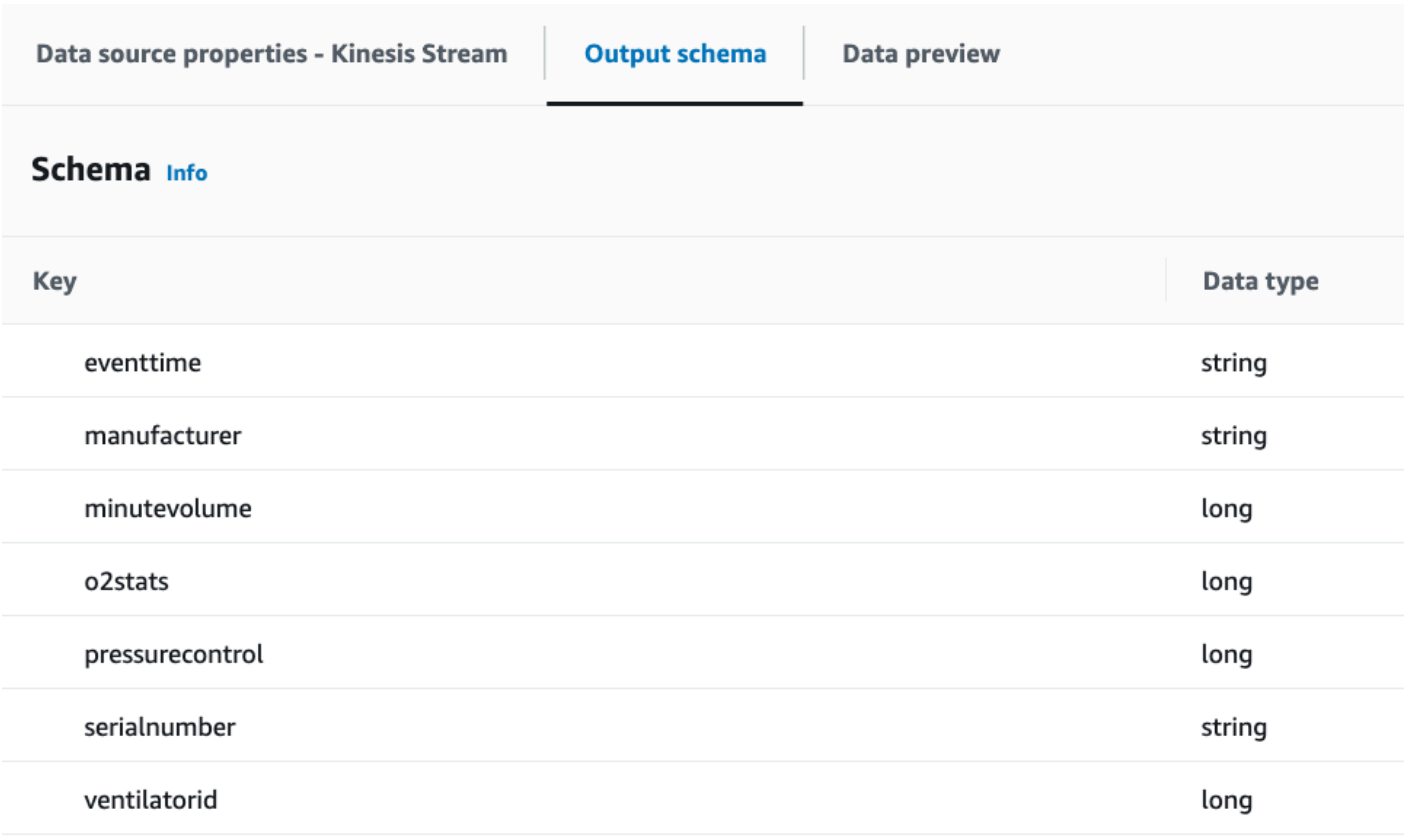
Data preview (100) Info

Previewing 7 of 7 fields

Filter sample dataset

eventtime	manufacturer	minutevolume	o2stats	pressurecontrol	serialnumber	ventilatorid
2023-06-26 14:25:37	Vyaire	5	95	7	9e79ae66-33a7-48e5-ab78-a61271199d5d	92
2023-06-26 14:25:37	3M	5	98	17	cfb845ca-b513-4c27-9543-74dd222fc537	10
2023-06-26 14:25:37	GE	8	98	23	90ba966c-6676-4567-a584-e267e714e57d	37
2023-06-26 14:25:37	Vyaire	8	92	16	77f78f41-be24-47dc-b25c-05428bd76a0b	56
2023-06-26 14:25:37	Getinge	6	92	23	ddf7b9e1-d0f7-4381-8aea-06a934583f5c	28
2023-06-26 14:25:37	Getinge	5	92	6	c3ca9991-9b97-43e7-a866-59acbc6c5b17	84
2023-06-26 14:25:37	3M	8	98	21	93c49e41-868b-4b5b-b725-06b4b1fb0a09	68
2023-06-26 14:25:37	Vyaire	8	92	18	e46abe8d-b02f-43e6-91bf-c4700719f846	10
2023-06-26 14:25:37	Vyaire	8	93	16	b3946e38-6292-4afd-8695-ada5cc09d0dd	15
2023-06-26 14:25:37	GE	8	93	10	e3f7390d-1e68-4def-9dae-5c98b1d85d9d	3
2023-06-26 14:25:37	Vyaire	8	98	17	a3917233-fe7f-4105-8728-779bd7ab1379	8
2023-06-26 14:25:37	Getinge	8	98	16	06a8e8ff-cae4-4438-9714-33324f1524c9	93
2023-06-26 14:25:37	Getinge	6	96	14	7af06237-bbdf-4615-b9ac-05d05d484ba0	13
2023-06-26 14:25:37	3M	8	93	8	bf9985f6-04b8-442b-b7f9-24b1db6b5a37	81
2023-06-26 14:25:37	Getinge	6	97	28	e67f4220-3070-4951-b4e0-c86b7489de10	19
2023-06-26 14:25:37	3M	6	92	15	77954206-535e-4ef8-a1fe-0da5ece049a6	31
2023-06-26 14:25:37	Vyaire	7	94	25	81303a43-6206-46cb-851f-fc3986491bf9	32

Anda juga dapat melihat skema yang disimpulkan di tab skema Output.



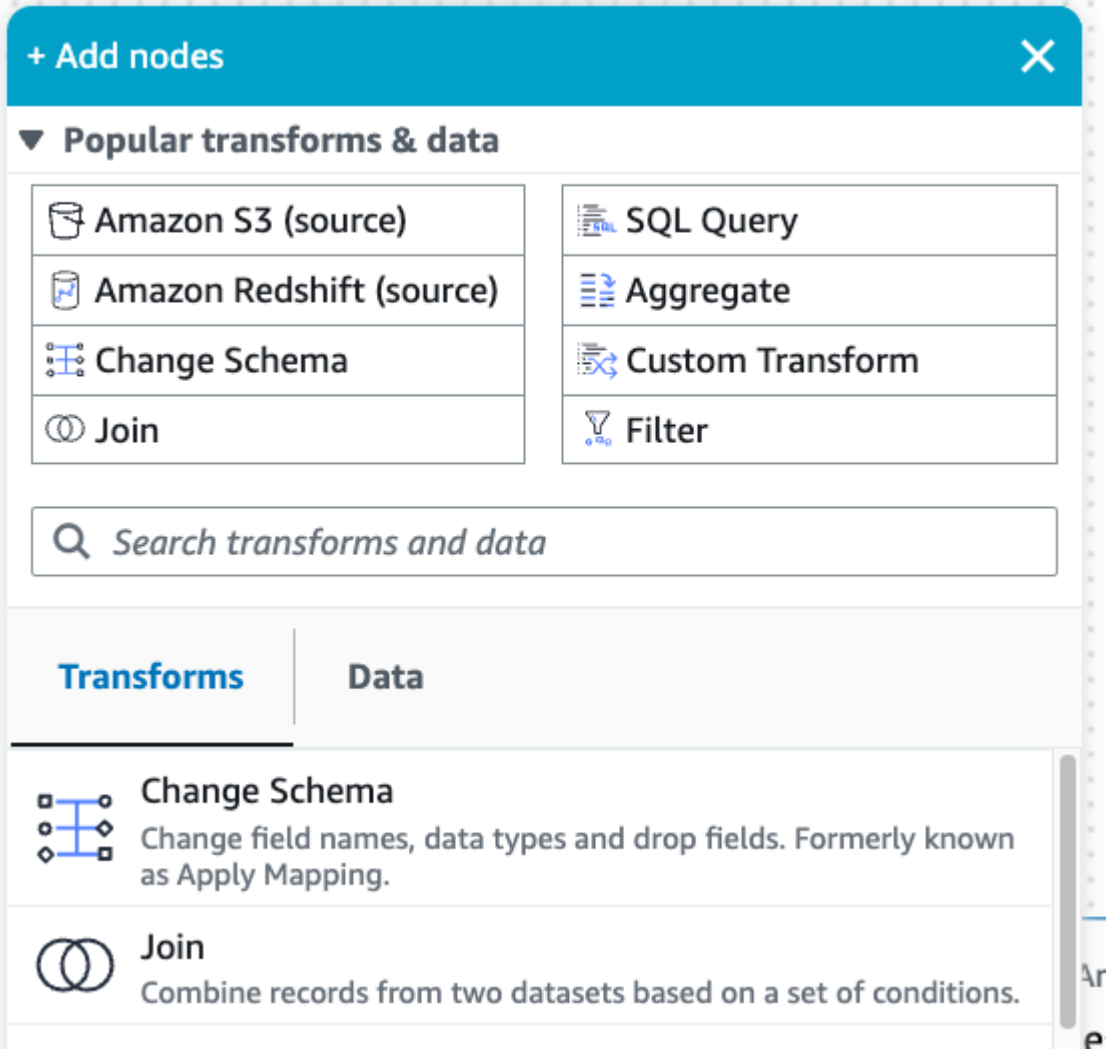
Data source properties - Kinesis Stream | Output schema | Data preview

Schema Info

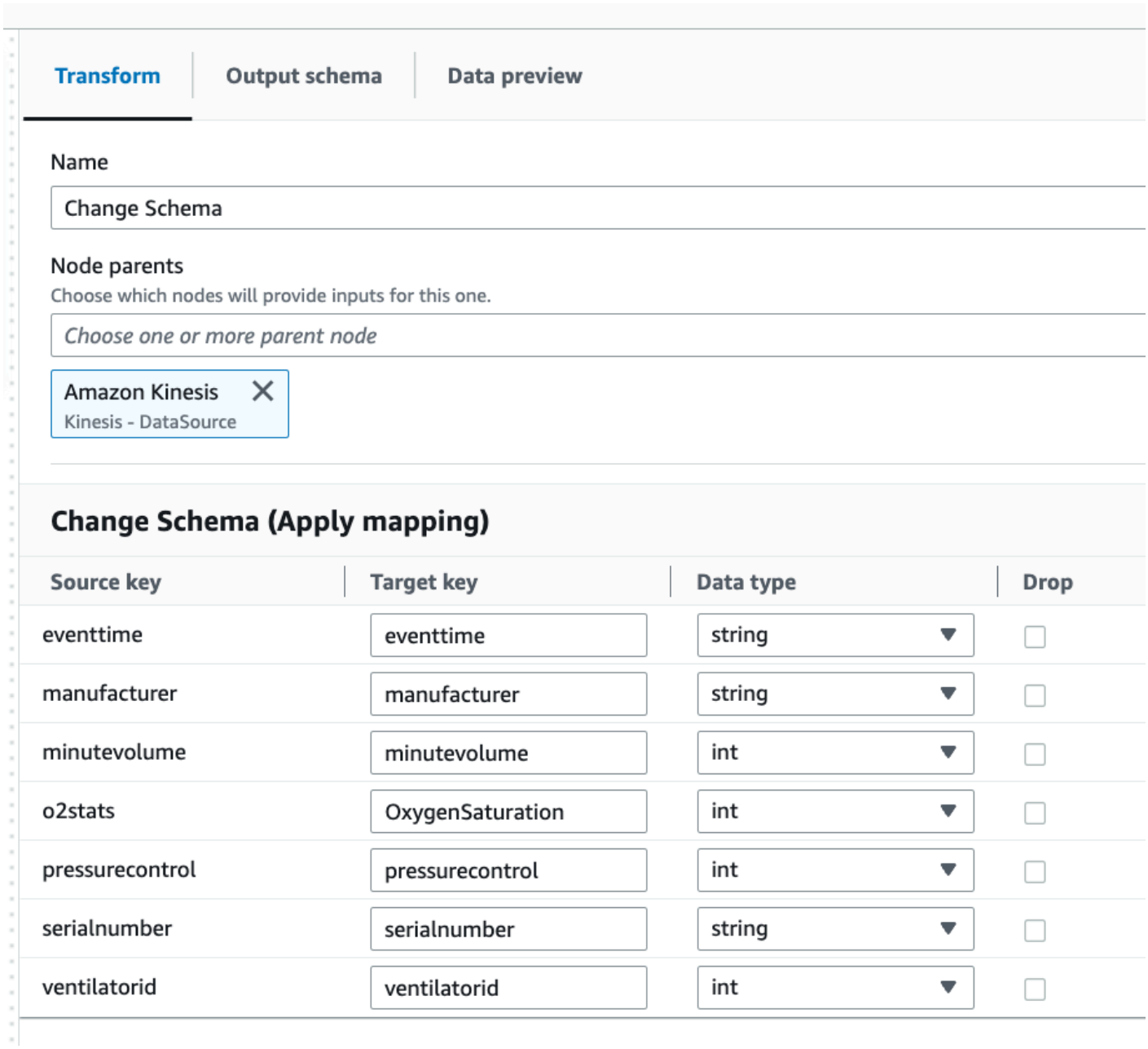
Key	Data type
eventtime	string
manufacturer	string
minutevolume	long
o2stats	long
pressurecontrol	long
serialnumber	string
ventilatorid	long

Melakukan transformasi dan menyimpan hasil yang ditransformasikan di Amazon S3

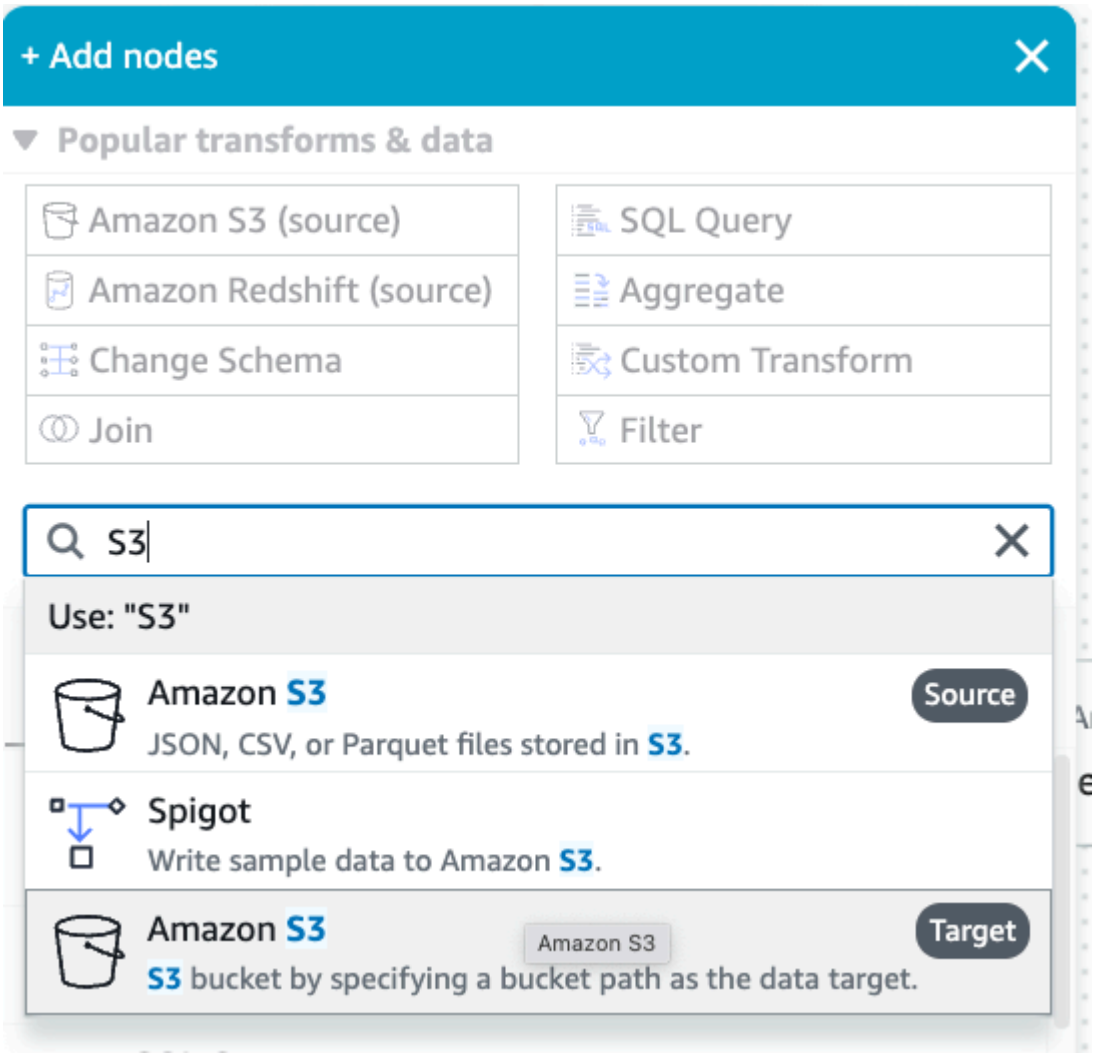
1. Dengan node sumber yang dipilih, klik ikon plus di kiri atas untuk menambahkan langkah Transforms.
2. Pilih langkah Ubah Skema.



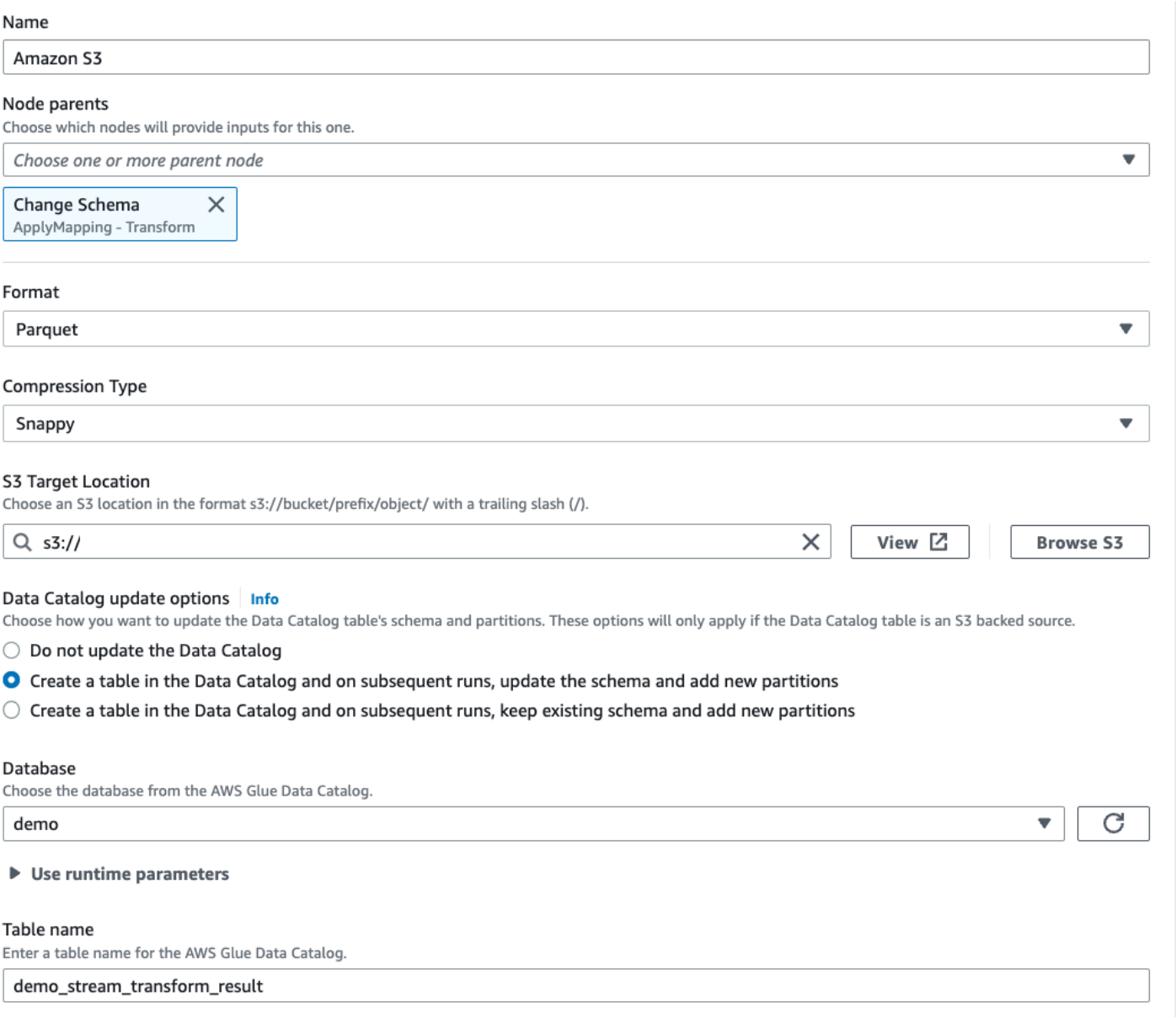
3. Anda dapat mengganti nama bidang dan mengonversi tipe data bidang dalam langkah ini. Ubah nama `o2stats` kolom menjadi `OxygenSaturation` dan ubah semua tipe `long` data menjadi`int`.



4. Klik ikon plus untuk menambahkan target Amazon S3. Masukkan S3 di kotak pencarian dan pilih Amazon S3 - Langkah transformasi target.

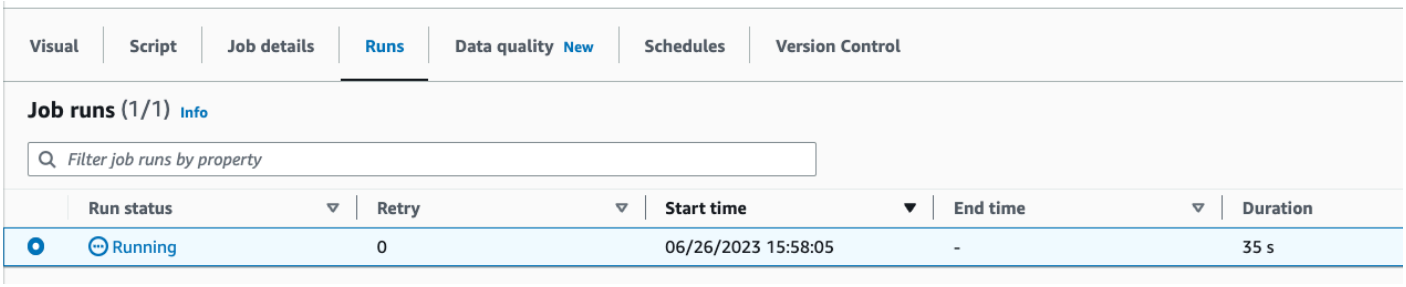


5. Pilih Parket sebagai format file target.
6. Pilih Snappy sebagai tipe kompresi.
7. Masukkan Lokasi Target S3 yang dibuat oleh CloudFormation template,`streaming-tutorial-s3-target-{AWS::AccountId}`.
8. Pilih untuk Membuat tabel di Katalog Data dan pada proses berikutnya, perbarui skema dan tambahkan partisi baru.
9. Masukkan Database target dan nama Tabel untuk menyimpan skema tabel target Amazon S3.

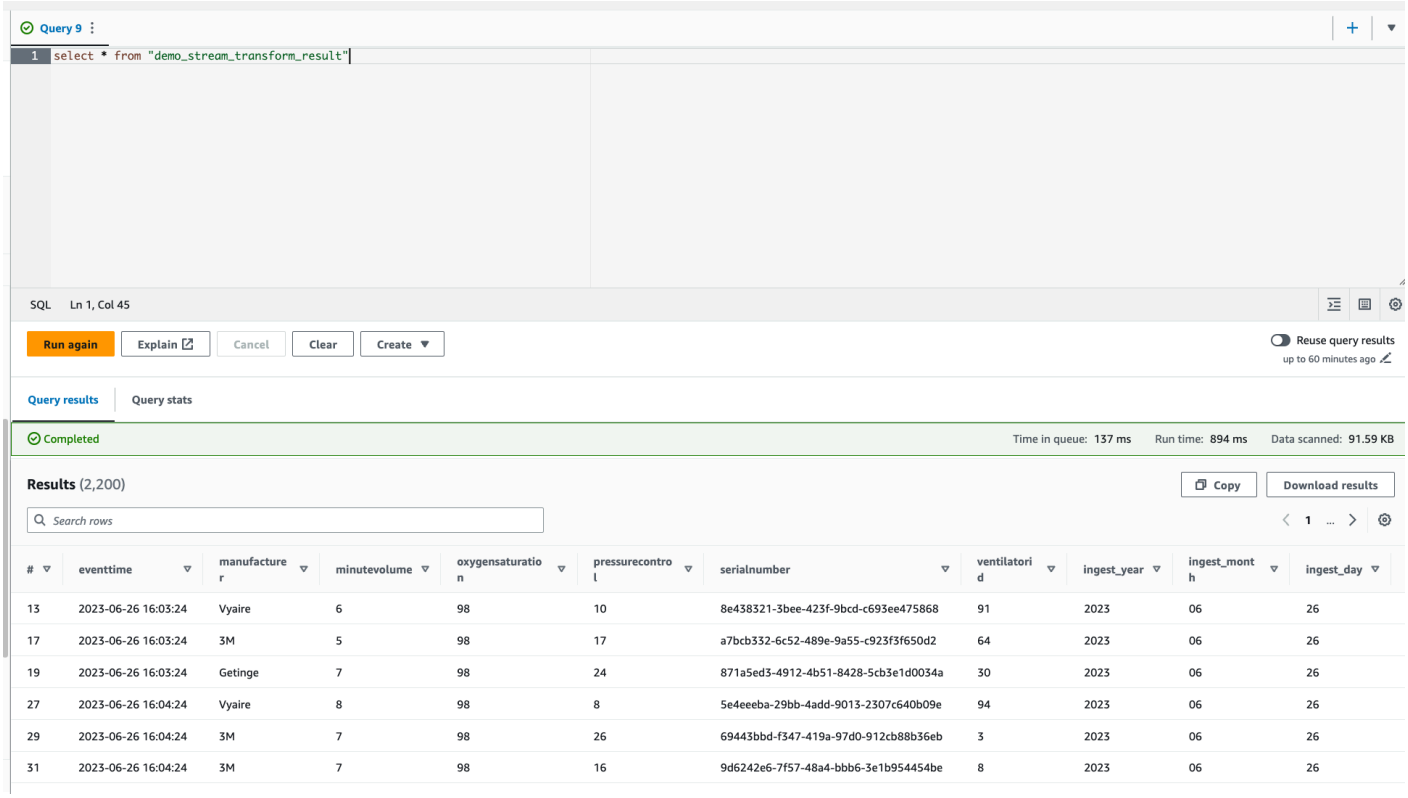


10. Klik pada tab Script untuk melihat kode yang dihasilkan.
11. Klik Simpan di kanan atas untuk menyimpan kode ETL dan kemudian klik Jalankan untuk memulai pekerjaan streaming. AWS Glue

    Anda dapat menemukan status Run di tab Runs. Biarkan pekerjaan berjalan selama 3-5 menit dan kemudian hentikan pekerjaan.



12.Verifikasi tabel baru yang dibuat di Amazon Athena.



Tutorial: Bangun beban kerja streaming pertama Anda menggunakan notebook AWS Glue Studio

Dalam tutorial ini, Anda akan mengeksplorasi cara memanfaatkan notebook AWS Glue Studio untuk secara interaktif membangun dan menyempurnakan pekerjaan ETL Anda untuk pemrosesan data mendekati waktu nyata. Apakah Anda baru AWS Glue atau ingin meningkatkan keahlian Anda, panduan ini akan memandu Anda melalui proses, memberdayakan Anda untuk memanfaatkan potensi penuh notebook sesi AWS Glue interaktif.

Dengan AWS Glue Streaming, Anda dapat membuat pekerjaan ekstrak, transformasi, dan pemuatan streaming (ETL) yang berjalan terus menerus dan mengkonsumsi data dari sumber streaming seperti Amazon Kinesis Data Streams, Apache Kafka, dan Amazon Managed Streaming untuk Apache Kafka (Amazon MSK).

Prasyarat

Untuk mengikuti tutorial ini, Anda memerlukan pengguna dengan izin AWS konsol untuk digunakanAWS Glue, Amazon Kinesis, Amazon S3, Amazon Athena,, AWS CloudFormation LambdaAWS, dan Amazon Cognito.

Konsumsi data streaming dari Amazon Kinesis

Topik

Menghasilkan data tiruan dengan Kinesis Data Generator

> **Note**
>
> Jika Anda telah menyelesaikan kami sebelumnyaTutorial: Bangun beban kerja streaming pertama Anda menggunakan Studio AWS Glue, Anda sudah menginstal Kinesis Data Generator di akun Anda dan Anda dapat melewati langkah 1-8 di bawah ini dan melanjutkan ke bagian. Membuat pekerjaan AWS Glue streaming dengan AWS Glue Studio

Anda dapat secara sintetis menghasilkan data sampel dalam format JSON menggunakan Kinesis Data Generator (KDG). Anda dapat menemukan instruksi dan detail lengkap dalam dokumentasi alat.

1. Untuk memulai, klik
   
   untuk menjalankan AWS CloudFormation template di AWS lingkungan Anda.

> **Note**
>
> Anda mungkin mengalami kegagalan CloudFormation template karena beberapa sumber daya, seperti pengguna Amazon Cognito untuk Kinesis Data Generator sudah ada di akun Anda. AWS Ini bisa jadi karena Anda sudah mengaturnya dari tutorial atau blog lain. Untuk mengatasi hal ini, Anda dapat mencoba template di AWS akun baru untuk awal yang baru, atau menjelajahi AWS Wilayah yang berbeda. Opsi ini memungkinkan Anda menjalankan tutorial tanpa bertentangan dengan sumber daya yang ada.

Template menyediakan aliran data Kinesis dan akun Kinesis Data Generator untuk Anda.

2. Masukkan Nama Pengguna dan Kata Sandi yang akan digunakan KDG untuk mengautentikasi. Perhatikan nama pengguna dan kata sandi untuk penggunaan lebih lanjut.
3. Pilih Berikutnya sampai ke langkah terakhir. Mengakui penciptaan sumber daya IAM. Periksa kesalahan apa pun di bagian atas layar, seperti kata sandi yang tidak memenuhi persyaratan minimum, dan gunakan templat.
4. Arahkan ke tab Output dari tumpukan. Setelah template digunakan, itu akan menampilkan properti KinesisDataGeneratorUrlyang dihasilkan. Klik URL tersebut.
5. Masukkan Nama Pengguna dan Kata Sandi yang Anda catat.
6. Pilih Wilayah yang Anda gunakan dan pilih Kinesis Stream `GlueStreamTest-{AWS::AccountId}`
7. Masukkan template berikut:

```
{
    "ventilatorid": {{random.number(100)}},
    "eventtime": "{{date.now("YYYY-MM-DD HH:mm:ss")}}",
    "serialnumber": "{{random.uuid}}",
    "pressurecontrol": {{random.number(
        {
            "min":5,
            "max":30
        }
    )}},
    "o2stats": {{random.number(
        {
            "min":92,
            "max":98
```

```
        }
    )}},
    "minutevolume": {{random.number(
        {
            "min":5,
            "max":8
        }
    )}},
    "manufacturer": "{{random.arrayElement(
        ["3M", "GE","Vyaire", "Getinge"]
    )}}"
}
```

Anda sekarang dapat melihat data tiruan dengan template Uji dan menelan data tiruan ke Kinesis dengan data Kirim.

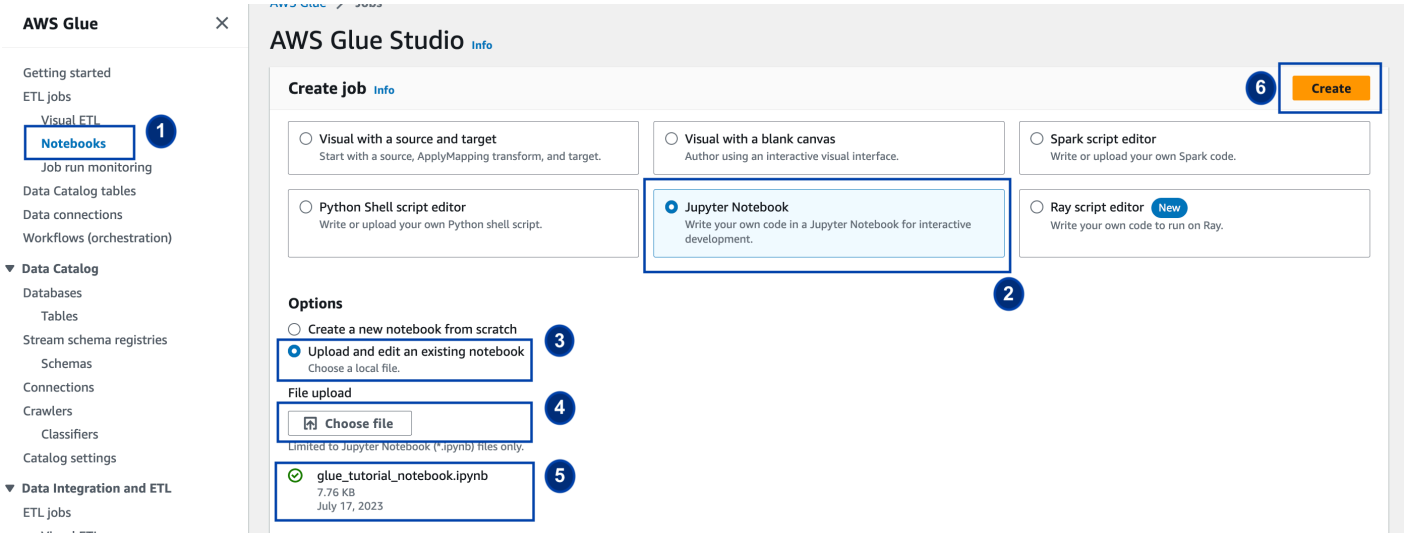
8. Klik Kirim data dan hasilkan 5-10K catatan ke Kinesis.

Membuat pekerjaan AWS Glue streaming dengan AWS Glue Studio

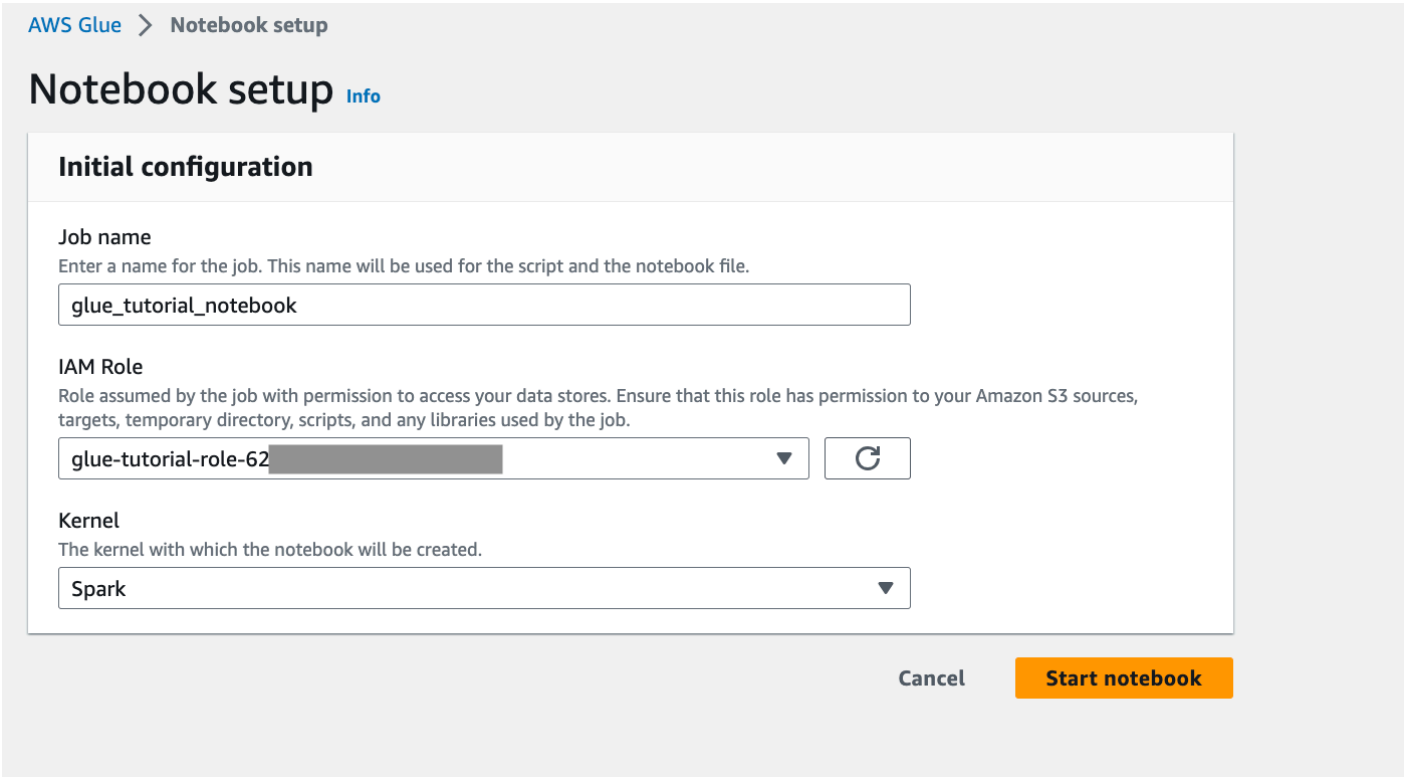
AWS GlueStudio adalah antarmuka visual yang menyederhanakan proses merancang, mengatur, dan memantau jaringan pipa integrasi data. Hal ini memungkinkan pengguna untuk membangun pipa transformasi data tanpa menulis kode ekstensif. Terlepas dari pengalaman penulisan pekerjaan visual, AWS Glue Studio juga menyertakan notebook Jupyter yang didukung oleh sesi AWS Glue Interaktif, yang akan Anda gunakan di sisa tutorial ini.

Siapkan pekerjaan sesi interaktif AWS Glue Streaming

1. Unduh file notebook yang disediakan dan simpan ke direktori lokal
2. Buka AWS Glue Console dan di panel kiri klik Notebooks > Jupyter Notebook > Upload dan edit notebook yang ada. Unggah buku catatan dari langkah sebelumnya dan klik Buat.



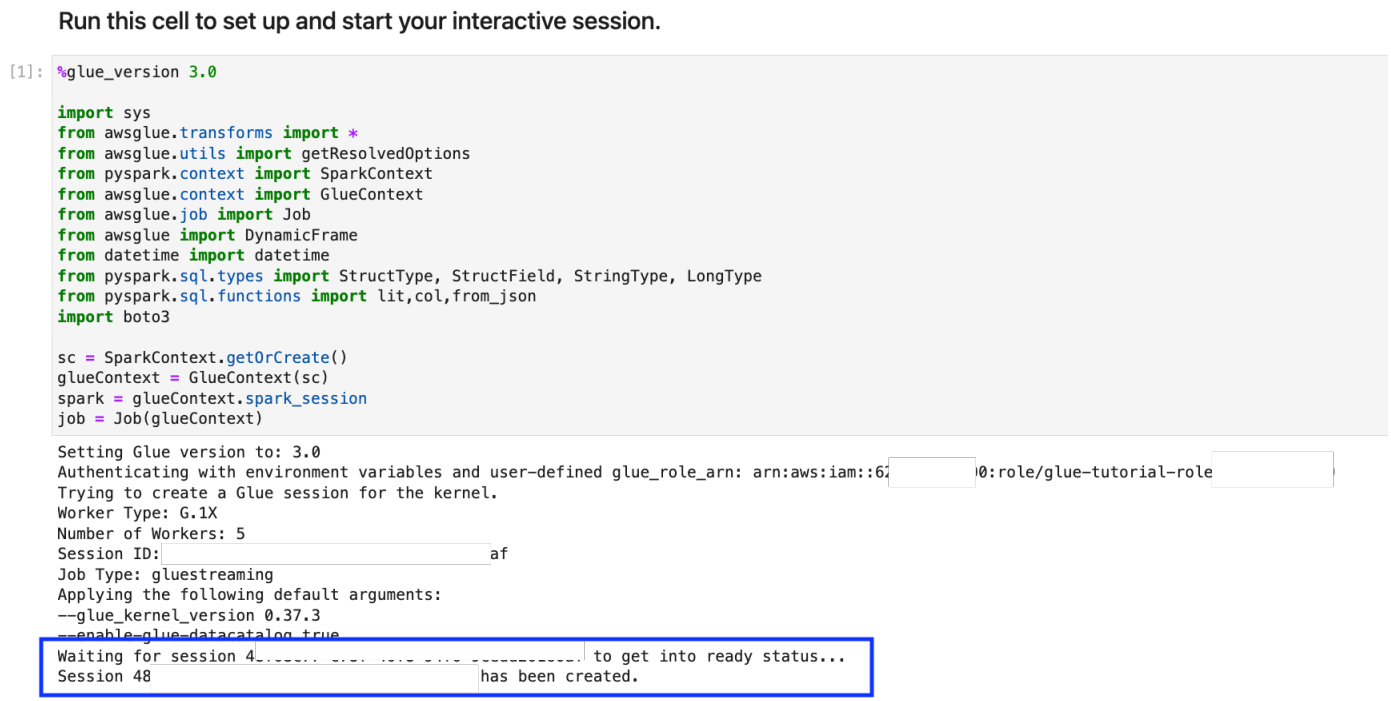
3. Berikan nama pekerjaan, peran, dan pilih kernel Spark default. Klik berikutnya Mulai notebook. Untuk Peran IAM, pilih peran yang disediakan oleh template. CloudFormation Anda dapat melihat ini di tab Output dari. CloudFormation



Notebook memiliki semua instruksi yang diperlukan untuk melanjutkan tutorial. Anda dapat menjalankan instruksi pada notebook atau mengikuti tutorial ini untuk melanjutkan pengembangan pekerjaan.

Jalankan sel notebook

1. (Opsional) Sel kode pertama, `%help` mencantumkan semua sihir notebook yang tersedia. Anda dapat melewati sel ini untuk saat ini, tetapi jangan ragu untuk menjelajahinya.
2. Mulailah dengan blok kode berikutnya`%streaming`. Keajaiban ini mengatur jenis pekerjaan ke streaming yang memungkinkan Anda mengembangkan, men-debug, dan menerapkan pekerjaan ETL AWS Glue streaming.
3. Jalankan sel berikutnya untuk membuat sesi AWS Glue interaktif. Sel output memiliki pesan yang mengonfirmasi pembuatan sesi.



4. Sel berikutnya mendefinisikan variabel. Ganti nilai dengan yang sesuai dengan pekerjaan Anda dan jalankan sel. Misalnya:

```
output_database_name="default"
output_table_name="test_stream_001"

account_id = boto3.client("sts").get_caller_identity()["Account"]
region_name=boto3.client('s3').meta.region_name
stream_arn_name = "arn:aws:kinesis:{}:{}:stream/GlueStreamTest-{}".format(region_name,account_id,account_id)
s3_bucket_name = "streaming-tutorial-s3-target-{}".format(account_id)

output_location = "s3://{}/streaming_output/".format(s3_bucket_name)
checkpoint_location = "s3://{}/checkpoint_location/".format(s3_bucket_name)
```

5. Karena data sudah dialirkan ke Kinesis Data Streams, sel Anda berikutnya akan mengkonsumsi hasil dari aliran. Jalankan sel berikutnya. Karena tidak ada pernyataan cetak, tidak ada output yang diharapkan dari sel ini.
6. Di sel berikut, Anda menjelajahi aliran masuk dengan mengambil kumpulan sampel dan mencetak skema dan data aktualnya. Misalnya:

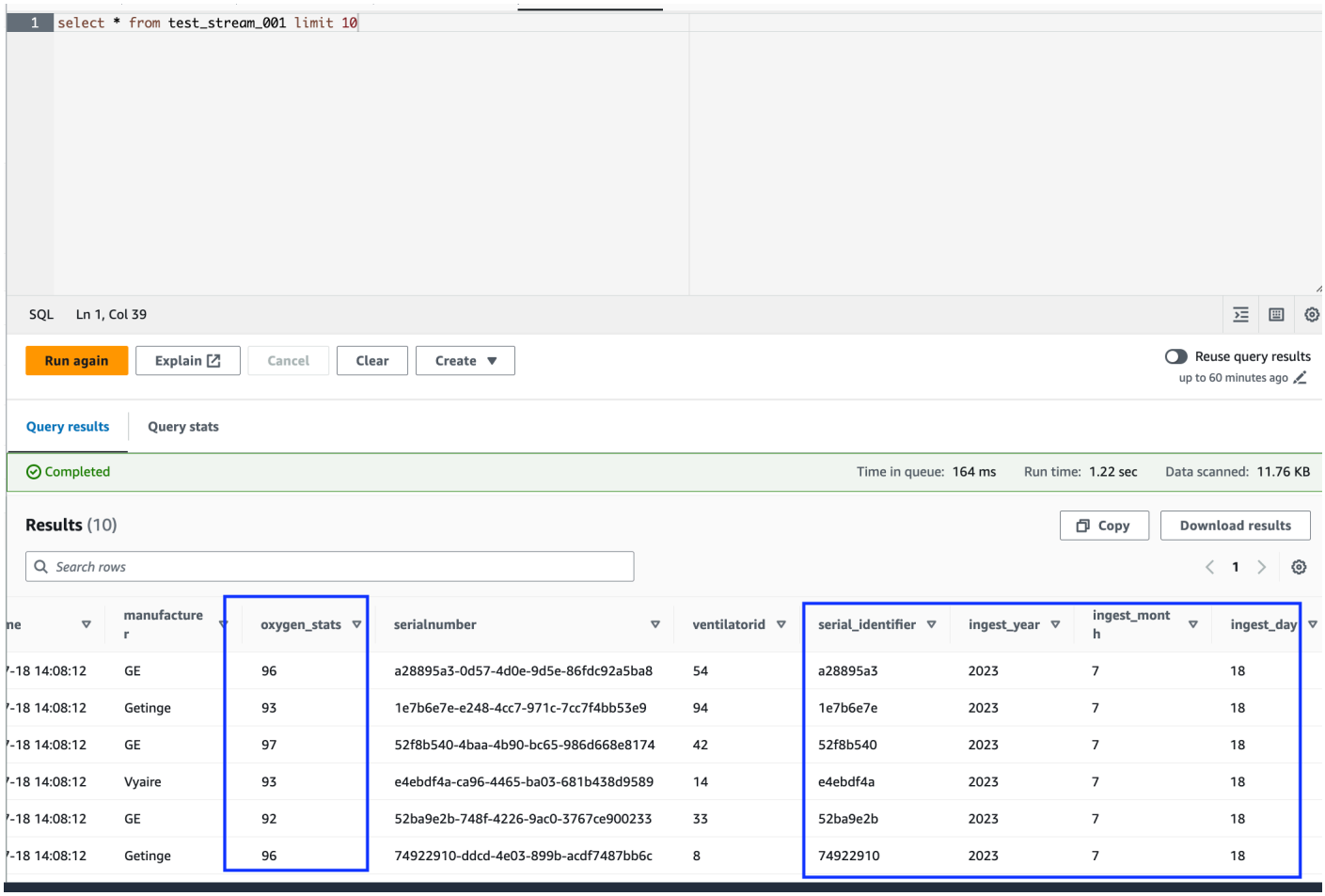
Sample and print the incoming records

the sampling is for debugging purpose. You may comment off the entire code cell below, before deploying the actual code

```
[4]: options = {
        "pollingTimeInMs": "20000",
        "windowSize": "5 seconds"
     }
     sampled_dynamic_frame = glueContext.getSampleStreamingDynamicFrame(data_frame, options, None)

     count_of_sampled_records = sampled_dynamic_frame.count()

     print(count_of_sampled_records)

     sampled_dynamic_frame.printSchema()

     sampled_dynamic_frame.toDF().show(10,False)
```

```
100
root
|-- eventtime: string
|-- manufacturer: string
|-- minutevolume: long
|-- o2stats: long
|-- pressurecontrol: long
|-- serialnumber: string
|-- ventilatorid: long

+-------------------+------------+------------+-------+---------------+------------------------------------+------------+
|eventtime          |manufacturer|minutevolume|o2stats|pressurecontrol|serialnumber                        |ventilatorid|
+-------------------+------------+------------+-------+---------------+------------------------------------+------------+
|2023-07-18 10:20:11|3M          |6           |92     |24             |a3e860ba-24b9-41c4-bc10-91c6b35e1406|6           |
|2023-07-18 10:20:11|Vyaire      |6           |95     |6              |96101dca-3e88-457f-b390-e3291df48a81|26          |
|2023-07-18 10:20:12|Getinge     |8           |96     |24             |18f3d448-1dee-4c80-835b-1a0daa818915|22          |
|2023-07-18 10:20:12|Getinge     |7           |98     |30             |25f425cd-b978-4953-9a03-4d607a639364|91          |
|2023-07-18 10:20:12|GE          |5           |93     |25             |2cd7cdc2-f5f5-4ff2-ae32-45e5a8922d53|93          |
```

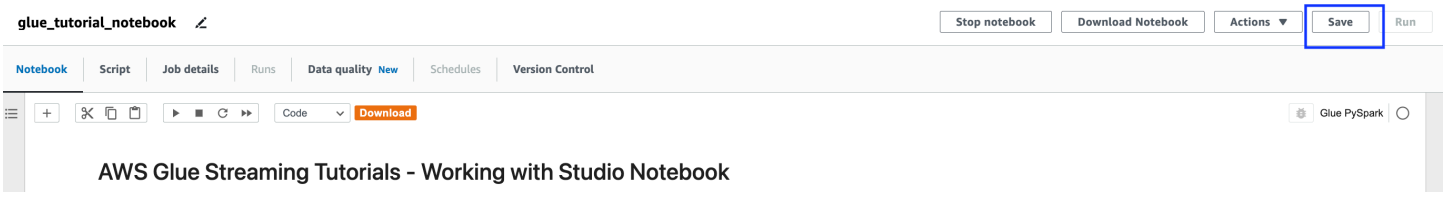
7. Selanjutnya, tentukan logika transformasi data aktual. Sel terdiri dari `processBatch` metode yang dipicu selama setiap batch mikro. Jalankan sel. Pada tingkat tinggi, kami melakukan hal berikut ke aliran masuk:
   a. Pilih subset dari kolom input.
   b. Ganti nama kolom (o2stats menjadi oxygen_stats).
   c. Turunkan kolom baru (serial_identifier, ingest_year, ingest_month dan ingest_day).
   d. Simpan hasilnya ke dalam ember Amazon S3 dan buat juga tabel katalog yang dipartisi AWS Glue
8. Di sel terakhir, Anda memicu batch proses setiap 10 detik. Jalankan sel dan tunggu sekitar 30 detik untuk mengisi ember Amazon S3 dan tabel katalogAWS Glue.
9. Terakhir, telusuri data yang disimpan menggunakan editor kueri Amazon Athena. Anda dapat melihat kolom yang diganti namanya dan juga partisi baru.



Notebook memiliki semua instruksi yang diperlukan untuk melanjutkan tutorial. Anda dapat menjalankan instruksi pada notebook atau mengikuti tutorial ini untuk melanjutkan pengembangan pekerjaan.

Simpan dan jalankan AWS Glue pekerjaan

Dengan pengembangan dan pengujian aplikasi Anda selesai menggunakan notebook sesi interaktif, klik Simpan di bagian atas antarmuka notebook. Setelah disimpan, Anda juga dapat menjalankan aplikasi sebagai pekerjaan.



Bersihkan

Untuk menghindari biaya tambahan ke akun Anda, hentikan pekerjaan streaming yang Anda mulai sebagai bagian dari instruksi. Anda dapat melakukan ini dengan menghentikan notebook, yang akan mengakhiri sesi. Kosongkan bucket Amazon S3 dan hapus AWS CloudFormation tumpukan yang Anda sediakan sebelumnya.

Kesimpulan

Dalam tutorial ini, kami menunjukkan bagaimana melakukan hal berikut menggunakan notebook AWS Glue Studio

- Penulis pekerjaan streaming ETL menggunakan notebook
- Pratinjau aliran data yang masuk
- Kode dan perbaiki masalah tanpa harus mempublikasikan AWS Glue pekerjaan
- Tinjau kode end-to-end kerja, hapus debugging, dan cetak pernyataan atau sel dari buku catatan
- Publikasikan kode sebagai AWS Glue pekerjaan

Tujuan dari tutorial ini adalah untuk memberi Anda pengalaman langsung bekerja dengan AWS Glue Streaming dan sesi interaktif. Kami mendorong Anda untuk menggunakan ini sebagai referensi untuk kasus penggunaan AWS Glue Streaming individual Anda. Untuk informasi selengkapnya, lihat Memulai dengan sesi AWS Glue interaktif.

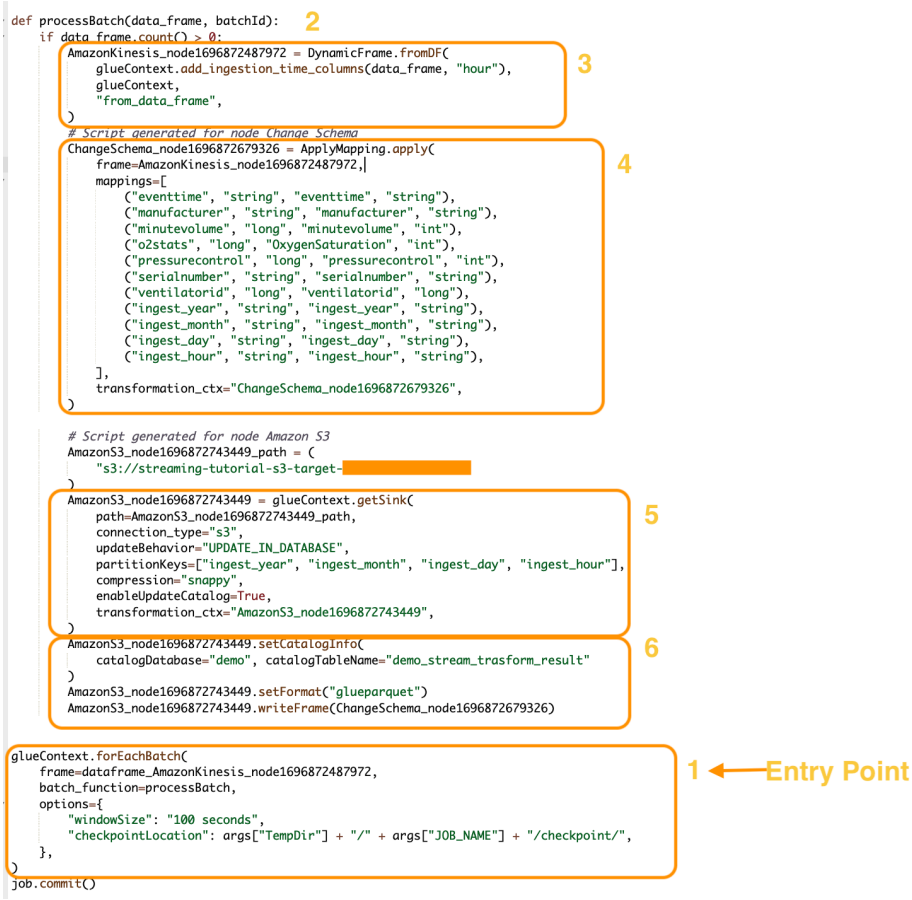
AWS GlueKonsep streaming

Bagian berikut memberikan informasi tentang konsep AWS Glue Streaming.

Topik

Anatomi pekerjaan AWS Glue streaming

AWS Gluepekerjaan streaming beroperasi pada paradigma streaming Spark dan memanfaatkan streaming terstruktur dari kerangka Spark. Pekerjaan streaming terus-menerus melakukan polling pada sumber data streaming, pada interval waktu tertentu, untuk mengambil catatan sebagai batch mikro. Bagian berikut memeriksa bagian-bagian yang berbeda dari pekerjaan AWS Glue streaming.



forEachBatch

`forEachBatch`Metode ini adalah titik masuk dari pekerjaan AWS Glue streaming. AWS Gluepekerjaan streaming menggunakan `forEachBatch` metode untuk polling data yang berfungsi seperti iterator yang tetap aktif selama siklus hidup pekerjaan streaming dan secara teratur melakukan polling sumber streaming untuk data baru dan memproses data terbaru dalam batch mikro.

```
glueContext.forEachBatch(
    frame=dataFrame_AmazonKinesis_node1696872487972,
    batch_function=processBatch,
    options={
```

```
        "windowSize": "100 seconds",
        "checkpointLocation": args["TempDir"] + "/" + args["JOB_NAME"] + "/
checkpoint/",
    },
)
```

Konfigurasikan `frame` properti `forEachBatch` untuk menentukan sumber streaming. Dalam contoh ini, simpul sumber yang Anda buat di kanvas kosong selama pembuatan pekerjaan diisi dengan default DataFrame pekerjaan. Tetapkan `batch_function` properti sebagai `function` yang Anda putuskan untuk dipanggil untuk setiap operasi batch mikro. Anda harus menentukan fungsi untuk menangani transformasi batch pada data yang masuk.

Sumber

Pada langkah pertama `processBatch` fungsi, program memverifikasi jumlah catatan DataFrame yang Anda definisikan sebagai properti bingkai. `forEachBatch` Program ini menambahkan stempel waktu konsumsi ke yang tidak kosong. DataFrame `data_frame.count()>0`Klausul menentukan apakah batch mikro terbaru tidak kosong dan siap untuk diproses lebih lanjut.

```
def processBatch(data_frame, batchId):
    if data_frame.count() >0:
        AmazonKinesis_node1696872487972 = DynamicFrame.fromDF(
            glueContext.add_ingestion_time_columns(data_frame, "hour"),
            glueContext,
            "from_data_frame",
        )
```

Pemetaan

Bagian selanjutnya dari program ini adalah menerapkan pemetaan. `Mapping.apply`Metode pada percikan DataFrame memungkinkan Anda untuk menentukan aturan transformasi di sekitar elemen data. Biasanya Anda dapat mengganti nama, mengubah tipe data, atau menerapkan fungsi kustom pada kolom data sumber dan memetakannya ke kolom target.

```
    #Script generated for node ChangeSchema
    ChangeSchema_node16986872679326 = ApplyMapping.apply(
        frame =  AmazonKinesis_node1696872487972,
```

```
        mappings = [
            ("eventtime", "string", "eventtime", "string"),
            ("manufacturer", "string", "manufacturer", "string"),
            ("minutevolume", "long", "minutevolume", "int"),
            ("o2stats", "long", "OxygenSaturation", "int"),
            ("pressurecontrol", "long", "pressurecontrol", "int"),
            ("serialnumber", "string", "serialnumber", "string"),
            ("ventilatorid", "long", "ventilatorid", "long"),
            ("ingest_year", "string", "ingest_year", "string"),
            ("ingest_month", "string", "ingest_month", "string"),
            ("ingest_day", "string", "ingest_day", "string"),
            ("ingest_hour", "string", "ingest_hour", "string"),
        ],
        transformation_ctx="ChangeSchema_node16986872679326",
    )
        )
```

Wastafel

Di bagian ini, kumpulan data yang masuk dari sumber streaming disimpan di lokasi target. Dalam contoh ini kita akan menulis data ke lokasi Amazon S3. Detail `AmazonS3_node_path` properti diisi sebelumnya sebagaimana ditentukan oleh pengaturan yang Anda gunakan selama pembuatan lapangan kerja dari kanvas. Anda dapat mengatur `updateBehavior` berdasarkan kasus penggunaan dan memutuskan untuk tidak memperbarui tabel katalog data, atau Membuat katalog data dan memperbarui skema katalog data pada proses berikutnya, atau membuat tabel katalog dan tidak memperbarui definisi skema pada proses berikutnya.

`partitionKeys`Properti mendefinisikan opsi partisi penyimpanan. Perilaku default adalah mempartisi data per data `ingestion_time_columns` yang dibuat tersedia di bagian sumber. `compression`Properti ini memungkinkan Anda untuk mengatur algoritma kompresi yang akan diterapkan selama penulisan target. Anda memiliki opsi untuk mengatur Snappy, LZO, atau GZIP sebagai teknik kompresi. `enableUpdateCatalog`Properti mengontrol apakah tabel AWS Glue katalog perlu diperbarui. Pilihan yang tersedia untuk properti ini adalah `True` atau`False`.

```
    #Script generated for node Amazon S3
    AmazonS3_node1696872743449 = glueContext.getSink(
        path =  AmazonS3_node1696872743449_path,
        connection_type = "s3",
        updateBehavior = "UPDATE_IN_DATABASE",
```

```
        partitionKeys = ["ingest_year", "ingest_month", "ingest_day", "ingest_hour"],
        compression = "snappy",
        enableUpdateCatalog = True,
        transformation_ctx = "AmazonS3_node1696872743449",
    )
```

AWS GlueWastafel katalog

Bagian pekerjaan ini mengontrol perilaku pembaruan tabel AWS Glue katalog. Set `catalogDatabase` dan `catalogTableName` properti per nama database AWS Glue Katalog Anda dan nama tabel yang terkait dengan AWS Glue pekerjaan yang Anda desain. Anda dapat menentukan format file dari data target melalui `setFormat` properti. Untuk contoh ini kita akan menyimpan data dalam format parket.

Setelah Anda mengatur dan menjalankan pekerjaan AWS Glue streaming merujuk tutorial ini, data streaming yang dihasilkan Amazon Kinesis Data Streams akan disimpan di lokasi Amazon S3 dalam format parket dengan kompresi tajam. Pada menjalankan pekerjaan streaming yang berhasil, Anda akan dapat menanyakan data melaluiAmazon Athena.

```
    AmazonS3_node1696872743449 = setCatalogInfo(
        catalogDatabase =  "demo", catalogTableName = "demo_stream_transform_result"
    )
    AmazonS3_node1696872743449.setFormat("glueparquet")
    AmazonS3_node1696872743449.writeFormat("ChangeSchema_node16986872679326")
    )
```

Koneksi Kafka

Mengkhususkan koneksi ke klaster Kafka atau klaster Amazon Managed Streaming for Apache Kafka.

Anda dapat membaca dan menulis ke aliran data Kafka menggunakan informasi yang disimpan dalam tabel Katalog Data, atau dengan memberikan informasi untuk langsung mengakses aliran data. Anda dapat membaca informasi dari Kafka menjadi Spark DataFrame, lalu mengubahnya menjadi Glue AWS . DynamicFrame Anda dapat menulis DynamicFrames ke Kafka dalam format

JSON. Jika Anda langsung mengakses aliran data, gunakan opsi ini untuk memberikan informasi tentang cara mengakses aliran data.

Jika Anda menggunakan `getCatalogSource` atau `create_data_frame_from_catalog` menggunakan catatan dari sumber streaming Kafka, `getCatalogSink` atau `write_dynamic_frame_from_catalog` untuk menulis catatan ke Kafka, dan pekerjaan tersebut memiliki database Katalog Data dan informasi nama tabel, dan dapat menggunakannya untuk mendapatkan beberapa parameter dasar untuk membaca dari sumber streaming Kafka. Jika Anda menggunakan`getSource`,`getCatalogSink`,`getSourceWithFormat`,`getSinkWithFormat`, `createDataFrameFromOptions` atau`create_data_frame_from_options`, atau`write_dynamic_frame_from_catalog`, Anda harus menentukan parameter dasar ini menggunakan opsi koneksi yang dijelaskan di sini.

Anda dapat menentukan opsi koneksi untuk Kafka menggunakan argumen berikut untuk metode yang ditentukan di `GlueContext` kelas.

- Skala
  - `connectionOptions`: Gunakan dengan`getSource`,`createDataFrameFromOptions`, `getSink`
  - `additionalOptions`: Gunakan dengan`getCatalogSource`, `getCatalogSink`
  - `options`: Gunakan dengan`getSourceWithFormat`, `getSinkWithFormat`
- Python
  - `connection_options`: Gunakan dengan`create_data_frame_from_options`, `write_dynamic_frame_from_options`
  - `additional_options`: Gunakan dengan`create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog`
  - `options`: Gunakan dengan`getSource`, `getSink`

Untuk catatan dan batasan tentang streaming pekerjaan ETL, konsultasikanthe section called “Streaming catatan dan batasan ETL”.

Konfigurasikan Kafka

Tidak ada AWS prasyarat untuk menghubungkan ke aliran Kafka yang tersedia melalui internet.

Anda dapat membuat koneksi AWS Glue Kafka untuk mengelola kredenal koneksi Anda. Untuk informasi selengkapnya, lihat the section called “Membuat koneksi untuk aliran data Kafka”.

Dalam konfigurasi pekerjaan AWS Glue Anda, berikan *connectionName* sebagai koneksi jaringan Tambahan, lalu, dalam panggilan metode Anda, berikan *connectionName* ke parameter. `connectionName`

Dalam kasus tertentu, Anda perlu mengkonfigurasi prasyarat tambahan:

- Jika menggunakan Amazon Managed Streaming for Apache Kafka dengan autentikasi IAM, Anda memerlukan konfigurasi IAM yang sesuai.
- Jika menggunakan Amazon Managed Streaming for Apache Kafka dalam Amazon VPC, Anda memerlukan konfigurasi Amazon VPC yang sesuai. Anda perlu membuat koneksi AWS Glue yang menyediakan informasi koneksi Amazon VPC. Anda akan memerlukan konfigurasi pekerjaan Anda untuk menyertakan koneksi AWS Glue sebagai koneksi jaringan Tambahan.

Untuk informasi lebih lanjut tentang prasyarat pekerjaan Streaming ETL, lihat. the section called “Lowongan kerja Streaming ETL”

Contoh: Membaca dari aliran Kafka

Digunakan bersama denganthe section called “forEachBatch”.

Contoh untuk sumber streaming Kafka:

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "startingOffsets": "earliest",
      "inferSchema": "true",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

Contoh: Menulis ke aliran Kafka

Contoh untuk menulis ke Kafka:

Contoh dengan `getSink` metode:

```
data_frame_datasource0 =
```

```
glueContext.getSink(
 connectionType="kafka",
 connectionOptions={
  JsonOptions("""{
   "connectionName": "ConfluentKafka",
   "classification": "json",
   "topic": "kafka-auth-topic",
   "typeOfData": "kafka"}
  """)},
 transformationContext="dataframe_ApacheKafka_node1711729173428")
 .getDataFrame()
```

Contoh dengan `write_dynamic_frame.from_options` metode:

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.write_dynamic_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

Referensi opsi koneksi Kafka

Saat membaca, gunakan opsi koneksi berikut dengan`"connectionType": "kafka"`:

- `"bootstrap.servers"` (Wajib) Daftar URL server bootstrap, misalnya, sebagai `b-1.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094`. Opsi ini harus ditentukan dalam panggilan API atau didefinisikan dalam metadata tabel dalam Katalog Data.
- `"security.protocol"`(Wajib) Protokol yang digunakan untuk berkomunikasi dengan broker. Nilai yang mungkin adalah `"SSL"` atau `"PLAINTEXT"`.
- `"topicName"`(Wajib) Daftar topik yang dipisahkan koma untuk berlangganan. Anda harus menentukan satu dan hanya satu dari`"topicName"`, `"assign"` atau`"subscribePattern"`.
- `"assign"`: (Diperlukan) String JSON yang menentukan spesifik `TopicPartitions` untuk dikonsumsi. Anda harus menentukan satu dan hanya satu dari`"topicName"`, `"assign"` atau`"subscribePattern"`.

  Contoh: '{"topicA”: [0,1], "topiCB”: [2,4]}'

- `"subscribePattern"`: (Wajib) Sebuah string regex Java yang mengidentifikasi daftar topik untuk berlangganan. Anda harus menentukan satu dan hanya satu dari`"topicName"`, `"assign"` atau`"subscribePattern"`.

  Contoh: 'topik. *'

- `"classification"`(Wajib) Format file yang digunakan oleh data dalam catatan. Diperlukan kecuali disediakan melalui Katalog Data.
- `"delimiter"`(Opsional) Pemisah nilai yang digunakan saat `classification` CSV. Defaultnya adalah `",`.”
- `"startingOffsets"`: (Opsional) Posisi awal dalam topik Kafka tempat untuk membaca data. Nilai yang mungkin adalah `"earliest"` atau `"latest"`. Nilai default-nya adalah `"latest"`.
- `"startingTimestamp"`: (Opsional, hanya didukung untuk AWS Glue versi 4.0 atau yang lebih baru) Stempel waktu catatan dalam topik Kafka untuk membaca data dari. Nilai yang mungkin adalah string Timestamp dalam format UTC dalam pola `yyyy-mm-ddTHH:MM:SSZ` (di mana `Z` mewakili zona waktu UTC offset dengan +/-. Misalnya: “2023-04-04T 08:00:00-04:00 “).

  Catatan: Hanya satu dari 'startingOffsets' atau 'startingTimeStamp' yang dapat hadir dalam daftar Opsi Koneksi dari skrip streaming AWS Glue, termasuk kedua properti ini akan mengakibatkan kegagalan pekerjaan.

- `"endingOffsets"`: (Opsional) Titik akhir ketika kueri batch berakhir. Nilai yang mungkin adalah `"latest"` atau string JSON yang menentukan sebuah ending offset untuk setiap `TopicPartition`.

  Untuk string JSON, formatnya adalah `{"topicA":{"0":23,"1":-1},"topicB":{"0":-1}}`. Nilai `-1` sebagai offset mewakili `"latest"`.

- `"pollTimeoutMs"`: (Opsional) Waktu habis dalam milidetik untuk memeriksa status data dari Kafka di pelaksana tugas Spark. Nilai default-nya adalah `512`.
- `"numRetries"`: (Opsional) Jumlah percobaan sebelum gagal untuk mengambil offset Kafka. Nilai default-nya adalah `3`.
- `"retryIntervalMs"`: (Opsional) Waktu dalam milidetik yang digunakan untuk menunggu sebelum mencoba kembali untuk mengambil offset Kafka. Nilai default-nya adalah `10`.
- `"maxOffsetsPerTrigger"`: (Opsional) Batas tingkat pada jumlah maksimum offset yang diproses untuk setiap interval pemicu. Jumlah total offset yang ditentukan dibagi secara proporsional di seluruh `topicPartitions` dengan volume yang berbeda. Nilai default-nya adalah nol, yang berarti bahwa konsumen membaca semua offset sampai diketahui offset terbaru.

- `"minPartitions"`: (Opsional) Jumlah minimum partisi yang diinginkan yang akan dibaca dari Kafka. Nilai default-nya adalah nol, yang berarti bahwa jumlah partisi spark sama dengan jumlah partisi Kafka.
- `"includeHeaders"`: (Opsional) Apakah akan menyertakan header Kafka. Ketika opsi diatur ke “true”, output data akan berisi kolom tambahan bernama “glue_streaming_kafka_headers” dengan tipe. `Array[Struct(key: String, value: String)]` Nilai defaultnya adalah “false”. Opsi ini tersedia dalam AWS Glue versi 3.0 atau yang lebih baru.
- `"schema"`: (Diperlukan saat InferSchema disetel ke false) Skema yang digunakan untuk memproses muatan. Jika klasifikasi adalah `avro` skema yang disediakan harus dalam format skema Avro. Jika klasifikasi tidak, skema `avro` yang disediakan harus dalam format skema DDL.

  Berikut ini adalah contoh skema.

  Example in DDL schema format

  ```
  'column1' INT, 'column2' STRING , 'column3' FLOAT
  ```

  Example in Avro schema format

  ```
  {
  "type":"array",
  "items":
  {
  "type":"record",
  "name":"test",
  "fields":
  [
    {
      "name":"_id",
      "type":"string"
    },
    {
      "name":"index",
      "type":
      [
        "int",
        "string",
        "float"
      ]
    }
  ]
  ```

```
}
}
```

- `"inferSchema"`: (Opsional) Nilai default adalah 'salah'. Jika disetel ke 'true', skema akan terdeteksi saat runtime dari payload di dalamnya. `foreachbatch`
- `"avroSchema"`: (Usang) Parameter yang digunakan untuk menentukan skema data Avro saat format Avro digunakan. Parameter ini sekarang tidak digunakan lagi. Gunakan parameter `schema`.
- `"addRecordTimestamp"`: (Opsional) Ketika opsi ini diatur ke 'true', output data akan berisi kolom tambahan bernama “__src_timestamp” yang menunjukkan waktu ketika catatan terkait diterima oleh topik. Nilai defaultnya adalah 'salah'. Opsi ini didukung di AWS Glue versi 4.0 atau yang lebih baru.
- `"emitConsumerLagMetrics"`: (Opsional) Ketika opsi disetel ke 'true', untuk setiap batch, itu akan memancarkan metrik untuk durasi antara catatan tertua yang diterima oleh topik dan waktu tiba. AWS Glue CloudWatch Nama metriknya adalah “glue.driver.streaming. maxConsumerLagInMs”. Nilai defaultnya adalah 'salah'. Opsi ini didukung di AWS Glue versi 4.0 atau yang lebih baru.

Saat menulis, gunakan opsi koneksi berikut dengan`"connectionType": "kafka"`:

- `"connectionName"`(Wajib) Nama sambungan AWS Glue yang digunakan untuk menghubungkan ke cluster Kafka (mirip dengan sumber Kafka).
- `"topic"`(Wajib) Jika kolom topik ada maka nilainya digunakan sebagai topik saat menulis baris yang diberikan ke Kafka, kecuali jika opsi konfigurasi topik disetel. Artinya, opsi `topic` konfigurasi mengesampingkan kolom topik.
- `"partition"`(Opsional) Jika nomor partisi yang valid ditentukan, itu `partition` akan digunakan saat mengirim catatan.

  Jika tidak ada partisi yang ditentukan tetapi a `key` hadir, partisi akan dipilih menggunakan hash dari kunci.

  Jika tidak `partition` ada `key` atau tidak ada, partisi akan dipilih berdasarkan partisi lengket perubahan tersebut ketika setidaknya batch.size byte diproduksi ke partisi.
- `"key"`(Opsional) Digunakan untuk partisi jika nol`partition`.
- `"classification"`(Opsional) Format file yang digunakan oleh data dalam catatan. Kami hanya mendukung JSON, CSV dan Avro.

Dengan format Avro, kami dapat menyediakan AvroSchema khusus untuk diserialisasikan, tetapi perhatikan bahwa ini perlu disediakan pada sumber untuk deserialisasi juga. Lain, secara default menggunakan Apache AvroSchema untuk serialisasi.

Selain itu, Anda dapat menyempurnakan wastafel Kafka sesuai kebutuhan dengan memperbarui parameter konfigurasi produsen Kafka. Perhatikan bahwa tidak ada daftar izin pada opsi koneksi, semua pasangan nilai kunci tetap ada di wastafel apa adanya.

Namun, ada daftar kecil opsi penolakan yang tidak akan berlaku. Untuk informasi selengkapnya, lihat konfigurasi khusus Kafka.

Koneksi Kinesis

Anda dapat membaca dan menulis ke aliran data Amazon Kinesis menggunakan informasi yang disimpan dalam tabel Katalog Data, atau dengan memberikan informasi untuk mengakses aliran data secara langsung. Anda dapat membaca informasi dari Kinesis menjadi Spark DataFrame, lalu mengubahnya menjadi Glue. AWS DynamicFrame Anda dapat menulis DynamicFrames ke Kinesis dalam format JSON. Jika Anda langsung mengakses aliran data, gunakan opsi ini untuk memberikan informasi tentang cara mengakses aliran data.

Jika Anda menggunakan `getCatalogSource` atau `create_data_frame_from_catalog` menggunakan catatan dari sumber streaming Kinesis, pekerjaan tersebut memiliki database Katalog Data dan informasi nama tabel, dan dapat menggunakannya untuk mendapatkan beberapa parameter dasar untuk membaca dari sumber streaming Kinesis. Jika Anda menggunakan`getSource`,`getSourceWithFormat`, `createDataFrameFromOptions` atau`create_data_frame_from_options`, Anda harus menentukan parameter dasar ini menggunakan opsi koneksi yang dijelaskan di sini.

Anda dapat menentukan opsi koneksi untuk Kinesis menggunakan argumen berikut untuk metode yang ditentukan di `GlueContext` kelas.

- Skala
  - `connectionOptions`: Gunakan dengan`getSource`,`createDataFrameFromOptions`, `getSink`
  - `additionalOptions`: Gunakan dengan`getCatalogSource`, `getCatalogSink`
  - `options`: Gunakan dengan`getSourceWithFormat`, `getSinkWithFormat`
- Python

- `connection_options`: Gunakan dengan`create_data_frame_from_options`, `write_dynamic_frame_from_options`
- `additional_options`: Gunakan dengan`create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog`
- `options`: Gunakan dengan`getSource`, `getSink`

Untuk catatan dan batasan tentang pekerjaan Streaming ETL, konsultasikanthe section called "Streaming catatan dan batasan ETL".

Konfigurasikan Kinesis

Untuk terhubung ke aliran data Kinesis dalam pekerjaan AWS Glue Spark, Anda memerlukan beberapa prasyarat:

- Jika membaca, tugas AWS Glue harus memiliki izin IAM tingkat akses Baca ke aliran data Kinesis.
- Jika menulis, pekerjaan AWS Glue harus memiliki izin IAM tingkat akses Tulis ke aliran data Kinesis.

Dalam kasus tertentu, Anda perlu mengkonfigurasi prasyarat tambahan:

- Jika pekerjaan AWS Glue Anda dikonfigurasi dengan koneksi jaringan Tambahan (biasanya untuk terhubung ke kumpulan data lain) dan salah satu koneksi tersebut menyediakan opsi Jaringan VPC Amazon, ini akan mengarahkan pekerjaan Anda untuk berkomunikasi melalui Amazon VPC. Dalam hal ini Anda juga perlu mengonfigurasi aliran data Kinesis Anda untuk berkomunikasi melalui Amazon VPC. Anda dapat melakukan ini dengan membuat titik akhir VPC antarmuka antara VPC Amazon dan aliran data Kinesis Anda. Untuk informasi selengkapnya, lihat Menggunakan Kinesis Data Streams dengan Titik Akhir VPC Antarmuka.
- Saat menentukan Amazon Kinesis Data Streams di akun lain, Anda harus menyiapkan peran dan kebijakan untuk mengizinkan akses lintas akun. Untuk informasi selengkapnya, lihat Contoh: Baca Dari Pengaliran Kinesis di Akun Berbeda.

Untuk informasi lebih lanjut tentang prasyarat pekerjaan Streaming ETL, lihat. the section called "Lowongan kerja Streaming ETL"

Baca dari Kinesis

Contoh: Membaca dari aliran Kinesis

Digunakan bersama denganthe section called “forEachBatch”.

Contoh untuk sumber streaming Amazon Kinesis:

```
kinesis_options =
   { "streamARN": "arn:aws:kinesis:us-east-2:777788889999:stream/fromOptionsStream",
     "startingPosition": "TRIM_HORIZON",
     "inferSchema": "true",
     "classification": "json"
   }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kinesis",
 connection_options=kinesis_options)
```

Menulis ke Kinesis

Contoh: Menulis ke aliran Kinesis

Digunakan bersama denganthe section called “forEachBatch”. Anda DynamicFrame akan ditulis ke aliran dalam format JSON. Jika pekerjaan tidak dapat menulis setelah beberapa kali percobaan ulang, itu akan gagal. Secara default, setiap DynamicFrame catatan akan dikirim ke aliran Kinesis satu per satu. Anda dapat mengonfigurasi perilaku ini menggunakan `aggregationEnabled` dan parameter terkait.

Contoh menulis ke Amazon Kinesis dari pekerjaan streaming:

Python

```
glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite
    connection_type="kinesis",
    connection_options={
        "partitionKey": "part1",
        "streamARN": "arn:aws:kinesis:us-east-1:111122223333:stream/streamName",
    }
)
```

Scala

```
glueContext.getSinkWithFormat(
                connectionType="kinesis",
                options=JsonOptions("""{
                    "streamARN": "arn:aws:kinesis:us-
east-1:111122223333:stream/streamName",
                    "partitionKey": "part1"
                }"""),
            )
            .writeDynamicFrame(frameToWrite)
```

Parameter koneksi Kinesis

Menetapkan opsi koneksi untuk Amazon Kinesis Data Streams.

Gunakan opsi koneksi berikut untuk sumber data streaming Kinesis:

- `"streamARN"`(Wajib) Digunakan untuk Baca/Tulis. ARN dari aliran data Kinesis.
- `"classification"`(Diperlukan untuk dibaca) Digunakan untuk Baca. Format file yang digunakan oleh data dalam catatan. Diperlukan kecuali disediakan melalui Katalog Data.
- `"streamName"`— (Opsional) Digunakan untuk Baca. Nama aliran data Kinesis untuk dibaca. Digunakan dengan`endpointUrl`.
- `"endpointUrl"`— (Opsional) Digunakan untuk Baca. Default: "https://kinesis.us-east-1.amazonaws.com”. AWS Titik akhir dari aliran Kinesis. Anda tidak perlu mengubah ini kecuali Anda terhubung ke wilayah khusus.
- `"partitionKey"`— (Opsional) Digunakan untuk Menulis. Kunci partisi Kinesis digunakan saat memproduksi catatan.
- `"delimiter"`(Opsional) Digunakan untuk Baca. Pemisah nilai yang digunakan saat `classification` CSV. Defaultnya adalah `","`.”
- `"startingPosition"`: (Opsional) Digunakan untuk Baca. Posisi awal dalam aliran data Kinesis untuk membaca data dari. Nilai yang mungkin adalah`"latest"`,, `"trim_horizon"``"earliest"`, atau string Timestamp dalam format UTC dalam pola `yyyy-mm-ddTHH:MM:SSZ` (di mana `Z` mewakili offset zona waktu UTC dengan +/-. Misalnya “2023-04-04T 08:00:00-04:00 “). Nilai default-nya adalah `"latest"`. Catatan: string Timestamp dalam Format UTC untuk hanya didukung untuk `"startingPosition"` AWS Glue versi 4.0 atau yang lebih baru.

- `"failOnDataLoss"`: (Opsional) Gagal pekerjaan jika ada pecahan aktif yang hilang atau kedaluwarsa. Nilai default-nya adalah `"false"`.
- `"awsSTSRoleARN"`: (Opsional) Digunakan untuk Baca/Tulis. Nama Sumber Daya Amazon (ARN) dari peran yang akan diasumsikan menggunakan AWS Security Token Service ()AWS STS. Peran ini harus memiliki izin untuk mendeskripsikan atau membaca operasi rekaman untuk aliran data Kinesis. Anda harus menggunakan parameter ini saat mengakses aliran data di akun yang berbeda. Digunakan bersama dengan`"awsSTSSessionName"`.
- `"awsSTSSessionName"`: (Opsional) Digunakan untuk Baca/Tulis. Pengidentifikasi untuk sesi dengan asumsi peran menggunakan. AWS STS Anda harus menggunakan parameter ini saat mengakses aliran data di akun yang berbeda. Digunakan bersama dengan`"awsSTSRoleARN"`.
- `"awsSTSEndpoint"`: (Opsional) AWS STS Titik akhir yang digunakan saat menghubungkan ke Kinesis dengan peran yang diasumsikan. Ini memungkinkan penggunaan AWS STS titik akhir regional dalam VPC, yang tidak dimungkinkan dengan titik akhir global default.
- `"maxFetchTimeInMs"`: (Opsional) Digunakan untuk Baca. Waktu maksimum yang dihabiskan untuk pelaksana pekerjaan untuk membaca catatan untuk batch saat ini dari aliran data Kinesis, ditentukan dalam milidetik (ms). Beberapa panggilan `GetRecords` API dapat dilakukan dalam waktu ini. Nilai default-nya adalah `1000`.
- `"maxFetchRecordsPerShard"`: (Opsional) Digunakan untuk Baca. Jumlah maksimum catatan yang diambil per pecahan dalam aliran data Kinesis per mikrobatch. Catatan: Klien dapat melampaui batas ini jika pekerjaan streaming telah membaca catatan tambahan dari Kinesis (dalam panggilan get-records yang sama). Jika `maxFetchRecordsPerShard` perlu ketat maka itu harus kelipatan`maxRecordPerRead`. Nilai default-nya adalah `100000`.
- `"maxRecordPerRead"`: (Opsional) Digunakan untuk Baca. Jumlah maksimum catatan untuk diambil dari aliran data Kinesis dalam `getRecords` setiap operasi. Nilai default-nya adalah `10000`.
- `"addIdleTimeBetweenReads"`: (Opsional) Digunakan untuk Baca. Menambahkan penundaan waktu antara dua operasi berturut-turut`getRecords`. Nilai default-nya adalah `"False"`. Opsi ini hanya dapat dikonfigurasi untuk Glue versi 2.0 dan di atasnya.
- `"idleTimeBetweenReadsInMs"`: (Opsional) Digunakan untuk Baca. Waktu tunda minimum antara dua `getRecords` operasi berturut-turut, ditentukan dalam ms. Nilai default-nya adalah `1000`. Opsi ini hanya dapat dikonfigurasi untuk Glue versi 2.0 dan di atasnya.
- `"describeShardInterval"`: (Opsional) Digunakan untuk Baca. Interval waktu minimum antara dua panggilan `ListShards` API untuk skrip Anda untuk mempertimbangkan resharding. Untuk informasi selengkapnya, lihat Strategi Penyerpihan Kembali di Panduan Developer Amazon Kinesis Data Streams. Nilai default-nya adalah `1s`.

- `"numRetries"`: (Opsional) Digunakan untuk Baca. Jumlah maksimum percobaan ulang untuk permintaan API Kinesis Data Streams. Nilai default-nya adalah `3`.
- `"retryIntervalMs"`: (Opsional) Digunakan untuk Baca. Periode waktu pendinginan (ditentukan dalam ms) sebelum mencoba kembali panggilan API Kinesis Data Streams. Nilai default-nya adalah `1000`.
- `"maxRetryIntervalMs"`: (Opsional) Digunakan untuk Baca. Periode waktu pendinginan maksimum (ditentukan dalam ms) antara dua percobaan ulang panggilan API Kinesis Data Streams. Nilai default-nya adalah `10000`.
- `"avoidEmptyBatches"`: (Opsional) Digunakan untuk Baca. Hindari membuat pekerjaan microbatch kosong dengan memeriksa data yang belum dibaca di aliran data Kinesis sebelum batch dimulai. Nilai default-nya adalah `"False"`.
- `"schema"`: (Diperlukan saat InferSchema disetel ke false) Digunakan untuk Baca. Skema yang digunakan untuk memproses muatan. Jika klasifikasi adalah `avro` skema yang disediakan harus dalam format skema Avro. Jika klasifikasi tidak, skema `avro` yang disediakan harus dalam format skema DDL.

  Berikut ini adalah contoh skema.

  Example in DDL schema format

  ```
  `column1` INT, `column2` STRING , `column3` FLOAT
  ```

  Example in Avro schema format

  ```
  {
    "type":"array",
    "items":
    {
      "type":"record",
      "name":"test",
      "fields":
      [
        {
          "name":"_id",
          "type":"string"
        },
        {
          "name":"index",
          "type":
          [
  ```

```
            "int",
            "string",
            "float"
          ]
        }
      ]
    }
}
```

- `"inferSchema"`: (Opsional) Digunakan untuk Baca. Nilai defaultnya adalah 'salah'. Jika disetel ke 'true', skema akan terdeteksi saat runtime dari payload di dalamnya. `foreachbatch`
- `"avroSchema"`: (Usang) Digunakan untuk Baca. Parameter yang digunakan untuk menentukan skema data Avro saat format Avro digunakan. Parameter ini sekarang tidak digunakan lagi. Gunakan parameter `schema`.
- `"addRecordTimestamp"`: (Opsional) Digunakan untuk Baca. Ketika opsi ini diatur ke 'true', output data akan berisi kolom tambahan bernama “__src_timestamp” yang menunjukkan waktu ketika catatan terkait diterima oleh aliran. Nilai defaultnya adalah 'salah'. Opsi ini didukung di AWS Glue versi 4.0 atau yang lebih baru.
- `"emitConsumerLagMetrics"`: (Opsional) Digunakan untuk Baca. Ketika opsi disetel ke 'true', untuk setiap batch, itu akan memancarkan metrik untuk durasi antara rekaman tertua yang diterima oleh aliran dan waktu tiba. AWS Glue CloudWatch Nama metriknya adalah “glue.driver.streaming. maxConsumerLagInMs”. Nilai defaultnya adalah 'salah'. Opsi ini didukung di AWS Glue versi 4.0 atau yang lebih baru.
- `"fanoutConsumerARN"`: (Opsional) Digunakan untuk Baca. ARN dari konsumen aliran Kinesis untuk aliran yang ditentukan dalam. `streamARN` Digunakan untuk mengaktifkan mode fan-out yang disempurnakan untuk koneksi Kinesis Anda. Untuk informasi lebih lanjut tentang mengonsumsi aliran Kinesis dengan fan-out yang ditingkatkan, lihat. the section called “Menggunakan fan-out yang disempurnakan dalam pekerjaan streaming Kinesis”
- `"recordMaxBufferedTime"`— (Opsional) Digunakan untuk Menulis. Default: 1000 (ms). Waktu maksimum sebuah rekaman di-buffer sambil menunggu untuk ditulis.
- `"aggregationEnabled"`— (Opsional) Digunakan untuk Menulis. Default: benar. Menentukan apakah catatan harus dikumpulkan sebelum mengirim mereka ke Kinesis.
- `"aggregationMaxSize"`— (Opsional) Digunakan untuk Menulis. Default: 51200 (byte). Jika catatan lebih besar dari batas ini, itu akan melewati agregator. Catatan Kinesis memberlakukan batas 50KB pada ukuran rekaman. Jika Anda menetapkan ini di luar 50KB, catatan kebesaran akan ditolak oleh Kinesis.

- `"aggregationMaxCount"`— (Opsional) Digunakan untuk Menulis. Standar: 4294967295. Jumlah maksimum item untuk dikemas ke dalam catatan agregat.
- `"producerRateLimit"`— (Opsional) Digunakan untuk Menulis. Default: 150 (%). Membatasi throughput per shard yang dikirim dari satu produsen (seperti pekerjaan Anda), sebagai persentase dari batas backend.
- `"collectionMaxCount"`— (Opsional) Digunakan untuk Menulis. Default: 500. Jumlah maksimum item untuk dikemas ke dalam PutRecords permintaan.
- `"collectionMaxSize"`— (Opsional) Digunakan untuk Menulis. Default: 5242880 (byte). Jumlah maksimum data untuk dikirim dengan PutRecords permintaan.

AWS Glue Opsi streaming

Mengkhususkan koneksi ke klaster Kafka atau klaster Amazon Managed Streaming for Apache Kafka.

Anda dapat membaca dan menulis ke aliran data Kafka menggunakan informasi yang disimpan dalam tabel Katalog Data, atau dengan memberikan informasi untuk langsung mengakses aliran data. Anda dapat membaca informasi dari Kafka menjadi Spark DataFrame, lalu mengubahnya menjadi Glue AWS . DynamicFrame Anda dapat menulis DynamicFrames ke Kafka dalam format JSON. Jika Anda langsung mengakses aliran data, gunakan opsi ini untuk memberikan informasi tentang cara mengakses aliran data.

Jika Anda menggunakan `getCatalogSource` atau `create_data_frame_from_catalog` menggunakan catatan dari sumber streaming Kafka, `getCatalogSink` atau `write_dynamic_frame_from_catalog` untuk menulis catatan ke Kafka, dan pekerjaan tersebut memiliki database Katalog Data dan informasi nama tabel, dan dapat menggunakannya untuk mendapatkan beberapa parameter dasar untuk membaca dari sumber streaming Kafka. Jika Anda menggunakan`getSource`,`getCatalogSink`,`getSourceWithFormat`,`getSinkWithFormat`, `createDataFrameFromOptions` atau`create_data_frame_from_options`, atau`write_dynamic_frame_from_catalog`, Anda harus menentukan parameter dasar ini menggunakan opsi koneksi yang dijelaskan di sini.

Anda dapat menentukan opsi koneksi untuk Kafka menggunakan argumen berikut untuk metode yang ditentukan di `GlueContext` kelas.

- Skala

  - `connectionOptions`: Gunakan dengan`getSource`,`createDataFrameFromOptions`, `getSink`
  - `additionalOptions`: Gunakan dengan`getCatalogSource`, `getCatalogSink`
  - `options`: Gunakan dengan`getSourceWithFormat`, `getSinkWithFormat`
- Python
  - `connection_options`: Gunakan dengan`create_data_frame_from_options`, `write_dynamic_frame_from_options`
  - `additional_options`: Gunakan dengan`create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog`
  - `options`: Gunakan dengan`getSource`, `getSink`

Untuk catatan dan batasan tentang streaming pekerjaan ETL, konsultasikanthe section called “Streaming catatan dan batasan ETL”.

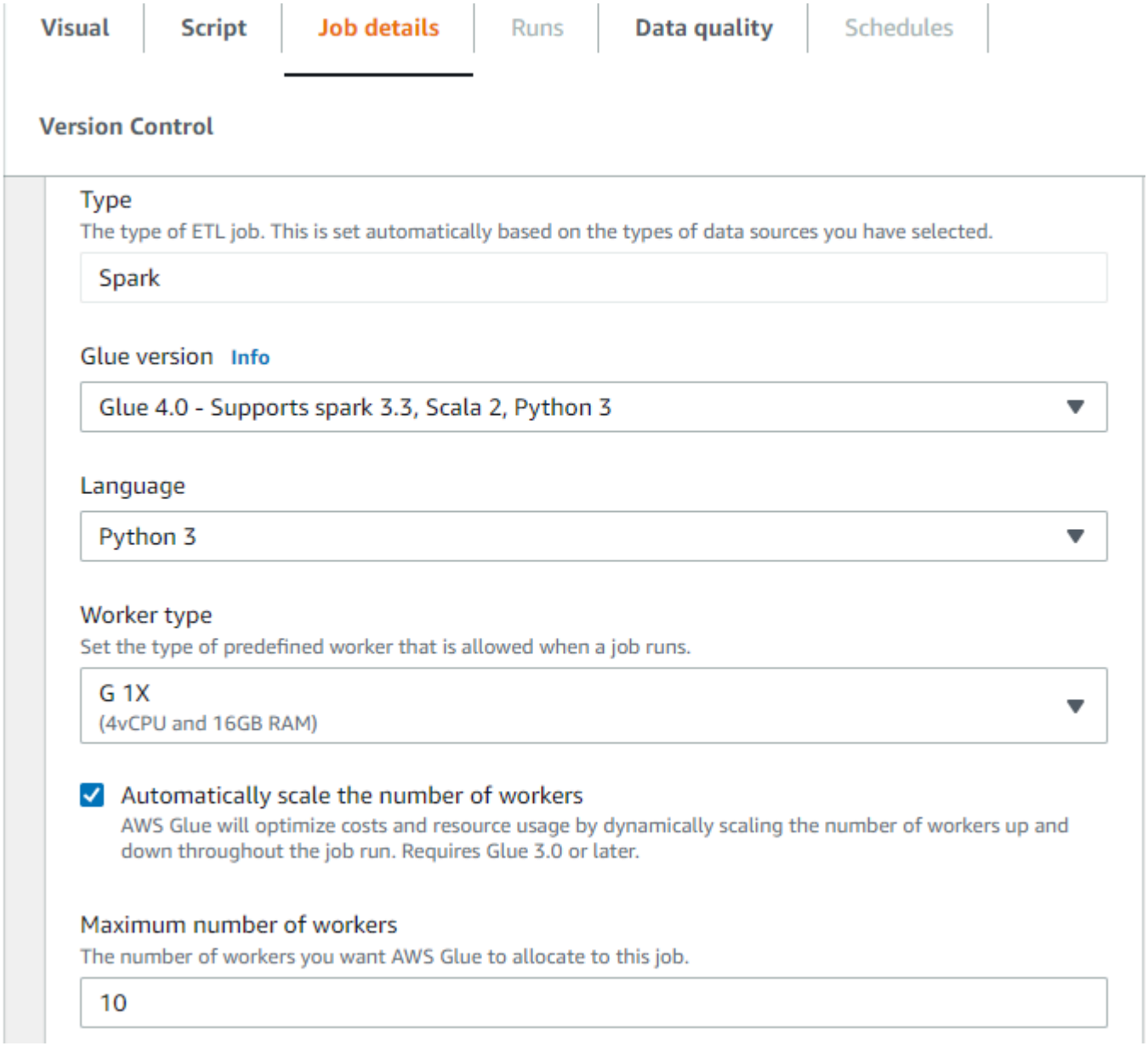
AWS Gluestreaming penskalaan otomatis

Bagian berikut memberikan informasi tentang AWS Glue streaming autoscaling

Mengaktifkan Auto Scaling di AWS Glue Studio

Pada tab Job details di AWS Glue Studio, pilih jenis sebagai Spark atau Spark Streaming, dan versi Glue sebagai **`Glue 3.0`** atau. **`Glue 4.0`** Kemudian kotak centang akan muncul di bawah Jenis Pekerja.

- Pilih opsi Skala jumlah pekerja secara otomatis.
- Tetapkan jumlah maksimum pekerja untuk menentukan jumlah maksimum pekerja yang dapat dijual untuk menjalankan pekerjaan.



Mengaktifkan Auto Scaling dengan AWS CLI atau SDK

Untuk mengaktifkan Auto Scaling Dari AWS CLI untuk menjalankan pekerjaan Anda, jalankan `start-job-run` dengan konfigurasi berikut:

```
{
    "JobName": "<your job name>",
    "Arguments": {
        "--enable-auto-scaling": "true"
    },
    "WorkerType": "G.2X", // G.1X and G.2X are allowed for Auto Scaling Jobs
```

```
    "NumberOfWorkers": 20, // represents Maximum number of workers
    ...other job run configurations...
}
```

Setelah menjalankan tugas ETL selesai, Anda juga dapat menelepon `get-job-run` untuk memeriksa penggunaan sumber daya aktual dari pekerjaan yang dijalankan dalam DPU-detik. Catatan: bidang baru DPUSeconds hanya akan muncul untuk pekerjaan batch Anda di AWS Glue 3.0 atau yang lebih baru diaktifkan dengan Auto Scaling. Bidang ini tidak didukung untuk pekerjaan streaming.

```
$ aws glue get-job-run --job-name your-job-name --run-id jr_xx --endpoint https://
glue.us-east-1.amazonaws.com --region us-east-1
{
    "JobRun": {
        ...
        "GlueVersion": "3.0",
        "DPUSeconds": 386.0
    }
}
```

Anda juga dapat mengonfigurasi job run dengan Auto Scaling menggunakan AWS GlueSDK dengan konfigurasi yang sama.

Cara kerjanya

Penskalaan di seluruh microbatch

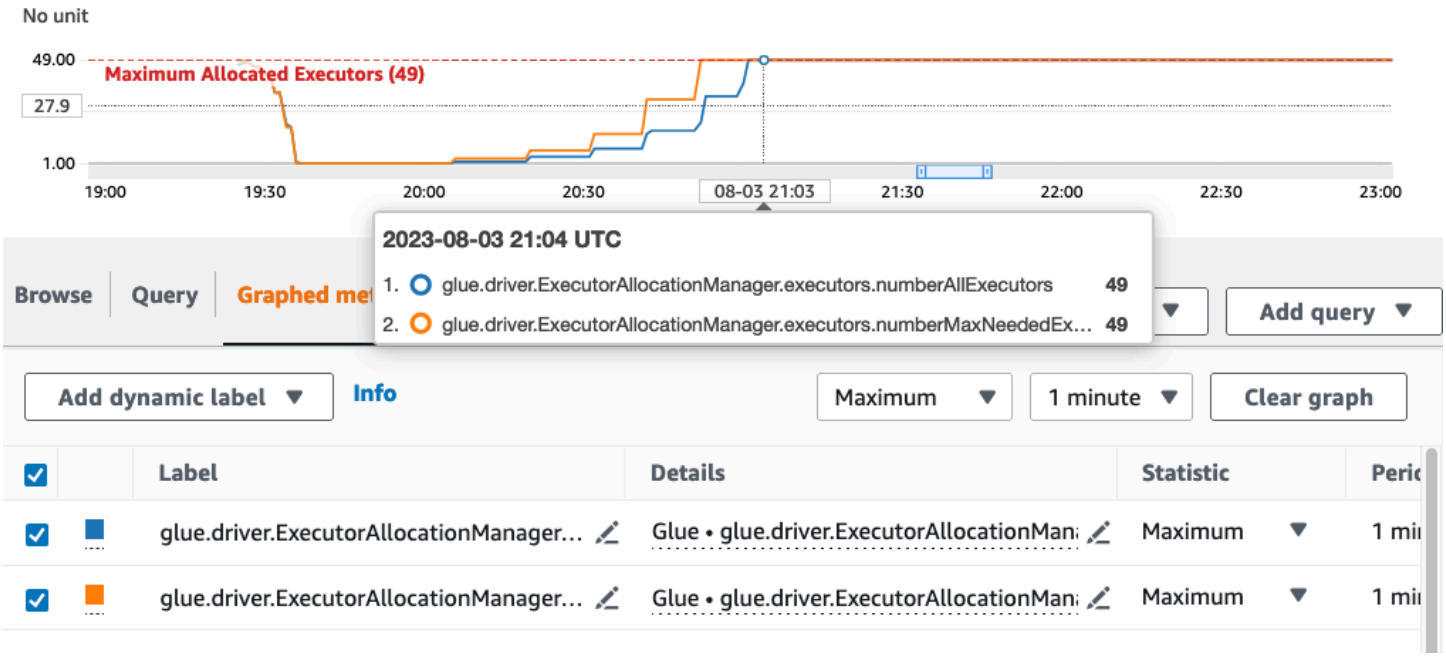
Contoh berikut digunakan untuk menggambarkan cara kerja penskalaan otomatis.

- Anda memiliki AWS Glue pekerjaan yang dimulai dengan 50 DPU.
- Penskalaan otomatis diaktifkan.

Dalam contoh ini, AWS Glue lihat metrik batchProcessingTime InMs ““untuk beberapa batch mikro dan tentukan apakah pekerjaan Anda selesai dalam ukuran jendela yang telah Anda tetapkan. Jika pekerjaan Anda selesai lebih cepat dan tergantung pada seberapa cepat mereka selesai, AWS Glue dapat menurunkan skala. Metrik ini, diplot dengan” numberAllExecutors “dapat dipantau Amazon CloudWatch untuk melihat cara kerja penskalaan otomatis.

Jumlah eksekutor secara eksponensial naik atau turun hanya setelah setiap batch mikro selesai. Seperti yang dapat Anda lihat dari log Amazon CloudWatch Pemantauan, AWS Glue lihat jumlah

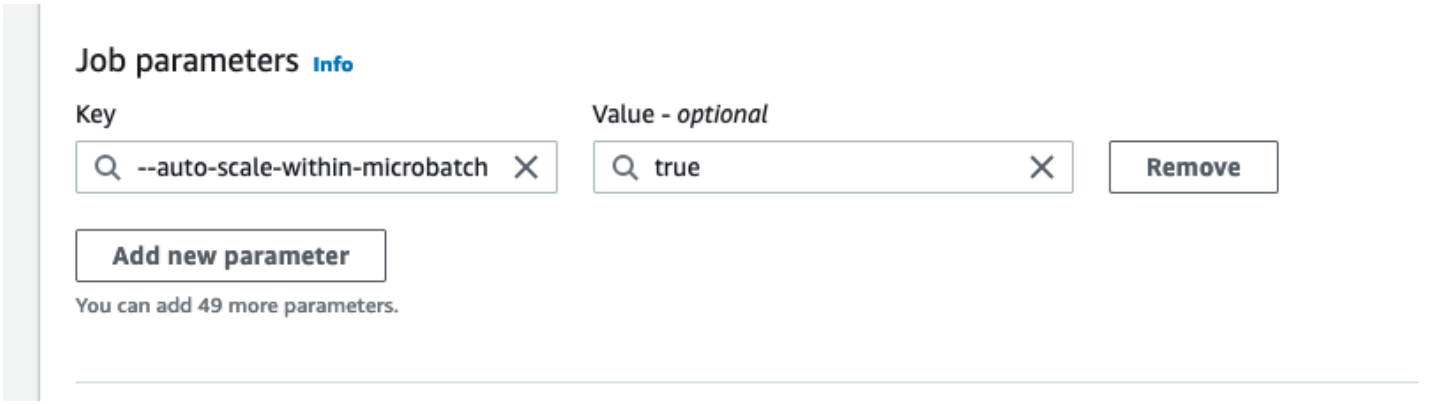
pelaksana yang dibutuhkan (Garis Oranye) dan skala pelaksana (garis biru) untuk mencocokkannya secara otomatis.



Setelah AWS Glue menurunkan jumlah pelaksana dan mengamati bahwa volume data meningkat, akibatnya meningkatkan waktu pemrosesan batch mikro, akan AWS Glue menskalakan hingga 50 DPU, yang merupakan batas atas yang ditentukan.

Penskalaan dalam microbatch

Dalam contoh di atas, sistem memantau beberapa batch mikro yang telah selesai untuk membuat keputusan apakah akan naik atau turun. Jendela yang lebih panjang membutuhkan penskalaan otomatis untuk merespons lebih cepat dalam microbatch, daripada menunggu beberapa batch mikro. Untuk kasus ini, Anda dapat menggunakan konfigurasi tambahan `--auto-scaIe-within-microbatch` untuk`true`. Anda dapat menambahkan ini ke properti AWS Glue pekerjaan AWS Glue Studio seperti yang ditunjukkan di bawah ini.



Jendela pemeliharaan untuk AWS Glue Streaming

AWS Glue secara berkala melakukan kegiatan pemeliharaan. Selama jendela pemeliharaan ini, Anda AWS Glue perlu memulai ulang pekerjaan streaming Anda. Anda dapat mengontrol kapan pekerjaan dimulai ulang dengan menentukan jendela pemeliharaan. Di bagian ini, kami menguraikan di mana Anda dapat mengatur jendela pemeliharaan dan perilaku tertentu yang harus Anda pertimbangkan.

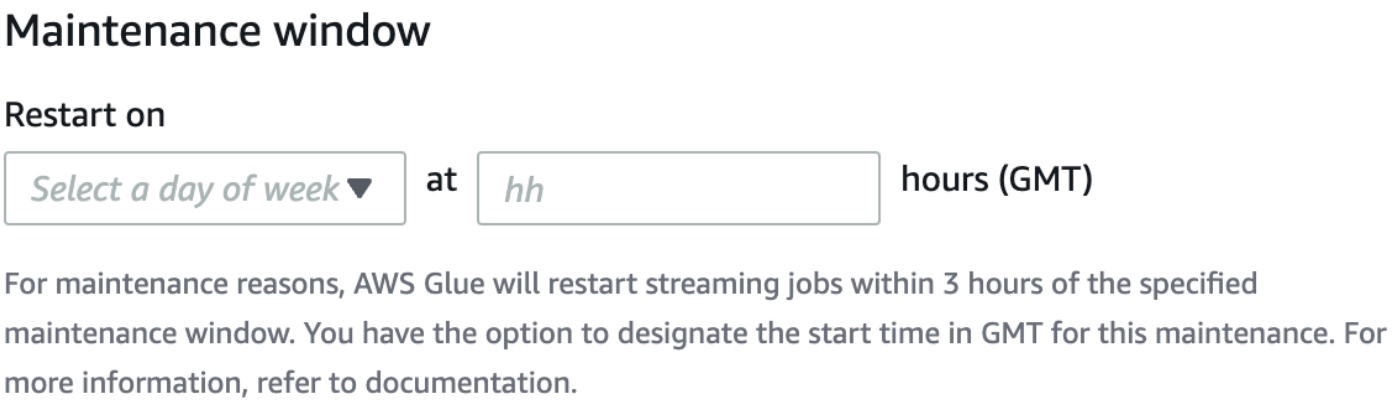
Topik

Menyiapkan jendela pemeliharaan

Anda dapat mengatur jendela pemeliharaan menggunakan AWS Glue Studio atau API.

Menyiapkan jendela pemeliharaan di AWS Glue Studio

Anda dapat menentukan jendela pemeliharaan di halaman Detail Pekerjaan AWS Glue Streaming Anda. Anda dapat menentukan hari dan waktu di GMT. AWS Glue akan memulai kembali pekerjaan Anda dalam jendela waktu yang ditentukan.



Menyiapkan jendela pemeliharaan di API

Anda juga dapat mengatur jendela pemeliharaan di Create Job API. Berikut adalah contoh mengonfigurasi jendela pemeliharaan melalui API.

```
aws glue create-job –name jobName –role roleArnForTheJob –command
 Name=gluestreaming,ScriptLocation=s3-path-to-the-script --maintenance-window="Sun:10"
```

Contoh perintah adalah sebagai berikut:

```
aws glue create-job –name testMaintenance –role arn:aws:iam::012345678901:role/
Glue_DefaultRole –command Name=gluestreaming,ScriptLocation=s3://glue-example-test/
example.py –maintenance-window="Sun:10
```

Perilaku jendela pemeliharaan

AWS Glue melalui serangkaian langkah untuk memutuskan kapan harus memulai kembali pekerjaan:

1. Ketika pekerjaan streaming baru dimulai, AWS Glue pertama-tama periksa apakah ada batas waktu yang terkait dengan pekerjaan yang dijalankan. Batas waktu memungkinkan Anda mengonfigurasi waktu akhir pekerjaan. Jika batas waktu kurang dari 7 hari, maka pekerjaan tidak akan dimulai ulang.
2. Jika batas waktu lebih besar dari 7 hari, maka AWS Glue periksa apakah jendela pemeliharaan dikonfigurasi untuk pekerjaan itu. Jika ya maka jendela itu diambil dan jendela akan ditetapkan untuk menjalankan pekerjaan. AWS Glue akan memulai kembali pekerjaan dalam waktu 3 jam dari jendela pemeliharaan yang ditentukan. Misalnya, jika Anda mengatur jendela pemeliharaan untuk hari Senin pukul 10:00 GMT, pekerjaan Anda akan dimulai kembali antara 10:00 GMT hingga 1:00 GMT.
3. Jika jendela pemeliharaan tidak dikonfigurasi, AWS Glue secara otomatis menyetel waktu restart ke 7 hari setelah waktu inisiasi menjalankan pekerjaan. Misalnya, jika Anda memulai pekerjaan Anda pada 7/1/2024 12:00 GMT dan Anda tidak menentukan jendela pemeliharaan, pekerjaan Anda akan diatur untuk dimulai ulang pada 7/8/2024 pukul 12:00 GMT.

> **Note**
>
> Jika Anda sudah menjalankan pekerjaan streaming, perubahan ini akan berdampak pada Anda mulai 1 Juli 2024. Anda akan memiliki waktu hingga 30 Juni untuk mengkonfigurasi jendela pemeliharaan Anda. Setelah 1 Juli, pekerjaan streaming apa pun yang Anda mulai akan dimulai ulang sesuai dokumentasi ini. Jika Anda memerlukan dukungan tambahan, Anda dapat menghubungi AWS Support.

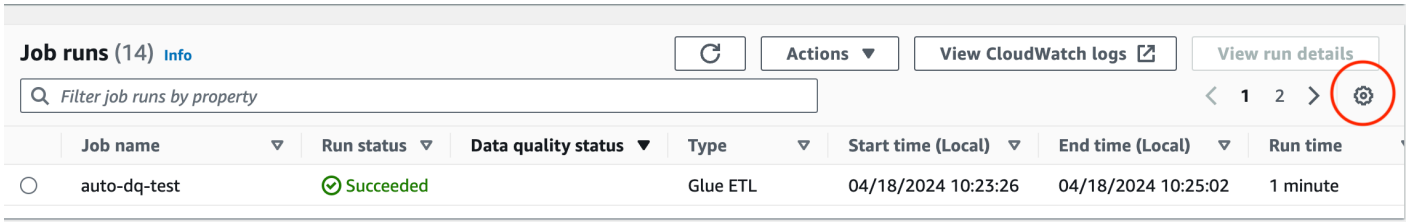
4. Terkadang, AWS Glue mungkin tidak dapat memulai kembali pekerjaan, terutama ketika batch mikro yang sedang berlangsung tidak diproses. Dalam hal ini, pekerjaan tidak akan terganggu. Dalam hal ini, AWS Glue akan restart pekerjaan setelah 14 hari, dan dalam hal ini, jendela pemeliharaan tidak dihormati.

Pemantauan Job

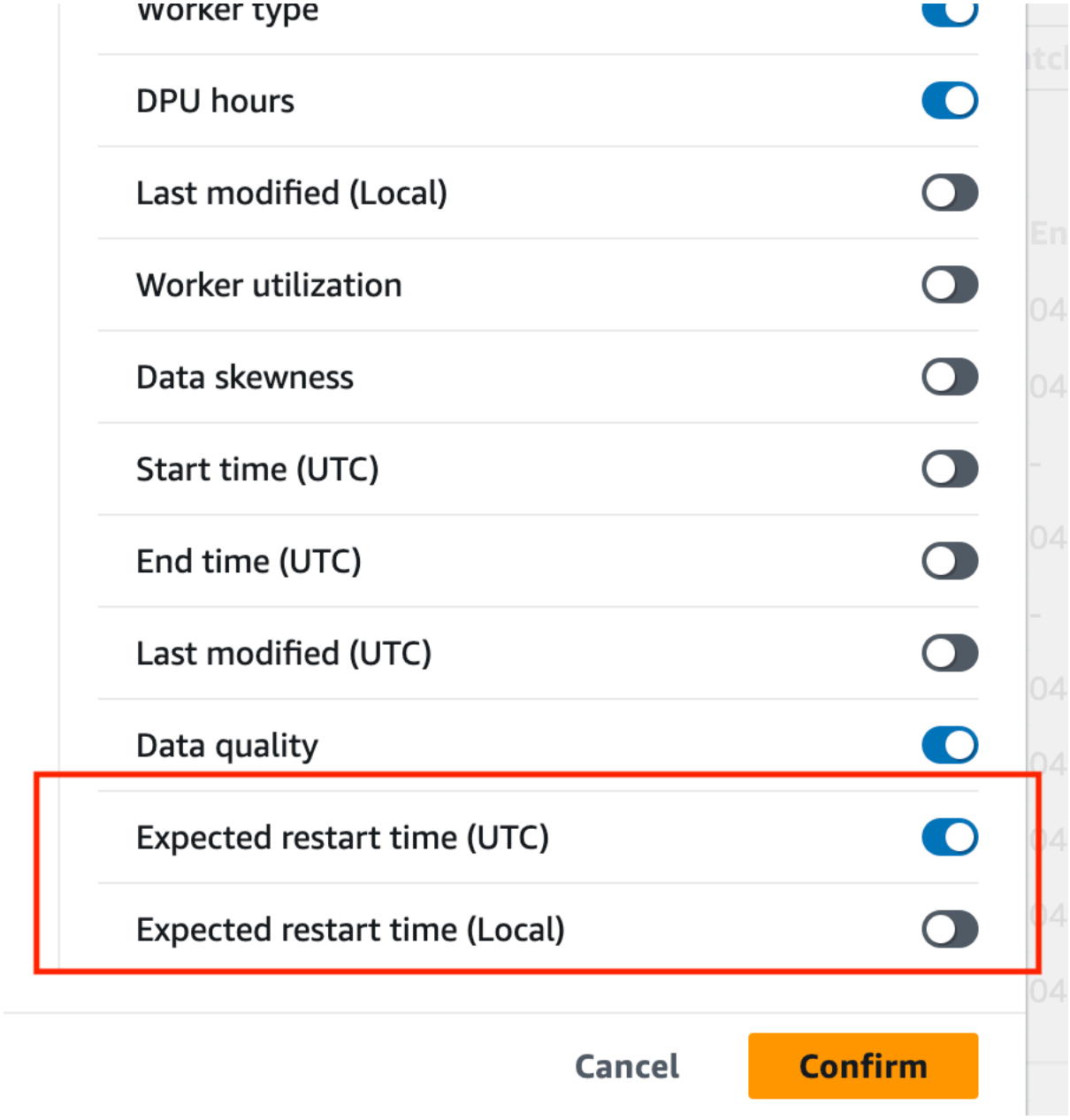
Anda dapat memantau pekerjaan di halaman Pemantauan AWS Glue Studio.

Untuk melihat waktu restart pekerjaan streaming berikutnya yang diharapkan, tampilkan kolom pada tabel Job running di halaman Monitoring.

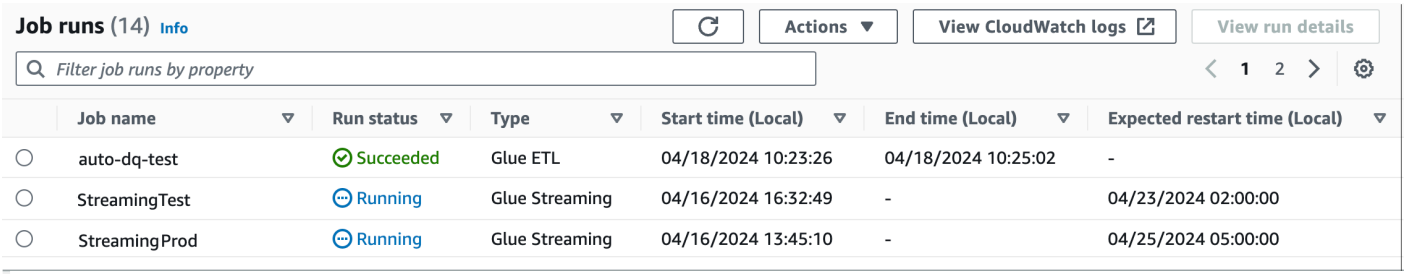
1. Klik ikon Gear di kanan atas tabel.



2. Gulir ke bawah, dan nyalakan kolom Waktu restart yang diharapkan. Pilihan waktu UTC dan Lokal tersedia.



3. Anda kemudian dapat melihat kolom dalam tabel.



Pekerjaan asli akan memiliki status “EXPIRED” dan instance pekerjaan baru akan memiliki status “RUNNING”. Job run baru yang dimulai ulang akan memiliki ID job run sebagai gabungan dari ID job run awal ditambah awalan “restart_” yang mewakili jumlah restart. Misalnya, jika ID job run awal adalah`jr_1234`, maka job run yang dimulai ulang akan memiliki ID `jr1234_restart_1` untuk restart pertama. Restart kedua adalah `jr1234_restart_2` untuk restart kedua dan seterusnya.

Upaya coba lagi Anda tidak akan terpengaruh karena restart. Jika proses gagal dan proses baru dimulai karena percobaan ulang otomatis, penghitung restart akan dimulai dari 1 lagi. Misalnya, jika proses gagal`jr_1234_attempt_3_restart_5`, maka percobaan ulang otomatis akan memulai proses baru dengan ID: `jr_id1_attempt_4` dan ketika upaya ini dimulai ulang setelah 7 hari, ID run baru akan menjadi. `jr_id1_attempt_4_restart_1`

Penanganan kehilangan data

Selama pemeliharaan dimulai ulang, AWS Glue Streaming mengikuti proses yang memastikan integritas dan konsistensi data antara pekerjaan sebelumnya dan menjalankan pekerjaan yang dimulai ulang. Perhatikan bahwa AWS Glue tidak menjamin integritas dan konsistensi data antara restart pekerjaan dan kami merekomendasikan pertimbangan arsitektur untuk menangani data duplikat dalam pekerjaan streaming.

1. Mendeteksi kondisi restart pemeliharaan: AWS Glue Streaming memantau kondisi yang menunjukkan kapan restart pemeliharaan harus dipicu, seperti ketika jendela pemeliharaan tercapai setelah 7 hari atau hard restart diperlukan setelah 14 hari.
2. Memanggil penghentian yang anggun: Ketika kondisi restart pemeliharaan terpenuhi, AWS Glue Streaming memulai proses penghentian yang anggun untuk pekerjaan yang sedang berjalan. Proses ini melibatkan langkah-langkah berikut:
   a. Menghentikan konsumsi data baru: Pekerjaan streaming berhenti mengkonsumsi data baru dari sumber input (misalnya, topik Kafka, aliran Kinesis, atau file).
   b. Memproses data yang tertunda: Pekerjaan terus memproses data apa pun yang sudah ada dalam buffer atau antrian internalnya.
   c. Melakukan offset dan pos pemeriksaan: Pekerjaan melakukan offset atau pos pemeriksaan terbaru ke sistem eksternal (misalnya, Kafka, Kinesis, atau Amazon S3) untuk memastikan bahwa pekerjaan yang dimulai ulang dapat diambil dari tempat pekerjaan sebelumnya ditinggalkan.
3. Memulai ulang pekerjaan: Setelah proses penghentian yang anggun selesai, AWS Glue Streaming memulai ulang pekerjaan menggunakan status dan pos pemeriksaan yang diawetkan. Pekerjaan

yang dimulai ulang mengambil pemrosesan dari offset atau pos pemeriksaan terakhir yang dilakukan, memastikan bahwa tidak ada data yang hilang atau diduplikasi.

4. Melanjutkan pemrosesan data: Pekerjaan yang dimulai ulang melanjutkan pemrosesan data dari titik di mana pekerjaan sebelumnya ditinggalkan. Ini terus menelan data baru dari sumber input, mulai dari offset atau pos pemeriksaan terakhir yang dilakukan, dan memproses data sesuai dengan logika ETL yang ditentukan.

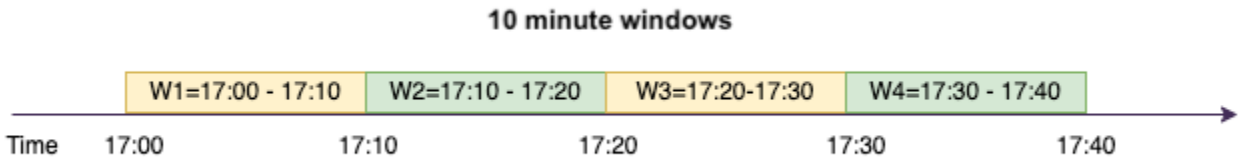
Konsep AWS Glue streaming tingkat lanjut

Dalam aplikasi berbasis data kontemporer, signifikansi data berkurang dari waktu ke waktu dan transisi nilainya dari prediktif menjadi reaktif. Akibatnya, pelanggan ingin memproses data secara real-time untuk membuat keputusan yang lebih cepat. Saat berhadapan dengan umpan data real-time, seperti dari sensor IoT, data mungkin tiba tanpa urutan atau mengalami penundaan pemrosesan karena latensi jaringan dan kegagalan terkait sumber lainnya selama konsumsi. Sebagai bagian dari AWS Glue platform, AWS Glue Streaming dibangun di atas kemampuan ini untuk menyediakan ETL streaming tanpa server yang dapat diskalakan, didukung oleh streaming terstruktur Apache Spark, memberdayakan pengguna dengan pemrosesan data waktu nyata.

Dalam topik ini, kita akan mengeksplorasi konsep streaming canggih dan kemampuan AWS Glue Streaming.

Pertimbangan waktu saat memproses aliran

Ada empat pengertian waktu saat memproses aliran:



- Event-time — Waktu di mana peristiwa itu terjadi. Dalam kebanyakan kasus, bidang ini disematkan ke dalam data-peristiwa itu sendiri, di sumbernya.
- E vent-time-window - Kerangka waktu antara dua waktu peristiwa. Seperti yang ditunjukkan pada diagram di atas, W1 adalah event-time-window dari 17:00 hingga 17:10. Masing-masing event-time-window adalah pengelompokan beberapa peristiwa.
- Trigger-time — Waktu pemicu mengontrol seberapa sering pemrosesan data dan pemutakhiran hasil terjadi. Ini adalah waktu ketika pemrosesan batch mikro dimulai.

- Waktu konsumsi — Waktu ketika data-aliran dicerna ke dalam layanan streaming. Jika waktu acara tidak disematkan ke dalam acara itu sendiri, kali ini dapat digunakan untuk windowing dalam beberapa kasus.

Jendela

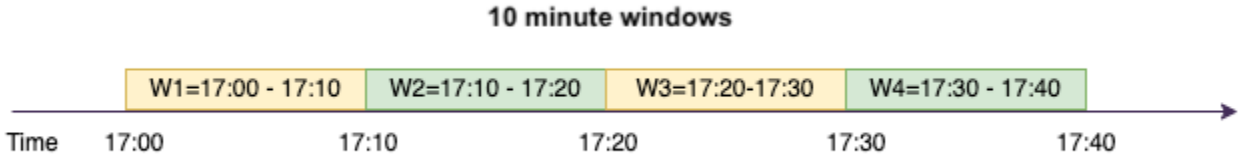
Windowing adalah teknik di mana Anda mengelompokkan dan menggabungkan beberapa peristiwa dengan. event-time-window Kami akan mengeksplorasi manfaat windowing dan kapan Anda akan menggunakannya dalam contoh berikut.

Tergantung pada kasus penggunaan bisnis, ada tiga jenis jendela waktu yang didukung oleh percikan.

- Tumbling window — serangkaian ukuran event-time-windows tetap yang tidak tumpang tindih di mana Anda agregat.
- Jendela geser — mirip dengan jendela yang jatuh dari titik “berukuran tetap”, tetapi jendela dapat tumpang tindih atau meluncur selama durasi slide lebih kecil dari durasi jendela itu sendiri.
- Jendela sesi — dimulai dengan peristiwa data input dan terus memperluas dirinya sendiri selama menerima input dalam celah atau durasi tidak aktif. Jendela sesi dapat memiliki ukuran statis atau dinamis dari panjang jendela, tergantung pada input.

Jendela tumbling

Tumbling window adalah serangkaian ukuran event-time-windows tetap yang tidak tumpang tindih yang Anda agregat. Mari kita pahami ini dengan contoh dunia nyata.



Perusahaan ABC Auto ingin melakukan kampanye pemasaran untuk merek baru mobil sport. Mereka ingin memilih kota di mana mereka memiliki penggemar mobil sport terbesar. Untuk mencapai tujuan ini, mereka menampilkan iklan pendek 15 detik yang memperkenalkan mobil di situs web mereka. Semua “klik “dan” kota “yang sesuai direkam dan dialirkan ke. Amazon Kinesis Data Streams Kami ingin menghitung jumlah klik dalam jendela 10 menit dan mengelompokkannya berdasarkan kota untuk melihat kota mana yang memiliki permintaan tertinggi. Berikut ini adalah output dari agregasi.

window_start_time	window_end_time	kota	total_clicks
2023-07-10 17:00:00	2023-07-10 17:10:00	Dallas	75
2023-07-10 17:00:00	2023-07-10 17:10:00	Chicago	10
2023-07-10 17:20:00	2023-07-10 17:30:00	Dallas	20
2023-07-10 17:20:00	2023-07-10 17:30:00	Chicago	50

Seperti dijelaskan di atas, ini event-time-windows berbeda dari interval waktu pemicu. Misalnya, bahkan jika waktu pemicu Anda setiap menit, hasil output hanya akan menampilkan jendela agregasi 10 menit yang tidak tumpang tindih. Untuk pengoptimalan, lebih baik agar interval pemicu selaras dengan. event-time-window

Pada tabel di atas, Dallas melihat 75 klik di jendela 17:00-17:10, sementara Chicago memiliki 10 klik. Juga, tidak ada data untuk jendela 17:10 - 17:20 untuk kota mana pun, jadi jendela ini dihilangkan.

Sekarang Anda dapat menjalankan analisis lebih lanjut tentang data ini di aplikasi analitik hilir untuk menentukan kota paling eksklusif untuk menjalankan kampanye pemasaran.

Menggunakan jendela jatuh di AWS Glue

1. Buat Amazon Kinesis Data Streams DataFrame dan baca darinya. Contoh:

```
parsed_df = kinesis_raw_df \
                .selectExpr('CAST(data AS STRING)') \
                .select(from_json("data", ticker_schema).alias("data")) \
                .select('data.event_time','data.ticker','data.trade','data.volume',
 'data.price')
```

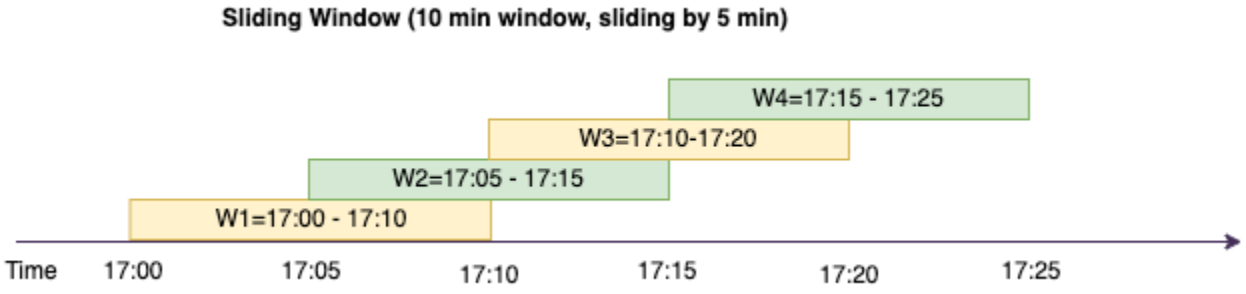
2. Memproses data di jendela yang jatuh. Dalam contoh di bawah ini, data dikelompokkan berdasarkan bidang input “event_time” dalam jendela tumbling 10 menit dan menulis output ke danau data Amazon S3.

```
grouped_df = parsed_df \
                .groupBy(window("event_time", "10 minutes"), "city") \
                .agg(sum("clicks").alias("total_clicks"))
```

```
summary_df = grouped_df \
                .withColumn("window_start_time", col("window.start")) \
                .withColumn("window_end_time", col("window.end")) \
                .withColumn("year", year("window_start_time")) \
                .withColumn("month", month("window_start_time")) \
                .withColumn("day", dayofmonth("window_start_time")) \
                .withColumn("hour", hour("window_start_time")) \
                .withColumn("minute", minute("window_start_time")) \
                .drop("window")

write_result = summary_df \
                .writeStream \
                .format("parquet") \
                .trigger(processingTime="10 seconds") \
                .option("checkpointLocation", "s3a://bucket-stock-stream/stock-
stream-catalog-job/checkpoint/") \
                .option("path", "s3a://bucket-stock-stream/stock-stream-catalog-
job/summary_output/") \
                .partitionBy("year", "month", "day") \
                .start()
```

Jendela geser

Jendela geser mirip dengan jendela yang jatuh dari titik “berukuran tetap”, tetapi jendela dapat tumpang tindih atau meluncur selama durasi slide lebih kecil dari durasi jendela itu sendiri. Karena sifat geser, input dapat terikat ke beberapa jendela.



Untuk lebih memahami, mari kita perhatikan contoh bank yang ingin mendeteksi potensi penipuan kartu kredit. Aplikasi streaming dapat memantau aliran transaksi kartu kredit yang berkelanjutan. Transaksi ini dapat digabungkan ke dalam jendela dengan durasi 10 menit dan setiap 5 menit, jendela akan meluncur ke depan, menghilangkan data 5 menit tertua dan menambahkan 5 menit data baru terbaru. Dalam setiap jendela, transaksi dapat dikelompokkan berdasarkan negara yang

memeriksa pola yang mencurigakan, seperti transaksi di AS segera diikuti oleh yang lain di Australia. Untuk mempermudah, mari kita mengkategorikan transaksi tersebut sebagai penipuan ketika jumlah total transaksi lebih besar dari $100. Jika pola seperti itu terdeteksi, itu menandakan potensi penipuan dan kartu bisa dibekukan.

Sistem pemrosesan kartu kredit mengirimkan uap peristiwa transaksi ke kinesis untuk setiap card-id bersama dengan negara. AWS Glue Pekerjaan menjalankan analisis dan menghasilkan output agregat berikut.

window_start_time	window_end_time	card_last_four	negeri	total_amount
2023-07-10 17:00:00	2023-07-10 17:10:00	6544	AS	85
2023-07-10 17:00:00	2023-07-10 17:10:00	6544	Australia	10
2023-07-10 17:05:45	2023-07-10 17:15:45	6544	AS	50
2023-07-10 17:10:45	2023-07-10 17:20:45	6544	AS	50
2023-07-10 17:10:45	2023-07-10 17:20:45	6544	Australia	150

Berdasarkan agregasi di atas, Anda dapat melihat jendela 10 menit meluncur setiap 5 menit, dijumlahkan dengan jumlah transaksi. Anomali terdeteksi di jendela 17:10 - 17:20 di mana ada outlier, yang merupakan transaksi seharga $150 di Australia. AWS Gluedapat mendeteksi anomali ini dan mendorong peristiwa alarm dengan kunci yang menyinggung ke topik SNS menggunakan boto3. Selanjutnya fungsi Lambda dapat berlangganan topik ini dan mengambil tindakan.

Memproses data di jendela geser

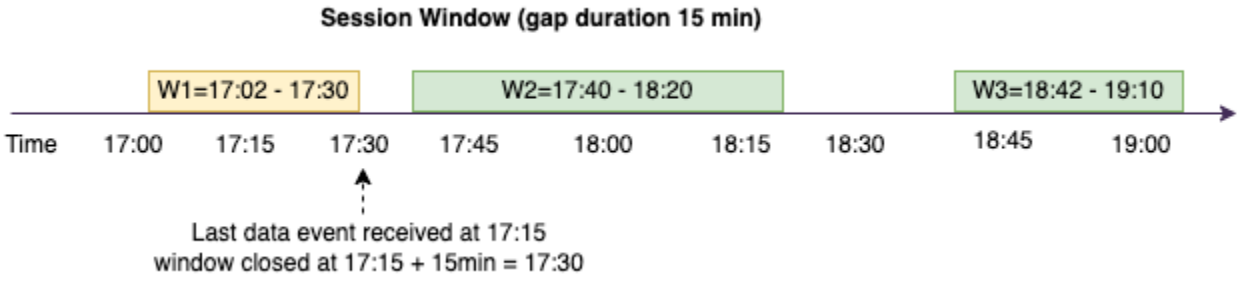
`group-by`Klausa dan fungsi jendela digunakan untuk mengimplementasikan jendela geser seperti yang ditunjukkan di bawah ini.

```
grouped_df = parsed_df \
```

```
            .groupBy(window(col("event_time"), "10 minute", "5 min"), "country",
"card_last_four") \
            .agg(sum("tx_amount").alias("total_amount"))
```

Jendela sesi

Berbeda dengan dua jendela di atas yang memiliki ukuran tetap, jendela sesi dapat memiliki ukuran statis atau dinamis dari panjang jendela, tergantung pada input. Jendela sesi dimulai dengan peristiwa data input dan terus memperluas dirinya sendiri selama menerima input dalam celah atau durasi tidak aktif.



Mari kita ambil contoh. Perusahaan ABC hotel ingin mencari tahu kapan waktu tersibuk dalam seminggu dan memberikan penawaran yang lebih baik untuk tamu mereka. Segera setelah tamu check-in, jendela sesi dimulai dan spark mempertahankan status dengan agregasi untuk itu. event-time-window Setiap kali tamu check-in, acara dibuat dan dikirim ke Amazon Kinesis Data Streams. Hotel membuat keputusan bahwa jika tidak ada check-in untuk jangka waktu 15 menit, event-time-window dapat ditutup. Selanjutnya event-time-window akan dimulai lagi ketika ada check-in baru. Outputnya terlihat sebagai berikut.

window_start_time	window_end_time	kota	total_checkins
2023-07-10 17:02:00	2023-07-10 17:30:00	Dallas	50
2023-07-10 17:02:00	2023-07-10 17:30:00	Chicago	25
2023-07-10 17:40:00	2023-07-10 18:20:00	Dallas	75
2023-07-10 18:50:45	2023-07-10 19:15:45	Dallas	20

Check-in pertama terjadi pada event_time= 17:02. Agregasi event-time-window akan dimulai pada 17:02. Agregasi ini akan berlanjut selama kami menerima acara dalam durasi 15 menit. Dalam contoh di atas, acara terakhir yang kami terima adalah pukul 17:15 dan kemudian selama 15 menit berikutnya tidak ada acara. Akibatnya, Spark menutupnya event-time-window pada 17:15 +15 menit = 17:30 dan mengaturnya sebagai 17:02 - 17:30. Ini memulai yang baru event-time-window pada 17:47 ketika menerima acara data check-in baru.

Memproses data di jendela sesi

`group-by`Klausa dan fungsi jendela digunakan untuk mengimplementasikan jendela geser.

```
grouped_df = parsed_df \
            .groupBy(session_window(col("event_time"), "10 minute"), "city") \
            .agg(count("check_in").alias("total_checkins"))
```

Mode keluaran

Mode output adalah mode di mana hasil dari tabel tak terbatas ditulis ke wastafel eksternal. Ada tiga mode yang tersedia. Dalam contoh berikut, Anda menghitung kemunculan kata saat baris data sedang dialirkan dan diproses di setiap batch mikro.

- Mode lengkap - Seluruh tabel hasil akan ditulis ke wastafel setelah setiap pemrosesan batch mikro meskipun jumlah kata tidak diperbarui saat ini event-time-window.
- Mode Append — Ini adalah mode default, di mana hanya kata-kata dan atau baris baru yang ditambahkan ke tabel hasil karena pemicu terakhir akan ditulis ke wastafel. Mode ini bagus untuk streaming stateless untuk kueri seperti peta, FlatMap, filter, dll.
- Mode pembaruan - Hanya kata dan atau baris di Tabel Hasil yang diperbarui atau ditambahkan sejak pemicu terakhir yang akan ditulis ke wastafel.

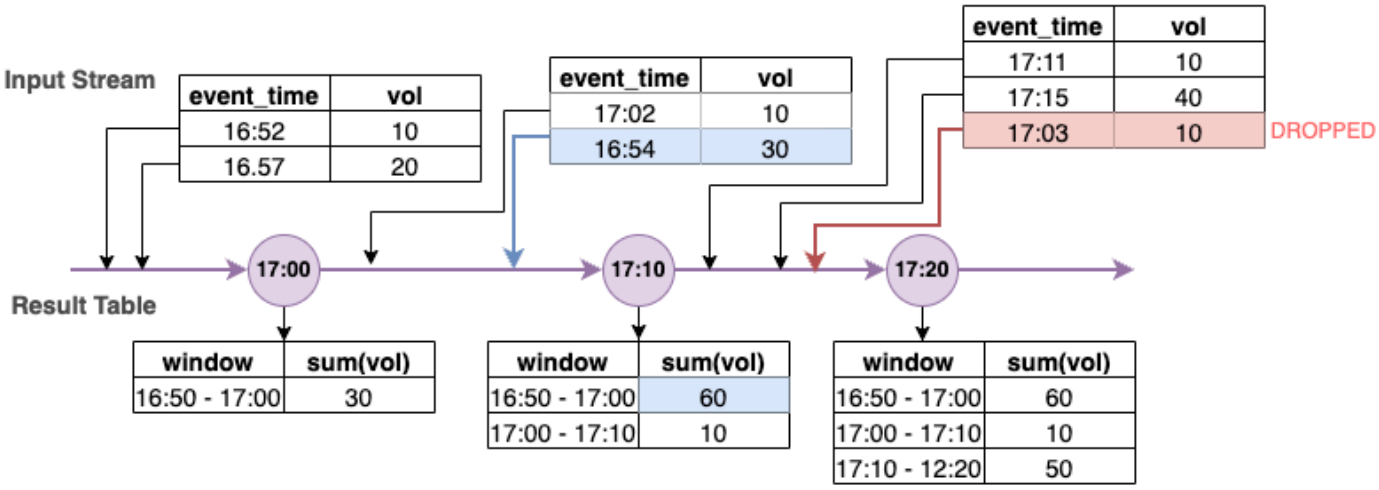
> **Note**
>
> Mode keluaran = “pembaruan” tidak didukung untuk jendela sesi.

Menangani data dan tanda air yang terlambat

Saat bekerja dengan data real-time mungkin ada penundaan kedatangan data karena latensi jaringan dan kegagalan hulu dan kami memerlukan mekanisme untuk melakukan agregasi lagi pada yang

terlewat. event-time-window Namun, untuk melakukan ini, negara perlu dipertahankan. Pada saat yang sama, data yang lebih lama perlu dibersihkan untuk membatasi ukuran negara. Spark versi 2.1 menambahkan dukungan untuk fitur yang disebut watermarking yang mempertahankan status dan memungkinkan pengguna untuk menentukan ambang batas untuk data terlambat.

Dengan mengacu pada contoh ticker saham kami di atas, mari pertimbangkan ambang batas yang diizinkan untuk data terlambat sebagai tidak lebih dari 10 menit. Untuk membuatnya tetap sederhana, kita akan menganggap jendela jatuh, ticker sebagai AMZ, diperdagangkan sebagai BELI.



Dalam diagram di atas, kita menghitung total volume di atas jendela 10 menit yang jatuh. Kami memiliki pemicu pada 17:00, 17:10 dan 17:20. Di atas panah timeline, kami memiliki aliran data input dan di bawah ini adalah tabel hasil tak terbatas.

Di jendela jatuh 10 menit pertama kami agregasi berdasarkan event_time dan total_volume dihitung sebagai 30. Yang kedua event-time-window, spark mendapatkan peristiwa data pertama dengan event_time= 17:02. Karena ini adalah event_time maks yang terlihat sejauh ini oleh percikan, ambang batas tanda air disetel 10 menit yang lalu (yaitu, watermark_event_time= 16:52). Setiap peristiwa data dengan event_time setelah 16:52 akan dipertimbangkan untuk agregasi terikat waktu dan peristiwa data apa pun sebelum itu akan dihapus. Hal ini memungkinkan percikan untuk mempertahankan status perantara selama 10 menit tambahan untuk mengakomodasi data yang terlambat. Sekitar waktu jam dinding 17:08 Spark menerima acara dengan event_time= 16:54 yang berada dalam ambang batas. Oleh karena itu spark menghitung ulang “16:50 - 17:00 “ event-time-window dan total volume diperbarui dari 30 menjadi 60.

Namun, pada waktu pemicu 17:20, ketika spark menerima acara dengan event_time= 17:15 itu mengatur watermark_event_time= 17:05. Oleh karena itu peristiwa data terlambat dengan event_time= 17:03 dianggap “terlambat” dan diabaikan.

```
Watermark Boundary = Max(Event Time) - Watermark Threshold
```

Menggunakan tanda air di AWS Glue

Spark tidak akan memancarkan atau menulis data ke wastafel eksternal sampai batas watermark dilewati. Untuk menerapkan tanda airAWS Glue, lihat contoh di bawah ini.

```
grouped_df = parsed_df \
                .withWatermark("event_time", "10 minutes") \
                .groupBy(window("event_time", "5 minutes"), "ticker") \
                .agg(sum("volume").alias("total_volume"))
```

Memantau pekerjaan AWS Glue streaming

Memantau pekerjaan streaming Anda adalah bagian penting dalam membangun saluran ETL Anda. Selain menggunakan UI Spark, Anda juga dapat menggunakan Amazon CloudWatch untuk memantau metrik. Di bawah ini adalah daftar metrik streaming yang dipancarkan oleh kerangka kerja. AWS Glue Untuk daftar lengkap semua AWS Glue metrik, lihat Memantau AWS Glue menggunakan CloudWatch metrik Amazon.

AWS Gluemenggunakan kerangka kerja streaming terstruktur untuk memproses peristiwa input. Anda dapat menggunakan Spark API secara langsung dalam kode Anda atau memanfaatkan yang `ForEachBatch` disediakan oleh`GlueContext`, yang menerbitkan metrik ini. Untuk memahami metrik ini, kita harus terlebih dahulu memahami`windowSize`.

WindowSize: `windowSize` adalah interval batch mikro yang Anda berikan. Jika Anda menentukan ukuran jendela 60 detik, pekerjaan AWS Glue streaming akan menunggu selama 60 detik (atau lebih jika batch sebelumnya belum selesai saat itu) sebelum akan membaca data dalam batch dari sumber streaming dan menerapkan transformasi yang disediakan di`ForEachBatch`. Ini juga disebut sebagai interval pemicu.

Mari kita tinjau metrik secara lebih rinci untuk memahami karakteristik kesehatan dan kinerja.

> **Note**
>
> Metrik dipancarkan setiap 30 detik. Jika Anda `windowSize` kurang dari 30 detik maka metrik yang dilaporkan adalah agregasi. Misalnya katakanlah Anda `windowSize` 10 detik dan

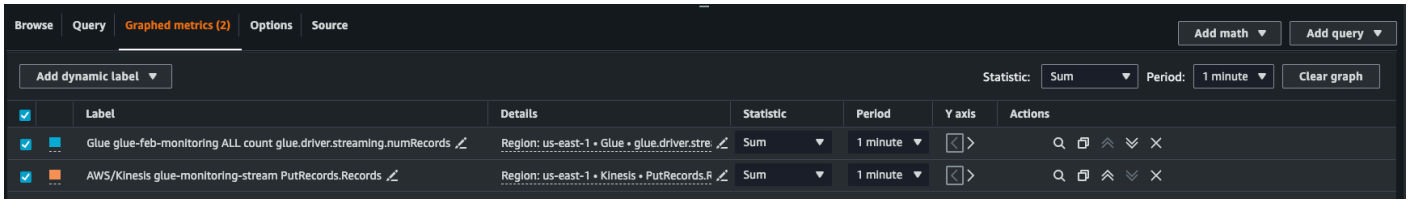
Anda terus memproses 20 catatan per batch mikro. Dalam skenario ini, nilai metrik yang dipancarkan untuk NumRecords adalah 60.
Metrik tidak dipancarkan jika tidak ada data yang tersedia untuk itu. Juga, dalam kasus metrik lag konsumen, Anda harus mengaktifkan fitur untuk mendapatkan metrik untuk itu.

Memvisualisasikan metrik

Untuk memplot metrik visual:

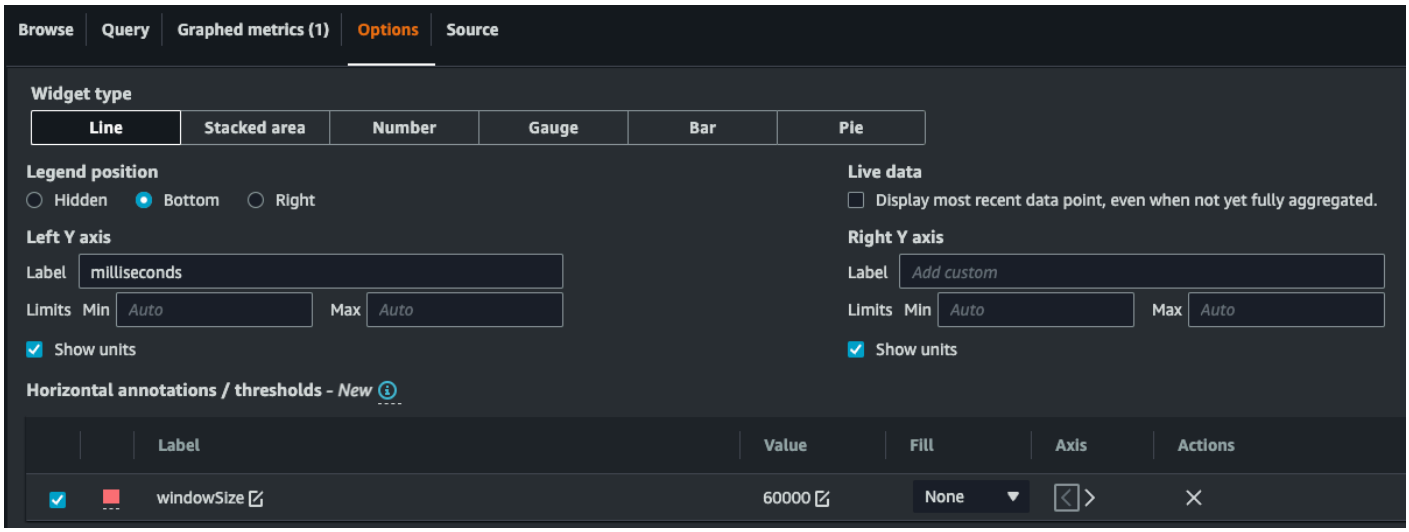
1. Buka Metrik di CloudWatch konsol Amazon dan kemudian pilih tab Browse. Kemudian pilih Glue di bawah “Ruang nama khusus”.

2. Pilih Job Metrics untuk menampilkan metrik untuk semua pekerjaan Anda.
3. Filter metrik berdasarkan JobName = Anda glue-feb-monitoring dan kemudian JobRunId = SEMUA. Anda dapat mengklik tanda “+” seperti yang ditunjukkan pada gambar di bawah ini untuk menambahkannya ke filter pencarian.
4. Pilih kotak centang untuk metrik yang Anda minati. Pada gambar di bawah ini kami telah memilih `numberAllExecutors` dan`numberMaxNeededExecutors`.

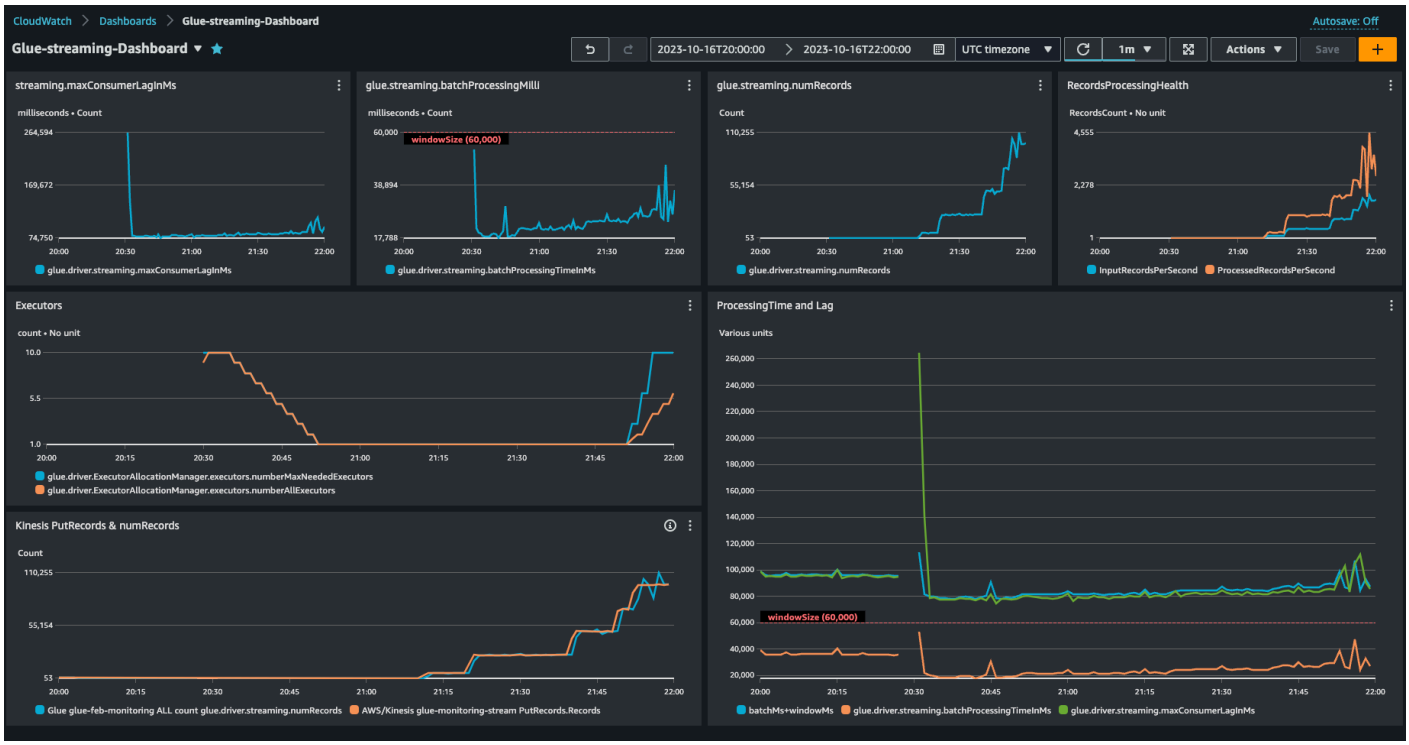


5. Setelah Anda memilih metrik ini, Anda dapat pergi ke tab Metrik grafik dan menerapkan statistik Anda.
6. Karena metrik dipancarkan setiap menit, Anda dapat menerapkan “rata-rata” selama satu menit untuk dan. `batchProcessingTimeInMs` `maxConsumerLagInMs` Untuk `numRecords` Anda dapat menerapkan “jumlah” setiap menit.

7. Anda dapat menambahkan `windowSize` anotasi horizontal ke grafik Anda menggunakan tab Opsi.



8. Setelah metrik dipilih, buat dasbor dan tambahkan. Berikut adalah contoh dasbor.

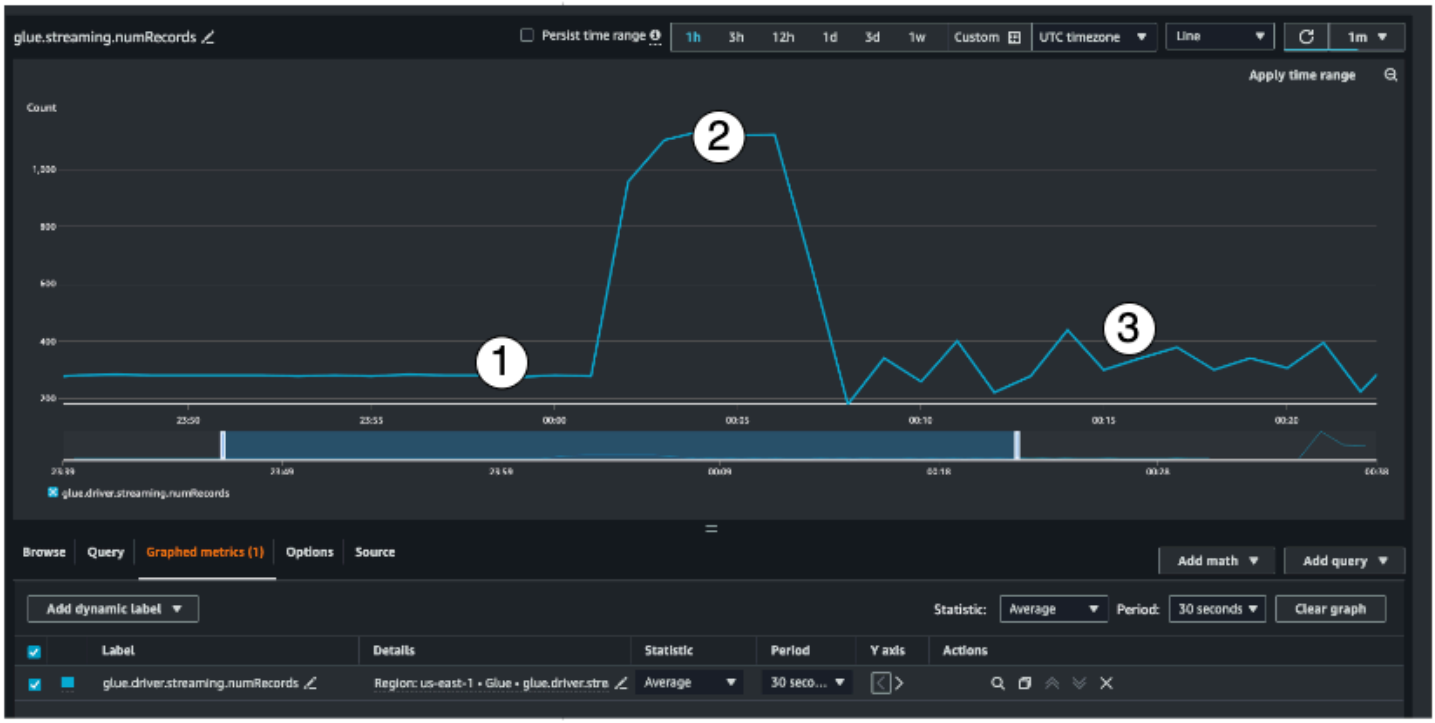


Metrik penyelaman mendalam

Bagian ini menjelaskan masing-masing metrik dan bagaimana mereka saling berhubungan satu sama lain.

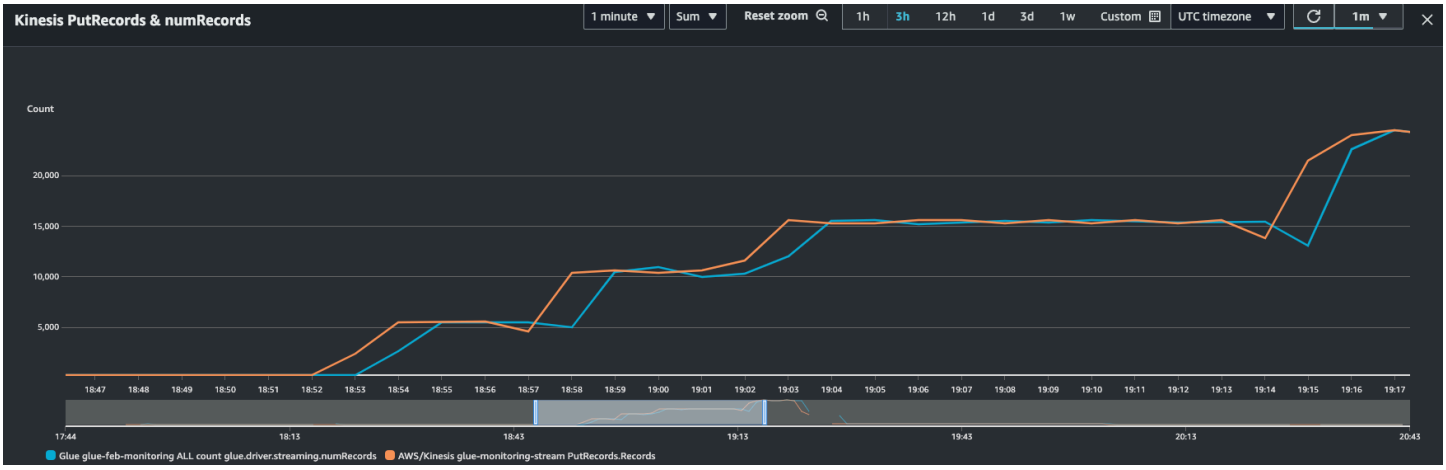
Jumlah catatan (metrik: streaming.numRecords)

Metrik ini menunjukkan berapa banyak catatan yang sedang diproses.



Metrik streaming ini memberikan visibilitas ke dalam jumlah catatan yang Anda proses di jendela. Seiring dengan jumlah catatan yang diproses, ini juga akan membantu Anda memahami perilaku lalu lintas input.

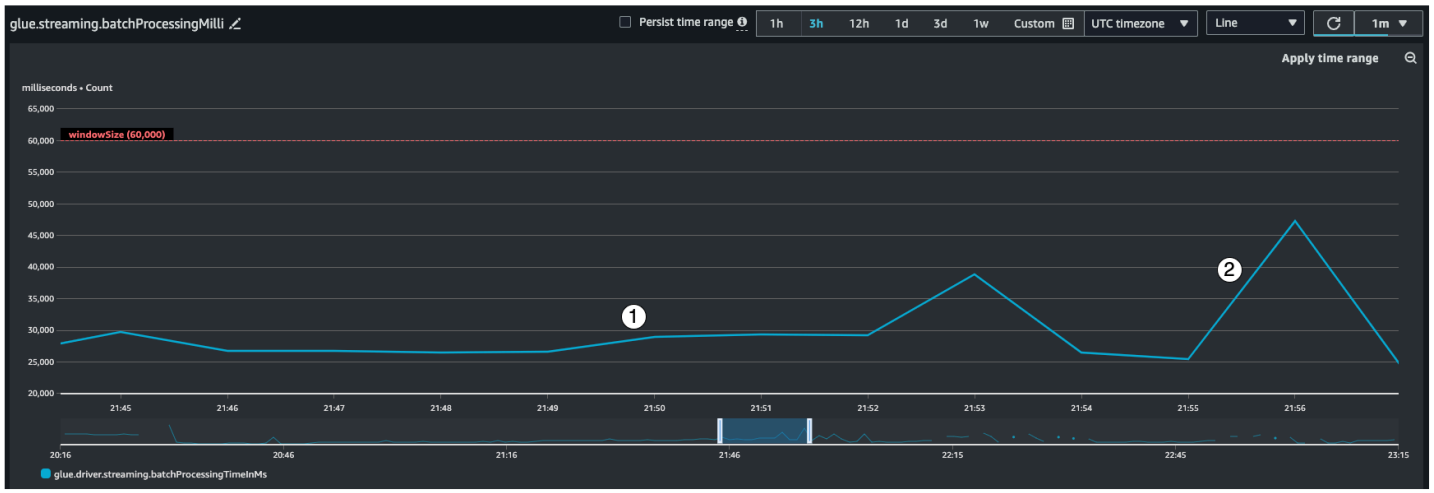
- Indikator #1 menunjukkan contoh lalu lintas stabil tanpa semburan apapun. Biasanya ini akan menjadi aplikasi seperti sensor IoT yang mengumpulkan data secara berkala dan mengirimkannya ke sumber streaming.
- Indikator #2 menunjukkan contoh ledakan lalu lintas yang tiba-tiba pada beban yang stabil. Ini bisa terjadi di aplikasi clickstream ketika ada acara pemasaran seperti Black Friday dan ada ledakan dalam jumlah klik
- Indikator #3 menunjukkan contoh lalu lintas yang tidak dapat diprediksi. Lalu lintas yang tidak dapat diprediksi tidak berarti ada masalah. Itu hanya sifat dari data input. Kembali ke contoh sensor IoT, Anda dapat memikirkan ratusan sensor yang mengirimkan peristiwa perubahan cuaca ke sumber streaming. Karena perubahan cuaca tidak dapat diprediksi, begitu pula datanya. Memahami pola lalu lintas adalah kunci untuk mengukur pelaksana Anda. Jika inputnya sangat runcing, Anda dapat mempertimbangkan untuk menggunakan penskalaan otomatis (lebih lanjut tentang itu nanti).



Anda dapat menggabungkan metrik ini dengan PutRecords metrik Kinesis untuk memastikan jumlah peristiwa yang dicerna dan jumlah catatan yang dibaca hampir sama. Ini sangat berguna ketika Anda mencoba memahami lag. Saat tingkat konsumsi meningkat, begitu juga dengan `numRecords` membaca. AWS Glue

Waktu pemrosesan batch (metrik: streaming. batchProcessingTimeInMs)

Metrik waktu pemrosesan batch membantu Anda menentukan apakah klaster tidak tersedia atau dilebih-lebihkan.

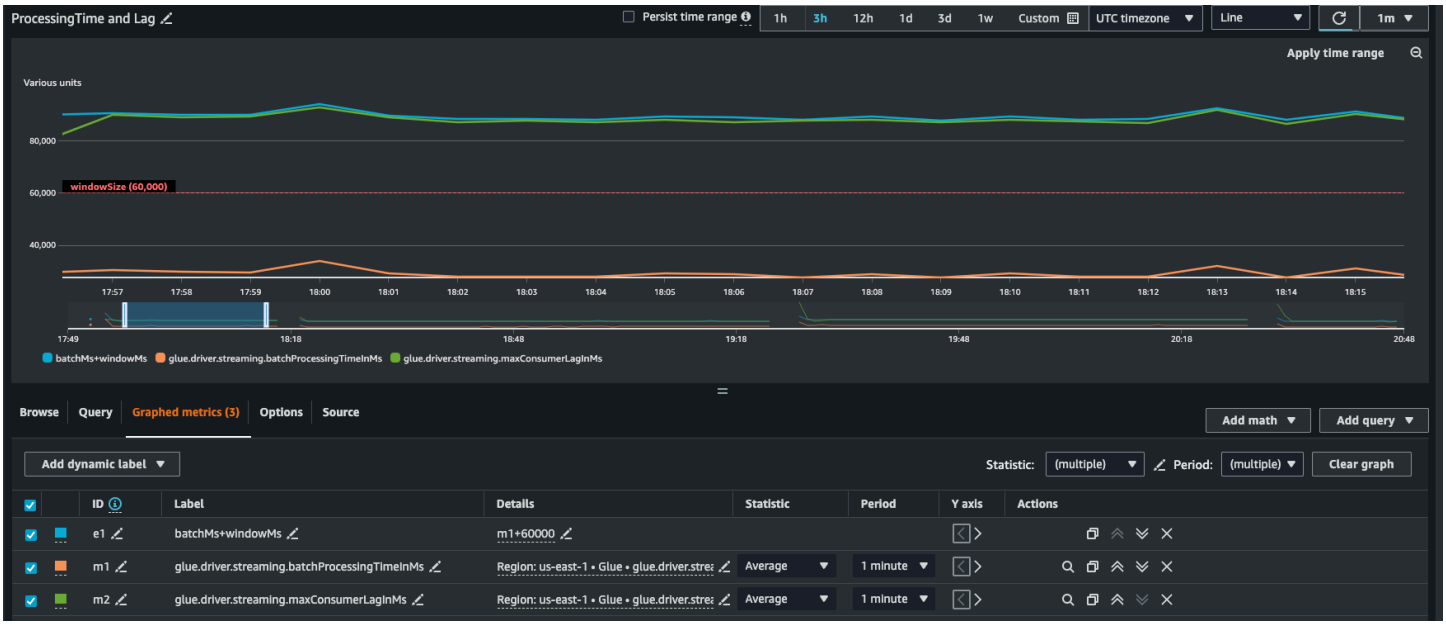


Metrik ini menunjukkan jumlah milidetik yang diperlukan untuk memproses setiap kumpulan catatan mikro. Tujuan utama di sini adalah untuk memantau waktu ini untuk memastikannya kurang dari `windowSize` interval. Tidak apa-apa jika `batchProcessingTimeInMs` berjalan sementara selama pulih dalam interval jendela berikut. Indikator #1 menunjukkan waktu yang kurang lebih stabil yang dibutuhkan untuk memproses pekerjaan. Namun jika jumlah catatan masukan meningkat, waktu yang dibutuhkan untuk memproses pekerjaan akan meningkat serta ditunjukkan oleh indikator

#2. Jika `numRecords` tidak naik, tetapi waktu pemrosesan naik, maka Anda perlu melihat lebih dalam pemrosesan pekerjaan pada pelaksana. Ini adalah praktik yang baik untuk menetapkan ambang batas dan alarm untuk memastikan `batchProcessingTimeInMs` tidak tinggal lebih dari 120% selama lebih dari 10 menit. Untuk informasi selengkapnya tentang menyetel alarm, lihat Menggunakan CloudWatch alarm Amazon.

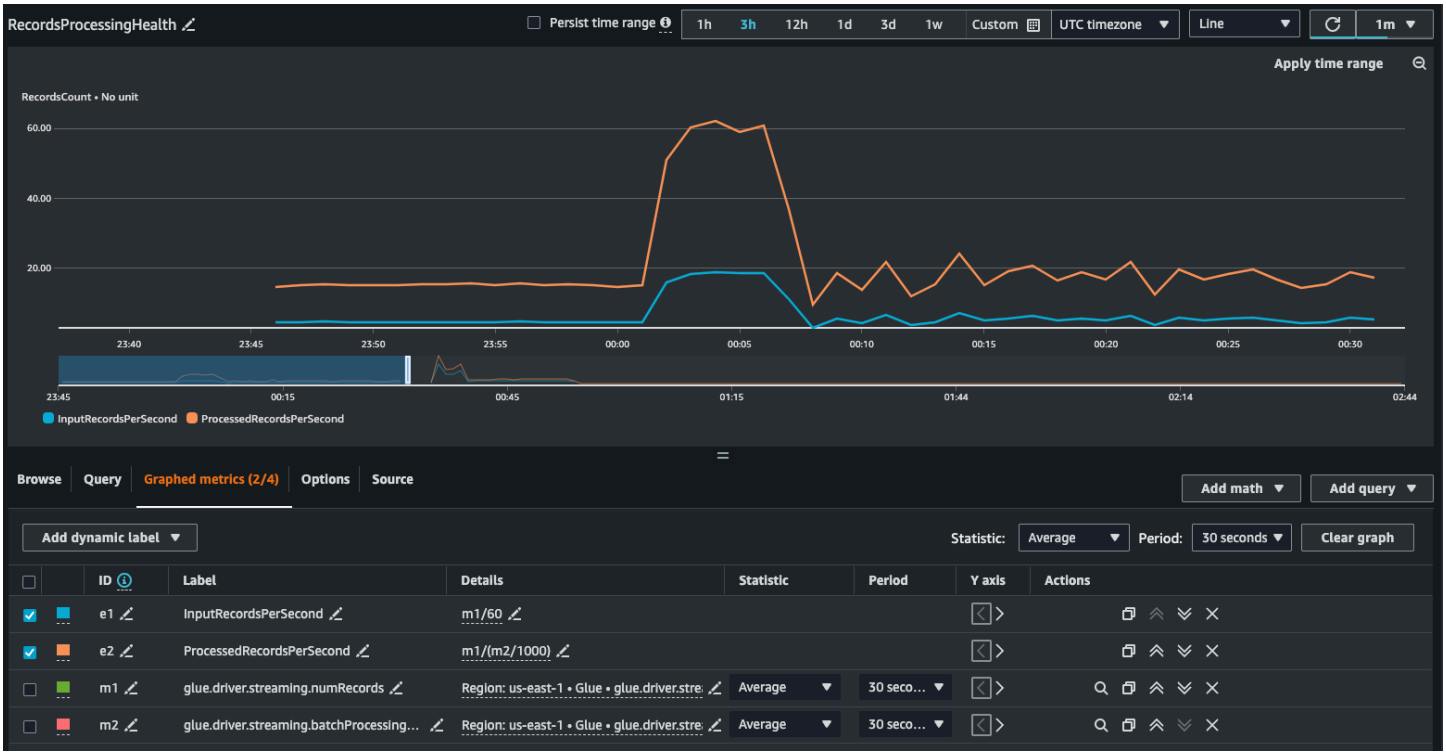
Kelambatan konsumen (metrik: streaming. maxConsumerLagInMs)

Metrik lag konsumen membantu Anda memahami jika ada kelambatan dalam memproses peristiwa. Jika lag Anda terlalu tinggi, maka Anda bisa melewatkan pemrosesan SLA yang bergantung pada bisnis Anda, meskipun Anda memiliki WindowSize yang benar. Anda harus mengaktifkan metrik ini secara eksplisit menggunakan opsi koneksi. `emitConsumerLagMetrics` Untuk informasi lebih lanjut, lihat KinesisStreamingSourceOptions.



Metrik turunan

Untuk mendapatkan wawasan yang lebih dalam, Anda dapat membuat metrik turunan untuk memahami lebih lanjut tentang pekerjaan streaming Anda di Amazon. CloudWatch



Anda dapat membuat grafik dengan metrik turunan untuk memutuskan apakah Anda perlu menggunakan lebih banyak DPU. Meskipun penskalaan otomatis membantu Anda melakukan ini secara otomatis, Anda dapat menggunakan metrik turunan untuk menentukan apakah penskalaan otomatis bekerja secara efektif.

- InputRecordsPerSecondmenunjukkan tingkat di mana Anda mendapatkan catatan input. Hal ini diturunkan sebagai berikut: jumlah catatan input (Glue.driver.streaming.numRecords)/. WindowSize
- ProcessingRecordsPerSecondmenunjukkan tingkat di mana catatan Anda sedang diproses. Hal ini diturunkan sebagai berikut: jumlah catatan input (Glue.driver.streaming.numRecords)/. batchProcessingTime InMs

Jika tingkat input lebih tinggi dari tingkat pemrosesan, maka Anda mungkin perlu menambahkan lebih banyak kapasitas untuk memproses pekerjaan Anda atau meningkatkan paralelisme.

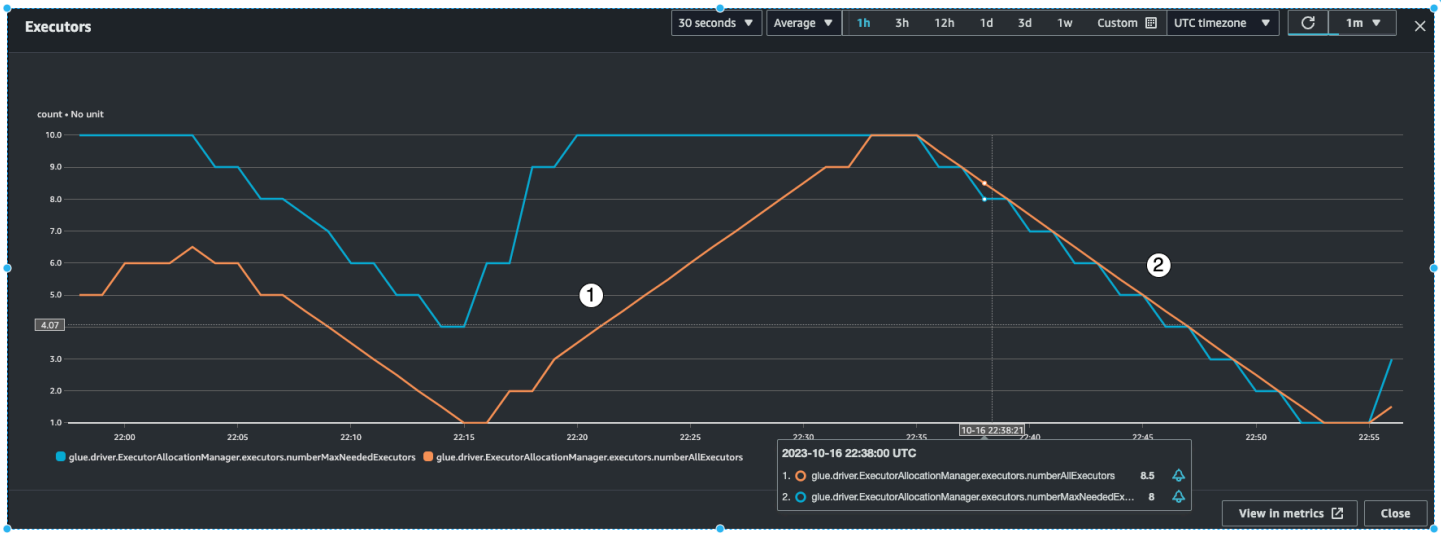
Metrik penskalaan otomatis

Ketika lalu lintas input Anda runcing, maka Anda harus mempertimbangkan untuk mengaktifkan penskalaan otomatis dan menentukan pekerja maks. Dengan itu Anda mendapatkan dua metrik tambahan, `numberAllExecutors` dan`numberMaxNeededExecutors`.

- numberAllExecutorsadalah jumlah pelaksana pekerjaan yang aktif menjalankan

- numberMaxNeededPelaksana adalah jumlah pelaksana pekerjaan maksimum (aktif berjalan dan tertunda) yang diperlukan untuk memenuhi beban saat ini.

Kedua metrik ini akan membantu Anda memahami apakah penskalaan otomatis Anda berfungsi dengan benar.



AWS Glueakan memantau `batchProcessingTimeInMs` metrik melalui beberapa batch mikro dan melakukan salah satu dari dua hal. Ini akan meningkatkan skala pelaksana, jika `batchProcessingTimeInMs` lebih dekat ke, atau skala-di pelaksana`windowSize`, jika relatif lebih rendah dari. `batchProcessingTimeInMs` `windowSize` Juga, itu akan menggunakan algoritma untuk langkah-scaling pelaksana.

- indikator #1 menunjukkan kepada Anda bagaimana pelaksana aktif ditingkatkan untuk mengejar ketinggalan dengan eksekutor maksimal yang dibutuhkan untuk memproses beban.
- indikator #2 menunjukkan kepada Anda bagaimana eksekutif aktif menskalakan sejak rendah`batchProcessingTimeInMs`.

Anda dapat menggunakan metrik ini untuk memantau paralelisme tingkat eksekusi saat ini dan menyesuaikan jumlah pekerja maksimal dalam konfigurasi auto-scaling Anda.

Cara mendapatkan kinerja terbaik

Spark akan mencoba membuat satu tugas per pecahan, untuk dibaca dari, di aliran Amazon Kinesis. Data di setiap pecahan menjadi partisi. Ini kemudian akan mendistribusikan tugas-tugas ini di seluruh pelaksana/pekerja, tergantung dari jumlah inti pada setiap pekerja (jumlah inti per pekerja tergantung

pada jenis pekerja yang Anda pilihG.025X,, G.1X dll). Namun tidak deterministik bagaimana tugas didistribusikan. Semua tugas dijalankan secara paralel pada inti masing-masing. Jika ada lebih banyak pecahan daripada jumlah inti pelaksana yang tersedia, tugas akan diantri.

Anda dapat menggunakan kombinasi metrik di atas dan jumlah pecahan, untuk menyediakan eksekutor Anda untuk beban yang stabil dengan beberapa ruang untuk semburan. Disarankan agar Anda menjalankan beberapa iterasi pekerjaan Anda untuk menentukan perkiraan jumlah pekerja. Untuk beban kerja yang tidak stabil/runcing, Anda dapat melakukan hal yang sama dengan menyiapkan penskalaan otomatis dan pekerja maks.

Tetapkan `windowSize` sesuai kebutuhan SLA bisnis Anda. Misalnya, jika bisnis Anda mengharuskan data yang diproses tidak boleh lebih dari 120 detik basi, maka atur `windowSize` ke setidaknya 60 detik sehingga kelambatan konsumen rata-rata Anda kurang dari 120 detik (lihat bagian tentang kelambatan konsumen di atas). Dari sana tergantung pada `numRecords` dan jumlah pecahan, rencanakan kapasitas dalam DPU untuk memastikan Anda `batchProcessingTimeInMs` kurang dari 70% dari `windowSize` sebagian besar waktu Anda.

> **Note**
>
> Pecahan panas dapat menyebabkan kemiringan data yang berarti bahwa beberapa piringan/partisi jauh lebih besar daripada yang lain. Hal ini dapat menyebabkan beberapa tugas yang berjalan secara paralel membutuhkan waktu lebih lama menyebabkan tugas straggler. Akibatnya, batch berikutnya tidak dapat dimulai sampai semua tugas dari yang sebelumnya selesai, ini akan berdampak pada `batchProcessingTimeInMillis` dan lag maksimal.

AWS Glue Kualitas Data

AWS Glue Kualitas Data memungkinkan Anda untuk mengukur dan memantau kualitas data Anda sehingga Anda dapat membuat keputusan bisnis yang baik. Dibangun di atas DeeQu kerangka kerja sumber terbuka, Kualitas AWS Glue Data memberikan pengalaman tanpa server yang dikelola. AWS Glue Kualitas Data bekerja dengan Data Quality Definition Language (DQDL), yang merupakan bahasa khusus domain yang Anda gunakan untuk menentukan aturan kualitas data. Untuk mempelajari lebih lanjut tentang DQDL dan jenis aturan yang didukung, lihat. Referensi Bahasa Definisi Kualitas Data (DQDL)

Untuk detail dan harga produk tambahan, lihat halaman layanan untuk Kualitas AWS Glue Data.

Manfaat dan fitur utama

Manfaat dan fitur utama Kualitas AWS Glue Data meliputi:

- Tanpa server - tidak ada instalasi, penambalan, atau pemeliharaan.
- Memulai dengan cepat — Kualitas AWS Glue Data menganalisis data Anda dengan cepat dan membuat aturan kualitas data untuk Anda. Anda dapat memulai dengan dua klik: “Buat Aturan Kualitas Data → Rekomendasikan aturan”.
- Deteksi masalah kualitas data — Gunakan pembelajaran mesin (ML) untuk mendeteksi anomali dan masalah kualitas hard-to-detect data.
- Improvisasi aturan Anda — dengan 25+ aturan out-of-the-box DQ untuk memulai, Anda dapat membuat aturan yang sesuai dengan kebutuhan spesifik Anda.
- Evaluasi kualitas dan buat keputusan bisnis yang percaya diri — Setelah Anda mengevaluasi aturan, Anda mendapatkan skor Kualitas Data yang memberikan gambaran umum tentang kesehatan data Anda. Gunakan skor Kualitas Data untuk membuat keputusan bisnis yang percaya diri.
- Nol pada data buruk — Kualitas AWS Glue Data membantu Anda mengidentifikasi catatan yang tepat yang menyebabkan skor kualitas Anda turun. Mudah mengidentifikasi mereka, karantina dan memperbaikinya.
- Bayar saat Anda pergi - Tidak ada lisensi tahunan yang Anda perlukan untuk menggunakan Kualitas AWS Glue Data.
- Tidak ada penguncian - Kualitas AWS Glue Data dibangun di atas sumber terbuka DeeQu, memungkinkan Anda untuk menjaga aturan yang Anda buat dalam bahasa terbuka.

- Pemeriksaan kualitas data — Kualitas AWS Glue Data Anda dapat menerapkan pemeriksaan kualitas data Data Catalog dan saluran AWS Glue ETL yang memungkinkan Anda mengelola kualitas data saat istirahat dan dalam perjalanan.
- Deteksi kualitas data berbasis ML — Gunakan pembelajaran mesin (ML) untuk mendeteksi anomali dan hard-to-detect masalah kualitas data.

Cara kerjanya

Ada dua titik masuk untuk Kualitas AWS Glue Data: pekerjaan AWS Glue ETL AWS Glue Data Catalog dan pekerjaan. Bagian ini memberikan gambaran umum tentang kasus penggunaan dan AWS Glue fitur yang didukung oleh setiap titik masuk.

Kualitas data untuk AWS Glue Data Catalog

AWS Glue Kualitas Data mengevaluasi objek yang disimpan dalam AWS Glue Data Catalog Ini menawarkan non-coders cara mudah untuk mengatur aturan kualitas data. Persona ini termasuk pengelola data dan analis bisnis.

Anda dapat memilih opsi ini untuk kasus penggunaan berikut:

- Anda ingin melakukan tugas kualitas data pada kumpulan data yang telah dikatalogkan di. AWS Glue Data Catalog
- Anda bekerja pada tata kelola data dan perlu mengidentifikasi atau mengevaluasi masalah kualitas data di danau data Anda secara berkelanjutan.

Anda dapat mengelola kualitas data untuk Katalog Data menggunakan antarmuka berikut:

- Konsol AWS Glue manajemen
- AWS Glue API

Untuk memulai dengan Kualitas AWS Glue Data untuk AWS Glue Data Catalog melihatMemulai dengan AWS Glue Data Quality untuk Data Catalog.

Kualitas data untuk pekerjaan AWS Glue ETL

AWS Glue Kualitas Data untuk pekerjaan AWS Glue ETL memungkinkan Anda melakukan tugas kualitas data proaktif. Tugas proaktif membantu Anda mengidentifikasi dan menyaring data buruk sebelum Anda memuat kumpulan data ke danau data Anda.

Video: Memperkenalkan Kualitas AWS Glue Data untuk Pipa ETL

Anda dapat memilih kualitas data untuk pekerjaan ETL untuk kasus penggunaan berikut:

- Anda ingin memasukkan tugas kualitas data ke dalam pekerjaan ETL Anda
- Anda ingin menulis kode yang mendefinisikan tugas kualitas data dalam skrip ETL
- Anda ingin mengelola kualitas data yang mengalir di pipeline data visual Anda

Anda dapat mengelola kualitas data untuk pekerjaan ETL menggunakan antarmuka berikut:

- AWS Glue Studio, AWS Glue Studio notebook, dan sesi AWS Glue interaktif
- AWS Glue pustaka untuk skrip ETL
- AWS Glue API

Untuk memulai kualitas data untuk pekerjaan ETL, lihat Tutorial: Memulai Kualitas Data di Panduan AWS Glue Studio Pengguna.

Membandingkan kualitas data untuk Katalog Data dengan kualitas data untuk pekerjaan ETL

Tabel ini memberikan ikhtisar fitur yang didukung oleh setiap titik masuk untuk Kualitas AWS Glue Data.

Fitur	Kualitas data untuk Katalog Data	Kualitas data untuk pekerjaan ETL
Sumber data	Amazon S3, Amazon Redshift, sumber JDBC yang kompatibel dengan Katalog Data, dan format danau data transaksional seperti Apache Iceberg,	Semua sumber data yang didukung oleh AWS Glue, termasuk konektor khusus dan konektor pihak ketiga.

Fitur	Kualitas data untuk Katalog Data	Kualitas data untuk pekerjaan ETL
	Apache Hudi, dan Delta Lake. Perhatikan bahwa jika tabel AWS Lake Formation dikelola, tabel Iceberg, Delta, dan HUDI tidak didukung. Amazon Athena tampilan yang dikatalogkan tidak AWS Glue Data Catalog didukung.	
Rekomendasi aturan Kualitas Data	Didukung	Tidak Support
Penulis dan jalankan aturan DQDL	Didukung	Didukung
Penskalaan otomatis	Tidak didukung	Didukung
AWS Glue Dukungan Flex	Tidak didukung	Didukung
Penjadwalan	Didukung saat mengevalu asi aturan Kualitas Data dan melalui Step Functions.	Didukung saat menggunakan Step Functions dan alur kerja.
Mengidentifikasi catatan yang gagal memeriksa kualitas data	Tidak didukung	Didukung
Integrasi dengan Amazon Eventbridge	Didukung	Didukung
Integrasi dengan AWS Cloudwatch	Didukung	Didukung
Menulis hasil kualitas data ke Amazon S3	Didukung	Didukung

Fitur	Kualitas data untuk Katalog Data	Kualitas data untuk pekerjaan ETL
Kualitas data tambahan	Didukung melalui predikat pushdown	Didukung melalui AWS Glue bookmark
AWS CloudFormation dukungan	Didukung	Didukung
Deteksi anomali berbasis ML	Tidak didukung	Pratinjau
Aturan dinamis	Tidak didukung	Didukung

Pertimbangan

Pertimbangkan hal-hal berikut sebelum Anda menggunakan Kualitas AWS Glue Data:

- Aturan kualitas data tidak dapat mengevaluasi sumber data bertingkat atau tipe daftar. Lihat Ratakan struct bersarang.

Terminologi

Daftar berikut mendefinisikan istilah yang terkait dengan Kualitas AWS Glue Data.

Bahasa Definisi Kualitas Data (DQDL)

Bahasa khusus domain yang dapat Anda gunakan untuk menulis aturan Kualitas AWS Glue Data.

Untuk mempelajari lebih lanjut tentang DQDL, lihat panduannya. Referensi Bahasa Definisi Kualitas Data (DQDL)

kualitas data

Menjelaskan seberapa baik dataset melayani tujuan spesifiknya. AWS Glue Kualitas Data mengevaluasi aturan terhadap kumpulan data untuk mengukur kualitas data. Setiap aturan memeriksa karakteristik tertentu seperti kesegaran atau integritas data. Untuk mengukur kualitas data, Anda dapat menggunakan skor kualitas data.

skor kualitas data

Persentase aturan kualitas data yang lulus (menghasilkan true) saat Anda mengevaluasi kumpulan aturan dengan Kualitas AWS Glue Data.

aturan

Ekspresi DQDL yang memeriksa data Anda untuk karakteristik tertentu dan mengembalikan nilai Boolean. Untuk informasi selengkapnya, lihat Struktur aturan.

analyzer

Ekspresi DQDL yang mengumpulkan statistik data. Analyzer mengumpulkan statistik data yang dapat digunakan oleh algoritma ML untuk mendeteksi anomali dan masalah kualitas hard-to-detect data dari waktu ke waktu.

aturan-aturan

AWS Glue Sumber daya yang terdiri dari seperangkat aturan kualitas data. Sebuah aturan harus dikaitkan dengan tabel di. AWS Glue Data Catalog Saat Anda menyimpan kumpulan aturan, AWS Glue tetapkan Nama Sumber Daya Amazon (ARN) ke kumpulan aturan.

skor kualitas data

Persentase aturan kualitas data yang lulus (menghasilkan true) saat Anda mengevaluasi kumpulan aturan dengan Kualitas AWS Glue Data.

observasi

Wawasan yang belum dikonfirmasi dihasilkan AWS Glue dengan menganalisis statistik data yang dikumpulkan dari aturan dan penganalisis dari waktu ke waktu.

Batas

AWS Glue Batas layanan Kualitas Data:

- Anda dapat memiliki 2000 aturan dalam kumpulan aturan. Jika aturan Anda lebih besar, kami sarankan untuk membagi menjadi beberapa aturan.
- Ukuran ruleset adalah 65KB. Jika aturan Anda lebih besar, kami sarankan untuk membagi menjadi beberapa aturan.

Catatan rilis untuk Kualitas AWS Glue Data

Topik ini menjelaskan fitur yang diperkenalkan dalam Kualitas AWS Glue Data.

Ketersediaan umum: fitur baru

Fitur-fitur baru berikut tersedia dengan ketersediaan umum Kualitas AWS Glue Data:

- Kemampuan untuk mengidentifikasi catatan mana yang gagal pemeriksaan kualitas data sekarang didukung AWS Glue Studio
- Jenis aturan kualitas data baru seperti memvalidasi integritas referensial data antara dua set data, membandingkan data antara dua kumpulan data, dan pemeriksaan tipe data
- Pengalaman pengguna yang lebih baik di AWS Glue Data Catalog
- Support untuk Apache Iceberg, Apache Hudi dan Delta Lake
- Dukungan untuk Amazon Redshift
- Pemberitahuan yang disederhanakan dengan Amazon EventBridge
- AWS CloudFormation dukungan untuk membuat rulesets
- Peningkatan kinerja: opsi caching di ETL dan AWS Glue Studio untuk kinerja yang lebih cepat saat mengevaluasi kualitas data

27 November 2023 (Pratinjau)

- Kemampuan deteksi anomali bertenaga ML sekarang tersedia di ETL dan. AWS Glue AWS Glue Studio Dengan ini, Anda sekarang dapat mendeteksi anomali dan masalah kualitas hard-to-detect data.
- Aturan Dinamis memungkinkan Anda untuk memberikan ambang dinamis (mis:`RowCount > avg(last(10))`).

Mar 12, 2024

- Perbaikan DQDL
  - Support untuk Kata Kunci seperti NULL, BLANKS, WHITESPACES_ONLY
  - Opsi untuk menentukan bagaimana Kualitas AWS Glue Data harus menangani aturan Komposit
  - ColumnValues tipe aturan tidak akan mengizinkan nilai NULL lewat selama perbandingan

- Support untuk operator NOT di DQDL

Juni 26, 2024

- Perbaikan DQDL
  - DQDL sekarang mendukung klausa where sehingga Anda dapat memfilter data sebelum menerapkan aturan DQ

Deteksi anomali dalam Kualitas Data AWS Glue

> **Note**
>
> AWS GlueKualitas Data tersedia dalam pratinjau di wilayah berikut:
>
> - AS Timur (Ohio, Virginia N.)
> - AS Barat (Oregon)
> - Asia Pasifik (Tokyo)
> - Eropa (Irlandia)

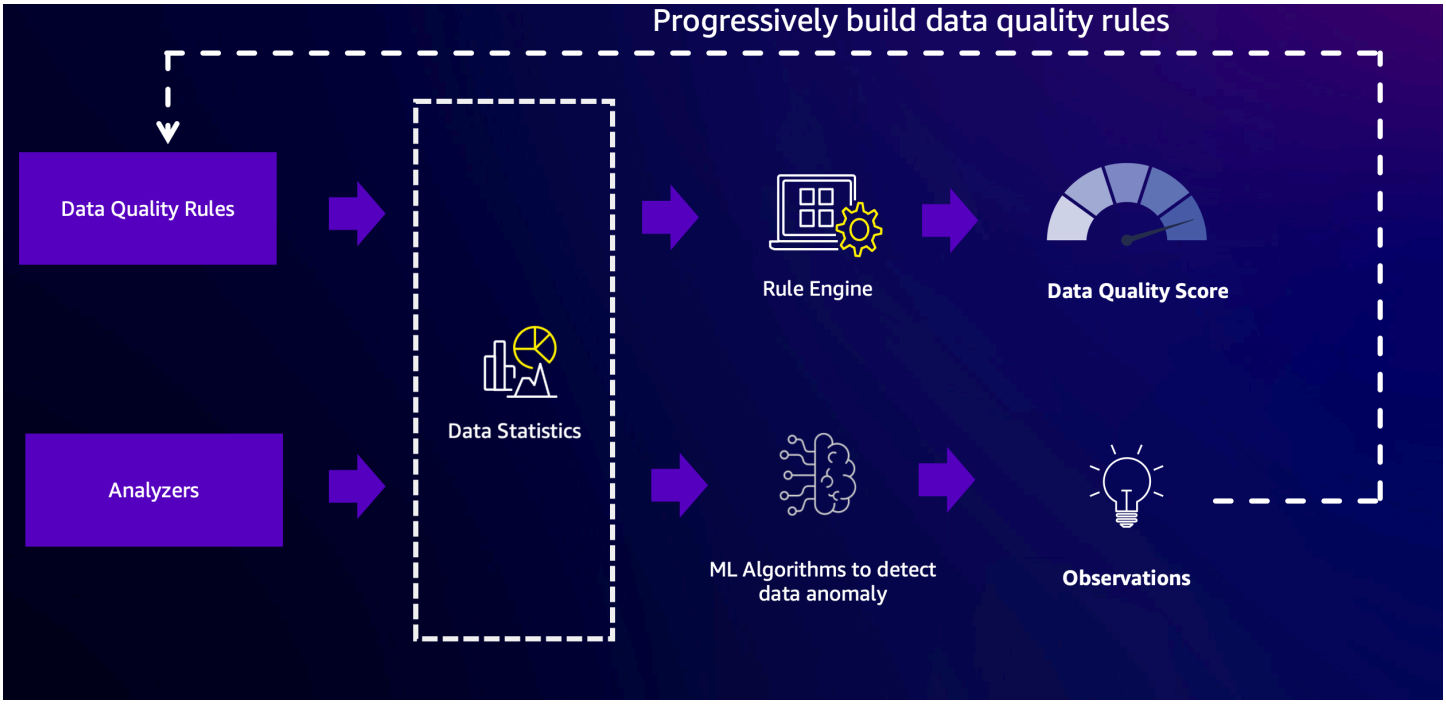
AWS GlueDeteksi anomali Kualitas Data menerapkan algoritma pembelajaran mesin (ML) pada statistik data dari waktu ke waktu untuk mendeteksi pola abnormal dan masalah kualitas data tersembunyi yang sulit dideteksi melalui aturan. Saat ini, deteksi anomali hanya tersedia untuk AWS Glue 4.0. Fitur ini saat ini hanya tersedia di AWS Glue Studio Visual ETL dan AWS Glue ETL. Kemampuan ini tidak berfungsi pada AWS Glue Studio Notebook, Katalog AWS Glue Data, Sesi AWS Glue Interaktif, dan Pratinjau AWS Glue Data.

Cara kerjanya

Saat mengevaluasi aturan Kualitas Data, AWS Glue menangkap statistik data yang diperlukan untuk menentukan apakah data sesuai dengan aturan. Misalnya, Kualitas Data akan menghitung jumlah nilai yang berbeda dalam kumpulan data, dan kemudian membandingkan nilai itu dengan harapan.

Mesin aturan Kualitas Data membandingkan nilai statistik dengan ambang batas yang ditentukan, dan mengevaluasi persyaratan kualitas Anda. Karena statistik ini dikumpulkan dari waktu ke waktu, Anda dapat mengaktifkan deteksi anomali pada pipa ETL Anda untuk membiarkan AWS Glue belajar

dari statistik masa lalu dan melaporkan pola tersembunyi sebagai Pengamatan. Pengamatan adalah wawasan yang belum dikonfirmasi yang diidentifikasi oleh algoritma AWS Glue ML. Mereka datang dengan aturan Kualitas Data yang direkomendasikan yang dapat Anda terapkan pada kumpulan aturan Anda untuk memantau pola yang ditemukan. Kami merekomendasikan menjalankan pekerjaan dengan jadwal reguler (misalnya, setiap jam dan harian). Lari yang tidak teratur dapat menghasilkan wawasan yang buruk.



Menggunakan analyzer untuk memeriksa data Anda

Terkadang, Anda mungkin tidak punya waktu untuk membuat aturan kualitas data. Di sinilah penganalisis berguna. Analyzer adalah bagian dari aturan Anda dan sangat mudah untuk dikonfigurasi. Misalnya, Anda dapat menulis ini di set aturan Anda:

```
Analyzers = [
      RowCount,
      Completeness “AllColumns”
]
```

Ini akan mengumpulkan statistik berikut:

- Hitungan Baris untuk seluruh kumpulan data
- Kelengkapan setiap kolom dalam dataset Anda

Kami merekomendasikan menggunakan Analyzers karena Anda tidak perlu khawatir tentang ambang batas. Anda dapat menjalankan pipeline data Anda dan setelah tiga kali berjalan, Kualitas AWS Glue Data akan mulai menghasilkan pengamatan dan rekomendasi aturan ketika melihat adanya anomali. Anda dapat meninjau pengamatan, statistik terkait dan dapat dengan mudah memasukkan rekomendasi aturan dalam kumpulan aturan Anda. Untuk memulai lihat Mengkonfigurasi deteksi Anomali dan menghasilkan wawasan . Perhatikan bahwa Analyzer tidak akan memengaruhi skor kualitas data Anda. Mereka menghasilkan statistik yang dapat dianalisis dari waktu ke waktu untuk menghasilkan pengamatan.

Menggunakan DetectAnomaly Aturan

Terkadang, Anda ingin pekerjaan Anda gagal ketika mendeteksi anomali. Untuk menerapkan batasan, Anda harus mengonfigurasi aturan. Analyzer tidak akan menghentikan pekerjaan. Sebaliknya, mereka akan mengumpulkan statistik dan menganalisis data. Mengkonfigurasi `DetectAnomaly` aturan di bagian aturan dari kumpulan aturan akan mengonfirmasi bahwa pemindaian DQ melaporkan pekerjaan telah gagal melewati semua aturan dalam pemindaian.

Manfaat dan penggunaan kasus Deteksi Anomali

Insinyur dapat mengelola ratusan pipa data pada waktu tertentu. Setiap pipa dapat mengekstrak data dari sumber yang berbeda dan memuatnya ke danau data. Karena setiap pipeline mungkin mengekstrak data dari sumber yang berbeda dan memuatnya ke data lake, sulit untuk mendapatkan umpan balik langsung pada data — apakah bentuknya telah berubah secara signifikan, atau telah menyimpang dari tren yang ada.

Di masa lalu, sumber data hulu telah berubah tanpa peringatan kepada tim rekayasa data, memperkenalkan hard-to-track “bug data” ke dalam proses ini. Dengan menambahkan node Kualitas Data ke pekerjaan, ini membuat hidup lebih mudah, karena pekerjaan gagal ketika masalah terlihat. Namun, ini tidak menghapus semua mode kegagalan yang dikhawatirkan tim data, yang membuat pintu tetap terbuka untuk bug data lain masuk.

Salah satu mode kegagalan adalah sekitar volume data. Ketika penyimpanan data perusahaan tumbuh dari waktu ke waktu, jumlah catatan yang dihasilkan oleh pipa data dapat tumbuh secara eksponensial. Setiap minggu, tim data mungkin perlu memperbarui pekerjaan ETL secara manual untuk meningkatkan setiap aturan Kualitas Data yang menetapkan batas jumlah baris yang dicerna.

Mode kegagalan lainnya adalah bahwa beberapa batas aturan kualitas data sangat luas untuk mengakomodasi fakta bahwa volume transaksi bervariasi berdasarkan hari dalam seminggu. Pada

akhir pekan, hampir tidak ada transaksi, dan pada hari Senin ada sekitar tiga kali lebih banyak transaksi daripada pada hari kerja lainnya. Tim data memiliki dua opsi - menerapkan logika untuk mengubah aturan dengan cepat tergantung pada hari, atau menetapkan harapan yang sangat luas.

Akhirnya, tim data juga peduli dengan bug data yang kurang terdefinisi dengan baik. Model telah dilatih pada data dengan karakteristik tertentu, dan jika ini mulai miring dengan cara yang tidak terduga, tim ingin tahu. Misalnya, pada bulan Februari sebuah perusahaan dapat memperluas ke Montana, sehingga transaksi mulai berisi kode “MT” muncul lebih sering. Ini dapat mematahkan inferensi ML, dan sebagai hasilnya model secara salah memperkirakan bahwa setiap transaksi Montana adalah penipuan.

Di sinilah Deteksi anomali kualitas data dapat membantu memecahkan masalah ini. Beberapa manfaat deteksi anomali Kualitas Data meliputi:

- Pemindaian data berdasarkan jadwal, berbasis peristiwa, atau manual.
- Deteksi anomali yang dapat menjadi indikasi peristiwa yang tidak diinginkan, musiman, atau kelainan statistik.
- Tawarkan Rekomendasi Aturan untuk mengambil tindakan pada pengamatan yang ditemukan oleh deteksi anomali Kualitas Data.

Ini berguna jika Anda:

- ingin mendeteksi anomali pada data Anda secara otomatis tanpa perlu menulis aturan kualitas data.
- ingin menangkap potensi masalah dalam data Anda yang tidak dapat ditemukan oleh aturan kualitas data saja.
- ingin mengotomatiskan beberapa tugas yang berkembang dari waktu ke waktu, seperti membatasi jumlah baris yang dicerna untuk pemantauan kualitas data.

Konfigurasikan izin IAM untuk Kualitas Data AWS Glue

Topik ini memberikan informasi untuk membantu Anda memahami tindakan dan sumber daya yang dapat Anda gunakan oleh administrator IAM dalam kebijakan AWS Identity and Access Management (IAM) untuk Kualitas Data AWS Glue. Ini juga mencakup contoh kebijakan IAM dengan izin minimum yang Anda perlukan untuk menggunakan Kualitas Data AWS Glue dengan Katalog Data AWS Glue.

Untuk informasi tambahan tentang keamanan di AWS Glue, lihatKeamanan di AWS Glue.

Izin IAM untuk Kualitas Data AWS Glue

Tabel berikut mencantumkan izin yang dibutuhkan pengguna untuk melakukan operasi Kualitas Data AWS Glue tertentu. Untuk menetapkan otorisasi halus untuk AWS Glue Data Quality, Anda dapat menentukan tindakan ini dalam elemen pernyataan `Action` kebijakan IAM.

AWSGlue Data Tindakan Kualitas

Action	Deskripsi	Jenis sumber daya
`glue:CreateDataQualityRuleset`	Memberikan izin untuk membuat kumpulan aturan kualitas data.	`::dataQualityRuleset/<name>`
`glue:DeleteDataQualityRuleset`	Memberikan izin untuk menghapus kumpulan aturan kualitas data.	`::dataQualityRuleset/<name>`
`glue:GetDataQualityRuleset`	Memberikan izin untuk mengambil kumpulan aturan kualitas data.	`::dataQualityRuleset/<name>`
`glue:ListDataQualityRulesets`	Memberikan izin untuk mengambil semua aturan kualitas data.	`::dataQualityRuleset/*`
`glue:UpdateDataQualityRuleset`	Memberikan izin untuk memperbarui kumpulan aturan kualitas data.	`::dataQualityRuleset/<name>`
`glue:GetDataQualityResult`	Memberikan izin untuk mengambil hasil tugas kualitas data yang dijalankan.	`::dataQualityRuleset/<name>`
`glue:ListDataQualityResults`	Memberikan izin untuk mengambil semua hasil tugas kualitas data yang dijalankan.	`::dataQualityRuleset/*`

Action	Deskripsi	Jenis sumber daya
glue:CancelDataQualityRuleRecommendationRun	Memberikan izin untuk menghentikan menjalankan tugas rekomendasi kualitas data yang sedang berlangsung.	::dataQualityRuleset/*
glue:GetDataQualityRuleRecommendationRun	Memberikan izin untuk mengambil tugas rekomendasi kualitas data yang dijalankan.	::dataQualityRuleset/*
glue:ListDataQualityRuleRecommendationRuns	Memberikan izin untuk mengambil semua tugas rekomendasi kualitas data yang berjalan.	::dataQualityRuleset/*
glue:StartDataQualityRuleRecommendationRun	Memberikan izin untuk memulai tugas rekomendasi kualitas data yang dijalankan.	::dataQualityRuleset/*
glue:CancelDataQualityRulesetEvaluationRun	Memberikan izin untuk menghentikan menjalankan tugas kualitas data yang sedang berlangsung.	::dataQualityRuleset/*
glue:GetDataQualityRulesetEvaluationRun	Memberikan izin untuk mengambil tugas kualitas data yang dijalankan.	::dataQualityRuleset/*
glue:ListDataQualityRulesetEvaluationRuns	Memberikan izin untuk mengambil semua tugas kualitas data yang berjalan.	::dataQualityRuleset/*
glue:StartDataQualityRulesetEvaluationRun	Memberikan izin untuk memulai menjalankan tugas kualitas data.	::dataQualityRuleset/<name>

Action	Deskripsi	Jenis sumber daya
`glue:PublishDataQuality`	Memberikan izin untuk mempublikasikan hasil kualitas data	`::dataQualityRuleset/<name>`

Penyiapan IAM diperlukan untuk evaluasi penjadwalan berjalan

Izin IAM

Untuk menjalankan evaluasi Kualitas Data terjadwal, Anda harus menambahkan `IAM:PassRole` tindakan ke kebijakan izin.

AWS EventBridge Izin yang diperlukan penjadwal

Action	Deskripsi	Jenis sumber daya
`iam:PassRole`	Memberikan izin kepada IAM untuk memungkinkan pengguna melewati peran yang disetujui.	ARN dari peran yang digunakan untuk memanggil `StartDataQualityRulesetEvaluationRun`

Tanpa izin ini terjadi kesalahan berikut:

```
"errorCode": "AccessDenied"
"errorMessage": "User: arn:aws:sts::account_id:assumed-role/AWSGlueServiceRole is not
authorized to perform: iam:PassRole on resource: arn:aws:iam::account_id:role/service-
role/AWSGlueServiceRole
because no identity-based policy allows the iam:PassRole action"
```

Entitas tepercaya IAM

Layanan AWS Glue dan AWS EventBridge Scheduler harus terdaftar di entitas tepercaya untuk membuat dan menjalankan jadwal`StartDataQualityEvaluationRun`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Principal": {
                "Service": "glue.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "scheduler.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Kebijakan contoh IAM

Peran IAM untuk AWS Glue Data Quality memerlukan jenis izin berikut:

- Izin untuk operasi AWS Glue Data Quality sehingga Anda bisa mendapatkan aturan kualitas data yang direkomendasikan dan menjalankan tugas kualitas data terhadap tabel di Katalog Data AWS Glue. Contoh kebijakan IAM di bagian ini mencakup izin minimum yang diperlukan untuk operasi AWS Glue Data Quality.
- Izin yang memberikan akses ke tabel Katalog Data Anda dan data yang mendasarinya. Izin ini bervariasi tergantung pada kasus penggunaan Anda. Misalnya, untuk data yang Anda katalog di Amazon S3, izin harus menyertakan akses ke Amazon S3.

> **Note**
>
> Anda harus mengonfigurasi izin Amazon S3 selain izin yang dijelaskan di bagian ini.

Izin minimum untuk mendapatkan aturan kualitas data yang direkomendasikan

Kebijakan contoh ini mencakup izin yang Anda perlukan untuk menghasilkan aturan kualitas data yang direkomendasikan.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Sid": "AllowGlueRuleRecommendationRunActions",
      "Effect": "Allow",
      "Action": [
        "glue:GetDataQualityRuleRecommendationRun",
        "glue:PublishDataQuality",
        "glue:CreateDataQualityRuleset"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*"
    },
 {
     "Sid": "AllowCatalogPermissions",
     "Effect": "Allow",
     "Action": [
        "glue:GetPartitions",
        "glue:GetTable"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Sid": "AllowS3GetObjectToRunRuleRecommendationTask",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": "arn:aws:s3:::aws-glue-*"
    },
 {   // Optional for Logs
        "Sid": "AllowPublishingCloudwatchLogs",
        "Effect": "Allow",
        "Action": [
          "logs:CreateLogStream",
          "logs:CreateLogGroup",
          "logs:PutLogEvents"
        ],
        "Resource": "*"
    },
  ]
}
```

Izin minimum untuk menjalankan tugas kualitas data

Kebijakan contoh ini mencakup izin yang Anda perlukan untuk menjalankan tugas evaluasi kualitas data.

Pernyataan kebijakan berikut bersifat opsional, tergantung pada kasus penggunaan Anda:

- `AllowCloudWatchPutMetricDataToPublishTaskMetrics`- Diperlukan jika Anda ingin mempublikasikan metrik run kualitas data keAmazon CloudWatch.
- `AllowS3PutObjectToWriteTaskResults`- Diperlukan jika Anda ingin menulis hasil menjalankan kualitas data ke Amazon S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowGlueGetDataQualityRuleset",
      "Effect": "Allow",
      "Action": [
        "glue:GetDataQualityRuleset"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/<YOUR-
RULESET-NAME>"
    },
    {
      "Sid": "AllowGlueRulesetEvaluationRunActions",
      "Effect": "Allow",
      "Action": [
        "glue:GetDataQualityRulesetEvaluationRun",
        "glue:PublishDataQuality"
      ],
      "Resource": "arn:aws:glue:us-east-1:111122223333:dataQualityRuleset/*"
    },
 {
      "Sid": "AllowCatalogPermissions",
      "Effect": "Allow",
      "Action": [
        "glue:GetPartitions",
        "glue:GetTable"
      ],
      "Resource": [
        "*"
```

```
      ]
    },
    {
      "Sid": "AllowS3GetObjectForRulesetEvaluationRun",
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": "arn:aws:s3:::aws-glue-*"
    },
    {
      "Sid": "AllowCloudWatchPutMetricDataToPublishTaskMetrics",
      "Effect": "Allow",
      "Action": [
        "cloudwatch:PutMetricData"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "cloudwatch:namespace": "Glue Data Quality"
        }
      }
    },
    {
      "Sid": "AllowS3PutObjectToWriteTaskResults",
      "Effect": "Allow",
      "Action": [
        "s3:PutObject*"
      ],
      "Resource": "arn:aws:s3:::<YOUR-BUCKET-NAME>/*"
    }
  ]
}
```

Memulai dengan AWS Glue Data Quality untuk Data Catalog

Bagian memulai ini memberikan petunjuk untuk membantu Anda memulai AWS Glue Data Quality di AWS Glue konsol. Anda akan belajar cara menyelesaikan tugas-tugas penting seperti menghasilkan rekomendasi aturan kualitas data dan mengevaluasi kumpulan aturan terhadap data Anda.

Topik

- Prasyarat

Prasyarat

Sebelum Anda menggunakanAWS Glue Data Quality, Anda harus terbiasa menggunakan Data Catalog dan crawler diAWS Glue. DenganAWS Glue Data Quality, Anda dapat mengevaluasi kualitas untuk tabel dalam Data Catalog database. Anda juga memerlukan hal berikut:

- Tabel di Data Catalog untuk mengevaluasi aturan kualitas data Anda terhadap.
- Peran IAM untuk AWS Glue yang Anda berikan saat Anda membuat rekomendasi aturan atau menjalankan tugas kualitas data. Peran ini harus memiliki izin untuk mengakses sumber daya yang diperlukan berbagai AWS Glue Data Quality proses untuk dijalankan atas nama Anda. Sumber daya ini termasukAWS Glue, Amazon S3, dan. CloudWatch Untuk melihat contoh kebijakan yang menyertakan izin minimumAWS Glue Data Quality, lihatKebijakan contoh IAM.

  Untuk mempelajari lebih lanjut tentang peran IAMAWS Glue, lihat Membuat kebijakan IAM untuk AWS Glue layanan dan Membuat peran IAM untuk layanan. AWS Glue Anda juga dapat melihat daftar semua AWS Glue izin yang khusus untuk kualitas data di Otorisasi untuk AWS Glue Data Quality tindakan.
- Database dengan setidaknya satu tabel yang berisi berbagai data. Tabel yang digunakan dalam tutorial ini diberi nama`yyz-tickets`, dengan tabel`tickets`. Data ini adalah kumpulan informasi yang tersedia untuk umum dari Kota Toronto untuk kutipan parkir. Jika Anda membuat tabel Anda sendiri, pastikan bahwa itu diisi dengan berbagai data yang valid untuk mendapatkan set terbaik dari aturan yang direkomendasikan.

tep-by-step Contoh S

Untuk step-by-step contoh dengan kumpulan data sampel, lihat posting blog AWS Glue Data Quality.

Menghasilkan rekomendasi aturan

Rekomendasi aturan memudahkan untuk memulai dengan kualitas data tanpa menulis kode. Dengan AWS Glue Data Quality, Anda dapat menganalisis data Anda, mengidentifikasi aturan, dan membuat kumpulan aturan yang dapat Anda evaluasi dalam tugas kualitas data. Rekomendasi berjalan secara otomatis dihapus setelah 90 hari.

Untuk menghasilkan rekomendasi aturan kualitas data

1. Buka konsol Glue AWS di https://console.aws.amazon.com/glue/.
2. Pilih Tabel di panel navigasi. Kemudian pilih tabel yang ingin Anda hasilkan rekomendasi aturan kualitas data.
3. Pada halaman detail tabel, pilih tab Kualitas data untuk mengakses aturan dan pengaturan Kualitas Data AWS Glue untuk tabel Anda.
4. Pada tab Kualitas data, pilih Tambahkan aturan dan pantau kualitas data.
5. Pada halaman pembuat Ruleset, peringatan di bagian atas halaman akan meminta Anda untuk memulai tugas rekomendasi jika tidak ada rekomendasi aturan yang berjalan.
6. Pilih Rekomendasikan aturan untuk membuka modal dan memasukkan parameter Anda untuk tugas rekomendasi.
7. Pilih peran IAM dengan akses ke AWS Glue. Peran ini harus memiliki izin untuk mengakses sumber daya yang diperlukan oleh berbagai proses AWS Glue Data Quality untuk dijalankan atas nama Anda.
8. Setelah bidang selesai sesuai dengan preferensi Anda, pilih Rekomendasikan aturan untuk memulai tugas rekomendasi dijalankan. Jika rekomendasi berjalan atau selesai, Anda dapat mengelola proses Anda di peringatan ini. Anda mungkin perlu menyegarkan peringatan untuk melihat perubahan status. Tugas rekomendasi yang sudah selesai dan sedang berjalan muncul di halaman Riwayat Jalankan yang mencantumkan semua rekomendasi yang berjalan selama 90 hari terakhir.

Apa arti aturan yang direkomendasikan

AWSGlue Data Quality menghasilkan aturan berdasarkan data dari setiap kolom tabel input. Ini menggunakan aturan untuk mengidentifikasi batas-batas potensial di mana data dapat disaring untuk mempertahankan persyaratan kualitas. Daftar aturan yang dihasilkan berikut mencakup contoh yang berguna untuk memahami apa arti aturan dan apa yang mungkin mereka lakukan ketika diterapkan pada data Anda.

Untuk daftar lengkap tipe aturan Data Quality Definition Language (DQDL) yang dihasilkan, lihat referensi tipe aturan DQDL.

- `IsComplete "SET_FINE_AMOUNT"IsComplete`—Aturan memverifikasi bahwa kolom diisi untuk setiap baris yang diberikan. Gunakan aturan ini untuk menandai kolom sebagai non-opsional dalam data.
- `Uniqueness "TICKET_NUMBER" > 0.95`— `Uniqueness` Aturan memverifikasi bahwa data dalam kolom memenuhi beberapa ambang keunikan. Dalam contoh ini, data yang mengisi setiap baris tertentu ditentukan paling banyak 95% identik dalam konten untuk semua baris lainnya, yang menunjukkan aturan ini. `"TICKET_NUMBER"`
- `ColumnValues "PROVINCE" in ["ON", "QC", "AB", "NY",...]`— `ColumnValues` Aturan mendefinisikan nilai yang valid untuk kolom, berdasarkan isi kolom yang ada. Dalam contoh ini, data untuk setiap baris adalah plat kode nomor 2 huruf untuk negara bagian atau provinsi.
- `ColumnLength "INFRACTION_DESCRIPTION" between 15 and 31`— `ColumnLength` Aturan memberlakukan pembatasan panjang pada data kolom. Aturan ini dihasilkan dari data sampel berdasarkan panjang minimum dan maksimum yang direkam untuk kolom string.

Rekomendasi aturan pemantauan

Saat rekomendasi aturan kualitas data berjalan, halaman Tambahkan aturan dan monitor kualitas data menampilkan informasi dan tindakan tambahan yang dapat Anda lakukan di bilah atas.

Saat rekomendasi aturan sedang berlangsung, Anda dapat memilih Stop run sebelum tugas rekomendasi selesai. Saat tugas sedang berlangsung, Anda akan melihat status, dalam proses, dan tanggal dan waktu saat proses dimulai.

Ketika rekomendasi aturan selesai, bilah rekomendasi aturan menampilkan jumlah aturan yang direkomendasikan, status rekomendasi terakhir yang dijalankan, dan tanggal serta stempel waktu saat selesai.

Anda dapat menambahkan aturan yang disarankan dengan memilih Sisipkan Rekomendasi Aturan. Untuk melihat aturan yang direkomendasikan sebelumnya, pilih tanggal tertentu. Untuk menjalankan rekomendasi baru, pilih Tindakan lainnya, lalu pilih Aturan yang disarankan.

Tetapkan pengaturan default dengan memilih Kelola pengaturan pengguna. Anda dapat mengatur jalur default Amazon S3 untuk menyimpan kumpulan aturan atau mengatur peran default untuk menjalankan Katalog Data.

Mengedit set aturan yang direkomendasikan

Karena Kualitas Data AWS Glue menghasilkan aturan berdasarkan data yang ada yang tersedia, Anda mungkin melihat beberapa aturan yang tidak terduga atau tidak diinginkan dalam saran otomatis. Untuk mendapatkan hasil maksimal dari aturan yang direkomendasikan, Anda perlu mengevaluasi dan memodifikasinya. Untuk langkah tutorial ini, Anda mengambil aturan yang dihasilkan pada langkah sebelumnya dan menyesuaikannya untuk menegakkan kualitas yang lebih ketat pada beberapa data. Anda juga melonggarkan aturan lain untuk memastikan bahwa data unik yang benar dapat ditambahkan nanti.

Edit kumpulan aturan yang disarankan

1. Di konsol AWS Glue, pilih Katalog Data, lalu pilih tabel Database di panel navigasi. Pilih tabel `tickets`.
2. Pada halaman detail tabel, pilih tab Kualitas data untuk mengakses aturan dan pengaturan Kualitas Data AWS Glue untuk tabel.
3. Di bagian Rulesets, pilih ruleset yang dihasilkan di. Menghasilkan rekomendasi aturan
4. Pilih Tindakan, lalu pilih Edit di jendela konsol. Editor ruleset dimuat di konsol. Ini termasuk panel pengeditan untuk aturan Anda dan referensi cepat untuk DQDL.
5. 2Hapus baris skrip. Ini melonggarkan persyaratan bahwa ukuran database dibatasi dalam sejumlah baris tertentu. Setelah pengeditan, file Anda harus berisi yang berikut pada baris 1-3:

```
Rules = [
    IsComplete "TAG_NUMBER_MASKED",
    ColumnLength "TAG_NUMBER_MASKED" between 6 and 9,
```

6. 25Hapus baris skrip. Ini melonggarkan persyaratan bahwa 96% dari provinsi yang tercatat adalah. 0N Setelah pengeditan, file Anda harus berisi yang berikut dari baris 24 ke akhir kumpulan aturan:

```
ColumnValues "PROVINCE" in ["ON", "QC", "AB", "NY", "AZ", "NS", "BC", "MI", "PQ",
 "MB", "PA", "FL", "SK", "NJ", "OH", "NB", "IL", "MA", "CA",
    "VA", "TX", "NF", "MD", "PE", "CT", "NC", "GA", "IN", "OR", "MN", "TN", "WI",
 "KY", "MO", "WA", "NH", "SC", "CO", "OK", "VT", "RI", "ME", "AL",
    "YT", "IA", "DE", "AR", "LA", "XX", "WV", "MT", "KS", "NT", "DC", "NV", "NE",
 "UT", "MS", "NM", "ID", "SD", "ND", "AK", "NU", "GO", "WY", "HI"],
ColumnLength "PROVINCE" = 2
]
```

7. Ubah baris 14 menjadi berikut:

```
IsComplete "TIME_OF_INFRACTION",
```

Ini memperkuat persyaratan pada kolom dengan membatasi database hanya untuk tiket yang berisi waktu pelanggaran yang tercatat. Anda harus selalu menganggap tiket tanpa waktu pelanggaran yang tercatat sebagai data yang tidak valid dalam kumpulan data ini. Ini berbeda dari situasi di mana partisi atau transformasi mungkin lebih tepat untuk penggunaan data lebih lanjut atau inspeksi untuk menentukan aturan kualitas.

8. Pilih Perbarui Aturan di bagian bawah halaman konsol.

Membuat ruleset baru

Kumpulan aturan adalah sekelompok aturan kualitas data yang Anda evaluasi terhadap data Anda. Di AWS Glue konsol, Anda dapat membuat aturan khusus menggunakan Data Quality Definition Language (DQDL).

Untuk membuat kumpulan aturan kualitas data

1. Di konsol AWS Glue, pilih Katalog Data, pilih Database, lalu pilih Tabel di panel navigasi. Pilih tabel`tickets`.
2. Buka tab Kualitas data.
3. Di bagian Rulesets, pilih Create ruleset. Editor DQDL diluncurkan di konsol. Ini memiliki area teks untuk pengeditan langsung, dan referensi cepat untuk aturan DQDL dan skema tabel.
4. Mulai menambahkan aturan ke area teks editor DQDL. Anda dapat menulis aturan langsung dari tutorial ini, atau menggunakan fitur pembuat aturan DQDL dari editor aturan kualitas data.

> **Note**
>
> Cara menggunakan pembuat aturan DQDL
>
> 1. Pilih jenis aturan dari daftar, dan pilih tanda plus untuk menyisipkan sintaks contoh ke panel editor.
> 2. Tukarkan nama kolom placeholder dengan nama kolom Anda sendiri. Nama kolom dari tabel tersedia di tab Skema.

3. Perbarui parameter ekspresi sesuai keinginan Anda. Untuk daftar lengkap ekspresi yang didukung DQDL, lihat. Ekspresi

Sebagai contoh, aturan berikut adalah kendala untuk validasi data `ticket_number` kolom dalam tabel. `tickets` Untuk menambahkan aturan berikut, gunakan pembuat aturan DQDL atau langsung edit kumpulan aturan Anda:

```
IsComplete "ticket_number",
IsUnique "ticket_number",
ColumnValues "ticket_number" > 9000000000
```

5. Berikan nama untuk kumpulan aturan baru Anda di bidang nama Ruleset.
6. Pilih Simpan ruleset.

Mengevaluasi kualitas data di beberapa kumpulan data

Anda dapat mengatur aturan kualitas data di beberapa kumpulan data menggunakan ReferentialIntegrity dan DatasetMatch kumpulan aturan. ReferentialIntegritymemeriksa untuk melihat apakah data dalam kumpulan data utama ada di kumpulan data lain.

Untuk menambahkan kumpulan data referensi, pilih tab Skema dan kemudian pilih Perbarui tabel referensi. Anda akan diminta untuk memilih database dan tabel. Anda dapat menambahkan tabel dan kemudian mengatur aturan kualitas data. Jenis aturan seperti AggregateMatch, RowCountMatch, ReferentialIntegrity, SchemaMatch, dan DatasetMatch mendukung kemampuan untuk melakukan pemeriksaan kualitas data di beberapa kumpulan data.

Menjalankan kumpulan aturan untuk mengevaluasi kualitas data

Saat Anda menjalankan tugas kualitas data, AWS Glue Data Quality mengevaluasi kumpulan aturan terhadap data Anda dan menghitung skor kualitas data. Skor ini mewakili persentase aturan kualitas data yang diteruskan untuk input.

Untuk menjalankan tugas kualitas data

1. Di konsol AWS Glue, pilih Katalog Data, pilih Database, lalu pilih Tabel di panel navigasi. Pilih tabel`tickets`.
2. Pilih tab Kualitas data.

3. Dalam daftar Rulesets, pilih kumpulan aturan yang ingin Anda evaluasi terhadap tabel. Untuk langkah ini, sebaiknya gunakan kumpulan aturan yang sudah Anda tulis atau modifikasi daripada membuat aturan. Pilih Jalankan.
4. Dalam modal, pilih peran IAM Anda. Peran ini harus memiliki izin untuk mengakses sumber daya yang diperlukan oleh berbagai proses AWS Glue Data Quality untuk dijalankan atas nama Anda. Anda dapat menyimpan peran IAM sebagai default atau memodifikasinya dengan membuka halaman Pengaturan Default.
5. Di bawah Tindakan kualitas data, pilih apakah Anda ingin Menerbitkan metrik ke Amazon CloudWatch. Ketika opsi ini dipilih, AWS Glue Data Quality menerbitkan metrik yang menunjukkan jumlah aturan yang disahkan dan jumlah aturan yang gagal. Untuk mengambil tindakan pada metrik yang disimpan dengan cara ini, Anda dapat menggunakan CloudWatch alarm. Metrik kunci juga dipublikasikan Amazon EventBridge agar Anda dapat mengatur peringatan. Untuk informasi selengkapnya, lihat Menyiapkan peringatan, penerapan, dan penjadwalan.
6. Di Run Frequency, pilih run on demand atau jadwalkan aturan. Saat Anda menjadwalkan kumpulan aturan, Anda akan diminta untuk nama tugas. Jadwal akan dibuat diAmazon EventBridge. Anda dapat mengedit jadwal Anda diAmazon EventBridge.
7. Untuk menyimpan hasil kualitas data di Amazon S3, pilih lokasi hasil kualitas data. Peran IAM yang sebelumnya Anda pilih untuk tugas ini harus memiliki akses tulis ke lokasi ini.
8. Di bawah Konfigurasi Tambahan, masukkan jumlah pekerja yang diminta yang ingin AWS Glue alokasikan untuk tugas kualitas data Anda.
9. Anda dapat secara opsional mengatur filter di sumber data. Ini membantu Anda mengurangi data yang Anda baca. Anda juga dapat menggunakan filter untuk menjalankan validasi inkremental dengan memilih informasi partisi dan meneruskannya sebagai parameter melalui panggilan API. Untuk meningkatkan kinerja, Anda dapat memberikan predikat partisi.
10. Pilih Jalankan. Anda akan melihat tugas baru Anda dalam daftar tugas kualitas data berjalan. Ketika kolom status Jalankan untuk tugas ditampilkan sebagai Selesai, Anda dapat melihat hasil skor kualitas. Anda mungkin perlu menyegarkan jendela konsol agar status diperbarui dengan benar.
11. Untuk melihat kolom untuk detail hasil kualitas data, pilih ikon “+” untuk memperluas kumpulan aturan. Hasilnya menunjukkan kepada Anda aturan yang lulus dan gagal dalam evaluasi, dan apa yang memicu kegagalan aturan.

Melihat skor kualitas data dan hasil

Untuk melihat proses terbaru pada semua set aturan yang dibuat

1. Di konsol AWS Glue, pilih Tabel di panel navigasi. Kemudian pilih tabel yang ingin Anda jalankan tugas kualitas data.
2. Pilih tab Kualitas data.
3. Snapshot kualitas data menunjukkan tren umum berjalan dari waktu ke waktu. 10 run terakhir di semua set aturan ditampilkan secara default. Untuk memfilter berdasarkan aturan, pilih yang diinginkan dari daftar dropdown. Jika ada kurang dari 10 run, semua proses selesai yang tersedia akan ditampilkan.
4. Dalam tabel kualitas Data, setiap kumpulan aturan dengan proses terbarunya (jika ada) ditampilkan, bersama dengan skor. Memperluas ruleset menampilkan aturan yang ada di ruleset itu, bersama dengan hasil aturan untuk menjalankan itu.

Untuk melihat proses terbaru pada kumpulan aturan tertentu

1. Di konsol AWS Glue, pilih Tabel di panel navigasi. Kemudian pilih tabel yang ingin Anda jalankan tugas kualitas data.
2. Pilih tab Kualitas data.
3. Dalam tabel kualitas data, pilih pada set aturan tertentu.
4. Pada halaman Rincian Ruleset, pilih tab Run history.

   Semua evaluasi berjalan untuk kumpulan aturan khusus ini tercantum dalam tabel di dalam tab ini. Anda dapat melihat sejarah skor dan status lari.
5. Untuk melihat informasi selengkapnya tentang proses tertentu, pilih Run ID untuk membuka halaman Evaluasi run details. Di halaman ini, Anda dapat melihat secara spesifik tentang proses dan detail lebih lanjut tentang status hasil aturan individu.

Topik terkait

- Referensi tipe aturan DQDL
- Referensi Bahasa Definisi Kualitas Data (DQDL)

Mengevaluasi kualitas data dengan AWS Glue Studio

AWS GlueKualitas Data mengevaluasi dan memantau kualitas data Anda berdasarkan aturan yang Anda tetapkan. Ini memudahkan untuk mengidentifikasi data yang membutuhkan tindakan. DiAWS Glue Studio, Anda dapat menambahkan node kualitas data ke pekerjaan visual Anda untuk membuat aturan kualitas data pada tabel di Katalog Data Anda. Anda kemudian dapat memantau dan mengevaluasi perubahan pada kumpulan data Anda saat mereka berkembang dari waktu ke waktu. Untuk gambaran umum tentang cara bekerja dengan Kualitas AWS Glue Data diAWS Glue Studio, lihat video berikut.

Berikut ini adalah langkah-langkah tingkat tinggi untuk bagaimana Anda bekerja dengan Kualitas AWS Glue Data:

1. Buat aturan kualitas data — Buat seperangkat aturan kualitas data menggunakan pembuat DQDL dengan memilih kumpulan aturan bawaan yang Anda konfigurasikan.
2. Konfigurasikan pekerjaan kualitas data - Tentukan tindakan berdasarkan hasil kualitas data dan opsi keluaran.
3. Simpan dan jalankan pekerjaan berkualitas data — Buat dan jalankan pekerjaan. Menyimpan pekerjaan akan menyimpan aturan yang Anda buat untuk pekerjaan itu.
4. Pantau dan tinjau hasil kualitas data — Tinjau hasil kualitas data setelah pekerjaan selesai. Secara opsional, jadwalkan pekerjaan untuk kencan masa depan.

ManfaatIT

Analis data, insinyur data, dan ilmuwan data dapat menggunakan node Evaluate Data Quality AWS Glue Studio untuk menganalisis, mengkonfigurasi, memantau, dan meningkatkan kualitas data dari editor pekerjaan visual. Manfaat menggunakan node kualitas data meliputi:

- Anda dapat mendeteksi masalah kualitas data - Anda dapat memeriksa masalah dengan membuat aturan yang memeriksa karakteristik kumpulan data Anda.
- Sangat mudah untuk memulai - Anda dapat mulai dengan aturan dan tindakan pra-dibangun.
- Integrasi ketat - Anda dapat menggunakan node kualitas AWS Glue data AWS Glue Studio karena Kualitas Data berjalan di atas Katalog AWS Glue Data.

Mengevaluasi kualitas data untuk pekerjaan ETL di AWS Glue Studio

Dalam tutorial ini, Anda memulai dengan Kualitas AWS Glue Data diAWS Glue Studio. Anda akan belajar cara:

- Buat aturan menggunakan pembuat aturan Data Quality Definition Language (DQDL).
- Tentukan tindakan kualitas data, data ke output, dan lokasi output dari hasil kualitas data.
- Tinjau hasil kualitas data.

Untuk berlatih dengan contoh, tinjau posting blog Memulai dengan AWS Glue Data Quality untuk pipeline ETL.

Langkah 1: Tambahkan node transformasi Kualitas Data Evaluasi ke pekerjaan visual

Pada langkah ini, Anda menambahkan node Evaluate Data Quality ke editor pekerjaan visual.

Untuk menambahkan node kualitas data

1. Di konsol AWS Glue Studio, pilih Visual dengan sumber dan target dari bagian Buat pekerjaan, lalu pilih Buat.
2. Pilih node yang ingin Anda terapkan transformasi kualitas data. Biasanya, ini akan menjadi node transformasi atau sumber data.
3. Buka panel sumber daya di sebelah kiri dengan memilih ikon “+”. Kemudian cari Evaluasi Kualitas Data di bilah pencarian dan pilih Evaluasi Kualitas Data dari hasil pencarian.
4. Editor pekerjaan visual menampilkan Evaluate Data Quality transform node percabangan dari node yang Anda pilih. Di sisi kanan konsol, tab Transform dibuka secara otomatis. Jika Anda perlu mengubah node induk, pilih tab Properti Node, lalu pilih induk simpul dari menu tarik-turun.

   Ketika Anda memilih induk node baru, koneksi baru dibuat antara node induk dan node Evaluate Data Quality. Hapus semua node induk yang tidak diinginkan. Hanya satu node induk yang dapat dihubungkan ke satu node Evaluate Data Quality.
5. Transformasi Evaluasi Kualitas Data mendukung beberapa orang tua sehingga Anda dapat memvalidasi aturan kualitas data di beberapa kumpulan data. Aturan yang mendukung beberapa kumpulan data termasuk ReferentialIntegrity,,, DatasetMatch SchemaMatch RowCountMatch, dan. AggregateMatch

Saat Anda menambahkan beberapa input ke transformasi Evaluasi Kualitas Data, Anda harus memilih input “primer” Anda. Masukan utama Anda adalah kumpulan data yang ingin Anda validasi kualitas data. Semua node atau input lainnya diperlakukan sebagai referensi.

Anda dapat menggunakan transformasi Evaluasi Kualitas Data untuk mengidentifikasi catatan tertentu yang gagal memeriksa kualitas data. Kami menyarankan Anda memilih kumpulan data utama karena kolom baru yang menandai catatan buruk ditambahkan ke kumpulan data utama.

6. Anda dapat menentukan alias untuk sumber data input. Alias menyediakan cara lain untuk mereferensikan sumber input saat Anda menggunakan ReferentialIntegrity aturan. Karena hanya satu sumber data yang dapat ditetapkan sebagai sumber utama, setiap sumber data tambahan yang Anda tambahkan akan memerlukan alias.

   Dalam contoh berikut, ReferentialIntegrity aturan menentukan sumber data input dengan nama alias dan melakukan one-to-one perbandingan dengan sumber data primer.

   ```
   Rules = [
    ReferentialIntegrity “Aliasname.name” = 1
   ]
   ```

Langkah 2: Buat aturan menggunakan DQDL

Pada langkah ini, Anda membuat aturan menggunakan DQDL. Untuk tutorial ini, Anda membuat aturan tunggal menggunakan tipe aturan Completeness. Jenis aturan ini memeriksa persentase nilai lengkap (non-null) dalam kolom terhadap ekspresi yang diberikan. Untuk informasi selengkapnya tentang penggunaan DQDL, lihat DQDL.

1. Pada tab Transform, tambahkan tipe Rule dengan memilih tombol Insert. Ini menambahkan jenis aturan ke editor aturan, di mana Anda dapat memasukkan parameter untuk aturan.

   > **Note**
   >
   > Saat Anda mengedit aturan, pastikan aturan berada dalam tanda kurung dan aturan dipisahkan dengan koma. Misalnya, ekspresi aturan lengkap akan terlihat seperti berikut:
   >
   > ```
   > Rules= [
   >     Completeness "year">0.8, Completeness "month">0.8
   > ]
   > ```

Contoh ini menentukan parameter untuk kelengkapan untuk kolom bernama 'tahun' dan 'bulan'. Agar aturan dapat lulus, kolom ini harus lebih besar dari 80% 'lengkap', atau harus memiliki data di lebih dari 80% contoh untuk setiap kolom masing-masing.

Dalam contoh ini, cari dan masukkan tipe aturan Kelengkapan. Ini menambahkan jenis aturan ke editor aturan. Jenis aturan ini memiliki sintaks berikut:`Completeness <COL_NAME> <EXPRESSION>`.

Sebagian besar tipe aturan mengharuskan Anda memberikan ekspresi sebagai parameter untuk membuat respons Boolean. Untuk informasi selengkapnya tentang ekspresi DQDL yang didukung, lihat ekspresi DQDL. Selanjutnya, Anda akan menambahkan nama kolom.

2. Di pembuat aturan DQDL, pilih tab Skema. Gunakan bilah pencarian untuk menemukan nama kolom dalam skema input. Skema input menampilkan nama kolom dan tipe data.
3. Di editor aturan, klik di sebelah kanan jenis aturan untuk menyisipkan kursor tempat kolom akan dimasukkan. Bergantian, Anda dapat memasukkan nama kolom dalam aturan.

   Misalnya, dari daftar kolom dalam daftar skema input, pilih tombol Sisipkan di sebelah kolom (dalam contoh ini, tahun). Ini menambahkan kolom ke aturan.
4. Kemudian, di editor aturan, tambahkan ekspresi untuk mengevaluasi aturan. Karena tipe aturan Completeness memeriksa persentase nilai lengkap (non-null) dalam kolom terhadap ekspresi yang diberikan, masukkan ekspresi seperti. `> 0.8` Aturan ini memeriksa kolom jika lebih besar dari 80% nilai lengkap (non-null).

Langkah 3: Konfigurasikan output kualitas data

Setelah membuat aturan kualitas data, Anda dapat memilih opsi tambahan untuk menentukan output node kualitas data.

1. Dalam output transformasi kualitas data, pilih dari opsi berikut:

   - Data asli - Pilih untuk mengeluarkan data input asli. Saat Anda memilih opsi ini, simpul anak baru “rowLevelOutcomes” ditambahkan ke pekerjaan. Skema cocok dengan skema kumpulan data utama yang diteruskan sebagai input ke transformasi. Opsi ini berguna jika Anda hanya ingin meneruskan data dan gagal dalam pekerjaan ketika masalah kualitas terjadi.

Kasus penggunaan lainnya adalah ketika Anda ingin mendeteksi catatan buruk yang gagal memeriksa kualitas data. Untuk mendeteksi catatan buruk, pilih opsi Tambahkan kolom baru untuk menunjukkan kesalahan kualitas data. Tindakan ini menambahkan empat kolom baru ke skema transformasi “rowLevelOutcomes”.

- DataQualityRulesPass(string array) - Menyediakan array aturan yang lulus pemeriksaan kualitas data.
- DataQualityRulesFail(string array) - Menyediakan array aturan yang gagal pemeriksaan kualitas data.
- DataQualityRulesSkip(string array) - Menyediakan array aturan yang dilewati. Aturan berikut tidak dapat mengidentifikasi catatan kesalahan karena diterapkan pada tingkat kumpulan data.
  - AggregateMatch
  - ColumnCount
  - ColumnExists
  - ColumnNamesMatchPattern
  - CustomSql
  - RowCount
  - RowCountMatch
  - StandardDeviation
  - Berarti
  - ColumnCorrelation
- DataQualityEvaluationResult— Menyediakan status “Lulus” atau “Gagal” di tingkat baris. Perhatikan bahwa hasil keseluruhan Anda bisa GAGAL, tetapi catatan tertentu mungkin lolos. Misalnya, RowCount aturan mungkin gagal, tetapi semua aturan lain mungkin berhasil. Dalam kasus seperti itu, status bidang ini adalah 'Lulus'.

2. Hasil kualitas data — Pilih untuk menampilkan aturan yang dikonfigurasi dan status lulus atau gagalnya. Opsi ini berguna jika Anda ingin menulis hasil Anda ke Amazon S3 atau database lainnya.
3. Pengaturan keluaran kualitas data (Opsional) - Pilih pengaturan keluaran kualitas data untuk mengungkapkan bidang lokasi hasil kualitas data. Kemudian, pilih Browse untuk mencari lokasi Amazon S3 untuk ditetapkan sebagai target output kualitas data.

Langkah 4. Konfigurasikan tindakan kualitas data

Anda dapat menggunakan tindakan untuk mempublikasikan metrik ke CloudWatch atau menghentikan pekerjaan berdasarkan kriteria tertentu. Tindakan hanya tersedia setelah Anda membuat aturan. Saat Anda memilih opsi ini, metrik yang sama juga dipublikasikan keAmazon EventBridge. Anda dapat menggunakan opsi ini untuk membuat peringatan untuk pemberitahuan.

- Pada kegagalan ruleset — Anda dapat memilih apa yang harus dilakukan jika kumpulan aturan gagal saat pekerjaan sedang berjalan. Jika Anda ingin pekerjaan gagal jika kualitas data gagal, pilih kapan pekerjaan harus gagal dengan memilih salah satu opsi berikut. Secara default, tindakan ini tidak dipilih, dan pekerjaan menyelesaikan operasinya meskipun aturan kualitas data gagal.
  - None - Jika Anda memilih None (default), pekerjaan tidak gagal dan terus berjalan meskipun ruleset gagal.
  - Gagal pekerjaan setelah memuat data ke target - Pekerjaan gagal dan tidak ada data yang disimpan. Untuk menyimpan hasil, pilih lokasi Amazon S3 di mana hasil kualitas data akan disimpan.
  - Gagal pekerjaan tanpa memuat ke data target - Opsi ini gagal pekerjaan segera ketika kesalahan kualitas data terjadi. Itu tidak memuat target data apa pun, termasuk hasil dari transformasi kualitas data.

Langkah 5: Lihat hasil kualitas data

Setelah menjalankan pekerjaan, lihat hasil kualitas data dengan memilih tab Kualitas data.

1. Untuk setiap pekerjaan yang dijalankan, lihat hasil kualitas data. Setiap node menampilkan status kualitas data dan detail status. Pilih node untuk melihat semua aturan dan status setiap aturan.
2. Pilih Unduh hasil untuk mengunduh file CSV yang berisi informasi tentang pekerjaan dan hasil kualitas data.
3. Jika Anda memiliki lebih dari satu pekerjaan yang dijalankan dengan hasil kualitas data, Anda dapat memfilter hasil berdasarkan tanggal dan rentang waktu. Pilih Filter berdasarkan tanggal dan rentang waktu untuk memperluas jendela filter.
4. Pilih rentang relatif atau rentang absolut. Untuk rentang absolut, gunakan kalender untuk memilih tanggal, dan masukkan nilai untuk waktu mulai dan waktu akhir. Setelah selesai, pilih Terapkan.

Pembuat aturan Kualitas Data

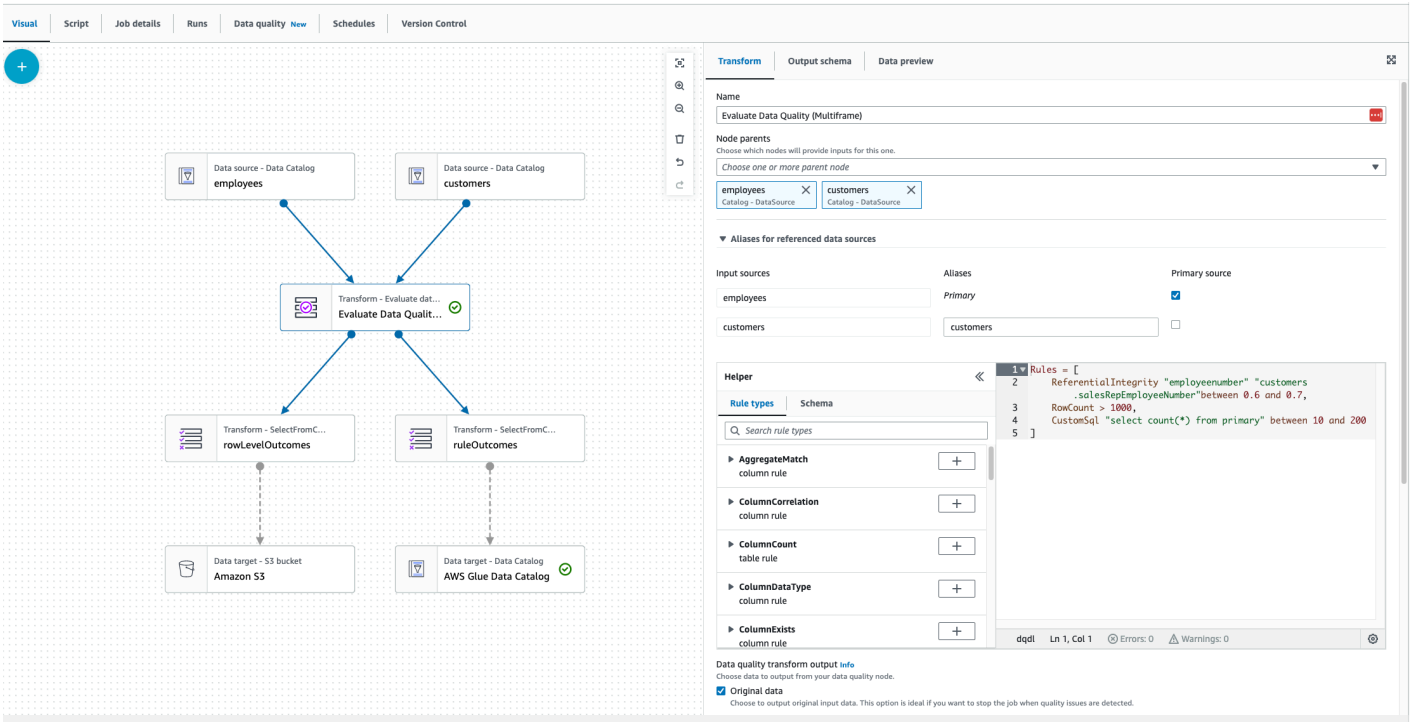
Dengan pembuat aturan Data Quality Definition Language (DQDL), Anda dapat membuat aturan kualitas data untuk mengevaluasi data Anda. Mulailah dengan memilih jenis aturan, lalu tentukan parameter di editor aturan. Editor aturan juga menunjukkan kesalahan dan peringatan apa pun saat Anda membuat aturan.

Panduan DQDL menyediakan dokumentasi komprehensif tentang cara membuat aturan menggunakan sintaks DQDL, tipe aturan bawaan, dan contoh.

Mengevaluasi node Kualitas Data

Saat Anda bekerja dengan node transformasi Evaluate Data Quality dan pembuat aturan DQDL, Anda dapat memperluas ruang kerja.

- Untuk memperluas tab Transform untuk mengisi seluruh layar, pilih ikon perluas di sudut kanan atas panel detail simpul.
- Untuk memperluas editor aturan DQDL, pilih ikon << untuk memperluas editor aturan dan menciutkan tab Rule types dan Schema.



Komponen

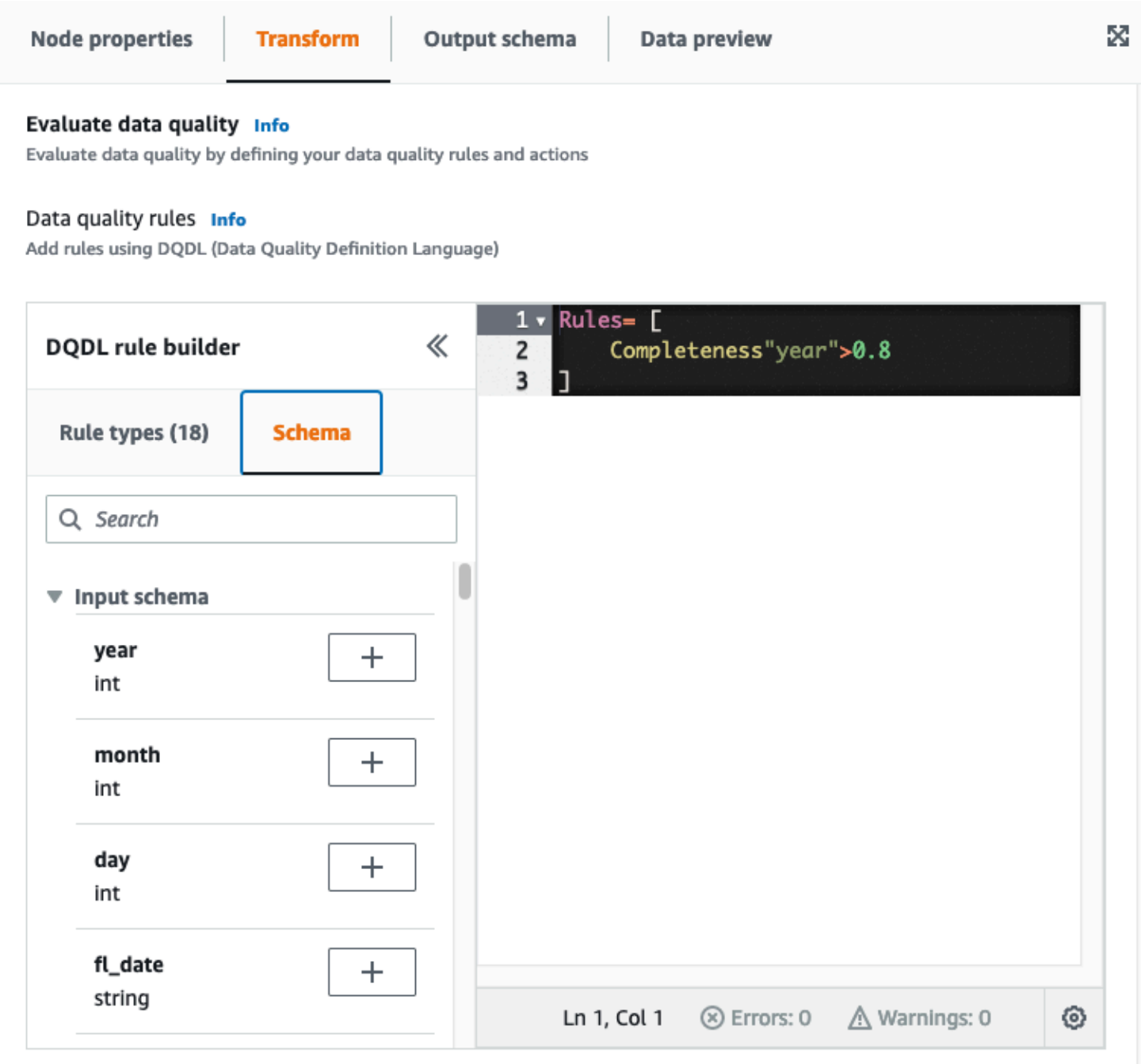
Ada 26 jenis aturan yang dibangun ke dalamAWS Glue Studio. Setiap jenis aturan memiliki deskripsi dan contoh bagaimana mereka dapat digunakan.

Jenis aturan kualitas data

AWS Glue Studiomenyediakan tipe aturan bawaan untuk kemudahan dalam membuat aturan. Untuk informasi selengkapnya tentang jenis aturan, lihat referensi tipe aturan DQDL.

Skema

Tab Skema menampilkan nama kolom dan tipe data dari node induk. Skema dari beberapa node ditampilkan. Anda dapat melihat skema input, mencari berdasarkan nama kolom, dan menyisipkan kolom ke editor aturan.



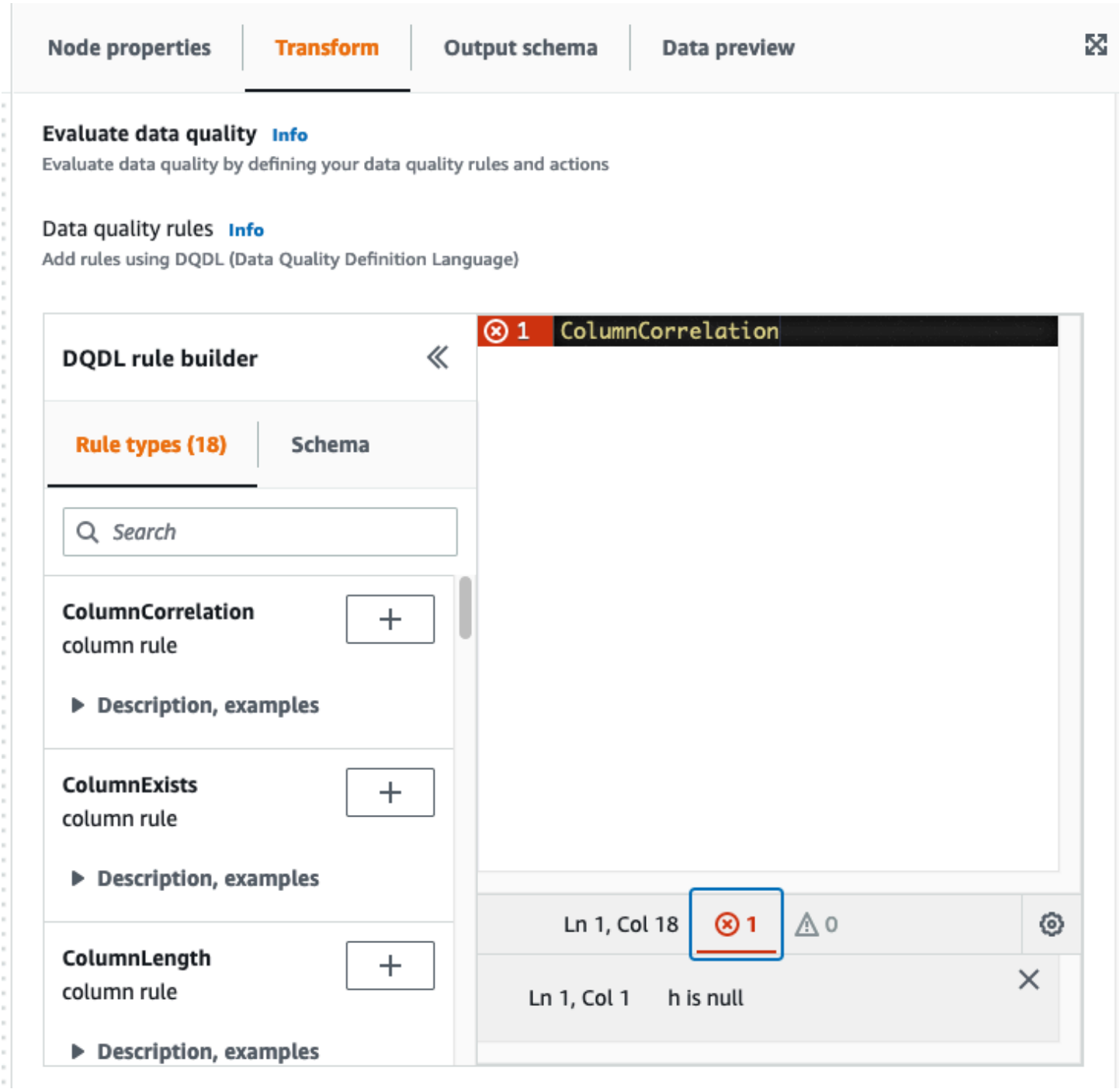
Editor aturan

Editor aturan adalah editor teks tempat Anda dapat menulis dan mengedit aturan. Jika Anda memilih jenis aturan dari pembuat aturan DQDL, jenis aturan ditambahkan ke editor aturan. Anda kemudian dapat menentukan parameter, menambahkan aturan, dan mengedit aturan sesuai kebutuhan dengan memodifikasi teks. AWS Glue Studiomemvalidasi aturan di editor aturan dan menampilkan kesalahan dan peringatan jika ada.

Kesalahan dan peringatan

Jika aturan tidak mengikuti sintaks aturan DQDL, editor aturan menunjukkan beberapa indikator visual bahwa ada kesalahan:

- Editor aturan menampilkan ikon kesalahan dan warna merah pada baris dengan kesalahan.
- Editor aturan menampilkan jumlah kesalahan di sebelah ikon kesalahan merah.
- Ketika Anda memilih baris dengan kesalahan, deskripsi kesalahan dan lokasi (baris dan kolom) ditampilkan di bagian bawah editor aturan.



Tindakan kualitas data

Secara default, tindakan ini tidak dipilih dan pekerjaan akan menyelesaikan jalannya bahkan jika aturan kualitas data gagal.

Pilih di antara tindakan berikut. Anda dapat menggunakan tindakan untuk mempublikasikan hasil CloudWatch atau menghentikan pekerjaan berdasarkan kriteria tertentu. Tindakan hanya tersedia setelah Anda membuat aturan.

- Publikasikan hasil ke CloudWatch — Saat Anda menjalankan pekerjaan, tambahkan hasilnya ke CloudWatch.
- Gagal pekerjaan ketika kualitas data gagal - Jika aturan kualitas data gagal, pekerjaan juga akan gagal sebagai hasilnya.

Output transformasi kualitas data

- Data asli - Pilih untuk mengeluarkan data input asli. Opsi ini sangat ideal jika Anda ingin menghentikan pekerjaan ketika masalah kualitas terdeteksi.
- Metrik kualitas data — Pilih untuk menampilkan aturan yang dikonfigurasi dan status lulus atau gagalnya. Opsi ini berguna jika Anda ingin mengambil tindakan khusus.

Pengaturan output kualitas data

Tetapkan lokasi hasil kualitas data dengan menentukan lokasi Amazon S3 sebagai target keluaran kualitas data.

Mengkonfigurasi deteksi Anomali dan menghasilkan wawasan

AWS GlueKualitas Data (DQ) mengevaluasi data Anda berdasarkan aturan kualitas data yang Anda tulis dan memberikan wawasan dan pengamatan tentang data Anda dari waktu ke waktu sehingga Anda dapat mengambil tindakan segera. Karena DQ memindai data Anda, DQ menghitung metrik statistik seperti jumlah baris, maksimum atau minimum, dan kemudian membandingkannya dengan ekspresi ambang batas.

Beberapa manfaat deteksi anomali Kualitas Data meliputi:

- pemindaian data otomatis terus menerus
- deteksi anomali yang dapat menjadi indikasi peristiwa yang tidak diinginkan atau kelainan statistik
- menawarkan Rekomendasi Aturan untuk mengambil tindakan pada pengamatan yang ditemukan oleh deteksi anomali Kualitas Data

Ini berguna jika Anda:

- ingin mendeteksi anomali pada data Anda secara otomatis, tanpa perlu menulis kualitas data
- ingin membuat profil data Anda dan melihat representasi visual seperti apa data itu
- ingin melacak bagaimana data Anda berubah dari waktu ke waktu

Pengamatan apa yang dapat saya lihat tentang data saya?

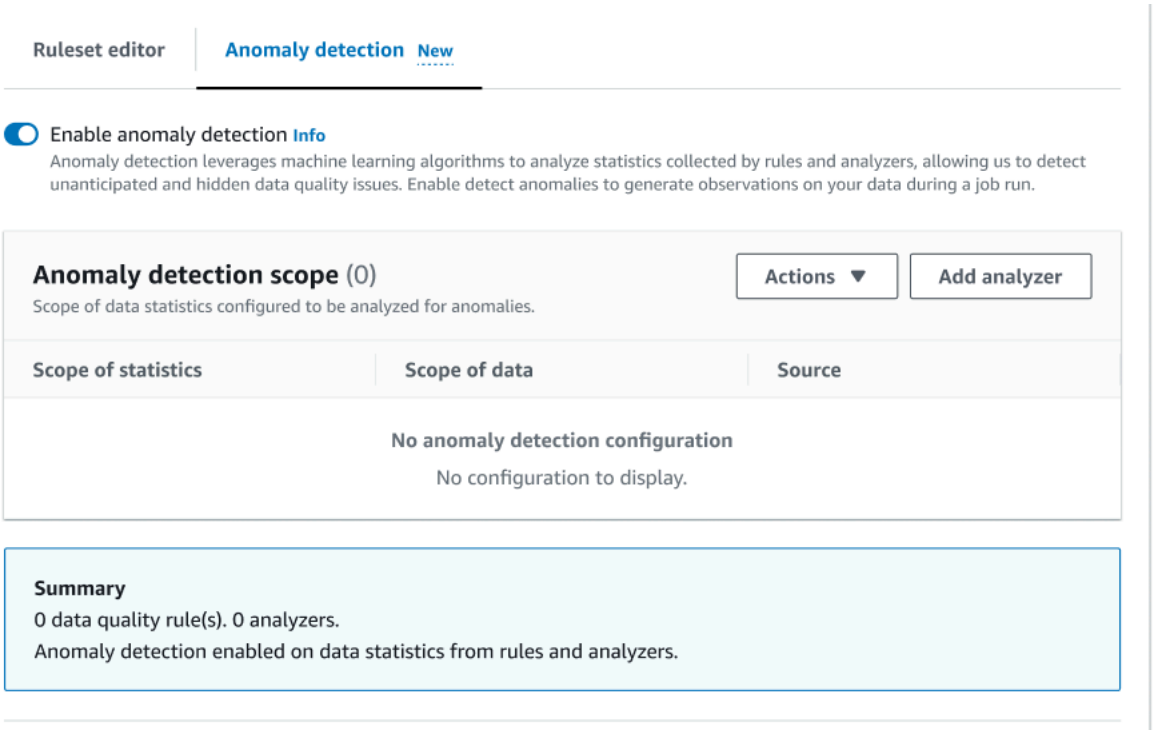
DQ mengidentifikasi outlier dalam statistik data yang dikumpulkan, perubahan format data, penyimpangan data, dan perubahan skema. Berdasarkan pengamatan, DQ merekomendasikan aturan kualitas data yang dapat dengan mudah dioperasionalkan oleh pengguna. Statistik meliputi Kelengkapan, Keunikan, Mean, Jumlah,, Entropi, StandardDeviation, dan. DistinctValuesCount UniqueValueRatio

Mengaktifkan deteksi anomali di AWS Glue Studio

Untuk mengaktifkan deteksi anomali, Anda dapat membuka AWS Glue Studio pekerjaan dan mengaktifkan “Aktifkan Deteksi Anomali”. Mengaktifkan ini memungkinkan deteksi anomali pada data Anda dengan menganalisis data Anda dari waktu ke waktu dan memberikan statistik data tentang data dan pengamatan yang dapat Anda lakukan.

Untuk mengaktifkan deteksi anomali di: AWS Glue Studio

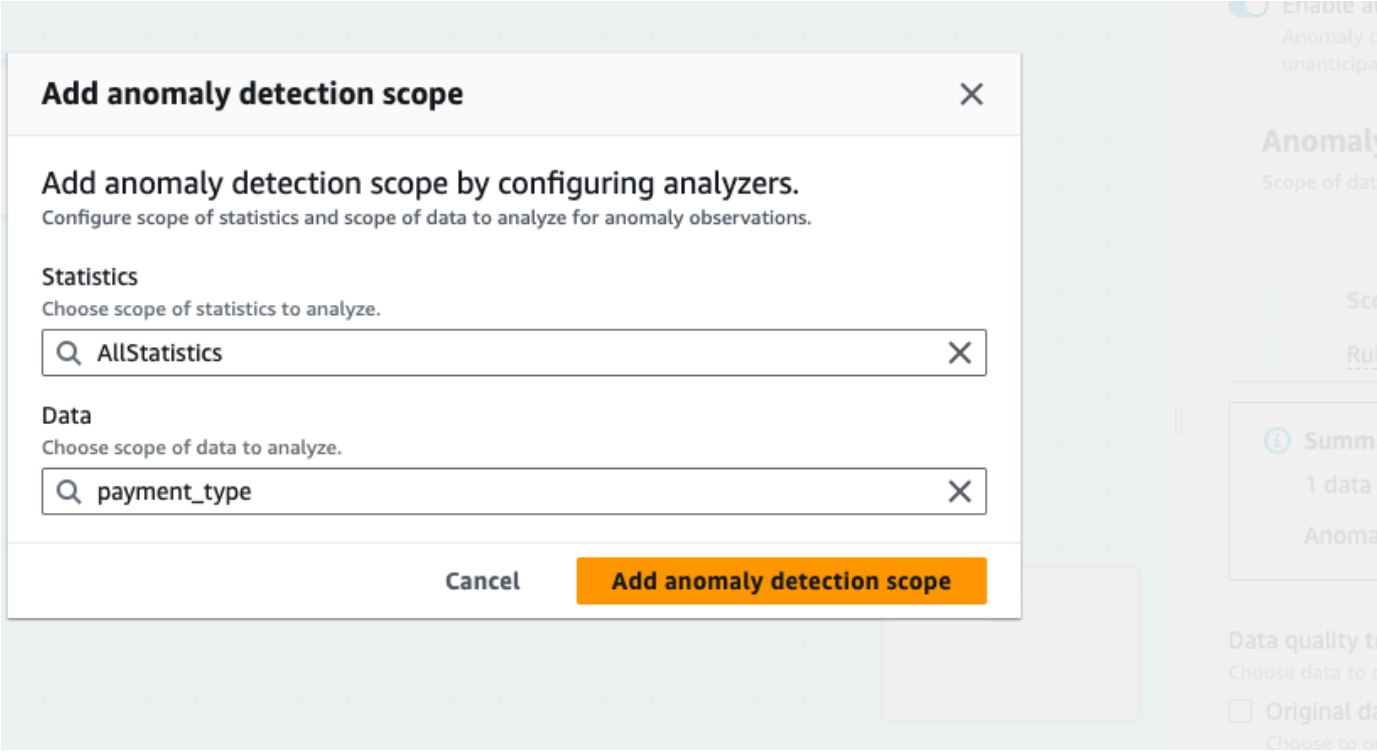
1. Pilih node Kualitas Data di pekerjaan Anda, lalu pilih tab Deteksi anomali. Aktifkan 'Aktifkan Deteksi Anomali'.



2. Tentukan data untuk memantau anomali dengan memilih Add analyzer. Ada dua bidang yang dapat Anda isi: Statistik dan Data.

   Statistik adalah informasi tentang bentuk data Anda dan properti lainnya. Anda dapat memilih satu atau lebih statistik sekaligus, atau memilih Semua statistik. Statistik meliputi: Kelengkapan, Keunikan, Mean, Jumlah,, Entropi, StandardDeviation, dan. DistinctValuesCount UniqueValueRatio

   Data adalah kolom dalam kumpulan data Anda. Anda dapat memilih semua kolom atau kolom individual.



3. Pilih Tambahkan cakupan deteksi anomali untuk menyimpan perubahan Anda. Saat Anda membuat penganalisis, Anda dapat melihatnya di bagian cakupan deteksi anomali.

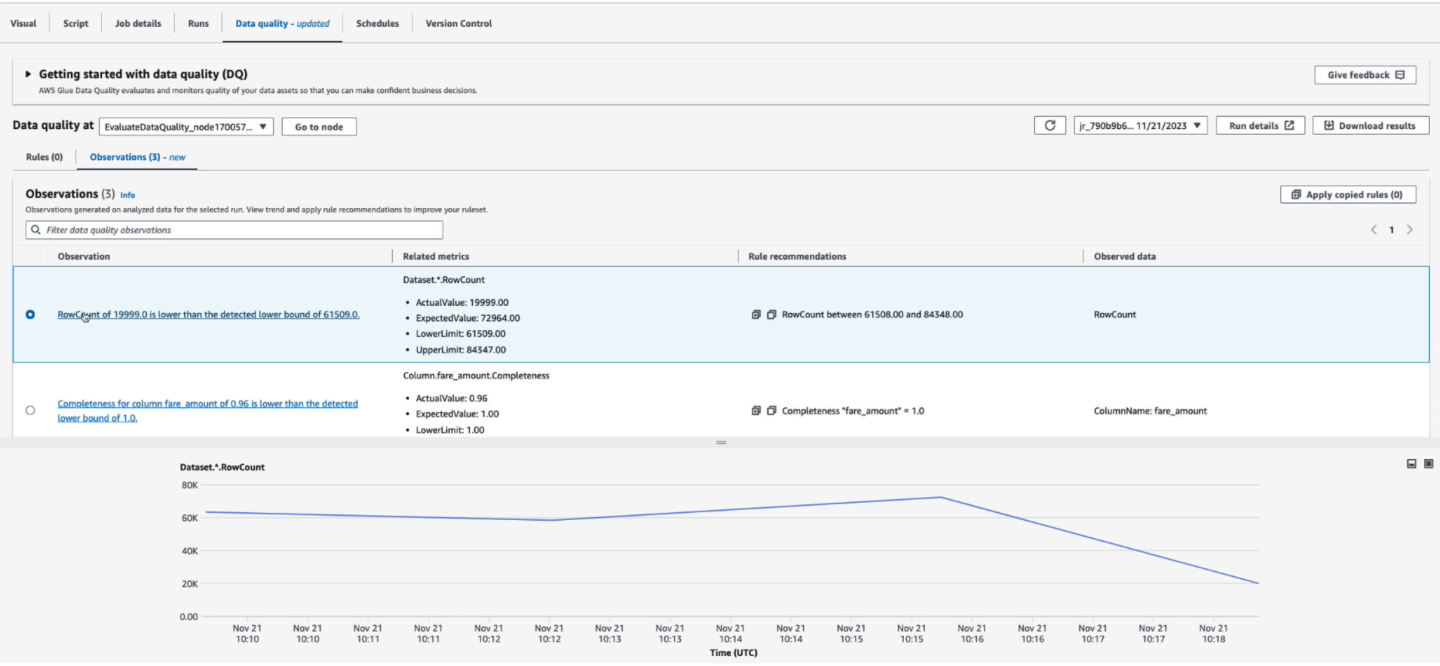
   Anda juga dapat menggunakan menu Tindakan untuk mengedit penganalisis Anda, atau memilih tab editor Ruleset dan mengedit penganalisis langsung di notepad editor ruleset. Anda akan melihat penganalisis yang Anda simpan tepat di bawah aturan apa pun yang telah Anda buat.

```
Rules = [
]

Analyzers = [
    Completeness “id”
]
```

   Dengan aturan yang diperbarui bersama dengan penganalisis, Kualitas Data terus memantau data yang masuk, menandakan anomali melalui peringatan atau penghentian pekerjaan berdasarkan pengaturan Anda.

> **Note**
>
> Pengamatan dihasilkan ketika minimal tiga nilai per statistik data diamati dalam kumpulan data Anda. Jika tidak ada pengamatan yang terlihat, kualitas data tidak memiliki cukup data untuk menghasilkan pengamatan. Setelah beberapa pekerjaan berjalan, kualitas data dapat memberikan wawasan tentang data Anda dan akan menampilkannya di bagian Pengamatan.

Analyzer menghasilkan pengamatan dengan mendeteksi anomali dalam data Anda dan memberi Anda rekomendasi untuk membangun aturan secara progresif. Anda dapat melihat pengamatan dengan memilih tab Kualitas Data. Pengamatan khusus untuk setiap pekerjaan yang dijalankan. Anda dapat melihat node Kualitas Data tertentu dan pekerjaan yang dijalankan di bagian atas bagian Observasi. Pilih node baru atau job run untuk melihat pengamatan khusus untuk node dan pekerjaan itu.



Pengamatan — setiap wawasan didasarkan pada pekerjaan tertentu yang dikonfigurasi oleh kumpulan aturan dan penganalisis yang Anda tentukan.

Metrik terkait — Saat pengamatan dibuat, kolom metrik Terkait menunjukkan aturan dan nilai aktual dan yang diharapkan, serta batas bawah dan atas.

Rekomendasi aturan — AWS Glue kemudian juga merekomendasikan aturan untuk mengatasi hal ini. Setiap aturan yang direkomendasikan dapat disalin dengan mengklik ikon salin. Anda dapat

menyalin semua aturan yang disarankan dengan mengklik ikon salin di samping setiap aturan, lalu mengklik Terapkan aturan yang disalin.

Data yang dipantau - Kolom data yang dipantau menyediakan kolom atau baris yang dipantau dan memicu pengamatan.

Menerapkan aturan rekomendasi ke node kualitas Data Anda

Setelah pengamatan dibuat dan aturan yang direkomendasikan disediakan, Anda dapat menerapkan aturan itu ke node kualitas data Anda. Untuk melakukannya:

1. Klik ikon salin di samping setiap rekomendasi aturan. Ini akan menambahkan rekomendasi aturan ke notepad yang dapat Anda ambil nanti.
2. Klik Terapkan rekomendasi aturan. Ini membuka notepad tempat Anda dapat melihat aturan yang sebelumnya Anda salin.
3. Pilih aturan Salin.
4. Pilih Terapkan ke editor ruleset. Ini membuka editor ruleset tempat Anda dapat menempelkan aturan yang disalin.
5. Tempelkan aturan yang disalin ke editor ruleset.

Kualitas Data untuk pekerjaan ETL di notebook AWS Glue Studio

Dalam tutorial ini, Anda mempelajari cara menggunakan Kualitas AWS Glue Data untuk mengekstrak, mengubah, dan memuat (ETL) pekerjaan di AWS Glue Studio notebook.

Anda dapat menggunakan buku catatan AWS Glue Studio untuk mengedit skrip pekerjaan dan melihat output tanpa harus menjalankan pekerjaan penuh. Anda juga dapat menambahkan penurunan harga dan menyimpan buku catatan sebagai `.ipynb` file dan skrip pekerjaan. Perhatikan bahwa Anda dapat memulai notebook tanpa menginstal perangkat lunak secara lokal atau mengelola server. Ketika Anda puas dengan kode Anda, Anda dapat menggunakannya AWS Glue Studio untuk dengan mudah mengonversi buku catatan Anda ke AWS Glue pekerjaan.

Kumpulan data yang Anda gunakan dalam contoh ini terdiri dari data pembayaran Penyedia Medicare yang diunduh dari dua kumpulan data data.cms.gov: “Ringkasan Penyedia Sistem Pembayaran Prospektif Rawat Inap untuk 100 Grup Terkait Diagnosis Teratas - FY2011" dan “Data Biaya Rawat Inap TA 2011".

Setelah mengunduh data, kami memodifikasi kumpulan data untuk memperkenalkan beberapa catatan yang salah di akhir file. File yang dimodifikasi ini terletak di bucket Amazon S3 publik di `s3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv`.

Prasyarat

- AWS Glueperan dengan izin Amazon S3 untuk menulis ke bucket Amazon S3 tujuan Anda
- Notebook baru (lihat Memulai notebook di AWS Glue Studio)

Membuat pekerjaan ETL di AWS Glue Studio

Untuk membuat pekerjaan ETL

1. Ubah versi sesi ke AWS Glue 3.0.

   Untuk melakukan ini, hapus semua sel kode boilerplate dengan sihir berikut dan jalankan sel. Perhatikan bahwa kode boilerplate ini secara otomatis disediakan di sel pertama saat notebook baru dibuat.

   ```
   %glue_version 3.0
   ```

2. Salin kode berikut dan jalankan di sel.

   ```
   import sys
   from awsglue.transforms import *
   from awsglue.utils import getResolvedOptions
   from pyspark.context import SparkContext
   from awsglue.context import GlueContext
   from awsglue.job import Job

   sc = SparkContext.getOrCreate()
   glueContext = GlueContext(sc)
   spark = glueContext.spark_session
   job = Job(glueContext)
   ```

3. Di sel berikutnya, impor `EvaluateDataQuality` kelas yang mengevaluasi Kualitas AWS Glue Data.

   ```
   from awsgluedq.transforms import EvaluateDataQuality
   ```

4. Di sel berikutnya, baca data sumber menggunakan file.csv yang disimpan di bucket Amazon S3 publik.

```
medicare = spark.read.format(
"csv").option(
"header", "true").option(
"inferSchema", "true").load(
's3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv')
medicare.printSchema()
```

5. Konversikan data ke file AWS Glue DynamicFrame.

```
from awsglue.dynamicframe import DynamicFrame
medicare_dyf = DynamicFrame.fromDF(medicare,glueContext,"medicare_dyf")
```

6. Buat kumpulan aturan menggunakan Data Quality Definition Language (DQDL).

```
EvaluateDataQuality_ruleset = """
    Rules = [
        ColumnExists "Provider Id",
        IsComplete "Provider Id",
        ColumnValues  " Total Discharges " > 15
]
    ]
"""
```

7. Validasi kumpulan data terhadap kumpulan aturan.

```
EvaluateDataQualityMultiframe = EvaluateDataQuality().process_rows(
    frame=medicare_dyf,
    ruleset=EvaluateDataQuality_ruleset,
    publishing_options={
        "dataQualityEvaluationContext": "EvaluateDataQualityMultiframe",
        "enableDataQualityCloudWatchMetrics": False,
        "enableDataQualityResultsPublishing": False,
    },
    additional_options={"performanceTuning.caching": "CACHE_NOTHING"},
)
```

8. Tinjau hasilnya.

```
ruleOutcomes = SelectFromCollection.apply(
    dfc=EvaluateDataQualityMultiframe,
    key="ruleOutcomes",
    transformation_ctx="ruleOutcomes",
)

ruleOutcomes.toDF().show(truncate=False)
```

Output:

```
--------------------------------------+-------
+-----------------------------------------------------
+-----------------------------------------+
|Rule                                   |Outcome|FailureReason
                  |EvaluatedMetrics                           |
+--------------------------------------+-------
+-----------------------------------------------------
+-----------------------------------------+
|ColumnExists "Provider Id"             |Passed |null
                  |{}                                         |
|IsComplete "Provider Id"               |Passed |null
                  |{Column.Provider Id.Completeness -> 1.0}   |
|ColumnValues " Total Discharges " > 15|Failed |Value: 11.0 does not meet the
 constraint requirement!|{Column. Total Discharges .Minimum -> 11.0}|
+--------------------------------------+-------
+-----------------------------------------------------
+-----------------------------------------+
```

9. Filter baris yang diteruskan dan tinjau baris yang gagal dari hasil tingkat baris Kualitas Data.

```
owLevelOutcomes = SelectFromCollection.apply(
dfc=EvaluateDataQualityMultiframe,
key="rowLevelOutcomes",
transformation_ctx="rowLevelOutcomes",
)
```

```
rowLevelOutcomes_df = rowLevelOutcomes.toDF() # Convert Glue DynamicFrame to
 SparkSQL DataFrame
rowLevelOutcomes_df_passed =
 rowLevelOutcomes_df.filter(rowLevelOutcomes_df.DataQualityEvaluationResult ==
 "Passed") # Filter only the Passed records.
rowLevelOutcomes_df.filter(rowLevelOutcomes_df.DataQualityEvaluationResult ==
 "Failed").show(5, truncate=False) # Review the Failed records
```

Output:

```
+----------------------------------------+-----------
+-------------------------------------+-------------------------+------------
+-------------+----------------+-----------------------------------
+-----------------+-------------------------+----------------------
+-------------------------+-------------------------
+-----------------------------------------+--------------------------
+--------------------------+
|DRG Definition                           |Provider Id|Provider Name
           |Provider Street Address   |Provider City|Provider State|Provider Zip
 Code|Hospital Referral Region Description| Total Discharges | Average Covered
 Charges | Average Total Payments |Average Medicare Payments|DataQualityRulesPass
      |DataQualityRulesFail                    |DataQualityRulesSkip        |
DataQualityEvaluationResult|
+----------------------------------------+-----------
+-------------------------------------+-------------------------+------------
+-------------+----------------+-----------------------------------
+-----------------+-------------------------+----------------------
+-------------------------+-------------------------
+-----------------------------------------+--------------------------
+--------------------------+
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|10005      |MARSHALL MEDICAL CENTER SOUTH
        |2505 U S HIGHWAY 431 NORTH|BOAZ         |AL            |35957
 |AL - Birmingham                      |14               |$15131.85
  |$5787.57               |$4976.71                |[IsComplete "Provider Id"]|
[ColumnValues " Total Discharges " > 15]|[ColumnExists "Provider Id"]|Failed
              |
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|10046      |RIVERVIEW REGIONAL MEDICAL
 CENTER    |600 SOUTH THIRD STREET    |GADSDEN      |AL            |35901
   |AL - Birmingham                      |14               |$67327.92
   |$5461.57               |$4493.57                |[IsComplete "Provider Id"]|
[ColumnValues " Total Discharges " > 15]|[ColumnExists "Provider Id"]|Failed
              |
```

```
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|10083      |SOUTH BALDWIN REGIONAL
 MEDICAL CENTER|1613 NORTH MCKENZIE STREET|FOLEY        |AL            |36535
        |AL - Mobile                          |15               |$25411.33
         |$5282.93               |$4383.73                 |[[IsComplete "Provider
 Id"]][ColumnValues " Total Discharges " > 15]][ColumnExists "Provider Id"]]|Failed
                    |
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|30002      |BANNER GOOD SAMARITAN MEDICAL
 CENTER |1111 EAST MCDOWELL ROAD   |PHOENIX      |AZ            |85006
 |AZ - Phoenix                         |11               |$34803.81
  |$7768.90               |$6951.45                 |[[IsComplete "Provider Id"]]|
[ColumnValues " Total Discharges " > 15]][ColumnExists "Provider Id"]]|Failed
              |
|039 - EXTRACRANIAL PROCEDURES W/O CC/MCC|30010      |CARONDELET ST  MARYS HOSPITAL
        |1601 WEST ST MARY'S ROAD  |TUCSON       |AZ            |85745
 |AZ - Tucson                          |12               |$35968.50
  |$6506.50               |$5379.83                 |[[IsComplete "Provider Id"]]|
[ColumnValues " Total Discharges " > 15]][ColumnExists "Provider Id"]]|Failed
              |
+---------------------------------------+-----------
+-----------------------------------+-------------------------+-------------
+--------------+-----------------+-----------------------------------
+------------------+-------------------------+----------------------
+-------------------------+-------------------------
+---------------------------------------+---------------------------
+--------------------------+
only showing top 5 rows
```

Perhatikan bahwa AWS Glue Data Quality menambahkan empat kolom baru (DataQualityRulesPass DataQualityRulesFail, DataQualityRulesSkip,, dan DataQualityEvaluationResult). Ini menunjukkan catatan yang lulus, catatan yang gagal, aturan yang dilewati untuk evaluasi tingkat baris, dan hasil tingkat baris keseluruhan.

10. Tulis output ke bucket Amazon S3 untuk menganalisis data dan memvisualisasikan hasilnya.

```
#Write the Passed records to the destination.

glueContext.write_dynamic_frame.from_options(
       frame = rowLevelOutcomes_df_passed,
       connection_type = "s3",
       connection_options = {"path": "s3://glue-sample-target/output-dir/
medicare_parquet"},
       format = "parquet")
```

Referensi Bahasa Definisi Kualitas Data (DQDL)

Data Quality Definition Language (DQDL) adalah bahasa khusus domain yang Anda gunakan untuk menentukan aturan untuk AWS Glue Data Quality.

Panduan ini memperkenalkan konsep kunci DQDL untuk membantu Anda memahami bahasa. Ini juga menyediakan referensi untuk jenis aturan DQDL dengan sintaks dan contoh. Sebelum Anda menggunakan panduan ini, kami menyarankan Anda untuk membiasakan diri dengan AWS Glue Data Quality. Untuk informasi selengkapnya, lihat AWS Glue Kualitas Data.

> **Note**
>
> DynamicRules hanya didukung di AWS Glue ETL.

Daftar Isi

- ColumnLength
- ColumnNamesMatchPattern
- ColumnValues
- Kelengkapan
- CustomSQL
- DataFreshness
- DatasetMatch
- DistinctValuesCount
- Entropi
- IsComplete
- IsPrimaryKey
- IsUnique
- Berarti
- ReferentialIntegrity
- RowCount
- RowCountMatch
- StandardDeviation
- Jumlah
- SchemaMatch
- Keunikan
- UniqueValueRatio
- DetectAnomalies

Sintaks DQDL

Dokumen DQDL peka huruf besar/kecil dan berisi kumpulan aturan, yang mengelompokkan aturan kualitas data individu bersama-sama. Untuk membuat kumpulan aturan, Anda harus membuat daftar bernama `Rules` (huruf besar), dibatasi oleh sepasang tanda kurung siku. Daftar harus berisi satu atau lebih aturan DQDL yang dipisahkan koma seperti contoh berikut.

```
Rules = [
    IsComplete "order-id",
```

```
    IsUnique "order-id"
]
```

Struktur aturan

Struktur aturan DQDL tergantung pada jenis aturan. Namun, aturan DQDL umumnya sesuai dengan format berikut.

```
<RuleType> <Parameter> <Parameter> <Expression>
```

`RuleType`adalah nama case-sensitive dari jenis aturan yang ingin Anda konfigurasikan. Sebagai contoh, `IsComplete`, `IsUnique`, atau `CustomSql`. Parameter aturan berbeda untuk setiap jenis aturan. Untuk referensi lengkap jenis aturan DQDL dan parameternya, lihat. Referensi tipe aturan DQDL

Aturan komposit

DQDL mendukung operator logis berikut yang dapat Anda gunakan untuk menggabungkan aturan. Aturan ini disebut aturan komposit.

and

> `and`Operator logis menghasilkan `true` if dan hanya jika aturan yang menghubungkannya`true`. Jika tidak, aturan gabungan menghasilkan`false`. Setiap aturan yang Anda hubungkan dengan `and` operator harus dikelilingi oleh tanda kurung.
>
> Contoh berikut menggunakan `and` operator untuk menggabungkan dua aturan DQDL.
>
> ```
> (IsComplete "id") and (IsUnique "id")
> ```

atau

> `or`Operator logis menghasilkan `true` jika dan hanya jika satu atau lebih aturan yang menghubungkannya`true`. Setiap aturan yang Anda hubungkan dengan `or` operator harus dikelilingi oleh tanda kurung.
>
> Contoh berikut menggunakan `or` operator untuk menggabungkan dua aturan DQDL.
>
> ```
> (RowCount "id" > 100) or (IsPrimaryKey "id")
> ```

Anda dapat menggunakan operator yang sama untuk menghubungkan beberapa aturan, sehingga kombinasi aturan berikut diperbolehkan.

```
(Mean "Star_Rating" > 3) and (Mean "Order_Total" > 500) and (IsComplete "Order_Id")
```

Namun, Anda tidak dapat menggabungkan operator logis menjadi satu ekspresi. Misalnya, kombinasi berikut tidak diperbolehkan.

```
(Mean "Star_Rating" > 3) and (Mean "Order_Total" > 500) or (IsComplete "Order_Id")
```

Cara kerja aturan Komposit

Secara default, Aturan Komposit dievaluasi sebagai aturan individual di seluruh kumpulan data atau tabel dan kemudian hasilnya digabungkan. Dengan kata lain, ini mengevaluasi seluruh kolom terlebih dahulu dan kemudian menerapkan operator. Perilaku default ini dijelaskan di bawah ini dengan contoh:

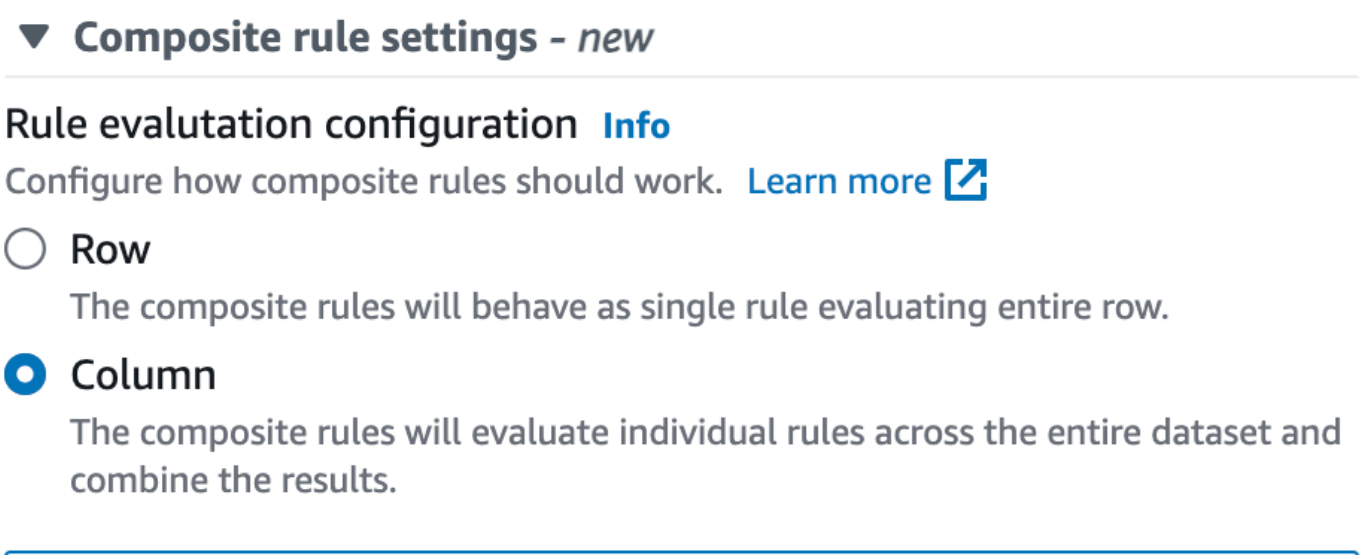
```
# Dataset

+------+------+
|myCol1|myCol2|
+------+------+
|     2|     1|
|     0|     3|
+------+------+

# Overall outcome

+-------------------------------------------------------------+-------+
|Rule                                                         |Outcome|
+-------------------------------------------------------------+-------+
|(ColumnValues "myCol1" > 1) OR (ColumnValues "myCol2" > 2)|Failed |
+-------------------------------------------------------------+-------+
```

Dalam contoh di atas, AWS Glue Data Quality pertama mengevaluasi `(ColumnValues "myCol1" > 1)` yang akan mengakibatkan kegagalan. Maka akan mengevaluasi `(ColumnValues "myCol2" > 2)` mana yang juga akan gagal. Kombinasi kedua hasil akan dicatat sebagai GAGAL.

Namun, jika Anda lebih suka perilaku seperti SQL, di mana Anda memerlukan seluruh baris untuk dievaluasi, Anda harus secara eksplisit mengatur `ruleEvaluation.scope` parameter seperti yang ditunjukkan pada cuplikan kode `additionalOptions` di bawah ini.

```
object GlueApp {
  val datasource = glueContext.getCatalogSource(
    database="<db>",
    tableName="<table>",
    transformationContext="datasource"
  ).getDynamicFrame()

  val ruleset = """
    Rules = [
        (ColumnValues "age" >= 26) OR (ColumnLength "name" >= 4)
    ]
  """

  val dq_results = EvaluateDataQuality.processRows(
    frame=datasource,
    ruleset=ruleset,
    additionalOptions=JsonOptions("""
        {
          "compositeRuleEvaluation.method":"ROW"
        }
      """
    )
  )
}
```

Di AWS Glue Studio dan AWS Glue Data Catalog, Anda dapat dengan mudah mengatur opsi ini di antarmuka pengguna seperti yang ditunjukkan di bawah ini.



Setelah ditetapkan, aturan komposit akan berperilaku sebagai aturan tunggal yang mengevaluasi seluruh baris. Contoh berikut menggambarkan perilaku ini.

```
# Row Level outcome

+------+------+-------------------------------------------------------------
+---------------------------+
|myCol1|myCol2|DataQualityRulesPass                                         |
DataQualityEvaluationResult|
+------+------+-------------------------------------------------------------
+---------------------------+
|2     |1     |[(ColumnValues "myCol1" > 1) OR (ColumnValues "myCol2" > 2)]|Passed
               |
|0     |3     |[(ColumnValues "myCol1" > 1) OR (ColumnValues "myCol2" > 2)]|Passed
               |
+------+------+-------------------------------------------------------------
+---------------------------+
```

Beberapa aturan tidak dapat didukung dalam fitur ini karena hasil keseluruhannya bergantung pada ambang batas atau rasio. Mereka tercantum di bawah ini.

Aturan yang mengandalkan rasio:

- Kelengkapan

- DatasetMatch
- ReferentialIntegrity
- Keunikan

Aturan tergantung pada ambang batas:

Ketika aturan berikut termasuk dengan ambang batas, mereka tidak didukung. Namun, aturan yang tidak melibatkan `with threshold` tetap didukung.

- ColumnDataType
- ColumnValues
- CustomSQL

Ekspresi

Jika tipe aturan tidak menghasilkan respons Boolean, Anda harus memberikan ekspresi sebagai parameter untuk membuat respons Boolean. Misalnya, aturan berikut memeriksa rata-rata (rata-rata) dari semua nilai dalam kolom terhadap ekspresi untuk mengembalikan hasil benar atau salah.

```
Mean "colA" between 80 and 100
```

Beberapa jenis aturan seperti `IsUnique` dan `IsComplete` sudah mengembalikan respons Boolean.

Tabel berikut mencantumkan ekspresi yang dapat Anda gunakan dalam aturan DQDL.

Ekspresi DQDL yang didukung

Ekspresi	Deskripsi	Contoh
= *x*	*Menyelesaikan true jika respons tipe aturan sama dengan x.*	`Completeness "colA" = "1.0", ColumnValues "colA" = "2022-06-30"`
! = *x*	*x Menyelesaikan ke true jika respons tipe aturan tidak sama dengan x.*	`ColumnValues "colA" != "a", ColumnValues "colA" != "2022-06-30"`

Ekspresi	Deskripsi	Contoh
> *x*	*Menyelesaikan true jika respons tipe aturan lebih besar dari x.*	`ColumnValues "colA" > 10`
< *x*	*Menyelesaikan true jika respons tipe aturan kurang dari x.*	`ColumnValues "colA" < 1000, ColumnValues "colA" < "2022-06-30"`
>= x	*Menyelesaikan true jika respons tipe aturan lebih besar dari atau sama dengan x.*	`ColumnValues "colA" >= 10`
<= x	*Menyelesaikan true jika respons tipe aturan kurang dari atau sama dengan x.*	`ColumnValues "colA" <= 1000`
antara *x* dan *y*	Menyelesaikan `true` jika respons tipe aturan jatuh dalam rentang tertentu (eksklusif). Hanya gunakan jenis ekspresi ini untuk tipe numerik dan tanggal.	`Mean "colA" between 8 and 100, ColumnValues "colA" between "2022-05-31" and "2022-06-30"`
bukan antara *x* dan *y*	Menyelesaikan ke true jika respons tipe aturan tidak termasuk dalam rentang tertentu (inklusif). Anda hanya harus menggunakan tipe ekspresi ini untuk tipe numerik dan tanggal.	`ColumnValues "colA" not between "2022-05-31" and "2022-06-30"`

Ekspresi	Deskripsi	Contoh
di [*a, b, c,...*]	Menyelesaikan `true` jika respons tipe aturan ada di set yang ditentukan.	`ColumnValues "colA" in [ 1, 2, 3 ], ColumnValues "colA" in [ "a", "b", "c" ]`
tidak di [*a, b, c,...*]	Menyelesaikan `true` jika respons tipe aturan tidak ada dalam set yang ditentukan.	`ColumnValues "colA" not in [ 1, 2, 3 ], ColumnValues "colA" not in [ "a", "b", "c" ]`
cocok */ab+c/i*	Menyelesaikan `true` jika respons tipe aturan cocok dengan ekspresi reguler.	`ColumnValues "colA" matches "[a-zA-Z]*"`
tidak cocok */ab+c/i*	Menyelesaikan `true` jika respons tipe aturan tidak cocok dengan ekspresi reguler.	`ColumnValues "colA" not matches "[a-zA-Z]*"`
`now()`	Bekerja hanya dengan jenis `ColumnValues` aturan untuk membuat ekspresi tanggal.	`ColumnValues "load_date" > (now() - 3 days)`
pertandingan/di [...] /tidak cocok/tidak di [...] `with threshold`	Menentukan persentase nilai yang cocok dengan kondisi aturan. Bekerja hanya dengan`ColumnValues` ,`ColumnDataType` , dan jenis `CustomSQL` aturan.	`ColumnValues "colA" in ["A", "B"] with threshold > 0.8, ColumnValues "colA" matches "[a-zA-Z]*" with threshold between 0.2 and 0.9 ColumnDataType "colA" = "Timestamp" with threshold > 0.9`

Kata kunci untuk NULL, EMPTY dan WHITESPACES_ONLY

Jika Anda ingin memvalidasi jika kolom string memiliki nol, kosong atau string dengan hanya spasi putih Anda dapat menggunakan kata kunci berikut:

- NULL/null — Kata kunci ini menyelesaikan true untuk `null` nilai dalam kolom string.

  `ColumnValues "colA" != NULL with threshold > 0.5`akan mengembalikan true jika lebih dari 50% data Anda tidak memiliki nilai nol.

  `(ColumnValues "colA" = NULL) or (ColumnLength "colA" > 5)`akan mengembalikan true untuk semua baris yang memiliki nilai nol atau memiliki panjang> 5. Perhatikan bahwa ini akan memerlukan penggunaan opsi “compositeRuleEvaluation.method” = “ROW”.
- EMPTY/empty - Kata kunci ini menyelesaikan true untuk nilai string kosong (“”) dalam kolom string. Beberapa format data mengubah nol dalam kolom string menjadi string kosong. Kata kunci ini membantu menyaring string kosong dalam data Anda.

  `(ColumnValues "colA" = EMPTY) or (ColumnValues "colA" in ["a", "b"])`akan mengembalikan true jika baris kosong, “a” atau “b”. Perhatikan bahwa ini memerlukan penggunaan opsi “compositeRuleEvaluation.method” = “ROW”.
- WHITESPACES_ONLY/whitespaces_only — Kata kunci ini menyelesaikan true untuk string dengan hanya spasi putih (“”) nilai dalam kolom string.

  `ColumnValues "colA" not in ["a", "b", WHITESPACES_ONLY]`akan mengembalikan true jika baris bukan “a” atau “b” atau hanya spasi putih.

  Aturan yang didukung:

  - ColumnValues

Untuk ekspresi berbasis numerik atau tanggal, jika Anda ingin memvalidasi jika kolom memiliki nol, Anda dapat menggunakan kata kunci berikut.

- NULL/null — Kata kunci ini menyelesaikan true untuk nilai null dalam kolom string.

  `ColumnValues "colA" in [NULL, "2023-01-01"]`akan mengembalikan true jika tanggal di kolom Anda salah satu `2023-01-01` atau nol.

  `(ColumnValues "colA" = NULL) or (ColumnValues "colA" between 1 and 9)`akan mengembalikan true untuk semua baris yang memiliki nilai nol atau memiliki nilai antara 1 dan 9.

Perhatikan bahwa ini akan memerlukan penggunaan opsi “compositeRuleEvaluation.method” = “ROW”.

Aturan yang didukung:

- ColumnValues

Pemfilteran dengan Klausul Where

Anda dapat memfilter data Anda saat membuat aturan. Ini sangat membantu ketika Anda ingin menerapkan aturan bersyarat.

```
<DQDL Rule> where "<valid SparkSQL where clause> "
```

Filter harus ditentukan dengan `where` kata kunci, diikuti oleh pernyataan SparkSQL yang valid yang terlampir dalam tanda kutip. `("")`

Jika aturan Anda ingin menambahkan klausa where ke aturan dengan ambang batas, klausa where harus ditentukan sebelum kondisi ambang batas.

```
<DQDL Rule> where "valid SparkSQL statement>" with threshold <threshold condition>
```

Dengan sintaks ini Anda dapat menulis aturan seperti berikut ini.

```
Completeness "colA" > 0.5 where "colB = 10"
ColumnValues "colB" in ["A", "B"] where "colC is not null" with threshold > 0.9
ColumnLength "colC" > 10 where "colD != Concat(colE, colF)"
```

Kami akan memvalidasi bahwa pernyataan SparkSQL yang diberikan valid. Jika tidak valid, evaluasi aturan akan gagal dan kami akan melempar a `IllegalArgumentException` dengan format berikut:

```
Rule <DQDL Rule> where "<invalid SparkSQL>" has provided an invalid where clause :
<SparkSQL Error>
```

Di mana perilaku klausa saat identifikasi catatan kesalahan tingkat baris diaktifkan

Dengan AWS Glue Data Quality, Anda dapat mengidentifikasi catatan spesifik yang gagal. Saat menerapkan klausa where ke aturan yang mendukung hasil tingkat baris, kami akan memberi label pada baris yang disaring oleh klausa where sebagai. `Passed`

Jika Anda lebih suka memberi label secara terpisah pada baris yang difilter sebagai`SKIPPED`, Anda dapat mengatur yang berikut ini `additionalOptions` untuk pekerjaan ETL.

```
object GlueApp {
  val datasource = glueContext.getCatalogSource(
    database="<db>",
    tableName="<table>",
    transformationContext="datasource"
  ).getDynamicFrame()

  val ruleset = """
    Rules = [
       IsComplete "att2" where "att1 = 'a'"
    ]
  """

  val dq_results = EvaluateDataQuality.processRows(
    frame=datasource,
    ruleset=ruleset,
    additionalOptions=JsonOptions("""
        {
          "rowLevelConfiguration.filteredRowLabel":"SKIPPED"
        }
      """
    )
  )
}
```

Sebagai contoh, lihat aturan dan kerangka data berikut:

```
IsComplete att2 where "att1 = 'a'"
```

id	att1	att2	Hasil tingkat baris (Default)	Hasil Tingkat Baris (Opsi Dilewati)	Komentar
1	a	f	BERLALU	BERLALU	

id	att1	att2	Hasil tingkat baris (Default)	Hasil Tingkat Baris (Opsi Dilewati)	Komentar
2	b	d	BERLALU	DILEWATI	Baris disaring, `att1` karena tidak `"a"`
3	a	null	FAILED	FAILED	
4	a	f	BERLALU	BERLALU	
5	b	null	BERLALU	DILEWATI	Baris disaring, `att1` karena tidak `"a"`
6	a	f	BERLALU	BERLALU	

Aturan dinamis

Anda sekarang dapat membuat aturan dinamis untuk membandingkan metrik saat ini yang dihasilkan oleh aturan Anda dengan nilai historisnya. Perbandingan historis ini diaktifkan dengan menggunakan `last()` operator dalam ekspresi. Misalnya, aturan `RowCount > last()` akan berhasil ketika jumlah baris dalam proses saat ini lebih besar dari jumlah baris sebelumnya terbaru untuk kumpulan data yang sama. `last()`mengambil argumen bilangan asli opsional yang menjelaskan berapa banyak metrik sebelumnya untuk dipertimbangkan; `last(k)` di mana `k >= 1` akan merujuk metrik terakhirk.

- Jika tidak ada titik data yang tersedia, `last(k)` akan mengembalikan nilai default 0.0.
- Jika kurang dari `k` metrik yang tersedia, `last(k)` akan mengembalikan semua metrik sebelumnya.

Untuk membentuk ekspresi yang valid gunakan`last(k)`, di mana `k > 1` memerlukan fungsi agregasi untuk mengurangi beberapa hasil historis menjadi satu nomor. Misalnya, `RowCount > avg(last(5))` akan memeriksa apakah jumlah baris kumpulan data saat ini benar-benar lebih besar dari rata-rata jumlah lima baris terakhir untuk kumpulan data yang sama. `RowCount >`

`last(5)`akan menghasilkan kesalahan karena jumlah baris kumpulan data saat ini tidak dapat dibandingkan secara bermakna dengan daftar.

Fungsi agregasi yang didukung:

- `avg`
- `median`
- `max`
- `min`
- `sum`
- `std`(standar deviasi)
- `abs`(nilai absolut)
- `index(last(k), i)`akan memungkinkan untuk memilih nilai terbaru dari yang terakhirk. `i` idiindeks nol, jadi `index(last(3), 0)` akan mengembalikan titik data terbaru dan `index(last(3), 3)` akan menghasilkan kesalahan karena hanya ada tiga titik data dan kami mencoba untuk mengindeks yang terbaru ke-4.

Ekspresi sampel

ColumnCorrelation

- `ColumnCorrelation "colA" "colB" < avg(last(10))`

DistinctValuesCount

- `DistinctValuesCount "colA" between min(last(10))-1 and max(last(10))+1`

Sebagian besar tipe aturan dengan kondisi numerik atau ambang batas mendukung aturan dinamis; lihat tabel yang disediakan, Analisa dan Aturan, untuk menentukan apakah aturan dinamis didukung untuk jenis aturan Anda.

Analisa

> **Note**
>
> Analyzer tidak didukung di AWS Glue Data Catalog.

Aturan DQDL menggunakan fungsi yang disebut penganalisis untuk mengumpulkan informasi tentang data Anda. Informasi ini digunakan oleh ekspresi Boolean aturan untuk menentukan apakah aturan harus berhasil atau gagal. Misalnya, RowCount aturan `RowCount > 5` akan menggunakan penganalisis hitungan baris untuk menemukan jumlah baris dalam kumpulan data Anda, dan membandingkan jumlah itu dengan ekspresi `> 5` untuk memeriksa apakah ada lebih dari lima baris dalam kumpulan data saat ini.

Terkadang, alih-alih membuat aturan, kami sarankan untuk membuat penganalisis dan kemudian membuatnya menghasilkan statistik yang dapat digunakan untuk mendeteksi anomali. Untuk contoh seperti itu, Anda dapat membuat penganalisis. Penganalisis berbeda dari aturan dengan cara berikut.

Karakteristik	Analisa	Aturan
Bagian dari ruleset	Ya	Ya
Menghasilkan statistik	Ya	Ya
Menghasilkan pengamatan	Ya	Ya
Dapat mengevaluasi dan menegaskan suatu kondisi	Tidak	Ya
Anda dapat mengonfigurasi tindakan seperti menghenti kan pekerjaan pada kegagalan , melanjutkan pekerjaan pemrosesan	Tidak	Ya

Analyzer dapat eksis secara independen tanpa aturan, sehingga Anda dapat dengan cepat mengonfigurasinya dan secara progresif membangun aturan kualitas data.

Beberapa jenis aturan dapat dimasukkan ke dalam `Analyzers` blok kumpulan aturan Anda untuk menjalankan aturan yang diperlukan untuk penganalisis dan mengumpulkan informasi tanpa menerapkan pemeriksaan untuk kondisi apa pun. Beberapa penganalisis tidak terkait dengan aturan dan hanya dapat dimasukkan ke dalam `Analyzers` blok. Tabel berikut menunjukkan apakah setiap item didukung sebagai aturan atau penganalisis mandiri, bersama dengan detail tambahan untuk setiap jenis aturan.

Contoh Ruleset dengan Analyzer

Ruleset berikut menggunakan:

- aturan dinamis untuk memeriksa apakah kumpulan data tumbuh di atas rata-rata trailing untuk tiga pekerjaan terakhir
- `DistinctValuesCount`analyzer untuk mencatat jumlah nilai yang berbeda di kolom dataset `Name`
- `ColumnLength`analyzer untuk melacak `Name` ukuran minimum dan maksimum dari waktu ke waktu

Hasil metrik penganalisis dapat dilihat di tab Kualitas Data untuk menjalankan pekerjaan Anda.

```
Rules = [
   RowCount > avg(last(3))
]
Analyzers = [
   DistinctValuesCount "Name",
   ColumnLength "Name"
]
```

Komentar

Anda dapat menggunakan karakter '#' untuk menambahkan komentar ke dokumen DQDL Anda. Apa pun setelah karakter '#' dan sampai akhir baris diabaikan oleh DQDL.

```
Rules = [
    # More items should generally mean a higher price, so correlation should be
 positive
    ColumnCorrelation "price" "num_items" > 0
]
```

Referensi tipe aturan DQDL

Bagian ini memberikan referensi untuk setiap jenis aturan yang didukung AWS Glue Data Quality.

> **Note**
>
> - DQDL saat ini tidak mendukung data kolom bersarang atau tipe daftar.

- Nilai kurung pada tabel di bawah ini akan diganti dengan informasi yang disediakan dalam argumen aturan.
- Aturan biasanya memerlukan argumen tambahan untuk ekspresi.

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?
Aggrega Match	Memerik apakah dua kumpula data cocok dengan memban gkan metrik ringkasa seperti jumlah total penjuala . Berguna bagi lembaga keuanga untuk memban gkan jika semua	Satu atau lebih agregasi	Ketika nama kolom agregasi pertama dan kedua cocok: `Column. [C olumn]. gregate tch` Ketika nama kolom agregasi pertama dan kedua berbeda: `Column. [C`	Ya	Tidak	Tidak	Tidak	Tidak	Tidak

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?
	data dicerna dari sistem sumber.		`olumn1, lumn2]. gregate tch`						

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?

AllStatis tics	Pengana sis mandiri untuk mengum kan beberap metrik untuk kolom yang disediak n, atau semua kolom dalam kumpula data.	Nama kolom tunggal, ATAU "AllColur ns”	Untuk kolom dari semua jenis:
			
			`Dataset .RowCou`
			
			`Column [C olumn]. mpleter s`
			
			`Column [C olumn]. iquenes`
			
			Metrik tambaha untuk kolom bernilai string:
			
			`ColumnL gth metrics`
				Tidak	Ya	Tidak	Tidak	Tidak	Tidak

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengama n	Mendukung Sintaks Klausul Dimana?
			Metrik tambaha untuk kolom bernilai numerik: `Column\ ues metrics`						
ColumnC relation	Memerik seberapa baik dua kolom dikorelas ikan.	Tepat dua nama kolom	`Multic mn. [Colum n1], [Colu mn2].Cc mnCorre tion`	Ya	Ya	Tidak	Ya	Tidak	Ya
ColumnC nt	Memerik apakah ada kolom yang dijatuhka n.	Tidak ada	`Dataset .Colum unt`	Ya	Tidak	Tidak	Ya	Ya	Tidak

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?
ColumnD aType	Memerik apakah kolom sesuai dengan tipe data.	Tepat satu nama kolom	`Column. [C olumn]. lumnDat ype.Cor iance`	Ya	Tidak	Tidak	Ya, dalam ekspresi ambang batas tingkat baris	Tidak	Ya
ColumnE sts	Memerik apakah kolom ada dalam kumpula data. Hal ini memung kan pelangga membant platform data layanan mandiri untuk memastik n kolom tertentu tersedia.	Tepat satu nama kolom	N/A	Ya	Tidak	Tidak	Tidak	Tidak	Tidak

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?

ColumnL gth	Memerik apakah panjang data konsister .	Tepat satu nama kolom	`Column [C olumn]. ximumLe th`
			
			`Column [C olumn]. nimumLe th`
			
			Metrik tambaha saat ambang tingkat baris disediak n:
			
			`Column [C olumn]. lumnVal s.Compl nce`
				Ya	Ya	Ya, ketika ambang batas tingkat baris disediak n	Tidak	Ya. Hanya menghas kan pengam n dengan mengan sis panjang Minimum dan Maksimu	Ya

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?
ColumnN esMatchI ttern	Memerik apakah nama kolom cocok dengan pola yang ditentuka n. Berguna bagi tim tata kelola untuk menegak n konsister si nama kolom.	Sebuah regex untuk nama kolom	`Dataset .Columr mesPatt nMatchF io`	Ya	Tidak	Tidak	Tidak	Tidak	Tidak

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukung sebagai Aturan?	Didukung sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengama n	Mendukung Sintaks Klausul Dimana?

ColumnV ues	Memerik apakah data konsister per nilai yang ditentuka n. Aturan ini menduku ekspresi reguler.	Tepat satu nama kolom	`Column. [C olumn]. ximum`
			
			`Column. [C olumn]. nimum`
			
			Metrik tambaha saat ambang tingkat baris disediaka n:
			
			`Column. [C olumn]. lumnVa] s.Comp] nce`
				Ya	Ya	Ya, ketika ambang batas tingkat baris disediaka n	Tidak	Ya. Hanya menghas kan pengama n dengan menganа sis nilai Minimum dan Maksimu	Ya

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?
Kelengka an	Memerik kosong atau NULL dalam data.	Tepat satu nama kolom	`Column [C olumn] mpleter s`	Ya	Ya	Ya	Ya	Ya	Ya

CustomS	Pelangg dapat menerap n hampir semua jenis pemeriks an kualitas data di SQL.	Pernyata n SQL
		
		(Opsiona) Ambang batas tingkat baris
			`Dataset .Custom L`
			
			Metrik tambaha saat ambang tingkat baris disediak n:
			
			`Dataset .Custom L.Compl nce`
				Ya	Tidak	Ya, ketika ambang batas tingkat baris disediak n	Ya	Tidak	Tidak

DataFres ness	Memerik apakah data masih segar.	Tepat satu nama kolom	`Column [C olumn] taFresh ss.Comp ance`	Ya	Tidak	Ya	Tidak	Tidak	Ya

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?
DatasetM tch	Membandingkan dua kumpulan data dan mengidentifikasi jika mereka sinkron.	Nama dataset referensi Pemetaan kolom (Opsional) Kolom untuk memerik kecocok	`Dataset [ Referen Dataset ias].Da setMatc`	Ya	Tidak	Ya	Ya	Tidak	Tidak
DistinctV aluesCou t	Memerik nilai duplikat.	Tepat satu nama kolom	`Column [C olumn]. stinctV uesCour`	Ya	Ya	Ya	Ya	Ya	Ya
DetectAr malies	Memerik anomali dalam metrik yang dilaporka n tipe aturan lain.	Jenis aturan	Metrik dilaporka n oleh argumen tipe aturan	Ya	Tidak	Tidak	Tidak	Tidak	Tidak

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?
Entropi	Memerik entropi data.	Tepat satu nama kolom	`Column. [C olumn]. tropy`	Ya	Ya	Tidak	Ya	Tidak	Ya
IsComple e	Memerik apakah 100% data selesai.	Tepat satu nama kolom	`Column. [C olumn]. mpleter s`	Ya	Tidak	Ya	Tidak	Tidak	Ya
IsPrimar Key	Memerik apakah kolom adalah kunci utama (bukan NULL dan unik).	Tepat satu nama kolom	Untuk kolom tunggal: `Column. [C olumn]. iquenes` Untuk beberap kolom: `Multic mn[Com elimite olumns] niquene`	Ya	Tidak	Ya	Tidak	Tidak	Ya

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?
IsUnique	Memerik apakah 100% data unik.	Tepat satu nama kolom	`Column [C olumn] iquenes`	Ya	Tidak	Ya	Tidak	Tidak	Ya
Berarti	Memerik apakah mean cocok dengan ambang batas yang ditetapka n.	Tepat satu nama kolom	`Column [C olumn] an`	Ya	Ya	Ya	Ya	Tidak	Ya

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?
Referent alIntegri ty	Memerik apakah dua kumpula data memiliki integrita s referensi al.	Satu atau beberap nama kolom dari dataset Satu atau lebih nama columb dari dataset referensi	`Column [R eferenc ataset/ as].Re ential egrity`	Ya	Tidak	Ya	Ya	Tidak	Tidak
RowCou	Memerik apakah jumlah catatan cocok dengan ambang batas.	Tidak ada	`Dataset .RowCou`	Ya	Ya	Tidak	Ya	Ya	Ya

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?
RowCou atch	Memerik apakah jumlah rekaman antara dua kumpula data cocok.	Alias dataset referensi	`Dataset [ Referer Dataset ias].Ro ountMat`	Ya	Tidak	Tidak	Ya	Tidak	Tidak
Standarc eviation	Memerik apakah standar deviasi cocok dengan ambang batas.	Tepat satu nama kolom	`Column [C olumn]. andardD iation`	Ya	Ya	Ya	Ya	Tidak	Ya
Schemal ch	Memerik apakah skema antara dua kumpula data cocok.	Alias dataset referensi	`Dataset [ Referer Dataset ias].Sc maMatch`	Ya	Tidak	Tidak	Ya	Tidak	Tidak

Tipe aturan	Deskrips	Pendapa	Metrik yang Dilaporka n	Didukun sebagai Aturan?	Didukun sebagai Analyzer	Mengem ikan Hasil tingkat baris?	Dukunga aturan dinamis?	Menghas kan Pengam n	Mendukung Sintaks Klausul Dimana?
Jumlah	Memerik apakah jumlah cocok dengan ambang batas yang ditetapka n.	Tepat satu nama kolom	`Column [C olumn] m`	Ya	Ya	Tidak	Ya	Tidak	Ya
Keunikar	Memerik apakah keunikan kumpula data cocok dengan ambang batas.	Tepat satu nama kolom	`Column [C olumn] iquenes`	Ya	Ya	Ya	Ya	Tidak	Ya
UniqueV ueRatio	Memerik apakah ransum nilai unik cocok dengan ambang batas.	Tepat satu nama kolom	`Column [C olumn] iqueVa Ratio`	Ya	Ya	Ya	Ya	Tidak	Ya

Topik

- AggregateMatch
- ColumnCorrelation
- ColumnCount
- ColumnDataType
- ColumnExists
- ColumnLength
- ColumnNamesMatchPattern
- ColumnValues
- Kelengkapan
- CustomSQL
- DataFreshness
- DatasetMatch
- DistinctValuesCount
- Entropi
- IsComplete
- IsPrimaryKey
- IsUnique
- Berarti
- ReferentialIntegrity
- RowCount
- RowCountMatch
- StandardDeviation
- Jumlah
- SchemaMatch
- Keunikan
- UniqueValueRatio
- DetectAnomalies

AggregateMatch

Memeriksa rasio agregasi dua kolom terhadap ekspresi yang diberikan. Ruletype ini bekerja pada beberapa dataset. Dua agregasi kolom dievaluasi dan rasio dihasilkan dengan membagi hasil agregasi kolom pertama dengan hasil agregasi kolom kedua. Rasio diperiksa terhadap ekspresi yang disediakan untuk menghasilkan respons boolean.

Sintaksis

Agregasi kolom

```
ColumnExists <AGG_OPERATION> (<OPTIONAL_REFERENCE_ALIAS>.<COL_NAME>)
```

- AGG_OPERATION — Operasi yang digunakan untuk agregasi. Saat ini, `sum` dan `avg` didukung.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- OPTIONAL_REFERENCE_ALIAS - Parameter ini perlu disediakan jika kolom berasal dari kumpulan data referensi dan bukan kumpulan data utama. <database_name>Jika Anda menggunakan aturan ini di Katalog Data AWS Glue, alias referensi Anda harus mengikuti format ". <table_name>. <column_name>

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- COL_NAME — Nama kolom untuk agregat.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek

Contoh: Rata-rata

```
"avg(rating)"
```

Contoh: Jumlah

```
"sum(amount)"
```

Contoh: Rata-rata kolom dalam kumpulan data referensi

```
"avg(reference.rating)"
```

Aturan

```
AggregateMatch <AGG_EXP_1> <AGG_EXP_2> <EXPRESSION>
```

- AGG_EXP_1 - Agregasi kolom pertama.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- AGG_EXP_2 - Agregasi kolom kedua.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Pencocokan Agregat menggunakan jumlah

Contoh aturan berikut memeriksa apakah jumlah nilai dalam `amount` kolom persis sama dengan jumlah nilai dalam `total_amount` kolom.

```
AggregateMatch "sum(amount)" "sum(total_amount)" = 1.0
```

Contoh: Pencocokan Agregat menggunakan rata-rata

Contoh aturan berikut memeriksa apakah rata-rata nilai dalam `ratings` kolom sama dengan setidaknya 90% dari rata-rata nilai dalam `ratings` kolom dalam `reference` dataset. Dataset referensi disediakan sebagai sumber data tambahan dalam pengalaman ETL atau Katalog Data.

Di AWS Glue ETL, Anda dapat menggunakan:

```
AggregateMatch "avg(ratings)" "avg(reference.ratings)" >= 0.9
```

Dalam Katalog Data AWS Glue, Anda dapat menggunakan:

```
AggregateMatch "avg(ratings)" "avg(database_name.tablename.ratings)" >= 0.9
```

Perilaku nol

`AggregateMatch`Aturan akan mengabaikan baris dengan nilai NULL dalam perhitungan metode agregasi (jumlah/mean). Sebagai contoh:

```
+---+-----------+
|id |units      |
+---+-----------+
|100|0          |
|101|null       |
|102|20         |
|103|null       |
|104|40         |
+---+-----------+
```

Rata-rata kolom `units` adalah (0 + 20 + 40)/3 = 20. Baris 101 dan 103 tidak dipertimbangkan dalam perhitungan ini.

ColumnCorrelation

Memeriksa korelasi antara dua kolom terhadap ekspresi yang diberikan. AWS Glue Data Quality menggunakan koefisien korelasi Pearson untuk mengukur korelasi linier antara dua kolom. Hasilnya adalah angka antara -1 dan 1 yang mengukur kekuatan dan arah hubungan.

Sintaksis

```
ColumnCorrelation <COL_1_NAME> <COL_2_NAME> <EXPRESSION>
```

- COL_1_NAME — Nama kolom pertama yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- COL_2_NAME — Nama kolom kedua yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Korelasi kolom

Contoh aturan berikut memeriksa apakah koefisien korelasi antara kolom `height` dan `weight` memiliki korelasi positif yang kuat (nilai koefisien lebih besar dari 0,8).

```
ColumnCorrelation "height" "weight" > 0.8
```

```
ColumnCorrelation "weightinkgs" "Salary" > 0.8 where "weightinkgs > 40"
```

Contoh aturan dinamis

- `ColumnCorrelation "colA" "colB" between min(last(10)) and max(last(10))`
- `ColumnCorrelation "colA" "colB" < avg(last(5)) + std(last(5))`

Perilaku nol

`ColumnCorrelation`Aturan akan mengabaikan baris dengan `NULL` nilai dalam perhitungan korelasi. Sebagai contoh:

```
+---+-----------+
|id |units      |
+---+-----------+
|100|0          |
|101|null       |
|102|20         |
|103|null       |
|104|40         |
+---+-----------+
```

Baris 101 dan 103 akan diabaikan, dan `ColumnCorrelation` akan menjadi 1.0.

ColumnCount

Memeriksa jumlah kolom dari dataset primer terhadap ekspresi yang diberikan. Dalam ekspresi, Anda dapat menentukan jumlah kolom atau rentang kolom menggunakan operator seperti > dan<.

Sintaksis

```
ColumnCount <EXPRESSION>
```

- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Pemeriksaan numerik hitungan kolom

Contoh aturan berikut memeriksa apakah jumlah kolom berada dalam rentang tertentu.

```
ColumnCount between 10 and 20
```

Contoh aturan dinamis

- `ColumnCount >= avg(last(10))`
- `ColumnCount between min(last(10))-1 and max(last(10))+1`

ColumnDataType

Memeriksa tipe data inheren dari nilai-nilai dalam kolom tertentu terhadap jenis yang diharapkan disediakan. Menerima `with threshold` ekspresi untuk memeriksa subset nilai di kolom.

Sintaksis

```
ColumnDataType <COL_NAME> = <EXPECTED_TYPE>
   ColumnDataType <COL_NAME> = <EXPECTED_TYPE> with threshold <EXPRESSION>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Jenis string

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- EXPECTED_TYPE — Jenis nilai yang diharapkan di kolom.

  Nilai yang didukung: Boolean, Date, Timestamp, Integer, Double, Float, Long

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- EKSPRESI - Ekspresi opsional untuk menentukan persentase nilai yang harus dari tipe yang diharapkan.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek

Contoh: Kolom tipe data bilangan bulat sebagai string

Contoh aturan berikut memeriksa apakah nilai-nilai dalam kolom yang diberikan, yang merupakan tipe string, sebenarnya bilangan bulat.

```
ColumnDataType "colA" = "INTEGER"
```

Contoh: Bilangan bulat tipe data kolom sebagai string memeriksa subset dari nilai

Contoh aturan berikut memeriksa apakah lebih dari 90% dari nilai-nilai dalam kolom yang diberikan, yang bertipe string, sebenarnya bilangan bulat.

```
ColumnDataType "colA" = "INTEGER" with threshold > 0.9
```

ColumnExists

Memeriksa apakah kolom ada.

Sintaksis

```
ColumnExists <COL_NAME>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Tipe kolom apa pun

Contoh: Kolom ada

Contoh aturan berikut memeriksa apakah kolom bernama `Middle_Name` ada.

```
ColumnExists "Middle_Name"
```

ColumnLength

Memeriksa apakah panjang setiap baris dalam kolom sesuai dengan ekspresi yang diberikan.

Sintaksis

```
ColumnLength <COL_NAME><EXPRESSION>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: String

- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Panjang baris kolom

Contoh aturan berikut memeriksa apakah nilai di setiap baris dalam kolom bernama `Postal_Code` adalah 5 karakter panjang.

```
ColumnLength "Postal_Code" = 5
ColumnLength "weightinkgs" = 2 where "weightinkgs > 10"
```

Perilaku nol

`ColumnLength`Aturan memperlakukan `NULL` s sebagai 0 string panjang. Untuk satu `NULL` baris:

```
ColumnLength "Postal_Code" > 4 # this will fail
```

```
ColumnLength "Postal_Code" < 6 # this will succeed
```

Contoh aturan majemuk berikut menyediakan cara untuk secara eksplisit gagal `NULL` nilai:

```
(ColumnLength "Postal_Code" > 4) AND (ColumnValues != NULL)
```

ColumnNamesMatchPattern

Memeriksa apakah nama semua kolom dalam kumpulan data utama cocok dengan ekspresi reguler yang diberikan.

Sintaksis

```
ColumnNamesMatchPattern <PATTERN>
```

- POLA — Pola yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek

Contoh: Pola pencocokan nama kolom

Contoh aturan berikut memeriksa apakah semua kolom dimulai dengan awalan “aws_”

```
ColumnNamesMatchPattern "aws_.*"
ColumnNamesMatchPattern "aws_.*" where "weightinkgs > 10"
```

ColumnValues

Menjalankan ekspresi terhadap nilai-nilai dalam kolom.

Sintaksis

```
ColumnValues <COL_NAME> <EXPRESSION>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Tipe kolom apa pun
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Nilai yang diizinkan

Contoh aturan berikut memeriksa apakah setiap nilai dalam kolom yang ditentukan berada dalam satu set nilai yang diizinkan (termasuk nol, kosong, dan string dengan hanya spasi putih).

```
ColumnValues "Country" in [ "US", "CA", "UK", NULL, EMPTY, WHITESPACES_ONLY ]
ColumnValues "gender" in ["F", "M"] where "weightinkgs < 10"
```

Contoh: Ekspresi reguler

Contoh aturan berikut memeriksa nilai-nilai dalam kolom terhadap ekspresi reguler.

```
ColumnValues "First_Name" matches "[a-zA-Z]*"
```

Contoh: Nilai tanggal

Contoh aturan berikut memeriksa nilai-nilai dalam kolom tanggal terhadap ekspresi tanggal.

```
ColumnValues "Load_Date" > (now() - 3 days)
```

Contoh: Nilai numerik

Contoh aturan berikut memeriksa apakah nilai kolom cocok dengan kendala numerik tertentu.

```
ColumnValues "Customer_ID" between 1 and 2000
```

Perilaku nol

Untuk semua `ColumnValues` aturan (selain `!=` dan`NOT IN`), `NULL` baris akan gagal aturan. Jika aturan gagal karena nilai nol, alasan kegagalan akan menampilkan yang berikut:

```
Value: NULL does not meet the constraint requirement!
```

Contoh aturan majemuk berikut menyediakan cara untuk secara eksplisit mengizinkan nilai: `NULL`

```
(ColumnValues "Age" > 21) OR (ColumnValues "Age" = NULL)
```

ColumnValues Aturan yang dinegasikan menggunakan `not in` sintaks `!=` dan akan diteruskan untuk `NULL` baris. Sebagai contoh:

```
ColumnValues "Age" != 21
```

```
ColumnValues "Age" not in [21, 22, 23]
```

Contoh berikut memberikan cara untuk secara eksplisit gagal nilai `NULL`

```
(ColumnValues "Age" != 21) AND (ColumnValues "Age" != NULL)
```

```
ColumnValues "Age" not in [21, 22, 23, NULL]
```

Kelengkapan

Memeriksa persentase nilai lengkap (non-null) dalam kolom terhadap ekspresi yang diberikan.

Sintaksis

```
Completeness <COL_NAME> <EXPRESSION>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Tipe kolom apa pun
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Persentase nilai nol

Contoh aturan berikut memeriksa apakah lebih dari 95 persen nilai dalam kolom selesai.

```
Completeness "First_Name" > 0.95
Completeness "First_Name" > 0.95 where "weightinkgs > 10"
```

Contoh aturan dinamis

- `Completeness "colA" between min(last(5)) - 1 and max(last(5)) + 1`
- `Completeness "colA" <= avg(last(10))`

Perilaku nol

Catatan tentang Format Data CSV: Baris kosong pada kolom CSV dapat menampilkan beberapa perilaku.

- Jika kolom `String` bertipe, baris kosong akan dikenali sebagai string kosong dan tidak akan gagal `Completeness` aturannya.
- Jika kolom adalah tipe data lain seperti`Int`, baris kosong akan dikenali sebagai `NULL` dan akan gagal `Completeness` aturan.

CustomSQL

Jenis aturan ini telah diperluas untuk mendukung dua kasus penggunaan:

- Jalankan pernyataan SQL kustom terhadap dataset dan periksa nilai kembali terhadap ekspresi yang diberikan.
- Jalankan pernyataan SQL kustom di mana Anda menentukan nama kolom dalam pernyataan SELECT Anda yang Anda bandingkan dengan beberapa kondisi untuk mendapatkan hasil tingkat baris.

Sintaksis

```
CustomSql <SQL_STATEMENT> <EXPRESSION>
```

- SQL_STATEMENT — Pernyataan SQL yang mengembalikan nilai numerik tunggal, dikelilingi oleh tanda kutip ganda.
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Kustom SQL untuk mengambil hasil aturan keseluruhan

Aturan contoh ini menggunakan pernyataan SQL untuk mengambil jumlah catatan untuk kumpulan data. Aturan kemudian memeriksa bahwa jumlah rekor adalah antara 10 dan 20.

```
CustomSql "select count(*) from primary" between 10 and 20
```

Contoh: SQL khusus untuk mengambil hasil tingkat baris

Aturan contoh ini menggunakan pernyataan SQL di mana Anda menentukan nama kolom dalam pernyataan SELECT Anda yang Anda bandingkan dengan beberapa kondisi untuk mendapatkan hasil tingkat baris. Ekspresi kondisi ambang batas mendefinisikan ambang batas berapa banyak catatan yang harus gagal agar seluruh aturan gagal. Perhatikan bahwa aturan mungkin tidak mengandung kondisi dan kata kunci bersama-sama.

```
CustomSql "select Name from primary where Age  > 18"
```

atau

```
CustomSql "select Name from primary where Age > 18" with threshold  > 3
```

> **Important**
>
> `primary`Alias singkatan dari nama kumpulan data yang ingin Anda evaluasi. Saat Anda bekerja dengan pekerjaan ETL visual di konsol, `primary` selalu mewakili `DynamicFrame` diteruskan ke `EvaluateDataQuality.apply()` transformasi. Bila Anda menggunakan AWS Glue Data Catalog untuk menjalankan tugas kualitas data terhadap tabel, `primary` mewakili tabel.

Jika Anda berada di Katalog AWS Glue Data, Anda juga dapat menggunakan nama tabel yang sebenarnya:

```
CustomSql "select count(*) from database.table" between 10 and 20
```

Anda juga dapat bergabung dengan beberapa tabel untuk membandingkan elemen data yang berbeda:

```
CustomSql "select count(*) from database.table inner join database.table2 on id1 = id2"
 between 10 and 20
```

Di AWS Glue ETL, CustomSQL dapat mengidentifikasi catatan yang gagal dalam pemeriksaan kualitas data. Agar ini berfungsi, Anda harus mengembalikan catatan yang merupakan bagian dari tabel utama yang Anda evaluasi kualitas data. Catatan yang dikembalikan sebagai bagian dari kueri dianggap berhasil dan catatan yang tidak dikembalikan dianggap gagal.

Aturan berikut akan memastikan bahwa catatan dengan usia < 100 diidentifikasi sebagai berhasil dan catatan yang di atas ditandai sebagai gagal.

```
CustomSql "select id from primary where age < 100"
```

Aturan CustomSQL ini akan berlalu ketika 50% dari catatan memiliki usia> 10 dan juga akan mengidentifikasi catatan yang gagal. Catatan yang dikembalikan oleh CustomSQL ini akan dianggap lulus sementara yang tidak dikembalikan akan dianggap gagal.

```
CustomSQL "select ID, CustomerID from primary where age > 10" with threshold > 0.5
```

Catatan: Aturan CustomSQL akan gagal jika Anda mengembalikan catatan yang tidak tersedia di kumpulan data.

DataFreshness

Memeriksa kesegaran data dalam kolom dengan mengevaluasi perbedaan antara waktu saat ini dan nilai kolom tanggal. Anda dapat menentukan ekspresi berbasis waktu untuk jenis aturan ini untuk memastikan bahwa nilai kolom up to date.

Sintaksis

```
DataFreshness <COL_NAME> <EXPRESSION>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Tanggal
- EKSPRESI — Ekspresi numerik dalam jam atau hari. Anda harus menentukan satuan waktu dalam ekspresi Anda.

Contoh: Kesegaran data

Contoh aturan berikut memeriksa kesegaran data.

```
DataFreshness "Order_Date" <= 24 hours
DataFreshness "Order_Date" between 2 days and 5 days
```

Perilaku nol

`DataFreshness`Aturan akan gagal untuk baris dengan `NULL` nilai. Jika aturan gagal karena nilai nol, alasan kegagalan akan menampilkan yang berikut:

```
80.00 % of rows passed the threshold
```

di mana 20% dari baris yang gagal termasuk baris dengan`NULL`.

Contoh aturan majemuk berikut menyediakan cara untuk secara eksplisit mengizinkan nilai: `NULL`

```
(DataFreshness "Order_Date" <= 24 hours) OR (ColumnValues "Order_Date" = NULL)
```

Kesegaran Data untuk objek Amazon S3

Terkadang Anda perlu memvalidasi kesegaran data berdasarkan waktu pembuatan file Amazon S3. Untuk melakukan ini, Anda dapat menggunakan kode berikut untuk mendapatkan stempel waktu dan menambahkannya ke kerangka data Anda, dan kemudian menerapkan pemeriksaan Kesegaran Data.

```
df = glueContext.create_data_frame.from_catalog(database = "default", table_name =
 "mytable")
df = df.withColumn("file_ts", df["_metadata.file_modification_time"])

Rules = [
 DataFreshness "file_ts" < 24 hours
]
```

DatasetMatch

Memeriksa apakah data dalam kumpulan data utama cocok dengan data dalam kumpulan data referensi. Kedua kumpulan data digabungkan menggunakan pemetaan kolom kunci yang disediakan. Pemetaan kolom tambahan dapat disediakan jika Anda ingin memeriksa kesetaraan data hanya di kolom tersebut. Perhatikan bahwa DataSetMatchagar berfungsi, kunci gabungan Anda harus unik dan tidak boleh NULL (harus menjadi kunci utama). Jika Anda tidak memenuhi kondisi ini, Anda akan mendapatkan pesan kesalahan, “Peta kunci yang disediakan tidak cocok untuk bingkai data yang diberikan”. Jika Anda tidak dapat memiliki kunci gabungan yang unik, pertimbangkan untuk menggunakan tipe aturan lain seperti AggregateMatchuntuk mencocokkan data ringkasan.

Sintaksis

```
DatasetMatch <REFERENCE_DATASET_ALIAS> <JOIN CONDITION WITH
 MAPPING> <OPTIONAL_MATCH_COLUMN_MAPPINGS> <EXPRESSION>
```

- REFERENCE_DATASET_ALIAS — Alias dari dataset referensi yang Anda gunakan untuk membandingkan data dari kumpulan data utama.
- KEY_COLUMN_MAPPINGS - Daftar nama kolom yang dipisahkan koma yang membentuk kunci dalam kumpulan data. Jika nama kolom tidak sama di kedua kumpulan data, Anda harus memisahkannya dengan `->`
- OPTIONAL_MATCH_COLUMN_MAPPINGS - Anda dapat menyediakan parameter ini jika Anda ingin memeriksa data yang cocok hanya di kolom tertentu. Ini menggunakan sintaks yang sama dengan pemetaan kolom kunci. Jika parameter ini tidak disediakan, kami akan mencocokkan data di semua kolom yang tersisa. Kolom non-kunci yang tersisa harus memiliki nama yang sama di kedua kumpulan data.
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Cocokkan set dataset menggunakan kolom ID

Aturan contoh berikut memeriksa bahwa lebih dari 90% kumpulan data utama cocok dengan kumpulan data referensi, menggunakan kolom “ID” untuk menggabungkan dua kumpulan data. Ini membandingkan semua kolom dalam kasus ini.

```
DatasetMatch "reference" "ID" >= 0.9
```

Contoh: Cocokkan set dataset menggunakan beberapa kolom kunci

Dalam contoh berikut, dataset primer dan dataset referensi memiliki nama yang berbeda untuk kolom kunci. `ID_1`dan `ID_2` bersama-sama membentuk kunci komposit dalam dataset primer. `ID_ref1`dan `ID_ref2` bersama-sama membentuk kunci komposit dalam dataset referensi. Dalam skenario ini, Anda dapat menggunakan sintaks khusus untuk menyediakan nama kolom.

```
DatasetMatch "reference" "ID_1->ID_ref1,ID_ref2->ID_ref2" >= 0.9
```

Contoh: Cocokkan kumpulan data set menggunakan beberapa kolom kunci dan periksa apakah kolom tertentu cocok

Contoh ini dibangun di atas contoh sebelumnya. Kami ingin memeriksa bahwa hanya kolom yang berisi jumlah yang cocok. Kolom ini dinamai `Amount1` dalam kumpulan data utama dan `Amount2` dalam kumpulan data referensi. Anda menginginkan kecocokan yang tepat.

```
DatasetMatch "reference" "ID_1->ID_ref1,ID_ref2->ID_ref2" "Amount1->Amount2" >= 0.9
```

DistinctValuesCount

Memeriksa jumlah nilai yang berbeda dalam kolom terhadap ekspresi yang diberikan.

Sintaksis

```
DistinctValuesCount <COL_NAME> <EXPRESSION>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Tipe kolom apa pun
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Jumlah nilai kolom yang berbeda

Contoh aturan berikut memeriksa bahwa kolom bernama `State` berisi lebih dari 3 nilai yang berbeda.

```
DistinctValuesCount "State" > 3
DistinctValuesCount "Customer_ID" < 6  where "Customer_ID < 10"
```

Contoh aturan dinamis

- `DistinctValuesCount "colA" between avg(last(10))-1 and avg(last(10))+1`
- `DistinctValuesCount "colA" <= index(last(10),2) + std(last(5))`

Entropi

Memeriksa apakah nilai entropi kolom cocok dengan ekspresi yang diberikan. Entropi mengukur tingkat informasi yang terkandung dalam pesan. Mengingat distribusi probabilitas atas nilai dalam kolom, entropi menjelaskan berapa banyak bit yang diperlukan untuk mengidentifikasi nilai.

Sintaksis

```
Entropy <COL_NAME> <EXPRESSION>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Tipe kolom apa pun
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Entropi kolom

Contoh aturan berikut memeriksa bahwa kolom bernama `Feedback` memiliki nilai entropi lebih besar dari satu.

```
Entropy "Star_Rating" > 1
Entropy "First_Name" > 1  where "Customer_ID < 10"
```

Contoh aturan dinamis

- `Entropy "colA" < max(last(10))`
- `Entropy "colA" between min(last(10)) and max(last(10))`

IsComplete

Memeriksa apakah semua nilai dalam kolom selesai (non-null).

Sintaksis

```
IsComplete <COL_NAME>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Tipe kolom apa pun

Contoh: Nilai nol

Contoh berikut memeriksa apakah semua nilai dalam kolom bernama `email` non-null.

```
IsComplete "email"
IsComplete "Email" where "Customer_ID between 1 and 50"
IsComplete "Customer_ID"  where "Customer_ID < 16 and Customer_ID != 12"
IsComplete "passenger_count" where "payment_type<>0"
```

Perilaku nol

Catatan tentang Format Data CSV: Baris kosong pada kolom CSV dapat menampilkan beberapa perilaku.

- Jika kolom `String` bertipe, baris kosong akan dikenali sebagai string kosong dan tidak akan gagal `Completeness` aturannya.
- Jika kolom adalah tipe data lain seperti`Int`, baris kosong akan dikenali sebagai `NULL` dan akan gagal `Completeness` aturan.

IsPrimaryKey

Memeriksa apakah kolom berisi kunci utama. Kolom berisi kunci primer jika semua nilai dalam kolom unik dan lengkap (non-null).

Sintaksis

```
IsPrimaryKey <COL_NAME>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

Jenis kolom yang didukung: Tipe kolom apa pun

Contoh: Kunci utama

Contoh aturan berikut memeriksa apakah kolom bernama `Customer_ID` berisi kunci utama.

```
IsPrimaryKey "Customer_ID"
IsPrimaryKey "Customer_ID" where "Customer_ID < 10"
```

Contoh: Kunci primer dengan beberapa kolom. Salah satu contoh berikut ini valid.

```
IsPrimaryKey "colA" "colB"
IsPrimaryKey "colA" "colB" "colC"
IsPrimaryKey colA "colB" "colC"
```

IsUnique

Memeriksa apakah semua nilai dalam kolom unik, dan mengembalikan nilai Boolean.

Sintaksis

```
IsUnique <COL_NAME>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Tipe kolom apa pun

Contoh: Nilai kolom unik

Contoh aturan berikut memeriksa apakah semua nilai dalam kolom bernama `email` unik.

```
IsUnique "email"
IsUnique "Customer_ID" where "Customer_ID < 10"]
```

Berarti

Memeriksa apakah mean (rata-rata) dari semua nilai dalam kolom cocok dengan ekspresi yang diberikan.

Sintaksis

```
Mean <COL_NAME> <EXPRESSION>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Nilai rata-rata

Contoh aturan berikut memeriksa apakah rata-rata semua nilai dalam kolom melebihi ambang batas.

```
Mean "Star_Rating" > 3
Mean "Salary" < 6200 where "Customer_ID < 10"
```

Contoh aturan dinamis

- `Mean "colA" > avg(last(10)) + std(last(2))`
- `Mean "colA" between min(last(5)) - 1 and max(last(5)) + 1`

Perilaku nol

`Mean`Aturan akan mengabaikan baris dengan `NULL` nilai dalam perhitungan mean. Sebagai contoh:

```
+---+-----------+
|id |units      |
+---+-----------+
|100|0          |
|101|null       |
|102|20         |
|103|null       |
|104|40         |
+---+-----------+
```

Rata-rata kolom `units` adalah (0 + 20 + 40)/3 = 20. Baris 101 dan 103 tidak dipertimbangkan dalam perhitungan ini.

ReferentialIntegrity

Memeriksa sejauh mana nilai dari satu set kolom dalam dataset primer adalah bagian dari nilai-nilai dari satu set kolom dalam dataset referensi.

Sintaksis

```
ReferentialIntegrity <PRIMARY_COLS> <REFERENCE_DATASET_COLS> <EXPRESSION>
```

- PRIMARY_COLS — Daftar nama kolom yang dipisahkan koma dalam kumpulan data utama.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- REFERENCE_DATASET_COLS — Parameter ini berisi dua bagian yang dipisahkan oleh titik. Bagian pertama adalah alias dari dataset referensi. Bagian kedua adalah daftar nama kolom yang dipisahkan koma dalam kumpulan data referensi yang terlampir dalam tanda kurung gigi.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Periksa integritas referensial kolom kode pos

Contoh aturan berikut memeriksa bahwa lebih dari 90% dari nilai-nilai dalam `zipcode` kolom dalam dataset utama, hadir di `zipcode` kolom dalam `reference` dataset.

```
ReferentialIntegrity "zipcode" "reference.zipcode" >= 0.9
```

Contoh: Periksa integritas referensial kolom kota dan negara bagian

Dalam contoh berikut, kolom yang berisi informasi kota dan negara bagian ada di kumpulan data utama dan kumpulan data referensi. Nama-nama kolom berbeda di kedua kumpulan data. Aturan memeriksa apakah himpunan nilai kolom dalam dataset primer persis sama dengan himpunan nilai kolom dalam dataset referensi.

```
ReferentialIntegrity "city,state" "reference.{ref_city,ref_state}" = 1.0
```

Contoh aturan dinamis

- `ReferentialIntegrity "city,state" "reference.{ref_city,ref_state}" > avg(last(10))`
- `ReferentialIntegrity "city,state" "reference.{ref_city,ref_state}" between min(last(10)) - 1 and max(last(10)) + 1`

RowCount

Memeriksa jumlah baris dataset terhadap ekspresi yang diberikan. Dalam ekspresi, Anda dapat menentukan jumlah baris atau rentang baris menggunakan operator seperti > dan<.

Sintaksis

```
RowCount <EXPRESSION>
```

- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Cek numerik hitungan baris

Contoh aturan berikut memeriksa apakah jumlah baris berada dalam rentang tertentu.

```
RowCount between 10 and 100
RowCount between 1 and 50 where "Customer_ID < 10"
```

Contoh aturan dinamis

```
RowCount > avg(lats(10)) *0.8
```

RowCountMatch

Memeriksa rasio jumlah baris dari kumpulan data utama dan jumlah baris dari kumpulan data referensi terhadap ekspresi yang diberikan.

Sintaksis

```
RowCountMatch <REFERENCE_DATASET_ALIAS> <EXPRESSION>
```

- REFERENCE_DATASET_ALIAS — Alias dari dataset referensi yang digunakan untuk membandingkan jumlah baris.

Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek

- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Pemeriksaan jumlah baris terhadap kumpulan data referensi

Contoh aturan berikut memeriksa apakah jumlah baris dari dataset primer setidaknya 90% dari jumlah baris dari dataset referensi.

```
RowCountMatch "reference" >= 0.9
```

StandardDeviation

Memeriksa standar deviasi dari semua nilai dalam kolom terhadap ekspresi yang diberikan.

Sintaksis

```
StandardDeviation <COL_NAME> <EXPRESSION>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Standar deviasi

Contoh aturan berikut memeriksa apakah standar deviasi nilai dalam kolom bernama `colA` kurang dari nilai yang ditentukan.

```
StandardDeviation "Star_Rating" < 1.5
StandardDeviation "Salary" < 3500 where "Customer_ID < 10"
```

Contoh aturan dinamis

- `StandardDeviation "colA" > avg(last(10) + 0.1`
- `StandardDeviation "colA" between min(last(10)) - 1 and max(last(10)) + 1`

Perilaku nol

`StandardDeviation`Aturan akan mengabaikan baris dengan `NULL` nilai dalam perhitungan standar deviasi. Sebagai contoh:

```
+---+-----------+-----------+
|id |units1     |units2     |
+---+-----------+-----------+
|100|0          |0          |
|101|null       |0          |
|102|20         |20         |
|103|null       |0          |
|104|40         |40         |
+---+-----------+-----------+
```

Standar deviasi kolom tidak `units1` akan mempertimbangkan baris 101 dan 103 dan hasilnya menjadi 16,33. Standar deviasi untuk kolom `units2` akan menghasilkan 16.

Jumlah

Memeriksa jumlah semua nilai dalam kolom terhadap ekspresi yang diberikan.

Sintaksis

```
Sum <COL_NAME> <EXPRESSION>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Jumlah

Contoh aturan berikut memeriksa apakah jumlah semua nilai dalam kolom melebihi ambang batas yang diberikan.

```
Sum "transaction_total" > 500000
Sum "Salary" < 55600 where "Customer_ID < 10"
```

Contoh aturan dinamis

- `Sum "ColA" > avg(last(10))`
- `Sum "colA" between min(last(10)) - 1 and max(last(10)) + 1`

Perilaku nol

SumAturan akan mengabaikan baris dengan NULL nilai dalam perhitungan jumlah. Sebagai contoh:

```
+---+-----------+
|id |units      |
+---+-----------+
|100|0          |
|101|null       |
|102|20         |
|103|null       |
|104|40         |
+---+-----------+
```

Jumlah kolom tidak `units` akan mempertimbangkan baris 101 dan 103 dan hasilnya menjadi (0 + 20 + 40) = 60.

SchemaMatch

Memeriksa apakah skema kumpulan data utama cocok dengan skema kumpulan data referensi. Pemeriksaan skema dilakukan kolom demi kolom. Skema dua kolom cocok jika nama identik dan jenisnya identik. Urutan kolom tidak masalah.

Sintaksis

```
SchemaMatch <REFERENCE_DATASET_ALIAS> <EXPRESSION>
```

- REFERENCE_DATASET_ALIAS — Alias dari dataset referensi yang digunakan untuk membandingkan skema.

  Jenis kolom yang didukung: Byte, Desimal, Ganda, Float, Integer, Panjang, Pendek
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: SchemaMatch

Contoh aturan berikut memeriksa apakah skema dataset primer sama persis dengan skema dataset referensi.

```
SchemaMatch "reference" = 1.0
```

Keunikan

Memeriksa persentase nilai unik dalam kolom terhadap ekspresi yang diberikan. Nilai unik terjadi tepat sekali.

Sintaksis

```
Uniqueness <COL_NAME> <EXPRESSION>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Tipe kolom apa pun
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Persentase keunikan

Contoh aturan berikut memeriksa apakah persentase nilai unik dalam kolom cocok dengan kriteria numerik tertentu.

```
Uniqueness "email" = 1.0
Uniqueness "Customer_ID" != 1.0 where "Customer_ID < 10"
```

Contoh aturan dinamis

- `Uniqueness "colA" between min(last(10)) and max(last(10))`
- `Uniqueness "colA" >= avg(last(10))`

UniqueValueRatio

Memeriksa rasio nilai unik kolom terhadap ekspresi yang diberikan. Rasio nilai unik adalah pecahan dari nilai unik dibagi dengan jumlah semua nilai yang berbeda dalam kolom. Nilai unik terjadi tepat satu kali, sedangkan nilai yang berbeda terjadi setidaknya sekali.

Misalnya, himpunan `[a, a, b]` berisi satu nilai unik (b) dan dua nilai berbeda (adanb). Jadi rasio nilai unik dari himpunan adalah ½ = 0,5.

Sintaksis

```
UniqueValueRatio <COL_NAME> <EXPRESSION>
```

- COL_NAME — Nama kolom yang ingin Anda evaluasi terhadap aturan kualitas data.

  Jenis kolom yang didukung: Tipe kolom apa pun
- EXPRESSION — Ekspresi yang dijalankan terhadap respons tipe aturan untuk menghasilkan nilai Boolean. Untuk informasi selengkapnya, lihat Ekspresi.

Contoh: Rasio nilai unik

Contoh ini memeriksa rasio nilai unik kolom terhadap rentang nilai.

```
UniqueValueRatio "test_score" between 0 and 0.5
UniqueValueRatio "Customer_ID" between 0 and 0.9 where "Customer_ID < 10"
```

Contoh aturan dinamis

- `UniqueValueRatio "colA" > avg(last(10))`
- `UniqueValueRatio "colA" <= index(last(10),2) + std(last(5))`

DetectAnomalies

Mendeteksi anomali untuk aturan kualitas data tertentu. Setiap eksekusi DetectAnomalies aturan menghasilkan penghematan nilai yang dievaluasi untuk aturan yang diberikan. Ketika ada cukup data yang dikumpulkan, algoritma deteksi anomali mengambil semua data historis untuk aturan yang diberikan dan menjalankan deteksi anomali. DetectAnomalies aturan gagal ketika anomali terdeteksi. Info lebih lanjut tentang anomali apa yang terdeteksi dapat diperoleh dari Pengamatan.

Sintaksis

```
      DetectAnomalies <RULE_NAME> <RULE_PARAMETERS>
```

RULE_NAME— Nama aturan yang ingin Anda evaluasi dan deteksi anomali. Aturan yang didukung:

- "RowCount"
- “Kelengkapan”
- “Keunikan”
- “Berarti”
- “Jumlah”
- "StandardDeviation"
- “Entropi”
- "DistinctValuesCount"
- "UniqueValueRatio"
- "ColumnLength"
- "ColumnValues"
- "ColumnCorrelation"

RULE_PARAMETERS— beberapa aturan memerlukan parameter tambahan untuk dijalankan. Lihat dokumentasi aturan yang diberikan untuk melihat parameter yang diperlukan.

Contoh: Anomali untuk RowCount

Misalnya, jika kami ingin mendeteksi RowCount anomali, kami berikan RowCount sebagai nama aturan.

```
DetectAnomalies "RowCount"
```

Contoh: Anomali untuk ColumnLength

Misalnya, jika kita ingin mendeteksi ColumnLength anomali, kita berikan ColumnLength sebagai nama aturan dan nama kolom.

```
DetectAnomalies "ColumnLength" "id"
```

Menggunakan API untuk mengukur dan mengelola kualitas data

Topik ini menjelaskan cara menggunakan API untuk mengukur dan mengelola kualitas data.

Daftar Isi

Prasyarat

- Pastikan versi boto3 Anda mutakhir sehingga menyertakan API Kualitas Data AWS Glue terbaru.
- Pastikan versi AWS CLI Anda mutakhir, sehingga menyertakan CLI terbaru.

Jika Anda menggunakan tugas AWS Glue untuk menjalankan API ini, Anda dapat menggunakan opsi berikut untuk memperbarui pustaka boto3 ke versi terbaru:

```
—additional-python-modules boto3==<version>
```

Bekerja dengan rekomendasi Kualitas Data AWS Glue

Untuk memulai rekomendasi AWS Glue Data Quality, jalankan:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client

    def start_data_quality_rule_recommendation_run(self, database_name, table_name,
 role_arn):
        """
        Starts a recommendation run that is used to generate rules when you don't
 know what rules to write. AWS Glue Data Quality analyzes the data and comes up with
 recommendations for a potential ruleset. You can then triage the ruleset and modify
 the generated ruleset to your liking.
```

```
        :param database_name: The name of the AWS Glue database which contains the
dataset.
        :param table_name: The name of the AWS Glue table against which we want a
recommendation
        :param role_arn: The Amazon Resource Name (ARN) of an AWS Identity and Access
Management (IAM) role that grants permission to let AWS Glue access the resources it
needs.

        """
        try:
            response = self.client.start_data_quality_rule_recommendation_run(
                DataSource={
                    'GlueTable': {
                        'DatabaseName': database_name,
                        'TableName': table_name
                    }
                },
                Role=role_arn
            )
        except ClientError as err:
            logger.error(
                "Couldn't start data quality recommendation run %s. Here's why: %s:
%s", name,
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response['RunId']
```

Untuk menjalankan rekomendasi, Anda dapat menggunakan `pushDownPredicates` atau `catalogPartitionPredicates` untuk meningkatkan kinerja dan menjalankan rekomendasi hanya pada partisi tertentu dari sumber katalog Anda.

```
client.start_data_quality_rule_recommendation_run(
            DataSource={
                'GlueTable': {
                    'DatabaseName': database_name,
                    'TableName': table_name,
                    'AdditionalOptions': {
                        'pushDownPredicate': "year=2022"
                    }
                }
            },
            Role=role_arn,
```

```
            NumberOfWorkers=2,
            CreatedRulesetName='<rule_set_name>'
  )
```

Untuk mendapatkan hasil rekomendasi AWS Glue Data Quality, jalankan:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_data_quality_rule_recommendation_run(self, run_id):
        """
        Gets the specified recommendation run that was used to generate rules.

        :param run_id: The id of the data quality recommendation run

        """
        try:
            response =
 self.client.get_data_quality_rule_recommendation_run(RunId=run_id)
        except ClientError as err:
            logger.error(
                "Couldn't get data quality recommendation run %. Here's why: %s: %s",
 run_id,
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Dari objek respon di atas, Anda dapat mengekstrak RuleSet yang direkomendasikan oleh run, untuk digunakan dalam langkah selanjutnya:

```
print(response['RecommendedRuleset'])

Rules = [
    RowCount between 2000 and 8000,
    IsComplete "col1",
    IsComplete "col2",
    StandardDeviation "col3" between 58138330.8 and 64258155.09,
```

```
    ColumnValues "col4" between 1000042965 and 1214474826,
    IsComplete "col5"
]
```

Untuk mendapatkan daftar semua rekomendasi Anda berjalan yang dapat difilter dan dicantumkan:

```
response = client.list_data_quality_rule_recommendation_runs(
    Filter={
        'DataSource': {
            'GlueTable': {
                'DatabaseName': '<database_name>',
                'TableName': '<table_name>'
            }
        }
)
```

Untuk membatalkan tugas rekomendasi Kualitas Data AWS Glue yang ada:

```
response = client.cancel_data_quality_rule_recommendation_run(
    RunId='dqrun-d4b6b01957fdd79e59866365bf9cb0e40fxxxxxxx'
)
```

Bekerja dengan Aturan Kualitas Data AWS Glue

Untuk membuat aturan Kualitas Data AWS Glue:

```
response = client.create_data_quality_ruleset(
    Name='<ruleset_name>',
    Ruleset='Rules = [IsComplete "col1", IsPrimaryKey "col2", RowCount between 2000 and
 8000]',
    TargetTable={
        'TableName': '<table_name>',
        'DatabaseName': '<database_name>'
    }
)
```

Untuk mendapatkan aturan kualitas data:

```
response = client.get_data_quality_ruleset(
    Name='<ruleset_name>'
```

```
)
print(response)
```

Anda dapat menggunakan API ini untuk kemudian mengekstrak set aturan:

```
print(response['Ruleset'])
```

Untuk membuat daftar semua aturan kualitas data untuk tabel:

```
response = client.list_data_quality_rulesets()
```

Anda dapat menggunakan kondisi filter dalam API untuk memfilter semua aturan yang dilampirkan ke database atau tabel tertentu:

```
response = client.list_data_quality_rulesets(
    Filter={
        'TargetTable': {
            'TableName': '<table_name>',
            'DatabaseName': '<database_name>'
        }
    },
)
```

Untuk memperbarui kumpulan aturan kualitas data:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def update_data_quality_ruleset(self, ruleset_name, ruleset_string):
        """
        Update an AWS Glue Data Quality Ruleset

        :param ruleset_name: The name of the AWS Glue Data Quality ruleset to update
        :param ruleset_string: The DQDL ruleset string to update the ruleset with

        """
```

```
        try:
            response = self.client.update_data_quality_ruleset(
                Name=ruleset_name,
                Ruleset=ruleset_string
            )
        except ClientError as err:
            logger.error(
                "Couldn't update the AWS Glue Data Quality ruleset. Here's why: %s:
 %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Untuk menghapus kumpulan aturan kualitas data:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def delete_data_quality_ruleset(self, ruleset_name):
        """
        Delete a AWS Glue Data Quality Ruleset

        :param ruleset_name: The name of the AWS Glue Data Quality ruleset to delete

        """
        try:
            response = self.client.delete_data_quality_ruleset(
                Name=ruleset_name
            )
        except ClientError as err:
            logger.error(
                "Couldn't delete the AWS Glue Data Quality ruleset. Here's why: %s:
 %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Bekerja dengan AWS Glue Data Quality berjalan

Untuk memulai menjalankan AWS Glue Data Quality:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def start_data_quality_ruleset_evaluation_run(self, database_name, table_name,
 role_name, ruleset_list):
        """
        Start an AWS Glue Data Quality evaluation run

        :param database_name: The name of the AWS Glue database which contains the
 dataset.
        :param table_name: The name of the AWS Glue table against which we want to
 evaluate.
        :param role_arn: The Amazon Resource Name (ARN) of an AWS Identity and Access
 Management (IAM) role that grants permission to let AWS Glue access the resources it
 needs.
        :param ruleset_list: The list of AWS Glue Data Quality ruleset names to
 evaluate.

        """
        try:
            response = client.start_data_quality_ruleset_evaluation_run(
                DataSource={
                    'GlueTable': {
                        'DatabaseName': database_name,
                        'TableName': table_name
                    }
                },
                Role=role_name,
                RulesetNames=ruleset_list
            )
        except ClientError as err:
            logger.error(
                "Couldn't start the AWS Glue Data Quality Run. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
```

```
            raise
        else:
            return response['RunId']
```

Ingat bahwa Anda dapat meneruskan `catalogPartitionPredicate` parameter `pushDownPredicate` atau untuk memastikan kualitas data Anda berjalan hanya menargetkan sekumpulan partisi tertentu dalam tabel katalog Anda. Misalnya:

```
response = client.start_data_quality_ruleset_evaluation_run(
    DataSource={
        'GlueTable': {
            'DatabaseName': '<database_name>',
            'TableName': '<table_name>',
            'AdditionalOptions': {
                'pushDownPredicate': 'year=2023'
            }
        }
    },
    Role='<role_name>',
    NumberOfWorkers=5,
    Timeout=123,
    AdditionalRunOptions={
        'CloudWatchMetricsEnabled': False
    },
    RulesetNames=[
        '<ruleset_name>',
    ]
)
```

Untuk mendapatkan informasi tentang AWS Glue Data Quality run:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_data_quality_ruleset_evaluation_run(self, run_id):
        """
        Get details about an AWS Glue Data Quality Run
```

```
        :param run_id: The AWS Glue Data Quality run ID to look up

        """
        try:
            response = self.client.get_data_quality_ruleset_evaluation_run(
                RunId=run_id
            )
        except ClientError as err:
            logger.error(
                "Couldn't look up the AWS Glue Data Quality run ID. Here's why: %s:
 %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Untuk mendapatkan hasil dari AWS Glue Data Quality menjalankan:

Untuk menjalankan Kualitas Data AWS Glue tertentu, Anda dapat mengekstrak hasil evaluasi run menggunakan metode berikut:

```
response = client.get_data_quality_ruleset_evaluation_run(
    RunId='d4b6b01957fdd79e59866365bf9cb0e40fxxxxxxx'
)

resultID = response['ResultIds'][0]

response = client.get_data_quality_result(
    ResultId=resultID
)

print(response['RuleResults'])
```

Untuk mencantumkan semua AWS Glue Data Quality Anda berjalan:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
```

```
        self.glue_client = glue_client

    def list_data_quality_ruleset_evaluation_runs(self, database_name, table_name):
        """
        Lists all the AWS Glue Data Quality runs against a given table

        :param database_name: The name of the database where the data quality runs
        :param table_name: The name of the table against which the data quality runs
 were created

        """
        try:
            response = self.client.list_data_quality_ruleset_evaluation_runs(
                Filter={
                    'DataSource': {
                        'GlueTable': {
                            'DatabaseName': database_name,
                            'TableName': table_name
                        }
                    }
                }
            )
        except ClientError as err:
            logger.error(
                "Couldn't list the AWS Glue Quality runs. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Anda dapat memodifikasi klausa filter untuk hanya menampilkan hasil antara waktu tertentu atau berjalan terhadap tabel tertentu.

Untuk menghentikan proses AWS Glue Data Quality yang sedang berlangsung:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client
```

```
    def cancel_data_quality_ruleset_evaluation_run(self, result_id):
        """
        Cancels a given AWS Glue Data Quality run

        :param result_id: The result id of a AWS Glue Data Quality run to cancel

        """
        try:
            response = self.client.cancel_data_quality_ruleset_evaluation_run(
                ResultId=result_id
            )
        except ClientError as err:
            logger.error(
                "Couldn't cancel the AWS Glue Data Quality run. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Bekerja dengan hasil Kualitas Data AWS Glue

Untuk mendapatkan hasil AWS Glue Data Quality run Anda:

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 AWS Glue client.
        """
        self.glue_client = glue_client

    def get_data_quality_result(self, result_id):
        """
        Outputs the result of an AWS Glue Data Quality Result

        :param result_id: The result id of an AWS Glue Data Quality run

        """
        try:
            response = self.client.get_data_quality_result(
                ResultId=result_id
            )
        except ClientError as err:
```

```
            logger.error(
                "Couldn't get the AWS Glue Data Quality result. Here's why: %s: %s",
                err.response['Error']['Code'], err.response['Error']['Message'])
            raise
        else:
            return response
```

Untuk membatalkan tugas rekomendasi Kualitas Data AWS Glue yang ada:

Diberikan AWS Glue Data Quality run ID, Anda dapat mengekstrak ID hasil untuk kemudian mendapatkan hasil aktual, seperti yang ditunjukkan di bawah ini:

```
response = client.get_data_quality_ruleset_evaluation_run(
    RunId='dqrun-abca77ee126abe1378c1da1ae0750xxxxxxxx'
)

resultID = response['ResultIds'][0]

response = client.get_data_quality_result(
    ResultId=resultID
)

print(resp['RuleResults'])
```

Menyiapkan peringatan, penerapan, dan penjadwalan

Topik ini menjelaskan cara mengatur peringatan, penerapan, dan penjadwalan untuk Kualitas Data AWS Glue.

Daftar Isi

Menyiapkan peringatan dan pemberitahuan dalam integrasi Amazon EventBridge

AWS Glue Data Quality mendukung penerbitan EventBridge peristiwa, yang dipancarkan setelah menyelesaikan evaluasi aturan Kualitas Data. Dengan ini, Anda dapat dengan mudah mengatur peringatan ketika aturan kualitas data gagal.

Berikut adalah contoh peristiwa saat Anda mengevaluasi kumpulan aturan kualitas data di Katalog Data. Dengan informasi ini, Anda dapat meninjau data yang tersedia dengan Amazon EventBridge. Anda dapat mengeluarkan panggilan API tambahan untuk mendapatkan detail selengkapnya. Misalnya, panggil `get_data_quality_result` API dengan ID hasil untuk mendapatkan detail eksekusi tertentu.

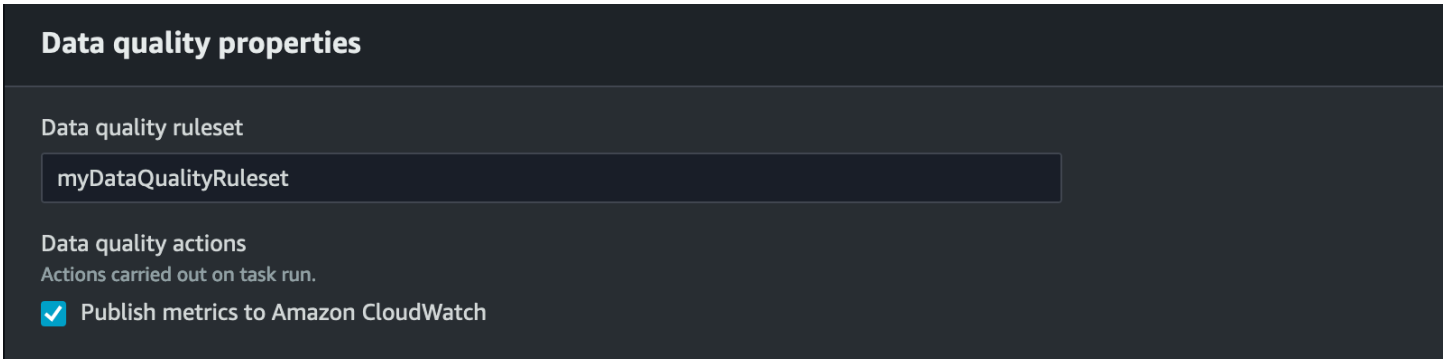
```
{
    "version":"0",
    "id":"abcdef00-1234-5678-9abc-def012345678",
    "detail-type":"Data Quality Evaluation Results Available",
    "source":"aws.glue-dataquality",
    "account":"123456789012",
    "time":"2017-09-07T18:57:21Z",
    "region":"us-west-2",
    "resources":[],
    "detail":{
        "context": {
                    "contextType": "GLUE_DATA_CATALOG",
                    "runId":"dqrun-12334567890",
                    "databaseName": "db-123",
                    "tableName": "table-123",
                    "catalogId": "123456789012"
                    },
        "resultID": "dqresult-12334567890",
        "rulesetNames": ["rulset1"],
        "state":"SUCCEEDED",
        "score": 1.00,
        "rulesSucceeded": 100,
        "rulesFailed": 0,
        "rulesSkipped": 0
    }
}
```

Berikut adalah contoh peristiwa yang dipublikasikan saat Anda mengevaluasi set aturan kualitas data di notebook Glue ETL atau AWS AWS Glue Studio.

```
{
    "version":"0",
    "id":"abcdef00-1234-5678-9abc-def012345678",
    "detail-type":"Data Quality Evaluation Results Available",
    "source":"aws.glue-dataquality",
    "account":"123456789012",
    "time":"2017-09-07T18:57:21Z",
    "region":"us-west-2",
    "resources":[],
    "detail":{
        "context": {
                    "contextType": "GLUE_JOB",
                    "jobId": "jr-12334567890",
                    "jobName": "dq-eval-job-1234",
                    "evaluationContext": "",
                    }
        "resultID": "dqresult-12334567890",
        "rulesetNames": ["rulset1"],
        "state":"SUCCEEDED",
        "score": 1.00
        "rulesSucceeded": 100,
        "rulesFailed": 0,
        "rulesSkipped": 0
    }
}
```

Untuk evaluasi Kualitas Data berjalan baik di Katalog Data dan dalam pekerjaan ETL, Amazon CloudWatch opsi Publikasikan metrik ke, yang dipilih secara default, harus tetap dipilih agar EventBridge penerbitan berfungsi.

Menyiapkan EventBridge notifikasi



Untuk menerima peristiwa yang dipancarkan dan menentukan target, Anda harus mengonfigurasi aturan Amazon EventBridge . Untuk membuat aturan:

1. Buka EventBridge konsol Amazon.
2. Pilih Aturan di bawah bagian Bus pada bilah navigasi.
3. Pilih Buat aturan.
4. Tentang Tentukan Rincian Aturan:
   a. Untuk Nama, masukkan`myDQRule`.
   b. Masukkan deskripsi (opsional).
   c. Untuk bus acara, pilih bus acara Anda. Jika Anda tidak memilikinya, biarkan sebagai default.
   d. Untuk Jenis aturan pilih Aturan dengan pola acara lalu pilih Berikutnya.
5. Pada Pola Acara Bangun:
   a. Untuk sumber acara pilih AWS acara atau acara EventBridge mitra.
   b. Lewati bagian contoh acara.
   c. Untuk metode pembuatan pilih Gunakan formulir pola.
   d. Untuk pola acara:
      i. Pilih AWS layanan untuk sumber Acara.
      ii. Pilih Glue Data Quality untuk AWS layanan.
      iii. Pilih Hasil Evaluasi Kualitas Data yang Tersedia untuk jenis Acara.
      iv. Pilih GAGAL untuk status tertentu. Kemudian Anda melihat pola acara yang mirip dengan berikut ini:

```
{
  "source": ["aws.glue-dataquality"],
  "detail-type": ["Data Quality Evaluation Results Available"],
  "detail": {
```

```
    "state": ["FAILED"]
  }
}
```

v. Untuk opsi konfigurasi lainnya, lihatOpsi konfigurasi tambahan untuk pola acara.

6. Pada Target Pilih:
   a. Untuk Jenis Target pilih AWS layanan.
   b. Gunakan menu tarik-turun Pilih target untuk memilih AWS layanan yang Anda inginkan untuk terhubung (SNS, Lambda, SQS, dll.), Lalu pilih Berikutnya.
7. Pada Konfigurasi tag klik Tambahkan tag baru untuk menambahkan tag opsional lalu pilih Berikutnya.
8. Anda melihat halaman ringkasan dari semua pilihan. Pilih Buat aturan di bagian bawah.

Opsi konfigurasi tambahan untuk pola acara

Selain memfilter acara Anda tentang keberhasilan atau kegagalan, Anda mungkin ingin memfilter lebih lanjut peristiwa pada parameter yang berbeda.

Untuk melakukan ini, buka bagian Pola Acara, dan pilih Edit pola untuk menentukan parameter tambahan. Perhatikan bahwa bidang dalam pola peristiwa peka huruf besar/kecil. Berikut ini adalah contoh konfigurasi pola acara.

Untuk menangkap peristiwa dari tabel tertentu yang mengevaluasi kumpulan aturan tertentu, gunakan jenis pola ini:

```
{
  "source": ["aws.glue-dataquality"],
  "detail-type": ["Data Quality Evaluation Results Available"],
  "detail": {
    "context": {
      "contextType": ["GLUE_DATA_CATALOG"],
      "databaseName": "db-123",
       "tableName": "table-123",
    },
    "rulesetNames": ["ruleset1", "ruleset2"]
    "state": ["FAILED"]
  }
}
```

Untuk menangkap peristiwa dari pekerjaan tertentu dalam pengalaman ETL, gunakan jenis pola ini:

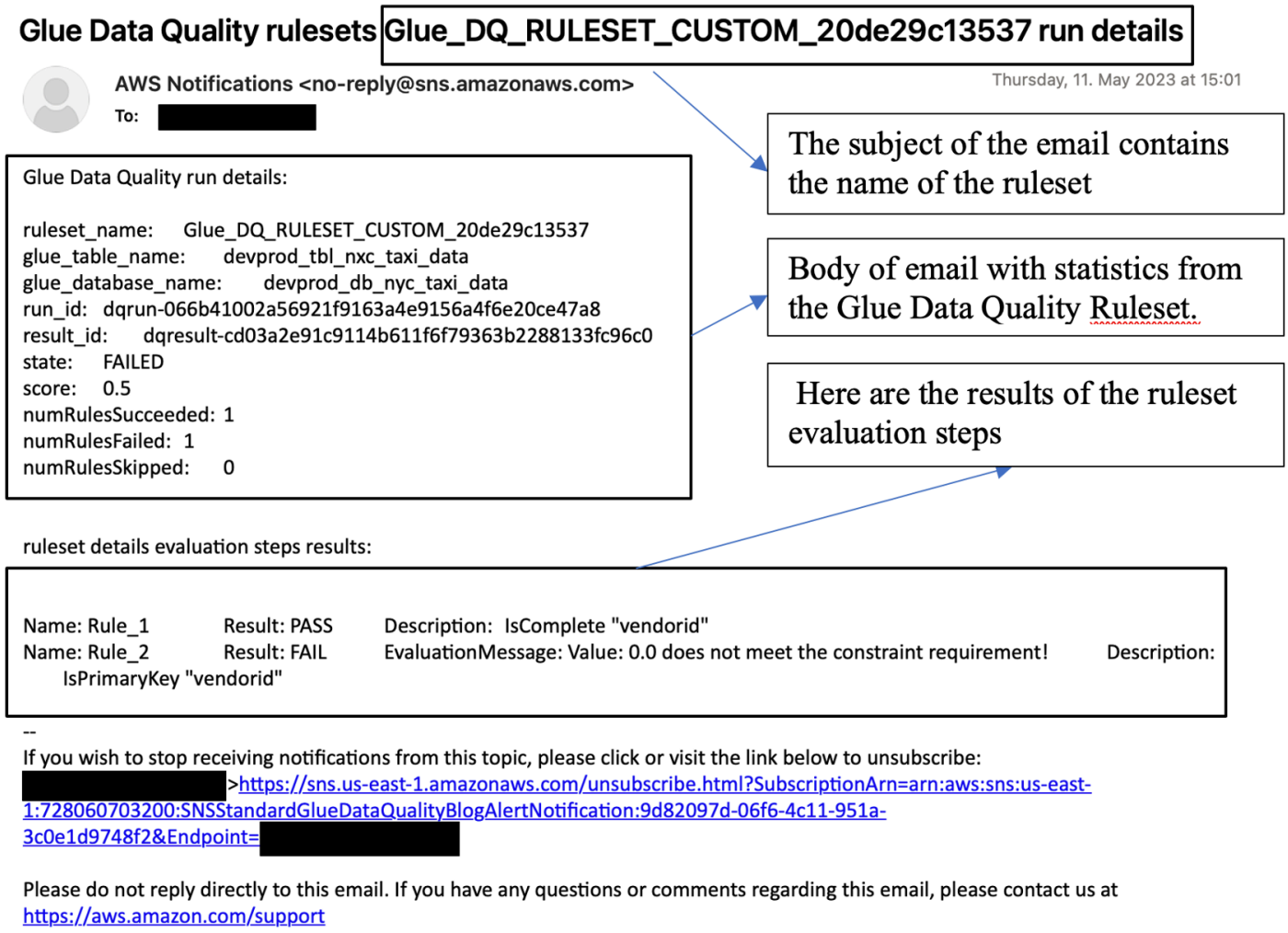
```
{
  "source": ["aws.glue-dataquality"],
  "detail-type": ["Data Quality Evaluation Results Available"],
  "detail": {
    "context": {
      "contextType": ["GLUE_JOB"],
      "jobName": ["dq_evaluation_job1", "dq_evaluation_job2"]
    },
    "state": ["FAILED"]
  }
}
```

Untuk menangkap peristiwa dengan skor di bawah ambang tertentu (misalnya 70%):

```
{
  "source": ["aws.glue-dataquality"],
  "detail-type": ["Data Quality Evaluation Results Available"],
  "detail": {
    "score": [{
      "numeric": ["<=", 0.7]
    }]
  }
}
```

Memformat pemberitahuan sebagai email

Terkadang Anda perlu mengirim pemberitahuan email yang diformat dengan baik ke tim bisnis Anda. Anda dapat menggunakan Amazon EventBridge dan AWS Lambda untuk mencapai ini.



Kode contoh berikut dapat digunakan untuk memformat pemberitahuan kualitas data Anda untuk menghasilkan email.

```
import boto3
import json
from datetime import datetime

sns_client = boto3.client('sns')
glue_client = boto3.client('glue')

sns_topic_arn = 'arn:aws:sns:<region-code>:<account-id>:<sns-topic-name>'



def lambda_handler(event, context):
    log_metadata = {}
```

```
    message_text = ""
    subject_text = ""

    if event['detail']['context']['contextType'] == 'GLUE_DATA_CATALOG':
        log_metadata['ruleset_name'] = str(event['detail']['rulesetNames'][0])
        log_metadata['tableName'] = str(event['detail']['context']['tableName'])
        log_metadata['databaseName'] = str(event['detail']['context']['databaseName'])
        log_metadata['runId'] = str(event['detail']['context']['runId'])
        log_metadata['resultId'] = str(event['detail']['resultId'])
        log_metadata['state'] = str(event['detail']['state'])
        log_metadata['score'] = str(event['detail']['score'])
        log_metadata['numRulesSucceeded'] = str(event['detail']['numRulesSucceeded'])
        log_metadata['numRulesFailed'] = str(event['detail']['numRulesFailed'])
        log_metadata['numRulesSkipped'] = str(event['detail']['numRulesSkipped'])

        message_text += "Glue Data Quality run details:\n"
        message_text += "ruleset_name: {}\n".format(log_metadata['ruleset_name'])
        message_text += "glue_table_name: {}\n".format(log_metadata['tableName'])
        message_text += "glue_database_name: {}\n".format(log_metadata['databaseName'])
        message_text += "run_id: {}\n".format(log_metadata['runId'])
        message_text += "result_id: {}\n".format(log_metadata['resultId'])
        message_text += "state: {}\n".format(log_metadata['state'])
        message_text += "score: {}\n".format(log_metadata['score'])
        message_text += "numRulesSucceeded:
{}\n".format(log_metadata['numRulesSucceeded'])
        message_text += "numRulesFailed: {}\n".format(log_metadata['numRulesFailed'])
        message_text += "numRulesSkipped: {}\n".format(log_metadata['numRulesSkipped'])

        subject_text = "Glue Data Quality ruleset {} run
details".format(log_metadata['ruleset_name'])

    else:
        log_metadata['ruleset_name'] = str(event['detail']['rulesetNames'][0])
        log_metadata['jobName'] = str(event['detail']['context']['jobName'])
        log_metadata['jobId'] = str(event['detail']['context']['jobId'])
        log_metadata['resultId'] = str(event['detail']['resultId'])
        log_metadata['state'] = str(event['detail']['state'])
        log_metadata['score'] = str(event['detail']['score'])

        log_metadata['numRulesSucceeded'] = str(event['detail']['numRulesSucceeded'])
        log_metadata['numRulesFailed'] = str(event['detail']['numRulesFailed'])
        log_metadata['numRulesSkipped'] = str(event['detail']['numRulesSkipped'])

        message_text += "Glue Data Quality run details:\n"
```

```
        message_text += "ruleset_name: {}\n".format(log_metadata['ruleset_name'])
        message_text += "glue_job_name: {}\n".format(log_metadata['jobName'])
        message_text += "job_id: {}\n".format(log_metadata['jobId'])
        message_text += "result_id: {}\n".format(log_metadata['resultId'])
        message_text += "state: {}\n".format(log_metadata['state'])
        message_text += "score: {}\n".format(log_metadata['score'])
        message_text += "numRulesSucceeded:
{}\n".format(log_metadata['numRulesSucceeded'])
        message_text += "numRulesFailed: {}\n".format(log_metadata['numRulesFailed'])
        message_text += "numRulesSkipped: {}\n".format(log_metadata['numRulesSkipped'])

        subject_text = "Glue Data Quality ruleset {} run
details".format(log_metadata['ruleset_name'])

    resultID = str(event['detail']['resultId'])
    response = glue_client.get_data_quality_result(ResultId=resultID)
    RuleResults = response['RuleResults']
    message_text += "\n\nruleset details evaluation steps results:\n\n"
    subresult_info = []

    for dic in RuleResults:
        subresult = "Name: {}\t\tResult: {}\t\tDescription: \t{}".format(dic['Name'],
dic['Result'], dic['Description'])
        if 'EvaluationMessage' in dic:
            subresult += "\t\tEvaluationMessage: {}".format(dic['EvaluationMessage'])
        subresult_info.append({
            'Name': dic['Name'],
            'Result': dic['Result'],
            'Description': dic['Description'],
            'EvaluationMessage': dic.get('EvaluationMessage', '')
        })
        message_text += "\n" + subresult

    log_metadata['resultrun'] = subresult_info



    sns_client.publish(
        TopicArn=sns_topic_arn,
        Message=message_text,
        Subject=subject_text
    )

    return {
```

```
        'statusCode': 200,
        'body': json.dumps('Message published to SNS topic')
    }
```

Siapkan peringatan dan notifikasi dalam integrasi CloudWatch

Pendekatan yang kami rekomendasikan adalah menyiapkan peringatan kualitas data menggunakan Amazon EventBridge, karena Amazon EventBridge memerlukan penyiapan satu kali untuk mengingatkan pelanggan. Namun, beberapa pelanggan lebih suka Amazon CloudWatch karena keakraban. Untuk pelanggan seperti itu, kami menawarkan integrasi dengan Amazon CloudWatch.

Setiap evaluasi Kualitas Data AWS Glue memancarkan sepasang metrik bernama `glue.data.quality.rules.passed` (menunjukkan sejumlah aturan yang lolos) dan `glue.data.quality.rules.failed` (menunjukkan jumlah aturan yang gagal) per kualitas data yang dijalankan. Anda dapat menggunakan metrik yang dipancarkan ini untuk membuat alarm untuk mengingatkan pengguna jika kualitas data tertentu berjalan di bawah ambang batas. Untuk memulai dengan mengatur alarm yang akan mengirim email melalui pemberitahuan Amazon SNS, ikuti langkah-langkah di bawah ini:

Untuk memulai dengan mengatur alarm yang akan mengirim email melalui pemberitahuan Amazon SNS, ikuti langkah-langkah di bawah ini:

1. Buka CloudWatch konsol Amazon.
2. Pilih Semua metrik di bawah Metrik. Anda akan melihat namespace tambahan di bawah ruang nama Kustom berjudul Glue Data Quality.

   > **Note**
   >
   > Saat memulai proses AWS Glue Data Quality, pastikan CloudWatch kotak centang Publikasikan metrik ke Amazon diaktifkan. Jika tidak, metrik untuk proses tertentu tidak akan dipublikasikan ke Amazon CloudWatch.

   Di bawah `Glue Data Quality` namespace, Anda dapat melihat metrik yang dipancarkan per tabel, per kumpulan aturan. Untuk tujuan topik ini, kami akan menggunakan `glue.data.quality.rules.failed` aturan dan alarm jika nilai ini melebihi 1 (menunjukkan bahwa, jika kami melihat sejumlah evaluasi aturan yang gagal lebih besar dari 1, kami ingin diberi tahu).

3. Untuk membuat alarm, pilih Semua alarm di bawah Alarm.
4. Pilih Buat alarm.
5. Pilih Pilih Metrik.
6. Pilih `glue.data.quality.rules.failed` metrik yang sesuai dengan tabel yang telah Anda buat, lalu pilih Pilih metrik.
7. Di bawah tab Tentukan metrik dan kondisi, di bawah bagian Metrik:
   a. Untuk Statistik pilih Jumlah.
   b. Untuk Periode, pilih 1 menit.
8. Di bawah bagian Kondisi:
   a. Untuk Jenis ambang batas, pilih Statis.
   b. Untuk Setiap kali glue.data.quality.rules.failed adalah... , pilih Greater/Equal.
   c. Untuk dari... , masukkan 1 sebagai nilai ambang batas.

   Pilihan ini menyiratkan bahwa jika `glue.data.quality.rules.failed` metrik memancarkan nilai lebih besar dari atau sama dengan 1, kami akan memicu alarm. Namun, jika tidak ada data, kami akan memperlakukannya sebagai dapat diterima.
9. Pilih Selanjutnya.
10. Pada tindakan Konfigurasi:
    a. Untuk bagian Pemicu status alarm, pilih Dalam alarm.
    b. Untuk Kirim pemberitahuan ke bagian topik SNS berikut, pilih Buat topik baru untuk mengirim pemberitahuan melalui topik SNS baru.
    c. Untuk titik akhir Email yang akan menerima notifikasi, masukkan alamat email Anda. Kemudian klik Buat Topik.
    d. Pilih Selanjutnya.
11. Untuk nama Alarm, masukkan`myFirstDQAlarm`, lalu pilih Berikutnya.
12. Anda melihat halaman ringkasan dari semua pilihan. Pilih Buat alarm di bagian bawah.

Anda sekarang dapat melihat alarm dibuat dari dasbor CloudWatch alarm Amazon.

Menanyakan hasil kualitas data untuk membangun dasbor

Anda mungkin ingin membangun dasbor untuk menampilkan hasil kualitas data Anda. Ada dua cara untuk melakukan hal ini:

Siapkan Amazon EventBridge dengan kode berikut untuk menulis data ke Amazon S3:

```
import boto3
import json
from datetime import datetime


s3_client = boto3.client('s3')
glue_client = boto3.client('glue')


s3_bucket = 's3-bucket-name'

def write_logs(log_metadata):
    try:
        filename = datetime.now().strftime("%m%d%Y%H%M%S") + ".json"
        key_opts = {
            'year': datetime.now().year,
            'month': "{:02d}".format(datetime.now().month),
            'day': "{:02d}".format(datetime.now().day),
            'filename': filename
        }
        s3key = "gluedataqualitylogs/year={year}/month={month}/day={day}/
{filename}".format(**key_opts)
        s3_client.put_object(Bucket=s3_bucket, Key=s3key,
 Body=json.dumps(log_metadata))
    except Exception as e:
        print(f'Error writing logs to S3: {e}')


def lambda_handler(event, context):
    log_metadata = {}
    message_text = ""
    subject_text = ""

    if event['detail']['context']['contextType'] == 'GLUE_DATA_CATALOG':
        log_metadata['ruleset_name'] = str(event['detail']['rulesetNames'][0])
        log_metadata['tableName'] = str(event['detail']['context']['tableName'])
        log_metadata['databaseName'] = str(event['detail']['context']['databaseName'])
        log_metadata['runId'] = str(event['detail']['context']['runId'])
        log_metadata['resultId'] = str(event['detail']['resultId'])
        log_metadata['state'] = str(event['detail']['state'])
        log_metadata['score'] = str(event['detail']['score'])
        log_metadata['numRulesSucceeded'] = str(event['detail']['numRulesSucceeded'])
```

```
        log_metadata['numRulesFailed'] = str(event['detail']['numRulesFailed'])
        log_metadata['numRulesSkipped'] = str(event['detail']['numRulesSkipped'])

        message_text += "Glue Data Quality run details:\n"
        message_text += "ruleset_name: {}\n".format(log_metadata['ruleset_name'])
        message_text += "glue_table_name: {}\n".format(log_metadata['tableName'])
        message_text += "glue_database_name: {}\n".format(log_metadata['databaseName'])
        message_text += "run_id: {}\n".format(log_metadata['runId'])
        message_text += "result_id: {}\n".format(log_metadata['resultId'])
        message_text += "state: {}\n".format(log_metadata['state'])
        message_text += "score: {}\n".format(log_metadata['score'])
        message_text += "numRulesSucceeded:
{}\n".format(log_metadata['numRulesSucceeded'])
        message_text += "numRulesFailed: {}\n".format(log_metadata['numRulesFailed'])
        message_text += "numRulesSkipped: {}\n".format(log_metadata['numRulesSkipped'])

        subject_text = "Glue Data Quality ruleset {} run
details".format(log_metadata['ruleset_name'])

    else:
        log_metadata['ruleset_name'] = str(event['detail']['rulesetNames'][0])
        log_metadata['jobName'] = str(event['detail']['context']['jobName'])
        log_metadata['jobId'] = str(event['detail']['context']['jobId'])
        log_metadata['resultId'] = str(event['detail']['resultId'])
        log_metadata['state'] = str(event['detail']['state'])
        log_metadata['score'] = str(event['detail']['score'])

        log_metadata['numRulesSucceeded'] = str(event['detail']['numRulesSucceeded'])
        log_metadata['numRulesFailed'] = str(event['detail']['numRulesFailed'])
        log_metadata['numRulesSkipped'] = str(event['detail']['numRulesSkipped'])

        message_text += "Glue Data Quality run details:\n"
        message_text += "ruleset_name: {}\n".format(log_metadata['ruleset_name'])
        message_text += "glue_job_name: {}\n".format(log_metadata['jobName'])
        message_text += "job_id: {}\n".format(log_metadata['jobId'])
        message_text += "result_id: {}\n".format(log_metadata['resultId'])
        message_text += "state: {}\n".format(log_metadata['state'])
        message_text += "score: {}\n".format(log_metadata['score'])
        message_text += "numRulesSucceeded:
{}\n".format(log_metadata['numRulesSucceeded'])
        message_text += "numRulesFailed: {}\n".format(log_metadata['numRulesFailed'])
        message_text += "numRulesSkipped: {}\n".format(log_metadata['numRulesSkipped'])
```

```
        subject_text = "Glue Data Quality ruleset {} run
details".format(log_metadata['ruleset_name'])

    resultID = str(event['detail']['resultId'])
    response = glue_client.get_data_quality_result(ResultId=resultID)
    RuleResults = response['RuleResults']
    message_text += "\n\nruleset details evaluation steps results:\n\n"
    subresult_info = []

    for dic in RuleResults:
        subresult = "Name: {}\t\tResult: {}\t\tDescription: \t{}".format(dic['Name'],
dic['Result'], dic['Description'])
        if 'EvaluationMessage' in dic:
            subresult += "\t\tEvaluationMessage: {}".format(dic['EvaluationMessage'])
        subresult_info.append({
            'Name': dic['Name'],
            'Result': dic['Result'],
            'Description': dic['Description'],
            'EvaluationMessage': dic.get('EvaluationMessage', '')
        })
        message_text += "\n" + subresult

    log_metadata['resultrun'] = subresult_info

    write_logs(log_metadata)


    return {
        'statusCode': 200,
        'body': json.dumps('Message published to SNS topic')
    }
```

Setelah menulis ke Amazon S3, Anda dapat menggunakan perayap AWS Glue untuk mendaftar ke Athena dan menanyakan tabel.

Konfigurasikan lokasi Amazon S3 selama evaluasi kualitas data::

Saat menjalankan tugas kualitas data di AWS Glue Data Catalog atau AWS Glue ETL, Anda dapat memberikan lokasi Amazon S3 untuk menulis hasil kualitas data ke Amazon S3. Anda dapat menggunakan sintaks di bawah ini untuk membuat tabel dengan mereferensikan target untuk membaca hasil kualitas data.

Perhatikan bahwa Anda harus menjalankan `MSCK REPAIR TABLE` kueri `CREATE EXTERNAL TABLE` dan secara terpisah.

```
CREATE EXTERNAL TABLE <my_table_name>(
    catalogid string,
    databasename string,
    tablename string,
    dqrunid string,
    evaluationstartedon timestamp,
    evaluationcompletedon timestamp,
    rule string,
    outcome string,
    failurereason string,
    evaluatedmetrics string)
PARTITIONED BY (
    `year` string,
    `month` string,
    `day` string)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
WITH SERDEPROPERTIES (

 'paths'='catalogId,databaseName,dqRunId,evaluatedMetrics,evaluationCompletedOn,evaluationStart
STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://glue-s3-dq-bucket-us-east-2-results/'
TBLPROPERTIES (
    'classification'='json',
    'compressionType'='none',
    'typeOfData'='file');
```

```
MSCK REPAIR TABLE <my_table_name>;
```

Setelah Anda membuat tabel di atas, Anda dapat menjalankan kueri analitik menggunakan Amazon Athena.

Menerapkan aturan kualitas data menggunakan AWS CloudFormation

Anda dapat menggunakan AWS CloudFormation untuk membuat aturan kualitas data. Untuk informasi lebih lanjut, lihat AWS CloudFormation untuk AWS Glue.

Menjadwalkan aturan kualitas data

Anda dapat menjadwalkan aturan kualitas data menggunakan metode berikut:

- Jadwalkan aturan kualitas data dari Katalog Data: tidak ada pengguna kode yang dapat menggunakan opsi ini untuk dengan mudah menjadwalkan pemindaian kualitas data mereka. AWS Glue Data Quality akan membuat jadwal di Amazon EventBridge. Untuk menjadwalkan aturan kualitas data:
  - Arahkan ke ruleset dan klik Run.
  - Dalam frekuensi Jalankan, pilih jadwal yang diinginkan dan berikan Nama Tugas. Nama Tugas ini adalah nama jadwal Anda di EventBridge.
- Gunakan Amazon EventBridge dan AWS Step Functions untuk mengatur evaluasi dan rekomendasi untuk aturan kualitas data.

Memecahkan masalah kesalahan Kualitas Data AWS Glue

Jika Anda menemukan kesalahan dalam Kualitas Data AWS Glue, gunakan solusi berikut untuk membantu Anda menemukan sumber masalah dan memperbaikinya.

Daftar Isi

- Aturan Dinamis
- Pengecualian di Kelas Pengguna: org.apache.spark.sql. AnalysisException: org.apache.hadoop.hive.ql.metadata. HiveException
- UNCLASSIFIED_ERROR; IllegalArgumentException: Kesalahan Penguraian: Tidak ada aturan atau penganalisis yang disediakan., tidak ada alternatif yang layak pada input

Kesalahan: modul Kualitas Data AWS Glue hilang

Pesan kesalahan: Tidak ada modul bernama 'awsgluedq'.

Resolusi: Kesalahan ini terjadi ketika Anda menjalankan AWS Glue Data Quality dalam versi yang tidak didukung. AWS Glue Data Quality hanya didukung di Glue versi 3.0 dan yang lebih baru.

Kesalahan: izin AWS Lake Formation tidak mencukupi

Pesan kesalahan: Pengecualian di Kelas Pengguna:: Izin Lake Formation tidak mencukupi pada impact_sdg_involvement (Layanan`com.amazonaws.services.glue.model.AccessDeniedException`:; Kode Status: 400; Kode Kesalahan:AWS Glue; ID Permintaan: 465ae693-b7ba-4df0-a4e4-6b17xxxxxxxx AccessDeniedException; Proxy: null).

Resolusi: Anda harus memberikan izin yang cukup di AWS Lake Formation.

Kesalahan: rulesets tidak diberi nama unik

Pesan kesalahan: Pengecualian di Kelas Pengguna:... services.glue.model. AlreadyExistsException: Aturan lain dengan nama yang sama sudah ada.

Resolusi: Aturan bersifat global dan harus unik.

Kesalahan: tabel dengan karakter khusus

Pesan galat: Pengecualian di Kelas Pengguna: org.apache.spark.sql. AnalysisException: tidak dapat menyelesaikan kolom masukan “C” yang diberikan: [primary.data_end_time, primary.data_start_time, primary.end_time, primary.last_updated, primary.message, primary.process_date, primary.rowhash, primary.run_by, primary.run_id, primary.start_time, primary.status]; baris 1 pos 44;.

Resolusi: Ada batasan saat ini bahwa AWS Glue Data Quality tidak dapat dijalankan pada tabel yang memiliki karakter khusus seperti “.”.

Kesalahan: kesalahan overflow dengan kumpulan aturan besar

Pesan galat: Pengecualian di Kelas Pengguna: java.lang. StackOverflowError.

Resolusi: Jika Anda memiliki aturan besar lebih dari aturan 2K, Anda mungkin mengalami masalah ini. Pecahkan aturan Anda menjadi beberapa aturan.

Kesalahan: status aturan keseluruhan gagal

Kondisi kesalahan: Set Aturan Saya berhasil, tetapi status aturan keseluruhan saya gagal.

Resolusi: Kesalahan ini kemungkinan besar terjadi karena Anda memilih opsi untuk menerbitkan metrik ke Amazon CloudWatch saat menerbitkan. Jika kumpulan data Anda ada dalam VPC, VPC Anda mungkin tidak mengizinkan AWS Glue mempublikasikan metrik ke Amazon. CloudWatch Dalam hal ini, Anda> harus menyiapkan titik akhir untuk VPC Anda untuk mengakses Amazon. CloudWatch

AnalysisException: Tidak dapat memverifikasi keberadaan basis data default

Kondisi kesalahan: AnalysisException: Tidak dapat memverifikasi keberadaan database default: com.amazonaws.services.glue.model. AccessDeniedException: Izin Lake Formation tidak mencukupi secara default (Layanan:; Kode Status: 400; Kode Kesalahan:AWS Glue; ID Permintaan: XXXXXX-XXXX-XXXX-XXXX-XXXX-XXXX-XXXX-XXXX-XXXX-XXXXXX AccessDeniedException; Proxy: null)

Resolusi: Dalam integrasi katalog AWS Glue pekerjaan, AWS Glue selalu mencoba untuk memeriksa apakah database default ada atau tidak menggunakan AWS Glue`GetDatabase API`. Ketika izin `DESCRIBE` Lake Formation tidak diberikan, atau `GetDatabase IAM` izin diberikan, maka pekerjaan gagal saat memverifikasi keberadaan database default.

Untuk menyelesaikan:

1. Tambahkan `DESCRIBE` izin di Lake Formation untuk database default.
2. Konfigurasikan peran IAM yang dilampirkan ke AWS Glue pekerjaan sebagai Pembuat Database di Lake Formation. Ini akan secara otomatis membuat database default dan memberikan izin Lake Formation yang diperlukan untuk peran tersebut.
3. Nonaktifkan `--enable-data-catalog` opsi. (Ini ditampilkan sebagai Use Data Catalog as the Hive metastore di). AWS Glue Studio

Jika Anda tidak memerlukan Data Catalog integrasi Spark SQL dalam pekerjaan, Anda dapat menonaktifkannya.

Pesan Kesalahan: Peta kunci yang disediakan tidak cocok untuk bingkai data yang diberikan

Kondisi kesalahan: Peta kunci yang disediakan tidak cocok untuk bingkai data yang diberikan.

Resolusi: Anda menggunakan DataSetMatchruletype dan tombol join memiliki duplikat. Kunci bergabung Anda harus unik dan tidak boleh NULL. Jika Anda tidak dapat memiliki kunci gabungan yang unik, pertimbangkan untuk menggunakan tipe aturan lain seperti AggregateMatchuntuk mencocokkan data ringkasan.

Pengecualian di Kelas Pengguna: java.lang. RuntimeException : Gagal mengambil data. Periksa log masuk CloudWatch untuk mendapatkan detail lebih lanjut

Kondisi kesalahan: Pengecualian di Kelas Pengguna: java.lang. RuntimeException : Gagal mengambil data. Periksa log masuk CloudWatch untuk mendapatkan detail lebih lanjut.

Resolusi: Ini terjadi ketika Anda membuat aturan DQ pada tabel berbasis Amazon S3 yang dibandingkan dengan Amazon RDS atau. Amazon Redshift Dalam kasus ini, AWS Glue tidak dapat memuat koneksi. Sebagai gantinya, cobalah untuk mengatur aturan DQ pada Amazon Redshift atau kumpulan data Amazon RDS. Ini adalah bug yang diketahui.

LAUNCH ERROR: Kesalahan mengunduh dari S3 untuk ember

Kondisi kesalahan: LAUNCH ERROR: Kesalahan mengunduh dari S3 untuk ember:`aws-glue-ml-data-quality-assets-us-east-1, key: jars/aws-glue-ml-data-quality-etl.jar.Access Denied (Service: Amazon S3; Status Code: 403; Please refer logs for details)` .

Resolusi: Izin dalam peran yang diteruskan ke AWS Glue Data Quality harus mengizinkan pembacaan dari lokasi Amazon S3 sebelumnya. Kebijakan IAM ini harus dilampirkan pada peran:

```
{
  "Sid": "allowS3",
```

```
  "Effect": "Allow",
  "Action": "s3:GetObject",
  "Resource": "arn:aws:s3:::aws-glue-ml-data-quality-assets-<region>/*"
}
```

Lihat otorisasi Kualitas Data untuk izin terperinci. Pustaka ini diperlukan untuk mengevaluasi kualitas data untuk kumpulan data Anda.

InvalidInputException (status: 400): DataQuality aturan tidak dapat diuraikan

Kondisi kesalahan: InvalidInputException (status: 400): DataQuality aturan tidak dapat diuraikan.

Resolusi: Ada banyak kemungkinan untuk kesalahan ini. Salah satu kemungkinan adalah bahwa aturan Anda mungkin memiliki tanda kutip tunggal. Verifikasi bahwa mereka berada dalam tanda kutip ganda. Sebagai contoh:

```
Rules = [
ColumnValues "tipo_vinculo" in ["CODO", "DOCO", "COCO", "DODO"] AND "categoria" = 'ES'
    AND "cod_bandera" = 'CEP'
```

Ubah ini menjadi:

```
Rules = [
(ColumnValues "tipovinculo" in [ "CODO", "DOCO", "COCO", "DODO"]) AND (ColumnValues
 "categoria" = "ES")
    AND (ColumnValues "codbandera" = "CEP")
]
```

Kesalahan: Eventbridge tidak memicu pekerjaan Glue DQ berdasarkan jadwal yang saya siapkan

Kondisi kesalahan: Eventbridge tidak AWS Glue Data Quality memicu pekerjaan berdasarkan jadwal yang saya siapkan.

Resolusi: Peran yang memicu pekerjaan mungkin tidak memiliki izin yang tepat. Pastikan bahwa peran yang Anda gunakan untuk memulai pekerjaan memiliki izin yang disebutkan dalam penyiapan IAM yang diperlukan untuk evaluasi penjadwalan berjalan.

Kesalahan CustomSQL

Kondisi kesalahan: `The output from CustomSQL must contain at least one column that matches the input dataset for AWS Glue Data Quality to provide row level results. The SQL query is a valid query but no columns from the SQL result are present in the Input Dataset. Ensure that matching columns are returned from the SQL.`

Resolusi: Kueri SQL valid tetapi verifikasi bahwa Anda hanya memilih kolom dari tabel utama. Memilih fungsi agregat seperti jumlah, mengandalkan kolom dari primer dapat mengakibatkan kesalahan ini.

Kondisi kesalahan: `There was a problem when executing your SQL statement: cannot resolve "Col"`.

Resolusi: Kolom ini tidak ada di tabel utama.

Kondisi kesalahan: `The columns that are returned from the SQL statement should only belong to the primary table. "In this case, some columns ( Col ) belong to reference table"`.

Resolusi: Dalam kueri SQL saat Anda bergabung dengan tabel primer dengan tabel referensi lainnya, verifikasi bahwa pernyataan pilih Anda hanya memiliki nama kolom dari tabel utama Anda untuk menghasilkan hasil tingkat baris untuk tabel utama.

Aturan Dinamis

Kondisi kesalahan: `Dynamic rules require job context, and cannot be evaluated in interactive session or data preview.`.

Penyebab: Pesan galat ini mungkin muncul di hasil pratinjau data Anda, atau di sesi interaktif lainnya, saat aturan DQ dinamis ada di kumpulan aturan Anda. Aturan dinamis mengacu pada metrik historis yang terkait dengan nama pekerjaan dan konteks evaluasi tertentu, sehingga tidak dapat dievaluasi dalam sesi interaktif.

Resolusi: Menjalankan AWS Glue pekerjaan Anda akan menghasilkan metrik historis, yang dapat direferensikan dalam pekerjaan selanjutnya untuk pekerjaan yang sama.

Kondisi kesalahan:

- `[RuleType] rule only supports simple atomic operands in thresholds.`.
- `Function last not yet implemented for [RuleType] rule.`

Resolusi: Aturan dinamis umumnya didukung untuk semua tipe aturan DQDL dalam ekspresi numerik (lihat Referensi DQDL). Namun, beberapa aturan yang menghasilkan beberapa metrik, ColumnValues dan ColumnLength, belum didukung.

Kondisi kesalahan: `Binary expression operands must resolve to a single number.`.

Penyebab: Aturan dinamis mendukung ekspresi biner, sepertiRowCount > avg(last(5)) * 0.9. Di sini, ekspresi biner adalahavg(last(5)) * 0.9. Aturan ini valid karena kedua operan avg(last(5)) dan 0.9 penyelesaian ke nomor tunggal. Contoh yang salah adalahRowCount > last(5) * 0.9, karena last(5) akan menghasilkan daftar yang tidak dapat dibandingkan secara bermakna dengan jumlah baris saat ini.

Resolusi: Gunakan fungsi agregasi untuk mengurangi operan bernilai daftar menjadi satu nomor.

Kondisi kesalahan:

- `Rule threshold results in list, and a single value is expected. Use aggregation functions to produce a single value. Valid example: sum(last(10)), avg(last(10)).`
- `Rule threshold results in empty list, and a single value is expected.`

Penyebab: Aturan dinamis dapat digunakan untuk membandingkan beberapa fitur kumpulan data Anda dengan nilai historisnya. Fungsi terakhir memungkinkan untuk pengambilan beberapa nilai historis, jika argumen integer positif disediakan. Misalnya, `last(5)` akan mengambil lima nilai terbaru terakhir yang diamati dalam menjalankan pekerjaan untuk aturan Anda.

Resolusi: Fungsi agregasi harus digunakan untuk mengurangi nilai-nilai ini menjadi satu angka untuk membuat perbandingan yang berarti dengan nilai yang diamati dalam menjalankan pekerjaan saat ini.

Contoh yang valid:

- `RowCount >= avg(last(5))`
- `RowCount > last(1)`

- `RowCount < last()`

Contoh tidak valid:. `RowCount > last(5)`

Kondisi kesalahan:

- `Function index used in threshold requires positive integer argument.`
- `Index argument must be an integer. Valid syntax example: RowCount > index(last(10, 2)), which means RowCount must be greater than third most recent execution from last 10 job runs.`

Resolusi: Saat membuat aturan dinamis, Anda dapat menggunakan fungsi `index` agregasi untuk memilih satu nilai historis dari daftar. Misalnya`RowCount > index(last(5)`, 1) akan memeriksa apakah jumlah baris yang diamati dalam pekerjaan saat ini benar-benar lebih besar daripada jumlah baris terbaru kedua yang diamati untuk pekerjaan Anda. `index`diindeks nol.

Kondisi kesalahan: `IllegalArgumentException: Parsing Error: Rule Type: DetectAnomalies is not valid`.

Resolusi: Deteksi anomali hanya tersedia di AWS Glue 4.0.

Kondisi kesalahan: `IllegalArgumentException: Parsing Error: Unexpected condition for rule of type ... no viable alternative at input ...`.

Catatan: `...` dinamis. Contoh:`IllegalArgumentException: Parsing Error: Unexpected condition for rule of type RowCount with number return type, line 4:19 no viable alternative at input '>last'`.

Resolusi: Deteksi anomali hanya tersedia di AWS Glue 4.0.

Pengecualian di Kelas Pengguna: org.apache.spark.sql. AnalysisException: org.apache.hadoop.hive.ql.metadata. HiveException

Kondisi kesalahan : `Exception in User Class: org.apache.spark.sql.AnalysisException: org.apache.hadoop.hive.ql.metadata.HiveException: Unable to fetch table mailpiece_submitted. StorageDescriptor#InputFormat cannot be null for table: mailpiece_submitted (Service: null; Status Code: 0; Error Code: null; Request ID: null; Proxy: null)`

Penyebab: Anda menggunakan Apache Iceberg di AWS Glue Data Catalog dan atribut Format Input di AWS Glue Data Catalog kosong.

Resolusi: Masalah ini terjadi ketika Anda menggunakan CustomSQL ruletype dalam aturan DQ Anda. Salah satu cara untuk memperbaikinya adalah dengan menggunakan “primer “atau menambahkan nama katalog `glue_catalog.` ke`<database>.<table> in Custom ruletype`.

UNCLASSIFIED_ERROR; IllegalArgumentException: Kesalahan Penguraian: Tidak ada aturan atau penganalisis yang disediakan., tidak ada alternatif yang layak pada input

Kondisi kesalahan `: UNCLASSIFIED_ERROR; IllegalArgumentException: Parsing Error: No rules or analyzers provided., no viable alternative at input`

Resolusi: DQDL tidak dapat diurai. Ada beberapa contoh di mana ini bisa terjadi. Jika Anda menggunakan aturan komposit, pastikan mereka memiliki tanda kurung yang tepat.

```
(RowCount >= avg(last(10)) * 0.6) and (RowCount <= avg(last(10)) * 1.4) instead of
      RowCount >= avg(last(10)) * 0.6 and RowCount <= avg(last(10)) * 1.4
```

Integrasi data Amazon Q di AWS Glue

Integrasi data Amazon Q AWS Glue adalah kemampuan AI generatif baru AWS Glue yang memungkinkan insinyur data dan pengembang ETL untuk membangun pekerjaan integrasi data menggunakan bahasa alami. Insinyur dan pengembang dapat meminta Amazon Q untuk membuat pekerjaan, memecahkan masalah, dan menjawab pertanyaan tentang AWS Glue dan integrasi data.

Apa itu Amazon Q?

> **Note**
>
> Didukung oleh Amazon Bedrock: AWS mengimplementasikan deteksi penyalahgunaan otomatis. Karena integrasi data Amazon Q dibangun di Amazon Bedrock, pengguna dapat memanfaatkan sepenuhnya kontrol yang diterapkan di Amazon Bedrock untuk menegakkan keselamatan, keamanan, dan penggunaan kecerdasan buatan (AI) yang bertanggung jawab.

Amazon Q adalah asisten percakapan yang didukung kecerdasan buatan (AI) generatif yang dapat membantu Anda memahami, membangun, memperluas, dan mengoperasikan AWS aplikasi. Model yang mendukung Amazon Q telah ditambah dengan AWS konten berkualitas tinggi untuk memberi Anda jawaban yang lebih lengkap, dapat ditindaklanjuti, dan direferensikan untuk mempercepat pembangunan Anda. AWS Untuk informasi selengkapnya, lihat Apa itu Amazon Q?

Apa itu integrasi data Amazon Q AWS Glue?

Integrasi data Amazon Q AWS Glue mencakup kemampuan berikut:

- Obrolan — Integrasi data Amazon Q AWS Glue dapat menjawab pertanyaan bahasa alami dalam bahasa Inggris tentang AWS Glue dan domain integrasi data seperti konektor AWS Glue sumber dan tujuan, pekerjaan AWS Glue ETL, Katalog Data, crawler dan AWS Lake Formation, dan dokumentasi fitur lainnya, dan praktik terbaik. Integrasi data Amazon Q dalam AWS Glue merespons dengan step-by-step instruksi, dan menyertakan referensi ke sumber informasinya.
- Pembuatan kode integrasi data — Integrasi data Amazon Q AWS Glue dapat menjawab pertanyaan tentang skrip AWS Glue ETL, dan menghasilkan kode baru yang diberikan pertanyaan bahasa alami dalam bahasa Inggris.

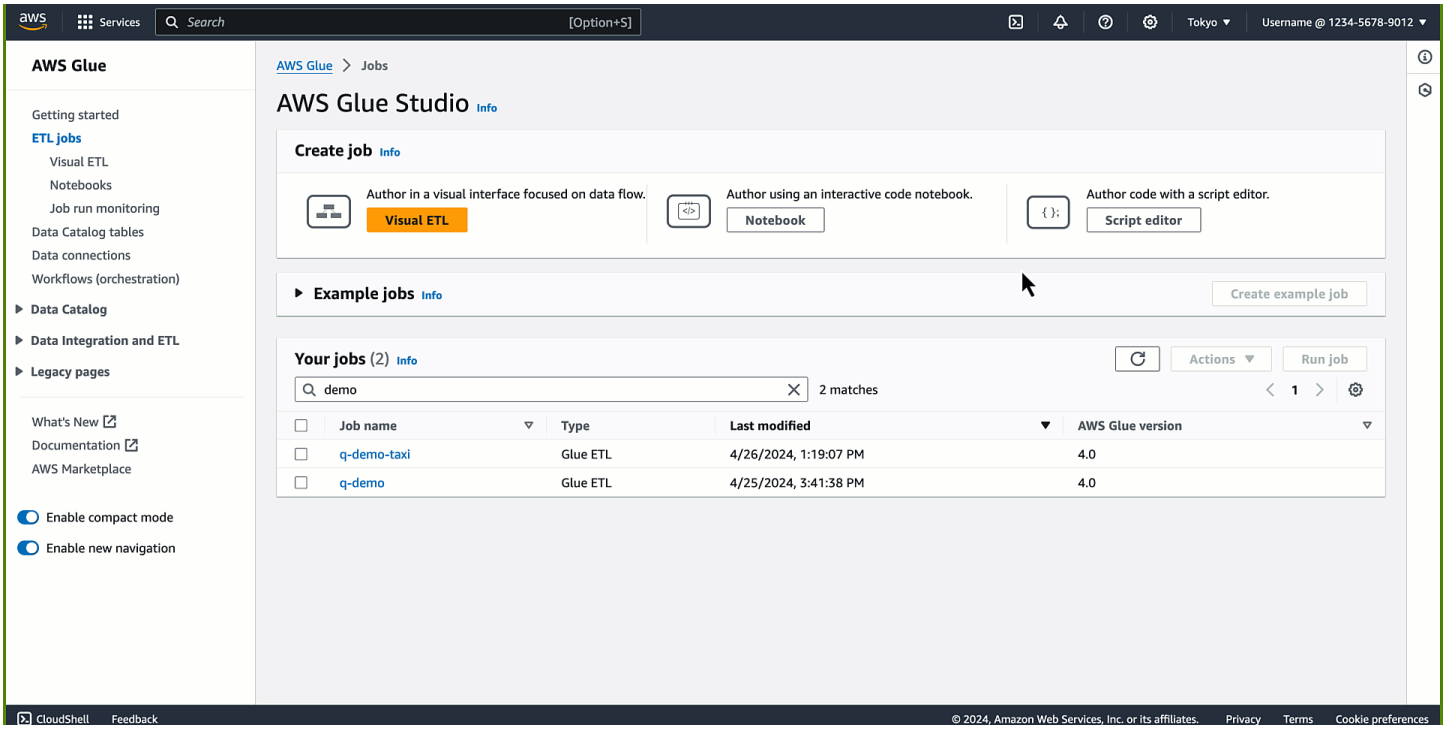
- Pemecahan masalah — Integrasi data Amazon Q AWS Glue dibuat untuk membantu Anda memahami kesalahan dalam AWS Glue pekerjaan dan memberikan step-by-step instruksi, untuk akar penyebab dan menyelesaikan masalah Anda.

> **Note**
>
> Integrasi data Amazon Q di AWS Glue tidak menggunakan konteks percakapan Anda untuk menginformasikan tanggapan future selama durasi percakapan Anda. Setiap percakapan dengan integrasi data Amazon Q AWS Glue tidak tergantung pada percakapan Anda sebelumnya atau masa depan.

Bekerja dengan integrasi data Amazon Q di AWS Glue?

Di panel Amazon Q Anda dapat meminta Amazon Q menghasilkan kode untuk skrip AWS Glue ETL, atau menjawab pertanyaan tentang AWS Glue fitur atau memecahkan masalah kesalahan. Responsnya adalah skrip ETL PySpark dengan step-by-step instruksi untuk menyesuaikan skrip, meninjau dan menjalankannya. Untuk pertanyaan, respons dihasilkan berdasarkan basis pengetahuan integrasi data dengan ringkasan dan URL sumber untuk referensi.

Misalnya, Anda dapat meminta Amazon Q untuk "Harap berikan skrip Glue yang berbunyi dari Snowflake, mengganti nama bidang, dan menulis ke Redshift" dan sebagai tanggapan, integrasi data Amazon Q AWS Glue akan mengembalikan skrip AWS Glue pekerjaan yang dapat melakukan tindakan yang diminta. Anda dapat meninjau kode yang dihasilkan untuk memastikan kode tersebut memenuhi maksud yang diminta. Jika puas, Anda dapat menerapkannya sebagai AWS Glue pekerjaan dalam produksi. Anda dapat memecahkan masalah pekerjaan dengan meminta integrasi untuk menjelaskan kesalahan dan kegagalan, dan untuk mengusulkan solusi. Amazon Q dapat menjawab pertanyaan tentang AWS Glue atau praktik terbaik integrasi data.



Berikut ini adalah contoh pertanyaan yang menunjukkan bagaimana integrasi data Amazon Q AWS Glue dapat membantu Anda membangun AWS Glue:

AWS Glue Pembuatan kode ETL:

- Tulis AWS Glue skrip yang membaca JSON dari S3, mengubah bidang menggunakan pemetaan terapan dan menulis ke Amazon Redshift
- Bagaimana cara menulis AWS Glue skrip untuk membaca dari DynamoDB, menerapkan transformasi dan menulis ke DropNullFields S3 sebagai Parket?
- Beri saya AWS Glue skrip yang membaca dari MySQL, menjatuhkan beberapa bidang berdasarkan logika bisnis saya, dan menulis ke Snowflake
- Menulis AWS Glue pekerjaan untuk membaca dari DynamoDB dan menulis ke S3 sebagai JSON
- Bantu saya mengembangkan AWS Glue skrip untuk Katalog AWS Glue Data ke S3
- Tulis AWS Glue pekerjaan untuk membaca JSON dari S3, jatuhkan nulls dan tulis ke Redshift

AWS Glue penjelasan fitur:

- Bagaimana cara menggunakan Kualitas AWS Glue Data?
- Bagaimana cara menggunakan bookmark AWS Glue pekerjaan?
- Bagaimana cara mengaktifkan AWS Glue penskalaan otomatis?

- Apa perbedaan antara frame AWS Glue dinamis dan frame data Spark?
- Apa saja jenis koneksi yang didukung oleh AWS Glue?

AWS Glue pemecahan masalah:

- Bagaimana cara mengatasi kesalahan Out Of Memory (OOM) pada pekerjaan? AWS Glue
- Apa saja pesan kesalahan yang mungkin Anda lihat saat mengatur Kualitas AWS Glue Data dan bagaimana Anda bisa memperbaikinya?
- Bagaimana cara memperbaiki AWS Glue pekerjaan dengan kesalahan akses Amazon S3 ditolak?
- Bagaimana cara mengatasi masalah dengan pengocokan data pada AWS Glue pekerjaan?

Praktik terbaik untuk berinteraksi dengan integrasi data Amazon Q

Berikut ini adalah praktik terbaik untuk berinteraksi dengan integrasi data Amazon Q:

- Saat berinteraksi dengan integrasi data Amazon Q, ajukan pertanyaan spesifik, ulangi ketika Anda memiliki permintaan yang rumit, dan verifikasi jawaban untuk akurasi.
- Saat memberikan petunjuk integrasi data dalam bahasa alami, sespesifik mungkin untuk membantu asisten memahami dengan tepat apa yang Anda butuhkan. Alih-alih meminta “ekstrak data dari S3,” berikan detail lebih lanjut seperti “tulis AWS Glue skrip yang mengekstrak file JSON dari S3.”
- Tinjau skrip yang dihasilkan sebelum menjalankannya untuk memastikan akurasi. Jika skrip yang dihasilkan memiliki kesalahan atau tidak sesuai dengan maksud Anda, berikan instruksi kepada asisten tentang cara memperbaikinya.
- Teknologi AI generatif adalah hal baru dan mungkin ada kesalahan, kadang-kadang disebut halusinasi, dalam tanggapannya. Uji dan tinjau semua kode untuk kesalahan dan kerentanan sebelum menggunakannya di lingkungan atau beban kerja Anda.

Integrasi data Amazon Q dalam peningkatan AWS Glue layanan

Untuk membantu integrasi data Amazon Q dalam AWS Glue memberikan informasi yang paling relevan tentang AWS layanan, kami dapat menggunakan konten tertentu dari Amazon Q, seperti pertanyaan yang Anda ajukan kepada Amazon Q dan tanggapannya, untuk peningkatan layanan.

Untuk informasi tentang konten apa yang kami gunakan dan cara memilih keluar, lihat Peningkatan layanan Pengembang Amazon Q di Panduan Pengguna Pengembang Amazon Q.

Pertimbangan

Pertimbangkan item berikut sebelum Anda menggunakan integrasi data Amazon Q di AWS Glue:

- Saat ini, pembuatan kode hanya berfungsi dengan PySpark kernel. Kode yang dihasilkan adalah untuk AWS Glue pekerjaan berdasarkan Python Spark.
- Untuk informasi tentang kombinasi kemampuan pembuatan kode yang didukung dari integrasi data Amazon Q AWS Glue, lihatKemampuan pembuatan kode yang didukung.

Menyiapkan integrasi data Amazon Q di AWS Glue

Bagian berikut menyediakan informasi pengaturan integrasi data Amazon Q di AWS Glue.

Topik

- Mengonfigurasi izin IAM

Mengonfigurasi izin IAM

Topik ini menjelaskan izin IAM yang Anda konfigurasikan untuk pengalaman obrolan Amazon Q, dan pengalaman notebook AWS Glue Studio.

Topik

- Mengonfigurasi izin IAM untuk obrolan Amazon Q
- Mengkonfigurasi izin IAM untuk notebook Studio AWS Glue

Mengonfigurasi izin IAM untuk obrolan Amazon Q

Pemberian izin ke API yang digunakan oleh integrasi data Amazon Q AWS Glue memerlukan izin AWS Identity and Access Management (IAM) yang sesuai. Anda dapat memperoleh izin dengan melampirkan AWS kebijakan kustom berikut ke identitas IAM Anda (seperti pengguna, peran, atau grup):

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:StartCompletion",
                "glue:GetCompletion"
            ],
            "Resource": [
                "arn:aws:glue:*:*:completion/*"
            ]
        }
    ]
}
```

Mengkonfigurasi izin IAM untuk notebook Studio AWS Glue

Untuk mengaktifkan integrasi data Amazon Q di notebook AWS Glue Studio, pastikan izin berikut dilampirkan ke peran IAM notebook:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:StartCompletion",
                "glue:GetCompletion"
            ],
            "Resource": [
                "arn:aws:glue:*:*:completion/*"
            ]
        },
        {
            "Sid": "CodeWhispererPermissions",
            "Effect": "Allow",
            "Action": [
                "codewhisperer:GenerateRecommendations"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Integrasi data Amazon Q AWS Glue tidak memiliki API yang tersedia melalui AWS SDK yang dapat Anda gunakan secara terprogram. Dua API berikut digunakan dalam kebijakan IAM untuk mengaktifkan pengalaman ini melalui panel obrolan Amazon Q atau notebook AWS Glue Studio: dan. `StartCompletion` `GetCompletion`

Menetapkan izin

Untuk memberikan akses, tambahkan izin ke pengguna, grup, atau peran Anda:

- Pengguna dan grup di Pusat AWS Identitas IAM: Buat set izin. Ikuti petunjuk di Buat set izin di Panduan Pengguna Pusat AWS Identitas IAM.
- Pengguna yang dikelola di IAM melalui penyedia identitas: Buat peran untuk federasi identitas. Ikuti petunjuk dalam Membuat peran untuk penyedia identitas pihak ketiga (federasi) dalam Panduan Pengguna IAM.
- Pengguna IAM:
  - Buat peran yang dapat diambil pengguna Anda. Ikuti petunjuk dalam Membuat peran untuk pengguna IAM dalam Panduan Pengguna IAM.
  - (Tidak disarankan) Pasang kebijakan langsung ke pengguna atau tambahkan pengguna ke grup pengguna. Ikuti petunjuk dalam Menambahkan izin ke pengguna (konsol) dalam Panduan Pengguna IAM.

Kemampuan pembuatan kode yang didukung

Berikut ini adalah kombinasi dari kemampuan pembuatan kode integrasi data Amazon Q di AWS Glue.

Sumber	Transformasi	Target
S3 dengan jenis format berikut: json, csv, parket, hudi, delta	ApplyMapping	S3 dengan jenis format berikut: json, csv, avro, orc, parket, hudi, delta
Katalog Data Glue	RenameField	Katalog Data Glue

Sumber	Transformasi	Target
Amazon Redshift	DropFields	Amazon Redshift
MySQL	SelectFields	MySQL
Postgres	DropNullFields	Postgres
Oracle	Filter	Oracle
SQL Server	Spigot	SQL Server
DynamoDB	Kode SQL Kustom	DynamoDB
Kepingan salju	Agregat	Kepingan salju
MongoDB	DropDuplicates	MongoDB
Konektor JDBC Kustom	Join	Konektor JDBC Kustom
Konektor Spark Kustom	Union	Konektor Spark Kustom
Google BigQuery		Google BigQuery
Teradata		Teradata
OpenSearch Layanan Amazon		OpenSearch Layanan Amazon
Vertica		Vertica
Azure SQL		Azure DQL
SAP HANA		SAP HANA
Kosmos Azure		Kosmos Azure

Contoh interaksi

Integrasi data Amazon Q AWS Glue memungkinkan Anda memasukkan pertanyaan Anda di panel Amazon Q. Anda dapat memasukkan pertanyaan tentang fungsionalitas integrasi data

yang disediakan oleh AWS Glue. Jawaban terperinci, bersama dengan dokumen referensi, akan dikembalikan.

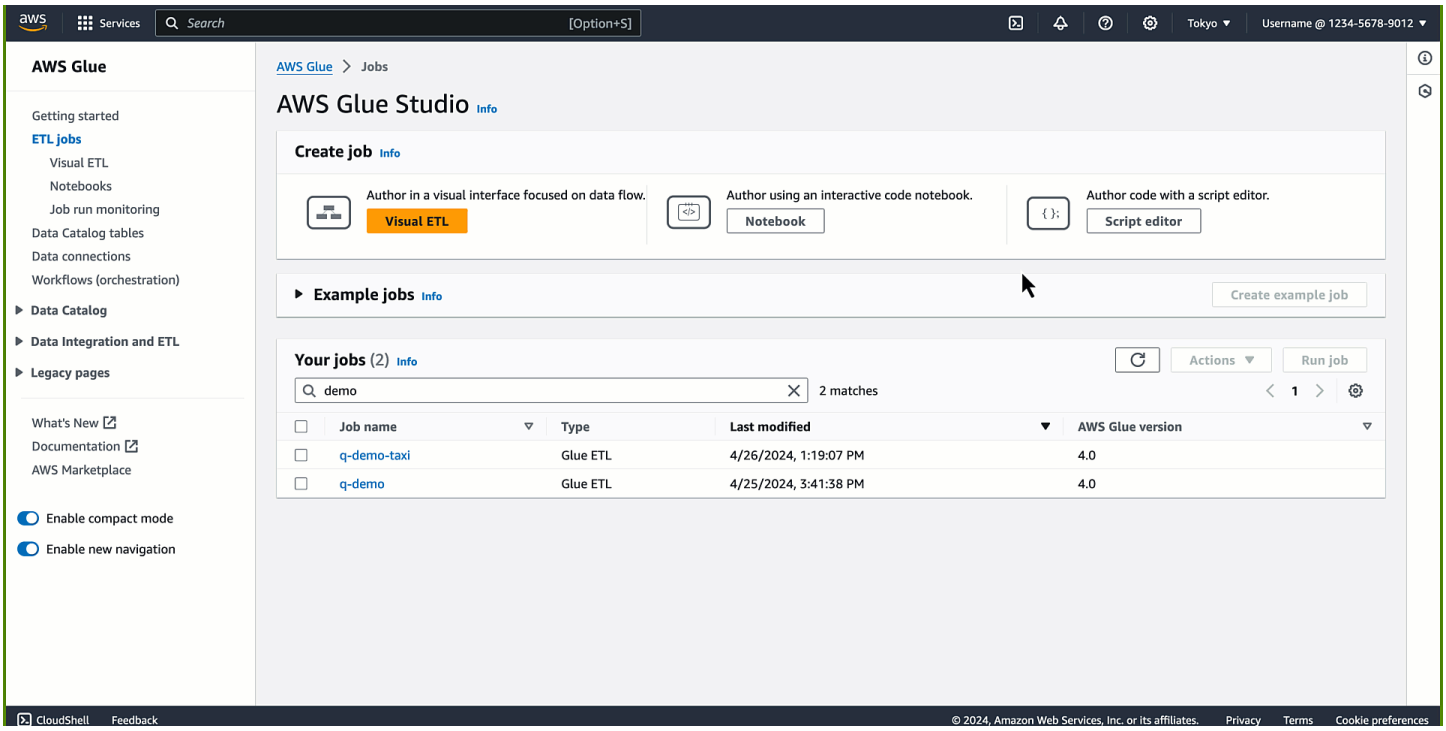
Kasus penggunaan lainnya adalah menghasilkan skrip pekerjaan AWS Glue ETL. Anda dapat mengajukan pertanyaan tentang cara melakukan ekstrak data, mengubah, memuat pekerjaan. PySpark Skrip yang dihasilkan akan dikembalikan.

Topik

- Interaksi obrolan Amazon Q
- AWS Glue Interaksi notebook studio

Interaksi obrolan Amazon Q

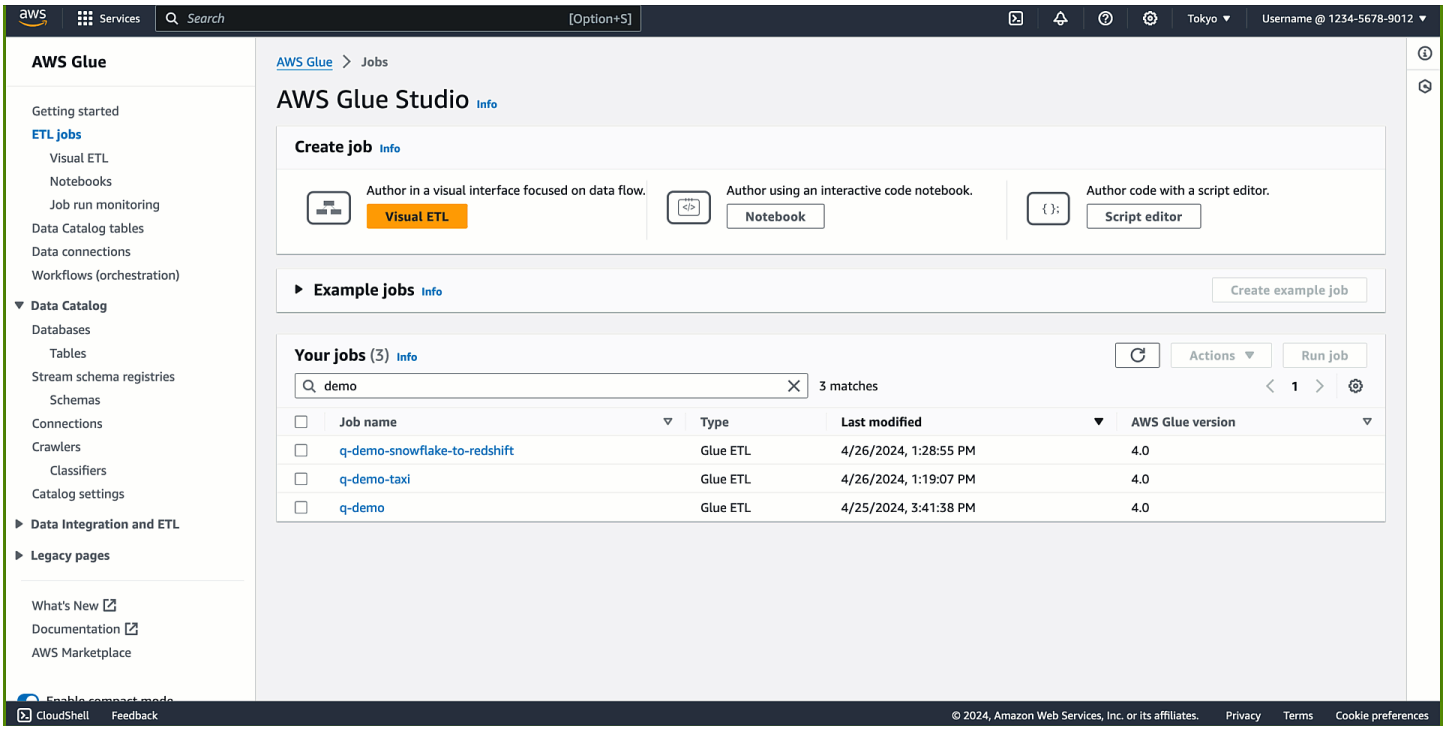
Di AWS Glue konsol, mulailah membuat pekerjaan baru, dan tanyakan Amazon Q: “Harap berikan skrip Glue yang berbunyi dari Snowflake, ganti nama bidang, dan tulis ke Redshift.”



Anda akan melihat bahwa kode dihasilkan. Dengan tanggapan ini, Anda dapat mempelajari dan memahami bagaimana Anda dapat membuat AWS Glue kode untuk tujuan Anda. Anda dapat menyalin/menempelkan kode yang dihasilkan ke editor skrip dan mengonfigurasi placeholder. Setelah Anda mengonfigurasi peran dan AWS Glue koneksi AWS Identity and Access Management (IAM) and Access Management (IAM) di tempat kerja, simpan dan jalankan pekerjaan. Saat

pekerjaan selesai, Anda dapat mulai menanyakan tabel yang diekspor dari Snowflake di Amazon Redshift.

Prompt berikut membaca data dari dua sumber yang berbeda, memfilter dan memproyeksikannya secara individual, bergabung pada kunci umum, dan menulis output ke target ketiga. Tanyakan Amazon Q: “Saya ingin membaca data dari S3 dalam format Parket, dan pilih beberapa bidang. Saya juga ingin membaca data dari DynamoDB, memilih beberapa bidang, dan memfilter beberapa baris. Saya ingin menyatukan dua kumpulan data ini dan menulis hasilnya. OpenSearch

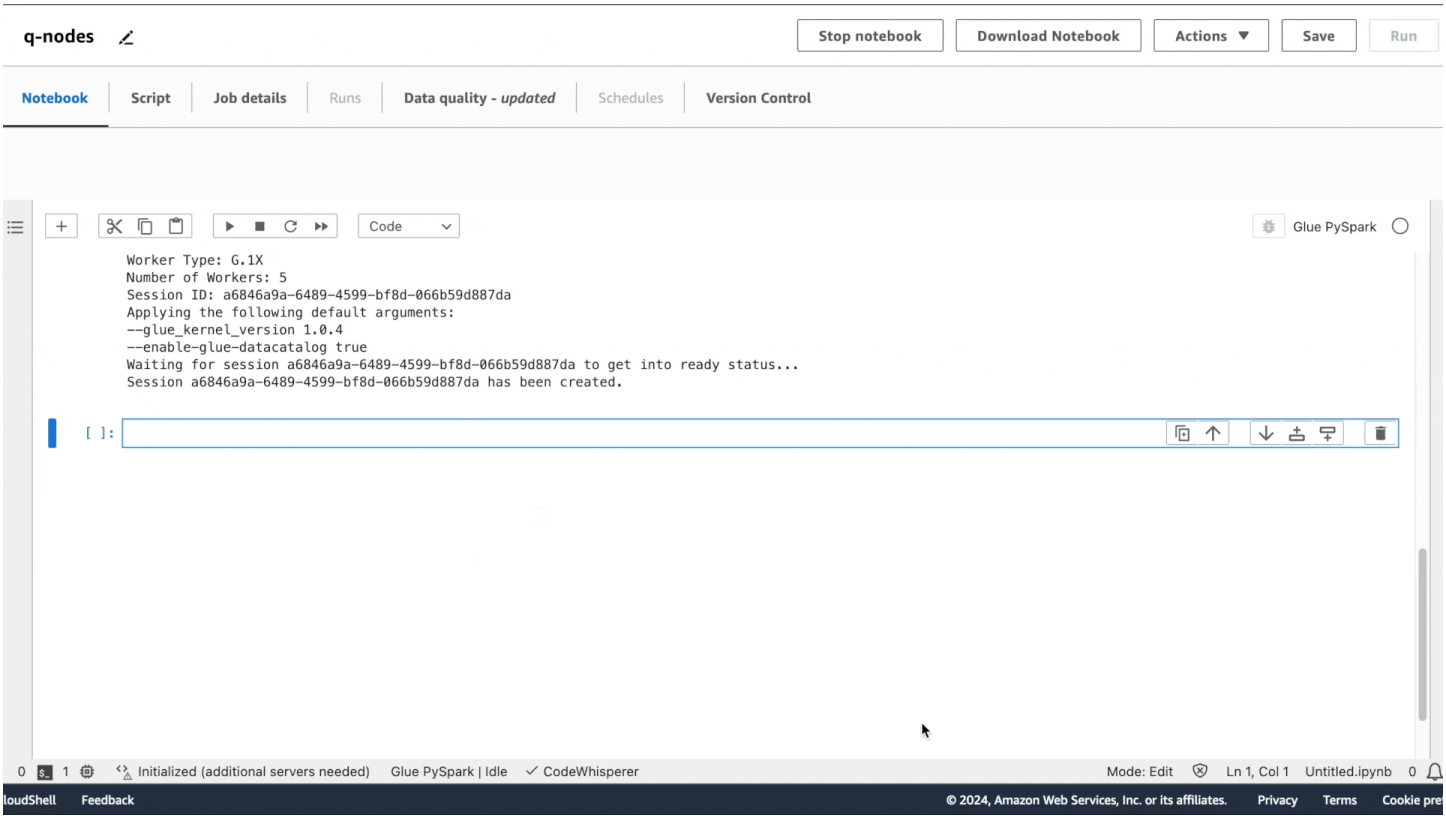


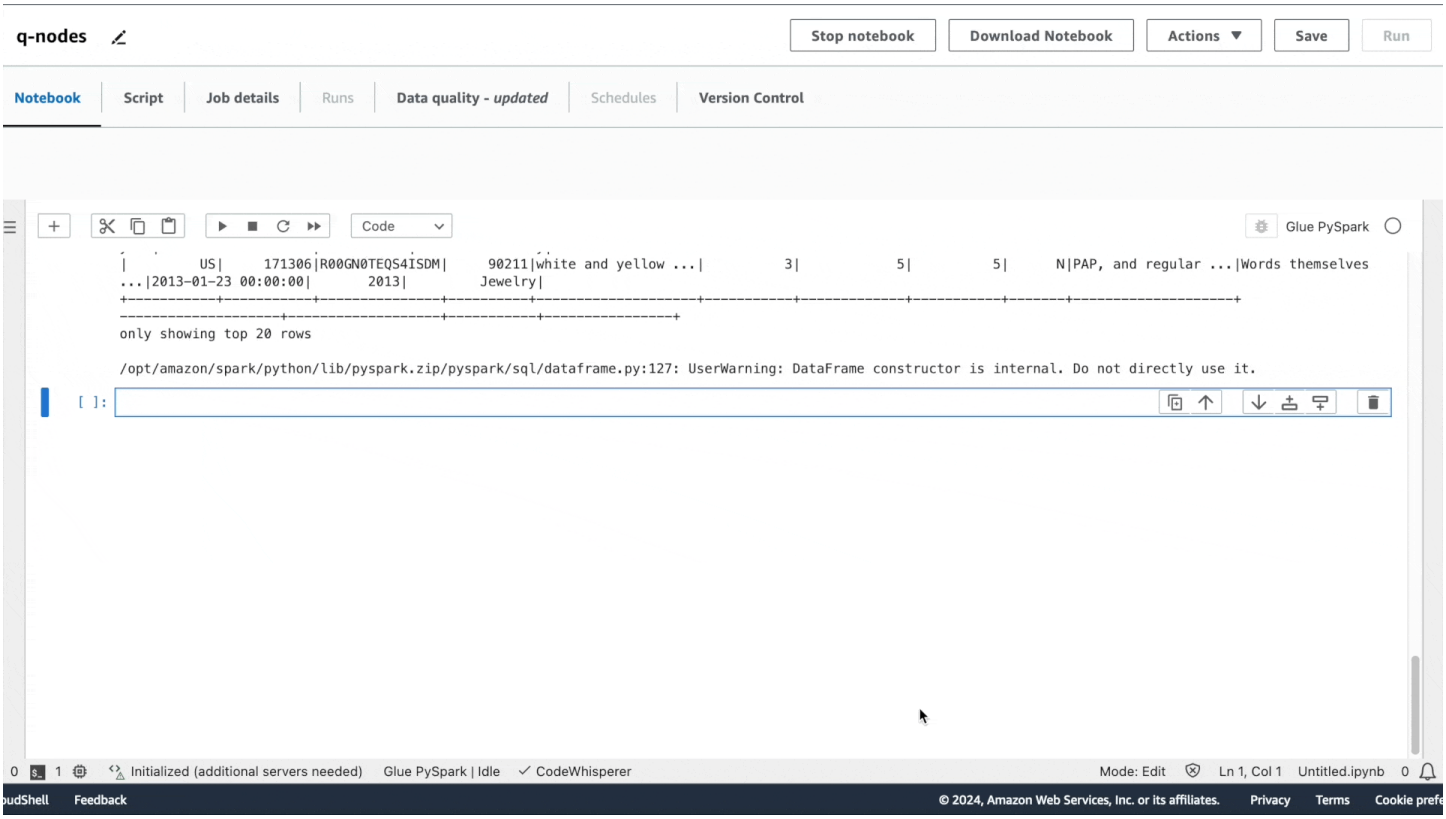
Kode dihasilkan. Ketika pekerjaan selesai, indeks Anda tersedia OpenSearch dan dapat digunakan oleh beban kerja hilir Anda.

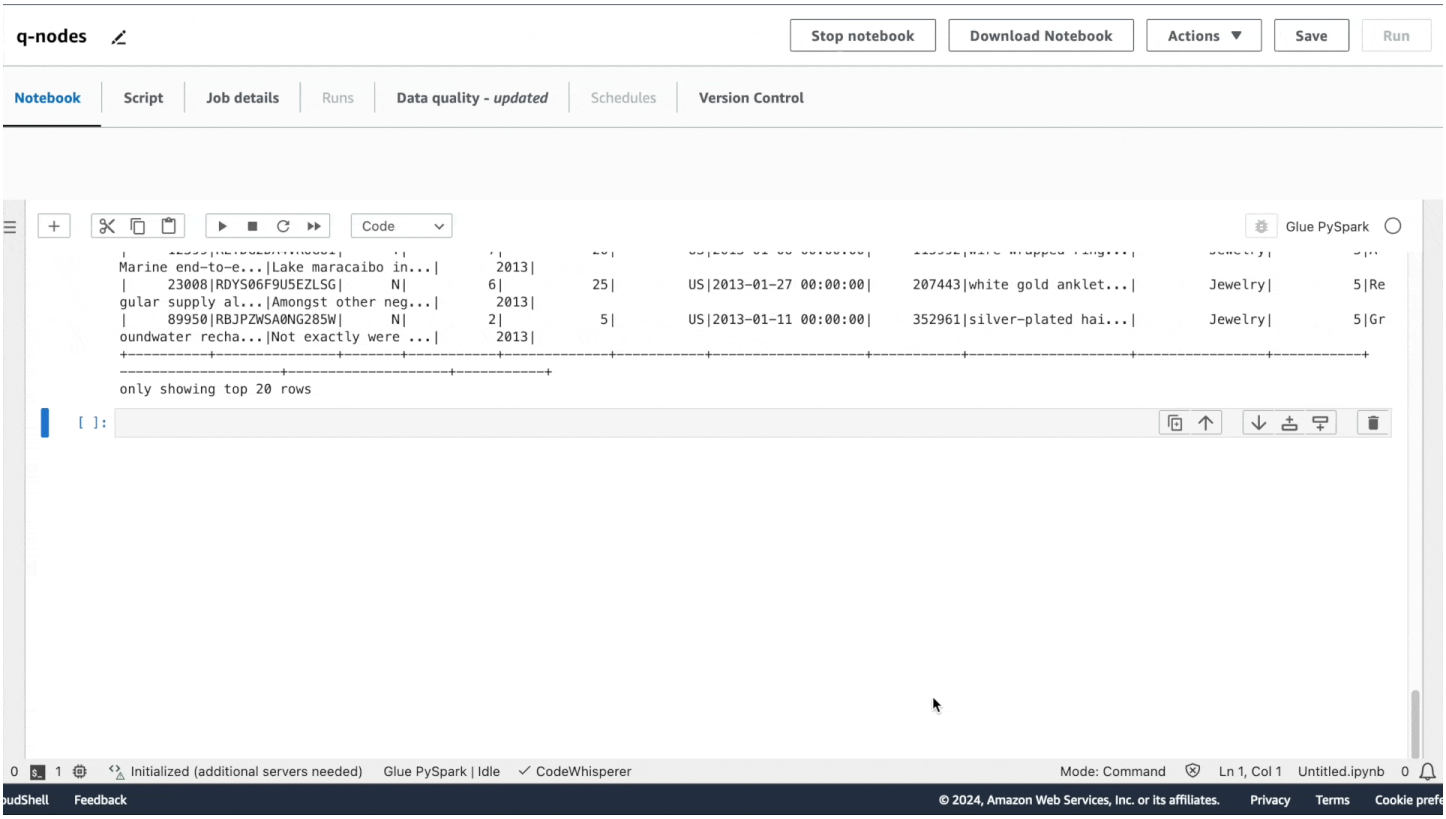
AWS Glue Interaksi notebook studio

Tambahkan sel baru dan masukkan komentar Anda untuk menjelaskan apa yang ingin Anda capai. Setelah Anda menekan Tab dan Enter, kode yang disarankan ditampilkan.

Maksud pertama adalah untuk mengekstrak data: “Beri saya kode yang membaca tabel Katalog Data Glue”, diikuti oleh “Beri saya kode untuk menerapkan transformasi filter dengan star_rating>3" dan “Beri saya kode yang menulis bingkai menjadi S3 sebagai Parket”.


q-nodes
Stop notebook
Download Notebook
Actions
Save
Run
Notebook
Script
Job details
Runs
Data quality - updated
Schedules
Version Control
Code
Glue PySpark
Worker Type: G.1X
Number of Workers: 5
Session ID: a6846a9a-6489-4599-bf8d-066b59d887da
Applying the following default arguments:
--glue_kernel_version 1.0.4
--enable-glue-datacatalog true
Waiting for session a6846a9a-6489-4599-bf8d-066b59d887da to get into ready status...
Session a6846a9a-6489-4599-bf8d-066b59d887da has been created.
[ ]:
Initialized (additional servers needed)
Glue PySpark | Idle
CodeWhisperer
Mode: Edit
Ln 1, Col 1
Untitled.ipynb
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms


q-nodes
Stop notebook
Download Notebook
Actions
Save
Run
Notebook
Script
Job details
Runs
Data quality - updated
Schedules
Version Control
Code
Glue PySpark
| US| 171306|R00GN0TEQS4ISDM| 90211|white and yellow ...| 3| 5| 5| N|PAP, and regular ...|Words themselves
...|2013-01-23 00:00:00| 2013| Jewelry|
only showing top 20 rows
/opt/amazon/spark/python/lib/pyspark.zip/pyspark/sql/dataframe.py:127: UserWarning: DataFrame constructor is internal. Do not directly use it.
[ ]:
Initialized (additional servers needed)
Glue PySpark | Idle
CodeWhisperer
Mode: Edit
Ln 1, Col 1
Untitled.ipynb
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms

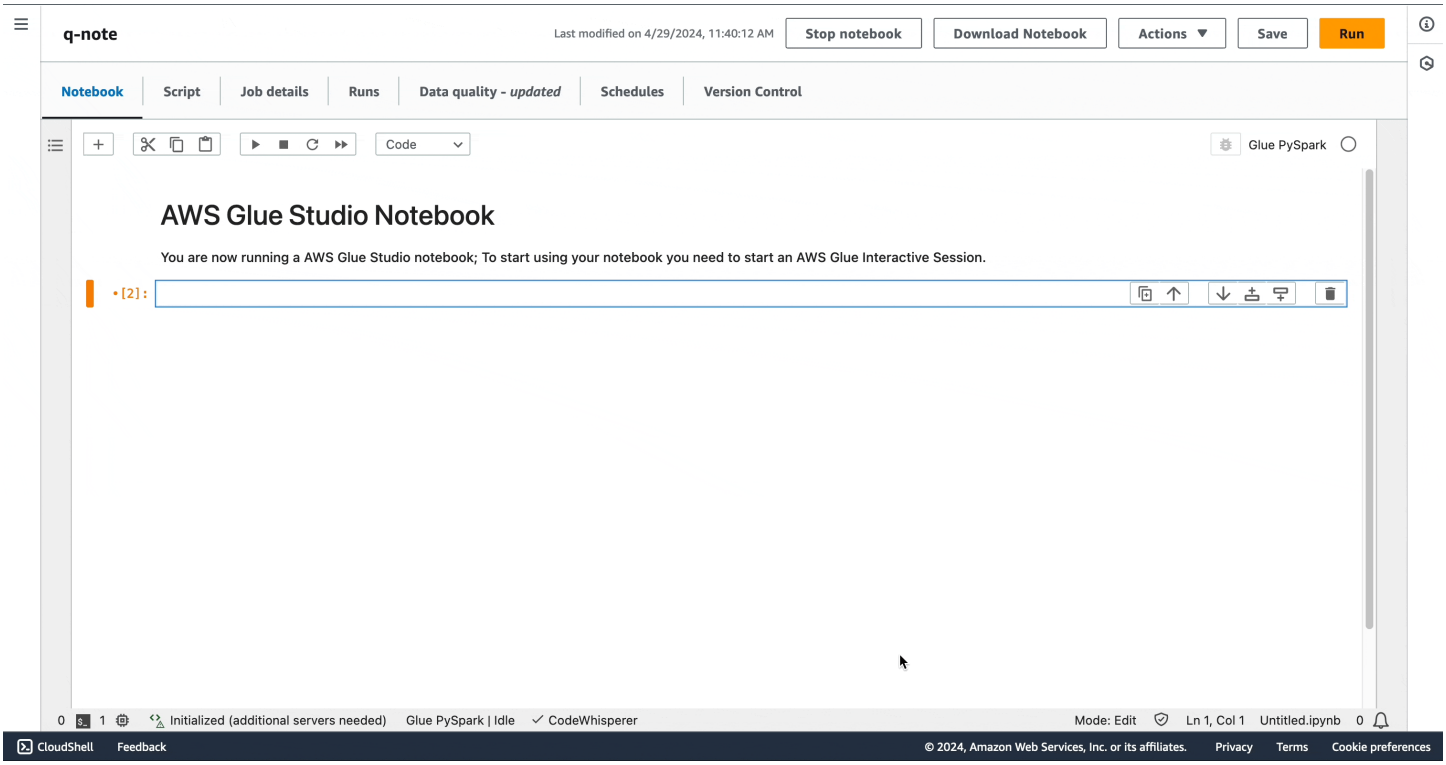


Mirip dengan pengalaman obrolan Amazon Q, kode ini direkomendasikan. Jika Anda menekan Tab, maka kode yang disarankan dipilih.

Anda dapat menjalankan setiap sel dengan mengisi opsi yang sesuai untuk sumber Anda dalam kode yang dihasilkan. Kapan pun dalam proses, Anda juga dapat melihat pratinjau sampel kumpulan data Anda dengan menggunakan `show()` metode ini.

Petunjuk kompleks

Anda dapat menghasilkan skrip lengkap dengan satu prompt kompleks. “Saya memiliki data JSON di S3 dan data di Oracle yang perlu digabungkan. Harap berikan skrip Glue yang membaca dari kedua sumber, bergabung, dan kemudian menulis hasilnya ke Redshift.”



Anda mungkin memperhatikan bahwa, di buku catatan, integrasi data Amazon Q AWS Glue menghasilkan cuplikan kode yang sama yang dihasilkan di obrolan Amazon Q.

Anda dapat menjalankan notebook sebagai pekerjaan, baik dengan memilih Jalankan atau secara terprogram.

Orkestrasi di AWS Glue

Bagian berikut memberikan informasi tentang orkestrasi pekerjaan di. AWS Glue

Topik

Memulai pekerjaan dan crawler menggunakan pemicu

Di AWS Glue, Anda dapat membuat objek Katalog Data yang disebut pemicu, yang dapat Anda gunakan untuk memulai secara manual atau secara otomatis satu atau beberapa crawler atau tugas extract, transform, and load (ETL). Dengan menggunakan pemicu, Anda dapat merancang sebuah rantai tugas dan crawler yang dependen.

> **Note**
>
> Anda dapat melakukan hal yang sama dengan menentukan alur kerja. Alur kerja lebih disukai untuk menciptakan operasi ETL multi-tugas yang kompleks. Untuk informasi selengkapnya, lihat the section called “Melakukan aktivitas ETL yang kompleks menggunakan cetak biru dan alur kerja”.

Topik

AWS Gluepemicu

Saat diaktifkan, pemicu dapat memulai tugas dan crawler yang ditentukan. Sebuah pemicu dapat aktif sesuai permintaan, berdasarkan jadwal, atau berdasarkan kombinasi peristiwa.

> **Note**
>
> Hanya dua crawler yang dapat diaktifkan oleh satu pemicu tunggal. Jika Anda ingin melakukan crawling pada beberapa penyimpanan data, maka Anda harus menggunakan beberapa sumber untuk masing-masing crawler, alih-alih menjalankan beberapa crawler secara bersamaan.

Sebuah pemicu bisa berada dalam salah satu dari beberapa status berikut. Sebuah pemicu berada dalam status `CREATED`, `ACTIVATED`, atau `DEACTIVATED`. Ada juga status transisi, seperti `ACTIVATING`. Untuk menghentikan pemicu agar tidak aktif untuk sementara waktu, Anda dapat menonaktifkannya. Anda kemudian dapat mengaktifkannya kembali nanti.

Ada tiga jenis pemicu:

Terjadwal

Sebuah pemicu berbasis waktu berdasarkan pada `cron`.

Anda dapat membuat pemicu untuk serangkaian tugas atau crawler berdasarkan jadwal. Anda dapat menentukan batasan-batasan, seperti frekuensi yang dijalankan oleh tugas atau crawler, pada hari apa dalam seminggu pemicu itu berjalan, dan kapan berjalan. Batasan-batasan tersebut ini didasarkan pada `cron`. Ketika Anda menyiapkan jadwal untuk memicu, Anda harus mempertimbangkan fitur dan keterbatasan cron. Misalnya, jika Anda memilih untuk menjalankan crawler pada hari ke 31 setiap bulan, ingatlah bahwa ada bulan yang tidak terdiri dari 31 hari. Untuk informasi selengkapnya tentang cron, lihat Jadwal berbasis waktu untuk pekerjaan dan crawler.

Bersyarat

Pemicu yang aktif saat tugas atau crawler atau beberapa tugas atau crawler sebelumnya memenuhi daftar syarat.

Saat membuat sebuah pemicu bersyarat, Anda menentukan daftar tugas dan daftar crawler yang akan diawasi. Untuk setiap tugas atau crawler yang diawasi, tetapkan status yang akan diawasi, seperti berhasil, gagal, habis waktu, dan seterusnya. Pemicu aktif jika tugas yang atau crawler diawasi berakhir dengan status yang ditentukan. Anda dapat mengkonfigurasi pemicu untuk aktif ketika salah satu atau semua peristiwa yang diawasi terjadi.

Misalnya, Anda bisa mengkonfigurasi T1 pemicu untuk memulai tugas J3 ketika kedua tugas J1 dan tugas J2 berhasil diselesaikan, dan pemicu T2 untuk memulai tugas J4 jika tugas J1 atau tugas J2 gagal.

Tabel berikut mencantumkan status penyelesaian tugas dan crawler (peristiwa) yang diawasi pemicu.

Status penyelesaian tugas	Status penyelesaian crawler
• `SUCCEEDED` • `STOPPED` • `FAILED` • `TIMEOUT`	• `SUCCEEDED` • `FAILED` • `CANCELLED`

Sesuai permintaan

Pemicu yang aktif saat anda mengaktifkannya. Pemicu sesuai permintaan tidak pernah ada dalam status `ACTIVATED` atau `DEACTIVATED`. Mereka selalu ada dalam status `CREATED`.

Sehingga mereka siap untuk langsung aktif setelah mereka ada, Anda dapat mengatur bendera untuk mengaktifkan pemicu terjadwal dan bersyarat saat Anda membuatnya.

> **Important**
>
> Tugas atau crawler yang berjalan sebagai hasil dari tugas lain atau penyelesaian crawler disebut sebagai dependen. Tugas dependen atau crawler dependen hanya dimulai jika tugas atau crawler yang selesai dimulai oleh pemicu. Semua tugas atau crawler dalam rantai dependensi harus berupa keturunan dari satu pemicu terjadwal atau pemicu sesuai permintaan.

Melewati parameter pekerjaan dengan pemicu

Pemicu dapat memberikan parameter untuk tugas yang dimulai. Parameter termasuk argumen tugas, nilai habis waktu, konfigurasi keamanan, dan banyak lagi. Jika pemicu memulai beberapa tugas, maka parameter tersebut diberikan ke setiap tugas.

Berikut ini adalah aturan untuk argumen tugas yang diberikan oleh pemicu:

- Jika kunci dalam pasangan nilai kunci cocok dengan argumen tugas default, maka argumen yang diberikan menimpa argumen default. Jika kunci tidak cocok dengan argumen default, maka argumen tersebut akan diberikan sebagai argumen tambahan untuk tugas tersebut.
- Jika kunci dalam pasangan nilai kunci cocok argumen yang tidak dapat ditimpa, maka argumen yang diberikan akan diabaikan.

Untuk informasi selengkapnya, lihat the section called “Pemicu” di AWS Glue API.

Menambahkan pemicu

Anda dapat menambahkan pemicu menggunakan konsol AWS Glue, AWS Command Line Interface (AWS CLI), atau API AWS Glue.

> **Note**
>
> Saat ini, konsol AWS Glue hanya mendukung tugas, bukan crawler, ketika bekerja dengan pemicu. Anda dapat menggunakan API AWS CLI atau AWS Glue untuk mengkonfigurasi pemicu dengan tugas dan crawler.

Untuk menambahkan pemicu (konsol)

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada ETL pilih Pemicu. Lalu pilih Tambahkan pemicu.
3. Berikan properti berikut:

   Nama

   Berikan pemicu Anda nama yang unik.

   Jenis pemicu

   Tentukan satu dari yang berikut ini:

   - Jadwal: Pemicu aktif pada frekuensi dan waktu tertentu.
   - Peristiwa tugas: Pemicu bersyarat. Pemicu aktif ketika salah satu atau semua tugas dalam daftar cocok dengan statusnya yang ditentukan. Agar pemicu aktif, tugas yang diawasi

harus dimulai oleh pemicu. Untuk tugas apa pun yang Anda pilih, Anda hanya dapat mengawasi satu peristiwa tugas (status penyelesaian).

- Sesuai permintaan: Pemicu aktif saat diaktifkan.

4. Selesaikan penuntun pemicu. Di halaman Tinjau, Anda dapat langsung mengaktifkan pemicu (bersyarat) Jadwal dan Peristiwa tugas dengan memilih Aktifkan pemicu pada saat dibuat.

Untuk menambahkan pemicu (AWS CLI)

- Masukkan perintah yang serupa dengan yang berikut ini.

```
aws glue create-trigger --name MyTrigger --type SCHEDULED --schedule  "cron(0 12 *
 * ? *)" --actions CrawlerName=MyCrawler --start-on-creation
```

Perintah ini membuat pemicu jadwal bernama `MyTrigger`, yang berjalan setiap hari pada pukul 12:00 siang UTC dan memulai crawler bernama `MyCrawler`. Pemicu dibuat dalam status diaktifkan.

Untuk informasi selengkapnya, lihat the section called “AWS Gluepemicu”.

Jadwal berbasis waktu untuk pekerjaan dan crawler

Anda dapat menentukan jadwal berbasis waktu untuk crawler dan tugas di AWS Glue. Penentuan jadwal ini menggunakan sintaksis cron yang mirip UNIX. Anda menentukan waktu dalam Waktu Universal Terkoordinasi (UTC), dan presisi minimum untuk jadwal adalah 5 menit.

Untuk mempelajari lebih lanjut tentang cara mengonfigurasi tugas dan crawler agar dijalankan menggunakan jadwal, lihat Memulai pekerjaan dan crawler menggunakan pemicu.

Ekspresi Cron

Ekspresi cron memiliki enam bidang yang diperlukan, yang dipisahkan oleh spasi putih.

Sintaks

```
cron(Minutes Hours Day-of-month Month Day-of-week Year)
```

Bidang	Nilai	Wildcard
Menit	0–59	, - * /
Jam	0–23	, - * /
D ay-of-month	1–31	, - * ? / L W
Bulan	1–12 atau JAN-DEC	, - * /
D ay-of-week	1–7 atau SUN-SAT	, - * ? / L
Tahun	1970–2199	, - * /

Wildcard

- Wildcard , (koma) mencakup nilai tambahan. Di kolom `Month`, `JAN,FEB,MAR` akan mencakup Januari, Februari, dan Maret.
- Wildcard - (tanda hubung) menentukan rentang. Di kolom `Day`, 1–15 akan mencakup tanggal 1 hingga 15 pada bulan yang ditentukan.
- Wildcard * (bintang) mencakup semua nilai di bidang. Di kolom `Hours`, * akan mencakup setiap jam.
- Wildcard / (garis miring) menentukan tambahan. Di kolom `Minutes`, Anda bisa memasukkan **`1/10`** untuk menentukan setiap menit ke-10, mulai dari menit pertama jam (sebagai contoh, menit ke-11, ke-21, dan ke-31, dan seterusnya).
- Wildcard ? (tanda tanya) menentukan satu atau yang lain. Di **`Day-of-month`** lapangan Anda bisa masuk 7, dan jika Anda tidak peduli hari apa dalam minggu ketujuh, Anda bisa masuk? di ay-of-week bidang D.
- Wildcard L di kolom `Day-of-month` atau `Day-of-week` menentukan hari terakhir pada bulan atau minggu.
- Wildcard W di kolom `Day-of-month` menentukan hari kerja. Di kolom `Day-of-month`, `3W` menentukan hari kerja yang paling dekat dengan pekan ketiga di bulan itu.

Batas

- Anda tidak dapat menentukan kolom `Day-of-month` dan `Day-of-week` dalam ekspresi cron yang sama. Jika Anda menentukan sebuah nilai di salah satu kolom, maka Anda harus menggunakan ? (tanda tanya) di kolom yang lain.
- Ekspresi cron yang mengarahkan ke rate lebih cepat dari 5 menit tidak didukung.

Contoh

Anda dapat membuat jadwal, Anda dapat menggunakan contoh cron berikut.

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
0	10	*	*	?	*	Jalankan pada pukul 10:00 pagi (UTC) setiap hari
15	12	*	*	?	*	Jalankan pada pukul 12.15 (UTC) setiap hari
0	18	?	*	MON-FRI	*	Jalankan pada pukul 18.00 (UTC) setiap Senin hingga Jumat
0	8	1	*	?	*	Jalankan pada pukul 8:00 (UTC)

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
						setiap tanggal satu di bulannya
0/15	*	*	*	?	*	Jalankan setiap 15 menit
0/10	*	?	*	MON-FRI	*	Jalankan setiap 10 menit Senin hingga Jumat
0/5	8–17	?	*	MON-FRI	*	Jalankan setiap 5 menit Senin hingga Jumat antara pukul 08.00 dan 17.55 (UTC)

Sebagai contoh, untuk berjalan pada jadwal setiap hari pada 12:15 UTC, tentukan:

```
cron(15 12 * * ? *)
```

Mengaktifkan dan menonaktifkan pemicu

Anda dapat mengaktifkan atau menonaktifkan pemicu menggunakan AWS Glue konsol, AWS Command Line Interface (AWS CLI), atau AWS Glue API.

Untuk mengaktifkan atau menonaktifkan sebuah pemicu (konsol)

1. Masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada ETL pilih Pemicu.
3. Pilih kotak centang yang ada di samping pemicu yang diinginkan, dan pada menu Tindakan, pilih Aktifkan pemicu untuk mengaktifkan pemicu atau pilih Menonaktifkan pemicu untuk menonaktifkan pemicu.

Untuk mengaktifkan atau menonaktifkan pemicu (AWS CLI)

- Gunakan salah satu perintah berikut ini.

```
aws glue start-trigger --name MyTrigger

aws glue stop-trigger --name MyTrigger
```

  Memulai pemicu akan mengaktifkannya, dan menghentikan pemicu akan menonaktifkannya. Saat Anda mengaktifkan pemicu sesuai permintaan, maka pemicu akan segera aktif.

Untuk informasi selengkapnya, lihat the section called “AWS Gluepemicu”.

Melakukan aktivitas ETL yang kompleks menggunakan cetak biru dan alur kerja di AWS Glue

Beberapa proses ekstrak, transformasi, dan pemuatan (ETL) kompleks organisasi Anda mungkin paling baik diimplementasikan dengan menggunakan beberapa AWS Glue pekerjaan dan crawler yang bergantung. Dengan menggunakan AWS Glue alur kerja, Anda dapat merancang proses ETL multi-job dan multi-crawler yang kompleks yang AWS Glue dapat dijalankan dan dilacak sebagai entitas tunggal. Setelah Anda membuat alur kerja dan menentukan tugas, crawler, dan pemicu dalam alur kerja, Anda dapat menjalankan alur kerja sesuai permintaan atau jadwal.

Topik

Ikhtisar alur kerja di AWS Glue

Di AWS Glue, Anda dapat menggunakan alur kerja untuk membuat dan memvisualisasikan kegiatan extract, transform, and load (ETL) kompleks yang melibatkan beberapa crawler, tugas, dan pemicu. Setiap alur kerja mengelola eksekusi dan pemantauan semua tugas dan crawlernya. Saat alur kerja menjalankan setiap komponen, ia mencatat kemajuan eksekusi dan status. Hal ini memberi Anda gambaran umum tentang tugas yang lebih besar dan detail dari setiap langkah. Konsol AWS Glue menyediakan sebuah representasi visual dari sebuah alur kerja dalam bentuk grafik.

Anda dapat membuat alur kerja dari cetak biru AWS Glue, atau Anda dapat secara manual membangun alur kerja komponen pada satu waktu dengan menggunakan AWS Management Console atau AWS Glue API. Untuk informasi selengkapnya tentang cetak biru, lihat the section called “Ikhtisar cetak biru”.

Pemicu dalam alur kerja dapat memulai tugas dan crawler dan dapat diaktifkan saat tugas atau crawler selesai. Dengan menggunakan pemicu, Anda dapat membuat rantai besar tugas dan crawler yang saling bergantung. Selain pemicu dalam sebuah alur kerja yang menentukan dependensi tugas dan crawler, setiap alur kerja memiliki pemicu awal. Ada tiga jenis pemicu awal:

- Terjadwal — Alur kerja dimulai sesuai jadwal yang Anda tetapkan. Jadwal dapat berupa jadwal harian, mingguan, bulanan, dan sebagainya, atau dapat menjadi jadwal kustom berdasarkan ekspresi `cron`.
- Sesuai permintaan — Alur kerja dimulai secara manual dari konsol AWS Glue, API, atau AWS CLI.
- EventBridge event — Alur kerja dimulai pada saat terjadinya satu EventBridge peristiwa Amazon atau sekumpulan EventBridge peristiwa Amazon. Dengan tipe pemicu ini, AWS Glue dapat menjadi konsumen peristiwa dalam arsitektur didorong-peristiwa. Setiap jenis EventBridge acara dapat memulai alur kerja. Kasus penggunaan umum adalah tibanya objek baru dalam bucket Amazon S3 (operasi `PutObject` S3).

  Memulai sebuah alur kerja dengan batch peristiwa berarti menunggu sampai sejumlah peristiwa tertentu telah diterima atau sampai jumlah waktu tertentu telah berlalu. Saat Anda membuat pemicu EventBridge peristiwa, Anda dapat menentukan kondisi batch secara opsional. Jika Anda menentukan syarat batch, maka Anda harus menentukan ukuran batch (jumlah peristiwa), dan secara opsional dapat menentukan jendela batch (jumlah detik). Jendela batch default dan maksimumnya adalah 900 detik (15 menit). Syarat batch yang terpenuhi pertama kali akan memulai alur kerja. Jendela batch dimulai ketika peristiwa pertama datang. Jika Anda tidak menentukan syarat batch saat membuat sebuah pemicu, maka ukuran batch default-nya adalah 1.

  Ketika alur kerja tersebut dimulai, syarat batch akan reset dan pemicu peristiwa mulai mengawasi syarat batch berikutnya yang harus dipenuhi untuk memulai alur kerja lagi.

  Tabel berikut menunjukkan bagaimana ukuran batch dan jendela batch beroperasi bersama-sama untuk memicu sebuah alur kerja.

Ukuran batch	Jendela batch	Syarat pemicu yang dihasilkan
10		Alur kerja dipicu pada saat kedatangan 10 EventBrid ge peristiwa, atau 15 menit setelah kedatangan acara pertama, mana yang terjadi lebih dulu. (Jika ukuran jendela tidak ditentukan, maka ukuran default-nya adalah 15 menit.)
10	2 menit	Alur kerja dipicu pada saat kedatangan 10 EventBrid ge peristiwa, atau 2 menit setelah kedatangan acara pertama, mana yang terjadi lebih dulu.

Ukuran batch	Jendela batch	Syarat pemicu yang dihasilkan
1		Alur kerja dipicu pada saat datangnya peristiwa pertama. Ukuran jendela tidak relevan. Ukuran batch default ke 1 jika Anda tidak menentukan kondisi batch saat Anda membuat pemicu peristiwa. EventBridge

Operasi API `GetWorkflowRun` mengembalikan syarat batch yang memicu alur kerja.

Terlepas dari bagaimana alur kerja dimulai, Anda dapat menentukan jumlah maksimum eksekusi alur kerja yang bersamaan saat Anda membuat alur kerja.

Jika peristiwa atau batch peristiwa mulai menjalankan sebuah alur kerja yang pada akhirnya gagal, maka peristiwa atau batch peristiwa tidak lagi dianggap untuk memulai eksekusi alur kerja. Eksekusi alur kerja baru akan dimulai hanya ketika peristiwa atau batch peristiwa berikutnya datang.

> **Important**
>
> Batasi jumlah total pekerjaan, crawler, dan pemicu dalam alur kerja hingga 100 atau kurang. Jika Anda menyertakan lebih dari 100, Anda mungkin mendapatkan kesalahan saat mencoba melanjutkan atau menghentikan alur kerja berjalan.

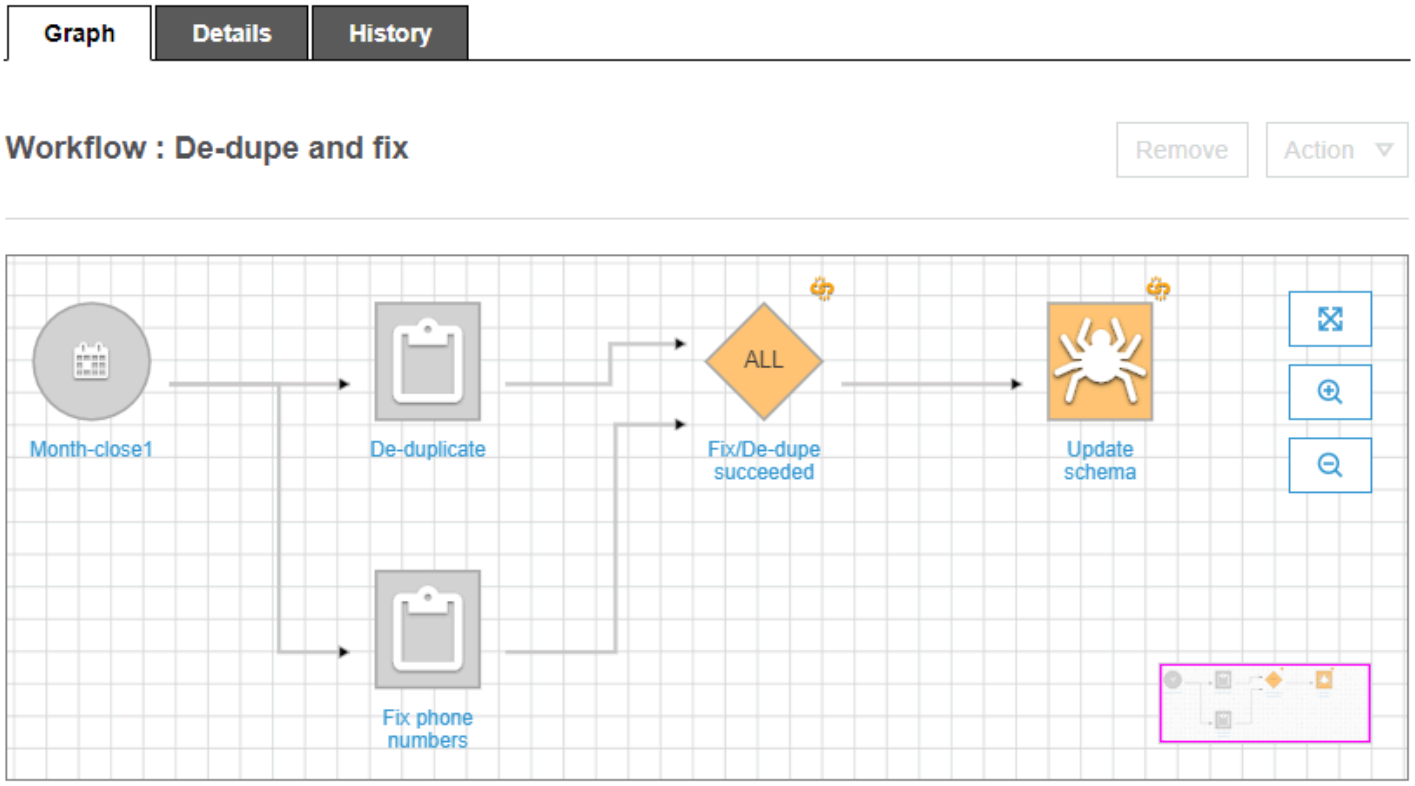
Eksekusi alur kerja tidak akan dimulai jika ia akan melebihi batas jumlah eksekusi alur kerja bersamaan yang ditetapkan untuk alur kerja, meskipun syarat peristiwa terpenuhi. Dianjurkan untuk menyesuaikan batas jumlah eksekusi alur kerja bersamaan berdasarkan volume peristiwa yang diharapkan. AWS Glue tidak akan mencoba lagi eksekusi alur kerja yang gagal karena melampaui batas jumlah eksekusi alur kerja bersamaan. Demikian juga, disarankan untuk menyesuaikan batas jumlah eksekusi alur kerja bersamaan untuk tugas dan crawler dalam alur kerja berdasarkan volume peristiwa yang diharapkan.

Properti jalankan alur kerja

Untuk berbagi dan mengelola status di seluruh alur kerja, Anda dapat menentukan properti eksekusi alur kerja default. Properti ini, yang pasangan nama/nilainya, tersedia untuk semua tugas di alur kerja. Dengan menggunakan AWS Glue API, tugas dapat mengambil properti eksekusi alur kerja dan mengubahnya untuk tugas yang datang kemudian dalam alur kerja tersebut.

Grafik alur kerja

Gambar berikut menampilkan grafik alur kerja yang sangat mendasar pada konsol AWS Glue. Alur kerja Anda bisa memiliki belasan komponen.



Alur kerja ini dimulai oleh pemicu terjadwal, `Month-close1`, yang memulai dua tugas, `De-duplicate` dan `Fix phone numbers`. Setelah berhasil menyelesaikan kedua tugas, pemicu peristiwa, `Fix/De-dupe succeeded`, memulai crawler, `Update schema`.

Tampilan alur kerja statis dan dinamis

Untuk setiap alur kerja, ada gagasan tampilan statis dan tampilan dinamis. Tampilan statis menunjukkan desain dari alur kerja. Tampilan dinamis adalah tampilan waktu aktif yang mencakup informasi eksekusi terbaru untuk masing-masing tugas dan crawler. Informasi yang dijalankan mencakup detail status sukses dan kesalahan.

Ketika sebuah alur kerja berjalan, konsol menampilkan tampilan dinamis, yang secara grafis menunjukkan tugas yang telah selesai dan yang belum dijalankan. Anda juga dapat mengambil tampilan dinamis dari sebuah alur kerja yang berjalan menggunakan AWS Glue API. Untuk informasi selengkapnya, lihat Menanyakan alur kerja menggunakan AWS Glue API.

Lihat juga

- the section called “Membuat alur kerja dari cetak biru”
- the section called “Membuat dan membangun alur kerja secara manual”
- Alur Kerja (untuk API alur kerja)

Membuat dan membangun alur kerja secara manual di AWS Glue

Anda dapat menggunakan konsol AWS Glue untuk membuat dan membangun alur kerja satu simpul secara manual pada satu waktu.

Alur kerja berisi tugas, crawler, dan pemicu. Sebelum membuat alur kerja secara manual, buat tugas dan crawler yang harus disertakan alur kerja. Yang terbaik adalah menentukan run-on-demand crawler untuk alur kerja. Anda dapat membuat pemicu baru saat sedang membangun alur kerja Anda, atau Anda dapat kloning pemicu yang ada ke dalam alur kerja tersebut. Ketika Anda melakukan kloning pemicu, semua objek katalog yang dikaitkan dengan pemicu—tugas atau crawler yang mengaktifkannya dan tugas atau crawler yang dimulai—ditambahkan ke alur kerja tersebut.

Important

Batasi jumlah total pekerjaan, crawler, dan pemicu dalam alur kerja hingga 100 atau kurang. Jika Anda menyertakan lebih dari 100, Anda mungkin mendapatkan kesalahan saat mencoba melanjutkan atau menghentikan alur kerja berjalan.

Anda membangun alur kerja dengan menambahkan pemicu ke grafik alur kerja, dan menentukan peristiwa dan tindakan yang diawasi untuk setiap pemicu. Anda mulai dengan pemicu mulai, yang dapat berupa pemicu sesuai permintaan atau terjadwal, dan selesaikan grafik dengan menambahkan pemicu peristiwa (bersyarat).

Langkah 1: Buat alur kerja

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada ETL, pilih Alur kerja.
3. Pilih Tambahkan alur kerja dan isi lengkap formulir Tambahkan alur kerja ETL baru.

Setiap properti eksekusi default opsional yang Anda tambahkan disediakan sebagai argumen untuk semua tugas di alur kerja. Untuk informasi selengkapnya, lihat Mendapatkan dan mengatur alur kerja menjalankan properti di AWS Glue.

4. Pilih Tambahkan alur kerja.

   Alur kerja baru muncul dalam daftar di halaman alur kerja.

Langkah 2: Tambahkan pemicu awal

1. Pada halaman alur kerja, pilih alur kerja baru Anda. Kemudian, di bagian bawah halaman, pastikan bahwa tab Grafik sudah dipilih.
2. Pilih Tambahkan pemicu, dan di kotak dialog Tambahkan pemicu, lakukan salah satu hal berikut:

   - Pilih Kloning yang ada, dan pilih pemicu yang akan dikloning. Kemudian pilih Add.

     Pemicu muncul pada grafik, bersama dengan tugas dan crawler yang diawasi dan tugas dan crawler yang dimulai.

     Jika Anda keliru memilih pemicu yang salah, pilih pemicu pada grafik, dan kemudian pilih Hapus.

   - Pilih Tambah baru, dan isi lengkap Tambahkan pemicu.

     1. Untuk jenis Pemicu, pilih Jadwal, Sesuai permintaan, atau EventBridgeacara.

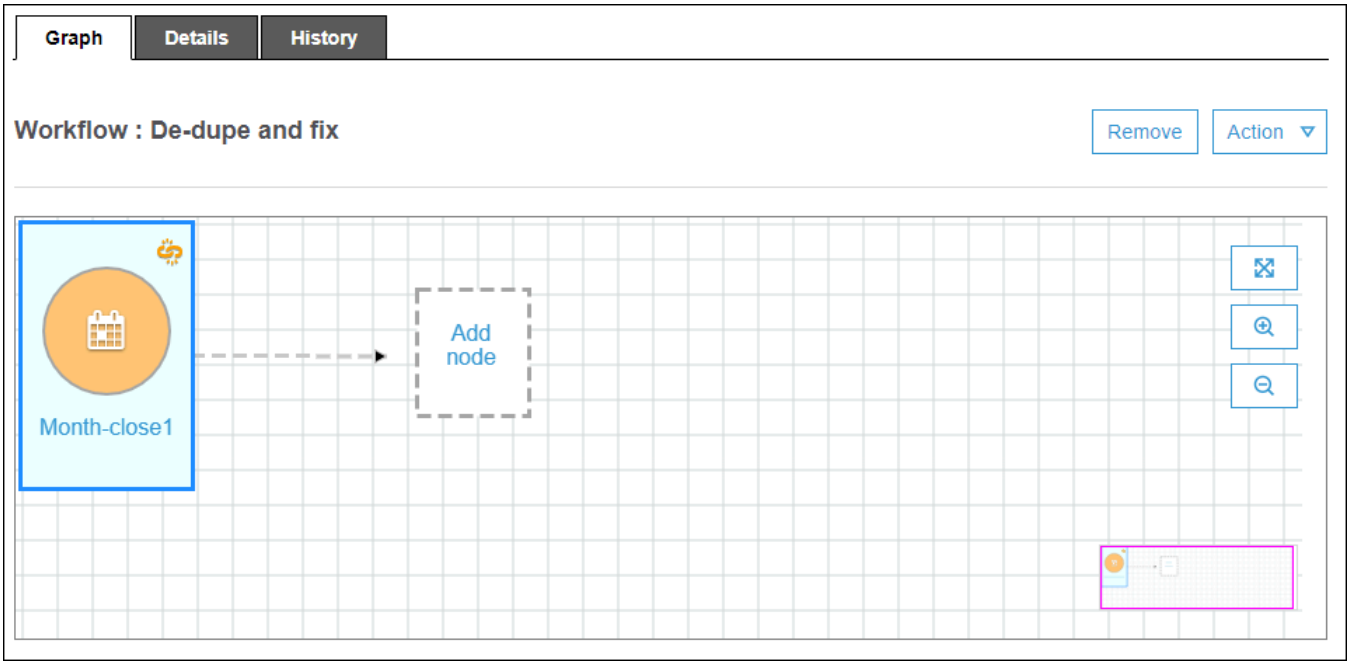
        Untuk tipe pemicu Jadwal, pilih salah satu pilihan Frekuensi. Pilih Kustom untuk memasukkan ekspresi `cron`.

        Untuk EventBridge peristiwa jenis pemicu, masukkan Jumlah peristiwa (ukuran batch), dan secara opsional masukkan Waktu tunda (jendela batch). Jika Anda menghilangkan Waktu tunda, jendela batch secara default menjadi 15 menit. Untuk informasi selengkapnya, lihat Ikhtisar alur kerja di AWS Glue.

     2. Pilih Tambahkan.

     Pemicu muncul pada grafik, bersama dengan simpul placeholder (berlabel Tambahkan simpul). Pada contoh di bawah ini, pemicu mulai adalah pemicu jadwal bernama `Month-close1`.

     Pada titik ini, pemicunya masih belum disimpan.



3. Jika Anda menambahkan sebuah pemicu baru, selesaikan langkah-langkah berikut:

   a. Lakukan salah satu dari berikut:

      - Pilih simpul placeholder (Tambahkan simpul).
      - Pastikan bahwa pemicu mulai sudah dipilih, dan pada menu Tindakan di atas grafik, pilih Tambahkan tugas/crawler yang akan dipicu.

   b. Di kotak dialog Tambahkan tugas dan crawler yang akan dipicu, pilih satu atau beberapa tugas atau crawler, dan kemudian pilih Tambahkan.

      Pemicu disimpan, dan tugas atau crawler yang dipilih muncul pada grafik dengan konektor dari pemicu.

      Jika Anda keliru menambahkan tugas atau crawler yang salah, maka Anda dapat memilih pemicu atau konektor dan kemudian pilih Hapus.

Langkah 3: Tambahkan lebih banyak pemicu

Lanjutkan untuk membangun alur kerja Anda dengan menambahkan lebih banyak pemicu jenis Peristiwa. Untuk memperbesar atau memperkecil kanvas grafik, gunakan ikon di sebelah kanan grafik. Untuk setiap pemicu yang akan ditambahkan, selesaikan langkah berikut:

> **Note**
>
> Tidak ada tindakan yang bisa digunakan untuk menyimpan alur kerja. Setelah Anda menambahkan pemicu terakhir Anda dan menetapkan tindakan ke pemicu, alur kerja selesai dan disimpan. Anda selalu dapat kembali dan menambahkan lebih banyak simpul nanti.

1. Lakukan salah satu dari berikut:

   - Untuk mengkloning pemicu yang ada, pastikan bahwa tidak ada simpul pada grafik yang dipilih, dan pada menu Tindakan, pilih Tambahkan pemicu.
   - Untuk menambahkan pemicu baru yang mengawasi tugas atau crawler tertentu pada grafik, pilih simpul tugas atau crawler, lalu pilih simpul placeholder Tambahkan pemicu.

     Anda dapat menambahkan lebih banyak tugas atau crawler untuk mengawasi pemicu ini di langkah berikutnya.

2. Di kotak dialog Tambahkan pemicu, lakukan salah satu hal berikut:

   - Pilih Tambah baru, dan isi lengkap Tambahkan pemicu. Kemudian pilih Add.

     Pemicu akan muncul pada grafik. Anda akan menyelesaikan pemicu di langkah berikutnya.
   - Pilih Kloning yang ada, dan pilih pemicu yang akan dikloning. Kemudian pilih Add.

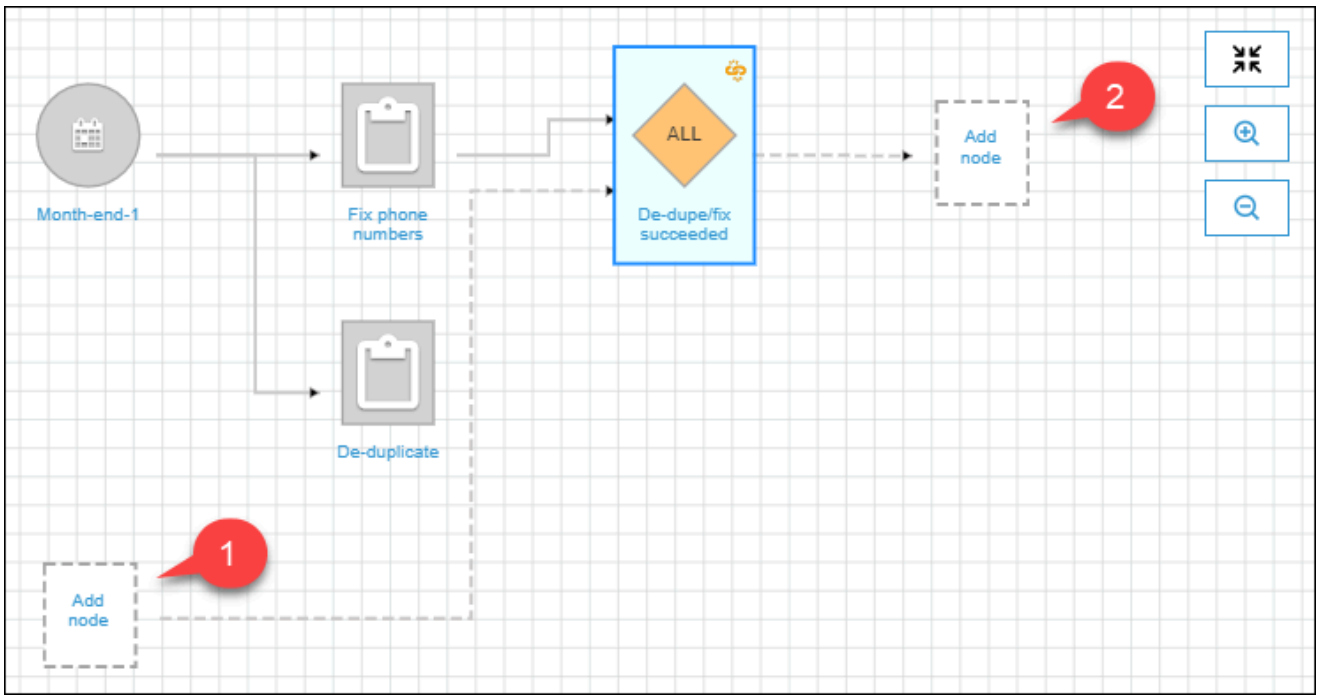
     Pemicu muncul pada grafik, bersama dengan tugas dan crawler yang diawasi dan tugas dan crawler yang dimulai.

     Jika Anda keliru memilih pemicu yang salah, pilih pemicu pada grafik, dan kemudian pilih Hapus.

3. Jika Anda menambahkan sebuah pemicu baru, selesaikan langkah-langkah berikut:

   a. Pilih pemicu baru.

      Sebagaimana yang ditunjukkan pada grafik berikut, pemicu `De-dupe/fix succeeded` sudah dipilih, dan simpul placeholder muncul untuk (1) peristiwa yang akan diawasi dan (2) tindakan.



b. (Opsional jika pemicu sudah mengawasi peristiwa dan Anda ingin menambahkan lebih banyak tugas atau crawler untuk diawasi.) Pilih node events-to-watch placeholder, dan di kotak dialog Tambah pekerjaan dan crawler untuk menonton, pilih satu atau beberapa pekerjaan atau crawler. Pilih peristiwa yang akan diawasi (BERHASIL, GAGAL, dll.), dan pilih Tambahkan.

c. Pastikan bahwa pemicu sudah dipilih, dan pilih simpul placeholder tindakan.

d. Di kotak dialog Tambahkan tugas dan crawler yang akan diawasi, pilih satu atau beberapa tugas atau crawler, lalu pilih Tambahkan.

   Tugas dan crawler yang dipilih muncul pada grafik dengan konektor dari pemicu.

Untuk informasi selengkapnya tentang alur kerja dan cetak biru, lihat topik berikut.

- Ikhtisar alur kerja di AWS Glue
- Menjalankan dan memantau alur kerja di AWS Glue
- Membuat alur kerja dari cetak biru di AWS Glue

Memulai AWS Glue alur kerja dengan acara Amazon EventBridge

Amazon EventBridge, juga dikenal sebagai CloudWatch Acara, memungkinkan Anda untuk mengotomatiskan AWS layanan Anda dan merespons secara otomatis peristiwa sistem seperti

masalah ketersediaan aplikasi atau perubahan sumber daya. Acara dari AWS layanan dikirim ke EventBridge dalam waktu dekat. Anda dapat menulis aturan sederhana untuk menunjukkan kejadian mana yang sesuai kepentingan Anda, dan tindakan otomatis apa yang diambil ketika suatu kejadian sesuai dengan suatu aturan.

Dengan EventBridge dukungan, AWS Glue dapat berfungsi sebagai produser acara dan konsumen dalam arsitektur berbasis acara. Untuk alur kerja, AWS Glue mendukung semua jenis EventBridge acara sebagai konsumen. Kasus penggunaan yang paling umum adalah kedatangan objek baru dalam bucket Amazon S3. Jika Anda memiliki data yang tiba dalam interval yang tidak teratur atau tidak tertentu, maka Anda dapat memproses data ini sedekat mungkin dengan kedatangannya.

> **Note**
>
> AWS Gluetidak memberikan jaminan pengiriman EventBridge pesan. AWS Gluetidak melakukan deduplikasi jika EventBridge mengirimkan pesan duplikat. Anda harus mengelola idempotensi berdasarkan kasus penggunaan Anda.
> Pastikan untuk mengonfigurasi EventBridge aturan dengan benar untuk menghindari pengiriman peristiwa yang tidak diinginkan.

Sebelum Anda memulai

Jika Anda ingin memulai alur kerja dengan peristiwa data Amazon S3, Anda harus memastikan bahwa peristiwa untuk bucket minat S3 dicatat dan. AWS CloudTrail EventBridge Untuk melakukannya, Anda harus membuat CloudTrail jejak. Untuk informasi selengkapnya, lihat Membuat jejak untuk akun AWS Anda.

Untuk memulai alur kerja dengan acara EventBridge

> **Note**
>
> Pada perintah berikut, ganti:
>
> - *<workflow-name>* dengan nama yang akan ditetapkan ke alur kerja.
> - *<trigger-name>* dengan nama untuk menetapkan pemicu.
> - *<bucket-name>* dengan nama bucket Amazon S3.
> - *<account-id>* dengan ID akun AWS yang valid.
> - *<region>* dengan nama Wilayah (misalnya, `us-east-1`).

- <rule-name>dengan nama untuk ditetapkan ke EventBridge aturan.

1. Pastikan Anda memiliki izin AWS Identity and Access Management (IAM) untuk membuat dan melihat EventBridge aturan dan target. Berikut ini adalah contoh kebijakan yang dapat Anda lampirkan. Anda mungkin ingin untuk membuat cakupan untuk menempatkan batas pada operasi dan sumber daya.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "events:PutRule",
                "events:DisableRule",
                "events:DeleteRule",
                "events:PutTargets",
                "events:RemoveTargets",
                "events:EnableRule",
                "events:List*",
                "events:Describe*"
            ],
            "Resource": "*"
        }
    ]
}
```

2. Buat peran IAM yang dapat diasumsikan oleh EventBridge layanan saat meneruskan AWS Glue acara.

   a. Pada halaman Buat peran di konsol IAM, pilih Layanan AWS. Kemudian pilih layanan CloudWatch Acara.

   b. Selesaikan penuntun Buat peran. Penuntun secara otomatis melampirkan kebijakan `CloudWatchEventsBuiltInTargetExecutionAccess` dan `CloudWatchEventsInvocationAccess`.

   c. Lampirkan kebijakan inline ke peran. Kebijakan ini memungkinkan EventBridge layanan untuk mengarahkan acara keAWS Glue.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:notifyEvent"
      ],
      "Resource": [
        "arn:aws:glue:<region>:<account-id>:workflow/<workflow-name>"
      ]
    }
  ]
}
```

3. Masukkan perintah berikut untuk membuat alur kerja.

   Lihat create-workflow di Referensi Perintah AWS CLI untuk informasi tentang tambahan parameter baris perintah opsional.

   ```
   aws glue create-workflow --name <workflow-name>
   ```

4. Masukkan perintah berikut untuk membuat pemicu EventBridge peristiwa untuk alur kerja. Ini akan menjadi pemicu mulai untuk alur kerja tersebut. Ganti *<actions>* dengan tindakan yang akan dilakukan (tugas dan crawler yang akan dimulai).

   Lihat create-trigger di Referensi Perintah AWS CLI untuk informasi tentang cara meng-coding argumen `actions`.

   ```
   aws glue create-trigger --workflow-name <workflow-name> --type EVENT --
   name <trigger-name> --actions <actions>
   ```

   Jika Anda ingin alur kerja dipicu oleh sekumpulan peristiwa, bukan satu EventBridge peristiwa, masukkan perintah berikut sebagai gantinya.

   ```
   aws glue create-trigger --workflow-name <workflow-name> --type EVENT
    --name <trigger-name> --event-batching-condition BatchSize=<number-of-
   events>,BatchWindow=<seconds> --actions <actions>
   ```

   Untuk argumen `event-batching-condition`, `BatchSize` wajib dan `BatchWindow` opsional. Jika `BatchWindow` dihilangkan, maka jendela default menjadi 900 detik, yang merupakan ukuran jendela maksimum.

Example

Contoh berikut menciptakan pemicu yang memulai `eventtest` alur kerja setelah tiga EventBridge peristiwa tiba, atau lima menit setelah acara pertama tiba, mana yang lebih dulu.

```
aws glue create-trigger --workflow-name eventtest --type EVENT --name objectArrival
 --event-batching-condition BatchSize=3,BatchWindow=300 --actions JobName=test1
```

5. Buat aturan di Amazon EventBridge.

   a. Buat objek JSON untuk detail aturan dalam editor teks yang Anda suka.

      Contoh berikut menentukan Amazon S3 sebagai sumber peristiwa, `PutObject` sebagai nama peristiwa, dan nama bucket sebagai parameter permintaan. Aturan ini memulai alur kerja ketika objek baru tiba di bucket tersebut.

```
{
  "source": [
    "aws.s3"
  ],
  "detail-type": [
    "AWS API Call via CloudTrail"
  ],
  "detail": {
    "eventSource": [
      "s3.amazonaws.com"
    ],
    "eventName": [
      "PutObject"
    ],
    "requestParameters": {
      "bucketName": [
        "<bucket-name>"
      ]
    }
  }
}
```

      Untuk memulai alur kerja ketika objek baru tiba di folder dalam bucket, Anda dapat mengganti kode berikut dengan `requestParameters`.

```
    "requestParameters": {
      "bucketName": [
        "<bucket-name>"
      ]
      "key" : [{ "prefix" : "<folder1>/<folder2>/*"}}]
  }
```

b. Gunakan alat pilihan Anda untuk mengkonversi aturan objek JSON menjadi escape string.

```
{\n  \"source\": [\n    \"aws.s3\"\n  ],\n  \"detail-type\": [\n    \"AWS API
 Call via CloudTrail\"\n  ],\n  \"detail\": {\n    \"eventSource\": [\n
 \"s3.amazonaws.com\"\n    ],\n    \"eventName\": [\n      \"PutObject\"\n
  ],\n    \"requestParameters\": {\n      \"bucketName\": [\n        \"<bucket-
name>\"\n      ]\n    }\n  }\n}
```

c. Jalankan perintah berikut untuk membuat templat parameter JSON yang dapat Anda edit untuk menentukan parameter input ke perintah `put-rule` berikutnya. Simpan output dalam sebuah file. Dalam contoh ini, file itu disebut `ruleCommand`.

```
aws events put-rule --name <rule-name> --generate-cli-skeleton >ruleCommand
```

Untuk informasi lebih lanjut tentang parameter `--generate-cli-skeleton`, lihat Menghasilkan kerangka AWS CLI dan parameter input dari file input JSON atau YAML dalam Panduan Pengguna Command Line Interface AWS.

File outputnya akan terlihat seperti berikut.

```
{
    "Name": "",
    "ScheduleExpression": "",
    "EventPattern": "",
    "State": "ENABLED",
    "Description": "",
    "RoleArn": "",
    "Tags": [
        {
            "Key": "",
            "Value": ""
        }
    ],
    "EventBusName": ""
```

```
}
```

d. Mengedit file untuk secara opsional menghapus parameter dan untuk menentukan parameter `Name`, `EventPattern`, dan `State` minimum. Untuk parameter `EventPattern`, berikan escape string untuk detail aturan yang Anda buat di langkah sebelumnya.

```
{
    "Name": "<rule-name>",
    "EventPattern": "{\n  \"source\": [\n    \"aws.s3\"\n  ],\n  \"detail-
type\": [\n    \"AWS API Call via CloudTrail\"\n  ],\n  \"detail\": {\n
 \"eventSource\": [\n      \"s3.amazonaws.com\"\n    ],\n    \"eventName\": [\n
      \"PutObject\"\n    ],\n    \"requestParameters\": {\n      \"bucketName
\": [\n        \"<bucket-name>\"\n      ]\n    }\n  }\n}",
    "State": "DISABLED",
    "Description": "Start an AWS Glue workflow upon new file arrival in an
 Amazon S3 bucket"
}
```

> **Note**
>
> Cara terbaik adalah membiarkan aturan dinonaktifkan sampai Anda selesai membangun alur kerja.

e. Masukkan perintah `put-rule` berikut, yang membaca parameter masukan dari file `ruleCommand`.

```
aws events put-rule --name <rule-name> --cli-input-json file://ruleCommand
```

Output berikut menunjukkan keberhasilan.

```
{
    "RuleArn": "<rule-arn>"
}
```

6. Masukkan perintah berikut untuk melampirkan aturan ke sebuah target. Targetnya adalah alur kerja di AWS Glue. Ganti *<role-name>* dengan peran yang Anda buat pada awal prosedur ini.

```
aws events put-targets --rule <rule-name> --targets
 "Id"="1","Arn"="arn:aws:glue:<region>:<account-id>:workflow/<workflow-
name>","RoleArn"="arn:aws:iam::<account-id>:role/<role-name>" --region <region>
```

Output berikut menunjukkan keberhasilan.

```
{
    "FailedEntryCount": 0,
    "FailedEntries": []
}
```

7. Konfirmasikan koneksi berhasil antara aturan dan target dengan memasukkan perintah berikut.

```
aws events list-rule-names-by-target --target-arn arn:aws:glue:<region>:<account-id>:workflow/<workflow-name>
```

Output berikut menunjukkan keberhasilan, di mana *<rule-name>* adalah nama aturan yang Anda buat.

```
{
    "RuleNames": [
        "<rule-name>"
    ]
}
```

8. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
9. Pilih alur kerja, dan verifikasi bahwa pemicu awal dan tindakannya—tugas atau crawler yang dimulai—muncul di grafik alur kerja. Lalu lanjutkan dengan prosedur di Langkah 3: Tambahkan lebih banyak pemicu. Atau tambahkan lebih banyak komponen ke alur kerja dengan menggunakan API AWS Glue atau AWS Command Line Interface.
10. Bila alur kerja benar-benar ditentukan, aktifkan aturan.

```
aws events enable-rule --name <rule-name>
```

Alur kerja sekarang siap untuk dimulai oleh event atau EventBridge event batch.

Lihat juga

- Panduan EventBridge Pengguna Amazon
- Ikhtisar alur kerja di AWS Glue

- Membuat dan membangun alur kerja secara manual di AWS Glue

Melihat EventBridge peristiwa yang memulai alur kerja

Anda dapat melihat ID peristiwa EventBridge peristiwa Amazon yang memulai alur kerja Anda. Jika alur kerja Anda dimulai oleh sekumpulan peristiwa, Anda dapat melihat ID peristiwa dari semua peristiwa dalam batch.

Untuk alur kerja dengan ukuran batch yang lebih besar dari satu, Anda juga dapat melihat syarat batch mana yang memulai alur kerja: datangnya jumlah peristiwa dalam ukuran batch, atau kedaluwarsa-nya jendela batch.

Untuk melihat EventBridge peristiwa yang memulai alur kerja (konsol)

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pilih Alur Kerja.
3. Pilih alur kerja. Kemudian di bagian bawah, pilih tab Riwayat.
4. Pilih eksekusi alur kerja, dan kemudian pilih Lihat detail eksekusi.
5. Pada halaman detail eksekusi, temukan kolom Properti eksekusi, dan cari kunci aws:eventIds.

   Nilai untuk kunci tersebut adalah daftar ID EventBridge peristiwa.

Untuk melihat EventBridge peristiwa yang memulai alur kerja (AWSAPI)

- Sertakan kode berikut dalam skrip Python Anda.

```
workflow_params =
 glue_client.get_workflow_run_properties(Name=workflow_name,RunId=workflow_run_id)
batched_events = workflow_params['aws:eventIds']
```

  `batched_events` adalah daftar string, di mana setiap string adalah ID peristiwa.

Lihat juga

- Panduan EventBridge Pengguna Amazon

- the section called “Ikhtisar alur kerja”

Menjalankan dan memantau alur kerja di AWS Glue

Jika pemicu awal untuk sebuah alur kerja adalah pemicu sesuai permintaan, Anda dapat memulai alur kerja tersebut dari konsol AWS Glue. Selesaikan langkah-langkah berikut untuk menjalankan dan memantau sebuah alur kerja. Jika sebuah alur kerja gagal, Anda dapat melihat grafik eksekusinya untuk menentukan simpul yang gagal. Untuk membantu memecahkan masalah, jika alur kerja dibuat dari sebuah cetak biru, Anda dapat melihat eksekusi cetak biru untuk melihat nilai parameter cetak biru yang digunakan untuk membuat alur kerja tersebut. Untuk informasi selengkapnya, lihat the section called “Melihat cetak biru berjalan”.

Anda dapat menjalankan dan memantau alur kerja dengan menggunakan konsol AWS Glue, API, atau AWS Command Line Interface (AWS CLI).

Untuk menjalankan dan memantau sebuah alur kerja (konsol)

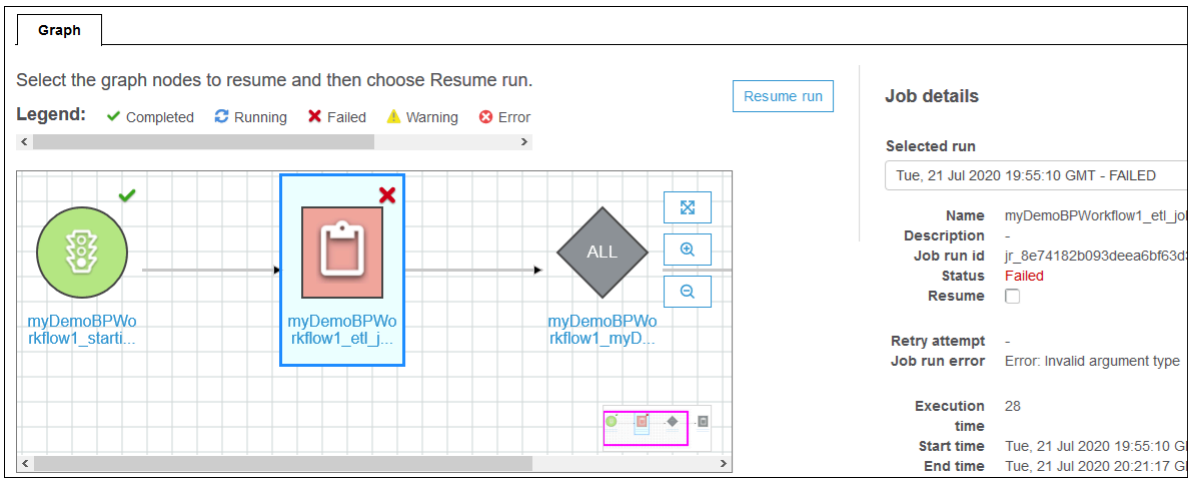
1. Buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada ETL, pilih Alur kerja.
3. Pilih sebuah alur kerja. Pada menu Tindakan, pilih Jalankan.
4. Periksa kolom Status eksekusi terakhir dalam daftar alur kerja. Pilih tombol refresh untuk melihat status alur kerja yang sedang berlangsung.
5. Saat alur kerja sedang berjalan atau setelah selesai (atau gagal), lihat detail eksekusi dengan menyelesaikan langkah-langkah berikut.

    a. Pastikan bahwa alur kerja sudah dipilih, dan pilih tab Riwayat.

    b. Pilih eksekusi alur kerja saat ini atau terbaru, lalu pilih Tampilkan detail eksekusi.

    Grafik waktu aktif alur kerja menunjukkan status eksekusi saat ini.

    c. Pilih simpul dalam grafik untuk melihat detail dan status simpul tersebut.



Untuk menjalankan dan memantau sebuah alur kerja (AWS CLI)

1. Masukkan perintah berikut. Ganti *<workflow-name>* dengan alur kerja yang akan dieksekusi.

```
aws glue start-workflow-run --name <workflow-name>
```

   Jika alur kerja berhasil dimulai, perintah mengembalikan ID eksekusi.

2. Tampilkan status eksekusi alur kerja dengan menggunakan perintah `get-workflow-run`. Berikan nama alur kerja dan ID eksekusi.

```
aws glue get-workflow-run --name myWorkflow --run-id
 wr_d2af14217e8eae775ba7b1fc6fc7a42c795aed3cbcd8763f9415452e2dbc8705
```

   Berikut adalah contoh output perintah.

```
{
    "Run": {
        "Name": "myWorkflow",
        "WorkflowRunId":
 "wr_d2af14217e8eae775ba7b1fc6fc7a42c795aed3cbcd8763f9415452e2dbc8705",
        "WorkflowRunProperties": {
            "run_state": "COMPLETED",
            "unique_id": "fee63f30-c512-4742-a9b1-7c8183bdaae2"
        },
        "StartedOn": 1578556843.049,
        "CompletedOn": 1578558649.928,
        "Status": "COMPLETED",
```

```
        "Statistics": {
            "TotalActions": 11,
            "TimeoutActions": 0,
            "FailedActions": 0,
            "StoppedActions": 0,
            "SucceededActions": 9,
            "RunningActions": 0,
            "ErroredActions": 0
        }
    }
}
```

Lihat juga:

- the section called “Ikhtisar alur kerja”
- the section called “Ikhtisar cetak biru”

Menghentikan alur kerja

Anda dapat menggunakan konsol AWS Glue, AWS Command Line Interface (AWS CLI) atau API AWS Glue untuk menghentikan eksekusi alur kerja. Bila Anda menghentikan sebuah eksekusi alur kerja, semua tugas dan crawler yang sedang berjalan akan segera diakhiri, dan tugas dan crawler yang belum dimulai tidak akan pernah dimulai. Mungkin diperlukan waktu hingga satu menit untuk semua tugas dan crawler yang sedang berjalan untuk berhenti. Status eksekusi alur kerja berubah dari Berjalan menjadi Akan berhenti, dan ketika eksekusi alur kerja benar-benar berhenti, statusnya akan menjadi Berhenti.

Setelah eksekusi alur kerja sudah dihentikan, Anda dapat melihat grafik eksekusi untuk melihat tugas dan crawler yang selesai dan yang tidak pernah dimulai. Anda kemudian dapat menentukan apakah Anda perlu melakukan langkah-langkah untuk memastikan integritas data. Menghentikan eksekusi alur kerja akan menyebabkan tidak ada operasi rollback otomatis yang akan dilakukan.

Untuk menghentikan eksekusi alur kerja (konsol)

1. Buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pada ETL, pilih Alur kerja.
3. Pilih alur kerja yang sedang berjalan, lalu pilih tab Riwayat.

4. Pilih eksekusi alur kerja, lalu pilih Hentikan eksekusi.

   Status eksekusi berubah menjadi Akan berhenti.

5. (Opsional) Pilih eksekusi alur kerja, pilih Tampilkan detail eksekusi, dan tinjau grafik eksekusi.

Untuk menghentikan eksekusi alur kerja (AWS CLI)

- Masukkan perintah berikut. Ganti *<workflow-name>* dengan nama alur kerja dan *<run-id>* dengan ID eksekusi dari eksekusi alur kerja yang akan dihentikan.

```
aws glue stop-workflow-run --name <workflow-name> --run-id <run-id>
```

  Berikut ini adalah contoh perintah stop-workflow-run.

```
aws glue stop-workflow-run --name my-workflow --run-id
 wr_137b88917411d128081069901e4a80595d97f719282094b7f271d09576770354
```

Memperbaiki dan melanjutkan alur kerja

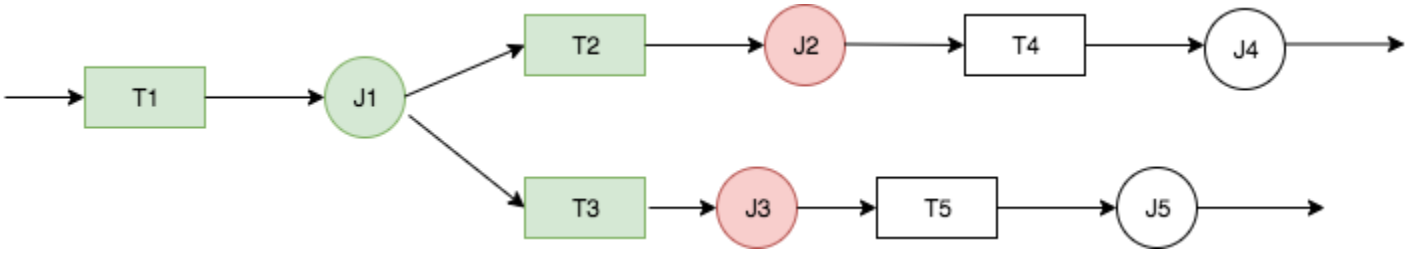
Jika satu atau beberapa simpul (tugas atau crawler) dalam sebuah alur kerja tidak berhasil selesai, ini berarti bahwa alur kerja hanya dieksekusi sebagian. Setelah Anda menemukan akar penyebab dan melakukan koreksi, Anda dapat memilih satu atau beberapa simpul sebagai tempat melanjutkan eksekusi alur kerja, dan kemudian melanjutkan eksekusi alur kerja. Simpul yang dipilih dan semua simpul yang menjadi hilir dari simpul tersebut kemudian dijalankan.

Topik

- Melanjutkan alur kerja: Cara kerjanya
- Melanjutkan alur kerja
- Catatan dan batasan untuk melanjutkan alur kerja berjalan

Melanjutkan alur kerja: Cara kerjanya

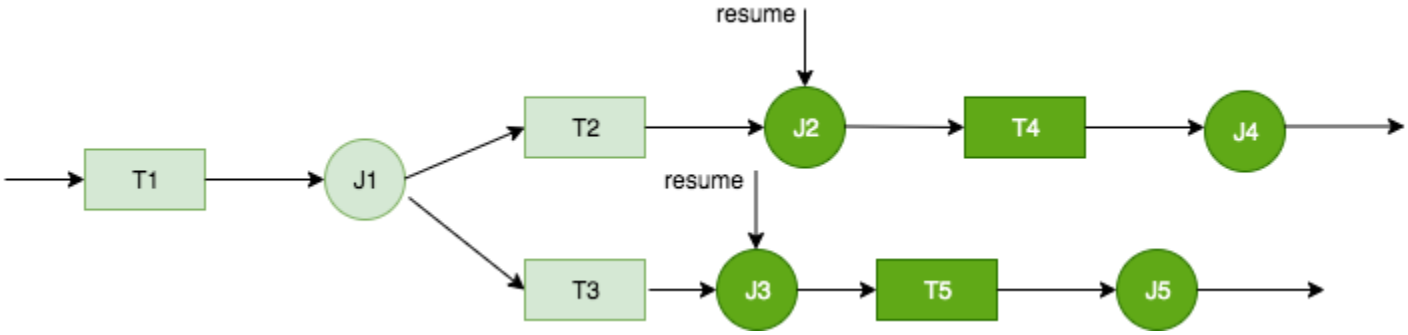
Perhatikan alur kerja W1 dalam diagram berikut.



Eksekusi alur kerja dilanjutkan sebagai berikut:

1. Pemicu T1 memulai tugas J1.
2. Keberhasilan penyelesaian J1 mengaktifkan pemicu T2 dan T3, yang masing-masing menjalankan tugas J2 dan J3.
3. Tugas J2 dan J3 gagal.
4. Pemicu T4 dan T5 tergantung pada keberhasilan penyelesaian J2 dan J3, sehingga pemicu tersebut tidak aktif, dan tugas J4 dan J5 tidak berjalan. Alur kerja W1 hanya dieksekusi sebagian.

Sekarang, anggap bahwa masalah-masalah yang menyebabkan J2 dan J3 gagal diperbaiki. J2 dan J3 dipilih sebagai titik awal untuk melanjutkan eksekusi alur kerja.



Eksekusi alur kerja berjalan kembali seperti berikut ini:

1. Tugas J2 dan J3 berjalan dengan sukses.
2. Pemicu T4 dan T5 aktif.
3. Tugas J4 dan J5 berjalan dengan sukses.

Eksekusi alur kerja yang dilanjutkan dilacak sebagai alur kerja terpisah yang dijalankan dengan ID eksekusi baru. Ketika Anda melihat riwayat alur kerja, Anda dapat melihat ID eksekusi sebelumnya untuk setiap eksekusi alur kerja. Dalam contoh pada tangkapan layar berikut, eksekusi alur kerja dengan ID eksekusi `wr_c7a22...` (baris kedua) mempunyai simpul yang tidak lengkap. Pengguna

memperbaiki masalah dan melanjutkan eksekusi alur kerja, yang menghasilkan ID eksekusi `wr_a07e55...` (baris pertama).

> **Note**
>
> Untuk sisa pembahasan ini, istilah "eksekusi alur kerja yang dilanjutkan" mengacu pada alur kerja yang dibuat ketika eksekusi alur kerja sebelumnya dilanjutkan. "Eksekusi alur kerja asli" mengacu pada eksekusi alur kerja yang hanya dieksekusi sebagian dan perlu dilanjutkan.

Grafik lari alur kerja yang dilanjutkan

Dalam eksekusi alur kerja yang dilanjutkan, meskipun hanya satu subset dari simpul yang dijalankan, tetapi grafik eksekusi adalah sebuah grafik yang lengkap. Artinya, simpul yang tidak dijalankan di alur kerja yang dilanjutkan disalin dari grafik eksekusi dari eksekusi alur kerja asli. Simpul tugas dan crawler yang disalin yang berjalan di eksekusi alur kerja asli mencakup detail eksekusi.

Pertimbangkan lagi alur kerja W1 pada diagram sebelumnya. Ketika eksekusi alur kerja yang dilanjutkan memulai dengan J2 dan J3, grafik eksekusi untuk eksekusi alur kerja yang dilanjutkan menunjukkan semua tugas, J1 hingga J5, dan semua pemicu, T1 hingga T5. Detail eksekusi untuk J1 disalin dari eksekusi alur kerja asli.

Alur kerja menjalankan snapshot

Ketika eksekusi alur kerja dimulai, AWS Glue mengambil snapshot dari grafik desain alur kerja pada saat itu. Snapshot yang digunakan di sepanjang eksekusi alur kerja. Jika Anda membuat perubahan pada pemicu setelah eksekusi dimulai, perubahan tersebut tidak akan mempengaruhi eksekusi alur kerja saat ini. Snapshot memastikan bahwa eksekusi alur kerja berjalan dengan cara yang konsisten.

Snapshot hanya membuat pemicu tetap. Perubahan yang Anda buat untuk tugas dan crawler hilir selama eksekusi alur kerja berlaku untuk eksekusi saat ini.

Melanjutkan alur kerja

Ikuti langkah-langkah ini untuk melanjutkan sebuah eksekusi alur kerja. Anda dapat melanjutkan eksekusi alur kerja dengan menggunakan konsol AWS Glue, API, atau AWS Command Line Interface (AWS CLI).

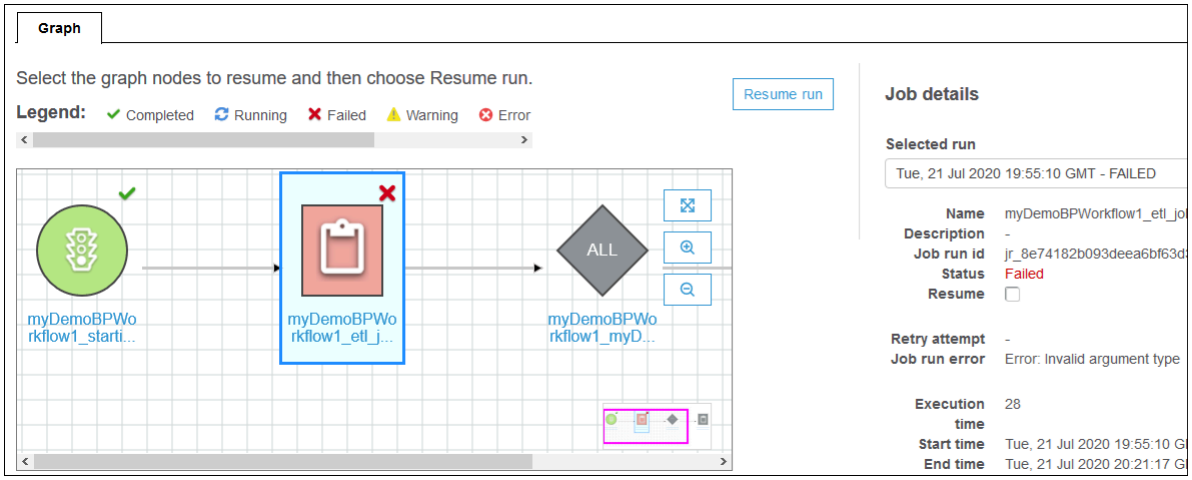
Untuk melanjutkan sebuah eksekusi alur kerja (konsol)

1. Buka konsol AWS Glue di https://console.aws.amazon.com/glue/.

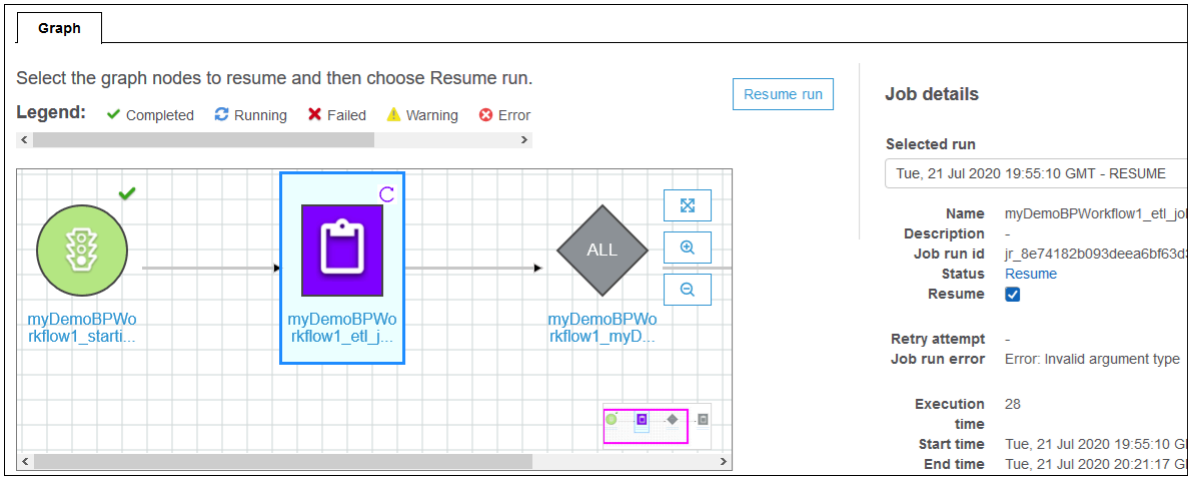
   Masuk sebagai pengguna yang memiliki izin untuk melihat alur kerja dan melanjutkan eksekusi alur kerja.

   > **Note**
   >
   > Untuk melanjutkan eksekusi alur kerja, Anda memerlukan izin AWS Identity and Access Management (IAM) `glue:ResumeWorkflowRun`.

2. Di panel navigasi, pilih Alur Kerja.
3. Pilih alur kerja, lalu pilih tab Riwayat.
4. Pilih eksekusi alur kerja yang hanya dieksekusi sebagian, dan kemudian pilih Lihat detail eksekusi.
5. Dalam grafik eksekusi, pilih simpul pertama (atau satu-satunya) yang ingin Anda mulai ulang dan yang ingin Anda jadikan tempat di mana eksekusi alur kerja dilanjutkan.
6. Di panel detail di sebelah kanan grafik, pilih kotak centang Lanjutkan.



   Simpul berubah warna dan menunjukkan ikon resume kecil di kanan atas.



7. Selesaikan dua langkah sebelumnya untuk setiap simpul tambahan untuk memulai ulang.
8. Pilih Lanjutkan eksekusi.

Untuk melanjutkan sebuah eksekusi alur kerja (AWS CLI)

1. Pastikan bahwa Anda memiliki Izin IAM `glue:ResumeWorkflowRun`.
2. Mengambil ID simpul untuk simpul yang ingin Anda mulai ulang.

   a. Jalankan perintah `get-workflow-run` untuk eksekusi alur kerja asli. Berikan nama alur kerja dan ID eksekusi, dan tambahkan opsi `--include-graph`, seperti yang ditunjukkan dalam contoh berikut. Dapatkan ID eksekusi dari tab Riwayat pada konsol, atau dengan menjalankan perintah `get-workflow`.

   ```
   aws glue get-workflow-run --name cloudtrailtest1 --run-id
    wr_a07e55f2087afdd415a404403f644a4265278f68b13ba3da08c71924ebe3c3a8 --include-
   graph
   ```

   Perintah tersebut akan mengembalikan simpul dan edge dari grafik tersebut sebagai objek JSON besar.

   b. Cari simpul yang menarik berdasarkan properti `Type` dan `Name` dari objek simpul tersebut.

   Berikut ini adalah sebuah simpul objek contoh dari output.

   ```
   {
       "Type": "JOB",
       "Name": "test1_post_failure_4592978",
   ```

```
    "UniqueId":
 "wnode_d1b2563c503078b153142ee76ce545fe5ceef66e053628a786ddd74a05da86fd",
    "JobDetails": {
        "JobRuns": [
            {
                "Id":
 "jr_690b9f7fc5cb399204bc542c6c956f39934496a5d665a42de891e5b01f59e613",
                "Attempt": 0,
                "TriggerName": "test1_aggregate_failure_649b2432",
                "JobName": "test1_post_failure_4592978",
                "StartedOn": 1595358275.375,
                "LastModifiedOn": 1595358298.785,
                "CompletedOn": 1595358298.785,
                "JobRunState": "FAILED",
                "PredecessorRuns": [],
                "AllocatedCapacity": 0,
                "ExecutionTime": 16,
                "Timeout": 2880,
                "MaxCapacity": 0.0625,
                "LogGroupName": "/aws-glue/python-jobs"
            }
        ]
    }
}
```

c. Dapatkan ID simpul dari properti `UniqueId` dari objek simpul tersebut.

3. Jalankan perintah `resume-workflow-run`. Berikan nama alur kerja, ID eksekusi, dan daftar ID simpul yang dipisahkan oleh spasi, seperti yang ditunjukkan dalam contoh berikut.

```
aws glue resume-workflow-run --name cloudtrailtest1 --run-id
 wr_a07e55f2087afdd415a404403f644a4265278f68b13ba3da08c71924ebe3c3a8 --node-
ids wnode_ca1f63e918fb855e063aed2f42ec5762ccf71b80082ae2eb5daeb8052442f2f3
 wnode_d1b2563c503078b153142ee76ce545fe5ceef66e053628a786ddd74a05da86fd
```

Perintah tersebut menghasilkan output ID eksekusi dari eksekusi alur kerja (baru) yang dilanjutkan dan daftar simpul yang akan dimulai.

```
{
    "RunId": "wr_2ada0d3209a262fc1156e4291134b3bd643491bcfb0ceead30bd3e4efac24de9",
    "NodeIds": [
        "wnode_ca1f63e918fb855e063aed2f42ec5762ccf71b80082ae2eb5daeb8052442f2f3"
    ]
```

```
}
```

Perhatikan bahwa meskipun contoh perintah `resume-workflow-run` mencantumkan dua simpul untuk memulai ulang, contoh output menunjukkan bahwa hanya satu simpul yang akan dimulai ulang. Hal ini karena satu simpul merupakan hilir dari simpul yang lain, dan simpul hilir bagaimanapun akan dimulai ulang oleh alur normal dari alur kerja tersebut.

Catatan dan batasan untuk melanjutkan alur kerja berjalan

Simpan catatan dan batasan berikut ini saat melanjutkan eksekusi alur kerja.

- Anda dapat melanjutkan eksekusi alur kerja hanya jika alur kerja tersebut dalam status `COMPLETED`.

> **Note**
>
> Bahkan jika satu simpul atau beberapa simpul dalam eksekusi alur kerja tidak selesai, status eksekusi alur kerja ditampilkan sebagai `COMPLETED`. Pastikan untuk memeriksa grafik eksekusi untuk menemukan simpul yang tidak berhasil diselesaikan.

- Anda dapat melanjutkan eksekusi alur kerja dari setiap simpul tugas atau crawler yang berusaha dijalankan oleh eksekusi alur kerja asli. Anda tidak dapat melanjutkan eksekusi alur kerja dari sebuah simpul pemicu.
- Memulai ulang simpul tidak akan me-reset statusnya. Setiap data yang diproses sebagian tidak akan dicabut.
- Anda dapat melanjutkan eksekusi alur kerja yang sama beberapa kali. Jika eksekusi alur kerja yang dilanjutkan hanya dieksekusi sebagian, Anda dapat mengatasi masalah dan melanjutkan eksekusi yang dilanjutkan tersebut.
- Jika Anda memilih dua simpul untuk memulai ulang dan mereka bergantung satu sama lain, maka simpul hulu akan dijalankan sebelum simpul hilir. Bahkan, memilih simpul hilir adalah sebuah redundansi, karena simpul hilir akan dieksekusi sesuai dengan alur normal dari alur kerja tersebut.

Mendapatkan dan mengatur alur kerja menjalankan properti di AWS Glue

Gunakan properti alur kerja untuk berbagi dan mengelola status di antara pekerjaan dalam AWS Glue alur kerja Anda. Anda dapat mengatur properti eksekusi default ketika Anda membuat alur

kerja. Kemudian, saat tugas Anda berjalan, mereka dapat mengambil nilai properti eksekusi dan secara opsional memodifikasinya untuk input ke tugas yang ada kemudian dalam alur kerja. Ketika sebuah tugas memodifikasi properti eksekusi, nilai baru ada hanya untuk eksekusi alur kerja. Properti eksekusi default tidak terpengaruh.

Jika pekerjaan AWS Glue Anda bukan bagian dari alur kerja, properti ini tidak akan disetel.

Kode Python sampel berikut dari tugas extract, transform, and load (ETL) menunjukkan bagaimana untuk mendapatkan properti eksekusi alur kerja.

```
import sys
import boto3
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from awsglue.context import GlueContext
from pyspark.context import SparkContext

glue_client = boto3.client("glue")
args = getResolvedOptions(sys.argv, ['JOB_NAME','WORKFLOW_NAME', 'WORKFLOW_RUN_ID'])
workflow_name = args['WORKFLOW_NAME']
workflow_run_id = args['WORKFLOW_RUN_ID']
workflow_params = glue_client.get_workflow_run_properties(Name=workflow_name,
                                       RunId=workflow_run_id)["RunProperties"]

target_database = workflow_params['target_database']
target_s3_location = workflow_params['target_s3_location']
```

Kode berikut berlanjut dengan menetapkan properti eksekusi `target_format` ke `'csv'`.

```
workflow_params['target_format'] = 'csv'
glue_client.put_workflow_run_properties(Name=workflow_name, RunId=workflow_run_id,
 RunProperties=workflow_params)
```

Untuk informasi selengkapnya, lihat yang berikut:

- GetWorkflowRunProperties tindakan (Python: get_workflow_run_properties)
- PutWorkflowRunProperties tindakan (Python: put_workflow_run_properties)

Menanyakan alur kerja menggunakan AWS Glue API

AWS Glue menyediakan API yang kaya untuk mengelola alur kerja. Anda dapat mengambil tampilan statis dari sebuah alur kerja atau tampilan dinamis dari sebuah alur kerja yang berjalan dengan menggunakan AWS Glue API. Untuk informasi selengkapnya, lihat Alur Kerja.

Topik

Menanyakan tampilan statis

Gunakan operasi API `GetWorkflow` untuk mendapatkan tampilan statis yang menunjukkan desain sebuah alur kerja. Operasi ini mengembalikan grafik diarahkan yang terdiri dari simpul dan edge, di mana simpul merupakan pemicu, tugas, atau crawler. Edge menentukan hubungan antara simpul. Mereka diwakili oleh konektor (panah) pada grafik di konsol AWS Glue.

Anda juga dapat menggunakan operasi ini dengan perpustakaan pengolahan grafis populer seperti NetworkX, igraph, JGrapht, dan Kerangka Kerja Java Universal Network/Graph (JUNG). Karena semua perpustakaan ini mewakili grafik yang sama, maka diperlukan hanya transformasi minimal.

Tampilan statis yang dikembalikan oleh API ini adalah up-to-date tampilan terbanyak menurut definisi pemicu terbaru yang terkait dengan alur kerja.

Definisi grafik

Sebuah grafik alur kerja G adalah pasangan diurutkan (N, E), di mana N adalah satu set simpul dan E satu set edge. Simpul adalah simpul dalam grafik yang diidentifikasi oleh nomor unik. Sebuah simpul dapat berupa jenis pemicu, tugas, atau crawler. Misalnya: `{name:T1, type:Trigger, uniqueId:1}, {name:J1, type:Job, uniqueId:2}`.

Edge adalah 2-tuple dalam bentuk (`src, dest`), di mana `src` dan `dest` adalah simpul dan ada edge diarahkan dari `src` ke `dest`.

Contoh kueri tampilan statis

Anggap pemicu bersyarat T, yang memicu tugas J2 setelah penyelesaian tugas J1.

```
J1 ---> T ---> J2
```

Simpul: J1, T, J2

Edge: (J1, T), (T, J2)

Menanyakan tampilan dinamis

Gunakan operasi API `GetWorkflowRun` untuk mendapatkan tampilan dinamis dari alur kerja yang berjalan. Operasi ini mengembalikan tampilan statis yang sama dari grafik bersama dengan metadata yang dikaitkan ke eksekusi alur kerja.

Untuk menjalankan, simpul yang mewakili tugas dalam panggilan `GetWorkflowRun` memiliki daftar eksekusi tugas yang dimulai sebagai bagian dari eksekusi alur kerja terbaru. Anda dapat menggunakan daftar ini untuk menampilkan status eksekusi dari setiap tugas dalam grafik itu sendiri. Untuk dependensi hilir yang belum dijalankan, kolom ini diatur ke `null`. Informasi grafik membuat Anda mengetahui status alur kerja saat ini pada setiap titik waktu.

Tampilan dinamis yang dikembalikan oleh API ini didasarkan pada tampilan statis yang disajikan saat eksekusi alur kerja dimulai.

Contoh simpul waktu aktif: `{name:T1, type: Trigger, uniqueId:1}`, `{name:J1, type:Job, uniqueId:2, jobDetails:{jobRuns}}`, `{name:C1, type:Crawler, uniqueId:3, crawlerDetails:{crawls}}`

Contoh 1: Tampilan dinamis

Contoh berikut mengilustrasikan sebuah alur kerja dua pemicu sederhana.

- Simpul: t1, j1, t2, j2
- Edge: (t1, j1), (j1, t2), (t2, j2)

Respons `GetWorkflow` berisi hal berikut ini.

```
{
    Nodes : [
        {
            "type" : Trigger,
            "name" : "t1",
            "uniqueId" : 1
        },
        {
            "type" : Job,
```

```
            "name" : "j1",
            "uniqueId" : 2
        },
        {
            "type" : Trigger,
            "name" : "t2",
            "uniqueId" : 3
        },
        {
            "type" : Job,
            "name" : "j2",
            "uniqueId" : 4
        }
    ],
    Edges : [
        {
            "sourceId" : 1,
            "destinationId" : 2
        },
        {
            "sourceId" : 2,
            "destinationId" : 3
        },
        {
            "sourceId" : 3,
            "destinationId" : 4
        }
}
```

Respons `GetWorkflowRun` berisi hal berikut ini.

```
{
    Nodes : [
        {
            "type" : Trigger,
            "name" : "t1",
            "uniqueId" : 1,
            "jobDetails" : null,
            "crawlerDetails" : null
        },
        {
            "type" : Job,
            "name" : "j1",
```

```
            "uniqueId" : 2,
            "jobDetails" : [
                {
                    "id" : "jr_12334",
                    "jobRunState" : "SUCCEEDED",
                    "errorMessage" : "error string"
                }
            ],
            "crawlerDetails" : null
        },
        {
            "type" : Trigger,
            "name" : "t2",
            "uniqueId" : 3,
            "jobDetails" : null,
            "crawlerDetails" : null
        },
        {
            "type" : Job,
            "name" : "j2",
            "uniqueId" : 4,
            "jobDetails" : [
                {
                    "id" : "jr_1233sdf4",
                    "jobRunState" : "SUCCEEDED",
                    "errorMessage" : "error string"
                }
            ],
            "crawlerDetails" : null
        }
    ],
    Edges : [
        {
            "sourceId" : 1,
            "destinationId" : 2
        },
        {
            "sourceId" : 2,
            "destinationId" : 3
        },
        {
            "sourceId" : 3,
            "destinationId" : 4
        }
```

```
}
```

Contoh 2: Beberapa pekerjaan dengan pemicu bersyarat

Contoh berikut menunjukkan sebuah alur kerja dengan beberapa eksekusi dan sebuah pemicu bersyarat (t3).

```
Consider Flow:
T(t1) ---> J(j1) ---> T(t2) ---> J(j2)
             |                     |
             |                     |
             >+------> T(t3) <-----+
                        |
                        |
                      J(j3)

Graph generated:
Nodes: t1, t2, t3, j1, j2, j3
Edges: (t1, j1), (j1, t2), (t2, j2), (j1, t3), (j2, t3), (t3, j3)
```

Pembatasan cetak biru dan alur kerja di AWS Glue

Berikut ini adalah pembatasan untuk cetak biru dan alur kerja.

Pembatasan cetak biru

Ingatlah pembatasan cetak biru berikut:

- Cetak biru harus didaftarkan dalam Wilayah AWS yang sama dengan wilayah bucket Amazon S3 berada.
- Untuk berbagi cetak biru di seluruh akun AWS Anda harus memberikan izin baca pada arsip ZIP cetak biru di Amazon S3. Pelanggan yang memiliki izin baca pada arsip ZIP cetak biru dapat mendaftarkan cetak biru di akun AWS dan menggunakannya.
- Sekumpulan parameter cetak biru disimpan sebagai satu objek JSON tunggal. Panjang maksimum objek tersebut adalah 128 KB.
- Ukuran maksimum tidak terkompresi dari arsip ZIP cetak biru adalah 5 MB. Ukuran terkompresi maksimum adalah 1 MB.
- Batasi jumlah total pekerjaan, crawler, dan pemicu dalam alur kerja hingga 100 atau kurang. Jika Anda menyertakan lebih dari 100, Anda mungkin mendapatkan kesalahan saat mencoba melanjutkan atau menghentikan alur kerja berjalan.

Pembatasan alur kerja

Perhatikan pembatasan alur kerja berikut ini. Beberapa komentar ini ditujukan lebih kepada para pengguna yang membuat alur kerja secara manual.

- Ukuran batch maksimum untuk pemicu EventBridge peristiwa Amazon adalah 100. Ukuran jendela maksimum adalah 900 detik (15 menit).
- Pemicu dapat dikaitkan hanya dengan satu alur kerja saja.
- Hanya satu pemicu awal (sesuai permintaan atau terjadwal) yang diizinkan.
- Jika tugas atau crawler dalam sebuah alur kerja dimulai oleh pemicu yang berada di luar alur kerja, maka pemicu apa pun yang ada di dalam alur kerja yang bergantung pada penyelesaian tugas atau crawler (berhasil atau lainnya) tidak akan aktif.
- Demikian pula, jika tugas atau crawler dalam sebuah alur kerja memiliki pemicu yang bergantung pada penyelesaian tugas atau crawler (berhasil atau lainnya) baik yang ada di dalam alur kerja maupun di luar alur kerja, dan jika tugas atau crawler dimulai dari dalam alur kerja, maka hanya pemicu di dalam alur kerja saja yang akan aktif saat tugas atau crawler selesai.

Memecahkan masalah kesalahan cetak biru di AWS Glue

Jika Anda mengalami kesalahan saat menggunakan AWS Glue cetak biru, gunakan solusi berikut untuk membantu Anda menemukan sumber masalah dan memperbaikinya.

Topik

Kesalahan: PySpark modul hilang

AWS Gluemengembalikan kesalahan “Kesalahan tidak diketahui menjalankan fungsi generator tata letak ModuleNotFoundError: Tidak ada modul bernama 'pyspark'”.

Saat Anda membuka zip arsip cetak biru, itu bisa seperti salah satu dari berikut ini:

```
$ unzip compaction.zip
Archive:  compaction.zip
   creating: compaction/
  inflating: compaction/blueprint.cfg
  inflating: compaction/layout.py
  inflating: compaction/README.md
  inflating: compaction/compaction.py

$ unzip compaction.zip
Archive:  compaction.zip
  inflating: blueprint.cfg
  inflating: compaction.py
  inflating: layout.py
  inflating: README.md
```

Dalam kasus pertama, semua file yang terkait dengan cetak biru ditempatkan di bawah folder bernama pemadatan dan kemudian diubah menjadi file zip bernama compaction.zip.

Dalam kasus kedua, semua file yang diperlukan untuk cetak biru tidak dimasukkan ke dalam folder dan ditambahkan sebagai file root di bawah file zip compaction.zip.

Membuat file dalam salah satu format di atas diperbolehkan. Namun pastikan bahwa `blueprint.cfg` memiliki jalur yang benar ke nama fungsi dalam skrip yang menghasilkan tata letak.

Contoh

Dalam kasus 1: `blueprint.cfg` harus memiliki `layoutGenerator` sebagai berikut:

```
layoutGenerator": "compaction.layout.generate_layout"
```

Dalam kasus 2: `blueprint.cfg` harus memiliki `layoutGenerator` sebagai berikut

```
layoutGenerator": "layout.generate_layout"
```

Jika jalur ini tidak disertakan dengan benar, Anda bisa melihat kesalahan seperti yang ditunjukkan. Misalnya, jika Anda memiliki struktur folder seperti yang disebutkan dalam kasus 2 dan Anda memiliki `layoutGenerator` indikasi seperti dalam kasus 1, Anda dapat melihat kesalahan di atas.

Kesalahan: file konfigurasi cetak biru hilang

AWS Gluemengembalikan kesalahan “Kesalahan tidak diketahui menjalankan fungsi generator tata letak FileNotFoundError: [Errno 2] Tidak ada file atau direktori seperti itu: '/tmp/compaction/blueprint.cfg'”.

blueprint.cfg harus ditempatkan pada tingkat root arsip ZIP atau di dalam folder yang memiliki nama yang sama dengan arsip ZIP.

Ketika kita mengekstrak arsip cetak biru ZIP, blueprint.cfg diharapkan dapat ditemukan di salah satu jalur berikut. Jika tidak ditemukan di salah satu jalur berikut, Anda dapat melihat kesalahan di atas.

```
$ unzip compaction.zip
Archive:  compaction.zip
   creating: compaction/
  inflating: compaction/blueprint.cfg

$ unzip compaction.zip
Archive:  compaction.zip
  inflating: blueprint.cfg
```

Kesalahan: file yang diimpor tidak ada

AWS Gluemengembalikan kesalahan “Kesalahan tidak diketahui menjalankan fungsi generator tata letak FileNotFoundError: [Errno 2] Tidak ada file atau direktori seperti itu: * * 'demo-project/foo.py'”.

Jika skrip pembuatan tata letak Anda memiliki fungsionalitas untuk membaca file lain, pastikan Anda memberikan jalur lengkap untuk file yang akan diimpor. Misalnya, skrip Conversion.py dapat direferensikan di Layout.py. Untuk informasi selengkapnya, lihat Contoh Blueprint Project.

Kesalahan: tidak diizinkan untuk tampil iamPassRole di sumber daya

AWS Gluemengembalikan kesalahan “User: arn:aws:sts: :123456789012: assumed-role// tidak diizinkan untuk melakukan: iam: on resource: arn:aws:iam: :123456789012: role/AWSGlueServiceRole” GlueSession PassRole AWSGlueServiceRole

Jika pekerjaan dan crawler dalam alur kerja memiliki peran yang sama dengan peran yang diteruskan untuk membuat alur kerja dari cetak biru, maka peran cetak biru perlu menyertakan izin itu sendiri. `iam:PassRole`

Jika pekerjaan dan crawler dalam alur kerja mengambil peran selain peran yang diteruskan untuk membuat entitas alur kerja dari cetak biru, maka peran cetak biru harus menyertakan `iam:PassRole` izin pada peran lain tersebut, bukan pada peran cetak biru.

Untuk informasi selengkapnya, lihat Izin untuk Peran cetak biru.

Kesalahan: jadwal cron tidak valid

AWS Gluemengembalikan kesalahan “Jadwal cron (0 0 * * * *) tidak valid.”

Berikan ekspresi cron yang valid. Untuk informasi selengkapnya, lihat Jadwal Berbasis Waktu untuk Tugas dan Perayap.

Kesalahan: pemicu dengan nama yang sama sudah ada

AWS Gluemengembalikan kesalahan “Pemicu dengan nama 'foo_starting_trigger' sudah dikirimkan dengan konfigurasi yang berbeda”.

Cetak biru tidak mengharuskan Anda untuk menentukan pemicu dalam skrip tata letak untuk pembuatan alur kerja. Pembuatan pemicu dikelola oleh pustaka cetak biru berdasarkan dependensi yang ditentukan antara dua tindakan.

Penamaan untuk pemicunya adalah sebagai berikut:

- Untuk pemicu awal dalam alur kerja penamaannya adalah <workflow_name>_starting_trigger.
- <workflow_name><node_name>Untuk node (job/crawler) dalam alur kerja yang bergantung pada penyelesaian salah satu atau beberapa node upstream; AWS Glue mendefinisikan pemicu dengan nama _ _trigger

Kesalahan ini berarti pemicu dengan nama yang sama sudah ada. Anda dapat menghapus pemicu yang ada dan menjalankan kembali pembuatan alur kerja.

> **Note**
>
> Menghapus alur kerja tidak menghapus node dalam alur kerja. Ada kemungkinan bahwa meskipun alur kerja dihapus, pemicu tertinggal. Karena ini, Anda mungkin tidak menerima

kesalahan 'alur kerja sudah ada', tetapi Anda mungkin menerima kesalahan 'pemicu sudah ada' dalam kasus di mana Anda membuat alur kerja, menghapusnya dan kemudian mencoba membuatnya kembali dengan nama yang sama dari cetak biru yang sama.

Kesalahan: alur kerja dengan nama: foo sudah ada.

Nama alur kerja harus unik. Silakan coba dengan nama yang berbeda.

Kesalahan: modul tidak ditemukan di jalur LayoutGenerator yang ditentukan

AWS Gluemengembalikan kesalahan “Kesalahan tidak diketahui menjalankan fungsi generator tata letak ModuleNotFoundError: Tidak ada modul bernama 'crawl_s3_locations'”.

```
layoutGenerator": "crawl_s3_locations.layout.generate_layout"
```

Misalnya, jika Anda memiliki jalur LayoutGenerator di atas, maka ketika Anda unzip arsip cetak biru, itu perlu terlihat seperti berikut:

```
$ unzip crawl_s3_locations.zip
Archive:  crawl_s3_locations.zip
   creating: crawl_s3_locations/
  inflating: crawl_s3_locations/blueprint.cfg
  inflating: crawl_s3_locations/layout.py
  inflating: crawl_s3_locations/README.md
```

Ketika Anda unzip arsip, jika arsip cetak biru terlihat seperti berikut, maka Anda bisa mendapatkan kesalahan di atas.

```
$ unzip crawl_s3_locations.zip
Archive:  crawl_s3_locations.zip
  inflating: blueprint.cfg
  inflating: layout.py
  inflating: README.md
```

Anda dapat melihat bahwa tidak ada folder bernama `crawl_s3_locations` dan ketika `layoutGenerator` path mengacu ke file tata letak melalui modul`crawl_s3_locations`, Anda bisa mendapatkan kesalahan di atas.

Kesalahan: kesalahan validasi di bidang Koneksi

AWS Gluemengembalikan kesalahan “Kesalahan tidak diketahui menjalankan fungsi generator tata letak TypeError: Nilai ['foo'] untuk kunci Koneksi harus bertipe <class 'dict'>!”.

Ini adalah kesalahan validasi. `Connections`Bidang di `Job` kelas mengharapkan kamus dan sebagai gantinya daftar nilai disediakan menyebabkan kesalahan.

```
User input was list of values
Connections= ['string']

Should be a dict like the following
Connections*=*{'Connections': ['string']}
```

Untuk menghindari kesalahan waktu proses ini saat membuat alur kerja dari cetak biru, Anda dapat memvalidasi definisi alur kerja, pekerjaan, dan perayap seperti yang diuraikan dalam Menguji cetak biru.

Lihat sintaks di Referensi Kelas AWS Glue cetak biru untuk menentukan AWS Glue pekerjaan, crawler, dan alur kerja dalam skrip tata letak.

Izin untuk persona dan peran untuk cetak biru AWS Glue

Berikut ini adalah kebijakan izin persona dan saran AWS Identity and Access Management (IAM) khas untuk persona dan peran untuk cetak biru. AWS Glue

Topik

- Blueprint persona
- Izin untuk persona cetak biru
- Izin untuk peran cetak biru

Blueprint persona

Berikut ini adalah persona yang biasanya terlibat dalam siklus hidup cetak biru. AWS Glue

Persona	Deskripsi
AWS Gluepengembang	Mengembangkan, menguji, dan menerbitkan cetak biru.

Persona	Deskripsi
AWS Glueadministrator	Mendaftar, memelihara, dan memberikan izin pada cetak biru.
Analis data	Menjalankan cetak biru untuk membuat alur kerja.

Untuk informasi selengkapnya, lihat the section called “Ikhtisar cetak biru”.

Izin untuk persona cetak biru

Berikut ini adalah izin yang disarankan untuk masing-masing persona cetak biru.

AWS Glueizin pengembang untuk cetak biru

AWS GluePengembang harus memiliki izin menulis di bucket Amazon S3 yang digunakan untuk menerbitkan cetak biru. Seringkali, developer mendaftarkan cetak biru setelah mengunggahnya. Dalam hal ini, developer membutuhkan izin yang tercantum dalam the section called “AWS Glueizin administrator untuk cetak biru”. Selain itu, jika developer ingin menguji cetak biru setelah terdaftar, maka ia juga membutuhkan izin yang tercantum di the section called “Izin analis data untuk cetak biru”.

AWS Glueizin administrator untuk cetak biru

Kebijakan berikut memberikan izin untuk mendaftar, melihat, dan memelihara AWS Glue cetak biru.

> **Important**
>
> Dalam kebijakan berikut, ganti *<s3-bucket-name>* dan *<prefix>* dengan path Amazon S3 menjadi arsip ZIP cetak biru yang sudah diunggah untuk mendaftar.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateBlueprint",
                "glue:UpdateBlueprint",
```

```
                "glue:DeleteBlueprint",
                "glue:GetBlueprint",
                "glue:ListBlueprints",
                "glue:BatchGetBlueprints"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::<s3-bucket-name>/<prefix>/*"
        }
    ]
}
```

Izin analis data untuk cetak biru

Kebijakan berikut memberikan izin untuk menjalankan cetak biru dan melihat komponen-komponen alur kerja dan alur kerja yang dihasilkan. Ini juga memberikan peran yang `PassRole` AWS Glue mengasumsikan untuk membuat alur kerja dan komponen alur kerja.

Kebijakan ini memberikan izin pada setiap sumber daya. Jika Anda ingin mengkonfigurasi akses detail ke cetak biru individu, gunakan format berikut untuk ARN cetak biru:

```
arn:aws:glue:<region>:<account-id>:blueprint/<blueprint-name>
```

> **Important**
>
> Dalam kebijakan berikut, ganti *<account-id>* dengan akun AWS yang valid dan ganti *<role-name>* dengan nama peran yang digunakan untuk menjalankan cetak biru. Lihat the section called “Izin untuk peran cetak biru” untuk melihat izin yang diperlukan peran ini.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "glue:ListBlueprints",
                "glue:GetBlueprint",
                "glue:StartBlueprintRun",
                "glue:GetBlueprintRun",
                "glue:GetBlueprintRuns",
                "glue:GetCrawler",
                "glue:ListTriggers",
                "glue:ListJobs",
                "glue:BatchGetCrawlers",
                "glue:GetTrigger",
                "glue:BatchGetWorkflows",
                "glue:BatchGetTriggers",
                "glue:BatchGetJobs",
                "glue:BatchGetBlueprints",
                "glue:GetWorkflowRun",
                "glue:GetWorkflowRuns",
                "glue:ListCrawlers",
                "glue:ListWorkflows",
                "glue:GetJob",
                "glue:GetWorkflow",
                "glue:StartWorkflowRun"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::<account-id>:role/<role-name>"
        }
    ]
}
```

Izin untuk peran cetak biru

Berikut ini adalah izin yang disarankan untuk IAM role yang digunakan untuk membuat alur kerja dari sebuah cetak biru. Peran tersebut harus memiliki hubungan kepercayaan dengan `glue.amazonaws.com`.

> **Important**
>
> Dalam kebijakan berikut, ganti *<account-id>* dengan akun AWS yang valid, dan ganti *<role-name>* dengan nama peran.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateJob",
                "glue:GetCrawler",
                "glue:GetTrigger",
                "glue:DeleteCrawler",
                "glue:CreateTrigger",
                "glue:DeleteTrigger",
                "glue:DeleteJob",
                "glue:CreateWorkflow",
                "glue:DeleteWorkflow",
                "glue:GetJob",
                "glue:GetWorkflow",
                "glue:CreateCrawler"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::<account-id>:role/<role-name>"
        }
    ]
}
```

> **Note**
>
> Jika tugas dan crawler dalam alur kerja tersebut mengambil peran selain peran ini, maka kebijakan ini harus menyertakan izin `iam:PassRole` pada peran lain, bukan pada peran cetak biru.

Mengembangkan cetak biru di AWS Glue

Organisasi Anda mungkin memiliki serangkaian kasus penggunaan ETL serupa yang dapat mengambil manfaat dari kemampuan membuat parameter satu alur kerja untuk menangani semuanya. Untuk mengatasi kebutuhan ini, AWS Glue memungkinkan Anda untuk menentukan cetak

biru, yang dapat Anda gunakan untuk menghasilkan alur kerja. Sebuah cetak biru menerima banyak parameter, sehingga dari cetak biru tunggal, seorang analis data dapat membuat alur kerja yang berbeda untuk menangani kasus penggunaan ETL yang sama. Setelah membuat sebuah cetak biru, Anda dapat menggunakannya kembali untuk departemen, tim, dan proyek yang berbeda.

Topik

Ikhtisar cetak biru di AWS Glue

> **Note**
>
> Fitur cetak biru saat ini tidak tersedia di Wilayah berikut di konsol AWS Glue: Asia Pasifik (Jakarta) dan Timur Tengah (UEA).

AWS Gluecetak biru menyediakan cara untuk membuat dan berbagi alur kerja. AWS Glue Ketika ada proses ETL kompleks yang dapat digunakan untuk kasus penggunaan serupa, daripada membuat AWS Glue alur kerja untuk setiap kasus penggunaan, Anda dapat membuat cetak biru tunggal.

Cetak biru menentukan tugas dan crawler untuk disertakan dalam alur kerja, dan menentukan parameter yang disediakan pengguna alur kerja saat mereka menjalankan cetak biru untuk membuat alur kerja. Penggunaan parameter memungkinkan satu cetak biru tunggal untuk menghasilkan alur kerja untuk berbagai kasus penggunaan serupa. Untuk informasi lebih lanjut tentang alur kerja, lihat Ikhtisar alur kerja di AWS Glue.

Berikut adalah contoh kasus penggunaan cetak biru:

- Anda ingin melakukan partisi pada set data yang ada. Parameter masukan untuk cetak biru adalah path sumber dan target Amazon Simple Storage Service (Amazon S3) dan daftar kolom partisi.

- Anda ingin membuat snapshot tabel Amazon DynamoDB ke penyimpanan data SQL seperti Amazon Redshift. Parameter input ke cetak biru adalah nama tabel DynamoDB dan koneksi, AWS Glue yang menunjuk cluster Amazon Redshift dan database tujuan.
- Anda ingin mengkonversi data CSV di beberapa path Amazon S3 ke Parquet. Anda ingin AWS Glue alur kerja menyertakan crawler dan pekerjaan terpisah untuk setiap jalur. Parameter input adalah database tujuan dalam Katalog AWS Glue Data dan daftar jalur Amazon S3 yang dibatasi koma. Perhatikan bahwa dalam kasus ini, jumlah crawler dan tugas yang dibuat alur kerja adalah variabel.

Komponen cetak biru

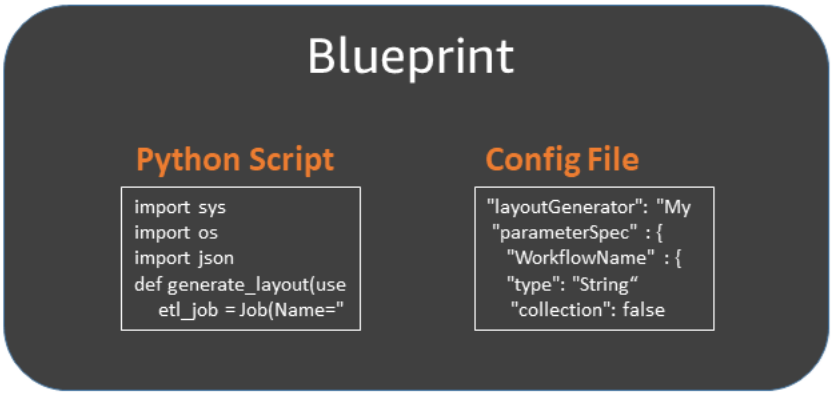
Cetak biru adalah arsip ZIP yang berisi komponen-komponen berikut:

- Sebuah skrip generator tata letak Python

  Berisi fungsi yang menentukan tata letak alur kerja—crawler dan tugas yang dapat dibuat untuk alur kerja, properti tugas dan crawler, serta dependensi antara tugas dan crawler. Fungsi menerima parameter cetak biru dan mengembalikan struktur alur kerja (objek JSON) yang AWS Glue digunakan untuk menghasilkan alur kerja. Karena Anda menggunakan skrip Python untuk menghasilkan alur kerja, dengan demikian Anda dapat menambahkan logika Anda sendiri yang cocok untuk kasus penggunaan Anda.
- Sebuah file konfigurasi

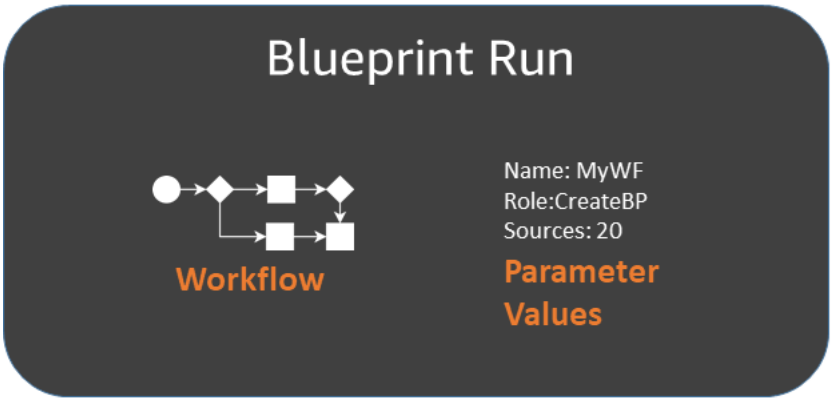
  Menentukan nama yang memenuhi syarat dari fungsi Python yang menghasilkan tata letak alur kerja. Juga menentukan nama, tipe data, dan properti lainnya dari semua parameter cetak biru yang digunakan oleh skrip.
- (Opsional) Skrip ETL dan file pendukung

  Sebagai kasus penggunaan lanjutan, Anda dapat melakukan parameterisasi pada lokasi skrip ETL yang digunakan tugas Anda. Anda dapat menyertakan file skrip tugas dalam arsip ZIP dan menentukan parameter cetak biru untuk lokasi Amazon S3 tempat di mana skrip akan disalin. Skrip generator tata letak dapat menyalin skrip ETL ke lokasi yang ditentukan dan menentukan lokasi itu sebagai properti lokasi skrip tugas. Anda juga dapat menyertakan perpustakaan atau file pendukung lainnya, dengan ketentuan bahwa skrip Anda menanganinya.



Eksekusi Cetak Biru

Saat Anda membuat alur kerja dari cetak biru, AWS Glue jalankan cetak biru, yang memulai proses asinkron untuk membuat alur kerja dan pekerjaan, crawler, dan pemicu yang dienkapsulasi alur kerja. AWS Gluemenggunakan blueprint run untuk mengatur pembuatan alur kerja dan komponennya. Anda melihat status proses pembuatan dengan melihat status eksekusi cetak biru. Eksekusi cetak biru juga menyimpan nilai-nilai yang Anda sediakan untuk parameter cetak biru.



Anda dapat melihat blueprint berjalan menggunakan AWS Glue konsol atau AWS Command Line Interface (). AWS CLI Saat melihat atau memecahkan masalah alur kerja, Anda selalu dapat kembali ke eksekusi cetak biru untuk melihat nilai parameter cetak biru yang digunakan untuk membuat alur kerja.

Siklus hidup cetak biru

cetak biru dikembangkan, diuji, didaftarkanAWS Glue, dan dijalankan untuk membuat alur kerja. Biasanya ada tiga persona yang terlibat dalam siklus hidup cetak biru.

Persona	Tugas
AWS Gluepengembang	• Menulis skrip tata letak alur kerja dan menciptakan file konfigurasi. • Menguji cetak biru secara lokal menggunakan pustaka yang disediakan oleh layanan. AWS Glue • Menciptakan arsip ZIP skrip, file konfigurasi, dan file pendukung dan menerbitkan arsip ke lokasi di Amazon S3. • Menambahkan kebijakan bucket ke bucket Amazon S3 yang memberikan izin baca pada objek bucket ke akun administrator. AWS Glue AWS • Memberikan izin baca IAM pada arsip ZIP di Amazon S3 kepada administrator. AWS Glue
AWS Glueadministrator	• Mendaftarkan cetak biru dengan. AWS Glue AWS Gluemembuat salinan arsip ZIP ke lokasi Amazon S3 yang dipesan. • Memberikan izin IAM pada cetak biru untuk analis data.
Analis data	• Menjalankan cetak biru untuk membuat sebuah alur kerja, dan memberikan nilai parameter cetak biru. Memeriksa status eksekusi cetak biru untuk memastikan bahwa alur kerja dan komponen alur kerja berhasil dibuat. • Menjalankan dan memecahkan masalah alur kerja. Sebelum menjalankan alur kerja, dapat memverifikasi alur kerja dengan melihat grafik desain alur kerja di konsol. AWS Glue

Lihat juga

- Mengembangkan cetak biru di AWS Glue
- Membuat alur kerja dari cetak biru di AWS Glue

- Izin untuk persona dan peran untuk cetak biru AWS Glue

Mengembangkan cetak biru di AWS Glue

Sebagai AWS Glue pengembang, Anda dapat membuat dan menerbitkan cetak biru yang dapat digunakan analis data untuk menghasilkan alur kerja.

Topik

- Ikhtisar pengembangan cetak biru
- Prasyarat untuk mengembangkan cetak biru
- Menulis kode cetak biru
- Contoh proyek cetak biru
- Menguji cetak biru
- Menerbitkan cetak biru
- AWS Gluereferensi kelas cetak biru
- Sampel cetak biru

Lihat juga

- Ikhtisar cetak biru di AWS Glue

Ikhtisar pengembangan cetak biru

Langkah pertama dalam proses pengembangan Anda adalah mengidentifikasi kasus penggunaan umum yang akan memanfaatkan sebuah cetak biru. Sebuah kasus penggunaan biasa melibatkan masalah ETL berulang yang Anda yakini harus diselesaikan secara umum. Berikutnya, rancang sebuah cetak biru yang mengimplementasikan kasus penggunaan umum, dan tentukan parameter masukan cetak biru yang bersama-sama dapat menentukan kasus penggunaan spesifik dari kasus penggunaan umum.

Sebuah cetak biru terdiri dari sebuah proyek yang berisi file konfigurasi parameter cetak biru dan skrip yang mendefinisikan tata letak alur kerja yang akan dibuat. Tata letak tersebut menentukan tugas dan crawler (atau entitas dalam terminologi skrip cetak biru) yang akan dibuat.

Anda tidak secara langsung menentukan pemicu apa pun dalam skrip tata letak tersebut. Sebagai gantinya, Anda menulis kode untuk menentukan dependensi antara pekerjaan dan crawler yang dibuat skrip. AWS Gluemenghasilkan pemicu berdasarkan spesifikasi ketergantungan Anda. Output dari skrip tata letak tersebut adalah sebuah objek alur kerja, yang berisi spesifikasi untuk semua entitas alur kerja.

Anda membangun objek alur kerja menggunakan pustaka AWS Glue cetak biru berikut:

- `awsglue.blueprint.base_resource` — Sebuah perpustakaan sumber daya dasar yang digunakan oleh perpustakaan tersebut.
- `awsglue.blueprint.workflow` — Sebuah perpustakaan untuk mendefinisikan kelas `Workflow`.
- `awsglue.blueprint.job` — Sebuah perpustakaan untuk mendefinisikan kelas `Job`.
- `awsglue.blueprint.crawler` — Sebuah perpustakaan untuk mendefinisikan kelas `Crawler`.

Satu-satunya perpustakaan lain yang didukung untuk pembuatan tata letak adalah perpustakaan yang tersedia untuk shell Python.

Sebelum menerbitkan cetak biru Anda, Anda dapat menggunakan metode yang didefinisikan dalam perpustakaan cetak biru untuk menguji cetak biru secara lokal.

Ketika Anda siap untuk membuat cetak biru yang tersedia untuk analis data, maka Anda mengemas skrip, file konfigurasi parameter, dan file pendukung apa pun, seperti skrip dan perpustakaan tambahan, menjadi satu aset yang dapat di-deploy. Anda kemudian mengunggah aset ke Amazon S3 dan meminta administrator untuk mendaftarkannya. AWS Glue

Untuk informasi tentang contoh proyek cetak biru lainnya, lihat Contoh proyek cetak biru dan Sampel cetak biru.

Prasyarat untuk mengembangkan cetak biru

Untuk mengembangkan cetak biru, Anda harus terbiasa menggunakan AWS Glue dan menulis skrip untuk pekerjaan Apache Spark ETL atau pekerjaan shell Python. Di samping itu, anda harus menyelesaikan tugas penyiapan berikut.

- Unduh empat perpustakaan Python AWS untuk digunakan dalam skrip tata letak cetak biru Anda.
- Siapkan AWS SDK.
- Menyiapkan AWS CLI.

Unduh pustaka Python

Unduh pustaka berikut dari GitHub, dan instal ke dalam proyek Anda:

- https://github.com/awslabs/ aws-glue-blueprint-libs /tree/master/awsglue/blueprint/base_resource.py
- https://github.com/awslabs/ aws-glue-blueprint-libs /tree/master/awsglue/blueprint/workflow.py
- https://github.com/awslabs/ aws-glue-blueprint-libs /tree/master/awsglue/blueprint/crawler.py
- https://github.com/awslabs/ aws-glue-blueprint-libs /tree/master/awsglue/blueprint/job.py

Siapkan AWS Java SDK

Untuk SDK Java AWS, Anda harus menambahkan file `jar` yang menyertakan API untuk cetak biru.

1. Jika Anda belum melakukannya, siapkan SDK for Java AWS.
   - Untuk Java 1.x, ikuti petunjuk di Menyiapkan AWS SDK for Java di Panduan Developer AWS SDK for Java.
   - Untuk Java 2.x, ikuti petunjuk di Menyiapkan AWS SDK for Java 2.x di Panduan Developer AWS SDK for Java 2.x.
2. Unduh `jar` file klien yang memiliki akses ke API untuk cetak biru.
   - Untuk Java 1.x: s3://-artifacts//awsglue-custom-blueprints-preview-1.11.x.jar awsglue-java-sdk-preview AWSGlueJavaClient
   - Untuk Java 2.x: s3://-artifacts/ 2-preview/ awsglue-custom-blueprints-preview -glue-2.0.jar awsglue-java-sdk-v AwsJavaSdk
3. Tambahkan klien `jar` ke bagian depan classpath Java untuk mengganti klien AWS Glue yang disediakan oleh Java SDKAWS.

   ```
   export CLASSPATH=<path-to-preview-client-jar>:$CLASSPATH
   ```
4. (Opsional) Uji SDK dengan aplikasi Java berikut. Aplikasi harus menampilkan sebuah daftar kosong.

   Ganti `accessKey` dan `secretKey` dengan kredensial Anda, dan ganti `us-east-1` dengan Wilayah Anda.

   ```
   import com.amazonaws.auth.AWSCredentials;
   ```

```
import com.amazonaws.auth.AWSCredentialsProvider;
import com.amazonaws.auth.AWSStaticCredentialsProvider;
import com.amazonaws.auth.BasicAWSCredentials;
import com.amazonaws.services.glue.AWSGlue;
import com.amazonaws.services.glue.AWSGlueClientBuilder;
import com.amazonaws.services.glue.model.ListBlueprintsRequest;

public class App{
    public static void main(String[] args) {
        AWSCredentials credentials = new BasicAWSCredentials("accessKey",
 "secretKey");
        AWSCredentialsProvider provider = new
 AWSStaticCredentialsProvider(credentials);
        AWSGlue glue = AWSGlueClientBuilder.standard().withCredentials(provider)
                .withRegion("us-east-1").build();
        ListBlueprintsRequest request = new
 ListBlueprintsRequest().withMaxResults(2);
        System.out.println(glue.listBlueprints(request));
    }
}
```

Siapkan SDK AWS Python

Langkah-langkah berikut dengan asumsi bahwa Anda memiliki Python versi 2.7 atau yang lebih baru, atau versi 3.6 atau yang lebih baru yang sudah diinstal pada komputer Anda.

1. Unduh file roda boto3 berikut. Jika diminta untuk membuka atau menyimpan, simpan file. s3://-artifacts/ awsglue-custom-blueprints-preview /boto3-1.17.31-py2.py3-none-any.whl aws-python-sdk-preview
2. Unduh file roda botocore berikut: s3://-artifacts/ /botocore-1.20.31-py2.py3-none-any.whl awsglue-custom-blueprints-preview aws-python-sdk-preview
3. Periksa versi Python anda.

   ```
   python --version
   ```

4. Tergantung pada versi Python Anda, masukkan perintah berikut (untuk Linux):

   - Untuk Python 2.7 atau yang lebih baru.

     ```
     python3 -m pip install --user virtualenv
     ```

```
source env/bin/activate
```

- Untuk Python 3.6 atau yang lebih baru.

```
python3 -m venv python-sdk-test
source python-sdk-test/bin/activate
```

5. Instal file roda botocore.

```
python3 -m pip install <download-directory>/botocore-1.20.31-py2.py3-none-any.whl
```

6. Instal file roda boto3.

```
python3 -m pip install <download-directory>/boto3-1.17.31-py2.py3-none-any.whl
```

7. Konfigurasi kredensial dan wilayah default Anda di `~/.aws/credentials` dan `~/.aws/config`. Untuk informasi lebih lanjut, lihat Mengonfigurasi AWS CLI di Panduan Pengguna AWS Command Line Interface.
8. (Opsional) Uji penyiapan Anda. Perintah berikut harus mengembalikan sebuah daftar kosong.

   Ganti `us-east-1` dengan Wilayah Anda.

```
$ python
>>> import boto3
>>> glue = boto3.client('glue', 'us-east-1')
>>> glue.list_blueprints()
```

Siapkan pratinjau AWS CLI

1. Jika Anda belum melakukannya, instal dan/atau perbarui AWS Command Line Interface (AWS CLI) di komputer Anda. Cara termudah untuk melakukannya adalah dengan `pip`, utilitas penginstal Python:

   ```
   pip install awscli --upgrade --user
   ```

   Anda dapat menemukan petunjuk instalasi lengkap untuk AWS CLI di sini: Menginstal AWS Command Line Interface.
2. Unduh file AWS CLI roda dari: s3://-artifacts/ awsglue-custom-blueprints-preview /awscli-1.19.31-py2.py3-none-any.whl awscli-preview-build

3. Instal file roda AWS CLI.

```
python3 -m pip install awscli-1.19.31-py2.py3-none-any.whl
```

4. Jalankan perintah `aws configure`. Konfigurasi kredensial AWS Anda (termasuk access key, dan kunci rahasia) dan Wilayah AWS. Anda dapat menemukan informasi tentang mengkonfigurasi AWS CLI di sini: Mengkonfigurasi AWS CLI.
5. Uji AWS CLI. Perintah berikut harus mengembalikan sebuah daftar kosong.

   Ganti `us-east-1` dengan Wilayah Anda.

```
aws glue list-blueprints --region us-east-1
```

Menulis kode cetak biru

Setiap proyek cetak biru yang Anda buat harus berisi paling tidak file-file berikut ini:

- Skrip tata letak Python yang mendefinisikan alur kerja. Skrip yang berisi fungsi yang mendefinisikan entitas (tugas dan crawler) dalam sebuah alur kerja, dan dependensi di antara mereka.
- File konfigurasi, `blueprint.cfg`, yang mendefinisikan:
  - Path lengkap dari fungsi definisi tata letak alur kerja.
  - Parameter yang diterima cetak biru.

Topik

- Membuat skrip tata letak cetak biru
- Membuat file konfigurasi
- Menentukan parameter cetak biru

Membuat skrip tata letak cetak biru

Skrip tata letak cetak biru harus menyertakan fungsi yang menghasilkan entitas dalam alur kerja Anda. Anda dapat memberi nama fungsi ini apa pun yang Anda suka. AWS Gluemenggunakan file konfigurasi untuk menentukan nama fungsi yang sepenuhnya memenuhi syarat.

Fungsi tata letak Anda melakukan hal-hal berikut:

- (Opsional) Instansiasi kelas `Job` untuk membuat objek `Job`, dan memberikan argumen seperti `Command` dan `Role`. Ini adalah properti pekerjaan yang akan Anda tentukan jika Anda membuat pekerjaan menggunakan AWS Glue konsol atau API.
- (Opsional) Instansiasi kelas `Crawler` untuk membuat objek `Crawler`, dan memberikan nama, peran, dan argumen target.
- Untuk menunjukkan dependensi antara objek (entitas alur kerja), berikan `DependsOn` dan argumen tambahan `WaitForDependencies` untuk `Job()` dan `Crawler()`. Argumen ini dijelaskan nanti dalam bagian ini.
- Membuat instance `Workflow` kelas untuk membuat objek alur kerja yang dikembalikan keAWS Glue, meneruskan argumen, `Name` argumen, dan `Entities` argumen opsional. `OnSchedule` Argumen `Entities` menentukan semua tugas dan crawler yang akan disertakan dalam alur kerja. Untuk melihat bagaimana membangun sebuah objek `Entities`, lihat contoh proyek dalam bagian ini nanti.
- Mengembalikan objek `Workflow`.

Untuk definisi `Job`, `Crawler`, dan kelas `Workflow`, lihat AWS Gluereferensi kelas cetak biru.

Fungsi tata letak harus menerima argumen-argumen masukan berikut.

Pendapat	Deskripsi
`user_params`	Kamus Python tentang nama parameter dan nilai-nilai cetak biru. Untuk informasi selengkapnya, lihat Menentukan parameter cetak biru.
`system_params`	Kamus Python berisi dua properti: `region` dan `accountId` .

Berikut ini adalah contoh skrip tata letak generator dalam sebuah file bernama `Layout.py`:

```
import argparse
import sys
import os
import json
from awsglue.blueprint.workflow import *
from awsglue.blueprint.job import *
from awsglue.blueprint.crawler import *
```

```
def generate_layout(user_params, system_params):

    etl_job = Job(Name="{}_etl_job".format(user_params['WorkflowName']),
                  Command={
                      "Name": "glueetl",
                      "ScriptLocation": user_params['ScriptLocation'],
                      "PythonVersion": "2"
                  },
                  Role=user_params['PassRole'])
    post_process_job = Job(Name="{}_post_process".format(user_params['WorkflowName']),
                           Command={
                               "Name": "pythonshell",
                               "ScriptLocation": user_params['ScriptLocation'],
                               "PythonVersion": "2"
                           },
                           Role=user_params['PassRole'],
                           DependsOn={
                               etl_job: "SUCCEEDED"
                           },
                           WaitForDependencies="AND")
    sample_workflow = Workflow(Name=user_params['WorkflowName'],
                           Entities=Entities(Jobs=[etl_job, post_process_job]))
    return sample_workflow
```

Skrip contoh mengimpor perpustakaan cetak biru yang diperlukan dan menyertakan fungsi `generate_layout` yang menghasilkan sebuah alur kerja dengan dua tugas. Ini adalah skrip yang sangat sederhana. Skrip yang lebih kompleks dapat menggunakan logika dan parameter tambahan untuk menghasilkan sebuah alur kerja dengan banyak tugas dan crawler, atau bahkan sejumlah variabel tugas dan crawler.

Menggunakan DependsOn argumen

Argumen `DependsOn` adalah sebuah representasi kamus dependensi yang dimiliki entitas ini atas entitas lain dalam alur kerja. Ia memiliki bentuk berikut.

```
DependsOn = {dependency1 : state, dependency2 : state, ...}
```

Kunci dalam kamus ini mewakili referensi objek, bukan nama, entitas, sedangkan nilainya adalah string yang sesuai dengan status yang harus diperhatikan. AWS Gluemenyimpulkan pemicu yang tepat. Untuk status yang valid, lihat Struktur Kondisi.

Sebagai contoh, sebuah tugas mungkin bergantung pada keberhasilan penyelesaian sebuah crawler. Jika anda menentukan sebuah objek crawler bernama `crawler2` sebagai berikut:

```
crawler2 = Crawler(Name="my_crawler", ...)
```

Maka sebuah objek yang tergantung pada `crawler2` akan mencakup argumen konstruktor seperti:

```
DependsOn = {crawler2 : "SUCCEEDED"}
```

Misalnya:

```
job1 = Job(Name="Job1", ..., DependsOn = {crawler2 : "SUCCEEDED", ...})
```

Jika `DependsOn` dihilangkan dan diganti sebuah entitas, maka entitas tersebut tergantung pada pemicu awal alur kerja.

Menggunakan WaitForDependencies argumen

Argumen `WaitForDependencies` mendefinisikan apakah sebuah entitas tugas atau crawler harus menunggu sampai semua entitas yang tergantung selesai atau sampai salah satu selesai.

Nilai yang diijinkan adalah "`AND`" atau "`ANY`".

Menggunakan OnSchedule argumen

Argumen `OnSchedule` untuk konstruktor kelas `Workflow` adalah sebuah ekspresi `cron` yang menentukan definisi pemicu awal untuk alur kerja.

Jika argumen ini ditentukan, AWS Glue membuat pemicu jadwal dengan jadwal yang sesuai. Jika tidak ditentukan, maka pemicu awal untuk alur kerja adalah pemicu sesuai permintaan.

Membuat file konfigurasi

File konfigurasi cetak biru adalah sebuah file yang diperlukan yang menentukan titik entri skrip untuk menghasilkan alur kerja, dan parameter yang diterima cetak biru. File tersebut harus diberi nama `blueprint.cfg`.

Berikut ini adalah file konfigurasi sampel.

```
{
    "layoutGenerator": "DemoBlueprintProject.Layout.generate_layout",
    "parameterSpec" : {
```

```
            "WorkflowName" : {
                 "type": "String",
                 "collection": false
            },
            "WorkerType" : {
                 "type": "String",
                 "collection": false,
                 "allowedValues": ["G1.X", "G2.X"],
                 "defaultValue": "G1.X"
            },
            "Dpu" : {
                 "type" : "Integer",
                 "allowedValues" : [2, 4, 6],
                 "defaultValue" : 2
            },
            "DynamoDBTableName": {
                 "type": "String",
                 "collection" : false
            },
            "ScriptLocation" : {
                 "type": "String",
                 "collection": false
     }
    }
}
```

Properti `layoutGenerator` menentukan nama memenuhi syarat fungsi dalam skrip yang menghasilkan tata letak.

Properti `parameterSpec` menentukan parameter yang diterima cetak biru ini. Untuk informasi selengkapnya, lihat Menentukan parameter cetak biru.

> **Important**
>
> File konfigurasi Anda harus menyertakan nama alur kerja sebagai sebuah parameter cetak biru, atau Anda harus membuat nama alur kerja yang unik dalam skrip tata letak Anda.

Menentukan parameter cetak biru

File konfigurasi berisi spesifikasi parameter cetak biru dalam objek JSON `parameterSpec`. `parameterSpec` berisi satu atau beberapa objek parameter.

```
"parameterSpec": {
    "<parameter_name>": {
      "type": "<parameter-type>",
      "collection": true|false,
      "description": "<parameter-description>",
      "defaultValue": "<default value for the parameter if value not specified>"
      "allowedValues": "<list of allowed values>"
    },
    "<parameter_name>": {
       ...
    }
  }
```

Berikut ini adalah aturan untuk melakukan coding pada setiap objek parameter:

- Nama parameter dan `type` adalah wajib. Semua properti lainnya opsional.
- Jika Anda menentukan properti `defaultValue`, maka parameter bersifat opsional. Jika tidak, parameter tersebut bersifat wajib dan analis data yang menciptakan alur kerja dari cetak biru tersebut harus memberikan nilai untuk itu.
- Jika Anda mengatur properti `collection` ke `true`, maka parameter dapat mengambil koleksi nilai-nilai. Koleksi tersebut dapat berupa jenis data apa pun.
- Jika Anda menentukan`allowedValues`, AWS Glue konsol akan menampilkan daftar tarik-turun nilai untuk dipilih analis data saat membuat alur kerja dari cetak biru.

Berikut ini adalah nilai yang diizinkan untuk `type`:

Jenis data parameter	Catatan
`String`	-
`Integer`	-
`Double`	-
`Boolean`	Kemungkinan nilainya adalah `true` and `false`. Menghasilkan kotak centang pada Buat alur kerja dari <blueprint>halaman di AWS Glue konsol.

Jenis data parameter	Catatan
`S3Uri`	Lengkapi path Amazon S3, yang dimulai dengan `s3://`. Buat bidang teks dan tombol Jelajahi pada halaman Membuat alur kerja dari <blueprint>.
`S3Bucket`	Nama bucket Amazon S3 saja. Buat pemilih bucket pada halaman Membuat alur kerja dari <blueprint>.
`IAMRoleArn`	Amazon Resource Name (ARN) dari sebuah AWS Identity and Access Management (IAM) role. Buat pemilih peran pada halaman Membuat alur kerja dari <blueprint>.
`IAMRoleName`	Nama dari sebuah IAM role. Buat pemilih peran pada halaman Membuat alur kerja dari <blueprint>.

Contoh proyek cetak biru

Konversi format data adalah kasus penggunaan extract, transform, and load (ETL) yang sering terjadi. Pada beban kerja analitik yang khas, format file berbasis kolom seperti Parquet atau ORC lebih disukai daripada format teks seperti CSV atau JSON. Sampel cetak biru ini memungkinkan Anda untuk mengkonversi data dari CSV/JSON/dll. menjadi Parquet untuk file di Amazon S3.

Cetak biru ini mengambil daftar path S3 yang didefinisikan oleh parameter cetak biru, mengkonversi data ke format Parquet, dan menulis ke lokasi S3 yang ditentukan oleh parameter cetak biru lain. Skrip tata letak menciptakan sebuah crawler dan tugas untuk setiap path. Skrip tata letak juga mengunggah skrip ETL di `Conversion.py` ke bucket S3 yang ditentukan oleh parameter cetak biru lain. Skrip tata letak kemudian menentukan skrip yang sudah diunggah sebagai skrip ETL untuk setiap tugas. Arsip ZIP untuk proyek berisi skrip tata letak, skrip ETL, dan file konfigurasi cetak biru.

Untuk informasi tentang contoh proyek cetak biru lainnya, lihat Sampel cetak biru.

Berikut ini adalah skrip tata letak, dalam file `Layout.py`.

```
from awsglue.blueprint.workflow import *
from awsglue.blueprint.job import *
from awsglue.blueprint.crawler import *
import boto3
```

```
s3_client = boto3.client('s3')

# Ingesting all the S3 paths as Glue table in parquet format
def generate_layout(user_params, system_params):
    #Always give the full path for the file
    with open("ConversionBlueprint/Conversion.py", "rb") as f:
        s3_client.upload_fileobj(f, user_params['ScriptsBucket'], "Conversion.py")
    etlScriptLocation = "s3://{}/Conversion.py".format(user_params['ScriptsBucket'])

    crawlers = []
    jobs = []
    workflowName = user_params['WorkflowName']
    for path in user_params['S3Paths']:
      tablePrefix = "source_"
      crawler = Crawler(Name="{}_crawler".format(workflowName),
                        Role=user_params['PassRole'],
                        DatabaseName=user_params['TargetDatabase'],
                        TablePrefix=tablePrefix,
                        Targets= {"S3Targets": [{"Path": path}]})
      crawlers.append(crawler)
      transform_job = Job(Name="{}_transform_job".format(workflowName),
                          Command={"Name": "glueetl",
                                   "ScriptLocation": etlScriptLocation,
                                   "PythonVersion": "3"},
                          Role=user_params['PassRole'],
                          DefaultArguments={"--database_name":
 user_params['TargetDatabase'],
                                            "--table_prefix": tablePrefix,
                                            "--region_name": system_params['region'],
                                            "--output_path":
 user_params['TargetS3Location']},
                          DependsOn={crawler: "SUCCEEDED"},
                          WaitForDependencies="AND")
      jobs.append(transform_job)
    conversion_workflow = Workflow(Name=workflowName, Entities=Entities(Jobs=jobs,
 Crawlers=crawlers))
    return conversion_workflow
```

Berikut ini adalah file `blueprint.cfg` konfigurasi cetak biru yang sesuai.

```
{
    "layoutGenerator": "ConversionBlueprint.Layout.generate_layout",
    "parameterSpec" : {
```

```
        "WorkflowName" : {
            "type": "String",
            "collection": false,
            "description": "Name for the workflow."
        },
        "S3Paths" : {
            "type": "S3Uri",
            "collection": true,
            "description": "List of Amazon S3 paths for data ingestion."
        },
        "PassRole" : {
            "type": "IAMRoleName",
            "collection": false,
            "description": "Choose an IAM role to be used in running the job/crawler"
        },
        "TargetDatabase": {
            "type": "String",
            "collection" : false,
            "description": "Choose a database in the Data Catalog."
        },
        "TargetS3Location": {
            "type": "S3Uri",
            "collection" : false,
            "description": "Choose an Amazon S3 output path: ex:s3://<target_path>/."
        },
        "ScriptsBucket": {
            "type": "S3Bucket",
            "collection": false,
            "description": "Provide an S3 bucket name(in the same AWS Region) to store
 the scripts."
        }
    }
}
```

Skrip yang ada dalam file `Conversion.py` berikut adalah skrip ETL yang telah diunggah. Perhatikan bahwa ia mempertahankan skema partisi selama konversi.

```
import sys
from pyspark.sql.functions import *
from pyspark.context import SparkContext
from awsglue.transforms import *
from awsglue.context import GlueContext
from awsglue.job import Job
```

```
from awsglue.utils import getResolvedOptions
import boto3

args = getResolvedOptions(sys.argv, [
    'JOB_NAME',
    'region_name',
    'database_name',
    'table_prefix',
    'output_path'])
databaseName = args['database_name']
tablePrefix = args['table_prefix']
outputPath = args['output_path']

glue = boto3.client('glue', region_name=args['region_name'])

glue_context = GlueContext(SparkContext.getOrCreate())
spark = glue_context.spark_session
job = Job(glue_context)
job.init(args['JOB_NAME'], args)

def get_tables(database_name, table_prefix):
    tables = []
    paginator = glue.get_paginator('get_tables')
    for page in paginator.paginate(DatabaseName=database_name, Expression=table_prefix
+"*"):
        tables.extend(page['TableList'])
    return tables

for table in get_tables(databaseName, tablePrefix):
    tableName = table['Name']
    partitionList = table['PartitionKeys']
    partitionKeys = []
    for partition in partitionList:
        partitionKeys.append(partition['Name'])

    # Create DynamicFrame from Catalog
    dyf = glue_context.create_dynamic_frame.from_catalog(
        name_space=databaseName,
        table_name=tableName,
        additional_options={
            'useS3ListImplementation': True
        },
        transformation_ctx='dyf'
    )
```

```
    # Resolve choice type with make_struct
    dyf = ResolveChoice.apply(
        frame=dyf,
        choice='make_struct',
        transformation_ctx='resolvechoice_' + tableName
    )

    # Drop null fields
    dyf = DropNullFields.apply(
        frame=dyf,
        transformation_ctx="dropnullfields_" + tableName
    )

    # Write DynamicFrame to S3 in glueparquet
    sink = glue_context.getSink(
        connection_type="s3",
        path=outputPath,
        enableUpdateCatalog=True,
        partitionKeys=partitionKeys
    )
    sink.setFormat("glueparquet")

    sink.setCatalogInfo(
        catalogDatabase=databaseName,
        catalogTableName=tableName[len(tablePrefix):]
    )
    sink.writeFrame(dyf)

job.commit()
```

> **Note**
>
> Hanya dua path Amazon S3 yang dapat diberikan sebagai masukan untuk cetak biru sampel. Ini karena AWS Glue pemicu terbatas untuk memanggil hanya dua tindakan crawler.

Menguji cetak biru

Sementara Anda mengembangkan kode Anda, Anda harus melakukan pengujian secara lokal untuk memverifikasi bahwa tata letak alur kerja sudah benar.

Pengujian lokal tidak menghasilkan AWS Glue pekerjaan, crawler, atau pemicu. Sebaliknya, Anda jalankan skrip tata letak secara lokal dan gunakan metode `to_json()` dan `validate()` untuk mencetak objek dan menemukan kesalahan. Metode ini tersedia di semua tiga kelas yang ditentukan dalam perpustakaan.

Ada dua cara untuk menangani `user_params` dan `system_params` argumen yang AWS Glue lolos ke fungsi tata letak Anda. Kode test-bench Anda dapat membuat sebuah kamus sampel nilai parameter cetak biru dan memberikannya untuk fungsi tata letak sebagai argumen `user_params`. Atau, Anda dapat menghapus referensi ke `user_params` dan menggantinya dengan string yang di-hardcoding.

Jika kode Anda menggunakan properti `region` dan `accountId` di `system_params`, Anda dapat memberikan dalam kamus Anda sendiri untuk `system_params`.

Untuk menguji sebuah cetak biru

1. Mulai sebuah interpreter Python dalam direktori yang memiliki perpustakaan, atau muat file cetak biru dan perpustakaan yang disediakan ke lingkungan pengembangan terpadu (IDE) pilihan Anda.
2. Pastikan bahwa kode Anda mengimpor perpustakaan yang disediakan.
3. Tambahkan kode ke fungsi tata letak Anda untuk memanggil `validate()` atau `to_json()` pada entitas apa pun atau objek `Workflow`. Sebagai contoh, jika kode Anda membuat objek `Crawler` bernama `mycrawler`, maka Anda dapat memanggil `validate()` sebagai berikut.

   ```
   mycrawler.validate()
   ```

   Anda dapat mencetak `mycrawler` sebagai berikut:

   ```
   print(mycrawler.to_json())
   ```

   Jika Anda memanggil `to_json` pada sebuah objek, maka Anda tidak perlu juga memanggil `validate()`, karena `to_json()` memanggil `validate()`.

   Hal ini akan sangat berguna untuk memanggil metode ini pada objek alur kerja. Dengan asumsi bahwa skrip Anda memberi nama objek alur kerja `my_workflow`, validasi dan cetak objek alur kerja sebagai berikut.

   ```
   print(my_workflow.to_json())
   ```

Untuk informasi selengkapnya tentang `to_json()` dan `validate()`, lihat Metode kelas.

Anda juga dapat mengimpor `pprint` dan melakukan pretty-print pada objek alur kerja, seperti yang ditunjukkan pada contoh nanti di bagian ini.

4. Jalankan kode, perbaiki kesalahan, dan akhirnya hapus setiap panggilan ke `validate()` atau `to_json()`.

Example

Contoh berikut menunjukkan cara membangun kamus parameter cetak biru sampel dan menyebarkannya sebagai argumen `user_params` ke fungsi tata letak `generate_compaction_workflow`. Hal ini juga menunjukkan cara melakukan pretty-print pada objek alur kerja yang dihasilkan.

```
from pprint import pprint
from awsglue.blueprint.workflow import *
from awsglue.blueprint.job import *
from awsglue.blueprint.crawler import *

USER_PARAMS = {"WorkflowName": "compaction_workflow",
               "ScriptLocation": "s3://awsexamplebucket1/scripts/threaded-
compaction.py",
               "PassRole": "arn:aws:iam::111122223333:role/GlueRole-ETL",
               "DatabaseName": "cloudtrial",
               "TableName": "ct_cloudtrail",
               "CoalesceFactor": 4,
               "MaxThreadWorkers": 200}


def generate_compaction_workflow(user_params: dict, system_params: dict) -> Workflow:
    compaction_job = Job(Name=f"{user_params['WorkflowName']}_etl_job",
                         Command={"Name": "glueetl",
                                  "ScriptLocation": user_params['ScriptLocation'],
                                  "PythonVersion": "3"},
                         Role="arn:aws:iam::111122223333:role/
AWSGlueServiceRoleDefault",
                         DefaultArguments={"DatabaseName": user_params['DatabaseName'],
                                           "TableName": user_params['TableName'],
                                           "CoalesceFactor":
 user_params['CoalesceFactor'],
```

```
                                        "max_thread_workers":
 user_params['MaxThreadWorkers']})

    catalog_target = {"CatalogTargets": [{"DatabaseName": user_params['DatabaseName'],
 "Tables": [user_params['TableName']]}]}

    compacted_files_crawler = Crawler(Name=f"{user_params['WorkflowName']}_post_crawl",
                                      Targets = catalog_target,
                                      Role=user_params['PassRole'],
                                      DependsOn={compaction_job: "SUCCEEDED"},
                                      WaitForDependencies="AND",
                                      SchemaChangePolicy={"DeleteBehavior": "LOG"})

    compaction_workflow = Workflow(Name=user_params['WorkflowName'],
                                   Entities=Entities(Jobs=[compaction_job],

 Crawlers=[compacted_files_crawler]))
    return compaction_workflow

generated = generate_compaction_workflow(user_params=USER_PARAMS, system_params={})
gen_dict = generated.to_json()

pprint(gen_dict)
```

Menerbitkan cetak biru

Setelah Anda mengembangkan sebuah cetak biru, Anda harus mengunggahnya ke Amazon S3. Anda harus memiliki izin tulis pada bucket Amazon S3 yang Anda gunakan untuk menerbitkan cetak biru. Anda juga harus memastikan bahwa AWS Glue administrator, yang akan mendaftarkan cetak biru, telah membaca akses ke bucket Amazon S3. Untuk kebijakan izin AWS Identity and Access Management (IAM) yang disarankan untuk persona dan peran untuk AWS Glue cetak biru, lihat. Izin untuk persona dan peran untuk cetak biru AWS Glue

Untuk menerbitkan sebuah cetak biru

1. Membuat skrip, sumber daya, dan file konfigurasi cetak biru yang diperlukan.
2. Tambahkan semua file ke arsip ZIP dan unggah file ZIP ke Amazon S3. Gunakan sebuah bucket S3 yang berada di Wilayah yang sama dengan Wilayah dimana pengguna akan mendaftar dan menjalankan cetak biru tersebut.

   Anda dapat membuat file ZIP dari baris perintah dengan menggunakan perintah berikut.

```
zip -r folder.zip folder
```

3. Tambahkan kebijakan bucket yang memberikan izin baca untuk akun AWS yang diinginkan. Berikut ini adalah contoh outputnya.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-blueprints/*"
        }
    ]
}
```

4. Berikan `s3:GetObject` izin IAM di bucket Amazon S3 kepada administrator atau kepada siapa pun AWS Glue yang akan mendaftarkan cetak biru. Untuk contoh kebijakan untuk memberikan izin kepada administrator, lihat AWS Glueizin administrator untuk cetak biru.

Setelah Anda menyelesaikan pengujian lokal cetak biru Anda, Anda mungkin juga ingin menguji cetak biru. AWS Glue Untuk menguji cetak biruAWS Glue, itu harus terdaftar. Anda dapat membatasi siapa yang melihat cetak biru terdaftar dengan menggunakan otorisasi IAM, atau dengan menggunakan akun pengujian terpisah.

Lihat juga:

- Mendaftarkan cetak biru di AWS Glue

AWS Gluereferensi kelas cetak biru

Pustaka untuk AWS Glue cetak biru menentukan tiga kelas yang Anda gunakan dalam skrip tata letak alur kerja Anda:,, dan. `Job` `Crawler` `Workflow`

Topik

- Kelas Job
- Kelas crawler
- Kelas alur kerja
- Metode kelas

Kelas Job

`Job`Kelas mewakili pekerjaan AWS Glue ETL.

Argumen konstruktor wajib

Berikut ini adalah argumen konstruktor wajib untuk kelas `Job`.

Nama argumen	Tipe	Deskripsi
`Name`	`str`	Nama untuk ditugaskan ke pekerjaan. AWS Gluemenambahkan akhiran yang dihasilka n secara acak ke nama untuk membedakan pekerjaan dari yang dibuat oleh blueprint run lainnya.
`Role`	`str`	Amazon Resource Name (ARN) dari peran yang harus diambil oleh tugas tersebut saat mengeksekusi.
`Command`	`dict`	Perintah tugas, sebagaimana yang ditentukan pada JobCommand struktur dalam dokumentasi API.

Argumen konstruktor opsional

Berikut ini adalah argumen konstruktor opsional untuk kelas `Job`.

Nama argumen	Tipe	Deskripsi
`DependsOn`	`dict`	Daftar entitas alur kerja yang padanya tugas bergantung. Untuk informasi selengkapnya, lihat Menggunakan DependsOn argumen.
`WaitForDependencies`	`str`	Menunjukkan apakah tugas harus menunggu sampai semua entitas yang padanya ia bergantung selesai sebelum mengeksekusi atau sampai salah satu selesai. Untuk informasi selengkapnya, lihat Menggunakan WaitForDependencies argumen. Abaikan jika tugas tergantung pada hanya satu entitas saja.
(Properti tugas)	-	Properti pekerjaan apa pun yang tercantum Struktur Job dalam dokumentasi AWS Glue API (kecuali `CreatedOn` dan `LastModifiedOn`).

Kelas crawler

`Crawler`Kelas mewakili AWS Glue crawler.

Argumen konstruktor wajib

Berikut ini adalah argumen konstruktor wajib untuk kelas `Crawler`.

Nama argumen	Tipe	Deskripsi
`Name`	`str`	Nama untuk ditetapkan ke crawler. AWS Gluemenambahkan akhiran yang dihasilkan secara acak ke nama untuk membedakan crawler dari yang dibuat oleh blueprint run lainnya.
`Role`	`str`	ARN dari peran yang harus diambil crawler saat berjalan.

Nama argumen	Tipe	Deskripsi
`Targets`	`dict`	Koleksi target yang harus di-crawling. Argumen konstruktor kelas `Targets` ditentukan dalam CrawlerTargets struktur dalam dokumentasi API. Semua argumen konstruktor `Targets` bersifat opsional, tetapi Anda harus memberikan setidaknya satu argumen.

Argumen konstruktor opsional

Berikut ini adalah argumen konstruktor opsional untuk kelas `Crawler`.

Nama argumen	Tipe	Deskripsi
`DependsOn`	`dict`	Daftar entitas alur kerja yang padanya crawler bergantung. Untuk informasi selengkapnya, lihat Menggunakan DependsOn argumen.
`WaitForDependencies`	`str`	Menunjukkan apakah crawler harus menunggu sampai semua entitas yang padanya ia bergantung selesai sebelum berjalan atau sampai salah satu selesai. Untuk informasi selengkapnya, lihat Menggunakan WaitForDependencies argumen. Abaikan jika crawler tergantung hanya pada satu entitas saja.
(Properti Crawler)	-	Properti crawler apa pun yang tercantum Struktur perayap dalam dokumentasi AWS Glue API, dengan pengecualian berikut: • `State` • `CrawlElapsedTime` • `CreationTime` • `LastUpdated` • `LastCrawl`

Nama argumen	Tipe	Deskripsi
		• `Version`

Kelas alur kerja

`Workflow`Kelas mewakili AWS Glue alur kerja. Skrip tata letak alur kerja mengembalikan `Workflow` objek. AWS Gluemembuat alur kerja berdasarkan objek ini.

Argumen konstruktor wajib

Berikut ini adalah argumen konstruktor wajib untuk kelas `Workflow`.

Nama argumen	Tipe	Deskripsi
`Name`	`str`	Nama yang akan ditetapkan untuk alur kerja tersebut.
`Entities`	`Entities`	Koleksi entitas (tugas dan crawler) yang akan disertakan dalam alur kerja. Kelas konstruktor `Entities` menerima sebuah argumen `Jobs`, yang merupakan daftar dari objek `Job`, dan `Crawlers`, yang merupakan daftar dari objek `Crawler`.

Argumen konstruktor opsional

Berikut ini adalah argumen konstruktor opsional untuk kelas `Workflow`.

Nama argumen	Tipe	Deskripsi
`Description`	`str`	Lihat Struktur alur kerja.
`DefaultRunProperties`	`dict`	Lihat Struktur alur kerja.
`OnSchedule`	`str`	Sebuah ekspresi `cron`.

Metode kelas

Ketiga kelas tersebut mencakup metode-metode berikut.

validate()

Memvalidasi properti objek dan apakah ada kesalahan yang ditemukan, membuat keluaran pesan dan menutup. Tidak menghasilkan output jika tidak ada kesalahan. Untuk kelas `Workflow`, memanggil dirinya sendiri pada setiap entitas dalam alur kerja.

to_json()

Melakukan serialisasi pada objek untuk JSON. Juga memanggil `validate()`. Untuk kelas `Workflow`, objek JSON termasuk tugas dan daftar crawler, serta daftar pemicu yang dihasilkan oleh spesifikasi dependensi tugas dan crawler.

Sampel cetak biru

Ada sejumlah proyek cetak biru sampel yang tersedia di repositori cetak AWS Gluebiru Github. Sampel ini hanya untuk referensi saja dan tidak dimaksudkan untuk penggunaan produksi.

Judul dari proyek sampel tersebut adalah:

- Perapatan: cetak biru ini menciptakan sebuah tugas yang merapatkan file input menjadi potongan yang lebih besar berdasarkan ukuran file yang diinginkan.
- Konversi: cetak biru ini mengubah file input dalam berbagai format file standar menjadi format Apache Parquet, yang dioptimalkan untuk beban kerja analitik.
- Melakukan crawling pada lokasi Amazon S3: cetak biru ini melakukan crawling pada beberapa lokasi Amazon S3 untuk menambahkan tabel metadata ke Katalog Data.
- Koneksi khusus ke Katalog Data: cetak biru ini mengakses penyimpanan data menggunakan konektor AWS Glue khusus, membaca catatan, dan mengisi definisi tabel di Katalog Data AWS Glue berdasarkan skema rekaman.
- Pengkodean: cetak biru ini mengubah file non-UTF Anda menjadi file UTF yang sudah dikodekan.
- Partisi: cetak biru ini menciptakan tugas pemartisian yang menempatkan file output ke partisi-partisi berdasarkan kunci partisi tertentu.
- Mengimpor data Amazon S3 ke dalam tabel DynamoDB: cetak biru ini mengimpor data dari Amazon S3 ke dalam sebuah tabel DynamoDB.

- Tabel standar yang diatur: cetak biru ini mengimpor tabel Katalog Data AWS Glue ke dalam tabel Lake Formation.

Mendaftarkan cetak biru di AWS Glue

Setelah AWS Glue pengembang mengkodekan cetak biru dan mengunggah arsip ZIP ke Amazon Simple Storage Service (Amazon S3) Simple Storage Service (Amazon S3), administrator harus mendaftarkan cetak biru. AWS Glue Mendaftarkan cetak biru akan membuatnya siap tersedia untuk digunakan.

Saat Anda mendaftarkan cetak biru, salin arsip cetak biru AWS Glue ke lokasi Amazon S3 yang dipesan. Anda kemudian dapat menghapus arsip tersebut dari lokasi pengunggahan.

Untuk mendaftarkan sebuah cetak biru, Anda perlu izin baca di lokasi Amazon S3 yang berisi arsip yang telah diunggah. Anda juga memerlukan izin AWS Identity and Access Management (IAM) `glue:CreateBlueprint`. Untuk izin yang disarankan untuk AWS Glue administrator yang harus mendaftar, melihat, dan memelihara cetak biru, lihat. AWS Glueizin administrator untuk cetak biru

Anda dapat mendaftarkan cetak biru menggunakan AWS Glue konsol, AWS Glue API, atau AWS Command Line Interface (). AWS CLI

Untuk mendaftarkan sebuah cetak biru (konsol)

1. Pastikan bahwa Anda memiliki izin baca (`s3:GetObject`) pada arsip ZIP cetak biru di Amazon S3.
2. Buka konsol AWS Glue di https://console.aws.amazon.com/glue/.

   Masuk sebagai pengguna yang memiliki izin untuk mendaftarkan sebuah cetak biru. Beralih ke Wilayah AWS yang sama sebagai bucket Amazon S3 yang berisi arsip ZIP cetak biru.
3. Di panel navigasi, pilih cetak biru. Kemudian pada halaman cetak biru, pilih Tambahkan cetak biru.
4. Masukkan nama dan deskripsi opsional untuk cetak biru.
5. Untuk Lokasi arsip ZIP (S3), masukkan path Amazon S3 dari arsip ZIP cetak biru yang telah diunggah. Sertakan nama file arsip di path dan mulai path dengan `s3://`.
6. (Opsional) Tambahkan satu tag satu atau beberapa tag.
7. Pilih Tambahkan cetak biru.

Halaman cetak biru mengembalikan dan menunjukkan bahwa status cetak biru adalah. `CREATING` Pilih tombol refresh hingga status berubah menjadi `ACTIVE` atau `FAILED`.

8. Jika statusnya adalah `FAILED`, pilih cetak biru, dan pada menu Tindakan, pilih Lihat.

   Halaman detail menunjukkan alasan kegagalan. Jika pesan kesalahan "Tidak dapat mengakses objek di lokasi..." atau "Akses ditolak pada objek di lokasi...", tinjau persyaratan berikut:

   - Pengguna yang Anda gunakan masuk harus memiliki izin baca pada arsip ZIP cetak biru di Amazon S3.
   - Bucket Amazon S3 yang berisi arsip ZIP harus memiliki kebijakan bucket yang memberikan izin baca pada objek untuk ID akun AWS Anda. Untuk informasi selengkapnya, lihat Mengembangkan cetak biru di AWS Glue.
   - Bucket Amazon S3 yang Anda gunakan harus berada dalam Wilayah yang sama dengan Wilayah yang Anda masuki di konsol tersebut.

9. Pastikan bahwa analis data memiliki izin pada cetak biru tersebut.

   Kebijakan IAM yang disarankan bagi analisis data ditunjukkan dalam Izin analis data untuk cetak biru. Kebijakan ini memberikan `glue:GetBlueprint` pada sumber daya apa pun. Jika kebijakan Anda lebih detail pada tingkat sumber daya, maka berikan izin analis data pada sumber daya yang baru dibuat ini.

Untuk mendaftarkan sebuah cetak biru (CLI AWS)

1. Masukkan perintah berikut.

```
aws glue create-blueprint --name <blueprint-name> [--description <description>] --
blueprint-location s3://<s3-path>/<archive-filename>
```

2. Masukkan perintah berikut untuk memeriksa status cetak biru. Ulangi perintah sampai status menjadi `ACTIVE` atau `FAILED`.

```
aws glue get-blueprint --name <blueprint-name>
```

   Jika statusnya adalah `FAILED` dan pesan kesalahannya adalah "Tidak dapat mengakses objek di lokasi..." atau "Akses ditolak pada objek di lokasi...", tinjau persyaratan berikut:

- Pengguna yang Anda gunakan masuk harus memiliki izin baca pada arsip ZIP cetak biru di Amazon S3.
- Bucket Amazon S3 yang berisi arsip ZIP harus memiliki kebijakan bucket yang memberikan izin baca pada objek untuk ID akun AWS Anda. Untuk informasi selengkapnya, lihat Menerbitkan cetak biru.
- Bucket Amazon S3 yang Anda gunakan harus berada dalam Wilayah yang sama dengan Wilayah yang Anda masuki di konsol tersebut.

Lihat juga:

- Ikhtisar cetak biru di AWS Glue

Melihat cetak biru di AWS Glue

Melihat cetak biru untuk meninjau deskripsi cetak biru, status, dan spesifikasi parameter, dan untuk mengunduh arsip ZIP tempat cetak biru.

Anda dapat melihat cetak biru menggunakan AWS Glue konsol, AWS Glue API, atau AWS Command Line Interface (). AWS CLI

Untuk melihat cetak biru (konsol)

1. Buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pilih cetak biru.
3. Pada halaman cetak biru, pilih cetak biru. Di menu Tindakan, pilih Lihat.

Untuk melihat sebuah cetak biru (CLI AWS)

- Masukkan perintah berikut untuk melihat nama cetak biru saja, deskripsi, dan status saja. Ganti *<blueprint-name>* dengan nama cetak biru untuk yang akan dilihat.

```
aws glue get-blueprint --name <blueprint-name>
```

Output perintah tersebut terlihat seperti berikut ini.

```
{
    "Blueprint": {
        "Name": "myDemoBP",
        "CreatedOn": 1587414516.92,
        "LastModifiedOn": 1587428838.671,
        "BlueprintLocation": "s3://awsexamplebucket1/demo/
DemoBlueprintProject.zip",
        "Status": "ACTIVE"
    }
}
```

Masukkan perintah berikut untuk juga melihat spesifikasi parameter.

```
aws glue get-blueprint --name <blueprint-name>  --include-parameter-spec
```

Output perintah tersebut terlihat seperti berikut ini.

```
{
    "Blueprint": {
        "Name": "myDemoBP",
        "CreatedOn": 1587414516.92,
        "LastModifiedOn": 1587428838.671,
        "ParameterSpec": "{\"WorkflowName\":{\"type\":\"String\",\"collection
\":false,\"description\":null,\"defaultValue\":null,\"allowedValues\":null},
\"PassRole\":{\"type\":\"String\",\"collection\":false,\"description\":null,
\"defaultValue\":null,\"allowedValues\":null},\"DynamoDBTableName\":{\"type
\":\"String\",\"collection\":false,\"description\":null,\"defaultValue\":null,
\"allowedValues\":null},\"ScriptLocation\":{\"type\":\"String\",\"collection
\":false,\"description\":null,\"defaultValue\":null,\"allowedValues\":null}}",
        "BlueprintLocation": "s3://awsexamplebucket1/demo/
DemoBlueprintProject.zip",
        "Status": "ACTIVE"
    }
}
```

Tambahkan `--include-blueprint` argumen untuk menyertakan URL dalam output yang dapat Anda tempel ke browser Anda untuk mengunduh arsip cetak biru ZIP yang disimpan. AWS Glue

Lihat juga:

- Ikhtisar cetak biru di AWS Glue

Memperbarui cetak biru di AWS Glue

Anda dapat memperbarui sebuah cetak biru jika Anda memiliki skrip tata letak yang direvisi, serangkaian parameter cetak biru yang direvisi, atau file pendukung yang direvisi. Memperbarui cetak biru akan menciptakan cetak biru versi baru.

Memperbarui cetak biru tidak akan memengaruhi alur kerja yang ada yang dibuat dari cetak biru tersebut.

Anda dapat memperbarui cetak biru menggunakan AWS Glue konsol, AWS Glue API, atau AWS Command Line Interface (). AWS CLI

Prosedur berikut mengasumsikan bahwa AWS Glue pengembang telah membuat dan mengunggah arsip ZIP cetak biru yang diperbarui ke Amazon S3.

Untuk memperbarui sebuah cetak biru (konsol)

1. Pastikan bahwa Anda memiliki izin baca (`s3:GetObject`) pada arsip ZIP cetak biru di Amazon S3.
2. Buka konsol AWS Glue di https://console.aws.amazon.com/glue/.

   Masuk sebagai pengguna yang memiliki izin untuk memperbarui sebuah cetak biru. Beralih ke Wilayah AWS yang sama sebagai bucket Amazon S3 yang berisi arsip ZIP cetak biru.
3. Di panel navigasi, pilih cetak biru.
4. Pada halaman cetak biru, pilih cetak biru, dan pada menu Tindakan, pilih Edit.
5. Pada halaman Edit cetak biru, perbarui Deskripsi cetak biru atau Lokasi arsip ZIP (S3). Pastikan untuk menyertakan nama file arsip dalam path.
6. Pilih Save (Simpan).

   Halaman cetak biru mengembalikan dan menunjukkan bahwa status cetak biru adalah. `UPDATING` Pilih tombol refresh hingga status berubah menjadi `ACTIVE` atau `FAILED`.
7. Jika statusnya adalah `FAILED`, pilih cetak biru, dan pada menu Tindakan, pilih Lihat.

Halaman detail menunjukkan alasan kegagalan. Jika pesan kesalahan "Tidak dapat mengakses objek di lokasi..." atau "Akses ditolak pada objek di lokasi...", tinjau persyaratan berikut:

- Pengguna yang Anda gunakan masuk harus memiliki izin baca pada arsip ZIP cetak biru di Amazon S3.
- Bucket Amazon S3 yang berisi arsip ZIP harus memiliki kebijakan bucket yang memberikan izin baca pada objek untuk ID akun AWS Anda. Untuk informasi selengkapnya, lihat Menerbitkan cetak biru.
- Bucket Amazon S3 yang Anda gunakan harus berada dalam Wilayah yang sama dengan Wilayah yang Anda masuki di konsol tersebut.

> **Note**
>
> Jika pembaruan gagal, eksekusi cetak biru berikutnya akan menggunakan cetak biru versi terbaru yang berhasil terdaftar atau diperbarui.

Untuk memperbarui sebuah cetak biru (AWS CLI)

1. Masukkan perintah berikut.

```
aws glue update-blueprint --name <blueprint-name> [--description <description>] --
blueprint-location s3://<s3-path>/<archive-filename>
```

2. Masukkan perintah berikut untuk memeriksa status cetak biru. Ulangi perintah sampai status menjadi `ACTIVE` atau `FAILED`.

```
aws glue get-blueprint --name <blueprint-name>
```

Jika statusnya adalah `FAILED` dan pesan kesalahannya adalah "Tidak dapat mengakses objek di lokasi..." atau "Akses ditolak pada objek di lokasi...", tinjau persyaratan berikut:

- Pengguna yang Anda gunakan masuk harus memiliki izin baca pada arsip ZIP cetak biru di Amazon S3.
- Bucket Amazon S3 yang berisi arsip ZIP harus memiliki kebijakan bucket yang memberikan izin baca pada objek untuk ID akun AWS Anda. Untuk informasi selengkapnya, lihat Menerbitkan cetak biru.

- Bucket Amazon S3 yang Anda gunakan harus berada dalam Wilayah yang sama dengan Wilayah yang Anda masuki di konsol tersebut.

Lihat juga

- Ikhtisar cetak biru di AWS Glue

Membuat alur kerja dari cetak biru di AWS Glue

Anda dapat membuat AWS Glue alur kerja secara manual, menambahkan satu komponen pada satu waktu, atau Anda dapat membuat alur kerja dari cetak biru. AWS Glue AWS Gluetermasuk cetak biru untuk kasus penggunaan umum. AWS GluePengembang Anda dapat membuat cetak biru tambahan.

Important

Batasi jumlah total pekerjaan, crawler, dan pemicu dalam alur kerja hingga 100 atau kurang. Jika Anda menyertakan lebih dari 100, Anda mungkin mendapatkan kesalahan saat mencoba melanjutkan atau menghentikan alur kerja berjalan.

Bila Anda menggunakan sebuah cetak biru, maka Anda dapat dengan cepat menghasilkan sebuah alur kerja untuk sebuah kasus penggunaan tertentu berdasarkan kasus penggunaan umum yang ditentukan oleh cetak biru tersebut. Anda menentukan kasus penggunaan tertentu dengan memberikan nilai untuk parameter cetak biru. Sebagai contoh, cetak biru yang memartisi set data bisa memiliki path sumber dan target Amazon S3 sebagai parameter.

AWS Gluemembuat alur kerja dari cetak biru dengan menjalankan cetak biru. Eksekusi cetak menyimpan nilai parameter yang Anda berikan, dan digunakan untuk melacak kemajuan dan hasil pembuatan alur kerja dan komponen-komponennya. Saat memecahkan masalah alur kerja, Anda dapat melihat eksekusi cetak biru untuk menentukan nilai parameter cetak biru yang digunakan untuk membuat sebuah alur kerja.

Untuk membuat dan melihat alur kerja, Anda memerlukan izin IAM tertentu. Untuk kebijakan IAM yang disarankan, lihat Izin analis data untuk cetak biru.

Anda dapat membuat alur kerja dari cetak biru menggunakan AWS Glue konsol, AWS Glue API, atau (). AWS Command Line Interface AWS CLI

Untuk membuat sebuah alur kerja dari sebuah cetak biru (konsol)

1. Buka konsol AWS Glue di https://console.aws.amazon.com/glue/.

   Masuk sebagai pengguna yang memiliki izin untuk membuat sebuah alur kerja.
2. Di panel navigasi, pilih cetak biru.
3. Pilih cetak biru, dan pada menu Tindakan, pilih Buat alur kerja.
4. Pada halaman Membuat alur kerja dari **<blueprint-name>**, masukkan informasi berikut:

   Parameter cetak biru

   Parameter ini bervariasi tergantung pada desain cetak biru. Untuk pertanyaan tentang parameter, lihat pengembang. cetak biru biasanya menyertakan parameter untuk nama alur kerja.

   IAM role

   Peran yang AWS Glue mengasumsikan untuk membuat alur kerja dan komponennya. Peran tersebut harus memiliki izin untuk membuat dan menghapus alur kerja, tugas, crawler, dan pemicu. Untuk kebijakan yang disarankan untuk peran tersebut, lihat Izin untuk peran cetak biru.
5. Pilih Kirim.

   Halaman Detail Cetak Biru muncul, menampilkan daftar eksekusi cetak biru di bagian bawah.
6. Dalam daftar eksekusi cetak biru, periksa cetak biru paling atas untuk status pembuatan alur kerja.

   Status awal adalah `RUNNING`. Pilih tombol refresh hingga status masuk ke `SUCCEEDED` atau `FAILED`.
7. Lakukan salah satu dari berikut:

   - Jika status penyelesaian adalah `SUCCEEDED`, maka Anda dapat masuk alur kerja, pilih alur kerja yang baru dibuat, dan jalankan. Sebelum menjalankan alur kerja, Anda dapat meninjau grafik desainnya.
   - Jika status penyelesaian adalah `FAILED`, pilih eksekusi cetak biru, dan pada menu Tindakan, pilih Lihat untuk melihat pesan kesalahan.

Untuk informasi selengkapnya tentang alur kerja dan cetak biru, lihat topik berikut.

- Ikhtisar alur kerja di AWS Glue
- Memperbarui cetak biru di AWS Glue
- Membuat dan membangun alur kerja secara manual di AWS Glue

Melihat cetak biru berjalan di AWS Glue

Melihat eksekusi cetak biru untuk melihat informasi berikut:

- Nama alur kerja yang dibuat.
- nilai parameter cetak biru yang digunakan untuk membuat alur kerja.
- Status operasi pembuatan alur kerja.

Anda dapat melihat cetak biru yang dijalankan menggunakan AWS Glue konsol, AWS Glue API, atau AWS Command Line Interface (). AWS CLI

Untuk melihat eksekusi cetak biru (konsol)

1. Buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Di panel navigasi, pilih cetak biru.
3. Pada halaman cetak biru, pilih cetak biru. Di menu Tindakan, pilih Lihat.
4. Pada halaman Detail Cetak Biru, pilih eksekusi cetak biru, dan pada menu Tindakan, pilih Lihat.

Untuk melihat eksekusi cetak biru (AWS CLI)

- Masukkan perintah berikut. Ganti *<blueprint-name>* dengan nama cetak biru. Ganti *< blueprint-run-id >* dengan blueprint run ID.

```
aws glue get-blueprint-run --blueprint-name <blueprint-name> --run-id <blueprint-run-id>
```

Lihat juga:

- Ikhtisar cetak biru di AWS Glue

AWS CloudFormationuntuk AWS Glue

AWS CloudFormation adalah layanan yang dapat membuat banyak sumber daya AWS. AWS Glue menyediakan operasi API untuk membuat objek di AWS Glue Data Catalog. Namun demikian, mungkin lebih mudah untuk menentukan dan membuat objek AWS Glue dan sumber daya terkait AWS lainnya dalam sebuah file templat AWS CloudFormation. Kemudian Anda bisa mengotomatisasi proses pembuatan objeknya.

AWS CloudFormationmenyediakan sintaks yang disederhanakan — baik JSON (JavaScript Object Notation) atau YAMAL (YAMAL Ain't Markup Language) —untuk mengekspresikan penciptaan sumber daya. AWS Anda dapat menggunakan templat AWS CloudFormation untuk menentukan objek Katalog Data seperti basis data, tabel, partisi, crawler, pengklasifikasi, dan koneksi. Anda juga dapat menentukan objek ETL seperti tugas, pemicu, dan titik akhir pengembangan. Anda membuat templat yang menjelaskan semua sumber daya AWS yang Anda inginkan, dan AWS CloudFormation yang akan mengurus penyediaan dan konfigurasi sumber daya tersebut untuk Anda.

Untuk informasi selengkapnya, lihat Apa itu AWS CloudFormation? dan Bekerja dengan Templat AWS CloudFormation dalam Panduan Pengguna AWS CloudFormation.

Jika Anda berencana untuk menggunakan templat AWS CloudFormation yang kompatibel dengan AWS Glue, sebagai administrator, Anda harus memberikan akses ke AWS CloudFormation dan ke layanan dan tindakan AWS yang menjadi tempat bergantungnya. Untuk memberikan izin untuk membuat AWS CloudFormation sumber daya, lampirkan kebijakan berikut ke pengguna yang bekerja denganAWS CloudFormation:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:*"
      ],
      "Resource": "*"
    }
  ]
}
```

Tabel berikut berisi tindakan yang dapat dilakukan oleh templat AWS CloudFormation atas nama Anda. Ia menyertakan tautan ke informasi tentang jenis sumber daya AWS dan jenis propertinya yang dapat Anda tambahkan ke templat AWS CloudFormation.

Sumber daya AWS Glue	templat AWS CloudFormation	AWS Gluesampel
Pengklasifikasi	AWS::Glue::Classifier	Pengklasifikasi Grok, Pengklasifikasi JSON, Pengklasifikasi XML
Koneksi	AWS::Glue::Connection	Koneksi MySQL
Crawler	AWS::Glue::Crawler	Amazon S3 crawler, MySQL crawler
Basis Data	AWS::Glue::Database	Basis data kosong, Basis data dengan tabel
Titik akhir pengembangan	AWS::Glue::DevEndpoint	Titik akhir pengembangan
Tugas	AWS::Glue::Job	Tugas Amazon S3, Tugas JDBC
Transformasi Machine Learning	AWS::Glue: :MLTransform	Transformasi pembelajaran mesin
Aturan kualitas data	AWS::Glue::DataQualitySet Aturan	Set aturan kualitas data, aturan kualitas data dengan penjadwal EventBridge
Partition	AWS::Glue::Partition	Partisi tabel
Tabel	AWS::Glue::Table	Tabel dalam database
Pemicu	AWS::Glue::Trigger	Pemicu sesuai permintaan, Pemicu terjadwal, Pemicu bersyarat

Untuk memulai, gunakan templat contoh berikut dan sesuaikan dengan metadata Anda sendiri. Kemudian gunakan konsol AWS CloudFormation untuk membuat tumpukan AWS CloudFormation untuk menambahkan objek ke AWS Glue dan layanan yang terkait. Banyak bidang dalam objek AWS Glue yang bersifat opsional. Templat ini menggambarkan bidang yang wajib atau diperlukan agar objek AWS Glue bekerja dan berfungsi.

Sebuah templat AWS CloudFormation dapat berupa format JSON atau YAML. Dalam contoh ini, YAML digunakan agar lebih mudah dibaca. Contoh tersebut berisi komentar (#) untuk menggambarkan nilai-nilai yang didefinisikan dalam templat.

Templat AWS CloudFormation dapat mencakup bagian `Parameters`. Bagian ini dapat diubah dalam teks contoh atau ketika file YAML dikirimkan ke konsol AWS CloudFormation untuk membuat sebuah tumpukan. Bagian `Resources` dari templat tersebut berisi definisi dan objek terkait AWS Glue. Definisi sintaksis templat AWS CloudFormation mungkin berisi properti yang mencakup sintaksis properti yang lebih detail. Tidak semua properti mungkin diperlukan untuk membuat sebuah objek AWS Glue. Sampel ini menunjukkan nilai contoh untuk properti umum untuk membuat sebuah objek AWS Glue.

Contoh AWS CloudFormation template untuk AWS Glue database

Sebuah basis data AWS Glue dalam Katalog Data berisi tabel metadata. Basis data terdiri dari properti yang sangat sedikit dan dapat dibuat dalam Katalog Data dengan templat AWS CloudFormation. Contoh templat berikut disediakan untuk Anda agar Anda bisa memulai dan untuk menggambarkan penggunaan tumpukan AWS CloudFormation dengan AWS Glue. Satu-satunya sumber daya yang dibuat oleh templat sampel adalah basis data bernama `cfn-mysampledatabase`. Anda dapat mengubahnya dengan mengedit teks sampel atau mengubah nilai pada konsol AWS CloudFormation saat Anda mengirimkan YAML.

Berikut ini menunjukkan nilai sampel untuk properti umum untuk membuat sebuah basis data AWS Glue. Untuk informasi selengkapnya tentang template AWS CloudFormation databaseAWS Glue, lihat AWS::Glue::Database.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CloudFormation template in YAML to demonstrate creating a database named
 mysampledatabase
# The metadata created in the Data Catalog points to the flights public S3 bucket
#
```

```
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:
  CFNDatabaseName:
    Type: String
    Default: cfn-mysampledatabse

# Resources section defines metadata for the Data Catalog
Resources:
# Create an AWS Glue database
  CFNDatabaseFlights:
    Type: AWS::Glue::Database
    Properties:
      # The database is created in the Data Catalog for your account
      CatalogId: !Ref AWS::AccountId
      DatabaseInput:
        # The name of the database is defined in the Parameters section above
        Name: !Ref CFNDatabaseName
        Description: Database to hold tables for flights data
        LocationUri: s3://crawler-public-us-east-1/flight/2016/csv/
        #Parameters: Leave AWS database parameters blank
```

Contoh AWS CloudFormation template untuk AWS Glue database, tabel, dan partisi

Sebuah tabel AWS Glue berisi metadata yang mendefinisikan struktur dan lokasi data yang ingin Anda proses dengan skrip ETL Anda. Dalam sebuah tabel, Anda dapat menentukan partisi untuk memparalelkan pengolahan data Anda. Sebuah partisi adalah sepotong data yang Anda tetapkan dengan sebuah kunci. Sebagai contoh, dengan menggunakan bulan sebagai kunci, maka semua data untuk Januari akan dimasukkan dalam partisi yang sama. Di AWS Glue, basis data dapat berisi tabel, dan tabel dapat berisi partisi.

Contoh berikut menunjukkan cara mengisi basis data, tabel, dan partisi dengan menggunakan templat AWS CloudFormation. Format data dasar adalah `csv` dan dibatasi oleh koma (,). Karena basis data harus ada sebelum ia dapat berisi tabel, dan tabel harus ada dahulu sebelum partisi dapat dibuat, maka templat menggunakan pernyataan `DependsOn` untuk menentukan dependensi objek-objek tersebut ketika mereka diciptakan.

Nilai-nilai dalam contoh ini menentukan tabel yang berisi data penerbangan dari bucket Amazon S3 yang tersedia untuk umum. Untuk ilustrasi, hanya beberapa kolom data dan satu kunci partisi yang

didefinisikan. Empat partisi juga didefinisikan dalam Katalog Data tersebut. Beberapa bidang untuk menggambarkan penyimpanan basis data juga ditampilkan dalam bidang `StorageDescriptor`.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CloudFormation template in YAML to demonstrate creating a database, a table,
 and partitions
# The metadata created in the Data Catalog points to the flights public S3 bucket
#
# Parameters substituted in the Resources section
# These parameters are names of the resources created in the Data Catalog
Parameters:
  CFNDatabaseName:
    Type: String
    Default: cfn-database-flights-1
  CFNTableName1:
    Type: String
    Default: cfn-manual-table-flights-1
# Resources to create metadata in the Data Catalog
Resources:
###
# Create an AWS Glue database
  CFNDatabaseFlights:
    Type: AWS::Glue::Database
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseInput:
        Name: !Ref CFNDatabaseName
        Description: Database to hold tables for flights data
###
# Create an AWS Glue table
  CFNTableFlights:
    # Creating the table waits for the database to be created
    DependsOn: CFNDatabaseFlights
    Type: AWS::Glue::Table
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableInput:
        Name: !Ref CFNTableName1
        Description: Define the first few columns of the flights table
        TableType: EXTERNAL_TABLE
```

```
        Parameters: {
    "classification": "csv"
  }
#       ViewExpandedText: String
        PartitionKeys:
        # Data is partitioned by month
        - Name: mon
          Type: bigint
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: year
            Type: bigint
          - Name: quarter
            Type: bigint
          - Name: month
            Type: bigint
          - Name: day_of_month
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
# Partition 1
# Create an AWS Glue partition
  CFNPartitionMon1:
    DependsOn: CFNTableFlights
    Type: AWS::Glue::Partition
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableName: !Ref CFNTableName1
      PartitionInput:
        Values:
        - 1
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: mon
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/mon=1/
```

```
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
# Partition 2
# Create an AWS Glue partition
  CFNPartitionMon2:
    DependsOn: CFNTableFlights
    Type: AWS::Glue::Partition
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableName: !Ref CFNTableName1
      PartitionInput:
        Values:
        - 2
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: mon
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/mon=2/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
# Partition 3
# Create an AWS Glue partition
  CFNPartitionMon3:
    DependsOn: CFNTableFlights
    Type: AWS::Glue::Partition
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableName: !Ref CFNTableName1
      PartitionInput:
        Values:
        - 3
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: mon
            Type: bigint
```

```
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/mon=3/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
# Partition 4
# Create an AWS Glue partition
  CFNPartitionMon4:
    DependsOn: CFNTableFlights
    Type: AWS::Glue::Partition
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseName: !Ref CFNDatabaseName
      TableName: !Ref CFNTableName1
      PartitionInput:
        Values:
        - 4
        StorageDescriptor:
          OutputFormat: org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
          Columns:
          - Name: mon
            Type: bigint
          InputFormat: org.apache.hadoop.mapred.TextInputFormat
          Location: s3://crawler-public-us-east-1/flight/2016/csv/mon=4/
          SerdeInfo:
            Parameters:
              field.delim: ","
            SerializationLibrary: org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
```

Contoh AWS CloudFormation template untuk pengklasifikasi AWS Glue grok

Sebuah pengklasifikasi AWS Glue menentukan skema dari data Anda. Salah satu jenis pengklasifikasi kustom menggunakan pola grok untuk mencocokkan data Anda. Jika pola cocok, maka pengklasifikasi kustom tersebut digunakan untuk membuat skema tabel Anda dan mengatur `classification` dengan nilai yang ditetapkan dalam definisi pengklasifikasi.

Sampel ini menciptakan sebuah pengklasifikasi yang menciptakan sebuah skema dengan satu kolom bernama `message` dan menetapkan klasifikasi ke `greedy`.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a classifier
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the classifier to be created
  CFNClassifierName:
    Type: String
    Default: cfn-classifier-grok-one-column-1

#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create classifier that uses grok pattern to put all data in one column and classifies
 it as "greedy".
  CFNClassifierFlights:
    Type: AWS::Glue::Classifier
    Properties:
      GrokClassifier:
        #Grok classifier that puts all data in one column
        Name: !Ref CFNClassifierName
        Classification: greedy

        GrokPattern: "%{GREEDYDATA:message}"
        #CustomPatterns: none
```

Contoh AWS CloudFormation template untuk pengklasifikasi AWS Glue JSON

Sebuah pengklasifikasi AWS Glue menentukan skema dari data Anda. Salah satu jenis pengklasifikasi kustom menggunakan `JsonPath` string yang mendefinisikan data JSON untuk klasifikasi untuk mengklasifikasikan. AWS Gluemendukung subset dari operator untuk`JsonPath`, seperti yang dijelaskan dalam Menulis Pengklasifikasi JsonPath Kustom.

Jika pola cocok, maka pengklasifikasi kustom tersebut digunakan untuk membuat skema tabel Anda.

Contoh ini menciptakan sebuah pengklasifikasi yang membuat skema dengan masing-masing catatan di array `Records3` dalam sebuah objek.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a JSON classifier
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the classifier to be created
  CFNClassifierName:
    Type: String
    Default: cfn-classifier-json-one-column-1

#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create classifier that uses a JSON pattern.
  CFNClassifierFlights:
    Type: AWS::Glue::Classifier
    Properties:
      JSONClassifier:
        #JSON classifier
        Name: !Ref CFNClassifierName
        JsonPath: $.Records3[*]
```

Contoh AWS CloudFormation template untuk pengklasifikasi AWS Glue XHTML

Sebuah pengklasifikasi AWS Glue menentukan skema dari data Anda. Salah satu jenis pengklasifikasi kustom menentukan tag XML untuk menunjuk elemen yang berisi masing-masing catatan dalam dokumen XML yang sedang diurai. Jika pola cocok, maka pengklasifikasi kustom tersebut digunakan untuk membuat skema tabel Anda dan mengatur `classification` dengan nilai yang ditetapkan dalam definisi pengklasifikasi.

Contoh ini menciptakan sebuah pengklasifikasi yang membuat sebuah skema dengan setiap catatan di tag `Record` dan mengatur klasifikasi ke `XML`.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating an XML classifier
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the classifier to be created
  CFNClassifierName:
    Type: String
    Default: cfn-classifier-xml-one-column-1

#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create classifier that uses the XML pattern and classifies it as "XML".
  CFNClassifierFlights:
    Type: AWS::Glue::Classifier
    Properties:
      XMLClassifier:
        #XML classifier
        Name: !Ref CFNClassifierName
        Classification: XML
        RowTag: <Records>
```

Contoh AWS CloudFormation template untuk AWS Glue crawler untuk Amazon S3

Sebuah crawler AWS Glue membuat tabel metadata dalam Katalog Data yang sesuai dengan data Anda. Anda kemudian dapat menggunakan definisi tabel ini sebagai sumber dan target dalam tugas ETL Anda.

Contoh ini membuat sebuah crawler, IAM role yang diperlukan, dan basis data AWS Glue dalam Katalog Data. Ketika crawler ini dijalankan, crawler tersebut mengambil IAM role dan membuat tabel dalam basis data untuk data penerbangan publik. Tabel dibuat dengan prefiks "`cfn_sample_1_`".

IAM role yang dibuat oleh templat ini memungkinkan izin global; Anda mungkin ingin membuat sebuah peran kustom. Tidak ada pengklasifikasi kustom yang didefinisikan oleh pengklasifikasi ini. Pengklasifikasi bawaan AWS Glue digunakan secara default.

Ketika Anda mengirimkan sampel ini ke konsol AWS CloudFormation, Anda harus mengonfirmasi bahwa Anda ingin membuat IAM role.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a crawler
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the crawler to be created
  CFNCrawlerName:
    Type: String
    Default: cfn-crawler-flights-1
  CFNDatabaseName:
    Type: String
    Default: cfn-database-flights-1
  CFNTablePrefixName:
    Type: String
    Default: cfn_sample_1_
#
#
# Resources section defines metadata for the Data Catalog
Resources:
#Create IAM Role assumed by the crawler. For demonstration, this role is given all
 permissions.
  CFNRoleFlights:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Effect: "Allow"
            Principal:
              Service:
                - "glue.amazonaws.com"
```

```
            Action:
              - "sts:AssumeRole"
      Path: "/"
      Policies:
        -
          PolicyName: "root"
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              -
                Effect: "Allow"
                Action: "*"
                Resource: "*"
 # Create a database to contain tables created by the crawler
  CFNDatabaseFlights:
    Type: AWS::Glue::Database
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseInput:
        Name: !Ref CFNDatabaseName
        Description: "AWS Glue container to hold metadata tables for the flights
 crawler"
 #Create a crawler to crawl the flights data on a public S3 bucket
  CFNCrawlerFlights:
    Type: AWS::Glue::Crawler
    Properties:
      Name: !Ref CFNCrawlerName
      Role: !GetAtt CFNRoleFlights.Arn
      #Classifiers: none, use the default classifier
      Description: AWS Glue crawler to crawl flights data
      #Schedule: none, use default run-on-demand
      DatabaseName: !Ref CFNDatabaseName
      Targets:
        S3Targets:
          # Public S3 bucket with the flights data
          - Path: "s3://crawler-public-us-east-1/flight/2016/csv"
      TablePrefix: !Ref CFNTablePrefixName
      SchemaChangePolicy:
        UpdateBehavior: "UPDATE_IN_DATABASE"
        DeleteBehavior: "LOG"
      Configuration: "{\"Version\":1.0,\"CrawlerOutput\":{\"Partitions\":
{\"AddOrUpdateBehavior\":\"InheritFromTable\"},\"Tables\":{\"AddOrUpdateBehavior\":
\"MergeNewColumns\"}}}"
```

Contoh AWS CloudFormation template untuk AWS Glue koneksi

Sebuah koneksi AWS Glue dalam Katalog Data berisi informasi JDBC dan jaringan yang diperlukan untuk connect ke basis data JDBC. Informasi ini digunakan ketika Anda connect ke basis data JDBC untuk melakukan crawling atau menjalankan tugas ETL.

Sampel ini membuat koneksi ke basis data Amazon RDS MySQL bernama `devdb`. Ketika koneksi ini digunakan, IAM role, kredensial basis data, dan nilai-nilai koneksi jaringan juga harus disediakan. Lihat detail bidang yang diperlukan dalam templat tersebut.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a connection
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the connection to be created
  CFNConnectionName:
    Type: String
    Default: cfn-connection-mysql-flights-1
  CFNJDBCString:
    Type: String
    Default: "jdbc:mysql://xxx-mysql.yyyyyyyyyyyyyy.us-east-1.rds.amazonaws.com:3306/
devdb"
  CFNJDBCUser:
    Type: String
    Default: "master"
  CFNJDBCPassword:
    Type: String
    Default: "12345678"
    NoEcho: true
#
#
# Resources section defines metadata for the Data Catalog
Resources:
  CFNConnectionMySQL:
    Type: AWS::Glue::Connection
    Properties:
      CatalogId: !Ref AWS::AccountId
```

```
      ConnectionInput:
        Description: "Connect to MySQL database."
        ConnectionType: "JDBC"
        #MatchCriteria: none
        PhysicalConnectionRequirements:
          AvailabilityZone: "us-east-1d"
          SecurityGroupIdList:
           - "sg-7d52b812"
          SubnetId: "subnet-84f326ee"
        ConnectionProperties: {
          "JDBC_CONNECTION_URL": !Ref CFNJDBCString,
          "USERNAME": !Ref CFNJDBCUser,
          "PASSWORD": !Ref CFNJDBCPassword
        }
        Name: !Ref CFNConnectionName
```

Contoh AWS CloudFormation template untuk AWS Glue crawler untuk JDBC

Sebuah crawler AWS Glue membuat tabel metadata dalam Katalog Data yang sesuai dengan data Anda. Anda kemudian dapat menggunakan definisi tabel ini sebagai sumber dan target dalam tugas ETL Anda.

Contoh ini membuat sebuah crawler, IAM role yang diperlukan, dan basis data AWS Glue dalam Katalog Data. Ketika crawler ini dijalankan, crawler tersebut mengambil IAM role dan membuat tabel dalam basis data untuk data penerbangan publik yang telah disimpan di basis data MySQL. Tabel dibuat dengan prefiks "`cfn_jdbc_1_`". IAM role yang dibuat oleh templat ini memungkinkan izin global; Anda mungkin ingin membuat sebuah peran kustom. Tidak ada pengklasifikasi kustom yang dapat didefinisikan untuk data JDBC. Pengklasifikasi bawaan AWS Glue digunakan secara default.

Ketika Anda mengirimkan sampel ini ke konsol AWS CloudFormation, Anda harus mengonfirmasi bahwa Anda ingin membuat IAM role.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a crawler
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
```

```
Parameters:

# The name of the crawler to be created
  CFNCrawlerName:
    Type: String
    Default: cfn-crawler-jdbc-flights-1
# The name of the database to be created to contain tables
  CFNDatabaseName:
    Type: String
    Default: cfn-database-jdbc-flights-1
# The prefix for all tables crawled and created
  CFNTablePrefixName:
    Type: String
    Default: cfn_jdbc_1_
# The name of the existing connection to the MySQL database
  CFNConnectionName:
    Type: String
    Default: cfn-connection-mysql-flights-1
# The name of the JDBC path (database/schema/table) with wildcard (%) to crawl
  CFNJDBCPath:
    Type: String
    Default: saldev/%
#
#
# Resources section defines metadata for the Data Catalog
Resources:
#Create IAM Role assumed by the crawler. For demonstration, this role is given all
 permissions.
  CFNRoleFlights:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          -
            Effect: "Allow"
            Principal:
              Service:
                - "glue.amazonaws.com"
            Action:
              - "sts:AssumeRole"
      Path: "/"
      Policies:
        -
```

```
          PolicyName: "root"
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              -
                Effect: "Allow"
                Action: "*"
                Resource: "*"
 # Create a database to contain tables created by the crawler
  CFNDatabaseFlights:
    Type: AWS::Glue::Database
    Properties:
      CatalogId: !Ref AWS::AccountId
      DatabaseInput:
        Name: !Ref CFNDatabaseName
        Description: "AWS Glue container to hold metadata tables for the flights
 crawler"
 #Create a crawler to crawl the flights data in MySQL database
  CFNCrawlerFlights:
    Type: AWS::Glue::Crawler
    Properties:
      Name: !Ref CFNCrawlerName
      Role: !GetAtt CFNRoleFlights.Arn
      #Classifiers: none, use the default classifier
      Description: AWS Glue crawler to crawl flights data
      #Schedule: none, use default run-on-demand
      DatabaseName: !Ref CFNDatabaseName
      Targets:
        JdbcTargets:
          # JDBC MySQL database with the flights data
          - ConnectionName: !Ref CFNConnectionName
            Path: !Ref CFNJDBCPath
          #Exclusions: none
      TablePrefix: !Ref CFNTablePrefixName
      SchemaChangePolicy:
        UpdateBehavior: "UPDATE_IN_DATABASE"
        DeleteBehavior: "LOG"
    Configuration: "{\"Version\":1.0,\"CrawlerOutput\":{\"Partitions\":
{\"AddOrUpdateBehavior\":\"InheritFromTable\"},\"Tables\":{\"AddOrUpdateBehavior\":
\"MergeNewColumns\"}}}"
```

Contoh AWS CloudFormation template untuk AWS Glue pekerjaan Amazon S3 ke Amazon S3

Sebuah tugas AWS Glue di Katalog Data berisi nilai parameter yang diperlukan untuk menjalankan skrip di AWS Glue.

Sampel ini menciptakan tugas yang membaca data penerbangan dari bucket Amazon S3 dalam format `csv` dan menulisnya ke file Amazon S3 Parquet. Skrip yang dijalankan oleh tugas ini harus sudah ada. Anda dapat membuat skrip ETL untuk lingkungan Anda dengan konsol AWS Glue. Ketika tugas ini dijalankan, IAM role dengan izin yang semestinya juga harus disediakan.

Nilai parameter umum ditunjukkan dalam templat tersebut. Misalnya, `AllocatedCapacity` (DPU) default untuk 5.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a job using the public flights S3 table in a
 public bucket
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the job to be created
  CFNJobName:
    Type: String
    Default: cfn-job-S3-to-S3-2
# The name of the IAM role that the job assumes. It must have access to data, script,
 temporary directory
  CFNIAMRoleName:
    Type: String
    Default: AWSGlueServiceRoleGA
# The S3 path where the script for this job is located
  CFNScriptLocation:
    Type: String
    Default: s3://aws-glue-scripts-123456789012-us-east-1/myid/sal-job-test2
#
#
# Resources section defines metadata for the Data Catalog
Resources:
```

```
# Create job to run script which accesses flightscsv table and write to S3 file as
 parquet.
# The script already exists and is called by this job
  CFNJobFlights:
    Type: AWS::Glue::Job
    Properties:
      Role: !Ref CFNIAMRoleName
      #DefaultArguments: JSON object
      # If script written in Scala, then set DefaultArguments={'--job-language';
 'scala', '--class': 'your scala class'}
      #Connections:  No connection needed for S3 to S3 job
      #  ConnectionsList
      #MaxRetries: Double
      Description: Job created with CloudFormation
      #LogUri: String
      Command:
        Name: glueetl
        ScriptLocation: !Ref CFNScriptLocation
             # for access to directories use proper IAM role with permission to buckets
 and folders that begin with "aws-glue-"
             # script uses temp directory from job definition if required (temp
 directory not used S3 to S3)
             # script defines target for output as s3://aws-glue-target/sal
      AllocatedCapacity: 5
      ExecutionProperty:
        MaxConcurrentRuns: 1
      Name: !Ref CFNJobName
```

Contoh AWS CloudFormation template untuk AWS Glue pekerjaan JDBC ke Amazon S3

Sebuah tugas AWS Glue di Katalog Data berisi nilai parameter yang diperlukan untuk menjalankan skrip di AWS Glue.

Sampel ini membuat sebuah tugas yang membaca data penerbangan dari basis data MySQL JDBC sebagaimana didefinisikan oleh koneksi bernama `cfn-connection-mysql-flights-1` dan menuliskannya ke file Amazon S3 Parquet. Skrip yang dijalankan oleh tugas ini harus sudah ada. Anda dapat membuat skrip ETL untuk lingkungan Anda dengan konsol AWS Glue. Ketika tugas ini dijalankan, IAM role dengan izin yang semestinya juga harus disediakan.

Nilai parameter umum ditunjukkan dalam templat tersebut. Misalnya, `AllocatedCapacity` (DPU) default untuk 5.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a job using a MySQL JDBC DB with the flights
 data to an S3 file
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the job to be created
  CFNJobName:
    Type: String
    Default: cfn-job-JDBC-to-S3-1
# The name of the IAM role that the job assumes. It must have access to data, script,
 temporary directory
  CFNIAMRoleName:
    Type: String
    Default: AWSGlueServiceRoleGA
# The S3 path where the script for this job is located
  CFNScriptLocation:
    Type: String
    Default: s3://aws-glue-scripts-123456789012-us-east-1/myid/sal-job-dec4a
# The name of the connection used for JDBC data source
  CFNConnectionName:
    Type: String
    Default: cfn-connection-mysql-flights-1
#
#
# Resources section defines metadata for the Data Catalog
Resources:
# Create job to run script which accesses JDBC flights table via a connection and write
 to S3 file as parquet.
# The script already exists and is called by this job
  CFNJobFlights:
    Type: AWS::Glue::Job
    Properties:
      Role: !Ref CFNIAMRoleName
      #DefaultArguments: JSON object
```

```
      # For example, if required by script, set temporary directory as
 DefaultArguments={'--TempDir'; 's3://aws-glue-temporary-xyc/sal'}
      Connections:
        Connections:
        - !Ref CFNConnectionName
      #MaxRetries: Double
      Description: Job created with CloudFormation using existing script
      #LogUri: String
      Command:
        Name: glueetl
        ScriptLocation: !Ref CFNScriptLocation
             # for access to directories use proper IAM role with permission to buckets
 and folders that begin with "aws-glue-"
             # if required, script defines temp directory as argument TempDir and used
 in script like redshift_tmp_dir = args["TempDir"]
             # script defines target for output as s3://aws-glue-target/sal
      AllocatedCapacity: 5
      ExecutionProperty:
        MaxConcurrentRuns: 1
      Name: !Ref CFNJobName
```

Contoh AWS CloudFormation template untuk pemicu AWS Glue sesuai permintaan

Sebuah pemicu AWS Glue dalam Katalog Data berisi nilai-nilai parameter yang diperlukan untuk memulai eksekusi tugas ketika pemicu aktif. Sebuah pemicu sesuai permintaan aktif ketika Anda mengaktifkannya.

Sampel ini menciptakan sebuah pemicu sesuai permintaan yang memulai satu tugas bernama `cfn-job-S3-to-S3-1`.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating an on-demand trigger
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:
  # The existing job to be started by this trigger
  CFNJobName:
```

```
    Type: String
    Default: cfn-job-S3-to-S3-1
  # The name of the trigger to be created
  CFNTriggerName:
    Type: String
    Default: cfn-trigger-ondemand-flights-1
#
# Resources section defines metadata for the Data Catalog
# Sample CFN YAML to demonstrate creating an on-demand trigger for a job
Resources:
# Create trigger to run an existing job (CFNJobName) on an on-demand schedule.
  CFNTriggerSample:
    Type: AWS::Glue::Trigger
    Properties:
      Name:
        Ref: CFNTriggerName
      Description: Trigger created with CloudFormation
      Type: ON_DEMAND
      Actions:
        - JobName: !Ref CFNJobName
        # Arguments: JSON object
      #Schedule:
      #Predicate:
```

Contoh AWS CloudFormation template untuk pemicu AWS Glue terjadwal

Sebuah pemicu AWS Glue dalam Katalog Data berisi nilai-nilai parameter yang diperlukan untuk memulai eksekusi tugas ketika pemicu aktif. Sebuah pemicu terjadwal aktif ketika diaktifkan dan pewaktu cron muncul.

Sampel ini menciptakan sebuah pemicu terjadwal yang memulai satu tugas bernama `cfn-job-S3-to-S3-1`. Pewaktu adalah ekspresi cron untuk menjalankan tugas setiap 10 menit pada hari kerja.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a scheduled trigger
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
```

```
Parameters:
  # The existing job to be started by this trigger
  CFNJobName:
    Type: String
    Default: cfn-job-S3-to-S3-1
  # The name of the trigger to be created
  CFNTriggerName:
    Type: String
    Default: cfn-trigger-scheduled-flights-1
#
# Resources section defines metadata for the Data Catalog
# Sample CFN YAML to demonstrate creating a scheduled trigger for a job
#
Resources:
# Create trigger to run an existing job (CFNJobName) on a cron schedule.
  TriggerSample1CFN:
    Type: AWS::Glue::Trigger
    Properties:
      Name:
        Ref: CFNTriggerName
      Description: Trigger created with CloudFormation
      Type: SCHEDULED
      Actions:
        - JobName: !Ref CFNJobName
        # Arguments: JSON object
      # # Run the trigger every 10 minutes on Monday to Friday
      Schedule: cron(0/10 * ? * MON-FRI *)
      #Predicate:
```

Contoh AWS CloudFormation template untuk AWS Glue pemicu bersyarat

Sebuah pemicu AWS Glue dalam Katalog Data berisi nilai-nilai parameter yang diperlukan untuk memulai eksekusi tugas ketika pemicu aktif. Pemicu bersyarat aktif ketika diaktifkan dan syarat terpenuhi, seperti sebuah tugas yang berhasil selesai.

Sampel ini menciptakan sebuah pemicu bersyarat yang memulai satu tugas bernama `cfn-job-S3-to-S3-1`. Pekerjaan ini dimulai ketika tugas bernama `cfn-job-S3-to-S3-2` berhasil diselesaikan.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a conditional trigger for a job, which starts
 when another job completes
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:
  # The existing job to be started by this trigger
  CFNJobName:
    Type: String
    Default: cfn-job-S3-to-S3-1
  # The existing job that when it finishes causes trigger to fire
  CFNJobName2:
    Type: String
    Default: cfn-job-S3-to-S3-2
  # The name of the trigger to be created
  CFNTriggerName:
    Type: String
    Default: cfn-trigger-conditional-1
#
Resources:
# Create trigger to run an existing job (CFNJobName) when another job completes
 (CFNJobName2).
  CFNTriggerSample:
    Type: AWS::Glue::Trigger
    Properties:
      Name:
        Ref: CFNTriggerName
      Description: Trigger created with CloudFormation
      Type: CONDITIONAL
      Actions:
        - JobName: !Ref CFNJobName
        # Arguments: JSON object
      #Schedule: none
      Predicate:
        #Value for Logical is required if more than 1 job listed in Conditions
        Logical: AND
        Conditions:
          - LogicalOperator: EQUALS
            JobName: !Ref CFNJobName2
            State: SUCCEEDED
```

Contoh AWS CloudFormation template untuk titik akhir AWS Glue pengembangan

Transformasi pembelajaran AWS Glue mesin adalah transformasi khusus untuk membersihkan data Anda. Saat ini ada satu transformasi yang tersedia bernama FindMatches. FindMatches Transformasi memungkinkan Anda mengidentifikasi duplikat atau pencocokan catatan dalam kumpulan data Anda, bahkan ketika catatan tidak memiliki pengidentifikasi unik umum dan tidak ada bidang yang sama persis.

Sampel ini menciptakan transformasi pembelajaran mesin. Untuk informasi selengkapnya tentang parameter yang Anda perlukan untuk membuat transformasi pembelajaran mesin, lihatRekam pencocokan dengan AWS Lake Formation FindMatches.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a machine learning transform
#
# Resources section defines metadata for the machine learning transform
Resources:
  MyMLTransform:
    Type: "AWS::Glue::MLTransform"
    Condition: "isGlueMLGARegion"
    Properties:
      Name: !Sub "MyTransform"
      Description: "The bestest transform ever"
      Role: !ImportValue MyMLTransformUserRole
      GlueVersion: "1.0"
      WorkerType: "Standard"
      NumberOfWorkers: 5
      Timeout: 120
      MaxRetries: 1
      InputRecordTables:
        GlueTables:
          - DatabaseName: !ImportValue MyMLTransformDatabase
            TableName: !ImportValue MyMLTransformTable
      TransformParameters:
        TransformType: "FIND_MATCHES"
        FindMatchesParameters:
          PrimaryKeyColumnName: "testcolumn"
          PrecisionRecallTradeoff: 0.5
```

```
          AccuracyCostTradeoff: 0.5
          EnforceProvidedLabels: True
      Tags:
        key1: "value1"
        key2: "value2"
      TransformEncryption:
        TaskRunSecurityConfigurationName: !ImportValue
 MyMLTransformSecurityConfiguration
        MLUserDataEncryption:
          MLUserDataEncryptionMode: "SSE-KMS"
          KmsKeyId: !ImportValue MyMLTransformEncryptionKey
```

Contoh AWS CloudFormation template untuk kumpulan AWS Glue Data Quality aturan

Kumpulan aturan Kualitas AWS Glue Data berisi aturan yang dapat dievaluasi pada tabel dalam Katalog Data. Setelah kumpulan aturan ditempatkan pada tabel yang ditargetkan, Anda dapat masuk ke Katalog Data dan menjalankan evaluasi yang menjalankan data Anda terhadap aturan tersebut dalam kumpulan aturan. Aturan ini dapat bervariasi dari mengevaluasi jumlah baris hingga mengevaluasi integritas referensial pada data Anda.

Contoh berikut adalah CloudFormation template yang membuat ruleset dengan berbagai aturan pada tabel target yang ditentukan.

```
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a DataQualityRuleset
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

  # The name of the ruleset to be created
  RulesetName:
    Type: String
    Default: "CFNRulesetName"
  RulesetDescription:
    Type: String
    Default: "CFN DataQualityRuleset"
  # Rules that will be associated with this ruleset
  Rules:
    Type: String
```

```
    Default: 'Rules = [
        RowCount > 100,
        IsUnique "id",
        IsComplete "nametype"
        ]'
  # Name of database and table within Data Catalog which the ruleset will
  # be applied too
  DatabaseName:
    Type: String
    Default: "ExampleDatabaseName"
  TableName:
    Type: String
    Default: "ExampleTableName"

# Resources section defines metadata for the Data Catalog
Resources:
  # Creates a Data Quality ruleset under specified rules
  DQRuleset:
    Type: AWS::Glue::DataQualityRuleset
    Properties:
      Name: !Ref RulesetName
      Description: !Ref RulesetDescription
      # The String within rules must be formatted in DQDL, a language
      # used specifically to make rules
      Ruleset: !Ref Rules
      # The targeted table must exist within Data Catalog alongside
      # the correct database
      TargetTable:
        DatabaseName: !Ref DatabaseName
        TableName: !Ref TableName
```

Contoh AWS CloudFormation template untuk AWS Glue Data Quality kumpulan aturan dengan penjadwal EventBridge

Kumpulan aturan Kualitas AWS Glue Data berisi aturan yang dapat dievaluasi pada tabel dalam Katalog Data. Setelah kumpulan aturan ditempatkan pada tabel yang ditargetkan, Anda dapat masuk ke Katalog Data dan menjalankan evaluasi yang menjalankan data Anda terhadap aturan tersebut dalam kumpulan aturan. Alih-alih harus secara manual masuk ke Katalog Data untuk mengevaluasi kumpulan aturan, Anda juga dapat menambahkan EventBridge Penjadwal dalam CloudFormation template kami untuk menjadwalkan evaluasi kumpulan aturan ini untuk Anda pada interval waktu.

Contoh berikut adalah CloudFormation template yang membuat aturan Kualitas Data dan EventBridge Scheduler untuk mengevaluasi aturan yang disebutkan di atas setiap lima menit.

```
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a DataQualityRuleset
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

  # The name of the ruleset to be created
  RulesetName:
    Type: String
    Default: "CFNRulesetName"
  # Rules that will be associated with this Ruleset
  Rules:
    Type: String
    Default: 'Rules = [
        RowCount > 100,
        IsUnique "id",
        IsComplete "nametype"
        ]'
  # The name of the Schedule to be created
  ScheduleName:
    Type: String
    Default: "ScheduleDQRulsetEvaluation"
  # This expression determines the rate at which the Schedule will evaluate
  # your data using the above ruleset
  ScheduleRate:
    Type: String
    Default: "rate(5 minutes)"
  # The Request that being sent must match the details of the Data Quality Ruleset
  ScheduleRequest:
    Type: String
    Default: '
        { "DataSource": { "GlueTable": { "DatabaseName": "ExampleDatabaseName",
         "TableName": "ExampleTableName" } },
         "Role": "role/AWSGlueServiceRoleDefault",
          "RulesetNames": [ ""CFNRulesetName"" ] }
        '

# Resources section defines metadata for the Data Catalog
Resources:
```

```
  # Creates a Data Quality ruleset under specified rules
  DQRuleset:
    Type: AWS::Glue::DataQualityRuleset
    Properties:
      Name: !Ref RulesetName
      Description: "CFN DataQualityRuleset"
      # The String within rules must be formatted in DQDL, a language
      # used specifically to make rules
      Ruleset: !Ref Rules
      # The targeted table must exist within Data Catalog alongside
      # the correct database
      TargetTable:
        DatabaseName: "ExampleDatabaseName"
        TableName: "ExampleTableName"
  # Create a Scheduler to schedule evaluation runs on the above ruleset
  ScheduleDQEval:
    Type: AWS::Scheduler::Schedule
    Properties:
      Name: !Ref ScheduleName
      Description: "Schedule DataQualityRuleset Evaluations"
      FlexibleTimeWindow:
        Mode: "OFF"
      ScheduleExpression: !Ref ScheduleRate
      ScheduleExpressionTimezone: "America/New_York"
      State: "ENABLED"
      Target:
        # The ARN is the API that will be run, since we want to evaluate our ruleset
        # we want this specific ARN
        Arn: "arn:aws:scheduler:::aws-sdk:glue:startDataQualityRulesetEvaluationRun"
        # Your RoleArn must have approval to schedule
        RoleArn: "arn:aws:iam::123456789012:role/AWSGlueServiceRoleDefault"
        # This is the Request that is being sent to the Arn
        Input: '
        { "DataSource": { "GlueTable": { "DatabaseName": "sampledb", "TableName":
"meteorite" } },
         "Role": "role/AWSGlueServiceRoleDefault",
          "RulesetNames": [ "TestCFN" ] }
        '
```

Contoh AWS CloudFormation template untuk titik akhir AWS Glue pengembangan

Sebuah titik akhir pengembangan AWS Glue adalah sebuah lingkungan yang dapat Anda gunakan untuk mengembangkan dan menguji skrip AWS Glue.

Sampel ini menciptakan sebuah titik akhir pengembangan dengan nilai parameter jaringan minimal yang diperlukan untuk berhasil membuatnya. Untuk informasi lebih lanjut tentang parameter yang Anda butuhkan untuk menyiapkan titik akhir pengembangan, lihat Menyiapkan jaringan untuk pengembangan AWS Glue.

Anda memberikan ARN (Amazon Resource Name) IAM role yang ada untuk membuat titik akhir pengembangan. Berikan kunci publik RSA yang benar dan sediakan kunci privat yang sesuai jika Anda berencana untuk membuat server notebook pada titik akhir pengembangan.

> **Note**
>
> Untuk setiap notebook server yang Anda buat yang dikaitkan dengan titik akhir pengembangan, Anda mengelolanya. Oleh karena itu, jika Anda menghapus titik akhir pengembangan, untuk menghapus server notebook, Anda harus menghapus tumpukan AWS CloudFormation pada konsol AWS CloudFormation.

```
---
AWSTemplateFormatVersion: '2010-09-09'
# Sample CFN YAML to demonstrate creating a development endpoint
#
# Parameters section contains names that are substituted in the Resources section
# These parameters are the names the resources created in the Data Catalog
Parameters:

# The name of the crawler to be created
  CFNEndpointName:
    Type: String
    Default: cfn-devendpoint-1
  CFNIAMRoleArn:
    Type: String
    Default: arn:aws:iam::123456789012/role/AWSGlueServiceRoleGA
#
```

```
#
# Resources section defines metadata for the Data Catalog
Resources:
  CFNDevEndpoint:
    Type: AWS::Glue::DevEndpoint
    Properties:
      EndpointName: !Ref CFNEndpointName
      #ExtraJarsS3Path: String
      #ExtraPythonLibsS3Path: String
      NumberOfNodes: 5
      PublicKey: ssh-rsa public.....key myuserid-key
      RoleArn: !Ref CFNIAMRoleArn
      SecurityGroupIds:
        - sg-64986c0b
      SubnetId: subnet-c67cccac
```

AWS Glue panduan pemrograman

Skrip berisi kode yang mengekstrak data dari sumber, mengubahnya, dan memuatnya menjadi target. AWS Glue menjalankan skrip ketika memulai pekerjaan.

AWS GlueSkrip ETL dikodekan dalam Python atau Scala. Sementara semua jenis pekerjaan dapat ditulis dengan Python, AWS Glue untuk pekerjaan Spark dapat ditulis dalam Scala juga. Ketika Anda secara otomatis menghasilkan logika kode sumber untuk pekerjaan Anda AWS Glue Studio, skrip dibuat. Anda dapat mengedit skrip ini, atau Anda dapat memberikan skrip Anda sendiri untuk memproses tugas ETL Anda.

Menyediakan skrip kustom Anda sendiri

Skrip melakukan pekerjaan ekstrak, transformasi, dan pemuatan (ETL) di. AWS Glue Sebuah skrip dibuat saat Anda secara otomatis membuat logika kode sumber untuk sebuah tugas. Anda dapat mengedit skrip yang dihasilkan secara otomatis ini, atau Anda dapat memberikan skrip kustom Anda sendiri.

Untuk menyediakan skrip kustom Anda sendiri di AWS Glue, ikuti langkah-langkah umum berikut ini:

1. Masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/.
2. Pilih tab ETL Jobs, lalu lihat bagian Buat pekerjaan. Pilih opsi editor skrip.
3. Pada Tugas ini menjalankan, pilih salah satu dari berikut:
   - Buat skrip baru dengan kode boilerplate
   - Unggah dan edit skrip yang ada
4. Pada halaman Job details, pilih peran IAM yang diperlukan untuk menjalankan skrip kustom Anda. Untuk informasi selengkapnya, lihat Manajemen identitas dan akses untuk AWS Glue.
5. Pilih koneksi yang di-referensi skrip Anda. Objek-objek ini diperlukan untuk connect ke penyimpanan data JDBC yang diperlukan.

   Antarmuka jaringan elastis adalah antarmuka jaringan virtual yang dapat Anda lampirkan ke sebuah instans di virtual private cloud (VPC). Pilih antarmuka jaringan elastis yang diperlukan untuk connect ke penyimpanan data yang digunakan dalam skrip.

6. Berikan konfigurasi tambahan, termasuk parameter, khusus untuk jenis pekerjaan Anda. Untuk informasi selengkapnya tentang konfigurasi untuk jenis pekerjaan Anda, lihat Membangun pekerjaan ETL visual dengan AWS Glue Studio bagian.
7. Pada tab Script, tempel atau tulis skrip kustom Anda.

Gunakan konten di bagian ini untuk memandu proses penulisan skrip kustom Anda.

Untuk informasi selengkapnya tentang cara menambahkan tugas di AWS Glue, lihat Membangun pekerjaan ETL visual dengan AWS Glue Studio.

Untuk step-by-step panduan, lihat tutorial Tambah pekerjaan di AWS Glue konsol.

Pemrograman skrip Spark

AWS Gluemembuatnya mudah untuk menulis atau membuat otomatis skrip ekstrak, transformasi, dan memuat (ETL), selain menguji dan menjalankannya. Bagian ini menjelaskan ekstensi ke Apache Spark yang AWS Glue telah diperkenalkan, dan memberikan contoh cara membuat kode dan menjalankan skrip ETL dengan Python dan Scala.

> **Important**
>
> Versi AWS Glue yang berbeda men-support versi Apache Spark yang berbeda. Skrip kustom Anda harus kompatibel dengan versi Apache Spark yang didukung. Untuk informasi selengkapnya tentang versi AWS Glue, lihat Glue version job property.

Topik

- Tutorial: Menulis AWS Glue untuk skrip Spark
- Program skrip AWS Glue ETL di PySpark
- Pemrograman skrip AWS Glue ETL dalam Scala
- Fitur dan optimasi untuk pemrograman AWS Glue untuk skrip Spark ETL

Tutorial: Menulis AWS Glue untuk skrip Spark

Tutorial ini memperkenalkan Anda pada proses penulisan skrip AWS Glue. Anda dapat menjalankan skrip sesuai jadwal dengan pekerjaan, atau secara interaktif dengan sesi interaktif. Untuk informasi

lebih lanjut tentang pekerjaan, lihatMembangun pekerjaan ETL visual dengan AWS Glue Studio. Untuk informasi selengkapnya tentang sesi interaktif, lihatthe section called “Ikhtisar sesi AWS Glue interaktif”.

Editor visual AWS Glue Studio menawarkan antarmuka grafis tanpa kode untuk membangun pekerjaan AWS Glue. AWS Glue script kembali pekerjaan visual. Mereka memberi Anda akses ke seperangkat alat yang diperluas yang tersedia untuk bekerja dengan program Apache Spark. Anda dapat mengakses API Spark asli, serta pustaka AWS Glue yang memfasilitasi alur kerja ekstrak, transformasi, dan pemuatan (ETL) dari dalam skrip Glue. AWS

Dalam tutorial ini, Anda mengekstrak, mengubah, dan memuat dataset tiket parkir. Script yang melakukan pekerjaan ini identik dalam bentuk dan fungsi dengan yang dihasilkan dalam Membuat ETL lebih mudah dengan AWS Glue Studio di AWS Big Data Blog, yang memperkenalkan editor visual AWS Glue Studio. Dengan menjalankan skrip ini dalam suatu pekerjaan, Anda dapat membandingkannya dengan pekerjaan visual dan melihat cara kerja skrip AWS Glue ETL. Ini mempersiapkan Anda untuk menggunakan fungsionalitas tambahan yang belum tersedia dalam pekerjaan visual.

Anda menggunakan bahasa Python dan pustaka dalam tutorial ini. Fungsionalitas serupa tersedia di Scala. Setelah melalui tutorial ini, Anda harus dapat menghasilkan dan memeriksa contoh skrip Scala untuk memahami bagaimana melakukan proses penulisan skrip Scala AWS Glue ETL.

Prasyarat

Tutorial ini memiliki prasyarat berikut ini:

- Prasyarat yang sama dengan posting blog AWS Glue Studio, yang menginstruksikan Anda untuk menjalankan template. AWS CloudFormation

  Template ini menggunakan AWS Glue Data Catalog untuk mengelola dataset tiket parkir yang tersedia di`s3://aws-bigdata-blog/artifacts/gluestudio/`. Ini menciptakan sumber daya berikut yang akan direferensikan:

- AWS Glue StudioPeran — Peran IAM untuk menjalankan pekerjaan AWS Glue
- AWS Glue StudioAmazon S3Bucket - Nama bucket Amazon S3 untuk menyimpan file terkait blog
- AWS Glue StudioTicketSyyzDB - Database Katalog Data AWS Glue
- AWS Glue StudioTableTickets— Tabel Katalog Data untuk digunakan sebagai sumber
- AWS Glue StudioTableTrials— Tabel Katalog Data untuk digunakan sebagai sumber

- AWS Glue StudioParkingTicketCount — Tabel Katalog Data untuk digunakan sebagai tujuan
- Skrip yang dihasilkan di posting blog AWS Glue Studio. Jika posting blog berubah, skrip juga tersedia dalam teks berikut.

Menghasilkan contoh skrip

Anda dapat menggunakan editor visual AWS Glue Studio sebagai alat pembuatan kode yang kuat untuk membuat perancah untuk skrip yang ingin Anda tulis. Anda akan menggunakan alat ini untuk membuat skrip sampel.

Jika Anda ingin melewati langkah-langkah ini, skrip disediakan.

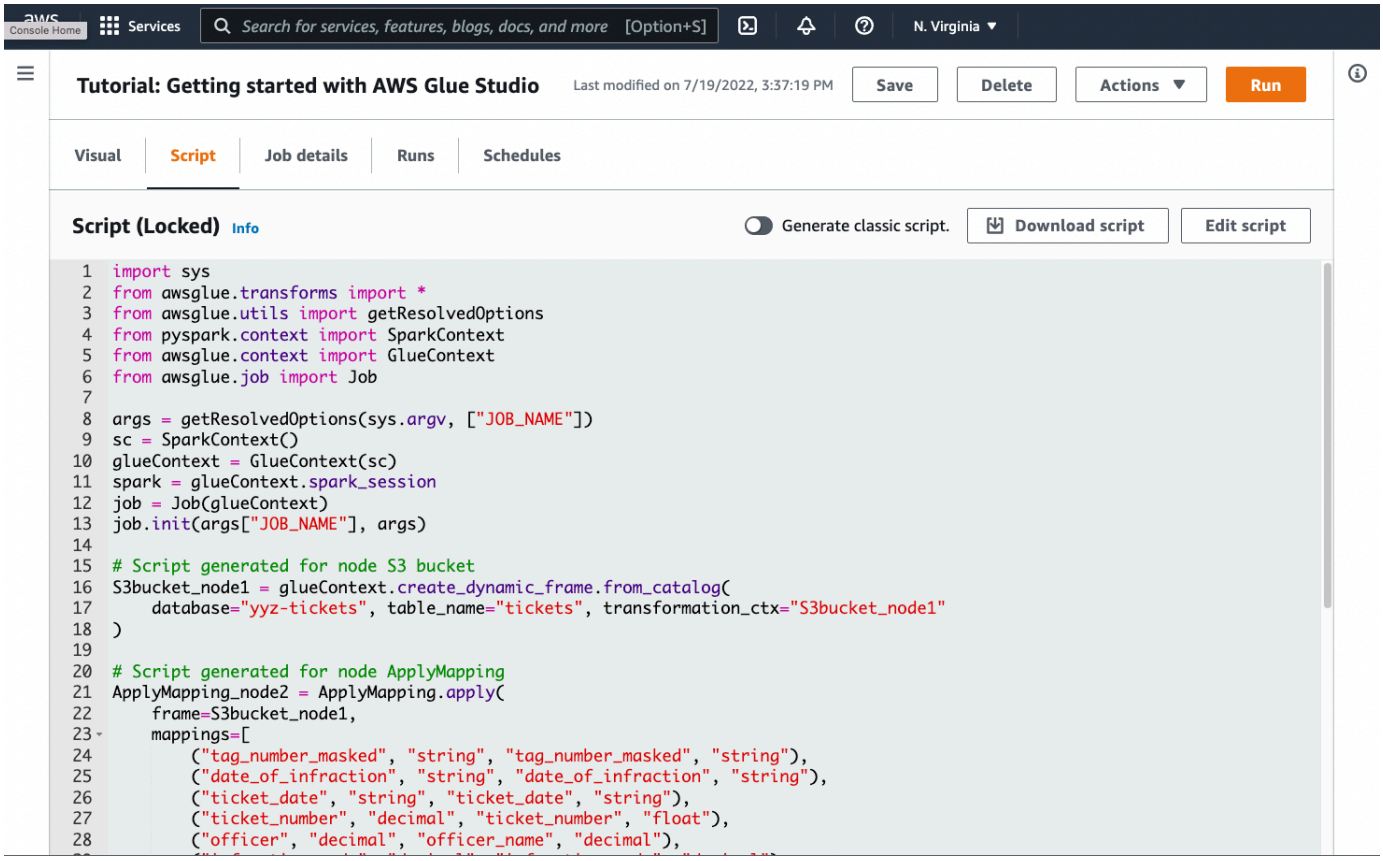
Contoh skrip tutorial

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args["JOB_NAME"], args)

# Script generated for node S3 bucket
S3bucket_node1 = glueContext.create_dynamic_frame.from_catalog(
    database="yyz-tickets", table_name="tickets", transformation_ctx="S3bucket_node1"
)

# Script generated for node ApplyMapping
ApplyMapping_node2 = ApplyMapping.apply(
    frame=S3bucket_node1,
    mappings=[
        ("tag_number_masked", "string", "tag_number_masked", "string"),
        ("date_of_infraction", "string", "date_of_infraction", "string"),
        ("ticket_date", "string", "ticket_date", "string"),
        ("ticket_number", "decimal", "ticket_number", "float"),
        ("officer", "decimal", "officer_name", "decimal"),
```

```
        ("infraction_code", "decimal", "infraction_code", "decimal"),
        ("infraction_description", "string", "infraction_description", "string"),
        ("set_fine_amount", "decimal", "set_fine_amount", "float"),
        ("time_of_infraction", "decimal", "time_of_infraction", "decimal"),
    ],
    transformation_ctx="ApplyMapping_node2",
)

# Script generated for node S3 bucket
S3bucket_node3 = glueContext.write_dynamic_frame.from_options(
    frame=ApplyMapping_node2,
    connection_type="s3",
    format="glueparquet",
    connection_options={"path": "s3://DOC-EXAMPLE-BUCKET", "partitionKeys": []},
    format_options={"compression": "gzip"},
    transformation_ctx="S3bucket_node3",
)

job.commit()
```

Untuk menghasilkan skrip sampel

1. Lengkapi tutorial AWS Glue Studio. Untuk menyelesaikan tutorial ini, lihat Membuat pekerjaan di AWS Glue Studio dari contoh pekerjaan.
2. Arahkan ke tab Script di halaman pekerjaan, seperti yang ditunjukkan pada gambar berikut:



3. Salin isi lengkap tab Script. Dengan mengatur bahasa skrip dalam rincian Job, Anda dapat beralih bolak-balik antara menghasilkan kode Python atau Scala.

Langkah 1. Buat pekerjaan dan tempel skrip Anda

Pada langkah ini, Anda membuat pekerjaan AWS Glue di AWS Management Console. Ini mengatur konfigurasi yang memungkinkan AWS Glue menjalankan skrip Anda. Ini juga menciptakan tempat bagi Anda untuk menyimpan dan mengedit skrip Anda.

Untuk membuat pekerjaan

1. Di AWS Management Console, navigasikan ke halaman landing AWS Glue.
2. Di panel navigasi samping, pilih Jobs.
3. Pilih Editor skrip Spark di Buat pekerjaan, lalu pilih Buat.
4. Opsional - Tempelkan teks lengkap skrip Anda ke panel Script. Atau, Anda dapat mengikuti tutorial.

Langkah 2. Pustaka Impor AWS Glue

Anda perlu mengatur skrip Anda untuk berinteraksi dengan kode dan konfigurasi yang didefinisikan di luar skrip. Pekerjaan ini dilakukan di belakang layar di AWS Glue Studio.

Pada langkah ini, Anda melakukan tindakan berikut.

- Impor dan inisialisasi `GlueContext` objek. Ini adalah impor yang paling penting, dari perspektif penulisan skrip. Ini memperlihatkan metode standar untuk mendefinisikan kumpulan data sumber dan target, yang merupakan titik awal untuk skrip ETL apa pun. Untuk mempelajari lebih lanjut tentang `GlueContext` kelas, lihatGlueContext kelas.
- Inisialisasi a `SparkContext` dan`SparkSession`. Ini memungkinkan Anda untuk mengkonfigurasi mesin Spark yang tersedia di dalam pekerjaan AWS Glue. Anda tidak perlu menggunakannya secara langsung dalam skrip AWS Glue pengantar.
- Panggil `getResolvedOptions` untuk menyiapkan argumen pekerjaan Anda untuk digunakan dalam skrip. Untuk informasi selengkapnya tentang menyelesaikan parameter pekerjaan, lihatthe section called “getResolvedOptions”.
- Inisialisasi a. `Job` `Job`Objek mengatur konfigurasi dan melacak keadaan berbagai fitur AWS Glue opsional. Skrip Anda dapat berjalan tanpa `Job` objek, tetapi praktik terbaik adalah menginisialisasi sehingga Anda tidak mengalami kebingungan jika fitur tersebut kemudian terintegrasi.

  Salah satu fitur ini adalah bookmark pekerjaan, yang dapat Anda konfigurasikan secara opsional dalam tutorial ini. Anda dapat mempelajari tentang bookmark pekerjaan di bagian berikut,the section called “Opsional - Aktifkan bookmark pekerjaan”.

Dalam prosedur ini, Anda menulis kode berikut. Kode ini adalah bagian dari skrip sampel yang dihasilkan.

```
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
```

```
job.init(args["JOB_NAME"], args)
```

Untuk mengimpor pustaka AWS Glue

- Salin bagian kode ini dan tempel ke editor Script.

> **Note**
>
> Anda mungkin menganggap menyalin kode sebagai praktik teknik yang buruk. Dalam tutorial ini, kami menyarankan ini untuk mendorong Anda untuk secara konsisten memberi nama variabel inti Anda di semua skrip AWS Glue ETL.

Langkah 3. Ekstrak data dari sumber

Dalam proses ETL apa pun, Anda harus terlebih dahulu menentukan kumpulan data sumber yang ingin Anda ubah. Di editor visual AWS Glue Studio, Anda memberikan informasi ini dengan membuat simpul Sumber.

Pada langkah ini, Anda memberikan `create_dynamic_frame.from_catalog` metode a `database` dan `table_name` untuk mengekstrak data dari sumber yang dikonfigurasi dalam Katalog Data AWS Glue.

Pada langkah sebelumnya, Anda menginisialisasi `GlueContext` objek. Anda menggunakan objek ini untuk menemukan metode yang digunakan untuk mengkonfigurasi sumber, seperti`create_dynamic_frame.from_catalog`.

Dalam prosedur ini, Anda menulis kode berikut menggunakan`create_dynamic_frame.from_catalog`. Kode ini adalah bagian dari skrip sampel yang dihasilkan.

```
S3bucket_node1 = glueContext.create_dynamic_frame.from_catalog(
    database="yyz-tickets", table_name="tickets", transformation_ctx="S3bucket_node1"
    )
```

Untuk mengekstrak data dari sumber

1. Periksa dokumentasi untuk menemukan metode `GlueContext` untuk mengekstrak data dari sumber yang ditentukan dalam Katalog Data AWS Glue. Metode-metode ini didokumentasikan

dalamthe section called “GlueContext”. Pilih metode create_dynamic_frame.from_catalog. Panggil metode iniglueContext.

2. Periksa dokumentasi untukcreate_dynamic_frame.from_catalog. Metode ini membutuhkan database dan table_name parameter. Berikan parameter yang diperlukan untukcreate_dynamic_frame.from_catalog.

   Katalog Data AWS Glue menyimpan informasi tentang lokasi dan format data sumber Anda, dan disiapkan di bagian prasyarat. Anda tidak harus secara langsung memberikan naskah Anda dengan informasi itu.

3. Opsional - Berikan transformation_ctx parameter ke metode untuk mendukung bookmark pekerjaan. Anda dapat mempelajari tentang bookmark pekerjaan di bagian berikut,the section called “Opsional - Aktifkan bookmark pekerjaan”.

> **Note**
>
> Metode umum untuk mengekstraksi data the section called “create_dynamic_frame_from_catalog”digunakan untuk menghubungkan ke tabel di Katalog Data AWS Glue.
> Jika Anda perlu secara langsung menyediakan pekerjaan Anda dengan konfigurasi yang menjelaskan struktur dan lokasi sumber Anda, lihat the section called “create_dynamic_frame_from_options” metodenya. Anda perlu memberikan parameter yang lebih rinci yang menjelaskan data Anda daripada saat menggunakancreate_dynamic_frame.from_catalog.
> Lihat dokumentasi tambahan tentang format_options dan connection_parameters untuk mengidentifikasi parameter yang diperlukan. Untuk penjelasan tentang cara memberikan informasi skrip Anda tentang format data sumber Anda, lihatthe section called “Opsi format data”. Untuk penjelasan tentang cara memberikan informasi skrip Anda tentang lokasi data sumber Anda, lihatthe section called “Parameter koneksi”.
> Jika Anda membaca informasi dari sumber streaming, Anda memberikan pekerjaan Anda dengan informasi sumber melalui the section called “create_data_frame_from_catalog” atau the section called “create_data_frame_from_options” metode. Perhatikan bahwa metode ini mengembalikan Apache SparkDataFrames.
> Kode yang kami hasilkan memanggil create_dynamic_frame.from_catalog sementara dokumentasi referensi mengacu padacreate_dynamic_frame_from_catalog. Metode ini pada akhirnya memanggil kode yang sama, dan disertakan sehingga Anda dapat menulis

kode yang lebih bersih. Anda dapat memverifikasi ini dengan melihat sumber pembungkus Python kami, tersedia di. aws-glue-libs

Langkah 4. Transformasi data dengan AWS Glue

Setelah mengekstrak data sumber dalam proses ETL, Anda perlu menjelaskan bagaimana Anda ingin mengubah data Anda. Anda memberikan informasi ini dengan membuat node Transform di editor visual AWS Glue Studio.

Pada langkah ini, Anda memberikan `ApplyMapping` metode dengan peta nama dan jenis bidang saat ini dan yang diinginkan untuk mengubah Anda`DynamicFrame`.

Anda melakukan transformasi berikut.

- Jatuhkan empat `location` dan `province` kunci.
- Ubah nama `officer` menjadi`officer_name`.
- Ubah jenis `ticket_number` dan `set_fine_amount` ke`float`.

`create_dynamic_frame.from_catalog`menyediakan Anda dengan `DynamicFrame` objek. A `DynamicFrame` mewakili kumpulan data di AWS Glue. AWS Glue transforms adalah operasi yang berubah`DynamicFrames`.

> **Note**
>
> Apa itu`DynamicFrame`?
> A `DynamicFrame` adalah abstraksi yang memungkinkan Anda menghubungkan kumpulan data dengan deskripsi nama dan jenis entri dalam data. Di Apache Spark, abstraksi serupa ada yang disebut a. DataFrame Untuk penjelasannya DataFrames, lihat Spark SQL Guide. Dengan`DynamicFrames`, Anda dapat mendeskripsikan skema kumpulan data secara dinamis. Pertimbangkan kumpulan data dengan kolom harga, di mana beberapa entri menyimpan harga sebagai string, dan lainnya menyimpan harga sebagai ganda. AWS Glue menghitung skema on-the-fly —itu menciptakan catatan yang menggambarkan diri untuk setiap baris.
> Bidang yang tidak konsisten (seperti harga) secara eksplisit direpresentasikan dengan type (`ChoiceType`) dalam skema untuk frame. Anda dapat mengatasi bidang yang tidak konsisten dengan menjatuhkannya `DropFields` atau menyelesaikannya. `ResolveChoice`

Ini adalah transformasi yang tersedia di. `DynamicFrame` Anda kemudian dapat menulis data Anda kembali ke danau data Anda dengan`writeDynamicFrame`.
Anda dapat memanggil banyak transformasi yang sama dari metode di `DynamicFrame` kelas, yang dapat menyebabkan skrip yang lebih mudah dibaca. Untuk informasi selengkapnya tentang `DynamicFrame`, lihat the section called “DynamicFrame”.

Dalam prosedur ini, Anda menulis kode berikut menggunakan`ApplyMapping`. Kode ini adalah bagian dari skrip sampel yang dihasilkan.

```
ApplyMapping_node2 = ApplyMapping.apply(
    frame=S3bucket_node1,
    mappings=[
        ("tag_number_masked", "string", "tag_number_masked", "string"),
        ("date_of_infraction", "string", "date_of_infraction", "string"),
        ("ticket_date", "string", "ticket_date", "string"),
        ("ticket_number", "decimal", "ticket_number", "float"),
        ("officer", "decimal", "officer_name", "decimal"),
        ("infraction_code", "decimal", "infraction_code", "decimal"),
        ("infraction_description", "string", "infraction_description", "string"),
        ("set_fine_amount", "decimal", "set_fine_amount", "float"),
        ("time_of_infraction", "decimal", "time_of_infraction", "decimal"),
    ],
    transformation_ctx="ApplyMapping_node2",
)
```

Untuk mengubah data dengan AWS Glue

1. Periksa dokumentasi untuk mengidentifikasi transformasi untuk mengubah dan menjatuhkan bidang. Lihat perinciannya di the section called “GlueTransform”. Pilih `ApplyMapping` transformasi. Untuk informasi selengkapnya tentang `ApplyMapping`, lihat the section called “ApplyMapping”. Panggil `apply` objek `ApplyMapping` transformasi.

   > **Note**
   >
   > Apa `ApplyMapping`?
   > `ApplyMapping`mengambil `DynamicFrame` dan mengubahnya. Dibutuhkan daftar tupel yang mewakili transformasi di lapangan — sebuah “pemetaan”. Dua elemen tupel pertama, nama bidang dan jenis, digunakan untuk mengidentifikasi bidang dalam bingkai. Dua parameter kedua juga merupakan nama bidang dan jenis.

ApplyMapping mengubah bidang sumber ke nama target dan ketik baru`DynamicFrame`, yang dikembalikan. Bidang yang tidak disediakan dijatuhkan dalam nilai pengembalian. Daripada memanggil`apply`, Anda dapat memanggil transformasi yang sama dengan `apply_mapping` metode pada `DynamicFrame` untuk membuat kode yang lebih lancar dan dapat dibaca. Untuk informasi selengkapnya, lihat the section called "apply_mapping".

2. Periksa dokumentasi `ApplyMapping` untuk mengidentifikasi parameter yang diperlukan. Lihat the section called "ApplyMapping". Anda akan menemukan bahwa metode ini membutuhkan `frame` dan `mappings` parameter. Berikan parameter yang diperlukan untuk`ApplyMapping`.
3. Opsional — Berikan `transformation_ctx` metode untuk mendukung bookmark pekerjaan. Anda dapat mempelajari tentang bookmark pekerjaan di bagian berikut,the section called "Opsional - Aktifkan bookmark pekerjaan".

> **Note**
>
> Fungsionalitas Apache Spark
> Kami menyediakan transformasi untuk merampingkan alur kerja ETL dalam pekerjaan Anda. Anda juga memiliki akses ke perpustakaan yang tersedia dalam program Spark di pekerjaan Anda, dibangun untuk tujuan yang lebih umum. Untuk menggunakannya, Anda mengonversi antara `DynamicFrame` dan`DataFrame`.
> Anda dapat membuat `DataFrame` dengan the section called "toDF". Kemudian, Anda dapat menggunakan metode yang tersedia di DataFrame untuk mengubah dataset Anda. Untuk informasi lebih lanjut tentang metode ini, lihat DataFrame. Anda kemudian dapat mengonversi mundur dengan the section called "fromDF" menggunakan operasi AWS Glue untuk memuat bingkai Anda ke target.

Langkah 5. Memuat data ke target

Setelah Anda mengubah data Anda, Anda biasanya menyimpan data yang diubah di tempat yang berbeda dari sumbernya. Anda melakukan operasi ini dengan membuat node target di editor visual AWS Glue Studio.

Pada langkah ini, Anda menyediakan `write_dynamic_frame.from_options` metode `aconnection_type`,, `connection_optionsformat`, dan `format_options` untuk memuat data ke dalam bucket target di Amazon S3.

Pada Langkah 1, Anda menginisialisasi `GlueContext` objek. Di AWS Glue, di sinilah Anda akan menemukan metode yang digunakan untuk mengkonfigurasi target, seperti sumber.

Dalam prosedur ini, Anda menulis kode berikut menggunakan`write_dynamic_frame.from_options`. Kode ini adalah bagian dari skrip sampel yang dihasilkan.

```
S3bucket_node3 = glueContext.write_dynamic_frame.from_options(
    frame=ApplyMapping_node2,
    connection_type="s3",
    format="glueparquet",
    connection_options={"path": "s3://DOC-EXAMPLE-BUCKET", "partitionKeys": []},
    format_options={"compression": "gzip"},
    transformation_ctx="S3bucket_node3",
    )
```

Untuk memuat data ke target

1. Periksa dokumentasi untuk menemukan metode untuk memuat data ke dalam bucket Amazon S3 target. Metode-metode ini didokumentasikan dalamthe section called “GlueContext”. Pilih the section called “write_dynamic_frame_from_options” metodenya. Panggil metode ini`glueContext`.

   > **Note**
   >
   > Metode umum untuk memuat data `write_dynamic_frame.from_options`adalah metode yang paling umum digunakan untuk memuat data. Ini mendukung semua target yang tersedia di AWS Glue.
   > Jika Anda menulis ke target JDBC yang ditentukan dalam koneksi AWS Glue, gunakan metode inithe section called “write_dynamic_frame_from_jdbc_conf”. AWS Koneksi Glue menyimpan informasi tentang cara menghubungkan ke sumber data. Ini menghilangkan kebutuhan untuk memberikan informasi ituconnection_options. Namun, Anda masih perlu menggunakannya `connection_options` untuk menyediakandbtable. `write_dynamic_frame.from_catalog`bukan metode umum untuk memuat data. Metode ini memperbarui Katalog Data AWS Glue tanpa memperbarui kumpulan data yang mendasarinya, dan digunakan dalam kombinasi dengan proses lain yang mengubah kumpulan data yang mendasarinya. Untuk informasi selengkapnya, lihat the section called “Memperbarui skema dan menambahkan partisi baru”.

2. Periksa dokumentasi untukthe section called “write_dynamic_frame_from_options”. Metode ini membutuhkan`frame`,`connection_type`,`format`,`connection_options`, dan`format_options`. Panggil metode ini`glueContext`.

   a. Lihat dokumentasi tambahan tentang `format_options` dan `format` untuk mengidentifikasi parameter yang Anda butuhkan. Untuk penjelasan tentang format data, lihatthe section called “Opsi format data”.

   b. Lihat dokumentasi tambahan tentang `connection_type` dan `connection_options` untuk mengidentifikasi parameter yang Anda butuhkan. Untuk penjelasan tentang koneksi, lihatthe section called “Parameter koneksi”.

   c. Berikan parameter yang diperlukan untuk`write_dynamic_frame.from_options`. Metode ini memiliki konfigurasi yang mirip dengan`create_dynamic_frame.from_options`.

3. Opsional — Menyediakan `transformation_ctx` `write_dynamic_frame.from_options` untuk mendukung bookmark pekerjaan. Anda dapat mempelajari tentang bookmark pekerjaan di bagian berikut,the section called “Opsional - Aktifkan bookmark pekerjaan”.

Langkah 6. Komit **Job** objek

Anda menginisialisasi `Job` objek di Langkah 1. Anda perlu menyimpulkan siklus hidupnya secara manual di akhir skrip Anda. Fitur opsional tertentu membutuhkan ini agar berfungsi dengan baik. Pekerjaan ini dilakukan di belakang layar di AWS Glue Studio.

Pada langkah ini, panggil `commit` metode pada `Job` objek.

Dalam prosedur ini, Anda menulis kode berikut. Kode ini adalah bagian dari skrip sampel yang dihasilkan.

```
job.commit()
```

Untuk melakukan **`Job`** objek

1. Jika Anda belum melakukan ini, lakukan langkah-langkah opsional yang diuraikan di bagian sebelumnya untuk disertakan`transformation_ctx`.
2. Panggil `commit`.

Opsional - Aktifkan bookmark pekerjaan

Di setiap langkah sebelumnya, Anda telah diinstruksikan untuk mengatur `transformation_ctx` parameter. Ini terkait dengan fitur yang disebut bookmark pekerjaan.

Dengan bookmark pekerjaan, Anda dapat menghemat waktu dan uang dengan pekerjaan yang berjalan secara berulang, terhadap kumpulan data di mana pekerjaan sebelumnya dapat dengan mudah dilacak. Bookmark Job melacak kemajuan transformasi AWS Glue di seluruh kumpulan data dari proses sebelumnya. Dengan melacak di mana proses sebelumnya berakhir, AWS Glue dapat membatasi pekerjaannya ke baris yang belum diproses sebelumnya. Untuk informasi selengkapnya tentang bookmark pekerjaan, lihatthe section called “Melacak data yang diproses menggunakan bookmark pekerjaan”.

Untuk mengaktifkan bookmark pekerjaan, pertama-tama tambahkan `transformation_ctx` pernyataan ke dalam fungsi yang kami sediakan, seperti yang dijelaskan dalam contoh sebelumnya. Status bookmark Job dipertahankan di seluruh proses. `transformation_ctx`parameter adalah kunci yang digunakan untuk mengakses status itu. Sendiri, pernyataan ini tidak akan melakukan apa-apa. Anda juga perlu mengaktifkan fitur dalam konfigurasi untuk pekerjaan Anda.

Dalam prosedur ini, Anda mengaktifkan bookmark pekerjaan menggunakan. AWS Management Console

Untuk mengaktifkan bookmark pekerjaan

1. Arahkan ke bagian Detail pekerjaan di pekerjaan Anda yang sesuai.
2. Tetapkan bookmark Job ke Aktifkan.

Langkah 7. Jalankan kode Anda sebagai pekerjaan

Pada langkah ini, Anda menjalankan pekerjaan Anda untuk memverifikasi bahwa Anda berhasil menyelesaikan tutorial ini. Ini dilakukan dengan mengklik tombol, seperti pada editor visual AWS Glue Studio.

Untuk menjalankan kode Anda sebagai pekerjaan

1. Pilih Pekerjaan tanpa judul di bilah judul untuk diedit dan atur nama pekerjaan Anda.
2. Arahkan ke tab Detail Pekerjaan. Tetapkan pekerjaan Anda Peran IAM. Anda dapat menggunakan yang dibuat oleh AWS CloudFormation template dalam prasyarat untuk tutorial

Glue AWS Studio. Jika Anda telah menyelesaikan tutorial itu, itu harus tersedia sebagai`AWS Glue StudioRole`.

3. Pilih Simpan untuk menyimpan skrip Anda.
4. Pilih Jalankan untuk menjalankan pekerjaan Anda.
5. Arahkan ke tab Runs untuk memverifikasi bahwa pekerjaan Anda selesai.
6. Arahkan ke *`DOC-EXAMPLE-BUCKET, target`* untuk. `write_dynamic_frame.from_options` Konfirmasikan bahwa output sesuai dengan harapan Anda.

Untuk informasi selengkapnya tentang mengonfigurasi dan mengelola pekerjaan, lihatthe section called “Menyediakan skrip kustom Anda sendiri”.

Informasi selengkapnya

Pustaka dan metode Apache Spark tersedia dalam skrip Glue AWS . Anda dapat melihat dokumentasi Spark untuk memahami apa yang dapat Anda lakukan dengan pustaka yang disertakan. Untuk informasi selengkapnya, lihat bagian contoh dari repositori sumber Spark.

AWS Glue 2.0+ mencakup beberapa pustaka Python umum secara default. Ada juga mekanisme untuk memuat dependensi Anda sendiri ke dalam pekerjaan AWS Glue di lingkungan Scala atau Python. Untuk informasi tentang dependensi Python, lihat. the section called “Pustaka Python”

Untuk contoh lebih lanjut tentang cara menggunakan fitur AWS Glue di Python, lihat. the section called “Sampel Python” Pekerjaan Scala dan Python memiliki paritas fitur, jadi contoh Python kami akan memberi Anda beberapa pemikiran tentang cara melakukan pekerjaan serupa di Scala.

Program skrip AWS Glue ETL di PySpark

Anda dapat menemukan contoh kode Python dan utilitas untuk AWS Glue di repositori AWS Glue sampel di situs web. GitHub

Menggunakan Python dengan AWS Glue

AWS Glue mendukung perpanjangan dialek PySpark Python untuk pekerjaan scripting extract, transform, and load (ETL). Bagian ini menjelaskan cara menggunakan Python dalam skrip ETL dan dengan API. AWS Glue

- Menyiapkan untuk menggunakan Python dengan AWS Glue

- Memanggil AWS Glue API dengan Python
- Menggunakan pustaka Python dengan AWS Glue
- AWS GlueContoh kode Python

AWS Glue PySpark ekstensi

AWS Gluetelah membuat ekstensi berikut untuk dialek PySpark Python.

- Mengakses parameter menggunakan `getResolvedOptions`
- PySpark jenis ekstensi
- DynamicFrame kelas
- DynamicFrameCollection kelas
- DynamicFrameWriter kelas
- DynamicFrameReader kelas
- GlueContext kelas

AWS Glue PySpark mengubah

AWS Gluetelah menciptakan Kelas transformasi berikut untuk digunakan dalam operasi PySpark ETL.

- GlueTransform kelas dasar
- ApplyMapping kelas
- DropFields kelas
- DropNullFields kelas
- ErrorsAsDynamicFrame kelas
- FillMissingValues kelas
- Kelas filter
- FindIncrementalMatches kelas
- FindMatches kelas
- FlatMap kelas
- Bergabunglah dengan kelas
- Kelas peta

- MapToCollection kelas
- mergeDynamicFrame
- Relationalisasi kelas
- RenameField kelas
- ResolveChoice kelas
- SelectFields kelas
- SelectFromCollection kelas
- Kelas keran
- SplitFields kelas
- SplitRows kelas
- Kelas buka kotak
- UnnestFrame kelas

Menyiapkan untuk menggunakan Python dengan AWS Glue

Gunakan Python untuk mengembangkan skrip ETL Anda untuk tugas Spark. Versi Python yang didukung untuk pekerjaan ETL bergantung pada AWS Glue versi pekerjaan. Untuk informasi lebih lanjut tentang AWS Glue versi, lihatGlue version job property.

Untuk mengatur sistem Anda untuk menggunakan Python dengan AWS Glue

Ikuti langkah-langkah ini untuk menginstal Python dan untuk dapat menjalankan API. AWS Glue

1. Jika Anda belum menginstal Python, unduh dan instal dari Halaman unduhan Python.org.
2. Instal AWS Command Line Interface (AWS CLI) seperti yang dijelaskan dalam dokumentasi CLI AWS.

   AWS CLI tidak secara langsung diperlukan untuk menggunakan Python. Walau bagaimanapun, menginstal dan mengkonfigurasinya adalah cara mudah untuk menyiapkan AWS dengan kredensial akun Anda dan untuk memverifikasi bahwa mereka bekerja.
3. Instal SDK for Python (Boto 3) AWS, sebagaimana yang didokumentasikan dalam Boto3 Quickstart.

   API sumber daya Boto 3 belum tersedia untukAWS Glue. Saat ini, hanya API klien Boto 3 yang dapat digunakan.

Untuk informasi selengkapnya tentang Boto 3, lihat Memulai SDK for Python (Boto3) AWS.

Anda dapat menemukan contoh kode Python dan utilitas untuk AWS Glue di repositori AWS Glue sampel di situs web. GitHub

Memanggil AWS Glue API dengan Python

Perhatikan bahwa API sumber daya Boto 3 belum tersedia untukAWS Glue. Saat ini, hanya API klien Boto 3 yang dapat digunakan.

AWS GlueNama API dalam Python

AWSNama Glue API di Java dan bahasa pemrograman lainnya umumnya CamelCased. Namun, ketika dipanggil dari Python, nama-nama generik ini diubah menjadi huruf kecil, di mana bagian-bagian dari nama tersebut dipisahkan oleh karakter garis bawah untuk membuatnya menjadi lebih "Pythonic". Dalam dokumentasi AWS Glue API referensi, nama-nama Pythonic ini tercantum dalam tanda kurung setelah nama generik. CamelCased

Namun, meskipun nama AWS Glue API itu sendiri diubah menjadi huruf kecil, nama parameternya tetap dikapitalisasi. Penting untuk diingat ini, karena parameter harus diteruskan dengan nama saat memanggil AWS Glue API, seperti yang dijelaskan di bagian berikut.

Melewati dan mengakses parameter Python di AWS Glue

Dalam panggilan Python ke AWS Glue API, yang terbaik adalah meneruskan parameter secara eksplisit berdasarkan nama. Misalnya:

```
job = glue.create_job(Name='sample', Role='Glue_DefaultRole',
                      Command={'Name': 'glueetl',
                               'ScriptLocation': 's3://my_script_bucket/scripts/
my_etl_script.py'})
```

Hal ini akan membantu untuk memahami bahwa Python menciptakan kamus tupel nama/nilai yang Anda tentukan sebagai argumen untuk skrip ETL dalam Struktur Job atau JobRun struktur. Boto 3 kemudian meneruskannya ke AWS Glue dalam format JSON melalui panggilan REST API. Ini berarti bahwa Anda tidak dapat mengandalkan urutan argumen ketika Anda mengaksesnya dalam skrip Anda.

Misalnya, anggaplah Anda memulai `JobRun` dalam fungsi penangan Python Lambda, dan Anda ingin menentukan beberapa parameter. Kode Anda mungkin terlihat seperti yang berikut ini:

```
from datetime import datetime, timedelta

client = boto3.client('glue')

def lambda_handler(event, context):
  last_hour_date_time = datetime.now() - timedelta(hours = 1)
  day_partition_value = last_hour_date_time.strftime("%Y-%m-%d")
  hour_partition_value = last_hour_date_time.strftime("%-H")

  response = client.start_job_run(
              JobName = 'my_test_Job',
              Arguments = {
                '--day_partition_key':   'partition_0',
                '--hour_partition_key':  'partition_1',
                '--day_partition_value':  day_partition_value,
                '--hour_partition_value': hour_partition_value } )
```

Untuk mengakses parameter ini dengan andal dalam skrip ETL Anda, tentukan dengan nama menggunakan AWS Glue `getResolvedOptions` fungsi dan kemudian akses dari kamus yang dihasilkan:

```
import sys
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv,
                          ['JOB_NAME',
                           'day_partition_key',
                           'hour_partition_key',
                           'day_partition_value',
                           'hour_partition_value'])
print "The day partition key is: ", args['day_partition_key']
print "and the day partition value is: ", args['day_partition_value']
```

Jika Anda ingin meneruskan argumen yang merupakan string JSON bersarang, untuk mempertahankan nilai parameter saat diteruskan ke pekerjaan AWS Glue ETL Anda, Anda harus menyandikan string parameter sebelum memulai pekerjaan, dan kemudian memecahkan kode string parameter sebelum mereferensikannya skrip pekerjaan Anda. Sebagai contoh, pertimbangkan string argumen berikut:

```
glue_client.start_job_run(JobName = "gluejobname", Arguments={
"--my_curly_braces_string": '{"a": {"b": {"c": [{"d": {"e": 42}}]}}}'
```

```
})
```

Untuk memberikan parameter ini dengan benar, Anda harus mengkodekan argumen sebagai string dengan encoding Base64.

```
import base64
...
sample_string='{"a": {"b": {"c": [{"d": {"e": 42}}]}}}'
sample_string_bytes = sample_string.encode("ascii")

base64_bytes = base64.b64encode(sample_string_bytes)
base64_string = base64_bytes.decode("ascii")
...
glue_client.start_job_run(JobName = "gluejobname", Arguments={
"--my_curly_braces_string": base64_bytes})
...
sample_string_bytes = base64.b64decode(base64_bytes)
sample_string = sample_string_bytes.decode("ascii")
print(f"Decoded string: {sample_string}")
...
```

Contoh: Membuat dan menjalankan pekerjaan

Contoh berikut menunjukkan bagaimana memanggil AWS Glue API menggunakan Python, untuk membuat dan menjalankan pekerjaan ETL.

Membuat dan menjalankan sebuah tugas

1. Buat instance AWS Glue klien:

   ```
   import boto3
   glue = boto3.client(service_name='glue', region_name='us-east-1',
                endpoint_url='https://glue.us-east-1.amazonaws.com')
   ```

2. Membuat sebuah tugas. Anda harus menggunakan `glueetl` sebagai nama untuk perintah ETL, seperti yang ditunjukkan dalam kode berikut:

   ```
   myJob = glue.create_job(Name='sample', Role='Glue_DefaultRole',
                           Command={'Name': 'glueetl',
                                    'ScriptLocation': 's3://my_script_bucket/
   scripts/my_etl_script.py'})
   ```

3. Mulai eksekusi baru pada tugas Anda yang Anda buat di langkah sebelumnya:

```
myNewJobRun = glue.start_job_run(JobName=myJob['Name'])
```

4. Ambil status tugas:

```
status = glue.get_job_run(JobName=myJob['Name'], RunId=myNewJobRun['JobRunId'])
```

5. Mencetak status eksekusi tugas saat ini:

```
print(status['JobRun']['JobRunState'])
```

Menggunakan pustaka Python dengan AWS Glue

AWS Gluememungkinkan Anda menginstal modul dan pustaka Python tambahan untuk digunakan dengan ETL. AWS Glue

Topik

- Menginstal modul Python tambahan dengan pip di 2.0+ AWS Glue
- Termasuk file Python dengan fitur asli PySpark
- Skrip pemrograman yang menggunakan transformasi visual
- Modul Python sudah disediakan di AWS Glue
- Pustaka ritsleting untuk dimasukkan
- Memuat pustaka Python di notebook Glue Studio AWS
- Memuat pustaka Python di titik akhir pengembangan
- Menggunakan pustaka Python dalam pekerjaan atau JobRun

Menginstal modul Python tambahan dengan pip di 2.0+ AWS Glue

AWS Gluemenggunakan Python Package Installer (pip3) untuk menginstal modul tambahan yang akan digunakan oleh ETL. AWS Glue Anda dapat menggunakan `--additional-python-modules` parameter dengan daftar modul Python yang dipisahkan koma untuk menambahkan modul baru atau mengubah versi modul yang ada. Anda dapat menginstal distribusi kustom pustaka dengan mengunggah distribusi ke Amazon S3, lalu menyertakan jalur ke objek Amazon S3 dalam daftar modul Anda.

Anda dapat meneruskan opsi tambahan ke pip3 dengan parameter. `--python-modules-installer-option` Misalnya, Anda dapat meneruskan `"--upgrade"` untuk memutakhirkan paket yang ditentukan oleh`"--additional-python-modules"`. Untuk contoh lainnya, lihat Membangun modul Python dari roda untuk beban kerja Spark ETL menggunakan Glue 2.0. AWS

Jika dependensi Python Anda secara transitif bergantung pada kode asli yang dikompilasi, Anda dapat menghadapi batasan berikut: AWS Glue tidak mendukung kompilasi kode asli di lingkungan pekerjaan. Namun, pekerjaan AWS Glue berjalan dalam lingkungan Amazon Linux 2. Anda mungkin dapat memberikan dependensi asli Anda dalam bentuk yang dikompilasi melalui Wheel yang dapat didistribusikan.

Misalnya untuk memperbarui atau menambahkan modul `scikit-learn` baru menggunakan nilai/kunci berikut: `"--additional-python-modules", "scikit-learn==0.21.3"`.

Selain itu, dalam opsi `--additional-python-modules` Anda dapat menentukan path Amazon S3 ke modul roda Python. Sebagai contoh:

```
--additional-python-modules s3://aws-glue-native-spark/tests/j4.2/ephem-3.7.7.1-cp37-cp37m-linux_x86_64.whl,s3://aws-glue-native-spark/tests/j4.2/fbprophet-0.6-py3-none-any.whl,scikit-learn==0.21.3
```

Anda menentukan `--additional-python-modules` dalam bidang Parameter Job AWS Glue konsol atau dengan mengubah argumen pekerjaan di AWS SDK. Untuk informasi selengkapnya tentang pengaturan parameter pekerjaan, lihatthe section called “Parameter Tugas”.

Termasuk file Python dengan fitur asli PySpark

AWS Gluedigunakan PySpark untuk menyertakan file Python dalam pekerjaan AWS Glue ETL. Anda akan ingin menggunakan `--additional-python-modules` untuk mengelola dependensi Anda bila tersedia. Anda dapat menggunakan parameter `--extra-py-files` pekerjaan untuk menyertakan file Python. Dependensi harus di-host di Amazon S3 dan nilai argumen harus berupa daftar jalur Amazon S3 yang dibatasi koma tanpa spasi. Fungsionalitas ini berperilaku seperti manajemen ketergantungan Python yang akan Anda gunakan dengan Spark. Untuk informasi selengkapnya tentang manajemen ketergantungan Python di Spark, lihat halaman Menggunakan Fitur PySpark Asli di dokumentasi Apache Spark. `--extra-py-files`berguna dalam kasus di mana kode tambahan Anda tidak dikemas, atau saat Anda memigrasikan program Spark dengan rantai alat yang ada untuk mengelola dependensi. Agar alat ketergantungan Anda dapat dipertahankan, Anda harus menggabungkan dependensi Anda sebelum mengirimkan.

Skrip pemrograman yang menggunakan transformasi visual

Saat Anda membuat pekerjaan AWS Glue menggunakan antarmuka visual AWS Glue Studio, Anda dapat mengubah data Anda dengan node transformasi data terkelola dan transformasi visual khusus. Untuk informasi selengkapnya tentang node transformasi data terkelola, lihatthe section called “Mengedit node transformasi data AWS Glue terkelola”. Untuk informasi selengkapnya tentang transformasi visual kustom, lihatthe section called “ Transformasi visual khusus ”. Skrip yang menggunakan transformasi visual hanya dapat dihasilkan ketika ketika pekerjaan Anda Bahasa diatur untuk menggunakan Python.

Saat membuat pekerjaan AWS Glue menggunakan transformasi visual, AWS Glue Studio akan menyertakan transformasi ini di lingkungan runtime menggunakan `--extra-py-files` parameter dalam konfigurasi pekerjaan. Untuk informasi selengkapnya tentang parameter pekerjaan, lihatthe section called “Parameter Tugas”. Saat membuat perubahan pada skrip atau lingkungan runtime yang dihasilkan, Anda harus mempertahankan konfigurasi pekerjaan ini agar skrip Anda berhasil berjalan.

Modul Python sudah disediakan di AWS Glue

Untuk mengubah versi modul yang disediakan ini, berikan versi baru dengan parameter `--additional-python-modules` pekerjaan.

AWS Glue version 2.0

AWS Glueversi 2.0 mencakup modul Python berikut di luar kotak:

- avro-python3 = 1.10.0
- awscli==1.27.60
- boto3==1.12.4
- botocore==1.15.4
- certifi==2019.11.28
- chardet==3.0.4
- klik==8.1.3
- colorama==0.4.4
- cycler==0.10.0
- Cython==0.29.15
- docutils==0.15.2

- enum34==1.1.9
- fsspec==0.6.2
- idna==2.9
- importlib-metadata== 6.0.0
- jmespath==0.9.4
- joblib==0.14.1
- kiwisolver==1.1.0
- matplotlib==3.1.3
- mpmath==1.1.0
- nltk=3,5
- numpy==1.18.1
- pandas==1.0.1
- patsy==0.5.1
- pmdarima==1.5.3
- ptvsd==4.3.2
- pyarrow==0.16.0
- pyasn1==0.4.8
- pydevd==1.9.0
- pyhocon==0.3.54
- PyMySQL=0.9.3
- pyparsing==2.4.6
- python-dateutil==2.8.1
- pytz==2019.3
- Pyyaml = 5.3.1
- regex==2022.10.31
- requests==2.23.0
- rsa = 4.7.2
- s3fs==0.4.0
- s3transfer==0.3.3

- scikit-learn==0.22.1
- scipy==1.4.1
- setuptools==45.2.0
- six==1.14.0
- Percikan = 1.0
- statsmodels==0.11.1
- subprocess32==3.5.4
- sympy==1.5.1
- tbats==1.0.9
- tqdm = 4.64.1
- mengetik-ekstensi== 4.4.0
- urllib3==1.25.8
- roda==0.35.1
- ritsleting = 3.12.0

AWS Glue versi 3.0

AWS Glueversi 3.0 mencakup modul Python berikut di luar kotak:,

- aiobotocore==1.4.2
- aiohttp = 3.8.3
- aioitertools==0.11.0
- aiosignal== 1.3.1
- batas waktu asinkron = 4.0.2
- asynctest = 0.13.0
- attrs==22.2.0
- avro-python3 = 1.10.2
- boto3 = 1.18.50
- botocore==1.21.50
- sertifikasi = 2021.5.30
- chardet==3.0.4

- charset-normalizer = 2.1.1
- klik==8.1.3
- cycler==0.10.0
- Cython==0.29.4
- dokumen==0.17.1
- enum34==1.1.10
- daftar beku==1.3.3
- fsspec==2021.8.1
- idna==2.10
- importlib-metadata== 6.0.0
- jmespath=0.10.0
- joblib==1.0.1
- kiwisolver==1.3.2
- matplotlib==3.4.3
- mpmath==1.2.1
- multidikt = 6.0.4
- nltk = 3.6.3
- numpy = 1.19.5
- pengepakan== 23.0
- panda ==1.3.2
- patsy==0.5.1
- Bantal==9.4.0
- pip=23.0
- pmdarima==1.8.2
- ptvsd==4.3.2
- pyarrow = 5.0.0
- pydevd=2.5.0
- pyhocon==0.3.58
- PyMySQL=1.0.2
- pyparsing = 2.4.7

- python-dateutil==2.8.2
- pytz==2021.1
- Pyyaml==5.4.1
- regex==2022.10.31
- requests==2.23.0
- s3fs==2021.8.1
- s3transfer== 0.5.0
- scikit-belajar==0.24.2
- scipy==1.7.1
- enam==1.16.0
- Percikan = 1.0
- statsmodels==0.12.2
- subprocess32==3.5.4
- simpati=1,8
- tbats==1.1.0
- threadpoolctl = 3.1.0
- tqdm = 4.64.1
- typing_extensions==4.4.0
- urllib3==1.25.11
- roda==0.37.0
- bungkus=1.14.1
- yarl=1.8.2
- ritsleting = 3.12.0

AWS Glue versi 4.0

AWS Glueversi 4.0 mencakup modul Python berikut di luar kotak:

- aiobotocore==2.4.1
- aiohttp = 3.8.3
- aioitertools==0.11.0

- aiosignal== 1.3.1
- batas waktu asinkron = 4.0.2
- asynctest = 0.13.0
- attrs==22.2.0
- avro-python3 = 1.10.2
- boto3 = 1.24.70
- botocore==1.27.59
- sertifikasi = 2021.5.30
- chardet==3.0.4
- charset-normalizer = 2.1.1
- klik==8.1.3
- cycler==0.10.0
- Cython==0.29.32
- dokumen==0.17.1
- enum34==1.1.10
- daftar beku==1.3.3
- fsspec==2021.8.1
- idna==2.10
- importlib-metadata== 5.0.0
- jmespath=0.10.0
- joblib==1.0.1
- kaleido==0.2.1
- kiwisolver==1.4.4
- matplotlib==3.4.3
- mpmath==1.2.1
- multidikt = 6.0.4
- nltk== 3.7
- numpy = 1.23.5
- pengepakan== 23.0
- panda ==1.5.1

- patsy==0.5.1
- Bantal==9.4.0
- pip=23.0.1
- plot== 5.16.0
- pmdarima==2.0.1
- ptvsd==4.3.2
- pyarrow = 10.0.0
- pydevd=2.5.0
- pyhocon==0.3.58
- PyMySQL=1.0.2
- pyparsing = 2.4.7
- python-dateutil==2.8.2
- pytz==2021.1
- Pyyaml==6.0.1
- regex==2022.10.31
- requests==2.23.0
- s3fs==2022.11.0
- s3transfer==0.6.0
- scikit-belajar==1.1.3
- scipy==1.9.3
- setuptool== 49.1.3
- enam==1.16.0
- statsmodels==0.13.5
- subprocess32==3.5.4
- simpati=1,8
- tbats==1.1.0
- threadpoolctl = 3.1.0
- tqdm = 4.64.1
- typing_extensions==4.4.0

- urllib3==1.25.11
- roda==0.37.0
- bungkus=1.14.1
- yarl=1.8.2
- ritsleting = 3.10.0

Pustaka ritsleting untuk dimasukkan

Kecuali sebuah perpustakaan terkandung dalam satu file `.py`, ia harus dikemas dalam sebuah arsip `.zip`. Direktori paket harus berada pada akar dari arsip, dan harus berisi file `__init__.py` untuk paket. Python kemudian akan dapat mengimpor paket tersebut dengan cara normal.

Jika perpustakaan anda hanya terdiri dari sebuah modul Python tunggal dalam satu file `.py`, Anda tidak perlu menempatkannya dalam sebuah file `.zip`.

Memuat pustaka Python di notebook Glue Studio AWS

Untuk menentukan library Python di notebook AWS Glue Studio, lihat Menginstal modul Python tambahan.

Memuat pustaka Python di titik akhir pengembangan

Jika Anda menggunakan set perpustakaan yang berbeda untuk skrip ETL yang berbeda, maka Anda dapat mengatur titik akhir pengembangan terpisah untuk setiap set, atau Anda dapat menimpa file `.zip` perpustakaan yang dimuat titik akhir pengembangan Anda setiap kali Anda beralih skrip.

Anda dapat menggunakan konsol untuk menentukan satu atau beberapa perpustakaan file .zip untuk sebuah titik akhir pengembangan ketika Anda membuatnya. Setelah menetapkan nama dan IAM role, pilih Perpustakaan Skrip dan parameter tugas (opsional) dan masukkan path Amazon S3 lengkap ke file `.zip` perpustakaan Anda dalam kotak path perpustakaan Python. Sebagai contoh:

```
s3://bucket/prefix/site-packages.zip
```

Jika Anda mau, Anda dapat menentukan beberapa path lengkap ke file, memisahkannya dengan koma tetapi tanpa spasi, seperti ini:

```
s3://bucket/prefix/lib_A.zip,s3://bucket_B/prefix/lib_X.zip
```

Jika Anda kemudian memperbarui file `.zip`, Anda dapat menggunakan konsol untuk mengimpornya kembali ke titik akhir pengembangan Anda. Arahkan ke titik akhir developer yang dimaksud, centang kotak di sampingnya, dan pilih Memperbarui perpustakaan ETL dari menu Tindakan.

Dengan cara yang sama, Anda dapat menentukan file pustaka menggunakan AWS Glue API. Ketika Anda membuat sebuah titik akhir pengembangan dengan memanggil CreateDevEndpoint tindakan (Python: create_dev_endpoint), Anda dapat menentukan satu atau beberapa path lengkap ke perpustakaan di parameter `ExtraPythonLibsS3Path`, dalam panggilan yang terlihat seperti ini:

```
dep = glue.create_dev_endpoint(
             EndpointName="testDevEndpoint",
             RoleArn="arn:aws:iam::123456789012",
             SecurityGroupIds="sg-7f5ad1ff",
             SubnetId="subnet-c12fdba4",
             PublicKey="ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCtp04H/y...",
             NumberOfNodes=3,
             ExtraPythonLibsS3Path="s3://bucket/prefix/lib_A.zip,s3://bucket_B/prefix/
lib_X.zip")
```

Ketika Anda memperbarui titik akhir pengembangan, Anda juga dapat memperbarui perpustakaan yang dimuatnya menggunakan objek DevEndpointCustomLibraries dan pengaturan parameter `UpdateEtlLibraries` untuk `True` ketika memanggil UpdateDevEndpoint (update_dev_endpoint).

Menggunakan pustaka Python dalam pekerjaan atau JobRun

Ketika Anda membuat sebuah Tugas baru pada konsol, Anda dapat menentukan satu atau beberapa perperpustakaanan file .zip dengan memilih Perpustakaan skrip dan parameter tugas (opsional) dan memasukkan path perpustakaan Amazon S3 lengkap dengan cara yang sama saat membuat titik akhir pengembangan:

```
s3://bucket/prefix/lib_A.zip,s3://bucket_B/prefix/lib_X.zip
```

Jika Anda memanggil CreateJob (create_job), Anda dapat menentukan satu atau beberapa path lengkap ke perpustakaan default menggunakan parameter `--extra-py-files` default, seperti ini:

```
job = glue.create_job(Name='sampleJob',
                      Role='Glue_DefaultRole',
                      Command={'Name': 'glueetl',
                               'ScriptLocation': 's3://my_script_bucket/scripts/
my_etl_script.py'},
```

```
                        DefaultArguments={'--extra-py-files': 's3://bucket/prefix/
lib_A.zip,s3://bucket_B/prefix/lib_X.zip'})
```

Kemudian ketika Anda memulai JobRun, Anda dapat mengganti pengaturan pustaka default dengan yang berbeda:

```
runId = glue.start_job_run(JobName='sampleJob',
                           Arguments={'--extra-py-files': 's3://bucket/prefix/
lib_B.zip'})
```

AWS GlueContoh kode Python

- Contoh kode: Bergabung dan menghubungkan data
- Contoh kode: Persiapan data menggunakan ResolveChoice, Lambda, dan ApplyMapping

Contoh kode: Bergabung dan menghubungkan data

Contoh ini menggunakan set data yang diunduh dari http://everypolitician.org/ ke bucket `sample-dataset` di Amazon Simple Storage Service (Amazon S3): `s3://awsglue-datasets/examples/us-legislators/all`. Set data tersebut berisi data dalam format JSON tentang legislator Amerika Serikat dan jabatan yang mereka telah jabat di DPR AS dan Senat, dan telah dimodifikasi sedikit dan dibuat tersedia dalam sebuah bucket Amazon S3 publik untuk tujuan tutorial ini.

Anda dapat menemukan kode sumber untuk contoh ini dalam `join_and_relationalize.py` file di repositori AWS Glue sampel di situs web. GitHub

Dengan menggunakan data ini, tutorial ini menunjukkan cara untuk melakukan hal berikut:

- Gunakan AWS Glue crawler untuk mengklasifikasikan objek yang disimpan di bucket Amazon S3 publik dan simpan skema mereka ke dalam AWS Katalog Data Glue.
- Memeriksa metadata tabel dan skema yang dihasilkan dari perayapan.
- Menulis sebuah skrip extract, transform, and load (ETL) Python yang menggunakan metadata tersebut dalam Katalog Data untuk melakukan hal berikut:
  - Menggabungkan data dalam file sumber yang berbeda bersama-sama ke dalam sebuah tabel data tunggal (yakni, denormalisasi data).
  - Mem-filter tabel yang digabungkan ke dalam tabel terpisah berdasarkan jenis legislator.

- Menuliskan data yang dihasilkan untuk memisahkan file Apache Parquet untuk analisis nanti.

Cara yang lebih disukai untuk men-debug Python PySpark atau skrip saat berjalan adalah dengan menggunakan Notebook AWS di Glue Studio. AWS

Langkah 1: Merayapi data di bucket Amazon S3

1. Masuk ke AWS Management Console, dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/.
2. Mengikuti langkah-langkahnyaMengkonfigurasi crawler, buat crawler baru yang dapat merayapi `s3://awsglue-datasets/examples/us-legislators/all` kumpulan data ke dalam database bernama `legislators` dalam Katalog Data AWS Glue. Contoh data sudah ada di bucket Amazon S3 ini.
3. Jalankan crawler baru tersebut, dan kemudian periksa basis data `legislators`.

   Crawler tersebut membuat tabel metadata berikut:

   - `persons_json`
   - `memberships_json`
   - `organizations_json`
   - `events_json`
   - `areas_json`
   - `countries_r_json`

   Ini adalah koleksi tabel semi-dinormalisasi yang berisi legislator dan riwayat mereka.

Langkah 2: Tambahkan skrip boilerplate ke notebook endpoint pengembangan

Rekatkan skrip boilerplate berikut ke notebook endpoint pengembangan untuk mengimpor AWS Glue pustaka yang Anda butuhkan, dan siapkan satu: `GlueContext`

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
```

```
from awsglue.job import Job

glueContext = GlueContext(SparkContext.getOrCreate())
```

Langkah 3: Periksa skema dari data dalam Katalog Data

Selanjutnya, Anda dapat dengan mudah membuat check a DynamicFrame dari AWS Glue Data Catalog, dan memeriksa skema data. Misalnya, untuk melihat skema tabel `persons_json`, tambahkan hal berikut di notebook Anda:

```
persons = glueContext.create_dynamic_frame.from_catalog(
             database="legislators",
             table_name="persons_json")
print "Count: ", persons.count()
persons.printSchema()
```

Berikut adalah output dari panggilan cetak:

```
Count:  1961
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
```

```
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

Setiap orang dalam tabel adalah anggota dari beberapa badan kongres AS.

Untuk melihat skema tabel `memberships_json`, ketik berikut ini:

```
memberships = glueContext.create_dynamic_frame.from_catalog(
                 database="legislators",
                 table_name="memberships_json")
print "Count: ", memberships.count()
memberships.printSchema()
```

Output adalah sebagai berikut:

```
Count:  10439
root
|-- area_id: string
|-- on_behalf_of_id: string
|-- organization_id: string
|-- role: string
|-- person_id: string
|-- legislative_period_id: string
|-- start_date: string
|-- end_date: string
```

`organizations` adalah partai dan dua majelis Kongres, Senat dan Dewan Perwakilan Rakyat. Untuk melihat skema tabel `organizations_json`, ketik berikut ini:

```
orgs = glueContext.create_dynamic_frame.from_catalog(
           database="legislators",
```

```
           table_name="organizations_json")
print "Count: ", orgs.count()
orgs.printSchema()
```

Output adalah sebagai berikut:

```
Count:  13
root
|-- classification: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- id: string
|-- name: string
|-- seats: int
|-- type: string
```

Langkah 4: Filter data

Berikutnya, simpan hanya bidang yang Anda inginkan, dan ubah nama `id` menjadi `org_id`. Set data cukup kecil sehingga Anda dapat melihat semuanya.

`toDF()` mengkonversi `DynamicFrame` menjadi `DataFrame` Apache Spark, sehingga Anda dapat menerapkan transformasi yang sudah ada di Apache Spark SQL:

```
orgs = orgs.drop_fields(['other_names',
                         'identifiers']).rename_field(
                             'id', 'org_id').rename_field(
                                'name', 'org_name')
```

```
orgs.toDF().show()
```

Berikut ini menunjukkan outputnya:

```
+--------------+--------------------+--------------------+--------------------+-----
+-----------+--------------------+
|classification|              org_id|            org_name|               links|seats|
     type|               image|
+--------------+--------------------+--------------------+--------------------+-----
+-----------+--------------------+
|         party|            party/al|                  AL|                null| null|
     null|                null|
|         party|      party/democrat|            Democrat|[[website,http://...| null|
     null|https://upload.wi...|
|         party|party/democrat-li...|    Democrat-Liberal|[[website,http://...| null|
     null|                null|
|   legislature|d56acebe-8fdc-47b...|House of Represen...|                null|  435|
lower house|                null|
|         party|   party/independent|         Independent|                null| null|
     null|                null|
|         party|party/new_progres...|     New Progressive|[[website,http://...| null|
     null|https://upload.wi...|
|         party|party/popular_dem...|    Popular Democrat|[[website,http://...| null|
     null|                null|
|         party|    party/republican|          Republican|[[website,http://...| null|
     null|https://upload.wi...|
|         party|party/republican-...|Republican-Conser...|[[website,http://...| null|
     null|                null|
|         party|      party/democrat|            Democrat|[[website,http://...| null|
     null|https://upload.wi...|
|         party|   party/independent|         Independent|                null| null|
     null|                null|
|         party|    party/republican|          Republican|[[website,http://...| null|
     null|https://upload.wi...|
|   legislature|8fa6c3d2-71dc-478...|             Senate|                null|  100|
upper house|                null|
+--------------+--------------------+--------------------+--------------------+-----
+-----------+--------------------+
```

Ketik berikut ini untuk melihat `organizations` yang muncul di `memberships`:

```
memberships.select_fields(['organization_id']).toDF().distinct().show()
```

Berikut ini menunjukkan outputnya:

```
+--------------------+
|     organization_id|
+--------------------+
|d56acebe-8fdc-47b...|
|8fa6c3d2-71dc-478...|
+--------------------+
```

Langkah 5: Satukan semuanya

Sekarang, gunakan AWS Glue untuk bergabung dengan tabel relasional ini dan buat satu tabel sejarah lengkap legislator `memberships` dan yang sesuai. `organizations`

1. Pertama, gabungkan `persons` dan `memberships` pada `id` dan `person_id`.
2. Selanjutnya, gabungkan hasilnya dengan `orgs` pada `org_id` dan `organization_id`.
3. Kemudian, buang bidang-bidang redundan-nya, `person_id` dan `org_id`.

Anda dapat melakukan semua operasi ini dalam satu baris kode (diperpanjang):

```
l_history = Join.apply(orgs,
                       Join.apply(persons, memberships, 'id', 'person_id'),
                       'org_id', 'organization_id').drop_fields(['person_id',
 'org_id'])
print "Count: ", l_history.count()
l_history.printSchema()
```

Output adalah sebagai berikut:

```
Count:  10439
root
|-- role: string
|-- seats: int
|-- org_name: string
|-- links: array
```

```
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- type: string
|-- sort_name: string
|-- area_id: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- on_behalf_of_id: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- name: string
|-- birth_date: string
|-- organization_id: string
|-- gender: string
|-- classification: string
|-- death_date: string
|-- legislative_period_id: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- image: string
|-- given_name: string
|-- family_name: string
|-- id: string
|-- start_date: string
|-- end_date: string
```

Anda sekarang memiliki tabel akhir yang dapat Anda gunakan untuk analisis. Anda dapat menuliskannya dalam format yang ringkas dan efisien untuk analitik — yaitu Parket — yang dapat Anda jalankan SQL, AWS Glue Amazon Athena, atau Amazon Redshift Spectrum.

Panggilan berikut menulis tabel di beberapa file untuk mendukung pembacaan paralel cepat ketika melakukan analisis kemudian:

```
glueContext.write_dynamic_frame.from_options(frame = l_history,
          connection_type = "s3",
          connection_options = {"path": "s3://glue-sample-target/output-dir/
legislator_history"},
          format = "parquet")
```

Untuk menempatkan semua data riwayat ke dalam satu file, Anda harus mengubahnya menjadi sebuah bingkai data, melakukan pemartisian ulang, dan menuliskannya:

```
s_history = l_history.toDF().repartition(1)
s_history.write.parquet('s3://glue-sample-target/output-dir/legislator_single')
```

Atau, jika Anda ingin memisahkannya berdasarkan Senat dan DPR:

```
l_history.toDF().write.parquet('s3://glue-sample-target/output-dir/legislator_part',
                               partitionBy=['org_name'])
```

Langkah 6: Mengubah data untuk database relasional

AWS Gluemembuatnya mudah untuk menulis data ke database relasional seperti Amazon Redshift, bahkan dengan data semi-terstruktur. Ia menawarkan transformasi `relationalize`, yang meratakan `DynamicFrames` tidak peduli seberapa kompleks objek dalam bingkai.

Dengan menggunakan `l_history` `DynamicFrame` dalam contoh ini, masukkan nama dari tabel akar (`hist_root`) dan path kerja sementara untuk `relationalize`. Ini akan mengembalikan `DynamicFrameCollection`. Anda kemudian dapat mencantumkan nama-nama `DynamicFrames` dalam koleksi itu:

```
dfc = l_history.relationalize("hist_root", "s3://glue-sample-target/temp-dir/")
dfc.keys()
```

Berikut ini adalah hasil dari panggilan `keys`:

```
[u'hist_root', u'hist_root_contact_details', u'hist_root_links',
 u'hist_root_other_names', u'hist_root_images', u'hist_root_identifiers']
```

`Relationalize` memecahkan tabel riwayat ke dalam enam tabel baru: tabel akar yang berisi catatan untuk setiap objek dalam `DynamicFrame`, dan tabel tambahan untuk array. Array penanganan dalam basis data relasional sering bersifat suboptimal, terutama karena array tersebut menjadi besar. Memisahkan array ke dalam tabel yang berbeda membuat kueri lebih cepat.

Selanjutnya, lihat pemisahan dengan memeriksa `contact_details`:

```
l_history.select_fields('contact_details').printSchema()
dfc.select('hist_root_contact_details').toDF().where("id = 10 or id =
 75").orderBy(['id','index']).show()
```

Berikut ini adalah hasil dari panggilan `show`:

```
root
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
+---+-----+-----------------------+-------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-------------------------+
| 10|    0|                    fax|                         |
| 10|    1|                       |             202-225-1314|
| 10|    2|                  phone|                         |
| 10|    3|                       |             202-225-3772|
| 10|    4|                twitter|                         |
| 10|    5|                       |          MikeRossUpdates|
| 75|    0|                    fax|                         |
| 75|    1|                       |             202-225-7856|
| 75|    2|                  phone|                         |
| 75|    3|                       |             202-225-2711|
| 75|    4|                twitter|                         |
| 75|    5|                       |                SenCapito|
+---+-----+-----------------------+-------------------------+
```

Bidang `contact_details` adalah sebuah array dari struct di `DynamicFrame` asli. Setiap elemen dari array tersebut adalah baris terpisah dalam tabel tambahan, yang diindeks berdasarkan `index`. `id` di sini adalah sebuah kunci asing dalam tabel `hist_root` dengan kunci `contact_details`:

```
dfc.select('hist_root').toDF().where(
    "contact_details = 10 or contact_details = 75").select(
       ['id', 'given_name', 'family_name', 'contact_details']).show()
```

Berikut hasilnya:

```
+--------------------+----------+-----------+---------------+
|                  id|given_name|family_name|contact_details|
+--------------------+----------+-----------+---------------+
|f4fc30ee-7b42-432...|      Mike|       Ross|             10|
|e3c60f34-7d1b-4c0...|   Shelley|     Capito|             75|
+--------------------+----------+-----------+---------------+
```

Perhatikan dalam perintah ini bahwa `toDF()` dan kemudian ekspresi `where` digunakan untuk memfilter baris yang ingin Anda lihat.

Jadi, dengan menggabungkan tabel `hist_root` dengan tabel tambahan akan memungkinkan Anda melakukan hal berikut:

- Memuat data ke dalam basis data tanpa dukungan array.
- Meng-kueri setiap item individu dalam array menggunakan SQL.

Simpan dan akses kredenal Amazon Redshift Anda dengan aman dengan koneksi. AWS Glue Untuk informasi tentang cara membuat koneksi Anda sendiri, lihat Menghubungkan ke data.

Anda sekarang siap untuk menulis data Anda ke koneksi dengan bersepeda melalui `DynamicFrames` satu per satu:

```
for df_name in dfc.keys():
  m_df = dfc.select(df_name)
  print "Writing to table: ", df_name
  glueContext.write_dynamic_frame.from_jdbc_conf(frame = m_df, connection settings
 here)
```

Pengaturan koneksi Anda akan berbeda berdasarkan jenis database relasional Anda:

- Untuk instruksi tentang menulis ke Amazon Redshift, konsultasikan. the section called “Koneksi Redshift”

- Untuk database lain, konsultasikanJenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark.

Kesimpulan

Secara keseluruhan, AWS Glue sangat fleksibel. Ia memungkinkan Anda mencapai, dalam beberapa baris kode, apa yang biasanya akan memerlukan waktu berhari-hari untuk ditulis. Anda dapat menemukan seluruh skrip source-to-target ETL dalam file Python dalam sampel `join_and_relationalize.py` pada. AWS Glue GitHub

Contoh kode: Persiapan data menggunakan ResolveChoice, Lambda, dan ApplyMapping

Dataset yang digunakan dalam contoh ini terdiri dari data pembayaran Penyedia Medicare yang diunduh dari dua kumpulan data data.cms.gov: “Ringkasan Penyedia Sistem Pembayaran Prospektif Rawat Inap untuk 100 Grup Terkait Diagnosis Teratas - FY2011" dan “Data Biaya Rawat Inap TA 2011". Setelah mengunduh data, kami memodifikasi kumpulan data untuk memperkenalkan beberapa catatan yang salah di akhir file. File yang dimodifikasi ini terletak di bucket Amazon S3 publik di `s3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv`.

Anda dapat menemukan kode sumber untuk contoh ini dalam `data_cleaning_and_lambda.py` file di GitHub repositori AWS Gluecontoh.

Cara yang lebih disukai untuk men-debug Python PySpark atau skrip saat berjalan adalah dengan menggunakan Notebook AWS di Glue Studio. AWS

Langkah 1: Merayapi data di bucket Amazon S3

1. Masuk ke AWS Management Console dan buka AWS Glue konsol di https://console.aws.amazon.com/glue/.
2. Mengikuti proses yang dijelaskan dalamMengkonfigurasi crawler, buat crawler baru yang dapat merayapi `s3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv` file, dan dapat menempatkan metadata yang dihasilkan ke dalam database bernama dalam `payments` Katalog Data Glue AWS .
3. Jalankan crawler baru tersebut, dan kemudian periksa basis data `payments`. Anda akan menemukan bahwa crawler tersebut telah membuat tabel metadata bernama `medicare` dalam basis data setelah pembacaan awal file untuk menentukan format dan pembatas.

   Skema baru tabel `medicare` adalah sebagai berikut:

```
Column  name                                Data type
================================================
drg definition                                string
provider id                                   bigint
provider name                                 string
provider street address                       string
provider city                                 string
provider state                                string
provider zip code                             bigint
hospital referral region description          string
total discharges                              bigint
average covered charges                       string
average total payments                        string
average medicare payments                     string
```

Langkah 2: Tambahkan skrip boilerplate ke notebook endpoint pengembangan

Rekatkan skrip boilerplate berikut ke notebook endpoint pengembangan untuk mengimpor AWS Glue pustaka yang Anda butuhkan, dan siapkan satu: `GlueContext`

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

glueContext = GlueContext(SparkContext.getOrCreate())
```

Langkah 3: Bandingkan parsing skema yang berbeda

Selanjutnya, Anda dapat melihat apakah skema yang dikenali oleh Apache Spark sama dengan skema `DataFrame` yang direkam oleh crawler AndaAWS Glue. Jalankan kode ini:

```
medicare = spark.read.format(
   "com.databricks.spark.csv").option(
   "header", "true").option(
   "inferSchema", "true").load(
   's3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv')
```

```
medicare.printSchema()
```

Berikut adalah output dari panggilan `printSchema`:

```
root
 |-- DRG Definition: string (nullable = true)
 |-- Provider Id: string (nullable = true)
 |-- Provider Name: string (nullable = true)
 |-- Provider Street Address: string (nullable = true)
 |-- Provider City: string (nullable = true)
 |-- Provider State: string (nullable = true)
 |-- Provider Zip Code: integer (nullable = true)
 |-- Hospital Referral Region Description: string (nullable = true)
 |--  Total Discharges : integer (nullable = true)
 |--  Average Covered Charges : string (nullable = true)
 |--  Average Total Payments : string (nullable = true)
 |-- Average Medicare Payments: string (nullable = true)
```

Selanjutnya, lihat skema yang AWS Glue `DynamicFrame` dihasilkan oleh:

```
medicare_dynamicframe = glueContext.create_dynamic_frame.from_catalog(
       database = "payments",
       table_name = "medicare")
medicare_dynamicframe.printSchema()
```

Output dari `printSchema` adalah sebagai berikut:

```
root
 |-- drg definition: string
 |-- provider id: choice
 |    |-- long
 |    |-- string
 |-- provider name: string
 |-- provider street address: string
 |-- provider city: string
 |-- provider state: string
 |-- provider zip code: long
 |-- hospital referral region description: string
 |-- total discharges: long
```

```
 |-- average covered charges: string
 |-- average total payments: string
 |-- average medicare payments: string
```

`DynamicFrame` menghasilkan skema di mana `provider id` bisa berupa jenis `long` atau `string`. Skema `DataFrame` mencantumkan skema `Provider Id` sebagai jenis `string`, dan Katalog Data mencantumkan `provider id` sebagai jenis `bigint`.

Yang mana yang benar? Ada dua catatan di akhir file (dari 160.000 catatan) dengan nilai `string` dalam kolom tersebut. Ini adalah catatan-catatan yang keliru yang diperkenalkan untuk menggambarkan masalah.

Untuk mengatasi masalah semacam ini, AWS Glue `DynamicFrame` memperkenalkan konsep jenis pilihan. Dalam kasus ini, `DynamicFrame` menunjukkan bahwa nilai `long` dan `string` dapat muncul di kolom tersebut. AWS GlueCrawler melewatkan `string` nilai karena hanya dianggap awalan 2 MB data. `DataFrame` Apache Spark dianggap sebagai seluruh set data, tetapi ia dipaksa untuk menetapkan jenis yang paling umum ke kolom, yaitu `string`. Sebenarnya, Spark sering menggunakan kasus yang paling umum bila ada jenis atau variasi kompleks yang tidak biasa.

Untuk meng-kueri kolom `provider id`, selesaikan jenis pilihan terlebih dahulu. Anda dapat menggunakan metode transformasi `resolveChoice` di `DynamicFrame` untuk mengkonversi nilai `string` ke nilai `long` dengan pilihan `cast:long`:

```
medicare_res = medicare_dynamicframe.resolveChoice(specs = [('provider
 id','cast:long')])
medicare_res.printSchema()
```

Output `printSchema` sekarang:

```
root
 |-- drg definition: string
 |-- provider id: long
 |-- provider name: string
 |-- provider street address: string
 |-- provider city: string
 |-- provider state: string
 |-- provider zip code: long
 |-- hospital referral region description: string
 |-- total discharges: long
```

```
 |-- average covered charges: string
 |-- average total payments: string
 |-- average medicare payments: string
```

Dimana nilainya adalah a `string` yang tidak bisa dilemparkan, AWS Glue dimasukkan an`null`.

Pilihan lain adalah mengkonversi jenis pilihan ke `struct`, yang menyimpan nilai dari kedua jenis tersebut.

Selanjutnya, lihat baris yang mengalami anomali:

```
medicare_res.toDF().where("'provider id' is NULL").show()
```

Jika Anda melihat seperti berikut:

```
+--------------------+----------+--------------+----------------------+-------------
+-------------+-----------------+------------------------------------+---------------
+-----------------------+---------------------+------------------------+
|      drg definition|provider id|  provider name|provider street address|provider
 city|provider state|provider zip code|hospital referral region description|total
 discharges|average covered charges|average total payments|average medicare payments|
+--------------------+----------+--------------+----------------------+-------------
+-------------+-----------------+------------------------------------+---------------
+-----------------------+---------------------+------------------------+
|948 - SIGNS & SYM...|       null|            INC|       1050 DIVISION ST|
 MAUSTON|            WI|            53948|                        WI - Madison|
      12|             $11961.41|             $4619.00|                $3775.33|
|948 - SIGNS & SYM...|       null| INC- ST JOSEPH|     5000 W CHAMBERS ST|
 MILWAUKEE|            WI|            53210|                        WI - Milwaukee|
        14|             $10514.28|             $5562.50|                $4522.78|
+--------------------+----------+--------------+----------------------+-------------
+-------------+-----------------+------------------------------------+---------------
+-----------------------+---------------------+------------------------+
```

Sekarang hapus dua catatan yang cacat, sebagai berikut:

```
medicare_dataframe = medicare_res.toDF()
medicare_dataframe = medicare_dataframe.where("'provider id' is NOT NULL")
```

Langkah 4: Petakan data dan gunakan fungsi Apache Spark Lambda

AWS Gluebelum secara langsung mendukung fungsi Lambda, juga dikenal sebagai fungsi yang ditentukan pengguna. Tapi Anda selalu dapat mengkonversi `DynamicFrame` ke dan dari `DataFrame` Apache Spark untuk memanfaatkan dari fungsionalitas Spark selain fitur khusus `DynamicFrames`.

Selanjutnya, ubah informasi pembayaran menjadi angka, sehingga mesin analitik seperti Amazon Redshift atau Amazon Athena dapat melakukan penderakan angkanya lebih cepat:

```
from pyspark.sql.functions import udf
from pyspark.sql.types import StringType

chop_f = udf(lambda x: x[1:], StringType())
medicare_dataframe = medicare_dataframe.withColumn(
        "ACC", chop_f(
            medicare_dataframe["average covered charges"])).withColumn(
                "ATP", chop_f(
                    medicare_dataframe["average total payments"])).withColumn(
                        "AMP", chop_f(
                            medicare_dataframe["average medicare payments"]))
medicare_dataframe.select(['ACC', 'ATP', 'AMP']).show()
```

Output dari panggilan `show` adalah sebagai berikut:

```
+--------+-------+-------+
|     ACC|    ATP|    AMP|
+--------+-------+-------+
|32963.07|5777.24|4763.73|
|15131.85|5787.57|4976.71|
|37560.37|5434.95|4453.79|
|13998.28|5417.56|4129.16|
|31633.27|5658.33|4851.44|
|16920.79|6653.80|5374.14|
|11977.13|5834.74|4761.41|
|35841.09|8031.12|5858.50|
|28523.39|6113.38|5228.40|
|75233.38|5541.05|4386.94|
|67327.92|5461.57|4493.57|
|39607.28|5356.28|4408.20|
|22862.23|5374.65|4186.02|
|31110.85|5366.23|4376.23|
|25411.33|5282.93|4383.73|
```

```
| 9234.51|5676.55|4509.11|
|15895.85|5930.11|3972.85|
|19721.16|6192.54|5179.38|
|10710.88|4968.00|3898.88|
|51343.75|5996.00|4962.45|
+--------+-------+-------+
only showing top 20 rows
```

Ini semua masih string dalam data. Kita bisa menggunakan metode transformasi `apply_mapping` yang kuat untuk membuang, mengubah nama, mengubah, dan meng-nest data sehingga bahasa pemrograman data dan sistem lain dapat dengan mudah mengaksesnya:

```
from awsglue.dynamicframe import DynamicFrame
medicare_tmp_dyf = DynamicFrame.fromDF(medicare_dataframe, glueContext, "nested")
medicare_nest_dyf = medicare_tmp_dyf.apply_mapping([('drg definition', 'string', 'drg',
 'string'),
                ('provider id', 'long', 'provider.id', 'long'),
                ('provider name', 'string', 'provider.name', 'string'),
                ('provider city', 'string', 'provider.city', 'string'),
                ('provider state', 'string', 'provider.state', 'string'),
                ('provider zip code', 'long', 'provider.zip', 'long'),
                ('hospital referral region description', 'string','rr', 'string'),
                ('ACC', 'string', 'charges.covered', 'double'),
                ('ATP', 'string', 'charges.total_pay', 'double'),
                ('AMP', 'string', 'charges.medicare_pay', 'double')])
medicare_nest_dyf.printSchema()
```

Output `printSchema` adalah sebagai berikut:

```
root
 |-- drg: string
 |-- provider: struct
 |    |-- id: long
 |    |-- name: string
 |    |-- city: string
 |    |-- state: string
 |    |-- zip: long
 |-- rr: string
 |-- charges: struct
 |    |-- covered: double
 |    |-- total_pay: double
 |    |-- medicare_pay: double
```

Mengubah data kembali menjadi `DataFrame` Spark, Anda dapat menunjukkan apa yang terlihat seperti sekarang:

```
medicare_nest_dyf.toDF().show()
```

Output adalah sebagai berikut:

```
+--------------------+--------------------+--------------+--------------------+
|                 drg|            provider|            rr|             charges|
+--------------------+--------------------+--------------+--------------------+
|039 - EXTRACRANIA...|[10001,SOUTHEAST ...|    AL - Dothan|[32963.07,5777.24...|
|039 - EXTRACRANIA...|[10005,MARSHALL M...|AL - Birmingham|[15131.85,5787.57...|
|039 - EXTRACRANIA...|[10006,ELIZA COFF...|AL - Birmingham|[37560.37,5434.95...|
|039 - EXTRACRANIA...|[10011,ST VINCENT...|AL - Birmingham|[13998.28,5417.56...|
|039 - EXTRACRANIA...|[10016,SHELBY BAP...|AL - Birmingham|[31633.27,5658.33...|
|039 - EXTRACRANIA...|[10023,BAPTIST ME...|AL - Montgomery|[16920.79,6653.8,...|
|039 - EXTRACRANIA...|[10029,EAST ALABA...|AL - Birmingham|[11977.13,5834.74...|
|039 - EXTRACRANIA...|[10033,UNIVERSITY...|AL - Birmingham|[35841.09,8031.12...|
|039 - EXTRACRANIA...|[10039,HUNTSVILLE...|AL - Huntsville|[28523.39,6113.38...|
|039 - EXTRACRANIA...|[10040,GADSDEN RE...|AL - Birmingham|[75233.38,5541.05...|
|039 - EXTRACRANIA...|[10046,RIVERVIEW ...|AL - Birmingham|[67327.92,5461.57...|
|039 - EXTRACRANIA...|[10055,FLOWERS HO...|    AL - Dothan|[39607.28,5356.28...|
|039 - EXTRACRANIA...|[10056,ST VINCENT...|AL - Birmingham|[22862.23,5374.65...|
|039 - EXTRACRANIA...|[10078,NORTHEAST ...|AL - Birmingham|[31110.85,5366.23...|
|039 - EXTRACRANIA...|[10083,SOUTH BALD...|    AL - Mobile|[25411.33,5282.93...|
|039 - EXTRACRANIA...|[10085,DECATUR GE...|AL - Huntsville|[9234.51,5676.55,...|
|039 - EXTRACRANIA...|[10090,PROVIDENCE...|    AL - Mobile|[15895.85,5930.11...|
|039 - EXTRACRANIA...|[10092,D C H REGI...|AL - Tuscaloosa|[19721.16,6192.54...|
|039 - EXTRACRANIA...|[10100,THOMAS HOS...|    AL - Mobile|[10710.88,4968.0,...|
|039 - EXTRACRANIA...|[10103,BAPTIST ME...|AL - Birmingham|[51343.75,5996.0,...|
+--------------------+--------------------+--------------+--------------------+
only showing top 20 rows
```

Langkah 5: Tulis data ke Apache Parquet

AWS Gluemembuatnya mudah untuk menulis data dalam format seperti Apache Parquet yang database relasional dapat secara efektif mengkonsumsi:

```
glueContext.write_dynamic_frame.from_options(
       frame = medicare_nest_dyf,
       connection_type = "s3",
```

```
        connection_options = {"path": "s3://glue-sample-target/output-dir/
medicare_parquet"},
        format = "parquet")
```

AWS Glue PySpark referensi ekstensi

AWS Gluetelah membuat ekstensi berikut untuk dialek PySpark Python.

- Mengakses parameter menggunakan `getResolvedOptions`
- PySpark jenis ekstensi
- DynamicFrame kelas
- DynamicFrameCollection kelas
- DynamicFrameWriter kelas
- DynamicFrameReader kelas
- GlueContext kelas

Mengakses parameter menggunakan **`getResolvedOptions`**

Fungsi AWS Glue `getResolvedOptions(args, options)` utilitas memberi Anda akses ke argumen yang diteruskan ke skrip Anda saat Anda menjalankan pekerjaan. Untuk menggunakan fungsi ini, mulailah dengan mengimpornya dari AWS Glue `utils` modul, bersama dengan `sys` modul:

```
import sys
from awsglue.utils import getResolvedOptions
```

`getResolvedOptions(args, options)`

- `args` — Daftar argumen yang ada dalam `sys.argv`.
- `options` — Sebuah array Python nama argumen yang ingin Anda ambil.

Example Mengambil argumen yang diteruskan ke a JobRun

Misalkan Anda membuat JobRun dalam skrip, mungkin dalam fungsi Lambda:

```
response = client.start_job_run(
             JobName = 'my_test_Job',
```

```
            Arguments = {
              '--day_partition_key':   'partition_0',
              '--hour_partition_key':  'partition_1',
              '--day_partition_value':  day_partition_value,
              '--hour_partition_value': hour_partition_value } )
```

Untuk mengambil argumen yang diberikan, Anda dapat menggunakan fungsi `getResolvedOptions` sebagai berikut:

```
import sys
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv,
                          ['JOB_NAME',
                           'day_partition_key',
                           'hour_partition_key',
                           'day_partition_value',
                           'hour_partition_value'])
print "The day-partition key is: ", args['day_partition_key']
print "and the day-partition value is: ", args['day_partition_value']
```

Perhatikan bahwa masing-masing argumen didefinisikan sebagai awal dengan dua tanda hubung, kemudian direferensikan dalam skrip tersebut tanpa tanda hubung. Argumen hanya menggunakan garis bawah, bukan tanda hubung. Argumen Anda harus mengikuti konvensi ini agar bisa diubah.

PySpark jenis ekstensi

Jenis yang digunakan oleh AWS Glue PySpark ekstensi.

DataType

Kelas dasar untuk jenis Glue AWS yang lain.

`__init__(properties={})`

- `properties` — Properti dari tipe data (opsional).

`typeName(cls)`

Mengembalikan AWS Glue jenis kelas tipe (yaitu, nama kelas dengan “Type” dihapus dari akhir).

- `cls`— Sebuah contoh AWS Glue kelas yang berasal dari`DataType`.

`jsonValue( )`

Mengembalikan sebuah objek JSON yang berisi tipe data dan properti kelas:

```
{
  "dataType": typeName,
  "properties": properties
}
```

AtomicType dan derivatif sederhana

Mewarisi dari dan memperluas DataType kelas, dan berfungsi sebagai kelas dasar untuk semua tipe data AWS Glue atom.

`fromJsonValue(cls, json_value)`

Menginisialisasi sebuah instans kelas dengan nilai-nilai dari objek JSON.

- `cls`- Sebuah contoh kelas AWS Glue tipe untuk menginisialisasi.
- `json_value` — Objek JSON tempat untuk memuat pasangan nilai-kunci.

Jenis berikut adalah derivatif sederhana dari kelas AtomicType:

- `BinaryType` — Data biner.
- `BooleanType` – nilai boolean.
- `ByteType` — Nilai byte.
- `DateType` — Nilai datetime.
- `DoubleType` — Nilai ganda floating-point.
- `IntegerType` — Nilai integer.
- `LongType` — Nilai integer panjang.
- `NullType` — Nilai nol.
- `ShortType` — Nilai integer pendek.
- `StringType` — String teks.

- `TimestampType` — Nilai timestamp (biasanya dalam detik dari 1/1/1970).
- `UnknownType` — Nilai dengan tipe tak dikenal.

DecimalType(AtomicType)

Mewarisi dari dan meng-ekstensi kelas AtomicType untuk mewakili angka desimal (angka dinyatakan dalam angka desimal, sebagai lawan dari biner basis-2 angka).

`__init__(precision=10, scale=2, properties={})`

- `precision` — Jumlah digit dalam bilangan desimal (opsional; default-nya adalah 10).
- `scale` — Jumlah digit dalam sebelah kanan titik desimal (opsional; default-nya adalah 2).
- `properties` — Properti dari bilangan desimal (opsional).

EnumType(AtomicType)

Mewarisi dari dan meng-ekstensi kelas AtomicType untuk mewakili enumerasi pilihan yang valid.

`__init__(options)`

- `options` — Daftar opsi yang dienumerasi.

jenis koleksi

- ArrayType(DataType)
- ChoiceType(DataType)
- MapType(DataType)
- Field(Object)
- StructType(DataType)
- EntityType(DataType)

ArrayType(DataType)

`__init__(elementType=UnknownType(), properties={})`

- `elementType`— Jenis elemen dalam array (opsional; defaultnya adalah UnknownType).

- `properties` — Properti array (opsional).

ChoiceType(DataType)

`__init__(choices=[], properties={})`

- `choices` — Daftar pilihan yang mungkin (opsional).
- `properties` — Properti dari pilihan-pilihan tersebut (opsional).

`add(new_choice)`

Menambahkan sebuah pilihan baru ke daftar pilihan yang mungkin.

- `new_choice` — Pilihan yang akan ditambahkan ke daftar pilihan yang mungkin.

`merge(new_choices)`

Menggabungkan daftar pilihan baru dengan daftar pilihan yang ada.

- `new_choices` — Daftar pilihan baru yang akan digabungkan dengan pilihan yang ada.

MapType(DataType)

`__init__(valueType=UnknownType, properties={})`

- `valueType`— Jenis nilai di peta (opsional; defaultnya adalah UnknownType).
- `properties` — Properti dari peta (opsional).

Field(Object)

Menciptakan sebuah objek bidang dari sebuah objek yang berasal dari DataType.

`__init__(name, dataType, properties={})`

- `name` — Nama yang akan ditetapkan ke bidang.
- `dataType` — Objek untuk yang darinya bidang akan dibuat.

- `properties` — Properti dari bidang (opsional).

StructType(DataType)

Mendefinisikan sebuah struktur data (`struct`).

`__init__(fields=[], properties={})`

- `fields` — Daftar bidang (tipe `Field`) yang akan dimasukkan ke dalam struktur (opsional).
- `properties` — Properti struktur (opsional).

`add(field)`

- `field` — Sebuah objek dari tipe `Field` yang akan ditambahkan ke struktur.

`hasField(field)`

Mengembalikan `True` jika struktur ini memiliki bidang dengan nama yang sama, atau `False` jika tidak.

- `field` — Sebuah nama bidang, atau objek dari tipe `Field` yang namanya digunakan.

`getField(field)`

- `field` — Sebuah nama bidang atau objek dari tipe `Field` yang namanya digunakan. Jika struktur memiliki bidang dengan nama yang sama, maka ia dikembalikan.

EntityType(DataType)

`__init__(entity, base_type, properties)`

Kelas ini belum diimplementasikan.

jenis lainnya

- DataSource(objek)
- DataSink(objek)

DataSource(objek)

`__init__(j_source, sql_ctx, name)`

- `j_source` — Sumber data.
- `sql_ctx` — Konteks SQL.
- `name` — Nama sumber data.

`setFormat(format, **options)`

- `format` — Format yang ditetapkan untuk sumber data.
- `options` — Kumpulan pilihan yang akan ditetapkan untuk sumber data. Untuk informasi selengkapnya tentang opsi format, lihatthe section called “Opsi format data”.

`getFrame()`

Mengembalikan `DynamicFrame` untuk sumber data.

DataSink(objek)

`__init__(j_sink, sql_ctx)`

- `j_sink` — Sink yang akan dibuat.
- `sql_ctx` — Konteks SQL untuk data sink.

`setFormat(format, **options)`

- `format` — Format yang akan ditetapkan untuk data sink.
- `options` — Kumpulan pilihan yang akan ditetapkan untuk data sink. Untuk informasi selengkapnya tentang opsi format, lihatthe section called “Opsi format data”.

`setAccumulableSize(size)`

- `size` — Ukuran terakumulasi yang akan ditetapkan, dalam byte.

writeFrame(dynamic_frame, info="")

- dynamic_frame — DynamicFrame yang akan ditulis.
- info — Informasi tentang DynamicFrame (opsional).

write(dynamic_frame_or_dfc, info="")

Menulis DynamicFrame atau DynamicFrameCollection.

- dynamic_frame_or_dfc — Salah satu objek, objek DynamicFrame atau DynamicFrameCollection yang akan ditulis.
- info — Informasi tentang DynamicFrame atau DynamicFrames yang akan ditulis (opsional).

DynamicFrame kelas

Salah satu abstraksi utama dalam Apache Spark adalah DataFrame SparkSQL, yang serupa dengan konstruksi DataFrame yang ditemukan di R dan Panda. Sebuah DataFrame mirip dengan tabel dan mendukung operasi dengan gaya fungsional (memetakan/mengurangi/mem-filter/dll.) dan operasi SQL (memilih, memproyeksikan, mengagregat).

DataFrames adalah operasi yang kuat dan banyak digunakan, tetapi mereka memiliki keterbatasan sehubungan dengan operasi extract, transform, and load (ETL). Yang paling signifikan, mereka memerlukan skema yang akan ditentukan sebelum data dimuat. SparkSQL mengatasi ini dengan membuat dua pemberian melalui data—yang pertama untuk menyimpulkan skema, dan yang kedua untuk memuat data. Namun demikian, kesimpulan ini terbatas dan tidak mengatasi realitas data yang berantakan. Misalnya, bidang yang sama mungkin mempunyai jenis yang berbeda dalam catatan yang berbeda. Apache Spark sering kali menyerah dan melaporkan jenis sebagai string dengan menggunakan teks bidang asli. Ini mungkin tidak benar, dan Anda mungkin ingin kontrol yang lebih baik atas bagaimana perbedaan skema diselesaikan. Dan untuk set data besar, pemberian tambahan atas sumber data mungkin mahal.

Untuk mengatasi keterbatasan ini, AWS Glue memperkenalkan. DynamicFrame Sebuah DynamicFrame mirip dengan DataFrame, kecuali bahwa setiap catatan bersifat self-describing, sehingga tidak ada skema yang diperlukan diawal. Sebaliknya, AWS Glue menghitung skema on-the-fly bila diperlukan, dan secara eksplisit mengkodekan inkonsistensi skema menggunakan tipe pilihan (atau gabungan). Anda dapat mengatasi inkonsistensi ini untuk membuat set data Anda kompatibel dengan penyimpanan data yang memerlukan sebuah skema tetap.

Demikian pula, `DynamicRecord` mewakili sebuah catatan logis dalam sebuah `DynamicFrame`. Hal ini seperti sebuah baris dalam `DataFrame` Spark, kecuali bahwa ia bersifat self-describing dan dapat digunakan untuk data yang tidak sesuai dengan sebuah skema tetap. Saat menggunakan AWS Glue with PySpark, Anda biasanya tidak memanipulasi secara independen`DynamicRecords`. Sebaliknya, Anda akan mengubah kumpulan data bersama-sama melaluinya`DynamicFrame`.

Anda dapat mengonversi `DynamicFrames` ke dan dari `DataFrames` setelah Anda menyelesaikan inkonsistensi skema.

— konstruksi —

- __init__
- fromDF
- toDF

__init__

`__init__(jdf, glue_ctx, name)`

- `jdf` — Sebuah referensi ke bingkai data di Java Virtual Machine (JVM).
- `glue_ctx` — Sebuah objek GlueContext kelas.
- `name` — Sebuah nama string opsional, secara default kosong.

fromDF

`fromDF(dataframe, glue_ctx, name)`

Mengkonversi `DataFrame` ke `DynamicFrame` dengan mengkonversi bidang `DataFrame` ke bidang `DynamicRecord`. Mengembalikan `DynamicFrame` yang baru.

Sebuah `DynamicRecord` mewakili catatan logis dalam sebuah `DynamicFrame`. Hal ini seperti sebuah baris dalam `DataFrame` Spark, kecuali bahwa ia bersifat self-describing dan dapat digunakan untuk data yang tidak sesuai dengan sebuah skema tetap.

Fungsi ini mengharapkan kolom dengan nama duplikat di Anda `DataFrame` telah diselesaikan.

- `dataframe` — `DataFrame` Apache Spark SQL yang akan dikonversi (wajib).
- `glue_ctx` — Objek GlueContext kelas yang menentukan konteks untuk transformasi ini (wajib).
- `name`— Nama yang dihasilkan `DynamicFrame` (opsional sejak AWS Glue 3.0).

toDF

`toDF(options)`

Mengkonversi sebuah `DynamicFrame` ke sebuah `DataFrame` Apache Spark dengan mengkonversi bidang `DynamicRecords` ke bidang `DataFrame`. Mengembalikan `DataFrame` yang baru.

Sebuah `DynamicRecord` mewakili catatan logis dalam sebuah `DynamicFrame`. Hal ini seperti sebuah baris dalam `DataFrame` Spark, kecuali bahwa ia bersifat self-describing dan dapat digunakan untuk data yang tidak sesuai dengan sebuah skema tetap.

- `options` — Daftar pilihan. Menentukan jenis target jika Anda memilih jenis tindakan `Project` dan `Cast`. Contohnya meliputi hal berikut.

```
>>>toDF([ResolveOption("a.b.c", "KeepAsStruct")])
>>>toDF([ResolveOption("a.b.c", "Project", DoubleType())])
```

— Informasi —

- count
- schema
- printSchema
- show
- repartition
- coalesce

count

`count( )` — Mengembalikan jumlah baris dalam `DataFrame` yang mendasari.

schema

`schema( )` — Mengembalikan skema dari `DynamicFrame` ini, atau jika itu tidak tersedia, skema dari `DataFrame` yang mendasari.

Untuk informasi selengkapnya tentang `DynamicFrame` jenis yang membentuk skema ini, lihatthe section called “Tipe”.

printSchema

`printSchema( )` — Mencetak skema dari `DataFrame` yang mendasari.

show

`show(num_rows)` — Mencetak sejumlah baris yang ditentukan dari `DataFrame` yang mendasari.

repartition

`repartition(numPartitions)` — Mengembalikan `DynamicFrame` yang baru dengan partisi `numPartitions`.

coalesce

`coalesce(numPartitions)` — Mengembalikan `DynamicFrame` yang baru dengan partisi `numPartitions`.

— transformasi —

- apply_mapping
- drop_fields
- filter
- join
- map
- mergeDynamicFrame
- relationalize
- rename_field
- resolveChoice
- select_fields
- spigot
- split_fields
- split_rows
- unbox
- the section called “serikat”

- unnest
- unnest_ddb_json
- tulis

apply_mapping

`apply_mapping(mappings, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Menerapkan pemetaan deklaratif ke `DynamicFrame` dan mengembalikan yang baru `DynamicFrame` dengan pemetaan yang diterapkan ke bidang yang Anda tentukan. Bidang yang tidak ditentukan dihilangkan dari yang baru. `DynamicFrame`

- `mappings`— Daftar tupel pemetaan (wajib). Masing-masing terdiri dari: (kolom sumber, tipe sumber, kolom target, tipe target).

  Jika kolom sumber memiliki titik `"."` dalam nama, Anda harus menempatkan backticks `"``"` di sekitarnya. Misalnya, untuk memetakan `this.old.name` (string) `kethisNewName`, Anda akan menggunakan tupel berikut:

  ```
  ("`this.old.name`", "string", "thisNewName", "string")
  ```

- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).
- `stageThreshold`— Jumlah kesalahan yang ditemui selama transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `totalThreshold`— Jumlah kesalahan yang ditemui hingga dan termasuk transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.

Contoh: Gunakan apply_mapping untuk mengganti nama bidang dan mengubah jenis bidang

Contoh kode berikut menunjukkan cara menggunakan `apply_mapping` metode untuk mengganti nama bidang yang dipilih dan mengubah jenis bidang.

> **Note**
>
> Untuk mengakses kumpulan data yang digunakan dalam contoh ini, lihat Contoh kode: Bergabung dan menghubungkan data dan ikuti petunjuk diLangkah 1: Merayapi data di bucket Amazon S3.

```
# Example: Use apply_mapping to reshape source data into
# the desired column names and types as a new DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create a DynamicFrame and view its schema
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
print("Schema for the persons DynamicFrame:")
persons.printSchema()

# Select and rename fields, change field type
print("Schema for the persons_mapped DynamicFrame, created with apply_mapping:")
persons_mapped = persons.apply_mapping(
    [
        ("family_name", "String", "last_name", "String"),
        ("name", "String", "first_name", "String"),
        ("birth_date", "String", "date_of_birth", "Date"),
    ]
)
persons_mapped.printSchema()
```

Output

```
Schema for the persons DynamicFrame:
root
|-- family_name: string
|-- name: string
```

```
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string

Schema for the persons_mapped DynamicFrame, created with apply_mapping:
root
|-- last_name: string
|-- first_name: string
|-- date_of_birth: date
```

drop_fields

`drop_fields(paths, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Panggil transformasi FlatMap kelas untuk menghapus bidang dari `DynamicFrame`. Mengembalikan sebuah `DynamicFrame` baru dengan bidang tertentu yang dibuang.

- `paths`— Daftar string. Masing-masing berisi jalur lengkap ke simpul bidang yang ingin Anda jatuhkan. Anda dapat menggunakan notasi titik untuk menentukan bidang bersarang. Misalnya, jika bidang `first` adalah anak dari bidang `name` di pohon, Anda menentukan `"name.first"` untuk jalurnya.

  Jika node bidang memiliki literal `.` dalam nama, Anda harus melampirkan nama di backticks (). `
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).
- `stageThreshold`— Jumlah kesalahan yang ditemui selama transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `totalThreshold`— Jumlah kesalahan yang ditemui hingga dan termasuk transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.

Contoh: Gunakan drop_fields untuk menghapus bidang dari **`DynamicFrame`**

Contoh kode ini menggunakan `drop_fields` metode untuk menghapus bidang tingkat atas dan bersarang yang dipilih dari a. `DynamicFrame`

Contoh dataset

Contoh menggunakan dataset berikut yang diwakili oleh `EXAMPLE-FRIENDS-DATA` tabel dalam kode:

```
{"name": "Sally", "age": 23, "location": {"state": "WY", "county": "Fremont"},
 "friends": []}
{"name": "Varun", "age": 34, "location": {"state": "NE", "county": "Douglas"},
 "friends": [{"name": "Arjun", "age": 3}]}
{"name": "George", "age": 52, "location": {"state": "NY"}, "friends": [{"name":
 "Fred"}, {"name": "Amy", "age": 15}]}
{"name": "Haruki", "age": 21, "location": {"state": "AK", "county": "Denali"}}
{"name": "Sheila", "age": 63, "friends": [{"name": "Nancy", "age": 22}]}
```

Contoh kode

```
# Example: Use drop_fields to remove top-level and nested fields from a DynamicFrame.
```

```
# Replace MY-EXAMPLE-DATABASE with your Glue Data Catalog database name.
# Replace EXAMPLE-FRIENDS-DATA with your table name.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create a DynamicFrame from Glue Data Catalog
glue_source_database = "MY-EXAMPLE-DATABASE"
glue_source_table = "EXAMPLE-FRIENDS-DATA"

friends = glueContext.create_dynamic_frame.from_catalog(
    database=glue_source_database, table_name=glue_source_table
)
print("Schema for friends DynamicFrame before calling drop_fields:")
friends.printSchema()

# Remove location.county, remove friends.age, remove age
friends = friends.drop_fields(paths=["age", "location.county", "friends.age"])
print("Schema for friends DynamicFrame after removing age, county, and friend age:")
friends.printSchema()
```

Output

```
Schema for friends DynamicFrame before calling drop_fields:
root
|-- name: string
|-- age: int
|-- location: struct
|    |-- state: string
|    |-- county: string
|-- friends: array
|    |-- element: struct
|    |    |-- name: string
|    |    |-- age: int

Schema for friends DynamicFrame after removing age, county, and friend age:
root
|-- name: string
|-- location: struct
```

```
|    |-- state: string
|-- friends: array
|    |-- element: struct
|    |    |-- name: string
```

filter

`filter(f, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Mengembalikan baru `DynamicFrame` yang berisi semua `DynamicRecords` dalam input `DynamicFrame` yang memenuhi fungsi `f` predikat tertentu.

- `f` — Fungsi predikat yang akan diterapkan pada `DynamicFrame`. Fungsi tersebut harus mengambil `DynamicRecord` sebagai argumen dan mengembalikan BETUL jika `DynamicRecord` memenuhi persyaratan filter, atau SALAH jika tidak (wajib).

  Sebuah `DynamicRecord` mewakili catatan logis dalam sebuah `DynamicFrame`. Ini mirip dengan baris di Spark`DataFrame`, kecuali bahwa itu menggambarkan diri sendiri dan dapat digunakan untuk data yang tidak sesuai dengan skema tetap.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).
- `stageThreshold`— Jumlah kesalahan yang ditemui selama transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `totalThreshold`— Jumlah kesalahan yang ditemui hingga dan termasuk transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.

Contoh: Gunakan filter untuk mendapatkan pilihan bidang yang difilter

Contoh ini menggunakan `filter` metode untuk membuat baru `DynamicFrame` yang mencakup pemilihan bidang lain `DynamicFrame` yang difilter.

Seperti `map` metode, `filter` mengambil fungsi sebagai argumen yang akan diterapkan untuk setiap record dalam aslinya`DynamicFrame`. Fungsi ini mengambil catatan sebagai input dan

mengembalikan nilai Boolean. Jika nilai pengembalian benar, catatan akan dimasukkan dalam hasil`DynamicFrame`. Jika salah, catatan ditinggalkan.

> **Note**
>
> Untuk mengakses kumpulan data yang digunakan dalam contoh ini, lihat Contoh kode: Persiapan data menggunakan ResolveChoice, Lambda, dan ApplyMapping dan ikuti petunjuk diLangkah 1: Merayapi data di bucket Amazon S3.

```
# Example: Use filter to create a new DynamicFrame
# with a filtered selection of records

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create DynamicFrame from Glue Data Catalog
medicare = glueContext.create_dynamic_frame.from_options(
    "s3",
    {
        "paths": [
            "s3://awsglue-datasets/examples/medicare/Medicare_Hospital_Provider.csv"
        ]
    },
    "csv",
    {"withHeader": True},
)

# Create filtered DynamicFrame with custom lambda
# to filter records by Provider State and Provider City
sac_or_mon = medicare.filter(
    f=lambda x: x["Provider State"] in ["CA", "AL"]
    and x["Provider City"] in ["SACRAMENTO", "MONTGOMERY"]
)

# Compare record counts
print("Unfiltered record count: ", medicare.count())
```

```
print("Filtered record count:  ", sac_or_mon.count())
```

Output

```
Unfiltered record count:  163065
Filtered record count:   564
```

join

join(paths1, paths2, frame2, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)

Melakukan kesetaraan penggabungan dengan `DynamicFrame` yang lain dan mengembalikan `DynamicFrame` yang dihasilkan.

- `paths1` — Daftar kunci dalam bingkai ini yang akan digabungkan.
- `paths2` — Daftar kunci dalam bingkai lain yang akan digabungkan.
- `frame2` — `DynamicFrame` yang lainnya yang akan digabungkan.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).
- `stageThreshold`— Jumlah kesalahan yang ditemui selama transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `totalThreshold`— Jumlah kesalahan yang ditemui hingga dan termasuk transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.

Contoh: Gunakan bergabung untuk menggabungkan **`DynamicFrames`**

Contoh ini menggunakan `join` metode untuk melakukan join pada tiga`DynamicFrames`. AWS Glue melakukan gabungan berdasarkan tombol bidang yang Anda berikan. Hasilnya `DynamicFrame` berisi baris dari dua frame asli di mana kunci yang ditentukan cocok.

Perhatikan bahwa `join` transformasi membuat semua bidang tetap utuh. Ini berarti bahwa bidang yang Anda tentukan untuk dicocokkan muncul di hasil DynamicFrame, meskipun mereka berlebihan

dan berisi kunci yang sama. Dalam contoh ini, kita gunakan `drop_fields` untuk menghapus kunci redundan ini setelah bergabung.

> **Note**
>
> Untuk mengakses kumpulan data yang digunakan dalam contoh ini, lihat Contoh kode: Bergabung dan menghubungkan data dan ikuti petunjuk diLangkah 1: Merayapi data di bucket Amazon S3.

```
# Example: Use join to combine data from three DynamicFrames

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load DynamicFrames from Glue Data Catalog
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
memberships = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="memberships_json"
)
orgs = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="organizations_json"
)
print("Schema for the persons DynamicFrame:")
persons.printSchema()
print("Schema for the memberships DynamicFrame:")
memberships.printSchema()
print("Schema for the orgs DynamicFrame:")
orgs.printSchema()

# Join persons and memberships by ID
persons_memberships = persons.join(
    paths1=["id"], paths2=["person_id"], frame2=memberships
)

# Rename and drop fields from orgs
```

```
# to prevent field name collisions with persons_memberships
orgs = (
    orgs.drop_fields(["other_names", "identifiers"])
    .rename_field("id", "org_id")
    .rename_field("name", "org_name")
)

# Create final join of all three DynamicFrames
legislators_combined = orgs.join(
    paths1=["org_id"], paths2=["organization_id"], frame2=persons_memberships
).drop_fields(["person_id", "org_id"])

# Inspect the schema for the joined data
print("Schema for the new legislators_combined DynamicFrame:")
legislators_combined.printSchema()
```

Output

```
Schema for the persons DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
```

```
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string

Schema for the memberships DynamicFrame:
root
|-- area_id: string
|-- on_behalf_of_id: string
|-- organization_id: string
|-- role: string
|-- person_id: string
|-- legislative_period_id: string
|-- start_date: string
|-- end_date: string

Schema for the orgs DynamicFrame:
root
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- id: string
|-- classification: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- image: string
|-- seats: int
|-- type: string

Schema for the new legislators_combined DynamicFrame:
root
|-- role: string
|-- seats: int
```

```
|-- org_name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- type: string
|-- sort_name: string
|-- area_id: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- on_behalf_of_id: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- name: string
|-- birth_date: string
|-- organization_id: string
|-- gender: string
|-- classification: string
|-- legislative_period_id: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- image: string
|-- given_name: string
|-- start_date: string
|-- family_name: string
|-- id: string
|-- death_date: string
|-- end_date: string
```

map

`map(f, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Mengembalikan `DynamicFrame` baru dihasilkan dari penerapan fungsi pemetaan yang ditentukan untuk semua catatan dalam `DynamicFrame` aslinya.

- `f` — Fungsi pemetaan yang akan diterapkan ke semua catatan di `DynamicFrame`. Fungsi harus mengambil `DynamicRecord` sebagai sebuah argumen dan mengembalikan sebuah `DynamicRecord` baru (wajib).

  Sebuah `DynamicRecord` mewakili catatan logis dalam sebuah `DynamicFrame`. Ini mirip dengan baris di Apache Spark`DataFrame`, kecuali bahwa itu menggambarkan diri sendiri dan dapat digunakan untuk data yang tidak sesuai dengan skema tetap.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info`— String yang dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

Contoh: Gunakan peta untuk menerapkan fungsi ke setiap catatan dalam **`DynamicFrame`**

Contoh ini menunjukkan bagaimana menggunakan `map` metode untuk menerapkan fungsi untuk setiap record dari sebuah`DynamicFrame`. Secara khusus, contoh ini menerapkan fungsi yang dipanggil `MergeAddress` ke setiap catatan untuk menggabungkan beberapa bidang alamat menjadi satu `struct` jenis.

> **Note**
>
> Untuk mengakses kumpulan data yang digunakan dalam contoh ini, lihat Contoh kode: Persiapan data menggunakan ResolveChoice, Lambda, dan ApplyMapping dan ikuti petunjuk diLangkah 1: Merayapi data di bucket Amazon S3.

```
# Example: Use map to combine fields in all records
# of a DynamicFrame
```

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create a DynamicFrame and view its schema
medicare = glueContext.create_dynamic_frame.from_options(
        "s3",
        {"paths": ["s3://awsglue-datasets/examples/medicare/
Medicare_Hospital_Provider.csv"]},
        "csv",
        {"withHeader": True})
print("Schema for medicare DynamicFrame:")
medicare.printSchema()

# Define a function to supply to the map transform
# that merges address fields into a single field
def MergeAddress(rec):
  rec["Address"] = {}
  rec["Address"]["Street"] = rec["Provider Street Address"]
  rec["Address"]["City"] = rec["Provider City"]
  rec["Address"]["State"] = rec["Provider State"]
  rec["Address"]["Zip.Code"] = rec["Provider Zip Code"]
  rec["Address"]["Array"] = [rec["Provider Street Address"], rec["Provider City"],
 rec["Provider State"], rec["Provider Zip Code"]]
  del rec["Provider Street Address"]
  del rec["Provider City"]
  del rec["Provider State"]
  del rec["Provider Zip Code"]
  return rec


# Use map to apply MergeAddress to every record
mapped_medicare = medicare.map(f = MergeAddress)
print("Schema for mapped_medicare DynamicFrame:")
mapped_medicare.printSchema()
```

Output

```
Schema for medicare DynamicFrame:
```

```
root
|-- DRG Definition: string
|-- Provider Id: string
|-- Provider Name: string
|-- Provider Street Address: string
|-- Provider City: string
|-- Provider State: string
|-- Provider Zip Code: string
|-- Hospital Referral Region Description: string
|-- Total Discharges: string
|-- Average Covered Charges: string
|-- Average Total Payments: string
|-- Average Medicare Payments: string

Schema for mapped_medicare DynamicFrame:
root
|-- Average Total Payments: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address: struct
|    |-- Zip.Code: string
|    |-- City: string
|    |-- Array: array
|    |    |-- element: string
|    |-- State: string
|    |-- Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string
```

mergeDynamicFrame

`mergeDynamicFrame(stage_dynamic_frame, primary_keys, transformation_ctx = "", options = {}, info = "", stageThreshold = 0, totalThreshold = 0)`

Menggabungkan `DynamicFrame` ini dengan pentahapan `DynamicFrame` berdasarkan kunci primer yang ditentukan untuk mengidentifikasi catatan. Rekaman duplikat (catatan dengan kunci utama yang sama) tidak dideduplikasi. Jika tidak ada catatan yang cocok dalam bingkai pentahapan, semua catatan (termasuk duplikat) akan dipertahankan dari sumber. Jika bingkai pementasan memiliki catatan yang cocok, catatan dari bingkai pementasan menimpa catatan di sumber. AWS Glue

- `stage_dynamic_frame` — Pentahapan `DynamicFrame` yang akan digabungkan.
- `primary_keys` — Daftar bidang kunci primer untuk mencocokkan catatan dari sumber dan pentahapan bingkai dinamis.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengambil metadata tentang transformasi saat ini (opsional).
- `options` — Sebuah string pasangan nama-nilai JSON yang memberikan informasi tambahan untuk transformasi ini. Argumen ini saat ini tidak digunakan.
- `info` — Sebuah `String`. Setiap string yang akan dikaitkan dengan kesalahan dalam transformasi ini.
- `stageThreshold` — Sebuah `Long`. Jumlah kesalahan dalam transformasi yang ditentukan yang memerlukan pengolahan untuk membersihkan kesalahan.
- `totalThreshold` — Sebuah `Long`. Jumlah total kesalahan hingga dan termasuk transformasi ini yang perlu disalahartikan oleh pemrosesan.

Metode ini mengembalikan baru `DynamicFrame` yang diperoleh dengan menggabungkan ini `DynamicFrame` dengan `DynamicFrame` pementasan.

`DynamicFrame` yang dikembalikan berisi catatan A dalam kasus ini:

- Jika `A` ada di bingkai sumber dan bingkai pentahapan, maka `A` dalam bingkai pentahapan akan dikembalikan.
- Jika `A` ada di tabel sumber dan tidak `A.primaryKeys` ada di`stagingDynamicFrame`, tidak `A` diperbarui dalam tabel pementasan.

Bingkai sumber dan bingkai pementasan tidak perlu memiliki skema yang sama.

Contoh: Gunakan mergeDynamicFrame untuk menggabungkan dua **`DynamicFrames`** berdasarkan kunci utama

Contoh kode berikut menunjukkan bagaimana menggunakan `mergeDynamicFrame` metode untuk menggabungkan `DynamicFrame` dengan “pementasan”`DynamicFrame`, berdasarkan kunci utama. `id`

Contoh dataset

Contoh menggunakan dua `DynamicFrames` dari yang `DynamicFrameCollection` dipanggil`split_rows_collection`. Berikut ini adalah daftar kunci di`split_rows_collection`.

```
dict_keys(['high', 'low'])
```

Contoh kode

```
# Example: Use mergeDynamicFrame to merge DynamicFrames
# based on a set of specified primary keys

from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import SelectFromCollection

# Inspect the original DynamicFrames
frame_low = SelectFromCollection.apply(dfc=split_rows_collection, key="low")
print("Inspect the DynamicFrame that contains rows where ID < 10")
frame_low.toDF().show()

frame_high = SelectFromCollection.apply(dfc=split_rows_collection, key="high")
print("Inspect the DynamicFrame that contains rows where ID > 10")
frame_high.toDF().show()

# Merge the DynamicFrames based on the "id" primary key
merged_high_low = frame_high.mergeDynamicFrame(
    stage_dynamic_frame=frame_low, primary_keys=["id"]
)

# View the results where the ID is 1 or 20
print("Inspect the merged DynamicFrame that contains the combined rows")
merged_high_low.toDF().where("id = 1 or id= 20").orderBy("id").show()
```

Output

```
Inspect the DynamicFrame that contains rows where ID < 10
+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
|  1|    0|                    fax|           202-225-3307|
|  1|    1|                  phone|           202-225-5731|
|  2|    0|                    fax|           202-225-3307|
|  2|    1|                  phone|           202-225-5731|
|  3|    0|                    fax|           202-225-3307|
|  3|    1|                  phone|           202-225-5731|
|  4|    0|                    fax|           202-225-3307|
```

```
|  4|    1|                  phone|            202-225-5731|
|  5|    0|                    fax|            202-225-3307|
|  5|    1|                  phone|            202-225-5731|
|  6|    0|                    fax|            202-225-3307|
|  6|    1|                  phone|            202-225-5731|
|  7|    0|                    fax|            202-225-3307|
|  7|    1|                  phone|            202-225-5731|
|  8|    0|                    fax|            202-225-3307|
|  8|    1|                  phone|            202-225-5731|
|  9|    0|                    fax|            202-225-3307|
|  9|    1|                  phone|            202-225-5731|
| 10|    0|                    fax|            202-225-6328|
| 10|    1|                  phone|            202-225-4576|
+---+-----+----------------------+------------------------+
only showing top 20 rows

Inspect the DynamicFrame that contains rows where ID > 10
+---+-----+----------------------+------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+----------------------+------------------------+
| 11|    0|                    fax|            202-225-6328|
| 11|    1|                  phone|            202-225-4576|
| 11|    2|                twitter|          RepTrentFranks|
| 12|    0|                    fax|            202-225-6328|
| 12|    1|                  phone|            202-225-4576|
| 12|    2|                twitter|          RepTrentFranks|
| 13|    0|                    fax|            202-225-6328|
| 13|    1|                  phone|            202-225-4576|
| 13|    2|                twitter|          RepTrentFranks|
| 14|    0|                    fax|            202-225-6328|
| 14|    1|                  phone|            202-225-4576|
| 14|    2|                twitter|          RepTrentFranks|
| 15|    0|                    fax|            202-225-6328|
| 15|    1|                  phone|            202-225-4576|
| 15|    2|                twitter|          RepTrentFranks|
| 16|    0|                    fax|            202-225-6328|
| 16|    1|                  phone|            202-225-4576|
| 16|    2|                twitter|          RepTrentFranks|
| 17|    0|                    fax|            202-225-6328|
| 17|    1|                  phone|            202-225-4576|
+---+-----+----------------------+------------------------+
only showing top 20 rows

Inspect the merged DynamicFrame that contains the combined rows
```

```
+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
|  1|    0|                    fax|           202-225-3307|
|  1|    1|                  phone|           202-225-5731|
| 20|    0|                    fax|           202-225-5604|
| 20|    1|                  phone|           202-225-6536|
| 20|    2|                twitter|              USRepLong|
+---+-----+-----------------------+-----------------------+
```

relationalize

`relationalize(root_table_name, staging_path, options, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Mengkonversi `DynamicFrame` ke dalam bentuk yang cocok dalam database relasional. Relationalisasi a `DynamicFrame` sangat berguna ketika Anda ingin memindahkan data dari lingkungan NoSQL seperti DynamoDB ke database relasional seperti MySQL.

Transformasi menghasilkan daftar bingkai dengan membuka kolom bersarang dan kolom array berputar. Anda dapat menggabungkan kolom array berputar ke tabel root dengan menggunakan tombol gabungan yang dihasilkan selama fase unnest.

- `root_table_name` — Nama untuk tabel akar.
- `staging_path`— Jalur di mana metode dapat menyimpan partisi tabel berputar dalam format CSV (opsional). Tabel berputar dibaca kembali dari path ini.
- `options` — Kamus parameter opsional.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).
- `stageThreshold`— Jumlah kesalahan yang ditemui selama transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `totalThreshold`— Jumlah kesalahan yang ditemui hingga dan termasuk transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.

Contoh: Gunakan relationalize untuk meratakan skema bersarang di **`DynamicFrame`**

Contoh kode ini menggunakan `relationalize` metode untuk meratakan skema bersarang menjadi bentuk yang cocok dengan database relasional.

Contoh dataset

Contoh menggunakan `DynamicFrame` dipanggil `legislators_combined` dengan skema berikut. `legislators_combined`memiliki beberapa bidang bersarang seperti`links`,`images`, dan`contact_details`, yang akan diratakan oleh transformasi. `relationalize`

```
root
|-- role: string
|-- seats: int
|-- org_name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- type: string
|-- sort_name: string
|-- area_id: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- on_behalf_of_id: string
|-- other_names: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- name: string
|    |    |-- lang: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- name: string
|-- birth_date: string
|-- organization_id: string
|-- gender: string
|-- classification: string
|-- legislative_period_id: string
|-- identifiers: array
|    |-- element: struct
```

```
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- image: string
|-- given_name: string
|-- start_date: string
|-- family_name: string
|-- id: string
|-- death_date: string
|-- end_date: string
```

Contoh kode

```
# Example: Use relationalize to flatten
# a nested schema into a format that fits
# into a relational database.
# Replace DOC-EXAMPLE-S3-BUCKET/tmpDir with your own location.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Apply relationalize and inspect new tables
legislators_relationalized = legislators_combined.relationalize(
    "l_root", "s3://DOC-EXAMPLE-BUCKET/tmpDir"
)
legislators_relationalized.keys()

# Compare the schema of the contact_details
# nested field to the new relationalized table that
# represents it
legislators_combined.select_fields("contact_details").printSchema()
legislators_relationalized.select("l_root_contact_details").toDF().where(
    "id = 10 or id = 75"
).orderBy(["id", "index"]).show()
```

Output

Output berikut memungkinkan Anda membandingkan skema bidang bersarang yang dipanggil `contact_details` ke tabel yang dibuat oleh `relationalize` transformasi. Perhatikan bahwa

catatan tabel menautkan kembali ke tabel utama menggunakan kunci asing yang disebut `id` dan `index` kolom yang mewakili posisi array.

```
dict_keys(['l_root', 'l_root_images', 'l_root_links', 'l_root_other_names',
 'l_root_contact_details', 'l_root_identifiers'])

root
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string

+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
| 10|    0|                    fax|           202-225-4160|
| 10|    1|                  phone|           202-225-3436|
| 75|    0|                    fax|           202-225-6791|
| 75|    1|                  phone|           202-225-2861|
| 75|    2|                twitter|             RepSamFarr|
+---+-----+-----------------------+-----------------------+
```

rename_field

`rename_field(oldName, newName, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Mengganti nama bidang dalam `DynamicFrame` hal ini dan mengembalikan `DynamicFrame` baru dengan bidang yang diganti namanya.

- `oldName` — Jalur lengkap ke simpul yang ingin Anda ganti namanya.

  Jika nama lama memiliki titik di dalamnya, `RenameField` tidak berfungsi kecuali Anda menempatkan backticks di sekitarnya (`). Sebagai contoh, untuk menggantikan `this.old.name` dengan `thisNewName`, Anda akan memanggil rename_field sebagai berikut.

  ```
  newDyF = oldDyF.rename_field("`this.old.name`", "thisNewName")
  ```

- `newName` — Nama baru, sebagai path lengkap.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).

- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).
- `stageThreshold`— Jumlah kesalahan yang ditemui selama transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `totalThreshold`— Jumlah kesalahan yang ditemui hingga dan termasuk transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.

Contoh: Gunakan rename_field untuk mengganti nama bidang dalam **`DynamicFrame`**

Contoh kode ini menggunakan `rename_field` metode untuk mengganti nama bidang dalam`DynamicFrame`. Perhatikan bahwa contoh menggunakan metode chaining untuk mengganti nama beberapa bidang pada saat yang sama.

> **Note**
>
> Untuk mengakses kumpulan data yang digunakan dalam contoh ini, lihat Contoh kode: Bergabung dan menghubungkan data dan ikuti petunjuk diLangkah 1: Merayapi data di bucket Amazon S3.

Contoh kode

```
# Example: Use rename_field to rename fields
# in a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Inspect the original orgs schema
orgs = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="organizations_json"
)
print("Original orgs schema: ")
```

```
orgs.printSchema()

# Rename fields and view the new schema
orgs = orgs.rename_field("id", "org_id").rename_field("name", "org_name")
print("New orgs schema with renamed fields: ")
orgs.printSchema()
```

Output

```
Original orgs schema:
root
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- id: string
|-- classification: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- image: string
|-- seats: int
|-- type: string

New orgs schema with renamed fields:
root
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- classification: string
```

```
|-- org_id: string
|-- org_name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- image: string
|-- seats: int
|-- type: string
```

resolveChoice

`resolveChoice(specs = None, choice = "" , database = None , table_name = None , transformation_ctx="", info="", stageThreshold=0, totalThreshold=0, catalog_id = None)`

Mengubah jenis pilihan dalam `DynamicFrame` ini dan mengembalikan `DynamicFrame`.

- `specs` — Daftar ambiguitas spesifik yang akan diubah, masing-masing dalam bentuk tupel: `(field_path, action)`.

  Ada dua cara untuk menggunakan `resolveChoice`. Yang pertama adalah dengan menggunakan argumen `specs` untuk menentukan urutan bidang tertentu dan cara mengubahnya. Mode lainnya `resolveChoice` adalah menggunakan `choice` argumen untuk menentukan resolusi tunggal untuk semua`ChoiceTypes`.

  Nilai untuk `specs` ditetapkan sebagai tupel terdiri dari pasangan `(field_path, action)`. Nilai `field_path` mengidentifikasi elemen ambigu tertentu, dan nilai `action` mengidentifikasi resolusi yang sesuai. Berikut ini adalah tindakan yang mungkin:

  - `cast:`*`type`* — Upaya untuk mengubah semua nilai ke jenis tertentu. Misalnya: `cast:int`.
  - `make_cols` — Mengkonversi setiap jenis yang berbeda menjadi kolom dengan nama *`columnName`*`_`*`type`*. Ini menyelesaikan ambiguitas potensial dengan meratakan data. Misalnya, jika `columnA` bisa berupa `int` atau `string`, maka resolusi akan menghasilkan dua kolom bernama `columnA_int` dan `columnA_string` dalam `DynamicFrame` yang dihasilkan.
  - `make_struct` — Mengubah ambiguitas potensial dengan menggunakan `struct` untuk mewakili data. Misalnya, jika data dalam kolom bisa berupa `int` atau `astring`, `make_struct` tindakan menghasilkan kolom struktur yang dihasilkan`DynamicFrame`. Setiap struktur berisi a `int` dan `astring`.

- `project:type` — Mengubah ambiguitas potensial dengan memproyeksikan semua data ke salah satu jenis data yang mungkin. Sebagai contoh, jika data dalam kolom bisa berupa `int` atau `string`, dengan menggunakan tindakan `project:string` menghasilkan kolom dalam `DynamicFrame` yang dihasilkan di mana semua nilai `int` telah dikonversi menjadi string.

Jika `field_path` mengidentifikasi sebuah array, menempatkan kurung persegi kosong setelah nama array untuk menghindari ambiguitas. Misalnya, anggap Anda bekerja dengan data yang terstruktur sebagai berikut:

```
"myList": [
  { "price": 100.00 },
  { "price": "$100.00" }
]
```

Anda dapat memilih numerik daripada versi string harga dengan menyetel `field_path` ke`"myList[].price"`, dan menyetel `action` ke`"cast:double"`.

> **Note**
>
> Anda hanya dapat menggunakan salah satu `choice` parameter `specs` dan. Jika parameter `specs` bukan `None`, maka parameter `choice` harus string kosong. Sebaliknya, jika `choice` bukan string kosong, maka `specs` parameternya harus`None`.

- `choice` — Menentukan resolusi tunggal untuk semua `ChoiceTypes`. Anda dapat menggunakan ini dalam kasus di mana daftar lengkap tidak `ChoiceTypes` diketahui sebelum runtime. Selain tindakan-tindakan yang tercantum sebelumnya untuk `specs`, argumen ini juga mendukung tindakan berikut:
  - `match_catalog` — Upaya untuk mengubah setiap `ChoiceType` menjadi jenis yang sesuai dalam tabel Katalog Data yang ditentukan.
- `database` — Basis data Katalog Data yang akan digunakan dengan tindakan `match_catalog`.
- `table_name` — Tabel Katalog Data yang akan digunakan dengan tindakan `match_catalog`.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).

- `stageThreshold`— Jumlah kesalahan yang ditemui selama transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `totalThreshold`— Jumlah kesalahan yang ditemui hingga dan termasuk transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error out.
- `catalog_id` — ID katalog Katalog Data yang sedang diakses (ID akun Katalog Data). Ketika diatur ke `None` (nilai default), ia menggunakan ID katalog akun yang memanggil.

Contoh: Gunakan ResolveChoice untuk menangani kolom yang berisi beberapa jenis

Contoh kode ini menggunakan `resolveChoice` metode untuk menentukan cara menangani `DynamicFrame` kolom yang berisi nilai-nilai dari beberapa jenis. Contoh menunjukkan dua cara umum untuk menangani kolom dengan jenis yang berbeda:

- Transmisikan kolom ke satu tipe data.
- Pertahankan semua jenis di kolom terpisah.

Contoh dataset

> **Note**
>
> Untuk mengakses kumpulan data yang digunakan dalam contoh ini, lihat Contoh kode: Persiapan data menggunakan ResolveChoice, Lambda, dan ApplyMapping dan ikuti petunjuk diLangkah 1: Merayapi data di bucket Amazon S3.

Contoh menggunakan `DynamicFrame` dipanggil `medicare` dengan skema berikut:

```
root
|-- drg definition: string
|-- provider id: choice
|    |-- long
|    |-- string
|-- provider name: string
|-- provider street address: string
|-- provider city: string
|-- provider state: string
```

```
|-- provider zip code: long
|-- hospital referral region description: string
|-- total discharges: long
|-- average covered charges: string
|-- average total payments: string
|-- average medicare payments: string
```

Contoh kode

```
# Example: Use resolveChoice to handle
# a column that contains multiple types

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load the input data and inspect the "provider id" column
medicare = glueContext.create_dynamic_frame.from_catalog(
    database="payments", table_name="medicare_hospital_provider_csv"
)
print("Inspect the provider id column:")
medicare.toDF().select("provider id").show()

# Cast provider id to type long
medicare_resolved_long = medicare.resolveChoice(specs=[("provider id", "cast:long")])
print("Schema after casting provider id to type long:")
medicare_resolved_long.printSchema()
medicare_resolved_long.toDF().select("provider id").show()

# Create separate columns
# for each provider id type
medicare_resolved_cols = medicare.resolveChoice(choice="make_cols")
print("Schema after creating separate columns for each type:")
medicare_resolved_cols.printSchema()
medicare_resolved_cols.toDF().select("provider id_long", "provider id_string").show()
```

Output

```
Inspect the 'provider id' column:
+-----------+
```

```
|provider id|
+-----------+
|   [10001,]|
|   [10005,]|
|   [10006,]|
|   [10011,]|
|   [10016,]|
|   [10023,]|
|   [10029,]|
|   [10033,]|
|   [10039,]|
|   [10040,]|
|   [10046,]|
|   [10055,]|
|   [10056,]|
|   [10078,]|
|   [10083,]|
|   [10085,]|
|   [10090,]|
|   [10092,]|
|   [10100,]|
|   [10103,]|
+-----------+
only showing top 20 rows

Schema after casting 'provider id' to type long:
root
|-- drg definition: string
|-- provider id: long
|-- provider name: string
|-- provider street address: string
|-- provider city: string
|-- provider state: string
|-- provider zip code: long
|-- hospital referral region description: string
|-- total discharges: long
|-- average covered charges: string
|-- average total payments: string
|-- average medicare payments: string

+-----------+
|provider id|
+-----------+
|      10001|
```

```
|      10005|
|      10006|
|      10011|
|      10016|
|      10023|
|      10029|
|      10033|
|      10039|
|      10040|
|      10046|
|      10055|
|      10056|
|      10078|
|      10083|
|      10085|
|      10090|
|      10092|
|      10100|
|      10103|
+-----------+
only showing top 20 rows

Schema after creating separate columns for each type:
root
|-- drg definition: string
|-- provider id_string: string
|-- provider id_long: long
|-- provider name: string
|-- provider street address: string
|-- provider city: string
|-- provider state: string
|-- provider zip code: long
|-- hospital referral region description: string
|-- total discharges: long
|-- average covered charges: string
|-- average total payments: string
|-- average medicare payments: string

+---------------+------------------+
|provider id_long|provider id_string|
+---------------+------------------+
|           10001|              null|
|           10005|              null|
|           10006|              null|
```

```
|           10011|              null|
|           10016|              null|
|           10023|              null|
|           10029|              null|
|           10033|              null|
|           10039|              null|
|           10040|              null|
|           10046|              null|
|           10055|              null|
|           10056|              null|
|           10078|              null|
|           10083|              null|
|           10085|              null|
|           10090|              null|
|           10092|              null|
|           10100|              null|
|           10103|              null|
+----------------+------------------+
only showing top 20 rows
```

select_fields

`select_fields(paths, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Mengembalikan baru `DynamicFrame` yang berisi bidang yang dipilih.

- `paths`— Daftar string. Setiap string adalah jalur ke node tingkat atas yang ingin Anda pilih.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).
- `stageThreshold`— Jumlah kesalahan yang ditemui selama transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `totalThreshold`— Jumlah kesalahan yang ditemui hingga dan termasuk transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.

Contoh: Gunakan select_fields untuk membuat yang baru **`DynamicFrame`** dengan bidang yang dipilih

Contoh kode berikut menunjukkan cara menggunakan `select_fields` metode untuk membuat yang baru `DynamicFrame` dengan daftar bidang yang dipilih dari yang sudah ada`DynamicFrame`.

> **Note**
>
> Untuk mengakses kumpulan data yang digunakan dalam contoh ini, lihat Contoh kode: Bergabung dan menghubungkan data dan ikuti petunjuk diLangkah 1: Merayapi data di bucket Amazon S3.

```
# Example: Use select_fields to select specific fields from a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create a DynamicFrame and view its schema
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
print("Schema for the persons DynamicFrame:")
persons.printSchema()

# Create a new DynamicFrame with chosen fields
names = persons.select_fields(paths=["family_name", "given_name"])
print("Schema for the names DynamicFrame, created with select_fields:")
names.printSchema()
names.toDF().show()
```

Output

```
Schema for the persons DynamicFrame:
root
|-- family_name: string
|-- name: string
```

```
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string

Schema for the names DynamicFrame:
root
|-- family_name: string
|-- given_name: string

+-----------+----------+
|family_name|given_name|
+-----------+----------+
|    Collins|   Michael|
|   Huizenga|      Bill|
|    Clawson|    Curtis|
|    Solomon|    Gerald|
|     Rigell|    Edward|
|      Crapo|   Michael|
|      Hutto|      Earl|
|      Ertel|     Allen|
```

```
|     Minish|    Joseph|
|    Andrews|    Robert|
|     Walden|      Greg|
|      Kazen|   Abraham|
|     Turner|   Michael|
|      Kolbe|     James|
|  Lowenthal|      Alan|
|    Capuano|   Michael|
|   Schrader|      Kurt|
|     Nadler|   Jerrold|
|     Graves|       Tom|
|   McMillan|      John|
+-----------+----------+
only showing top 20 rows
```

sederhana_ddb_json

`simplify_ddb_json(): DynamicFrame`

Menyederhanakan kolom bersarang dalam `DynamicFrame` yang secara khusus dalam struktur DynamoDB JSON, dan mengembalikan disederhanakan baru. `DynamicFrame` Jika ada beberapa jenis atau tipe Peta dalam tipe Daftar, elemen dalam Daftar tidak akan disederhanakan. Perhatikan bahwa ini adalah jenis transformasi tertentu yang berperilaku berbeda dari `unnest` transformasi biasa dan mengharuskan data sudah berada dalam struktur DynamoDB JSON. Untuk informasi selengkapnya, lihat DynamoDB JSON.

Misalnya, skema pembacaan ekspor dengan struktur DynamoDB JSON mungkin terlihat seperti berikut:

```
root
|-- Item: struct
|    |-- parentMap: struct
|    |    |-- M: struct
|    |    |    |-- childMap: struct
|    |    |    |    |-- M: struct
|    |    |    |    |    |-- appName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- packageName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- updatedAt: struct
|    |    |    |    |    |    |-- N: string
|    |-- strings: struct
```

```
|    |    |-- SS: array
|    |    |    |-- element: string
|    |-- numbers: struct
|    |    |-- NS: array
|    |    |    |-- element: string
|    |-- binaries: struct
|    |    |-- BS: array
|    |    |    |-- element: string
|    |-- isDDBJson: struct
|    |    |-- BOOL: boolean
|    |-- nullValue: struct
|    |    |-- NULL: boolean
```

`simplify_ddb_json()`Transformasi akan mengubah ini menjadi:

```
root
|-- parentMap: struct
|    |-- childMap: struct
|    |    |-- appName: string
|    |    |-- packageName: string
|    |    |-- updatedAt: string
|-- strings: array
|    |-- element: string
|-- numbers: array
|    |-- element: string
|-- binaries: array
|    |-- element: string
|-- isDDBJson: boolean
|-- nullValue: null
```

Contoh: Gunakan simplify_ddb_json untuk memanggil DynamoDB JSON menyederhanakan

Contoh kode ini menggunakan `simplify_ddb_json` metode untuk menggunakan konektor ekspor AWS Glue DynamoDB, memanggil DynamoDB JSON simplify, dan mencetak jumlah partisi.

Contoh kode

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext()
glueContext = GlueContext(sc)
```

```
dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type = "dynamodb",
    connection_options = {
        'dynamodb.export': 'ddb',
        'dynamodb.tableArn': '<table arn>',
        'dynamodb.s3.bucket': '<bucket name>',
        'dynamodb.s3.prefix': '<bucket prefix>',
        'dynamodb.s3.bucketOwner': '<account_id of bucket>'
    }
)
simplified = dynamicFrame.simplify_ddb_json()
print(simplified.getNumPartitions())
```

spigot

`spigot(path, options={})`

Menulis catatan sampel ke tujuan tertentu untuk membantu Anda memverifikasi transformasi yang dilakukan oleh pekerjaan Anda.

- `path`— Jalur tujuan untuk menulis (wajib).
- `options`— Pasangan kunci-nilai yang menentukan opsi (opsional). Pilihan `"topk"` menentukan bahwa catatan k pertama harus ditulis. `"prob"`Opsi menentukan probabilitas (sebagai desimal) untuk memilih catatan yang diberikan. Anda dapat menggunakannya dalam memilih catatan untuk menulis.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).

Contoh: Gunakan keran untuk menulis bidang contoh dari a **`DynamicFrame`** ke Amazon S3

Contoh kode ini menggunakan `spigot` metode untuk menulis catatan sampel ke bucket Amazon S3 setelah menerapkan transformasi. `select_fields`

Contoh dataset

> **Note**
>
> Untuk mengakses kumpulan data yang digunakan dalam contoh ini, lihat Contoh kode: Bergabung dan menghubungkan data dan ikuti petunjuk diLangkah 1: Merayapi data di bucket Amazon S3.

Contoh menggunakan `DynamicFrame` dipanggil `persons` dengan skema berikut:

```
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

Contoh kode

```
# Example: Use spigot to write sample records
# to a destination during a transformation
# from pyspark.context import SparkContext.
# Replace DOC-EXAMPLE-BUCKET with your own location.

from pyspark.context import SparkContext
from awsglue.context import GlueContext
```

```
# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load table data into a DynamicFrame
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)

# Perform the select_fields on the DynamicFrame
persons = persons.select_fields(paths=["family_name", "given_name", "birth_date"])

# Use spigot to write a sample of the transformed data
# (the first 10 records)
spigot_output = persons.spigot(
    path="s3://DOC-EXAMPLE-BUCKET", options={"topk": 10}
)
```

Output

Berikut ini adalah contoh data yang `spigot` menulis ke Amazon S3. Karena kode contoh yang ditentukan`options={"topk": 10}`, data sampel berisi 10 catatan pertama.

```
{"family_name":"Collins","given_name":"Michael","birth_date":"1944-10-15"}
{"family_name":"Huizenga","given_name":"Bill","birth_date":"1969-01-31"}
{"family_name":"Clawson","given_name":"Curtis","birth_date":"1959-09-28"}
{"family_name":"Solomon","given_name":"Gerald","birth_date":"1930-08-14"}
{"family_name":"Rigell","given_name":"Edward","birth_date":"1960-05-28"}
{"family_name":"Crapo","given_name":"Michael","birth_date":"1951-05-20"}
{"family_name":"Hutto","given_name":"Earl","birth_date":"1926-05-12"}
{"family_name":"Ertel","given_name":"Allen","birth_date":"1937-11-07"}
{"family_name":"Minish","given_name":"Joseph","birth_date":"1916-09-01"}
{"family_name":"Andrews","given_name":"Robert","birth_date":"1957-08-04"}
```

split_fields

`split_fields(paths, name1, name2, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Mengembalikan baru `DynamicFrameCollection` yang berisi dua`DynamicFrames`. Yang pertama `DynamicFrame` berisi semua node yang telah dipisahkan, dan yang kedua berisi node yang tersisa.

- `paths` — Daftar string, masing-masing merupakan path lengkap ke sebuah simpul yang ingin Anda bagi menjadi sebuah `DynamicFrame`.
- `name1` — Sebuah string nama untuk `DynamicFrame` yang terbelah.
- `name2` — Sebuah string nama untuk `DynamicFrame` yang tersisa setelah simpul yang ditentukan telah dipecah.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).
- `stageThreshold`— Jumlah kesalahan yang ditemui selama transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `totalThreshold`— Jumlah kesalahan yang ditemui hingga dan termasuk transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.

Contoh: Gunakan split_fields untuk membagi bidang yang dipilih menjadi terpisah **`DynamicFrame`**

Contoh kode ini menggunakan `split_fields` metode untuk membagi daftar bidang tertentu menjadi terpisah`DynamicFrame`.

Contoh dataset

Contoh menggunakan panggilan `DynamicFrame` `l_root_contact_details` yang berasal dari koleksi bernama`legislators_relationalized`.

`l_root_contact_details`memiliki skema dan entri berikut.

```
root
|-- id: long
|-- index: int
|-- contact_details.val.type: string
|-- contact_details.val.value: string

+---+-----+------------------------+-------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+------------------------+-------------------------+
```

```
|  1|    0|                  phone|             202-225-5265|
|  1|    1|                twitter|              kathyhochul|
|  2|    0|                  phone|             202-225-3252|
|  2|    1|                twitter|            repjackyrosen|
|  3|    0|                    fax|             202-225-1314|
|  3|    1|                  phone|             202-225-3772|
...
```

Contoh kode

```
# Example: Use split_fields to split selected
# fields into a separate DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Load the input DynamicFrame and inspect its schema
frame_to_split = legislators_relationalized.select("l_root_contact_details")
print("Inspect the input DynamicFrame schema:")
frame_to_split.printSchema()

# Split id and index fields into a separate DynamicFrame
split_fields_collection = frame_to_split.split_fields(["id", "index"], "left", "right")

# Inspect the resulting DynamicFrames
print("Inspect the schemas of the DynamicFrames created with split_fields:")
split_fields_collection.select("left").printSchema()
split_fields_collection.select("right").printSchema()
```

Output

```
Inspect the input DynamicFrame's schema:
root
|-- id: long
|-- index: int
|-- contact_details.val.type: string
|-- contact_details.val.value: string

Inspect the schemas of the DynamicFrames created with split_fields:
```

```
root
|-- id: long
|-- index: int

root
|-- contact_details.val.type: string
|-- contact_details.val.value: string
```

split_rows

`split_rows(comparison_dict, name1, name2, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

Membagi satu atau beberapa baris dalam `DynamicFrame` ke sebuah `DynamicFrame` yang baru.

Metode mengembalikan baru `DynamicFrameCollection` yang berisi dua`DynamicFrames`. Yang pertama `DynamicFrame` berisi semua baris yang telah dipisahkan, dan yang kedua berisi baris yang tersisa.

- `comparison_dict`— Kamus di mana kuncinya adalah jalur ke kolom, dan nilainya adalah kamus lain untuk memetakan komparator ke nilai yang dibandingkan dengan nilai kolom. Misalnya, `{"age": {">": 10, "<": 20}}` membagi semua baris yang nilainya di kolom usia lebih besar dari 10 dan kurang dari 20.
- `name1` — Sebuah string nama untuk `DynamicFrame` yang terbelah.
- `name2` — Sebuah string nama untuk `DynamicFrame` yang tersisa setelah simpul yang ditentukan telah dipecah.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).
- `stageThreshold`— Jumlah kesalahan yang ditemui selama transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `totalThreshold`— Jumlah kesalahan yang ditemui hingga dan termasuk transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.

Contoh: Gunakan split_rows untuk membagi baris dalam **`DynamicFrame`**

Contoh kode ini menggunakan `split_rows` metode untuk membagi baris `DynamicFrame` berdasarkan nilai `id` bidang.

Contoh dataset

Contoh menggunakan panggilan `DynamicFrame` `l_root_contact_details` yang dipilih dari koleksi bernama`legislators_relationalized`.

`l_root_contact_details`memiliki skema dan entri berikut.

```
root
|-- id: long
|-- index: int
|-- contact_details.val.type: string
|-- contact_details.val.value: string

+---+-----+------------------------+------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+------------------------+------------------------+
|  1|    0|                   phone|             202-225-5265|
|  1|    1|                 twitter|              kathyhochul|
|  2|    0|                   phone|             202-225-3252|
|  2|    1|                 twitter|            repjackyrosen|
|  3|    0|                     fax|             202-225-1314|
|  3|    1|                   phone|             202-225-3772|
|  3|    2|                 twitter|          MikeRossUpdates|
|  4|    0|                     fax|             202-225-1314|
|  4|    1|                   phone|             202-225-3772|
|  4|    2|                 twitter|          MikeRossUpdates|
|  5|    0|                     fax|             202-225-1314|
|  5|    1|                   phone|             202-225-3772|
|  5|    2|                 twitter|          MikeRossUpdates|
|  6|    0|                     fax|             202-225-1314|
|  6|    1|                   phone|             202-225-3772|
|  6|    2|                 twitter|          MikeRossUpdates|
|  7|    0|                     fax|             202-225-1314|
|  7|    1|                   phone|             202-225-3772|
|  7|    2|                 twitter|          MikeRossUpdates|
|  8|    0|                     fax|             202-225-1314|
+---+-----+------------------------+------------------------+
```

Contoh kode

```
# Example: Use split_rows to split up
# rows in a DynamicFrame based on value

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Retrieve the DynamicFrame to split
frame_to_split = legislators_relationalized.select("l_root_contact_details")

# Split up rows by ID
split_rows_collection = frame_to_split.split_rows({"id": {">": 10}}, "high", "low")

# Inspect the resulting DynamicFrames
print("Inspect the DynamicFrame that contains IDs < 10")
split_rows_collection.select("low").toDF().show()
print("Inspect the DynamicFrame that contains IDs > 10")
split_rows_collection.select("high").toDF().show()
```

Output

```
Inspect the DynamicFrame that contains IDs < 10
+---+-----+-----------------------+-------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-------------------------+
|  1|    0|                  phone|             202-225-5265|
|  1|    1|                twitter|              kathyhochul|
|  2|    0|                  phone|             202-225-3252|
|  2|    1|                twitter|            repjackyrosen|
|  3|    0|                    fax|             202-225-1314|
|  3|    1|                  phone|             202-225-3772|
|  3|    2|                twitter|          MikeRossUpdates|
|  4|    0|                    fax|             202-225-1314|
|  4|    1|                  phone|             202-225-3772|
|  4|    2|                twitter|          MikeRossUpdates|
|  5|    0|                    fax|             202-225-1314|
|  5|    1|                  phone|             202-225-3772|
|  5|    2|                twitter|          MikeRossUpdates|
```

```
|  6|    0|                   fax|             202-225-1314|
|  6|    1|                 phone|             202-225-3772|
|  6|    2|               twitter|          MikeRossUpdates|
|  7|    0|                   fax|             202-225-1314|
|  7|    1|                 phone|             202-225-3772|
|  7|    2|               twitter|          MikeRossUpdates|
|  8|    0|                   fax|             202-225-1314|
+---+-----+----------------------+-------------------------+
only showing top 20 rows

Inspect the DynamicFrame that contains IDs > 10
+---+-----+----------------------+-------------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+----------------------+-------------------------+
| 11|    0|                 phone|             202-225-5476|
| 11|    1|               twitter|            RepDavidYoung|
| 12|    0|                 phone|             202-225-4035|
| 12|    1|               twitter|           RepStephMurphy|
| 13|    0|                   fax|             202-226-0774|
| 13|    1|                 phone|             202-225-6335|
| 14|    0|                   fax|             202-226-0774|
| 14|    1|                 phone|             202-225-6335|
| 15|    0|                   fax|             202-226-0774|
| 15|    1|                 phone|             202-225-6335|
| 16|    0|                   fax|             202-226-0774|
| 16|    1|                 phone|             202-225-6335|
| 17|    0|                   fax|             202-226-0774|
| 17|    1|                 phone|             202-225-6335|
| 18|    0|                   fax|             202-226-0774|
| 18|    1|                 phone|             202-225-6335|
| 19|    0|                   fax|             202-226-0774|
| 19|    1|                 phone|             202-225-6335|
| 20|    0|                   fax|             202-226-0774|
| 20|    1|                 phone|             202-225-6335|
+---+-----+----------------------+-------------------------+
only showing top 20 rows
```

unbox

`unbox(path, format, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0, **options)`

Membuka kotak (memformat ulang) bidang string di a `DynamicFrame` dan mengembalikan yang baru yang berisi kotak yang `DynamicFrame` tidak dikotaknya. `DynamicRecords`

Sebuah `DynamicRecord` mewakili catatan logis dalam sebuah `DynamicFrame`. Ini mirip dengan baris di Apache Spark`DataFrame`, kecuali bahwa itu menggambarkan diri sendiri dan dapat digunakan untuk data yang tidak sesuai dengan skema tetap.

- `path` — Sebuah path lengkap ke simpul string ingin Anda buka kotaknya.
- `format` — Format spesifikasi (opsional). Anda menggunakan ini untuk Amazon S3 atau AWS Glue koneksi yang mendukung berbagai format. Untuk format yang didukung, lihatOpsi format data untuk input dan output untuk Spark AWS Glue.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).
- `stageThreshold`— Jumlah kesalahan yang ditemui selama transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `totalThreshold`— Jumlah kesalahan yang ditemui hingga dan termasuk transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `options` — Satu atau beberapa hal berikut:
  - `separator`— String yang berisi karakter pemisah.
  - `escaper`— Sebuah string yang berisi karakter escape.
  - `skipFirst`— Nilai Boolean yang menunjukkan apakah akan melewatkan contoh pertama.
  - `withSchema`— Sebuah string yang berisi representasi JSON dari skema node. Format representasi JSON skema didefinisikan oleh output dari. `StructType.json()`
  - `withHeader`— Nilai Boolean yang menunjukkan apakah header disertakan.

Contoh: Gunakan unbox untuk membuka kotak bidang string ke dalam struct

Contoh kode ini menggunakan `unbox` metode untuk membuka kotak, atau memformat ulang, bidang string di `DynamicFrame` dalam bidang tipe struct.

Contoh dataset

Contoh menggunakan `DynamicFrame` dipanggil `mapped_with_string` dengan skema dan entri berikut.

Perhatikan bidang bernama`AddressString`. Ini adalah bidang yang membuka kotak contoh menjadi struct.

```
root
|-- Average Total Payments: string
|-- AddressString: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address: struct
|    |-- Zip.Code: string
|    |-- City: string
|    |-- Array: array
|    |    |-- element: string
|    |-- State: string
|    |-- Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string

+----------------------+--------------------+-----------------------
+--------------------+------------------------+-----------------------------------
+--------------------+-----------+----------------+--------------------+
|Average Total Payments|       AddressString|Average Covered Charges|      DRG
 Definition|Average Medicare Payments|Hospital Referral Region Description|
 Address|Provider Id|Total Discharges|       Provider Name|
+----------------------+--------------------+-----------------------
+--------------------+------------------------+-----------------------------------
+--------------------+-----------+----------------+--------------------+
|              $5777.24|{"Street": "1108 ...|              $32963.07|039 -
 EXTRACRANIA...|                $4763.73|                         AL - Dothan|[36301,
 DOTHAN, [...|      10001|              91|SOUTHEAST ALABAMA...|
```

```
|             $5787.57|{"Street": "2505 ...|             $15131.85|039 -
 EXTRACRANIA...|                $4976.71|                   AL - Birmingham|[35957,
 BOAZ, [25...|      10005|             14|MARSHALL MEDICAL ...|
|             $5434.95|{"Street": "205 M...|             $37560.37|039 -
 EXTRACRANIA...|                $4453.79|                   AL - Birmingham|[35631,
 FLORENCE,...|      10006|             24|ELIZA COFFEE MEMO...|
|             $5417.56|{"Street": "50 ME...|             $13998.28|039 -
 EXTRACRANIA...|                $4129.16|                   AL - Birmingham|[35235,
 BIRMINGHA...|      10011|             25|   ST VINCENT'S EAST|
...
```

Contoh kode

```
# Example: Use unbox to unbox a string field
# into a struct in a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

unboxed = mapped_with_string.unbox("AddressString", "json")
unboxed.printSchema()
unboxed.toDF().show()
```

Output

```
root
|-- Average Total Payments: string
|-- AddressString: struct
|    |-- Street: string
|    |-- City: string
|    |-- State: string
|    |-- Zip.Code: string
|    |-- Array: array
|    |    |-- element: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
```

```
|-- Address: struct
|    |-- Zip.Code: string
|    |-- City: string
|    |-- Array: array
|    |    |-- element: string
|    |-- State: string
|    |-- Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string

+----------------------+--------------------+-----------------------
+--------------------+-------------------------+-----------------------------------
+--------------------+-----------+----------------+--------------------+
|Average Total Payments|       AddressString|Average Covered Charges|      DRG
 Definition|Average Medicare Payments|Hospital Referral Region Description|
 Address|Provider Id|Total Discharges|       Provider Name|
+----------------------+--------------------+-----------------------
+--------------------+-------------------------+-----------------------------------
+--------------------+-----------+----------------+--------------------+
|              $5777.24|[1108 ROSS CLARK ...|              $32963.07|039 -
 EXTRACRANIA...|                 $4763.73|                         AL - Dothan|[36301,
 DOTHAN, [...|      10001|              91|SOUTHEAST ALABAMA...|
|              $5787.57|[2505 U S HIGHWAY...|              $15131.85|039 -
 EXTRACRANIA...|                 $4976.71|                     AL - Birmingham|[35957,
 BOAZ, [25...|      10005|              14|MARSHALL MEDICAL ...|
|              $5434.95|[205 MARENGO STRE...|              $37560.37|039 -
 EXTRACRANIA...|                 $4453.79|                     AL - Birmingham|[35631,
 FLORENCE,...|      10006|              24|ELIZA COFFEE MEMO...|
|              $5417.56|[50 MEDICAL PARK ...|              $13998.28|039 -
 EXTRACRANIA...|                 $4129.16|                     AL - Birmingham|[35235,
 BIRMINGHA...|      10011|              25|   ST VINCENT'S EAST|
|              $5658.33|[1000 FIRST STREE...|              $31633.27|039 -
 EXTRACRANIA...|                 $4851.44|                     AL - Birmingham|[35007,
 ALABASTER...|      10016|              18|SHELBY BAPTIST ME...|
|              $6653.80|[2105 EAST SOUTH ...|              $16920.79|039 -
 EXTRACRANIA...|                 $5374.14|                     AL - Montgomery|[36116,
 MONTGOMER...|      10023|              67|BAPTIST MEDICAL C...|
|              $5834.74|[2000 PEPPERELL P...|              $11977.13|039 -
 EXTRACRANIA...|                 $4761.41|                     AL - Birmingham|[36801,
 OPELIKA, ...|      10029|              51|EAST ALABAMA MEDI...|
|              $8031.12|[619 SOUTH 19TH S...|              $35841.09|039 -
 EXTRACRANIA...|                 $5858.50|                     AL - Birmingham|[35233,
 BIRMINGHA...|      10033|              32|UNIVERSITY OF ALA...|
```

```
|              $6113.38|[101 SIVLEY RD, H...|              $28523.39|039 -
 EXTRACRANIA...|                $5228.40|                     AL - Huntsville|[35801,
 HUNTSVILL...|      10039|             135| HUNTSVILLE HOSPITAL|
|              $5541.05|[1007 GOODYEAR AV...|              $75233.38|039 -
 EXTRACRANIA...|                $4386.94|                     AL - Birmingham|[35903,
 GADSDEN, ...|      10040|              34|GADSDEN REGIONAL ...|
|              $5461.57|[600 SOUTH THIRD ...|              $67327.92|039 -
 EXTRACRANIA...|                $4493.57|                     AL - Birmingham|[35901,
 GADSDEN, ...|      10046|              14|RIVERVIEW REGIONA...|
|              $5356.28|[4370 WEST MAIN S...|              $39607.28|039 -
 EXTRACRANIA...|                $4408.20|                         AL - Dothan|[36305,
 DOTHAN, [...|      10055|              45|    FLOWERS HOSPITAL|
|              $5374.65|[810 ST VINCENT'S...|              $22862.23|039 -
 EXTRACRANIA...|                $4186.02|                     AL - Birmingham|[35205,
 BIRMINGHA...|      10056|              43|ST VINCENT'S BIRM...|
|              $5366.23|[400 EAST 10TH ST...|              $31110.85|039 -
 EXTRACRANIA...|                $4376.23|                     AL - Birmingham|[36207,
 ANNISTON,...|      10078|              21|NORTHEAST ALABAMA...|
|              $5282.93|[1613 NORTH MCKEN...|              $25411.33|039 -
 EXTRACRANIA...|                $4383.73|                         AL - Mobile|[36535,
 FOLEY, [1...|      10083|              15|SOUTH BALDWIN REG...|
|              $5676.55|[1201 7TH STREET ...|               $9234.51|039 -
 EXTRACRANIA...|                $4509.11|                     AL - Huntsville|[35609,
 DECATUR, ...|      10085|              27|DECATUR GENERAL H...|
|              $5930.11|[6801 AIRPORT BOU...|              $15895.85|039 -
 EXTRACRANIA...|                $3972.85|                         AL - Mobile|[36608,
 MOBILE, [...|      10090|              27| PROVIDENCE HOSPITAL|
|              $6192.54|[809 UNIVERSITY B...|              $19721.16|039 -
 EXTRACRANIA...|                $5179.38|                     AL - Tuscaloosa|[35401,
 TUSCALOOS...|      10092|              31|D C H REGIONAL ME...|
|              $4968.00|[750 MORPHY AVENU...|              $10710.88|039 -
 EXTRACRANIA...|                $3898.88|                         AL - Mobile|[36532,
 FAIRHOPE,...|      10100|              18|     THOMAS HOSPITAL|
|              $5996.00|[701 PRINCETON AV...|              $51343.75|039 -
 EXTRACRANIA...|                $4962.45|                     AL - Birmingham|[35211,
 BIRMINGHA...|      10103|              33|BAPTIST MEDICAL C...|
+----------------------+--------------------+-----------------------
+--------------------+------------------------+-----------------------------------
+--------------------+----------+---------------+--------------------+
only showing top 20 rows
```

serikat

union(frame1, frame2, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Serikat dua DynamicFrames. Pengembalian DynamicFrame yang berisi semua catatan dari kedua input DynamicFrames. Transformasi ini dapat mengembalikan hasil yang berbeda dari penyatuan dua DataFrames dengan data yang setara. Jika Anda membutuhkan perilaku DataFrame serikat Spark, pertimbangkan untuk menggunakan`toDF`.

- `frame1`Pertama DynamicFrame ke serikat pekerja.
- `frame2`Kedua setelah DynamicFrame serikat pekerja.
- `transformation_ctx`— (opsional) String unik yang digunakan untuk mengidentifikasi statistik/informasi negara
- `info`— (opsional) String apa pun yang akan dikaitkan dengan kesalahan dalam transformasi
- `stageThreshold`— (opsional) Jumlah maksimum kesalahan dalam transformasi sampai pemrosesan akan error
- `totalThreshold`— (opsional) Jumlah maksimum kesalahan total sampai pemrosesan akan error.

unnest

unnest(transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)

Menyingkirkan objek bersarang di a`DynamicFrame`, yang menjadikannya objek tingkat atas, dan mengembalikan unnested baru. `DynamicFrame`

- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).
- `stageThreshold`— Jumlah kesalahan yang ditemui selama transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.
- `totalThreshold`— Jumlah kesalahan yang ditemui hingga dan termasuk transformasi ini di mana proses harus error out (opsional). Defaultnya adalah nol, yang menunjukkan bahwa proses tidak boleh error.

Contoh: Gunakan unnest untuk mengubah bidang bersarang menjadi bidang tingkat atas

Contoh kode ini menggunakan `unnest` metode untuk meratakan semua bidang bersarang di dalam bidang tingkat atas. `DynamicFrame`

Contoh dataset

Contoh menggunakan `DynamicFrame` dipanggil `mapped_medicare` dengan skema berikut. Perhatikan bahwa `Address` bidang adalah satu-satunya bidang yang berisi data bersarang.

```
root
|-- Average Total Payments: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address: struct
|    |-- Zip.Code: string
|    |-- City: string
|    |-- Array: array
|    |    |-- element: string
|    |-- State: string
|    |-- Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string
```

Contoh kode

```
# Example: Use unnest to unnest nested
# objects in a DynamicFrame

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Unnest all nested fields
unnested = mapped_medicare.unnest()
unnested.printSchema()
```

Output

```
root
|-- Average Total Payments: string
|-- Average Covered Charges: string
|-- DRG Definition: string
|-- Average Medicare Payments: string
|-- Hospital Referral Region Description: string
|-- Address.Zip.Code: string
|-- Address.City: string
|-- Address.Array: array
|    |-- element: string
|-- Address.State: string
|-- Address.Street: string
|-- Provider Id: string
|-- Total Discharges: string
|-- Provider Name: string
```

unnest_ddb_json

Unnests kolom bersarang di a `DynamicFrame` yang secara khusus dalam struktur DynamoDB JSON, dan mengembalikan unnested baru. `DynamicFrame` Kolom yang terdiri dari array tipe struct tidak akan di-unnested. Perhatikan bahwa ini adalah jenis transformasi unnesting tertentu yang berperilaku berbeda dari `unnest` transformasi biasa dan mengharuskan data sudah berada dalam struktur DynamoDB JSON. Untuk informasi selengkapnya, lihat DynamoDB JSON.

`unnest_ddb_json(transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)`

- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan pelaporan kesalahan untuk transformasi ini (opsional).
- `stageThreshold` — Jumlah kesalahan yang dihadapi selama transformasi ini di mana proses mengeluarkan kesalahan (opsional: nol secara default, menunjukkan bahwa proses tidak mengeluarkan kesalahan).
- `totalThreshold` — Jumlah kesalahan yang dihadapi hingga dan termasuk transformasi ini di mana proses mengeluarkan kesalahan (opsional: nol secara default, menunjukkan bahwa proses tidak mengeluarkan kesalahan).

Misalnya, skema pembacaan ekspor dengan struktur DynamoDB JSON mungkin terlihat seperti berikut:

```
root
|-- Item: struct
|    |-- ColA: struct
|    |    |-- S: string
|    |-- ColB: struct
|    |    |-- S: string
|    |-- ColC: struct
|    |    |-- N: string
|    |-- ColD: struct
|    |    |-- L: array
|    |    |    |-- element: null
```

`unnest_ddb_json()`Transformasi akan mengubah ini menjadi:

```
root
|-- ColA: string
|-- ColB: string
|-- ColC: string
|-- ColD: array
|    |-- element: null
```

Contoh kode berikut menunjukkan cara menggunakan konektor ekspor AWS Glue DynamoDB, memanggil DynamoDB JSON unnest, dan mencetak jumlah partisi:

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dynamicFrame = glue_context.create_dynamic_frame.from_options(
    connection_type="dynamodb",
    connection_options={
        "dynamodb.export": "ddb",
```

```
        "dynamodb.tableArn": "<test_source>",
        "dynamodb.s3.bucket": "<bucket name>",
        "dynamodb.s3.prefix": "<bucket prefix>",
        "dynamodb.s3.bucketOwner": "<account_id>",
    }
)
unnested = dynamicFrame.unnest_ddb_json()
print(unnested.getNumPartitions())

job.commit()
```

tulis

`write(connection_type, connection_options, format, format_options, accumulator_size)`

Mendapatkan DataSink(objek) dari jenis koneksi yang ditentukan dari GlueContext kelas dari `DynamicFrame` ini, dan menggunakannya untuk memformat dan menulis isi dari `DynamicFrame` ini. Mengembalikan `DynamicFrame` baru yang sudah diformat dan ditulis sebagaimana ditentukan.

- `connection_type` — Jenis koneksi yang akan digunakan. Nilai yang valid termasuk `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, dan `oracle`.
- `connection_options` — Opsi koneksi yang akan digunakan (opsional). Untuk `connection_type` dari `s3`, path Amazon S3 didefinisikan.

  ```
  connection_options = {"path": "s3://aws-glue-target/temp"}
  ```

  Untuk koneksi JDBC, beberapa properti harus didefinisikan. Perhatikan bahwa nama basis data harus menjadi bagian dari URL. Secara opsional dapat disertakan dalam opsi koneksi.

  > **Warning**
  >
  > Menyimpan kata sandi dalam skrip Anda tidak disarankan. Pertimbangkan `boto3` untuk menggunakan untuk mengambilnya dari AWS Secrets Manager atau Katalog Data AWS Glue.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-path"}
```

- `format` — Format spesifikasi (opsional). Ini digunakan untuk Amazon Simple Storage Service (Amazon S3) atau koneksi AWS Glue yang mendukung berbagai format. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `format_options` — Pilihan format untuk format yang ditentukan. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `accumulator_size`— Ukuran yang dapat diakumulasikan untuk digunakan, dalam byte (opsional).

— kesalahan —

- assertErrorThreshold
- errorsAsDynamicBingkai
- errorsCount
- stageErrorsCount

assertErrorThreshold

`assertErrorThreshold( )` — Sebuah pernyataan untuk kesalahan dalam transformasi yang menciptakan `DynamicFrame` ini. Mengembalikan sebuah `Exception` dari `DataFrame` yang mendasari.

errorsAsDynamicBingkai

`errorsAsDynamicFrame( )` — Mengembalikan `DynamicFrame` yang memiliki catatan kesalahan yang di-nest di dalamnya.

Contoh: Gunakan errorsAsDynamic Frame untuk melihat catatan kesalahan

Contoh kode berikut menunjukkan cara menggunakan `errorsAsDynamicFrame` metode untuk melihat catatan kesalahan untuk file`DynamicFrame`.

Contoh dataset

Contoh menggunakan kumpulan data berikut yang dapat Anda unggah ke Amazon S3 sebagai JSON. Perhatikan bahwa catatan kedua cacat. Data yang salah bentuk biasanya merusak penguraian file saat Anda menggunakan SparkSQL. Namun, `DynamicFrame` mengenali masalah malformasi dan mengubah garis cacat menjadi catatan kesalahan yang dapat Anda tangani satu per satu.

```
{"id": 1, "name": "george", "surname": "washington", "height": 178}
{"id": 2, "name": "benjamin", "surname": "franklin",
{"id": 3, "name": "alexander", "surname": "hamilton", "height": 171}
{"id": 4, "name": "john", "surname": "jay", "height": 190}
```

Contoh kode

```
# Example: Use errorsAsDynamicFrame to view error records.
# Replace s3://DOC-EXAMPLE-S3-BUCKET/error_data.json with your location.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create errors DynamicFrame, view schema
errors = glueContext.create_dynamic_frame.from_options(
    "s3", {"paths": ["s3://DOC-EXAMPLE-S3-BUCKET/error_data.json"]}, "json"
)
print("Schema of errors DynamicFrame:")
errors.printSchema()

# Show that errors only contains valid entries from the dataset
print("errors contains only valid records from the input dataset (2 of 4 records)")
errors.toDF().show()

# View errors
print("Errors count:", str(errors.errorsCount()))
print("Errors:")
errors.errorsAsDynamicFrame().toDF().show()

# View error fields and error data
error_record = errors.errorsAsDynamicFrame().toDF().head()
```

```
error_fields = error_record["error"]
print("Error fields: ")
print(error_fields.asDict().keys())

print("\nError record data:")
for key in error_fields.asDict().keys():
    print("\n", key, ": ", str(error_fields[key]))
```

Output

```
Schema of errors DynamicFrame:
root
|-- id: int
|-- name: string
|-- surname: string
|-- height: int

errors contains only valid records from the input dataset (2 of 4 records)
+---+------+----------+------+
| id|  name|   surname|height|
+---+------+----------+------+
|  1|george|washington|   178|
|  4|  john|       jay|   190|
+---+------+----------+------+

Errors count: 1
Errors:
+--------------------+
|               error|
+--------------------+
|[[  File "/tmp/20...|
+--------------------+

Error fields:
dict_keys(['callsite', 'msg', 'stackTrace', 'input', 'bytesread', 'source',
 'dynamicRecord'])

Error record data:

 callsite :  Row(site='  File "/tmp/2060612586885849088", line 549, in <module>\n
 sys.exit(main())\n  File "/tmp/2060612586885849088", line 523, in main\n    response
 = handler(content)\n  File "/tmp/2060612586885849088", line 197, in execute_request
\n    result = node.execute()\n  File "/tmp/2060612586885849088", line 103, in
```

```
 execute\n    exec(code, global_dict)\n  File "<stdin>", line 10, in <module>\n
 File "/opt/amazon/lib/python3.6/site-packages/awsglue/dynamicframe.py", line 625, in
 from_options\n    format_options, transformation_ctx, push_down_predicate, **kwargs)\n
  File "/opt/amazon/lib/python3.6/site-packages/awsglue/context.py", line 233, in
 create_dynamic_frame_from_options\n    source.setFormat(format, **format_options)\n',
 info='')

 msg :  error in jackson reader

 stackTrace :  com.fasterxml.jackson.core.JsonParseException: Unexpected character
 ('{' (code 123)): was expecting either valid name character (for unquoted name) or
 double-quote (for quoted) to start field name
 at [Source: com.amazonaws.services.glue.readers.BufferedStream@73492578; line: 3,
 column: 2]
 at com.fasterxml.jackson.core.JsonParser._constructError(JsonParser.java:1581)
 at
 com.fasterxml.jackson.core.base.ParserMinimalBase._reportError(ParserMinimalBase.java:533)
 at
 com.fasterxml.jackson.core.base.ParserMinimalBase._reportUnexpectedChar(ParserMinimalBase.java
 at
 com.fasterxml.jackson.core.json.UTF8StreamJsonParser._handleOddName(UTF8StreamJsonParser.java:
 at
 com.fasterxml.jackson.core.json.UTF8StreamJsonParser._parseName(UTF8StreamJsonParser.java:1650
 at
 com.fasterxml.jackson.core.json.UTF8StreamJsonParser.nextToken(UTF8StreamJsonParser.java:740)
 at com.amazonaws.services.glue.readers.JacksonReader$$anonfun$hasNextGoodToken
$1.apply(JacksonReader.scala:57)
 at com.amazonaws.services.glue.readers.JacksonReader$$anonfun$hasNextGoodToken
$1.apply(JacksonReader.scala:57)
 at scala.collection.Iterator$$anon$9.next(Iterator.scala:162)
 at scala.collection.Iterator$$anon$16.hasNext(Iterator.scala:599)
 at scala.collection.Iterator$$anon$16.hasNext(Iterator.scala:598)
 at scala.collection.Iterator$class.foreach(Iterator.scala:891)
 at scala.collection.AbstractIterator.foreach(Iterator.scala:1334)
 at com.amazonaws.services.glue.readers.JacksonReader$$anonfun
$1.apply(JacksonReader.scala:120)
 at com.amazonaws.services.glue.readers.JacksonReader$$anonfun
$1.apply(JacksonReader.scala:116)
 at
 com.amazonaws.services.glue.DynamicRecordBuilder.handleErr(DynamicRecordBuilder.scala:209)
 at
 com.amazonaws.services.glue.DynamicRecordBuilder.handleErrorWithException(DynamicRecordBuilder
 at
 com.amazonaws.services.glue.readers.JacksonReader.nextFailSafe(JacksonReader.scala:116)
```

```
 at com.amazonaws.services.glue.readers.JacksonReader.next(JacksonReader.scala:109)
 at com.amazonaws.services.glue.readers.JSONReader.next(JSONReader.scala:247)
 at
 com.amazonaws.services.glue.hadoop.TapeHadoopRecordReaderSplittable.nextKeyValue(TapeHadoopRec
 at org.apache.spark.rdd.NewHadoopRDD$$anon$1.hasNext(NewHadoopRDD.scala:230)
 at org.apache.spark.InterruptibleIterator.hasNext(InterruptibleIterator.scala:37)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$13.hasNext(Iterator.scala:462)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$13.hasNext(Iterator.scala:462)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:409)
 at org.apache.spark.sql.execution.SparkPlan$$anonfun$2.apply(SparkPlan.scala:255)
 at org.apache.spark.sql.execution.SparkPlan$$anonfun$2.apply(SparkPlan.scala:247)
 at org.apache.spark.rdd.RDD$$anonfun$mapPartitionsInternal$1$$anonfun$apply
$24.apply(RDD.scala:836)
 at org.apache.spark.rdd.RDD$$anonfun$mapPartitionsInternal$1$$anonfun$apply
$24.apply(RDD.scala:836)
 at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.scala:52)
 at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:324)
 at org.apache.spark.rdd.RDD.iterator(RDD.scala:288)
 at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.scala:52)
 at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:324)
 at org.apache.spark.rdd.RDD.iterator(RDD.scala:288)
 at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:90)
 at org.apache.spark.scheduler.Task.run(Task.scala:121)
 at org.apache.spark.executor.Executor$TaskRunner$$anonfun$10.apply(Executor.scala:408)
 at org.apache.spark.util.Utils$.tryWithSafeFinally(Utils.scala:1360)
 at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:414)
 at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
 at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
 at java.lang.Thread.run(Thread.java:750)


 input :

 bytesread :  252

 source :

 dynamicRecord :  Row(id=2, name='benjamin', surname='franklin')
```

errorsCount

`errorsCount( )` — Mengembalikan jumlah kesalahan dalam sebuah `DynamicFrame`.

stageErrorsCount

`stageErrorsCount` — Mengembalikan jumlah kesalahan yang terjadi dalam proses membuat `DynamicFrame` ini.

DynamicFrameCollection kelas

Sebuah `DynamicFrameCollection` adalah kamus objek DynamicFrame kelas, di mana kuncinya adalah nama-nama dari `DynamicFrames`, dan nilai-nilai-nya adalah objek `DynamicFrame`.

__init__

`__init__(dynamic_frames, glue_ctx)`

- `dynamic_frames` — Kamus objek DynamicFrame kelas.
- `glue_ctx` — Sebuah objek GlueContext kelas.

Kunci

`keys( )` — Mengembalikan daftar kunci dalam koleksi ini, yang umumnya terdiri dari nama-nama yang sesuai dengan nilai `DynamicFrame`.

Nilai

`values(key)` — Mengembalikan daftar nilai `DynamicFrame` dalam koleksi ini.

Pilih

`select(key)`

Mengembalikan `DynamicFrame` yang sesuai dengan kunci yang ditentukan (yang umumnya nama dari `DynamicFrame`).

- `key` — Sebuah kunci dalam `DynamicFrameCollection`, yang biasanya mewakili nama `DynamicFrame`.

Map

`map(callable, transformation_ctx="")`

Menggunakan fungsi passed-in untuk membuat dan mengembalikan `DynamicFrameCollection` baru berdasarkan `DynamicFrames` dalam koleksi ini.

- `callable` — Sebuah fungsi yang mengambil `DynamicFrame` dan konteks transformasi yang ditentukan sebagai parameter dan mengembalikan sebuah `DynamicFrame`.
- `transformation_ctx` — Sebuah konteks transformasi yang akan digunakan oleh callable (opsional).

Peta datar

`flatmap(f, transformation_ctx="")`

Menggunakan fungsi passed-in untuk membuat dan mengembalikan `DynamicFrameCollection` baru berdasarkan `DynamicFrames` dalam koleksi ini.

- `f` — Sebuah fungsi yang mengambil `DynamicFrame` sebagai parameter dan mengembalikan `DynamicFrame` atau `DynamicFrameCollection`.
- `transformation_ctx` — Sebuah konteks transformasi yang akan digunakan oleh fungsi (opsional).

DynamicFrameWriter kelas

metode

- __init__
- from_options
- from_catalog
- from_jdbc_conf

__init__

`__init__(glue_context)`

- `glue_context` — GlueContext kelas yang akan digunakan.

from_options

`from_options(frame, connection_type, connection_options={}, format=None, format_options={}, transformation_ctx="")`

Menulis sebuah `DynamicFrame` menggunakan koneksi dan format yang ditentukan.

- `frame` — `DynamicFrame` yang akan ditulis.
- `connection_type` — Jenis koneksi. Nilai yang valid termasuk `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, dan `oracle`.
- `connection_options` — Pilihan koneksi, seperti path dan tabel basis data (opsional). Untuk `connection_type` dari `s3`, path Amazon S3 didefinisikan.

```
connection_options = {"path": "s3://aws-glue-target/temp"}
```

Untuk koneksi JDBC, beberapa properti harus didefinisikan. Perhatikan bahwa nama basis data harus menjadi bagian dari URL. Secara opsional dapat disertakan dalam opsi koneksi.

> **Warning**
>
> Menyimpan kata sandi dalam skrip Anda tidak disarankan. Pertimbangkan `boto3` untuk menggunakan untuk mengambilnya dari AWS Secrets Manager atau Katalog Data AWS Glue.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-path"}
```

Properti `dbtable` adalah nama tabel JDBC. Untuk penyimpanan data JDBC yang mendukung skema dalam basis data, tentukan `schema.table-name`. Jika skema tidak disediakan, maka skema "publik" default digunakan.

Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark.

- `format` — Format spesifikasi (opsional). Ini digunakan untuk Amazon Simple Storage Service (Amazon S3) atau koneksi AWS Glue yang mendukung berbagai format. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.

- `format_options` — Pilihan format untuk format yang ditentukan. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `transformation_ctx` — Sebuah konteks transformasi yang akan digunakan (opsional).

from_catalog

`from_catalog(frame, name_space, table_name, redshift_tmp_dir="", transformation_ctx="")`

Menulis sebuah `DynamicFrame` menggunakan basis data katalog dan nama tabel yang ditentukan.

- `frame` — `DynamicFrame` yang akan ditulis.
- `name_space` — Basis data yang akan digunakan.
- `table_name` — `table_name` yang akan digunakan.
- `redshift_tmp_dir` — Sebuah direktori sementara Amazon Redshift yang akan digunakan (opsional).
- `transformation_ctx` — Sebuah konteks transformasi yang akan digunakan (opsional).
- `additional_options`— Opsi tambahan yang disediakan untukAWS Glue.

  Untuk menulis ke tabel yang Lake Formation diatur, Anda dapat menggunakan opsi tambahan ini:

  - `transactionId`— (String) ID transaksi di mana untuk melakukan penulisan ke tabel yang Diatur. Transaksi ini tidak dapat dilakukan atau dibatalkan, atau penulisan akan gagal.
  - `callDeleteObjectsOnCancel` — (Boolean, opsional) Jika diatur ke `true` (default), AWS Glue secara otomatis memanggil `DeleteObjectsOnCancel` API setelah objek ditulis ke Amazon S3. Untuk informasi lebih lanjut, lihat DeleteObjectsOnCancel dalam Panduan DeveloperAWS Lake Formation.

  Example Contoh: Menulis ke tabel yang diatur di Lake Formation

```
txId = glueContext.start_transaction(read_only=False)
glueContext.write_dynamic_frame.from_catalog(
    frame=dyf,
    database = db,
    table_name = tbl,
    transformation_ctx = "datasource0",
    additional_options={"transactionId":txId})
...
```

```
glueContext.commit_transaction(txId)
```

from_jdbc_conf

`from_jdbc_conf(frame, catalog_connection, connection_options={}, redshift_tmp_dir = "", transformation_ctx="")`

Menulis sebuah `DynamicFrame` menggunakan informasi koneksi JDBC yang ditentukan.

- `frame` — `DynamicFrame` yang akan ditulis.
- `catalog_connection` — Koneksi katalog yang akan digunakan.
- `connection_options` — Pilihan koneksi, seperti path dan tabel basis data (opsional).
- `redshift_tmp_dir` — Sebuah direktori sementara Amazon Redshift yang akan digunakan (opsional).
- `transformation_ctx` — Sebuah konteks transformasi yang akan digunakan (opsional).

Example for write_dynamic_frame

Contoh ini menulis output secara lokal menggunakan `connection_type` dari S3 dengan argumen path POSIX di `connection_options`, yang memungkinkan penulisan ke penyimpanan lokal.

```
glueContext.write_dynamic_frame.from_options(\
frame = dyf_splitFields,\
connection_options = {'path': '/home/glue/GlueLocalOutput/'},\
connection_type = 's3',\
format = 'json')
```

DynamicFrameReader kelas

— metode —

- __init__
- from_rdd
- from_options
- from_catalog

__init__

`__init__(glue_context)`

- `glue_context` — GlueContext kelas yang akan digunakan.

from_rdd

`from_rdd(data, name, schema=None, sampleRatio=None)`

Membaca sebuah `DynamicFrame` dari Resilient Distributed Dataset (RDD).

- `data` — Set data tempat untuk membaca.
- `name` — Nama tempat untuk membaca.
- `schema` — Skema yang akan dibaca (opsional).
- `sampleRatio` — Rasio sampel (opsional).

from_options

`from_options(connection_type, connection_options={}, format=None, format_options={}, transformation_ctx="")`

Membaca sebuah `DynamicFrame` menggunakan koneksi dan format yang ditentukan.

- `connection_type` — Jenis koneksi. Nilai yang valid meliputi `s3`mysql,`postgresql`,,`redshift`,`sqlserver`,`oracle`,`dynamodb`, dan`snowflake`.
- `connection_options` — Pilihan koneksi, seperti path dan tabel basis data (opsional). Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue for Spark. Untuk sebuah `connection_type` dari `s3`, path Amazon S3 didefinisikan dalam sebuah array.

  ```
  connection_options = {"paths": [ "s3://mybucket/object_a", "s3://mybucket/object_b"]}
  ```

  Untuk koneksi JDBC, beberapa properti harus didefinisikan. Perhatikan bahwa nama basis data harus menjadi bagian dari URL. Secara opsional dapat disertakan dalam opsi koneksi.

> **Warning**
>
> Menyimpan kata sandi dalam skrip Anda tidak disarankan. Pertimbangkan `boto3` untuk menggunakan untuk mengambilnya dari AWS Secrets Manager atau Katalog Data AWS Glue.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-
path"}
```

Untuk koneksi JDBC yang melakukan pembacaan paralel, Anda dapat mengatur opsi hashfield. Sebagai contoh:

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-
path" , "hashfield": "month"}
```

Untuk informasi selengkapnya, lihat Membaca dari tabel JDBC secara paralel.

- `format` — Format spesifikasi (opsional). Ini digunakan untuk Amazon Simple Storage Service (Amazon S3) atau koneksi AWS Glue yang mendukung berbagai format. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `format_options` — Pilihan format untuk format yang ditentukan. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).
- `push_down_predicate` — Memfilter partisi tanpa harus mencantumkan dan membaca semua file dalam set data Anda. Untuk informasi selengkapnya, lihat Pra-Pemfilteran Menggunakan Predikat Pushdown.

from_catalog

`from_catalog(database, table_name, redshift_tmp_dir="", transformation_ctx="", push_down_predicate="", additional_options={})`

Membaca sebuah `DynamicFrame` menggunakan namespace katalog tertentu dan nama tabel.

- `database` — Basis data tempat untuk membaca.
- `table_name` — Nama tempat untuk dibaca.
- `redshift_tmp_dir` — Sebuah direktori sementara Amazon Redshift yang akan digunakan (opsional, jika bukan pembacaan data dari Redshift).
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).
- `push_down_predicate` — Memfilter partisi tanpa harus mencantumkan dan membaca semua file dalam set data Anda. Untuk informasi selengkapnya, lihat Pra-penyaringan menggunakan predikat pushdown.
- `additional_options`— Opsi tambahan yang disediakan untukAWS Glue.
  - Untuk menggunakan koneksi JDBC yang melakukan pembacaan paralel, Anda dapat mengatur pilihan `hashfield`, `hashexpression`, atau `hashpartitions`. Sebagai contoh:

    ```
    additional_options = {"hashfield": "month"}
    ```

    Untuk informasi selengkapnya, lihat Membaca dari tabel JDBC secara paralel.
  - Untuk meneruskan ekspresi katalog untuk memfilter berdasarkan kolom indeks, Anda dapat melihat `catalogPartitionPredicate` opsi.

    `catalogPartitionPredicate`— Anda dapat meneruskan ekspresi katalog untuk memfilter berdasarkan kolom indeks. Ini mendorong penyaringan ke sisi server. Untuk informasi selengkapnya, lihat Indeks AWS Glue Partisi. Perhatikan itu `push_down_predicate` dan `catalogPartitionPredicate` gunakan sintaks yang berbeda. Yang pertama menggunakan sintaks standar Spark SQL dan yang kemudian menggunakan parser JSQL.

    Untuk informasi selengkapnya, lihat Mengelola partisi untuk output ETL di AWS Glue.

GlueContext kelas

Membungkus SparkContextobjek Apache Spark, dan dengan demikian menyediakan mekanisme untuk berinteraksi dengan platform Apache Spark.

__init__

`__init__(sparkContext)`

- `sparkContext` — Konteks Apache Spark yang akan digunakan.

Membuat

- __init__
- getSource
- create_dynamic_frame_from_rdd
- create_dynamic_frame_from_catalog
- create_dynamic_frame_from_options
- create_sample_dynamic_frame_from_catalog
- create_sample_dynamic_frame_from_options
- add_ingestion_time_columns
- create_data_frame_from_catalog
- create_data_frame_from_options
- forEachBatch

getSource

`getSource(connection_type, transformation_ctx = "", **options)`

Membuat objek `DataSource` yang dapat digunakan untuk membaca `DynamicFrames` dari sumber eksternal.

- `connection_type` — Jenis koneksi yang digunakan, seperti Amazon Simple Storage Service (Amazon S3), Amazon Redshift, dan JDBC. Nilai yang valid termasuk `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, `oracle`, dan dynamodb.
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).
- `options` — Kumpulan pasangan nama-nilai opsional. Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark.

Berikut adalah contoh penggunaan `getSource`.

```
>>> data_source = context.getSource("file", paths=["/in/path"])
>>> data_source.setFormat("json")
>>> myFrame = data_source.getFrame()
```

create_dynamic_frame_from_rdd

`create_dynamic_frame_from_rdd(data, name, schema=None, sample_ratio=None, transformation_ctx="")`

Mengembalikan sebuah `DynamicFrame` yang dibuat dari Apache Spark Resilient Distributed Dataset (RDD).

- `data` — Sumber data yang akan digunakan.
- `name` — Nama data yang akan digunakan.
- `schema` — Skema yang akan digunakan (opsional).
- `sample_ratio` — Rasio sampel yang akan digunakan (opsional).
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).

create_dynamic_frame_from_catalog

`create_dynamic_frame_from_catalog(database, table_name, redshift_tmp_dir, transformation_ctx = "", push_down_predicate= "", additional_options = {}, catalog_id = None)`

Mengembalikan sebuah `DynamicFrame` yang dibuat menggunakan basis data dan nama tabel Katalog Data. Saat menggunakan metode ini, Anda menyediakan `format_options` properti tabel pada tabel Katalog Data AWS Glue yang ditentukan dan opsi lain melalui `additional_options` argumen.

- `Database` — Basis data tempat untuk membaca.
- `table_name` — Nama tempat untuk dibaca.
- `redshift_tmp_dir` — Sebuah direktori sementara Amazon Redshift yang akan digunakan (opsional).
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).
- `push_down_predicate` — Memfilter partisi tanpa harus mencantumkan dan membaca semua file dalam set data Anda. Untuk sumber dan batasan yang didukung, lihat Mengoptimalkan pembacaan dengan pushdown di Glue AWS ETL. Untuk informasi selengkapnya, lihat Pra-penyaringan menggunakan predikat pushdown.
- `additional_options` — Kumpulan pasangan nama-nilai opsional. Opsi yang mungkin adalah opsi-opsi yang tercantum dalam Jenis dan opsi koneksi untuk ETL di

AWS Glue untuk Spark kecuali `endpointUrl`, `streamName`, `bootstrap.servers`, `security.protocol`, `topicName`, `classification`, dan `delimiter`. Opsi lain yang didukung adalah`catalogPartitionPredicate`:

`catalogPartitionPredicate`— Anda dapat meneruskan ekspresi katalog untuk memfilter berdasarkan kolom indeks. Ini mendorong penyaringan ke sisi server. Untuk informasi selengkapnya, lihat Indeks AWS Glue Partisi. Perhatikan itu `push_down_predicate` dan `catalogPartitionPredicate` gunakan sintaks yang berbeda. Yang pertama menggunakan sintaks standar Spark SQL dan yang kemudian menggunakan parser JSQL.

- `catalog_id` — ID katalog (ID akun) dari Katalog Data yang sedang diakses. Bila Tidak Ada, maka ID akun default pemanggil yang akan digunakan.

create_dynamic_frame_from_options

`create_dynamic_frame_from_options(connection_type, connection_options={}, format=None, format_options={}, transformation_ctx = "")`

Mengembalikan sebuah `DynamicFrame` dibuat dengan koneksi dan format yang ditentukan.

- `connection_type` — Jenis koneksi, seperti Amazon S3, Amazon Redshift, dan JDBC. Nilai yang valid termasuk `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, `oracle`, dan dynamodb.
- `connection_options` — Pilihan koneksi, seperti path dan tabel basis data (opsional). Untuk `connection_type` dari `s3`, daftar path Amazon S3 didefinisikan.

```
connection_options = {"paths": ["s3://aws-glue-target/temp"]}
```

Untuk koneksi JDBC, beberapa properti harus didefinisikan. Perhatikan bahwa nama basis data harus menjadi bagian dari URL. Secara opsional dapat disertakan dalam opsi koneksi.

> **Warning**
>
> Menyimpan kata sandi dalam skrip Anda tidak disarankan. Pertimbangkan `boto3` untuk menggunakan untuk mengambilnya dari AWS Secrets Manager atau Katalog Data AWS Glue.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-
path"}
```

Properti `dbtable` adalah nama tabel JDBC. Untuk penyimpanan data JDBC yang mendukung skema dalam basis data, tentukan `schema.table-name`. Jika skema tidak disediakan, maka skema "publik" default digunakan.

Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark.

- `format`— Sebuah spesifikasi format. Ini digunakan untuk Amazon S3 atau AWS Glue koneksi yang mendukung berbagai format. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `format_options` — Pilihan format untuk format yang ditentukan. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).
- `push_down_predicate` — Memfilter partisi tanpa harus mencantumkan dan membaca semua file dalam set data Anda. Untuk sumber dan batasan yang didukung, lihat Mengoptimalkan pembacaan dengan pushdown di Glue AWS ETL. Untuk informasi selengkapnya, lihat Pra-Pemfilteran Menggunakan Predikat Pushdown.

create_sample_dynamic_frame_from_catalog

`create_sample_dynamic_frame_from_catalog(database, table_name, num, redshift_tmp_dir, transformation_ctx = "", push_down_predicate= "", additional_options = {}, sample_options = {}, catalog_id = None)`

Mengembalikan sampel `DynamicFrame` yang dibuat menggunakan database Data Catalog dan nama tabel. `DynamicFrame`Satu-satunya berisi `num` catatan pertama dari sumber data.

- `database` — Basis data tempat untuk membaca.
- `table_name` — Nama tempat untuk dibaca.
- num— Jumlah maksimum catatan dalam bingkai dinamis sampel yang dikembalikan.
- `redshift_tmp_dir` — Sebuah direktori sementara Amazon Redshift yang akan digunakan (opsional).

- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).
- `push_down_predicate` — Memfilter partisi tanpa harus mencantumkan dan membaca semua file dalam set data Anda. Untuk informasi selengkapnya, lihat Pra-penyaringan menggunakan predikat pushdown.
- `additional_options` — Kumpulan pasangan nama-nilai opsional. Opsi yang mungkin adalah opsi-opsi yang tercantum dalam Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark kecuali `endpointUrl`, `streamName`, `bootstrap.servers`, `security.protocol`, `topicName`, `classification`, dan `delimiter`.
- `sample_options`— Parameter untuk mengontrol perilaku pengambilan sampel (opsional). Parameter yang tersedia saat ini untuk sumber Amazon S3:
  - `maxSamplePartitions`— Jumlah maksimum partisi yang akan dibaca oleh sampling. Nilai default adalah 10
  - `maxSampleFilesPerPartition`— Jumlah maksimum file sampling akan dibaca dalam satu partisi. Nilai default adalah 10.

    Parameter ini membantu mengurangi waktu yang dikonsumsi oleh daftar file. Misalnya, dataset memiliki 1000 partisi, dan setiap partisi memiliki 10 file. Jika Anda mengatur `maxSamplePartitions` = 10, dan `maxSampleFilesPerPartition` = 10, alih-alih mencantumkan semua 10.000 file, pengambilan sampel hanya akan mencantumkan dan membaca 10 partisi pertama dengan 10 file pertama di masing-masing: 10* 10 = 100 file secara total.
- `catalog_id` — ID katalog Katalog Data yang sedang diakses (ID akun Katalog Data). Diatur ke `None` secara default. `None` default ke ID katalog dari akun yang melakukan panggilan dalam layanan.

create_sample_dynamic_frame_from_options

`create_sample_dynamic_frame_from_options(connection_type, connection_options={}, num, sample_options={}, format=None, format_options={}, transformation_ctx = "")`

Mengembalikan sampel `DynamicFrame` yang dibuat dengan koneksi dan format yang ditentukan. `DynamicFrame`Satu-satunya berisi `num` catatan pertama dari sumber data.

- `connection_type` — Jenis koneksi, seperti Amazon S3, Amazon Redshift, dan JDBC. Nilai yang valid termasuk `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, `oracle`, dan `dynamodb`.

- `connection_options` — Pilihan koneksi, seperti path dan tabel basis data (opsional). Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark.
- num— Jumlah maksimum catatan dalam bingkai dinamis sampel yang dikembalikan.
- `sample_options`— Parameter untuk mengontrol perilaku pengambilan sampel (opsional). Parameter yang tersedia saat ini untuk sumber Amazon S3:
  - `maxSamplePartitions`— Jumlah maksimum partisi yang akan dibaca oleh sampling. Nilai default adalah 10
  - `maxSampleFilesPerPartition`— Jumlah maksimum file sampling akan dibaca dalam satu partisi. Nilai default adalah 10.

    Parameter ini membantu mengurangi waktu yang dikonsumsi oleh daftar file. Misalnya, dataset memiliki 1000 partisi, dan setiap partisi memiliki 10 file. Jika Anda mengatur `maxSamplePartitions` = 10, dan `maxSampleFilesPerPartition` = 10, alih-alih mencantumkan semua 10.000 file, pengambilan sampel hanya akan mencantumkan dan membaca 10 partisi pertama dengan 10 file pertama di masing-masing: 10* 10 = 100 file secara total.
- `format`— Sebuah spesifikasi format. Ini digunakan untuk Amazon S3 atau AWS Glue koneksi yang mendukung berbagai format. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `format_options` — Pilihan format untuk format yang ditentukan. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).
- `push_down_predicate` — Memfilter partisi tanpa harus mencantumkan dan membaca semua file dalam set data Anda. Untuk informasi selengkapnya, lihat Pra-penyaringan menggunakan predikat pushdown.

add_ingestion_time_columns

`add_ingestion_time_columns(dataFrame, timeGranularity = "")`

Menambahkan kolom waktu penyerapan seperti `ingest_year`, `ingest_month`, `ingest_day`, `ingest_hour`, `ingest_minute` ke input `DataFrame`. Fungsi ini secara otomatis dihasilkan dalam skrip yang dihasilkan oleh AWS Glue saat Anda menentukan tabel Katalog Data dengan Amazon S3 sebagai target. Fungsi ini secara otomatis memperbarui partisi dengan kolom waktu penyerapan pada tabel output. Hal ini memungkinkan data output dipartisi secara otomatis pada waktu penyerapan tanpa memerlukan kolom waktu penyerapan eksplisit dalam data input.

- `dataFrame` — `dataFrame` yang akan ditambahi dengan kolom waktu penyerapan.
- `timeGranularity` — Kedetailan dari kolom waktu. Nilai yang benar adalah "`day`", "`hour`" dan "`minute`". Misalnya, jika "`hour`" diberikan dalam fungsi, maka `dataFrame` asli akan memiliki kolom waktu "`ingest_year`", "`ingest_month`", "`ingest_day`", dan "`ingest_hour`" yang ditambahkan.

Mengembalikan bingkai data setelah menambahkan kolom kedetailan waktu.

Contoh:

```
dynamic_frame = DynamicFrame.fromDF(glueContext.add_ingestion_time_columns(dataFrame,
 "hour"))
```

create_data_frame_from_catalog

`create_data_frame_from_catalog(database, table_name, transformation_ctx = "", additional_options = {})`

Mengembalikan sebuah `DataFrame` yang dibuat menggunakan informasi dari tabel Katalog Data.

- `database` — Basis data Katalog Data tempat membaca.
- `table_name` — Nama tabel Katalog Data tempat membaca.
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).
- `additional_options` — Kumpulan pasangan nama-nilai opsional. Opsi yang mungkin termasuk yang tercantum dalam Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark untuk sumber streaming, seperti `startingPosition`, `maxFetchTimeInMs`, dan `startingOffsets`.
  - `useSparkDataSource`— Ketika disetel ke true, memaksa AWS Glue untuk menggunakan Spark Data Source API asli untuk membaca tabel. Spark Data Source API mendukung format berikut: AVRO, biner, CSV, JSON, ORC, Parket, dan teks. Dalam tabel Katalog Data, Anda menentukan format menggunakan `classification` properti. Untuk mempelajari lebih lanjut tentang Spark Data Source API, lihat dokumentasi resmi Apache Spark.

    Menggunakan `create_data_frame_from_catalog` dengan `useSparkDataSource` memiliki manfaat sebagai berikut:

    - Langsung mengembalikan a `DataFrame` dan memberikan alternatif untuk`create_dynamic_frame.from_catalog().toDF()`.

- Mendukung kontrol AWS Lake Formation izin tingkat tabel untuk format asli.
- Mendukung membaca format danau data tanpa AWS Lake Formation kontrol izin tingkat tabel. Untuk informasi selengkapnya, lihat Menggunakan kerangka kerja data lake dengan pekerjaan AWS Glue ETL.

Saat Anda mengaktifkan`useSparkDataSource`, Anda juga dapat menambahkan salah satu opsi Sumber Data Spark sesuai `additional_options` kebutuhan. AWS Glue meneruskan opsi ini langsung ke pembaca Spark.

- `useCatalogSchema`— Ketika disetel ke true, AWS Glue menerapkan skema Data Catalog ke hasil`DataFrame`. Jika tidak, pembaca menyimpulkan skema dari data. Ketika Anda mengaktifkan`useCatalogSchema`, Anda juga harus mengatur `useSparkDataSource` ke true.

Batasan

Pertimbangkan batasan berikut saat Anda menggunakan `useSparkDataSource` opsi:

- Saat Anda menggunakan`useSparkDataSource`, AWS Glue membuat yang baru `DataFrame` dalam sesi Spark terpisah yang berbeda dari sesi Spark asli.
- DataFrame Pemfilteran partisi percikan tidak berfungsi dengan fitur AWS Glue berikut.
  - Bookmark Job
  - Tidak termasuk kelas penyimpanan Amazon S3
  - Katalog predikat partisi

  Untuk menggunakan penyaringan partisi dengan fitur-fitur ini, Anda dapat menggunakan predikat AWS pushdown Glue. Untuk informasi selengkapnya, lihat Pra-penyaringan menggunakan predikat pushdown. Pemfilteran pada kolom yang tidak dipartisi tidak terpengaruh.

  Contoh skrip berikut menunjukkan cara yang salah untuk melakukan pemfilteran partisi dengan opsi. `excludeStorageClasses`

```
// Incorrect partition filtering using Spark filter with excludeStorageClasses
read_df = glueContext.create_data_frame.from_catalog(
    database=database_name,
    table_name=table_name,
    additional_options = {
      "useSparkDataSource": True,
      "excludeStorageClasses" : ["GLACIER", "DEEP_ARCHIVE"]
    }
```

```
)

//  Suppose year and month are partition keys.
//  Filtering on year and month won't work, the filtered_df will still
//  contain data with other year/month values.
filtered_df = read_df.filter("year == '2017 and month == '04' and 'state == 'CA'")
```

Contoh skrip berikut menunjukkan cara yang benar untuk menggunakan predikat pushdown untuk melakukan pemfilteran partisi dengan opsi. `excludeStorageClasses`

```
// Correct partition filtering using the AWS Glue pushdown predicate
// with excludeStorageClasses
read_df = glueContext.create_data_frame.from_catalog(
    database=database_name,
    table_name=table_name,
    //  Use AWS Glue pushdown predicate to perform partition filtering
    push_down_predicate = "(year=='2017' and month=='04')"
    additional_options = {
      "useSparkDataSource": True,
      "excludeStorageClasses" : ["GLACIER", "DEEP_ARCHIVE"]
    }
)

//  Use Spark filter only on non-partitioned columns
filtered_df = read_df.filter("state == 'CA'")
```

Contoh: Membuat tabel CSV menggunakan pembaca sumber data Spark

```
//  Read a CSV table with '\t' as separator
read_df = glueContext.create_data_frame.from_catalog(
    database=<database_name>,
    table_name=<table_name>,
    additional_options = {"useSparkDataSource": True,  "sep": '\t'}
)
```

create_data_frame_from_options

`create_data_frame_from_options(connection_type, connection_options={}, format=None, format_options={}, transformation_ctx = "")`

API ini sekarang sudah usang. Sebagai gantinya gunakan `getSource()` API. Mengembalikan sebuah `DataFrame` dibuat dengan koneksi dan format yang ditentukan. Gunakan fungsi ini hanya dengan sumber AWS Glue streaming.

- `connection_type` — Jenis koneksi streaming. Nilai yang valid mencakup `kinesis` dan `kafka`.
- `connection_options` — Pilihan koneksi, yang berbeda untuk Kinesis dan Kafka. Anda dapat menemukan daftar semua opsi koneksi untuk setiap sumber data streaming di Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark. Perhatikan perbedaan berikut dalam pilihan koneksi streaming:
  - Sumber streaming Kinesis memerlukan `streamARN`, `startingPosition`, `inferSchema`, dan `classification`.
  - Sumber streaming Kafka membutuhkan `connectionName`, `topicName`, `startingOffsets`, `inferSchema`, dan `classification`.
- `format`— Sebuah spesifikasi format. Ini digunakan untuk Amazon S3 atau AWS Glue koneksi yang mendukung berbagai format. Untuk informasi tentang format-format yang didukung, lihat Opsi format data untuk input dan output untuk Spark AWS Glue.
- `format_options` — Pilihan format untuk format yang ditentukan. Untuk informasi tentang pilihan format yang didukung, lihat Opsi format data untuk input dan output untuk Spark AWS Glue.
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).

Contoh untuk sumber streaming Amazon Kinesis:

```
kinesis_options =
   { "streamARN": "arn:aws:kinesis:us-east-2:777788889999:stream/fromOptionsStream",
     "startingPosition": "TRIM_HORIZON",
     "inferSchema": "true",
     "classification": "json"
   }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kinesis",
 connection_options=kinesis_options)
```

Contoh untuk sumber streaming Kafka:

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "startingOffsets": "earliest",
      "inferSchema": "true",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

forEachBatch

`forEachBatch(frame, batch_function, options)`

Menerapkan `batch_function` yang diberikan ke setiap batch mikro yang dibaca dari sumber Streaming.

- `frame`— Yang DataFrame berisi batch mikro saat ini.
- `batch_function` — Sebuah fungsi yang akan diterapkan untuk setiap batch mikro.
- `options` — Kumpulan pasangan kunci-nilai yang menyimpan informasi tentang cara memproses batch mikro. Opsi-opsi berikut diperlukan:
  - `windowSize` — Jumlah waktu yang diperlukan untuk pemrosesan setiap batch.
  - `checkpointLocation` — Lokasi di mana pos pemeriksaan disimpan untuk tugas ETL streaming.
  - `batchMaxRetries` — Jumlah waktu maksimum untuk mengulang mencoba batch sekali lagi jika gagal. Nilai default-nya adalah 3. Opsi ini hanya dapat dikonfigurasi untuk Glue versi 2.0 dan di atasnya.

Contoh:

```
glueContext.forEachBatch(
    frame = data_frame_datasource0,
    batch_function = processBatch,
    options = {
        "windowSize": "100 seconds",
        "checkpointLocation": "s3://kafka-auth-dataplane/confluent-test/output/
checkpoint/"
    }
```

```
)

def processBatch(data_frame, batchId):
    if (data_frame.count() > 0):
        datasource0 = DynamicFrame.fromDF(
          glueContext.add_ingestion_time_columns(data_frame, "hour"),
          glueContext, "from_data_frame"
        )
        additionalOptions_datasink1 = {"enableUpdateCatalog": True}
        additionalOptions_datasink1["partitionKeys"] = ["ingest_yr", "ingest_mo",
 "ingest_day"]
        datasink1 = glueContext.write_dynamic_frame.from_catalog(
          frame = datasource0,
          database = "tempdb",
          table_name = "kafka-auth-table-output",
          transformation_ctx = "datasink1",
          additional_options = additionalOptions_datasink1
        )
```

Bekerja dengan kumpulan data di Amazon S3

- purge_table
- purge_s3_path
- transition_table
- transition_s3_path

purge_table

purge_table(catalog_id=None, database="", table_name="", options={}, transformation_ctx="")

Menghapus file dari Amazon S3 untuk basis data dan tabel katalog yang ditentukan. Jika semua file dalam sebuah partisi dihapus, maka partisi tersebut juga dihapus dari katalog.

Jika Anda ingin dapat memulihkan objek yang dihapus, Anda dapat mengaktifkan versioning objek pada bucket Amazon S3. Ketika sebuah objek dihapus dari sebuah bucket yang versioning objeknya tidak diaktifkan, objek tidak akan dapat dipulihkan. Untuk informasi tentang cara memulihkan objek yang dihapus dalam sebuah bucket dengan versioning yang diaktifkan, lihat Bagaimana saya bisa mengambil objek Amazon S3 yang telah dihapus? di Pusat Pengetahuan AWS Support .

- `catalog_id` — ID katalog Katalog Data yang sedang diakses (ID akun Katalog Data). Diatur ke None secara default. None default ke ID katalog dari akun yang melakukan panggilan dalam layanan.
- `database` — Basis data yang akan digunakan.
- `table_name` — Nama tabel yang akan digunakan.
- `options` — Pilihan untuk mem-filter file yang akan dihapus dan untuk pembuatan file manifes.
  - `retentionPeriod` — Menentukan periode dalam jumlah jam untuk mempertahankan file. File yang lebih baru dari periode penyimpanan akan tetap disimpan. Diatur ke 168 jam (7 hari) secara default.
  - `partitionPredicate` — Partisi yang memenuhi predikat ini akan dihapus. File dalam periode penyimpanan dalam partisi ini tidak dihapus. Diatur ke `""` — kosong secara default.
  - `excludeStorageClasses` — File dengan kelas penyimpanan di set `excludeStorageClasses` tidak dihapus. Default-nya adalah `Set()` — satu set kosong.
  - `manifestFilePath` — Path opsional untuk pembuatan file manifes. Semua file yang berhasil dibersihkan dicatat dalam `Success.csv`, dan yang gagal dicatat dalam `Failed.csv`
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional). Digunakan dalam path file manifes.

Example

```
glueContext.purge_table("database", "table", {"partitionPredicate": "(month=='march')",
 "retentionPeriod": 1, "excludeStorageClasses": ["STANDARD_IA"], "manifestFilePath":
 "s3://bucketmanifest/"})
```

purge_s3_path

`purge_s3_path(s3_path, options={}, transformation_ctx="")`

Menghapus file dari path Amazon S3 yang ditentukan secara rekursif.

Jika Anda ingin dapat memulihkan objek yang dihapus, Anda dapat mengaktifkan versioning objek pada bucket Amazon S3. Ketika sebuah objek dihapus dari sebuah bucket yang versioning objeknya tidak dinyalakan, objek tidak akan dapat dipulihkan. Untuk informasi selengkapnya tentang cara memulihkan objek yang dihapus dalam bucket dengan pembuatan versi, lihat Bagaimana cara mengambil objek Amazon S3 yang telah dihapus? di pusat AWS Support pengetahuan.

- `s3_path` — Path di Amazon S3 dari file yang akan dihapus dalam format `s3://<bucket>/<prefix>/`
- `options` — Pilihan untuk mem-filter file yang akan dihapus dan untuk pembuatan file manifes.
  - `retentionPeriod` — Menentukan periode dalam jumlah jam untuk mempertahankan file. File yang lebih baru dari periode penyimpanan akan tetap disimpan. Diatur ke 168 jam (7 hari) secara default.
  - `excludeStorageClasses` — File dengan kelas penyimpanan di set `excludeStorageClasses` tidak dihapus. Default-nya adalah `Set()` — satu set kosong.
  - `manifestFilePath` — Path opsional untuk pembuatan file manifes. Semua file yang berhasil dibersihkan dicatat dalam `Success.csv`, dan yang gagal dicatat dalam `Failed.csv`
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional). Digunakan dalam path file manifes.

Example

```
glueContext.purge_s3_path("s3://bucket/path/", {"retentionPeriod": 1,
 "excludeStorageClasses": ["STANDARD_IA"], "manifestFilePath": "s3://bucketmanifest/"})
```

transition_table

`transition_table(database, table_name, transition_to, options={}, transformation_ctx="", catalog_id=None)`

Transisi kelas penyimpanan file yang disimpan di Amazon S3 untuk basis data dan tabel katalog yang ditentukan.

Anda dapat melakukan transisi antara dua kelas penyimpanan. Untuk kelas penyimpanan `GLACIER` dan `DEEP_ARCHIVE`, Anda dapat melakukan transisi ke kelas-kelas ini. Namun, Anda harus menggunakan `S3 RESTORE` untuk melakukan transisi dari kelas penyimpanan `GLACIER` dan `DEEP_ARCHIVE`.

Jika Anda menjalankan pekerjaan AWS Glue ETL yang membaca file atau partisi dari Amazon S3, Anda dapat mengecualikan beberapa jenis kelas penyimpanan Amazon S3. Untuk informasi selengkapnya, lihat Mengecualikan Kelas Penyimpanan Amazon S3.

- `database` — Basis data yang akan digunakan.
- `table_name` — Nama tabel yang akan digunakan.

- `transition_to` — Kelas penyimpanan Amazon S3 tempat tujuan transisi.
- `options` — Pilihan untuk mem-filter file yang akan dihapus dan untuk pembuatan file manifes.
  - `retentionPeriod` — Menentukan periode dalam jumlah jam untuk mempertahankan file. File yang lebih baru dari periode penyimpanan akan tetap disimpan. Diatur ke 168 jam (7 hari) secara default.
  - `partitionPredicate` — Partisi yang memenuhi predikat ini akan ditransisi. File dalam periode penyimpanan dalam partisi ini tidak ditransisi. Diatur ke `""` — kosong secara default.
  - `excludeStorageClasses` — File dengan kelas penyimpanan di set `excludeStorageClasses` tidak ditransisi. Default-nya adalah `Set()` — satu set kosong.
  - `manifestFilePath` — Path opsional untuk pembuatan file manifes. Semua file yang berhasil ditransisi dicatat dalam `Success.csv`, dan yang gagal dicatat dalam `Failed.csv`
  - `accountId` — ID akun Amazon Web Services untuk menjalankan transformasi transisi. Wajib untuk transformasi ini.
  - `roleArn`— AWS Peran untuk menjalankan transformasi transisi. Wajib untuk transformasi ini.
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional). Digunakan dalam path file manifes.
- `catalog_id` — ID katalog Katalog Data yang sedang diakses (ID akun Katalog Data). Diatur ke None secara default. None default ke ID katalog dari akun yang melakukan panggilan dalam layanan.

Example

```
glueContext.transition_table("database", "table", "STANDARD_IA", {"retentionPeriod": 1,
 "excludeStorageClasses": ["STANDARD_IA"], "manifestFilePath": "s3://bucketmanifest/",
 "accountId": "12345678901", "roleArn": "arn:aws:iam::123456789012:user/example-
username"})
```

transition_s3_path

`transition_s3_path(s3_path, transition_to, options={}, transformation_ctx="")`

Transisi kelas penyimpanan file di path Amazon S3 yang ditentukan secara rekursif.

Anda dapat melakukan transisi antara dua kelas penyimpanan. Untuk kelas penyimpanan `GLACIER` dan `DEEP_ARCHIVE`, Anda dapat melakukan transisi ke kelas-kelas ini. Namun, Anda harus

menggunakan `S3 RESTORE` untuk melakukan transisi dari kelas penyimpanan `GLACIER` dan `DEEP_ARCHIVE`.

Jika Anda menjalankan pekerjaan AWS Glue ETL yang membaca file atau partisi dari Amazon S3, Anda dapat mengecualikan beberapa jenis kelas penyimpanan Amazon S3. Untuk informasi selengkapnya, lihat Mengecualikan Kelas Penyimpanan Amazon S3.

- `s3_path` — Path di Amazon S3 dari file yang akan ditransisi dalam format `s3://<bucket>/<prefix>/`
- `transition_to` — Kelas penyimpanan Amazon S3 tempat tujuan transisi.
- `options` — Pilihan untuk mem-filter file yang akan dihapus dan untuk pembuatan file manifes.
  - `retentionPeriod` — Menentukan periode dalam jumlah jam untuk mempertahankan file. File yang lebih baru dari periode penyimpanan akan tetap disimpan. Diatur ke 168 jam (7 hari) secara default.
  - `partitionPredicate` — Partisi yang memenuhi predikat ini akan ditransisi. File dalam periode penyimpanan dalam partisi ini tidak ditransisi. Diatur ke `""` — kosong secara default.
  - `excludeStorageClasses` — File dengan kelas penyimpanan di set `excludeStorageClasses` tidak ditransisi. Default-nya adalah `Set()` — satu set kosong.
  - `manifestFilePath` — Path opsional untuk pembuatan file manifes. Semua file yang berhasil ditransisi dicatat dalam `Success.csv`, dan yang gagal dicatat dalam `Failed.csv`
  - `accountId` — ID akun Amazon Web Services untuk menjalankan transformasi transisi. Wajib untuk transformasi ini.
  - `roleArn`— AWS Peran untuk menjalankan transformasi transisi. Wajib untuk transformasi ini.
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional). Digunakan dalam path file manifes.

Example

```
glueContext.transition_s3_path("s3://bucket/prefix/", "STANDARD_IA",
 {"retentionPeriod": 1, "excludeStorageClasses": ["STANDARD_IA"],
 "manifestFilePath": "s3://bucketmanifest/", "accountId": "12345678901", "roleArn":
 "arn:aws:iam::123456789012:user/example-username"})
```

Ekstraksi

- extract_jdbc_conf

extract_jdbc_conf

`extract_jdbc_conf(connection_name, catalog_id = None)`

Mengembalikan `dict` dengan kunci dengan properti konfigurasi dari objek AWS Glue koneksi dalam Katalog Data.

- `user`— Nama pengguna database.
- `password`— Kata sandi basis data.
- `vendor`- Menentukan vendor (`mysql`,`postgresql`,`oracle`,`sqlserver`, dll).
- `enforceSSL`— String boolean yang menunjukkan apakah koneksi aman diperlukan.
- `customJDBCCert`— Gunakan sertifikat klien tertentu dari jalur Amazon S3 yang ditunjukkan.
- `skipCustomJDBCCertValidation`— String boolean yang menunjukkan apakah `customJDBCCert` harus divalidasi oleh CA.
- `customJDBCCertString`— Informasi tambahan tentang sertifikat khusus, khusus untuk jenis driver.
- `url`— URL JDBC (Usang) hanya dengan protokol, server, dan port.
- `fullUrl`— URL JDBC seperti yang dimasukkan saat koneksi dibuat (Tersedia dalam AWS Glue versi 3.0 atau yang lebih baru).

Contoh mengambil konfigurasi JDBC:

```
jdbc_conf = glueContext.extract_jdbc_conf(connection_name="your_glue_connection_name")
print(jdbc_conf)
>>> {'enforceSSL': 'false', 'skipCustomJDBCCertValidation': 'false', 'url':
 'jdbc:mysql://myserver:3306', 'fullUrl': 'jdbc:mysql://myserver:3306/mydb',
 'customJDBCCertString': '', 'user': 'admin', 'customJDBCCert': '', 'password': '1234',
 'vendor': 'mysql'}
```

Transaksi

- start_transaction
- commit_transaction
- pembatalan_transaksi

start_transaction

`start_transaction(read_only)`

Mulai transaksi baru. Secara internal memanggil Lake Formation StartTransaction API.

- `read_only`— (Boolean) Menunjukkan apakah transaksi ini harus dibaca saja atau dibaca dan ditulis. Penulisan yang dibuat menggunakan ID transaksi hanya-baca akan ditolak. Transaksi read-only tidak perlu dilakukan.

Mengembalikan ID transaksi.

commit_transaction

`commit_transaction(transaction_id, wait_for_commit = True)`

Upaya untuk melakukan transaksi yang ditentukan. `commit_transaction`dapat kembali sebelum transaksi selesai dilakukan. Secara internal memanggil Lake Formation CommitTransaction API.

- `transaction_id` — (String) Transaksi untuk melakukan.
- `wait_for_commit`— (Boolean) Menentukan apakah `commit_transaction` pengembalian segera. Nilai default-nya adalah betul. Jika salah, `commit_transaction` polling dan menunggu sampai transaksi dilakukan. Jumlah waktu tunggu dibatasi hingga 1 menit menggunakan backoff eksponensial dengan maksimal 6 upaya coba lagi.

Mengembalikan Boolean untuk menunjukkan apakah komit dilakukan atau tidak.

pembatalan_transaksi

`cancel_transaction(transaction_id)`

Upaya untuk membatalkan transaksi yang ditentukan. Mengembalikan `TransactionCommittedException` pengecualian jika transaksi sebelumnya dilakukan. Secara internal memanggil Lake Formation CancelTransactionAPI.

- `transaction_id`— (String) Transaksi untuk membatalkan.

Menulis

- getSink

- write_dynamic_frame_from_options
- write_from_options
- write_dynamic_frame_from_catalog
- write_data_frame_from_catalog
- write_dynamic_frame_from_jdbc_conf
- write_from_jdbc_conf

getSink

`getSink(connection_type, format = None, transformation_ctx = "", **options)`

Mengambil sebuah objek `DataSink` yang dapat digunakan untuk menulis `DynamicFrames` ke sumber eksternal. Periksa `format` SparkSQL terlebih dahulu untuk memastikan Anda mendapatkan sink yang diharapkan.

- `connection_type` — Jenis koneksi yang akan digunakan, seperti Amazon S3, Amazon Redshift, dan JDBC. Nilai yang valid meliputi `s3mysql`,`postgresql`,,`redshift`,`sqlserver`,`oracle`,`kinesis`, dan`kafka`.
- `format` — Format SparkSQL yang akan digunakan (opsional).
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).
- `options`— Kumpulan pasangan nama-nilai yang digunakan untuk menentukan opsi koneksi. Beberapa nilai yang mungkin adalah:
  - `user`dan`password`: Untuk otorisasi
  - `url`: Titik akhir untuk penyimpanan data
  - `dbtable`: Nama tabel target
  - `bulkSize`: Tingkat paralelisme untuk operasi penyisipan

Opsi yang dapat Anda tentukan tergantung pada jenis koneksi. Lihat Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark untuk nilai dan contoh tambahan.

Contoh:

```
>>> data_sink = context.getSink("s3")
>>> data_sink.setFormat("json"),
```

```
>>> data_sink.writeFrame(myFrame)
```

write_dynamic_frame_from_options

`write_dynamic_frame_from_options(frame, connection_type, connection_options={}, format=None, format_options={}, transformation_ctx = "")`

Menulis dan mengembalikan sebuah `DynamicFrame` menggunakan koneksi dan format yang ditentukan.

- `frame` — `DynamicFrame` yang akan ditulis.
- `connection_type` — Jenis koneksi, seperti Amazon S3, Amazon Redshift, dan JDBC. Nilai yang valid meliputi `s3mysql`,`postgresql`,,`redshift`,`sqlserver`,`oracle`,`kinesis`, dan`kafka`.
- `connection_options` — Pilihan koneksi, seperti path dan tabel basis data (opsional). Untuk `connection_type` dari `s3`, path Amazon S3 didefinisikan.

  ```
  connection_options = {"path": "s3://aws-glue-target/temp"}
  ```

  Untuk koneksi JDBC, beberapa properti harus didefinisikan. Perhatikan bahwa nama basis data harus menjadi bagian dari URL. Secara opsional dapat disertakan dalam opsi koneksi.

  > **Warning**
  >
  > Menyimpan kata sandi dalam skrip Anda tidak disarankan. Pertimbangkan `boto3` untuk menggunakan untuk mengambilnya dari AWS Secrets Manager atau Katalog Data AWS Glue.

  ```
  connection_options = {"url": "jdbc-url/database", "user": "username",
   "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-path"}
  ```

  Properti `dbtable` adalah nama tabel JDBC. Untuk penyimpanan data JDBC yang mendukung skema dalam basis data, tentukan `schema.table-name`. Jika skema tidak disediakan, maka skema "publik" default digunakan.

  Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark.

- `format`— Sebuah spesifikasi format. Ini digunakan untuk Amazon S3 atau AWS Glue koneksi yang mendukung berbagai format. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `format_options` — Pilihan format untuk format yang ditentukan. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `transformation_ctx` — Sebuah konteks transformasi yang akan digunakan (opsional).

write_from_options

`write_from_options(frame_or_dfc, connection_type, connection_options={}, format={}, format_options={}, transformation_ctx = "")`

Menulis dan mengembalikan `DynamicFrame` atau `DynamicFrameCollection` yang dibuat dengan informasi koneksi dan format yang ditentukan.

- `frame_or_dfc` — `DynamicFrame` atau `DynamicFrameCollection` yang akan ditulis.
- `connection_type` — Jenis koneksi, seperti Amazon S3, Amazon Redshift, dan JDBC. Nilai yang valid termasuk `s3`, `mysql`, `postgresql`, `redshift`, `sqlserver`, dan `oracle`.
- `connection_options` — Pilihan koneksi, seperti path dan tabel basis data (opsional). Untuk `connection_type` dari `s3`, path Amazon S3 didefinisikan.

```
connection_options = {"path": "s3://aws-glue-target/temp"}
```

Untuk koneksi JDBC, beberapa properti harus didefinisikan. Perhatikan bahwa nama basis data harus menjadi bagian dari URL. Secara opsional dapat disertakan dalam opsi koneksi.

> **Warning**
>
> Menyimpan kata sandi dalam skrip Anda tidak disarankan. Pertimbangkan `boto3` untuk menggunakan untuk mengambilnya dari AWS Secrets Manager atau Katalog Data AWS Glue.

```
connection_options = {"url": "jdbc-url/database", "user": "username",
 "password": passwordVariable,"dbtable": "table-name", "redshiftTmpDir": "s3-tempdir-
path"}
```

Properti `dbtable` adalah nama tabel JDBC. Untuk penyimpanan data JDBC yang mendukung skema dalam basis data, tentukan `schema.table-name`. Jika skema tidak disediakan, maka skema "publik" default digunakan.

Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark.

- `format`— Sebuah spesifikasi format. Ini digunakan untuk Amazon S3 atau AWS Glue koneksi yang mendukung berbagai format. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `format_options` — Pilihan format untuk format yang ditentukan. Lihat Opsi format data untuk input dan output untuk Spark AWS Glue untuk format yang didukung.
- `transformation_ctx` — Sebuah konteks transformasi yang akan digunakan (opsional).

write_dynamic_frame_from_catalog

`write_dynamic_frame_from_catalog(frame, database, table_name, redshift_tmp_dir, transformation_ctx = "", additional_options = {}, catalog_id = None)`

Menulis dan mengembalikan `DynamicFrame` menggunakan informasi dari basis data dan tabel Katalog Data .

- `frame` — `DynamicFrame` yang akan ditulis.
- `Database` — Basis data Katalog Data yang berisi tabel.
- `table_name` — Nama tabel Katalog Data yang dikaitkan dengan target.
- `redshift_tmp_dir` — Sebuah direktori sementara Amazon Redshift yang akan digunakan (opsional).
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).
- `additional_options` — Kumpulan pasangan nama-nilai opsional.
- `catalog_id` — ID katalog (ID akun) dari Katalog Data yang sedang diakses. Bila Tidak Ada, maka ID akun default pemanggil yang akan digunakan.

write_data_frame_from_catalog

`write_data_frame_from_catalog(frame, database, table_name, redshift_tmp_dir, transformation_ctx = "", additional_options = {}, catalog_id = None)`

Menulis dan mengembalikan `DataFrame` menggunakan informasi dari basis data dan tabel Katalog Data . Metode ini mendukung penulisan ke format danau data (Hudi, Iceberg, dan Delta Lake). Untuk informasi selengkapnya, lihat Menggunakan kerangka kerja data lake dengan pekerjaan AWS Glue ETL.

- `frame` — `DataFrame` yang akan ditulis.
- `Database` — Basis data Katalog Data yang berisi tabel.
- `table_name`— Nama tabel Katalog Data yang dikaitkan dengan target.
- `redshift_tmp_dir` — Sebuah direktori sementara Amazon Redshift yang akan digunakan (opsional).
- `transformation_ctx` — Konteks transformasi yang akan digunakan (opsional).
- `additional_options` — Kumpulan pasangan nama-nilai opsional.
  - `useSparkDataSink`— Ketika diatur ke true, memaksa AWS Glue untuk menggunakan Spark Data Sink API asli untuk menulis ke tabel. Saat Anda mengaktifkan opsi ini, Anda dapat menambahkan opsi Sumber Data Spark apa pun sesuai `additional_options` kebutuhan. AWS Glue meneruskan opsi ini langsung ke penulis Spark.
- `catalog_id`— ID katalog (ID akun) dari Katalog Data yang sedang diakses. Bila Anda tidak menentukan nilai, ID akun default pemanggil akan digunakan.

Batasan

Pertimbangkan batasan berikut saat Anda menggunakan `useSparkDataSink` opsi:

- `enableUpdateCatalog`Opsi ini tidak didukung saat Anda menggunakan `useSparkDataSink` opsi.

Contoh: Menulis ke tabel Hudi menggunakan penulis Spark Data Source

```
hudi_options = {
    'useSparkDataSink': True,
```

```
    'hoodie.table.name': <table_name>,
    'hoodie.datasource.write.storage.type': 'COPY_ON_WRITE',
    'hoodie.datasource.write.recordkey.field': 'product_id',
    'hoodie.datasource.write.table.name': <table_name>,
    'hoodie.datasource.write.operation': 'upsert',
    'hoodie.datasource.write.precombine.field': 'updated_at',
    'hoodie.datasource.write.hive_style_partitioning': 'true',
    'hoodie.upsert.shuffle.parallelism': 2,
    'hoodie.insert.shuffle.parallelism': 2,
    'hoodie.datasource.hive_sync.enable': 'true',
    'hoodie.datasource.hive_sync.database': <database_name>,
    'hoodie.datasource.hive_sync.table': <table_name>,
    'hoodie.datasource.hive_sync.use_jdbc': 'false',
    'hoodie.datasource.hive_sync.mode': 'hms'}

glueContext.write_data_frame.from_catalog(
    frame = <df_product_inserts>,
    database = <database_name>,
    table_name = <table_name>,
    additional_options = hudi_options
)
```

write_dynamic_frame_from_jdbc_conf

`write_dynamic_frame_from_jdbc_conf(frame, catalog_connection, connection_options={}, redshift_tmp_dir = "", transformation_ctx = "", catalog_id = None)`

Menulis dan mengembalikan sebuah `DynamicFrame` menggunakan informasi koneksi JDBC yang ditentukan.

- `frame` — `DynamicFrame` yang akan ditulis.
- `catalog_connection` — Koneksi katalog yang akan digunakan.
- `connection_options` — Pilihan koneksi, seperti path dan tabel basis data (opsional). Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark.
- `redshift_tmp_dir` — Sebuah direktori sementara Amazon Redshift yang akan digunakan (opsional).
- `transformation_ctx` — Sebuah konteks transformasi yang akan digunakan (opsional).
- `catalog_id` — ID katalog (ID akun) dari Katalog Data yang sedang diakses. Bila Tidak Ada, maka ID akun default pemanggil yang akan digunakan.

write_from_jdbc_conf

`write_from_jdbc_conf(frame_or_dfc, catalog_connection, connection_options={}, redshift_tmp_dir = "", transformation_ctx = "", catalog_id = None)`

Menulis dan mengembalikan sebuah `DynamicFrame` atau `DynamicFrameCollection` menggunakan informasi koneksi JDBC yang ditentukan.

- `frame_or_dfc` — `DynamicFrame` atau `DynamicFrameCollection` yang akan ditulis.
- `catalog_connection` — Koneksi katalog yang akan digunakan.
- `connection_options` — Pilihan koneksi, seperti path dan tabel basis data (opsional). Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark.
- `redshift_tmp_dir` — Sebuah direktori sementara Amazon Redshift yang akan digunakan (opsional).
- `transformation_ctx` — Sebuah konteks transformasi yang akan digunakan (opsional).
- `catalog_id` — ID katalog (ID akun) dari Katalog Data yang sedang diakses. Bila Tidak Ada, maka ID akun default pemanggil yang akan digunakan.

AWS Glue PySpark mengubah referensi

AWS Glue menyediakan transformasi bawaan berikut yang dapat Anda gunakan dalam operasi PySpark ETL. Data Anda berpindah dari transformasi ke transformasi dalam struktur data yang disebut a DynamicFrame, yang merupakan ekstensi ke Apache Spark SQL. `DataFrame` `DynamicFrame` tersebut berisi data Anda, dan Anda me-referensi skemanya untuk memproses data Anda.

Sebagian besar transformasi ini juga ada sebagai metode `DynamicFrame` kelas. Untuk informasi lebih lanjut, lihat DynamicFrame transformasi.

- GlueTransform kelas dasar
- ApplyMapping kelas
- DropFields kelas
- DropNullFields kelas
- ErrorsAsDynamicFrame kelas
- EvaluateDataQuality kelas

- FillMissingValues kelas
- Kelas filter
- FindIncrementalMatches kelas
- FindMatches kelas
- FlatMap kelas
- Bergabunglah dengan kelas
- Kelas peta
- MapToCollection kelas
- mergeDynamicFrame
- Relationalisasi kelas
- RenameField kelas
- ResolveChoice kelas
- SelectFields kelas
- SelectFromCollection kelas
- Kelas Simplify_DDB_JSON
- Kelas keran
- SplitFields kelas
- SplitRows kelas
- Kelas buka kotak
- UnnestFrame kelas

GlueTransform kelas dasar

Kelas dasar yang semua kelas `awsglue.transforms` mendapat pewarisan.

Kelas-kelas semua mendefinisikan metode `__call__`. Mereka menimpa metode kelas `GlueTransform` yang tercantum dalam bagian berikut, atau mereka dipanggil menggunakan nama kelas secara default.

Metode

- apply(cls, *args, **kwargs)

- name(cls)
- describeArgs(cls)
- describeReturn(cls)
- describeTransform(cls)
- describeErrors(cls)
- describe(cls)

apply(cls, *args, **kwargs)

Memberlakukan transformasi dengan memanggil kelas transformasi, dan mengembalikan hasilnya.

- `cls` — Objek kelas `self`.

name(cls)

Mengembalikan nama kelas transformasi turunan.

- `cls` — Objek kelas `self`.

describeArgs(cls)

- `cls` — Objek kelas `self`.

Mengembalikan sebuah daftar kamus, masing-masing sesuai dengan argumen bernama, dalam format berikut:

```
[
  {
    "name": "(name of argument)",
    "type": "(type of argument)",
    "description": "(description of argument)",
    "optional": "(Boolean, True if the argument is optional)",
    "defaultValue": "(Default value string, or None)(String; the default value, or
 None)"
  },
...
]
```

Memunculkan sebuah pengecualian `NotImplementedError` ketika dipanggil dalam transformasi turunan di mana ia tidak diimplementasikan.

describeReturn(cls)

- `cls` — Objek kelas `self`.

Mengembalikan sebuah kamus dengan informasi tentang jenis pengembalian, dalam format berikut:

```
{
  "type": "(return type)",
  "description": "(description of output)"
}
```

Memunculkan sebuah pengecualian `NotImplementedError` ketika dipanggil dalam transformasi turunan di mana ia tidak diimplementasikan.

describeTransform(cls)

Mengembalikan sebuah string yang menggambarkan transformasi.

- `cls` — Objek kelas `self`.

Memunculkan sebuah pengecualian `NotImplementedError` ketika dipanggil dalam transformasi turunan di mana ia tidak diimplementasikan.

describeErrors(cls)

- `cls` — Objek kelas `self`.

Mengembalikan sebuah daftar kamus, masing-masing menggambarkan kemungkinan pengecualian yang dilemparkan oleh transformasi ini, dalam format berikut:

```
[
  {
    "type": "(type of error)",
    "description": "(description of error)"
  },
...
```

```
]
```

describe(cls)

- `cls` — Objek kelas `self`.

Mengembalikan sebuah objek dengan format berikut:

```
{
  "transform" : {
    "name" : cls.name( ),
    "args" : cls.describeArgs( ),
    "returns" : cls.describeReturn( ),
    "raises" : cls.describeErrors( ),
    "location" : "internal"
  }
}
```

ApplyMapping kelas

Menerapkan pemetaan dalam sebuah `DynamicFrame`.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.apply_mapping()`metode ini untuk menerapkan pemetaan di file. `DynamicFrame` Untuk melihat contoh kode, lihatContoh: Gunakan apply_mapping untuk mengganti nama bidang dan mengubah jenis bidang.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Jelaskan

__call__(frame, mappings, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Menerapkan pemetaan deklaratif untuk `DynamicFrame` yang ditentukan.

- `frame`— `DynamicFrame` Untuk menerapkan pemetaan ke (wajib).
- `mappings`— Daftar tupel pemetaan (wajib). Masing-masing terdiri dari: (kolom sumber, tipe sumber, kolom target, tipe target).

  Jika kolom sumber memiliki titik "`.`" dalam nama, Anda harus menempatkan back-ticks `` ` `` "di sekitarnya. Misalnya, untuk memetakan `this.old.name` (string) `kethisNewName`, Anda akan menggunakan tupel berikut:

  ```
  ("`this.old.name`", "string", "thisNewName", "string")
  ```

- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info`— String yang dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

Mengembalikan hanya bidang `DynamicFrame` yang ditentukan dalam “pemetaan” tupel.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

DropFields kelas

Membuang bidang dalam sebuah `DynamicFrame`.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.drop_fields()`metode untuk menjatuhkan bidang dari file`DynamicFrame`. Untuk melihat contoh kode, lihatContoh: Gunakan drop_fields untuk menghapus bidang dari `DynamicFrame`.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Jelaskan

__call__(frame, paths, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Membuang simpul dalam sebuah `DynamicFrame`.

- `frame`— `DynamicFrame` Untuk menjatuhkan node di (wajib).
- `paths` — Sebuah daftar path lengkap ke simpul yang akan dibuang (wajib).

- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

Mengembalikan sebuah `DynamicFrame` baru tanpa bidang tertentu yang dibuang.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

DropNullFields kelas

Membuang semua bidang nol dalam `DynamicFrame` yang tipenya adalah `NullType`. Ini adalah bidang dengan nilai yang hilang atau nol di setiap catatan dalam `DynamicFrame` kumpulan data.

Contoh

Contoh ini digunakan `DropNullFields` untuk membuat bidang tipe baru `DynamicFrame` di mana `NullType` telah dijatuhkan. Untuk mendemonstrasikan`DropNullFields`, kami menambahkan kolom baru bernama `empty_column` dengan tipe null ke dataset yang sudah dimuat`persons`.

> **Note**
>
> Untuk mengakses kumpulan data yang digunakan dalam contoh ini, lihat Contoh kode: Bergabung dan menghubungkan data dan ikuti petunjuk diLangkah 1: Merayapi data di bucket Amazon S3.

```
# Example: Use DropNullFields to create a new DynamicFrame without NullType fields

from pyspark.context import SparkContext
from awsglue.context import GlueContext
from pyspark.sql.functions import lit
from pyspark.sql.types import NullType
from awsglue.dynamicframe import DynamicFrame
from awsglue.transforms import DropNullFields

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Create DynamicFrame
persons = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="persons_json"
)
print("Schema for the persons DynamicFrame:")
persons.printSchema()

# Add new column "empty_column" with NullType
persons_with_nulls = persons.toDF().withColumn("empty_column",
 lit(None).cast(NullType()))
persons_with_nulls_dyf = DynamicFrame.fromDF(persons_with_nulls, glueContext,
 "persons_with_nulls")
print("Schema for the persons_with_nulls_dyf DynamicFrame:")
persons_with_nulls_dyf.printSchema()

# Remove the NullType field
```

```
persons_no_nulls = DropNullFields.apply(persons_with_nulls_dyf)
print("Schema for the persons_no_nulls DynamicFrame:")
persons_no_nulls.printSchema()
```

Output

```
Schema for the persons DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string

Schema for the persons_with_nulls_dyf DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
```

```
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
|-- empty_column: null

null_fields ['empty_column']
Schema for the persons_no_nulls DynamicFrame:
root
|-- family_name: string
|-- name: string
|-- links: array
|    |-- element: struct
|    |    |-- note: string
|    |    |-- url: string
|-- gender: string
|-- image: string
|-- identifiers: array
|    |-- element: struct
|    |    |-- scheme: string
|    |    |-- identifier: string
|-- other_names: array
```

```
|    |-- element: struct
|    |    |-- lang: string
|    |    |-- note: string
|    |    |-- name: string
|-- sort_name: string
|-- images: array
|    |-- element: struct
|    |    |-- url: string
|-- given_name: string
|-- birth_date: string
|-- id: string
|-- contact_details: array
|    |-- element: struct
|    |    |-- type: string
|    |    |-- value: string
|-- death_date: string
```

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Jelaskan

__call__(frame, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Membuang semua bidang nol dalam `DynamicFrame` yang tipenya adalah `NullType`. Ini adalah bidang dengan nilai yang hilang atau nol di setiap catatan dalam `DynamicFrame` kumpulan data.

- `frame`— `DynamicFrame` Untuk menjatuhkan bidang null di (wajib).
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi (opsional).

- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

Mengembalikan sebuah `DynamicFrame` baru yang tidak memiliki bidang nol.

apply(cls, *args, **kwargs)

- `cls` — cls

name(cls)

- `cls` — cls

describeArgs(cls)

- `cls` — cls

describeReturn(cls)

- `cls` — cls

describeTransform(cls)

- `cls` — cls

describeErrors(cls)

- `cls` — cls

describe(cls)

- `cls` — cls

ErrorsAsDynamicFrame kelas

Mengembalikan `DynamicFrame` yang berisi catatan bersarang untuk kesalahan yang terjadi saat sumber `DynamicFrame` dibuat.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.errorsAsDynamicFrame()`metode ini untuk mengambil dan melihat catatan kesalahan. Untuk melihat contoh kode, lihatContoh: Gunakan errorsAsDynamic Frame untuk melihat catatan kesalahan.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Jelaskan

__call__(frame)

Mengembalikan `DynamicFrame` yang berisi catatan kesalahan bersarang yang berhubungan dengan sumber`DynamicFrame`.

- `frame` — `DynamicFrame` sumber (wajib).

apply(cls, *args, **kwargs)

- `cls` — cls

name(cls)

- `cls` — cls

describeArgs(cls)

- `cls` — cls

describeReturn(cls)

- `cls` — cls

describeTransform(cls)

- `cls` — cls

describeErrors(cls)

- `cls` — cls

describe(cls)

- `cls` — cls

EvaluateDataQuality kelas

Mengevaluasi kumpulan aturan kualitas data terhadap a `DynamicFrame` dan mengembalikan yang baru `DynamicFrame` dengan hasil evaluasi.

Contoh

Kode contoh berikut menunjukkan bagaimana mengevaluasi kualitas data untuk `DynamicFrame` dan kemudian melihat hasil kualitas data.

```
from awsglue.transforms import *
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsgluedq.transforms import EvaluateDataQuality

#Create Glue context
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
```

```
# Define DynamicFrame
legislatorsAreas = glueContext.create_dynamic_frame.from_catalog(
    database="legislators", table_name="areas_json")

# Create data quality ruleset
ruleset = """Rules = [ColumnExists "id", IsComplete "id"]"""

# Evaluate data quality
dqResults = EvaluateDataQuality.apply(
    frame=legislatorsAreas,
    ruleset=ruleset,
    publishing_options={
        "dataQualityEvaluationContext": "legislatorsAreas",
        "enableDataQualityCloudWatchMetrics": True,
        "enableDataQualityResultsPublishing": True,
        "resultsS3Prefix": "DOC-EXAMPLE-BUCKET1",
    },
)


# Inspect data quality results
dqResults.printSchema()
dqResults.toDF().show()
```

Output

```
root
|-- Rule: string
|-- Outcome: string
|-- FailureReason: string
|-- EvaluatedMetrics: map
|    |-- keyType: string
|    |-- valueType: double


+----------------------+-------+-------------+--------------------------------------+
|Rule                  |Outcome|FailureReason|EvaluatedMetrics                      |
+----------------------+-------+-------------+--------------------------------------+
|ColumnExists "id"     |Passed |null         |{}                                    |
|IsComplete "id"       |Passed |null         |{Column.first_name.Completeness -> 1.0}|
+----------------------+-------+-------------+--------------------------------------+
```

Metode

- __call__
- apply
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (bingkai, kumpulan aturan, publishing_options = {})

- `frame`— `DynamicFrame` Yang Anda inginkan mengevaluasi kualitas data.
- `ruleset`— Aturan Bahasa Definisi Kualitas Data (DQDL) dalam format string. Untuk mempelajari lebih lanjut tentang DQDL, lihat panduannya. Referensi Bahasa Definisi Kualitas Data (DQDL)
- `publishing_options`— Kamus yang menentukan opsi berikut untuk mempublikasikan hasil evaluasi dan metrik:
  - `dataQualityEvaluationContext`— String yang menentukan namespace di mana AWS Glue harus mempublikasikan Amazon CloudWatch metrik dan hasil kualitas data. Metrik agregat muncul di CloudWatch, sementara hasil lengkap muncul di antarmuka AWS Glue Studio.
    - Wajib: Tidak
    - Nilai default: `default_context`
  - `enableDataQualityCloudWatchMetrics`— Menentukan apakah hasil evaluasi kualitas data harus dipublikasikan ke CloudWatch. Anda menentukan namespace untuk metrik menggunakan opsi. `dataQualityEvaluationContext`
    - Wajib: Tidak
    - Nilai default: Salah
  - `enableDataQualityResultsPublishing`— Menentukan apakah hasil kualitas data harus terlihat pada tab Kualitas Data di antarmuka AWS Glue Studio.
    - Wajib: Tidak
    - Nilai default: Benar

- `resultsS3Prefix`— Menentukan lokasi Amazon S3 di mana AWS Glue dapat menulis hasil evaluasi kualitas data.
  - Wajib: Tidak
  - Nilai default: “” (string kosong)

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

FillMissingValues kelas

Kelas `FillMissingValues` menempatkan nilai nol dan string kosong dalam sebuah `DynamicFrame` yang ditentukan dan menggunakan metode machine learning, seperti regresi linier dan hutan acak, untuk memprediksi nilai yang hilang. Tugas ETL menggunakan nilai-nilai dalam set data input untuk melatih model machine learning, yang kemudian memprediksi apa nilai-nilai yang hilang tersebut seharusnya.

> **Tip**
>
> Jika Anda menggunakan kumpulan data tambahan, maka setiap set tambahan digunakan sebagai data pelatihan untuk model machine learning, sehingga hasilnya mungkin tidak akurat.

Untuk mengimpor:

```
from awsglueml.transforms import FillMissingValues
```

Metode

- Terapkan

apply(frame, missing_values_column, output_column ="", transformation_ctx ="", info ="", stageThreshold = 0, totalThreshold = 0)

Mengisi nilai-nilai yang hilang dari bingkai dinamis dalam kolom yang ditentukan dan mengembalikan bingkai baru dengan perkiraan dalam sebuah kolom baru. Untuk baris tanpa nilai yang hilang, nilai kolom yang ditentukan diduplikasi ke kolom baru tersebut.

- `frame` — `DynamicFrame` tempat untuk mengisi nilai yang hilang. Wajib.
- `missing_values_column` — Kolom yang berisi nilai-nilai yang hilang (nilai `null` dan string kosong). Wajib.
- `output_column` — Nama kolom baru yang akan berisi perkiraan nilai untuk semua baris yang nilainya hilang. Opsional; default-nya adalah nama `missing_values_column` dengan sufiks `"_filled"`.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold` — Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional; default-nya adalah nol).
- `totalThreshold` — Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum kesalahan keluar (opsional; default-nya adalah nol).

Mengembalikan sebuah `DynamicFrame` baru dengan satu kolom tambahan yang berisi perkiraan untuk baris dengan nilai-nilai yang hilang dan nilai sekarang untuk baris lainnya.

Kelas filter

Membangun baru `DynamicFrame` yang berisi catatan dari input `DynamicFrame` yang memenuhi fungsi predikat tertentu.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.filter()`metode ini untuk memfilter catatan dalam file`DynamicFrame`. Untuk melihat contoh kode, lihatContoh: Gunakan filter untuk mendapatkan pilihan bidang yang difilter.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, f, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0))

Mengembalikan baru `DynamicFrame` yang dibangun dengan memilih catatan dari input `DynamicFrame` yang memenuhi fungsi predikat tertentu.

- `frame` — `DynamicFrame` sumber tempat akan diterapkannya fungsi filter yang ditentukan (wajib).
- `f` — Fungsi predikat yang akan diterapkan pada masing-masing `DynamicRecord` di `DynamicFrame`. Fungsi harus mengambil `DynamicRecord` sebagai argumennya dan mengembalikan True jika `DynamicRecord` memenuhi persyaratan filter, atau False jika tidak (wajib).

Sebuah `DynamicRecord` mewakili catatan logis dalam sebuah `DynamicFrame`. Ini mirip dengan baris di Spark`DataFrame`, kecuali bahwa itu menggambarkan diri sendiri dan dapat digunakan untuk data yang tidak sesuai dengan skema tetap.

- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info`— String yang dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

FindIncrementalMatches kelas

Mengidentifikasi pencocokan catatan di `DynamicFrame` yang ada dan tambahan dan menciptakan sebuah `DynamicFrame` baru dengan pengidentifikasi unik yang ditetapkan untuk setiap grup pencocokan catatan.

Untuk mengimpor:

```
from awsglueml.transforms import FindIncrementalMatches
```

Metode

- Terapkan

berlaku (ExistingFrame, IncrementalFrame, transformId, transformation_ctx = “”, info = “”, stageThreshold = 0, totalThreshold = 0, enforcedMatches = none, Skor = 0) computeMatchConfidence

Mengidentifikasi pencocokan catatan di `DynamicFrame` input dan menciptakan sebuah `DynamicFrame` baru dengan pengidentifikasi unik yang ditetapkan untuk setiap grup pencocokan catatan.

- `existingFrame`— Yang ada dan pra-cocok `DynamicFrame` untuk menerapkan transformasi. FindIncrementalMatches Diperlukan.
- `incrementalFrame`— Inkremental `DynamicFrame` untuk menerapkan FindIncrementalMatches transformasi agar sesuai dengan. `existingFrame` Diperlukan.
- `transformId`— ID unik yang terkait dengan FindIncrementalMatches transformasi untuk diterapkan pada catatan di`DynamicFrames`. Diperlukan.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi stats/status. Tidak wajib.
- `info` — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi. Tidak wajib.
- `stageThreshold` — Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar. Tidak wajib. Default-nya adalah nol.
- `totalThreshold` — Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum pemrosesan kesalahan keluar. Tidak wajib. Default-nya adalah nol.
- `enforcedMatches` — `DynamicFrame` digunakan untuk menegakkan pencocokan. Tidak wajib. Default-nya adalah Tidak Ada.

- `computeMatchConfidenceScores`— Nilai Boolean yang menunjukkan apakah akan menghitung skor kepercayaan untuk setiap kelompok catatan yang cocok. Tidak wajib. Default-nya adalah salah.

Mengembalikan sebuah `DynamicFrame` baru dengan pengidentifikasi unik yang ditetapkan untuk setiap grup catatan yang cocok.

FindMatches kelas

Mengidentifikasi pencocokan catatan di `DynamicFrame` input dan menciptakan sebuah `DynamicFrame` baru dengan pengidentifikasi unik yang ditetapkan untuk setiap grup pencocokan catatan.

Untuk mengimpor:

```
from awsglueml.transforms import FindMatches
```

Metode

- Terapkan

berlaku (bingkai, transformId, transformation_ctx = “”, info = “”, stageThreshold = 0, totalThreshold = 0, enforcedMatches = none, Skor = 0) computeMatchConfidence

Mengidentifikasi pencocokan catatan di `DynamicFrame` input dan menciptakan sebuah `DynamicFrame` baru dengan pengidentifikasi unik yang ditetapkan untuk setiap grup pencocokan catatan.

- `frame`— `DynamicFrame` Untuk menerapkan FindMatches transformasi. Diperlukan.
- `transformId`— ID unik yang terkait dengan FindMatches transformasi untuk diterapkan pada catatan di`DynamicFrame`. Diperlukan.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi stats/status. Tidak wajib.
- `info` — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi. Tidak wajib.
- `stageThreshold` — Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar. Tidak wajib. Default-nya adalah nol.
- `totalThreshold` — Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum pemrosesan kesalahan keluar. Tidak wajib. Default-nya adalah nol.

- `enforcedMatches` — `DynamicFrame` digunakan untuk menegakkan pencocokan. Tidak wajib. Default-nya adalah Tidak Ada.
- `computeMatchConfidenceScores`— Nilai Boolean yang menunjukkan apakah akan menghitung skor kepercayaan untuk setiap kelompok catatan yang cocok. Tidak wajib. Default-nya adalah salah.

Mengembalikan sebuah `DynamicFrame` baru dengan pengidentifikasi unik yang ditetapkan untuk setiap grup catatan yang cocok.

FlatMap kelas

Menerapkan transformasi ke masing-masing `DynamicFrame` dalam koleksi. Hasil tidak diratakan menjadi satu`DynamicFrame`, tetapi dipertahankan sebagai koleksi.

Contoh untuk FlatMap

Contoh cuplikan berikut menunjukkan bagaimana menggunakan `ResolveChoice` transformasi pada koleksi frame dinamis ketika diterapkan ke file. `FlatMap` Data yang digunakan untuk input ada di JSON yang terletak di alamat Amazon S3 placeholder `s3://bucket/path-for-data/sample.json` dan berisi data berikut.

Contoh data JSON

```
[{
    "firstname": "Arnav",
    "lastname": "Desai",
    "address": {
        "street": "6 Anyroad Avenue",
        "city": "London",
        "state": "England",
        "country": "UK"
    },
    "phone": 17235550101,
    "affiliations": [
        "General Anonymous Example Products",
        "Example Independent Research",
        "Government Department of Examples"
    ]
},
{
    "firstname": "Mary",
```

```
    "lastname": "Major",
    "address": {
        "street": "7821 Spot Place",
        "city": "Centerville",
        "state": "OK",
        "country": "US"
    },
    "phone": 19185550023,
    "affiliations": [
        "Example Dot Com",
        "Example Independent Research",
        "Example.io"
    ]
},
{
    "firstname": "Paulo",
    "lastname": "Santos",
    "address": {
        "street": "123 Maple Street",
        "city": "London",
        "state": "Ontario",
        "country": "CA"
    },
    "phone": 12175550181,
    "affiliations": [
        "General Anonymous Example Products",
        "Example Dot Com"
    ]
}]
```

Example Terapkan ResolveChoice ke DynamicFrameCollection dan tampilkan output.

```
#Read DynamicFrame
datasource = glueContext.create_dynamic_frame_from_options("s3", connection_options =
 {"paths":["s3://bucket/path/to/file/mysamplejson.json"]}, format="json")
datasource.printSchema()
datasource.show()

## Split to create a DynamicFrameCollection
split_frame=datasource.split_fields(["firstname","lastname","address"],"personal_info","busines
split_frame.keys()
print("---")
```

```
## Use FlatMap to run ResolveChoice
kwargs = {"choice": "cast:string"}
flat = FlatMap.apply(split_frame, ResolveChoice, frame_name="frame",
 transformation_ctx='tcx', **kwargs)
flat.keys()

##Select one of the DynamicFrames
personal_info = flat.select("personal_info")
personal_info.printSchema()
personal_info.show()
print("---")

business_info = flat.select("business_info")
business_info.printSchema()
business_info.show()
```

> **Important**
>
> Saat meneleponFlatMap.apply, frame_name parameternya harus"frame". Tidak ada nilai lain yang diterima saat ini.

Contoh keluaran

```
root
|-- firstname: string
|-- lastname: string
|-- address: struct
|    |-- street: string
|    |-- city: string
|    |-- state: string
|    |-- country: string
|-- phone: long
|-- affiliations: array
|    |-- element: string
---
{
    "firstname": "Mary",
    "lastname": "Major",
    "address": {
        "street": "7821 Spot Place",
        "city": "Centerville",
```

```
        "state": "OK",
        "country": "US"
    },
    "phone": 19185550023,
    "affiliations": [
        "Example Dot Com",
        "Example Independent Research",
        "Example.io"
    ]
}

{
    "firstname": "Paulo",
    "lastname": "Santos",
    "address": {
        "street": "123 Maple Street",
        "city": "London",
        "state": "Ontario",
        "country": "CA"
    },
    "phone": 12175550181,
    "affiliations": [
        "General Anonymous Example Products",
        "Example Dot Com"
    ]
}
---
root
|-- firstname: string
|-- lastname: string
|-- address: struct
|    |-- street: string
|    |-- city: string
|    |-- state: string
|    |-- country: string

{
    "firstname": "Mary",
    "lastname": "Major",
    "address": {
        "street": "7821 Spot Place",
        "city": "Centerville",
        "state": "OK",
        "country": "US"
```

```
    }
}

{
    "firstname": "Paulo",
    "lastname": "Santos",
    "address": {
        "street": "123 Maple Street",
        "city": "London",
        "state": "Ontario",
        "country": "CA"
    }
}
---
root
|-- phone: long
|-- affiliations: array
|    |-- element: string

{
    "phone": 19185550023,
    "affiliations": [
        "Example Dot Com",
        "Example Independent Research",
        "Example.io"
    ]
}

{
    "phone": 12175550181,
    "affiliations": [
        "General Anonymous Example Products",
        "Example Dot Com"
    ]
}
```

Metode

- __call__
- Terapkan
- Nama
- describeArgs

- describeReturn
- describeTransform
- describeErrors
- Jelaskan

__call__ (dfc,, frame_name BaseTransform, transformation_ctx = “”, **base_kwargs)

Menerapkan sebuah transformasi untuk setiap `DynamicFrame` dalam sebuah koleksi dan meratakan hasilnya.

- `dfc` — `DynamicFrameCollection` tempat untuk melakukan flatmap (wajib).
- `BaseTransform` — Sebuah transformasi yang berasal dari `GlueTransform` untuk diterapkan ke setiap anggota koleksi (wajib).
- `frame_name` — Nama argumen yang padanya elemen koleksi akan diberikan (wajib).
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `base_kwargs` — Argumen yang akan diberikan ke transformasi dasar (wajib).

Mengembalikan sebuah `DynamicFrameCollection` baru yang dibuat dengan menerapkan transformasi untuk setiap `DynamicFrame` dalam `DynamicFrameCollection` sumber.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

Bergabunglah dengan kelas

Melakukan penggabungan setara pada dua `DynamicFrames`.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.join()`metode untuk bergabung`DynamicFrames`. Untuk melihat contoh kode, lihatContoh: Gunakan bergabung untuk menggabungkan `DynamicFrames`.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Jelaskan

__call__(frame1, frame2, keys1, keys2, transformation_ctx = "")

Melakukan penggabungan setara pada dua `DynamicFrames`.

- `frame1` — `DynamicFrame` yang pertama yang akan bergabung (wajib).
- `frame2` — `DynamicFrame` yang kedua yang akan bergabung (wajib).
- `keys1` — Kunci untuk bergabung pada bingkai pertama (wajib).
- `keys2` — Kunci untuk bergabung pada bingkai kedua (wajib).

- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).

Mengembalikan baru `DynamicFrame` yang dibuat dengan bergabung dengan keduanya`DynamicFrames`.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

Kelas peta

Membangun sebuah `DynamicFrame` baru dengan menerapkan fungsi untuk semua catatan di input `DynamicFrame`.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.map()`metode ini untuk menerapkan fungsi ke semua catatan dalam file`DynamicFrame`. Untuk melihat contoh kode, lihatContoh: Gunakan peta untuk menerapkan fungsi ke setiap catatan dalam `DynamicFrame`.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- Jelaskan

__call__(frame, f, transformation_ctx="", info="", stageThreshold=0, totalThreshold=0)

Mengembalikan `DynamicFrame` baru dihasilkan dari penerapan fungsi yang ditentukan untuk semua `DynamicRecords` dalam `DynamicFrame` aslinya.

- `frame`— Asli `DynamicFrame` untuk menerapkan fungsi pemetaan ke (wajib).
- `f` — Fungsi yang akan diterapkan ke semua `DynamicRecords` di `DynamicFrame`. Fungsi harus mengambil `DynamicRecord` sebagai argumen dan mengembalikan yang baru `DynamicRecord` yang dihasilkan oleh pemetaan (wajib).

  Sebuah `DynamicRecord` mewakili catatan logis dalam sebuah `DynamicFrame`. Ini mirip dengan baris di Apache Spark`DataFrame`, kecuali bahwa itu menggambarkan diri sendiri dan dapat digunakan untuk data yang tidak sesuai dengan skema tetap.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

Mengembalikan `DynamicFrame` baru dihasilkan dari penerapan fungsi yang ditentukan untuk semua `DynamicRecords` dalam `DynamicFrame` aslinya.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

MapToCollection kelas

Menerapkan sebuah transformasi untuk setiap `DynamicFrame` dalam `DynamicFrameCollection` yang ditentukan.

Metode

- __call__
- Terapkan
- Nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors

- Jelaskan

__call__ (dfc,, frame_name BaseTransform, transformation_ctx = “”, **base_kwargs)

Menerapkan sebuah fungsi transformasi untuk setiap `DynamicFrame` dalam `DynamicFrameCollection` yang ditentukan.

- `dfc` — `DynamicFrameCollection` yang akan digunakan untuk menerapkan fungsi transformasi (wajib).
- `callable` — Sebuah fungsi transformasi yang dapat dipanggil untuk diterapkan ke setiap anggota koleksi (wajib).
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).

Mengembalikan sebuah `DynamicFrameCollection` baru yang dibuat dengan menerapkan transformasi untuk setiap `DynamicFrame` dalam `DynamicFrameCollection` sumber.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

Relationalisasi kelas

Meratakan skema bersarang di a `DynamicFrame` dan berputar keluar kolom array dari bingkai yang diratakan.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.relationalize()`metode ini untuk menghubungkan a`DynamicFrame`. Untuk melihat contoh kode, lihatContoh: Gunakan relationalize untuk meratakan skema bersarang di `DynamicFrame`.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, staging_path=None, name='roottable', options=None, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Menghubungkan sebuah `DynamicFrame` dengan membuat sebuah daftar bingkai yang dihasilkan oleh membuka nest dari kolom yang di-nest dan kolom array yang berputar. Anda dapat menggabungkan kolom array berputar ke tabel root dengan menggunakan kunci gabungan yang dihasilkan dalam fase unnest.

- `frame` — `DynamicFrame` yang direlasionalisasi (wajib).
- `staging_path`— Jalur di mana metode dapat menyimpan partisi tabel berputar dalam format CSV (opsional). Tabel berputar dibaca kembali dari path ini.
- `name` — Nama tabel akar (opsional).

- `options` — Kamus parameter opsional. Saat ini tidak digunakan.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

RenameField kelas

Mengganti nama sebuah simpul dalam `DynamicFrame`.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.rename_field()`metode ini untuk mengganti nama bidang di. `DynamicFrame` Untuk melihat contoh kode, lihatContoh: Gunakan rename_field untuk mengganti nama bidang dalam `DynamicFrame`.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, old_name, new_name, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Mengganti nama sebuah simpul dalam `DynamicFrame`.

- `frame` — `DynamicFrame` di mana nama simpul akan diubah (wajib).
- `old_name`— Jalur lengkap ke node untuk mengganti nama (diperlukan).

  Jika nama lama memiliki titik di dalamnya, tidak RenameField akan berfungsi kecuali Anda menempatkan backticks di sekitarnya ()``. Misalnya, untuk mengganti `this.old.name` dengan`thisNewName`, Anda akan memanggil RenameField sebagai berikut:

  ```
  newDyF = RenameField(oldDyF, "`this.old.name`", "thisNewName")
  ```

- `new_name`— Nama baru, termasuk jalur lengkap (wajib).
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.

- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

ResolveChoice kelas

Mengubah jenis pilihan dalam sebuah `DynamicFrame`.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.resolveChoice()`metode ini untuk menangani bidang yang berisi beberapa tipe dalam file`DynamicFrame`. Untuk melihat contoh kode, lihatContoh: Gunakan ResolveChoice untuk menangani kolom yang berisi beberapa jenis.

Metode

- __call__

- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (bingkai, spesifikasi = tidak ada, pilihan = “”, transformation_ctx = “”, info = “”, stageThreshold = 0, totalThreshold = 0)

Menyediakan informasi untuk mengubah jenis yang ambigu dalam `DynamicFrame`. Ini mengembalikan hasil`DynamicFrame`.

- `frame` — `DynamicFrame` di mana jenis pilihan akan diubah (wajib).
- `specs` — Daftar ambiguitas spesifik yang akan diubah, masing-masing dalam bentuk tupel:`(path, action)`. Nilai `path` mengidentifikasi elemen ambigu tertentu, dan nilai `action` mengidentifikasi resolusi yang sesuai.

  Anda hanya dapat menggunakan salah satu `choice` parameter `spec` dan. Jika parameter `spec` bukan None, maka parameter `choice` harus string kosong. Sebaliknya, jika `choice` bukan string kosong, maka `spec` parameternya harusNone. Jika parameter tidak disediakan, Glue AWS akan mencoba mengurai skema dan menggunakannya untuk mengubah ambiguitas.

  Anda dapat menentukan salah satu strategi resolusi berikut di `action` bagian `specs` Tuple:

  - `cast`— Memungkinkan Anda menentukan jenis yang akan dilemparkan (misalnya,`cast:int`).
  - `make_cols`— Menyelesaikan ambiguitas potensial dengan meratakan data. Misalnya, jika `columnA` bisa berupa `int` atau `string`, maka resolusi akan menghasilkan dua kolom bernama `columnA_int` dan `columnA_string` dalam `DynamicFrame` yang dihasilkan.
  - `make_struct`— Menyelesaikan ambiguitas potensial dengan menggunakan struct untuk mewakili data. Sebagai contoh, jika data dalam kolom bisa menjadi `int` atau `string`, menggunakan tindakan `make_struct` menghasilkan sebuah kolom struktur dalam `DynamicFrame` yang dihasilkan yang masing-masing berisi sebuah `int` dan sebuah `string`.
  - `project`— Menyelesaikan ambiguitas potensial dengan hanya mempertahankan nilai dari tipe tertentu dalam hasil. `DynamicFrame` Misalnya, jika data dalam `ChoiceType` kolom bisa berupa

`int` atau `astring`, menentukan `project:string` tindakan akan menurunkan nilai dari hasil `DynamicFrame` yang bukan tipe`string`.

Jika `path` mengidentifikasi sebuah array, menempatkan kurung persegi kosong setelah nama array untuk menghindari ambiguitas. Misalnya, anggap Anda bekerja dengan data yang terstruktur sebagai berikut:

```
"myList": [
  { "price": 100.00 },
  { "price": "$100.00" }
]
```

Anda dapat memilih numerik daripada versi string harga dengan menyetel `path` ke`"myList[].price"`, dan menyetel `action` ke`"cast:double"`.

- `choice` — Tindakan resolusi default jika parameter `specs` adalah `None`. Jika parameter `specs` bukan `None`, maka ini tidak harus diatur ke apa pun kecuali string kosong.

  Selain `specs` tindakan yang dijelaskan sebelumnya, argumen ini juga mendukung tindakan berikut:

  - `MATCH_CATALOG` — Upaya untuk mengubah setiap `ChoiceType` menjadi jenis yang sesuai dalam tabel Katalog Data yang ditentukan.
- `database`— Database AWS Glue Data Catalog untuk digunakan dengan `MATCH_CATALOG` pilihan (diperlukan untuk`MATCH_CATALOG`).
- `table_name`— Nama tabel AWS Glue Data Catalog untuk digunakan dengan `MATCH_CATALOG` tindakan (diperlukan untuk`MATCH_CATALOG`).
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

SelectFields kelas

`SelectFields`Kelas membuat yang baru `DynamicFrame` dari yang sudah ada`DynamicFrame`, dan hanya menyimpan bidang yang Anda tentukan. `SelectFields`menyediakan fungsionalitas yang mirip dengan `SELECT` pernyataan SQL.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.select_fields()`metode untuk memilih bidang dari`DynamicFrame`. Untuk melihat contoh kode, lihatContoh: Gunakan select_fields untuk membuat yang baru `DynamicFrame` dengan bidang yang dipilih.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn

- describeTransform
- describeErrors
- Jelaskan

__call__(frame, paths, transformation_ctx = "", info = "", stageThreshold = 0, totalThreshold = 0)

Mengambil bidang (simpul) dalam sebuah `DynamicFrame`.

- `frame`— `DynamicFrame` Untuk memilih bidang di (wajib).
- `paths` — Daftar path lengkap ke bidang yang akan dipilih (wajib).
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info`— String yang dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

Mengembalikan baru `DynamicFrame` yang hanya berisi bidang tertentu.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

SelectFromCollection kelas

Memilih satu `DynamicFrame` dalam sebuah `DynamicFrameCollection`.

Contoh

Contoh ini digunakan `SelectFromCollection` untuk memilih `DynamicFrame` dari a`DynamicFrameCollection`.

Contoh dataset

Contoh memilih dua `DynamicFrames` dari yang `DynamicFrameCollection` dipanggil`split_rows_collection`. Berikut ini adalah daftar kunci di`split_rows_collection`.

```
dict_keys(['high', 'low'])
```

Contoh kode

```
# Example: Use SelectFromCollection to select
# DynamicFrames from a DynamicFrameCollection

from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.transforms import SelectFromCollection

# Create GlueContext
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

# Select frames and inspect entries
frame_low = SelectFromCollection.apply(dfc=split_rows_collection, key="low")
frame_low.toDF().show()

frame_high = SelectFromCollection.apply(dfc=split_rows_collection, key="high")
frame_high.toDF().show()
```

Output

```
+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
|  1|    0|                    fax|           202-225-3307|
|  1|    1|                  phone|           202-225-5731|
|  2|    0|                    fax|           202-225-3307|
|  2|    1|                  phone|           202-225-5731|
|  3|    0|                    fax|           202-225-3307|
|  3|    1|                  phone|           202-225-5731|
|  4|    0|                    fax|           202-225-3307|
|  4|    1|                  phone|           202-225-5731|
|  5|    0|                    fax|           202-225-3307|
|  5|    1|                  phone|           202-225-5731|
|  6|    0|                    fax|           202-225-3307|
|  6|    1|                  phone|           202-225-5731|
|  7|    0|                    fax|           202-225-3307|
|  7|    1|                  phone|           202-225-5731|
|  8|    0|                    fax|           202-225-3307|
|  8|    1|                  phone|           202-225-5731|
|  9|    0|                    fax|           202-225-3307|
|  9|    1|                  phone|           202-225-5731|
| 10|    0|                    fax|           202-225-6328|
| 10|    1|                  phone|           202-225-4576|
+---+-----+-----------------------+-----------------------+
only showing top 20 rows

+---+-----+-----------------------+-----------------------+
| id|index|contact_details.val.type|contact_details.val.value|
+---+-----+-----------------------+-----------------------+
| 11|    0|                    fax|           202-225-6328|
| 11|    1|                  phone|           202-225-4576|
| 11|    2|                twitter|          RepTrentFranks|
| 12|    0|                    fax|           202-225-6328|
| 12|    1|                  phone|           202-225-4576|
| 12|    2|                twitter|          RepTrentFranks|
| 13|    0|                    fax|           202-225-6328|
| 13|    1|                  phone|           202-225-4576|
| 13|    2|                twitter|          RepTrentFranks|
| 14|    0|                    fax|           202-225-6328|
| 14|    1|                  phone|           202-225-4576|
| 14|    2|                twitter|          RepTrentFranks|
| 15|    0|                    fax|           202-225-6328|
```

```
| 15|    1|                   phone|             202-225-4576|
| 15|    2|                 twitter|           RepTrentFranks|
| 16|    0|                     fax|             202-225-6328|
| 16|    1|                   phone|             202-225-4576|
| 16|    2|                 twitter|           RepTrentFranks|
| 17|    0|                     fax|             202-225-6328|
| 17|    1|                   phone|             202-225-4576|
+---+-----+------------------------+-------------------------+
only showing top 20 rows
```

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(dfc, key, transformation_ctx = "")

Mengambil satu `DynamicFrame` dari sebuah `DynamicFrameCollection`.

- `dfc`— `DynamicFrameCollection` Yang `DynamicFrame` harus dipilih dari (wajib).
- `key` — Kunci dari `DynamicFrame` yang akan dipilih (wajib).
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

Kelas Simplify_DDB_JSON

Menyederhanakan kolom bersarang dalam `DynamicFrame` yang secara khusus dalam struktur DynamoDB JSON, dan mengembalikan disederhanakan baru. `DynamicFrame`

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.simplify_ddb_json()` metode ini untuk menyederhanakan kolom bersarang dalam `DynamicFrame` yang secara khusus dalam struktur DynamoDB JSON. Untuk melihat contoh kode, lihatContoh: Gunakan simplify_ddb_json untuk memanggil DynamoDB JSON menyederhanakan.

Kelas keran

Menulis catatan sampel ke tujuan tertentu untuk membantu Anda memverifikasi transformasi yang dilakukan oleh AWS Glue pekerjaan Anda.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.spigot()`metode ini untuk menulis subset catatan dari tujuan `DynamicFrame` ke tujuan tertentu. Untuk melihat contoh kode, lihatContoh: Gunakan keran untuk menulis bidang contoh dari a `DynamicFrame` ke Amazon S3.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, path, options, transformation_ctx = "")

Menulis catatan sampel ke sebuah tujuan tertentu selama transformasi.

- `frame` — `DynamicFrame` ke spigot (wajib).
- `path`— Jalur tujuan untuk menulis (wajib).
- `options`— Pasangan nilai kunci JSON yang menentukan opsi (opsional). `"topk"`Opsi menentukan bahwa catatan k pertama harus ditulis. `"prob"`Opsi menentukan probabilitas (sebagai desimal) untuk memilih catatan apa pun yang diberikan. Anda menggunakan ini dalam memilih catatan untuk menulis.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply

name(cls)

Warisan dari `GlueTransform` nama

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors

describe(cls)

Warisan dari `GlueTransform` describe

SplitFields kelas

Membagi `DynamicFrame` menjadi dua yang baru, dengan bidang yang ditentukan.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.split_fields()`metode ini untuk membagi bidang dalam file`DynamicFrame`. Untuk melihat contoh kode, lihatContoh: Gunakan split_fields untuk membagi bidang yang dipilih menjadi terpisah `DynamicFrame`.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (bingkai, jalur, name1 = none, name2 = none, transformation_ctx = “”, info = “”, stageThreshold = 0, totalThreshold = 0)

Membagi satu atau lebih bidang dalam `DynamicFrame` off menjadi yang baru`DynamicFrame`, dan membuat bidang baru lainnya `DynamicFrame` yang berisi bidang yang tersisa.

- `frame` — `DynamicFrame` sumber untuk dibagi menjadi dua yang baru (wajib).
- `paths` — Daftar path lengkap ke bidang yang akan dibagi (wajib).
- `name1` — Nama yang akan ditetapkan ke `DynamicFrame` yang akan berisi bidang yang akan dipisahkan (opsional). Jika tidak ada nama yang disediakan, maka nama bingkai sumber digunakan dengan ditambahkan "1".
- `name2` — Nama yang akan ditetapkan ke `DynamicFrame` yang akan berisi bidang yang tersisa setelah bidang yang ditentukan dipisahkan (opsional). Jika tidak ada nama yang disediakan, maka nama bingkai sumber digunakan dengan ditambahkan "2".
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

SplitRows kelas

Menciptakan `DynamicFrameCollection` yang berisi dua`DynamicFrames`. Satu hanya `DynamicFrame` berisi baris yang ditentukan untuk dibagi, dan yang lainnya berisi semua baris yang tersisa.

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.split_rows()`metode ini untuk membagi baris dalam file`DynamicFrame`. Untuk melihat contoh kode, lihatContoh: Gunakan split_rows untuk membagi baris dalam `DynamicFrame`.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (bingkai, comparison_dict, name1="frame1", name2="frame2", transformation_ctx = “”, info = tidak ada, stageThreshold = 0, totalThreshold = 0)

Membagi satu atau beberapa baris dalam `DynamicFrame` ke sebuah `DynamicFrame` yang baru.

- `frame` — `DynamicFrame` sumber untuk dibagi menjadi dua yang baru (wajib).
- `comparison_dict`— Kamus di mana kuncinya adalah jalur lengkap ke kolom, dan nilainya adalah kamus lain untuk memetakan komparator ke nilai yang dibandingkan dengan nilai kolom. Misalnya, `{"age": {">": 10, "<": 20}}` membagi baris di mana nilai "usia" adalah antara 10 dan 20, eksklusif, dari baris di mana "usia" berada di luar kisaran itu (wajib).
- `name1` — Nama yang akan ditetapkan ke `DynamicFrame` yang akan berisi baris yang akan dipisahkan (opsional).
- `name2` — Nama yang akan ditetapkan ke `DynamicFrame` yang akan berisi baris yang tersisa setelah baris yang ditentukan dipisahkan (opsional).
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `info` — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

Kelas buka kotak

Membuka kotak (memformat ulang) bidang string di file. `DynamicFrame`

Contoh

Kami menyarankan Anda menggunakan `DynamicFrame.unbox()`metode ini untuk membuka kotak bidang di. `DynamicFrame` Untuk melihat contoh kode, lihatContoh: Gunakan unbox untuk membuka kotak bidang string ke dalam struct.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, path, format, transformation_ctx = "", info="", stageThreshold=0, totalThreshold=0, **options)

Membuka kotak sebuah bidang string di sebuah `DynamicFrame`.

- `frame` — `DynamicFrame` tempat sebuah bidang akan dibuka. (wajib).
- `path` — Path lengkap ke `StringNode` tempat untuk membuka kotak (wajib).
- `format` — Format spesifikasi (opsional). Ini digunakan untuk Amazon S3 atau AWS Glue koneksi yang mendukung berbagai format. Untuk format yang didukung, lihatOpsi format data untuk input dan output untuk Spark AWS Glue.
- `transformation_ctx` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).

- `info` — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi (opsional).
- `stageThreshold`— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- `totalThreshold`— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.
- `separator` — Sebuah token pemisah (opsional).
- `escaper` — Sebuah token escape (opsional).
- `skipFirst` — `True` jika baris pertama data harus dilewati, atau `False` jika tidak boleh dilewati (opsional).
- withSchema — Sebuah string yang berisi skema untuk data yang akan di-unbox (opsional). Ini harus selalu dibuat menggunakan `StructType.json`.
- `withHeader` — `True` jika data yang di-unbox termasuk header, atau `False` jika tidak (opsional).

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

UnnestFrame kelas

Membuka sarangDynamicFrame, meratakan objek bersarang ke elemen tingkat atas, dan menghasilkan kunci gabungan untuk objek array.

Contoh

Kami menyarankan Anda menggunakan DynamicFrame.unnest()metode ini untuk meratakan struktur bersarang di a. DynamicFrame Untuk melihat contoh kode, lihatContoh: Gunakan unnest untuk mengubah bidang bersarang menjadi bidang tingkat atas.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__(frame, transformation_ctx = "", info="", stageThreshold=0, totalThreshold=0)

Membuka sarangDynamicFrame, meratakan objek bersarang ke elemen tingkat atas, dan menghasilkan kunci gabungan untuk objek array.

- frame — DynamicFrame untuk membuka nest (wajib).
- transformation_ctx — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- info — Sebuah string yang akan dikaitkan dengan kesalahan dalam transformasi (opsional).
- stageThreshold— Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional). Default-nya adalah nol.
- totalThreshold— Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum memproses kesalahan keluar (opsional). Default-nya adalah nol.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

FlagDuplicatesInColumn kelas

`FlagDuplicatesInColumn`Transformasi mengembalikan kolom baru dengan nilai tertentu di setiap baris yang menunjukkan apakah nilai di kolom sumber baris cocok dengan nilai di baris sebelumnya dari kolom sumber. Ketika kecocokan ditemukan, mereka ditandai sebagai duplikat. Kejadian awal tidak ditandai, karena tidak cocok dengan baris sebelumnya.

Contoh

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
```

```
spark = SparkSession(sc)

datasource1 = spark.read.json("s3://${BUCKET}/json/zips/raw/data")

try:
    df_output = column.FlagDuplicatesInColumn.apply(
        data_frame=datasource1,
        spark_context=sc,
        source_column="city",
        target_column="flag_col",
        true_string="True",
        false_string="False"
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

`FlagDuplicatesInColumn`Transformasi akan menambahkan kolom baru `flag_col` ke `df_output`. DataFrame Kolom ini akan berisi nilai string yang menunjukkan apakah baris yang sesuai memiliki nilai duplikat di kolom `kota` atau tidak. Jika sebuah baris memiliki nilai duplikat `city`, `flag_col` akan berisi nilai `true_string` “True”. Jika sebuah baris memiliki nilai `city` yang unik, `flag_col` akan berisi nilai `false_string` “False”.

Hasil `df_output` DataFrame akan berisi semua kolom dari `datasource1` asli, ditambah kolom `flag_col` tambahan yang menunjukkan nilai duplikat DataFrame `city`.

Metode

- __call__
- apply
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, source_column, target_column, true_string=default_true_string, false_string=default_false_string)

`FlagDuplicatesInColumn`Transformasi mengembalikan kolom baru dengan nilai tertentu di setiap baris yang menunjukkan apakah nilai di kolom sumber baris cocok dengan nilai di baris sebelumnya dari kolom sumber. Ketika kecocokan ditemukan, mereka ditandai sebagai duplikat. Kejadian awal tidak ditandai, karena tidak cocok dengan baris sebelumnya.

- `source_column`— Nama kolom sumber.
- `target_column`— Nama kolom target.
- `true_string`— String yang akan dimasukkan dalam kolom target ketika nilai kolom sumber menduplikasi nilai sebelumnya di kolom itu.
- `false_string`— String yang akan dimasukkan dalam kolom target ketika nilai kolom sumber berbeda dari nilai sebelumnya di kolom itu.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

FormatPhoneNumber kelas

`FormatPhoneNumber`Transformasi mengembalikan kolom di mana string nomor telepon diubah menjadi nilai yang diformat.

Contoh

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        ("408-341-5669",),
        ("4083415669",)
    ],
    ["phone"],
)

try:
    df_output = column_formatting.FormatPhoneNumber.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="phone",
        default_region="US"
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Outputnya akan menjadi:

```
```
+---------------+
| phone|
+---------------+
```
```

```
|(408) 341-5669|
|(408) 341-5669|
+---------------+
```

`FormatPhoneNumber`Transformasi mengambil `source_column` sebagai `'"telepon"'` dan `default_region` sebagai `'"US"'`.

Transformasi berhasil memformat kedua nomor telepon, terlepas dari format awalnya, ke format standar AS `(408) 341-5669`.

Metode

- __call__
- apply
- nama
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, source_column, phone_number_format=Tidak ada, default_region=tidak ada, default_region_column=tidak ada)

`FormatPhoneNumber`Transformasi mengembalikan kolom di mana string nomor telepon diubah menjadi nilai yang diformat.

- `source_column`— Nama kolom yang ada.
- `phone_number_format`— Format untuk mengonversi nomor telepon menjadi. Jika tidak ada format yang ditentukan, defaultnya adalah`E.164`, format nomor telepon standar yang diakui secara internasional. Nilai-nilai yang valid meliputi:
  - E164 (hilangkan periode setelah E)
- `default_region`— Kode wilayah yang valid yang terdiri dari dua atau tiga huruf besar yang menentukan wilayah untuk nomor telepon ketika tidak ada kode negara yang ada di nomor itu sendiri. Paling-paling, salah satu `defaultRegion` atau `defaultRegionColumn` dapat disediakan.

- `default_region_column`— Nama kolom tipe data lanjutan`Country`. Kode wilayah dari kolom yang ditentukan digunakan untuk menentukan kode negara untuk nomor telepon ketika tidak ada kode negara dalam nomor itu sendiri. Paling-paling, salah satu `defaultRegion` atau `defaultRegionColumn` dapat disediakan.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

FormatCase kelas

`FormatCase`Transformasi mengubah setiap string dalam kolom ke jenis kasus yang ditentukan.

Contoh

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *
```

```
sc = SparkContext()
spark = SparkSession(sc)

datasource1 = spark.read.json("s3://${BUCKET}/json/zips/raw/data")

try:
    df_output = data_cleaning.FormatCase.apply(
        data_frame=datasource1,
        spark_context=sc,
        source_column="city",
        case_type="LOWER"
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

`FormatCase`Transformasi akan mengubah nilai di kolom `city` menjadi huruf kecil berdasarkan parameter `case_type="lower"`. Hasil `df_output` DataFrame akan berisi semua kolom dari `datasource1` asli, tetapi dengan nilai kolom DataFrame `city` dalam huruf kecil.

Metode

- __call__
- apply
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, source_column, case_type)

`FormatCase`Transformasi mengubah setiap string dalam kolom ke jenis kasus yang ditentukan.

- `source_column`— Nama kolom yang ada.

- `case_type`— Jenis kasus yang didukung adalah`CAPITAL`,`LOWER`,`UPPER`,`SENTENCE`.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

FillWithMode kelas

`FillWithMode`Transformasi memformat kolom sesuai dengan format nomor telepon yang Anda tentukan. Anda juga dapat menentukan logika tie-breaker, di mana beberapa nilai identik. Misalnya, pertimbangkan nilai-nilai berikut: `1 2 2 3 3 4`

ModeType `MINIMUM` penyebab `FillWithMode` mengembalikan 2 sebagai nilai mode. Jika ModeType adalah`MAXIMUM`, modenya adalah 3. Untuk`AVERAGE`, modenya adalah 2.5.

Contoh

```
from awsglue.context import *
from pyspark.sql import SparkSession
from awsgluedi.transforms import *
```

```
sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (105.111, 13.12),
        (1055.123, 13.12),
        (None, 13.12),
        (13.12, 13.12),
        (None, 13.12),
    ],
    ["source_column_1", "source_column_2"],
)

try:
    df_output = data_quality.FillWithMode.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column_1",
        mode_type="MAXIMUM"
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Output dari kode yang diberikan adalah:

```
```
+---------------+---------------+
|source_column_1|source_column_2|
+---------------+---------------+
105.111	13.12
1055.123	13.12
1055.123	13.12
13.12	13.12
1055.123	13.12
+---------------+---------------+
```
```

`FillWithModeTransformasi` dari modul `awsglue.data_quality` diterapkan ke `input_df`. DataFrame Ini menggantikan nilai `null` di `source_column_1` kolom dengan nilai maksimum (`mode_type="maximum"`) dari nilai non-null di kolom itu.

Dalam hal ini, nilai maksimum dalam `source_column_1` kolom adalah `1055.123`. Oleh karena itu, nilai `null` di digantikan oleh `1055.123` dalam `source_column_1` output `df_output`. DataFrame

Metode

- __call__
- apply
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, source_column, mode_type)

`FillWithModeTransformasi` memformat kasus string dalam kolom.

- `source_column`— Nama kolom yang ada.
- `mode_type`— Cara mengatasi nilai dasi dalam data. Nilai ini harus salah satu dari`MINIMUM`,`NONE`,`AVERAGE`, atau`MAXIMUM`.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

FlagDuplicateRows kelas

`FlagDuplicateRows`Transformasi mengembalikan kolom baru dengan nilai tertentu di setiap baris yang menunjukkan apakah baris tersebut sama persis dengan baris sebelumnya dalam kumpulan data. Ketika kecocokan ditemukan, mereka ditandai sebagai duplikat. Kejadian awal tidak ditandai, karena tidak cocok dengan baris sebelumnya.

Contoh

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (105.111, 13.12),
        (13.12, 13.12),
        (None, 13.12),
        (13.12, 13.12),
        (None, 13.12),
    ],
    ["source_column_1", "source_column_2"],
)

try:
```

```
    df_output = data_quality.FlagDuplicateRows.apply(
        data_frame=input_df,
        spark_context=sc,
        target_column="flag_row",
        true_string="True",
        false_string="False",
        target_index=1
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

Outputnya akan berupa PySpark DataFrame kolom tambahan `flag_row` yang menunjukkan apakah baris adalah duplikat atau tidak, berdasarkan `source_column_1` kolom. Hasil `df_output` DataFrame akan berisi baris berikut:

```
```
+---------------+---------------+--------+
|source_column_1|source_column_2|flag_row|
+---------------+---------------+--------+
105.111	13.12	False
13.12	13.12	True
null	13.12	True
13.12	13.12	True
null	13.12	True
+---------------+---------------+--------+
```
```

`flag_row`Kolom menunjukkan apakah baris adalah duplikat atau tidak. `true_string` disetel ke “True”, dan `false_string` disetel ke “False”. `target_index` diatur ke 1, yang berarti bahwa `flag_row` kolom akan dimasukkan pada posisi kedua (indeks 1) dalam output. DataFrame

Metode

- __call__
- apply
- name
- describeArgs

- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, target_column, true_string=default_true_string, false_string=default_false_string, target_index=Tidak ada)

`FlagDuplicateRows`Transformasi mengembalikan kolom baru dengan nilai tertentu di setiap baris yang menunjukkan apakah baris tersebut sama persis dengan baris sebelumnya dalam kumpulan data. Ketika kecocokan ditemukan, mereka ditandai sebagai duplikat. Kejadian awal tidak ditandai, karena tidak cocok dengan baris sebelumnya.

- `true_string`— Nilai yang akan dimasukkan jika baris cocok dengan baris sebelumnya.
- `false_string`— Nilai yang akan dimasukkan jika barisnya unik.
- `target_column`— Nama kolom baru yang disisipkan dalam dataset.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

RemoveDuplicates kelas

`RemoveDuplicates`Transformasi menghapus seluruh baris, jika nilai duplikat ditemui di kolom sumber yang dipilih.

Contoh

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (105.111, 13.12),
        (13.12, 13.12),
        (None, 13.12),
        (13.12, 13.12),
        (None, 13.12),
    ],
    ["source_column_1", "source_column_2"],
)

try:
    df_output = data_quality.RemoveDuplicates.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column_1"
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

Outputnya akan a PySpark DataFrame dengan duplikat dihapus berdasarkan `source_column_1` kolom. Hasil `df_output` DataFrame akan berisi baris berikut:

```
```
+---------------+---------------+
|source_column_1|source_column_2|
+---------------+---------------+
105.111	13.12
13.12	13.12
null	13.12
+---------------+---------------+
```
```

Perhatikan bahwa baris dengan `source_column_1` nilai `13.12` dan `null` hanya muncul sekali dalam output DataFrame, karena duplikat telah dihapus berdasarkan kolom. `source_column_1`

Metode

- __call__
- apply
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, source_column)

`RemoveDuplicates`Transformasi menghapus seluruh baris, jika nilai duplikat ditemui di kolom sumber yang dipilih.

- `source_column`— Nama kolom yang ada.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

MonthName kelas

`MonthName`Transformasi membuat kolom baru yang berisi nama bulan, dari string yang mewakili tanggal.

Contoh

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

spark.conf.set("spark.sql.legacy.timeParserPolicy", "LEGACY")

input_df = spark.createDataFrame(
    [
        ("20-2018-12",),
        ("2018-20-12",),
        ("20182012",),
        ("12202018",),
        ("20122018",),
        ("20-12-2018",),
        ("12/20/2018",),
```

```
        ("02/02/02",),
        ("02 02 2009",),
        ("02/02/2009",),
        ("August/02/2009",),
        ("02/june/2009",),
        ("02/2020/june",),
        ("2013-02-21 06:35:45.658505",),
        ("August 02 2009",),
        ("2013/02/21",),
        (None,),
    ],
    ["column_1"],
)

try:
    df_output = datetime_functions.MonthName.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="column_1",
        target_column="target_column"
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Outputnya akan menjadi:

```
```
+------------+------------+
| column_1|target_column|
+------------+------------+
20-2018-12	December
2018-20-12	null
20182012	null
12202018	null
20122018	null
20-12-2018	December
12/20/2018	December
02/02/02	February
02 02 2009	February
```
```

```
|02/02/2009 | February |
|August/02/2009| August |
|02/june/2009| null |
|02/2020/june| null |
|2013-02-21 06:35:45.658505| February |
|August 02 2009| August |
| 2013/02/21| February |
| null | null |
+------------+------------+
```

`MonthNameTransformasi` mengambil `source_column` sebagai `'"column_1"'` dan `target_column` sebagai `'"target_column"'`. Ini mencoba untuk mengekstrak nama bulan dari string tanggal/waktu di kolom `'"column_1"'` dan menempatkannya di kolom `'"target_column"'`. Jika string tanggal/waktu dalam format yang tidak dikenal atau tidak dapat diuraikan, nilai `'"target_column"'` disetel ke `null`.

Transformasi berhasil mengekstrak nama bulan dari berbagai format tanggal/waktu, seperti “20-12-2018", “12/20/2018", “02/02/2009", “2013-02-21 06:35:45.658 505", dan “02 Agustus 2009".

Metode

- __call__
- apply
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, target_column, source_column=None, value=None)

`MonthNameTransformasi` membuat kolom baru yang berisi nama bulan, dari string yang mewakili tanggal.

- `source_column`— Nama kolom yang ada.
- `value`— String karakter untuk mengevaluasi..
- `target_column`— Nama untuk kolom yang baru dibuat.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

IsEven kelas

`IsEven`Transformasi mengembalikan nilai Boolean di kolom baru yang menunjukkan apakah kolom sumber atau nilai genap. Jika kolom sumber atau nilai adalah desimal, hasilnya salah.

Contoh

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [(5,), (0,), (-1,), (2,), (None,)],
    ["source_column"],
```

```
)

try:
    df_output = math_functions.IsEven.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column",
        target_column="target_column",
        value=None,
        true_string="Even",
        false_string="Not even",
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Outputnya akan menjadi:

```
```
+------------+------------+
|source_column|target_column|
+------------+------------+
5	Not even
0	Even
-1	Not even
2	Even
null	null
+------------+------------+
```
```

`IsEven`Transformasi mengambil `source_column` sebagai “source_column” dan `target_column` sebagai “target_column”. Ini memeriksa apakah nilai dalam `"source_column"` genap atau tidak. Jika nilainya genap, ia menetapkan nilai `"target_column"` ke `true_string` “Even”. Jika nilainya ganjil, ia menetapkan nilai `"target_column"` ke `false_string` “Tidak genap”. Jika nilai `"source_column"` adalah `null`, nilai `"target_column"` disetel ke `null`.

Transformasi dengan benar mengidentifikasi angka genap (0 dan 2) dan menetapkan nilai `"target_column"` menjadi “Even”. Untuk angka ganjil (5 dan -1), ia menetapkan nilai

`"target_column"` menjadi “Tidak genap”. Untuk nilai `null` di `"source_column"`, nilai `"target_column"` disetel ke `null`.

Metode

- __call__
- apply
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, target_column, source_column=None, true_string=default_true_string, false_string=default_false_string, nilai=tidak ada)

`IsEven`Transformasi mengembalikan nilai Boolean di kolom baru yang menunjukkan apakah kolom sumber atau nilai genap. Jika kolom sumber atau nilai adalah desimal, hasilnya salah.

- `source_column`— Nama kolom yang ada.
- `target_column`— Nama kolom baru yang akan dibuat.
- `true_string`— String yang menunjukkan apakah nilainya genap.
- `false_string`— String yang menunjukkan apakah nilainya tidak genap.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

CryptographicHash kelas

`CryptographicHash`Transformasi menerapkan algoritma untuk nilai hash di kolom.

Contoh

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

secret = "${SECRET}"
sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (1, "1234560000"),
        (2, "1234560001"),
        (3, "1234560002"),
        (4, "1234560003"),
        (5, "1234560004"),
        (6, "1234560005"),
        (7, "1234560006"),
        (8, "1234560007"),
        (9, "1234560008"),
        (10, "1234560009"),
    ],
    ["id", "phone"],
```

```
)

try:
    df_output = pii.CryptographicHash.apply(
        data_frame=input_df,
        spark_context=sc,
        source_columns=["id", "phone"],
        secret_id=secret,
        algorithm="HMAC_SHA256",
        output_format="BASE64",
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Outputnya akan menjadi:

```
```
+---+------------+------------------+------------------+
| id| phone | id_hashed | phone_hashed |
+---+------------+------------------+------------------+
1	1234560000	QUI1zXTJiXmfIb...	juDBAmiRnnO3g...
2	1234560001	ZAUWiZ3dVTzCo...	vC8lgUqBVDMNQ...
3	1234560002	ZP4VvZWkqYifu...	Kl3QAkgswYpzB...
4	1234560003	3u8vO3wQ8EQfj...	CPBzK1P8PZZkV...
5	1234560004	eWkQJk4zAOIzx...	aLf7+mHcXqbLs...
6	1234560005	xtI9fZCJZCvsa...	dy2DFgdYWmr0p...
7	1234560006	iW9hew7jnHuOf...	wwfGMCOEv6oOv...
8	1234560007	H9V1pqvgkFhfS...	g9WKhagIXy9ht...
9	1234560008	xDhEuHaxAUbU5...	b3uQLKPY+Q5vU...
10	1234560009	GRN6nFXkxk349...	VJdsKt8VbxBbt...
+---+------------+------------------+------------------+
```
```

Transformasi menghitung hash kriptografi dari nilai-nilai dalam kolom `id` dan `phone` menggunakan algoritma dan kunci rahasia yang ditentukan, dan mengkodekan hash dalam format Base64. Hasil `df_output` DataFrame berisi semua kolom dari `input_df` asli, ditambah kolom `id_hashed` dan `phone_hashed` DataFrame tambahan dengan hash yang dihitung.

Metode

- __call__
- apply
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, source_columns, secret_id, algoritma = None, secret_version=None, create_secret_if_missing=false, output_format=None, entity_type_filter=None)

`CryptographicHash`Transformasi menerapkan algoritma untuk nilai hash di kolom.

- `source_columns`— Array kolom yang ada.
- `secret_id`— ARN dari kunci rahasia Secrets Manager. Kunci yang digunakan dalam algoritma awalan kode otentikasi pesan berbasis hash (HMAC) untuk hash kolom sumber.
- `secret_version` – Opsional. Default ke versi rahasia terbaru.
- `entity_type_filter`— Array opsional tipe entitas. Dapat digunakan untuk mengenkripsi hanya PII yang terdeteksi di kolom teks bebas.
- `create_secret_if_missing`— Boolean opsional. Jika benar akan mencoba untuk membuat rahasia atas nama penelepon.
- `algorithm`— Algoritma yang digunakan untuk hash data Anda. Nilai enum yang valid: MD5, SHA1, SHA256, SHA512, HMAC_MD5, HMAC_SHA1, HMAC_SHA256, HMAC_SHA512.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

Dekripsi kelas

`DecryptTransformasi` mendekripsi di dalam Glue AWS . Data Anda juga dapat didekripsi di luar AWS Glue dengan AWS Encryption SDK. Jika ARN kunci KMS yang disediakan tidak cocok dengan apa yang digunakan untuk mengenkripsi kolom, operasi dekripsi gagal.

Contoh

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

kms = "${KMS}"
sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (1, "1234560000"),
        (2, "1234560001"),
        (3, "1234560002"),
        (4, "1234560003"),
        (5, "1234560004"),
```

```
        (6, "1234560005"),
        (7, "1234560006"),
        (8, "1234560007"),
        (9, "1234560008"),
        (10, "1234560009"),
    ],
    ["id", "phone"],
)

try:
    df_encrypt = pii.Encrypt.apply(
        data_frame=input_df,
        spark_context=sc,
        source_columns=["phone"],
        kms_key_arn=kms
    )
    df_decrypt = pii.Decrypt.apply(
        data_frame=df_encrypt,
        spark_context=sc,
        source_columns=["phone"],
        kms_key_arn=kms
    )
    df_decrypt.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Outputnya akan berupa PySpark DataFrame kolom `id` asli dan kolom `telepon` yang didekripsi:

```
```
+---+------------+
| id| phone|
+---+------------+
1	1234560000
2	1234560001
3	1234560002
4	1234560003
5	1234560004
6	1234560005
7	1234560006
8	1234560007
```
```

```
| 9| 1234560008|
| 10| 1234560009|
+---+------------+
```

`EncryptTransformasi` mengambil `source_columns` sebagai `["phone"]` dan `kms_key_arn` sebagai nilai variabel lingkungan `$ {KMS} `. Transformasi mengenkripsi nilai di kolom `telepon` menggunakan kunci KMS yang ditentukan. DataFrame `df_encrypt` terenkripsi kemudian diteruskan ke transformasi dari modul `awsglue.pii`. `Decrypt` Dibutuhkan `source_columns` sebagai `["phone"]` dan `kms_key_arn` sebagai nilai variabel lingkungan `$ {KMS} `. Transformasi mendekripsi nilai terenkripsi di kolom `telepon` menggunakan kunci KMS yang sama. Hasil `df_decrypt` DataFrame berisi kolom `id` asli dan kolom `telepon` yang didekripsi.

Metode

- __call__
- apply
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, source_columns, kms_key_arn)

`DecryptTransformasi` mendekripsi di dalam Glue AWS . Data Anda juga dapat didekripsi di luar AWS Glue dengan AWS Encryption SDK. Jika ARN kunci KMS yang disediakan tidak cocok dengan apa yang digunakan untuk mengenkripsi kolom, operasi dekripsi gagal.

- `source_columns`— Array kolom yang ada.
- `kms_key_arn`— Kunci ARN dari kunci Layanan Manajemen AWS Kunci yang digunakan untuk mendekripsi kolom sumber.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

Enkripsi kelas

`EncryptTransformasi` mengenkripsi kolom sumber menggunakan AWS kunci Key Management Service. `EncryptTransformasi` dapat mengenkripsi hingga 128 MiB per sel. Ini akan mencoba untuk mempertahankan format pada dekripsi. Untuk mempertahankan tipe data, metadata tipe data harus diserialisasikan hingga kurang dari 1KB. Jika tidak, Anda harus mengatur `preserve_data_type` parameter ke false. Metadata tipe data akan disimpan dalam plaintext dalam konteks enkripsi.

Contoh

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

kms = "${KMS}"
sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
```

```
    [
        (1, "1234560000"),
        (2, "1234560001"),
        (3, "1234560002"),
        (4, "1234560003"),
        (5, "1234560004"),
        (6, "1234560005"),
        (7, "1234560006"),
        (8, "1234560007"),
        (9, "1234560008"),
        (10, "1234560009"),
    ],
    ["id", "phone"],
)

try:
    df_encrypt = pii.Encrypt.apply(
        data_frame=input_df,
        spark_context=sc,
        source_columns=["phone"],
        kms_key_arn=kms
    )
except:
    print("Unexpected Error happened ")
    raise
```

Output

Outputnya akan berupa PySpark DataFrame kolom `id` asli dan kolom tambahan yang berisi nilai terenkripsi dari kolom `telepon`.

```
```
+---+------------+-------------------------+
| id| phone | phone_encrypted |
+---+------------+-------------------------+
1	1234560000	EncryptedData1234...abc
2	1234560001	EncryptedData5678...def
3	1234560002	EncryptedData9012...ghi
4	1234560003	EncryptedData3456...jkl
5	1234560004	EncryptedData7890...mno
6	1234560005	EncryptedData1234...pqr
7	1234560006	EncryptedData5678...stu
```
```

```
| 8| 1234560007| EncryptedData9012...vwx |
| 9| 1234560008| EncryptedData3456...yz0 |
| 10| 1234560009| EncryptedData7890...123 |
+---+------------+------------------------+
```

`EncryptTransformasi` mengambil `source_columns` sebagai `["phone"]` dan `kms_key_arn` sebagai nilai variabel lingkungan `$ {KMS} `. Transformasi mengenkripsi nilai di kolom `telepon` menggunakan kunci KMS yang ditentukan. Hasil `df_encrypt` DataFrame berisi kolom `id` asli, kolom `phone` asli, dan kolom tambahan bernama `phone_encrypted` yang berisi nilai terenkripsi dari kolom `phone`.

Metode

- __call__
- apply
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, source_columns, kms_key_arn, entity_type_filter=Tidak ada, preserve_data_type = tidak ada)

`EncryptTransformasi` mengenkripsi kolom sumber menggunakan AWS kunci Key Management Service.

- `source_columns`— Array kolom yang ada.
- `kms_key_arn`— Kunci ARN dari kunci Layanan Manajemen AWS Kunci yang akan digunakan untuk Mengenkripsi kolom sumber.
- `entity_type_filter`— Array opsional tipe entitas. Dapat digunakan untuk mengenkripsi hanya PII yang terdeteksi di kolom teks bebas.
- `preserve_data_type`— Boolean opsional. Default ke true. Jika false, tipe data tidak akan disimpan.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

IntToIp kelas

`IntToIpTransformasi` mengubah nilai integer kolom sumber atau nilai lainnya ke nilai IPv4 yang sesuai di kolom target, dan mengembalikan hasilnya di kolom baru.

Contoh

```
from pyspark.context import SparkContext
from pyspark.sql import SparkSession
from awsgluedi.transforms import *

sc = SparkContext()
spark = SparkSession(sc)

input_df = spark.createDataFrame(
    [
        (3221225473,),
        (0,),
```

```
        (1,),
        (100,),
        (168430090,),
        (4294967295,),
        (4294967294,),
        (4294967296,),
        (-1,),
        (None,),
    ],
    ["source_column_int"],
)

try:
    df_output = web_functions.IntToIp.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column_int",
        target_column="target_column",
        value=None
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Output

Outputnya adalah:

```
```
+---------------+---------------+
|source_column_int|target_column|
+---------------+---------------+
3221225473	192.0.0.1
0	0.0.0.0
1	0.0.0.1
100	0.0.0.100
168430090	10.0.0.10
4294967295	255.255.255.255
4294967294	255.255.255.254
4294967296	null
-1	null
null	null
```
```

```
+---------------+---------------+
```
```

`IntToIp.apply`Transformasi mengambil `source_column` sebagai `"source_column_int"` dan `target_column` sebagai `"target_column"` dan mengubah nilai integer di kolom `source_column_int` ke representasi alamat IPv4 yang sesuai dan menyimpan hasilnya di kolom `target_column`.

Untuk nilai integer yang valid dalam kisaran alamat IPv4 (0 hingga 4294967295), transformasi berhasil mengubahnya menjadi representasi alamat IPv4 mereka (misalnya, 192.0.0.1, 0.0.0.0, 10.0.0.10, 255.255.255.255).

Untuk nilai integer di luar rentang yang valid (misalnya, 4294967296, -1), nilai `target_column` diatur ke `null`. Untuk nilai `null` di kolom `source_column_int`, nilai `target_column` juga diatur ke `null`.

Metode

- __call__
- apply
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, target_column, source_column=None, value=None)

`IntToIp`Transformasi mengubah nilai integer kolom sumber atau nilai lainnya ke nilai IPv4 yang sesuai di kolom target, dan mengembalikan hasilnya di kolom baru.

- `sourceColumn`— Nama kolom yang ada.
- `value`— String karakter untuk dievaluasi.
- `targetColumn`— Nama kolom baru yang akan dibuat.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.
```

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

IpToInt kelas

`IpToInt`Transformasi mengubah nilai Internet Protocol versi 4 (IPv4) dari kolom sumber atau nilai lainnya ke nilai integer yang sesuai di kolom target, dan mengembalikan hasilnya di kolom baru.

Contoh

Untuk AWS Glue 4.0 dan yang lebih baru, buat atau perbarui argumen pekerjaan dengan `key: --enable-glue-di-transforms, value: true`

```
from pyspark.context import SparkContext
from awsgluedi.transforms import *

sc = SparkContext()

input_df = spark.createDataFrame(
    [
        ("192.0.0.1",),
        ("10.10.10.10",),
        ("1.2.3.4",),
        ("1.2.3.6",),
```

```
        ("http://12.13.14.15",),
        ("https://16.17.18.19",),
        ("1.2.3.4",),
        (None,),
        ("abc",),
        ("abc.abc.abc.abc",),
        ("321.123.123.123",),
        ("244.4.4.4",),
        ("255.255.255.255",),
    ],
    ["source_column_ip"],
)

    df_output = web_functions.IpToInt.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column_ip",
        target_column="target_column",
        value=None
    )
    df_output.show()
```

Output

Outputnya akan menjadi:

```
```
+----------------+---------------+
|source_column_ip| target_column|
+----------------+---------------+
192.0.0.1	3221225473
10.10.10.10	168427722
1.2.3.4	16909060
1.2.3.6	16909062
http://12.13.14.15	null
https://16.17.18.19	null
1.2.3.4	16909060
null	null
abc	null
abc.abc.abc.abc	null
321.123.123.123	null
244.4.4.4	4102444804
255.255.255.255	4294967295
```
```

```
+----------------+---------------+
```
```

`IpToIntTransformasi` mengambil `source_column` sebagai `'"source_column_ip"'` dan `target_column` sebagai `'"target_column"'` dan mengubah string alamat IPv4 yang valid di kolom `source_column_ip` ke representasi bilangan bulat 32-bit yang sesuai dan menyimpan hasilnya di kolom `target_column`.

Untuk string alamat IPv4 yang valid (misalnya, “192.0.0.1", “10.10.10.10", “1.2.3.4"), transformasi berhasil mengubahnya menjadi representasi bilangan bulat mereka (misalnya, 3221225473, 168427722, 16909060). Untuk string yang bukan alamat IPv4 yang valid (misalnya, URL, string non-IP seperti “abc”, format IP tidak valid seperti “abc.abc.abc.abc”), nilai `target_column` disetel ke `null`. Untuk nilai `null` di kolom `source_column_ip`, nilai `target_column` juga diatur ke `null`.

Metode

- __call__
- apply
- name
- describeArgs
- describeReturn
- describeTransform
- describeErrors
- describe

__call__ (spark_context, data_frame, target_column, source_column=None, value=None)

`IpToIntTransformasi` mengubah nilai Internet Protocol versi 4 (IPv4) dari kolom sumber atau nilai lainnya ke nilai integer yang sesuai di kolom target, dan mengembalikan hasilnya di kolom baru.

- `sourceColumn`— Nama kolom yang ada.
- `value`— String karakter untuk dievaluasi.
- `targetColumn`— Nama kolom baru yang akan dibuat.

apply(cls, *args, **kwargs)

Warisan dari `GlueTransform` apply.
```

name(cls)

Warisan dari `GlueTransform` nama.

describeArgs(cls)

Warisan dari `GlueTransform` describeArgs.

describeReturn(cls)

Warisan dari `GlueTransform` describeReturn.

describeTransform(cls)

Warisan dari `GlueTransform` describeTransform.

describeErrors(cls)

Warisan dari `GlueTransform` describeErrors.

describe(cls)

Warisan dari `GlueTransform` describe.

Integrasi data berubah

Untuk AWS Glue 4.0 dan di atasnya, buat atau perbarui argumen pekerjaan dengan`key: --enable-glue-di-transforms, value: true`.

Contoh skrip pekerjaan:

```
from pyspark.context import SparkContext

from awsgluedi.transforms import *
sc = SparkContext()

input_df = spark.createDataFrame(
    [(5,), (0,), (-1,), (2,), (None,)],
    ["source_column"],
)

try:
    df_output = math_functions.IsEven.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column",
```

```
        target_column="target_column",
        value=None,
        true_string="Even",
        false_string="Not even",
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Contoh Sesi menggunakan Notebook

```
%idle_timeout 2880
%glue_version 4.0
%worker_type G.1X
%number_of_workers 5
%region eu-west-1
```

```
%%configure
{
    "--enable-glue-di-transforms": "true"
}
```

```
from pyspark.context import SparkContext
from awsgluedi.transforms import *

sc = SparkContext()

input_df = spark.createDataFrame(
    [(5,), (0,), (-1,), (2,), (None,)],
    ["source_column"],
)

try:
    df_output = math_functions.IsEven.apply(
        data_frame=input_df,
        spark_context=sc,
        source_column="source_column",
        target_column="target_column",
        value=None,
        true_string="Even",
        false_string="Not even",
```

```
    )
    df_output.show()
except:
    print("Unexpected Error happened ")
    raise
```

Contoh Sesi menggunakan AWS CLI

```
aws glue create-session --default-arguments "--enable-glue-di-transforms=true"
```

DI mengubah:

- FlagDuplicatesInColumn kelas
- FormatPhoneNumber kelas
- FormatCase kelas
- FillWithMode kelas
- FlagDuplicateRows kelas
- RemoveDuplicates kelas
- MonthName kelas
- IsEven kelas
- CryptographicHash kelas
- Dekripsi kelas
- Enkripsi kelas
- IntToIp kelas
- IpToInt kelas

Maven: Bundel plugin dengan aplikasi Spark Anda

Anda dapat menggabungkan dependensi transformasi dengan aplikasi Spark dan distribusi Spark Anda (versi 3.3) dengan menambahkan ketergantungan plugin di `pom.xml` Maven Anda sambil mengembangkan aplikasi Spark Anda secara lokal.

```
<repositories>
   ...
    <repository>
        <id>aws-glue-etl-artifacts</id>
        <url>https://aws-glue-etl-artifacts.s3.amazonaws.com/release/ </url>
```

```
    </repository>
</repositories>
...
<dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>AWSGlueTransforms</artifactId>
    <version>4.0.0</version>
</dependency>
```

Anda dapat mengunduh binari dari artefak AWS Glue Maven secara langsung dan memasukkannya ke dalam aplikasi Spark Anda sebagai berikut.

```
#!/bin/bash
sudo wget -v https://aws-glue-etl-artifacts.s3.amazonaws.com/release/com/amazonaws/
AWSGlueTransforms/4.0.0/AWSGlueTransforms-4.0.0.jar -P /usr/lib/spark/jars/
```

Pemrograman skrip AWS Glue ETL dalam Scala

Anda dapat menemukan contoh kode Scala dan utilitas untuk AWS Glue dalam repositori AWS Glue sampel di situs web. GitHub

AWSGlue mendukung perpanjangan dialek PySpark Scala untuk pekerjaan scripting extract, transform, and load (ETL). Bagian berikut menjelaskan cara menggunakan pustaka AWS Glue Scala dan AWS Glue API dalam skrip ETL, dan menyediakan dokumentasi referensi untuk pustaka.

Daftar Isi

- Filter Def
- Def getName
- Def getNumPartitions
- Def Dihitung getSchemaIf
- Def isSchemaComputed
- Def javaToPython
- Def bergabung
- Def peta
- Def mergeDynamicFrames
- PrintSkema Cetak Def
- ReComputesChema Def
- Def relasialisasi
- Def RenameField
- Def repartisi
- Def ResolveChoice
- Skema Def
- SelectField Def
- SelectFields Def
- Pertunjukan def
- Def SederhanaDDBJSON
- Keran Def
- Def SplitFields
- Def SplitRows
- Def stageErrorsCount
- Def ToDF
- Def membuka kotak
- Def tidak bersarang
- Def Unnestddbjson
- Def withFrameSchema
- Def withName

        - def createDataFrame FromOptions
        - forEachBatch
        - def getCatalogSink
        - def getCatalogSource
        - def getJDBCSink
        - def getSink
        - Format def getSinkWith
        - def getSource
        - Format def getSourceWith
        - def getSparkSession
        - def StartTransaksi
        - def CommitTransaction
        - def batalkan Transaksi
        - def this
        - def this
        - def this
    - MappingSpec
        - MappingSpec kelas kasus
        - MappingSpec objek
        - Val orderingByTarget
        - Def berlaku
        - Def berlaku
        - Def berlaku
    - AWS GlueAPI Scala ResolveSpec
        - ResolveSpec objek
            - Def
            - Def
        - ResolveSpec kelas kasus
            - ResolveSpec metode def

- AWS GlueAPI Scala ArrayNode

Menggunakan Scala untuk memprogram skrip AWS Glue ETL

Anda dapat secara otomatis menghasilkan program ekstrak, transformasi, dan muat (ETL) Scala menggunakan AWS Glue konsol, dan memodifikasinya sesuai kebutuhan sebelum menetapkannya ke pekerjaan. Atau, Anda bisa tulis program Anda sendiri dari scratch. Untuk informasi lebih lanjut, lihatMengkonfigurasi properti pekerjaan untuk pekerjaan Spark di AWS Glue. AWS Gluekemudian mengkompilasi program Scala Anda di server sebelum menjalankan pekerjaan terkait.

Untuk memastikan bahwa program Anda dikompilasi tanpa kesalahan dan berjalan seperti yang diharapkan, penting bagi Anda untuk memuatnya pada titik akhir pengembangan di REPL (Read-Eval-Print Loop) atau Notebook Jupyter dan mengujinya di sana sebelum menjalankannya dalam suatu pekerjaan. Karena proses kompilasi terjadi pada server, Anda tidak akan memiliki visibilitas yang baik pada masalah yang terjadi di sana.

Menguji program Scala ETL di notebook Jupyter pada titik akhir pengembangan

Untuk menguji program Scala pada titik akhir AWS Glue pengembangan, siapkan titik akhir pengembangan seperti yang dijelaskan dalam. Menambahkan titik akhir pengembangan

Selanjutnya, sambungkan ke Notebook Jupyter yang berjalan secara lokal di komputer Anda atau dari jarak jauh di server notebook Amazon EC2. Untuk menginstal versi lokal Notebook Jupyter, ikuti petunjuk di. Tutorial: Notebook Jupyter di JupyterLab

Satu-satunya perbedaan antara menjalankan kode Scala dan menjalankan PySpark kode pada Notebook adalah Anda harus memulai setiap paragraf di Notebook dengan yang berikut:

```
%spark
```

Hal ini mencegah server Notebook dari default ke PySpark ragam interpreter Spark.

Menguji program Scala ETL dalam Scala REPL

Anda dapat menguji program Scala pada titik akhir pengembangan menggunakan AWS Glue Scala REPL. Ikuti instruksi di Tutorial: Gunakan SageMaker notebook, kecuali pada akhir perintah SSH-to-REPL, ganti `-t gluepyspark` dengan `-t glue-spark-shell`. Ini memanggil AWS Glue Scala REPL.

Untuk menutup REPL setelah Anda selesai, ketik `sys.exit`.

Contoh skrip scala - streaming ETL

Example

Contoh skrip berikut menghubungkan ke Amazon Kinesis Data Streams, menggunakan skema dari Katalog Data untuk mengurai aliran data, menggabungkan pengaliran ke set data statis di Amazon S3, dan meng-output hasil gabungannya ke Amazon S3 dalam format parket.

```
// This script connects to an Amazon Kinesis stream, uses a schema from the data
 catalog to parse the stream,
// joins the stream to a static dataset on Amazon S3, and outputs the joined results to
 Amazon S3 in parquet format.
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import java.util.Calendar
import org.apache.spark.SparkContext
import org.apache.spark.sql.Dataset
import org.apache.spark.sql.Row
import org.apache.spark.sql.SaveMode
import org.apache.spark.sql.SparkSession
import org.apache.spark.sql.functions.from_json
import org.apache.spark.sql.streaming.Trigger
import scala.collection.JavaConverters._

object streamJoiner {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val sparkSession: SparkSession = glueContext.getSparkSession
    import sparkSession.implicits._
    // @params: [JOB_NAME]
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val staticData = sparkSession.read          // read() returns type DataFrameReader
      .format("csv")
      .option("header", "true")
      .load("s3://awsexamplebucket-streaming-demo2/inputs/productsStatic.csv")  //
 load() returns a DataFrame

    val datasource0 = sparkSession.readStream   // readstream() returns type
 DataStreamReader
```

```
      .format("kinesis")
      .option("streamName", "stream-join-demo")
      .option("endpointUrl", "https://kinesis.us-east-1.amazonaws.com")
      .option("startingPosition", "TRIM_HORIZON")
      .load                                     // load() returns a DataFrame

    val selectfields1 = datasource0.select(from_json($"data".cast("string"),
 glueContext.getCatalogSchemaAsSparkSchema("stream-demos", "stream-join-demo2")) as
 "data").select("data.*")

    val datasink2 = selectfields1.writeStream.foreachBatch { (dataFrame: Dataset[Row],
 batchId: Long) => {   //foreachBatch() returns type DataStreamWriter
      val joined = dataFrame.join(staticData, "product_id")
      val year: Int = Calendar.getInstance().get(Calendar.YEAR)
      val month :Int = Calendar.getInstance().get(Calendar.MONTH) + 1
      val day: Int = Calendar.getInstance().get(Calendar.DATE)
      val hour: Int = Calendar.getInstance().get(Calendar.HOUR_OF_DAY)

      if (dataFrame.count() > 0) {
        joined.write                            // joined.write returns type
 DataFrameWriter
          .mode(SaveMode.Append)
          .format("parquet")
          .option("quote", " ")
          .save("s3://awsexamplebucket-streaming-demo2/output/" + "/year=" +
 "%04d".format(year) + "/month=" + "%02d".format(month) + "/day=" + "%02d".format(day)
 + "/hour=" + "%02d".format(hour) + "/")
      }
    }
    }  // end foreachBatch()
      .trigger(Trigger.ProcessingTime("100 seconds"))
      .option("checkpointLocation", "s3://awsexamplebucket-streaming-demo2/
checkpoint/")
      .start().awaitTermination()             // start() returns type StreamingQuery
    Job.commit()
  }
}
```

API di pustaka AWS Glue Scala

AWS Gluemendukung perpanjangan dialek PySpark Scala untuk pekerjaan ekstrak, transformasi, dan pemuatan skrip (ETL). Bagian berikut menjelaskan API di pustaka AWS Glue Scala.

com.amazonaws.services.glue

Paket com.amazonaws.services.glue di pustaka Scala berisi API berikut: AWS Glue

- ChoiceOption
- DataSink
- DataSource sifat
- DynamicFrame
- DynamicRecord
- GlueContext
- MappingSpec
- ResolveSpec

com.amazonaws.services.glue.ml~

Paket com.amazonaws.services.glue.mldi pustaka Scala berisi API berikut: AWS Glue

- FillMissingValues
- FindIncrementalMatches
- FindMatches

com.amazonaws.services.glue.dq

Paket com.amazonaws.services.glue.dq di pustaka Scala berisi API berikut: AWS Glue

- EvaluateDataQuality

com.amazonaws.services.glue.types

Paket com.amazonaws.services.glue.types di pustaka Scala berisi API berikut: AWS Glue

- ArrayNode
- BinaryNode
- BooleanNode
- ByteNode

- DateNode
- DecimalNode
- DoubleNode
- DynamicNode
- FloatNode
- IntegerNode
- LongNode
- MapLikeNode
- MapNode
- NullNode
- ObjectNode
- ScalarNode
- ShortNode
- StringNode
- TimestampNode

com.amazonaws.services.glue.util

Paket com.amazonaws.services.glue.util di pustaka Scala berisi API berikut: AWS Glue

- GlueArgParser
- Tugas

AWS GlueAPI Scala ChoiceOption

Topik

- ChoiceOption sifat
- ChoiceOption objek
- Kelas kasus ChoiceOptionWithResolver
- Kelas kasus MatchCatalogSchemaChoiceOption

Package: com.amazonaws.services.glue

ChoiceOption sifat

```
trait ChoiceOption extends Serializable
```

ChoiceOption objek

ChoiceOption

```
object ChoiceOption
```

Strategi umum untuk menyelesaikan pilihan yang berlaku untuk semua simpul `ChoiceType` dalam `DynamicFrame`.

- `val CAST`
- `val MAKE_COLS`
- `val MAKE_STRUCT`
- `val MATCH_CATALOG`
- `val PROJECT`

Def berlaku

```
def apply(choice: String): ChoiceOption
```

Kelas kasus ChoiceOptionWithResolver

```
case class ChoiceOptionWithResolver(name: String, choiceResolver: ChoiceResolver)
 extends ChoiceOption {}
```

Kelas kasus MatchCatalogSchemaChoiceOption

```
case class MatchCatalogSchemaChoiceOption() extends ChoiceOption {}
```

DataSink Kelas abstrak

Topik

- Def writeDynamicFrame

Package: com.amazonaws.services.glue

```
abstract class DataSink
```

Penulis meng-analog-kan ke `DataSource`. `DataSink` merangkum tujuan dan format yang dapat ditulis oleh `DynamicFrame`.

Def writeDynamicFrame

```
def writeDynamicFrame( frame : DynamicFrame,
                       callSite : CallSite = CallSite("Not provided", "")
                     ) : DynamicFrame
```

Bingkai Def pyWriteDynamic

```
def pyWriteDynamicFrame( frame : DynamicFrame,
                         site : String = "Not provided",
                         info : String = "" )
```

Def writeDataFrame

```
def writeDataFrame(frame: DataFrame,
                   glueContext: GlueContext,
```

```
                    callSite: CallSite = CallSite("Not provided", "")
                    ): DataFrame
```

Bingkai Def pyWriteData

```
def pyWriteDataFrame(frame: DataFrame,
                     glueContext: GlueContext,
                     site: String = "Not provided",
                     info: String = ""
                     ): DataFrame
```

Def setCatalogInfo

```
def setCatalogInfo(catalogDatabase: String,
                   catalogTableName : String,
                   catalogId : String = "")
```

SupportsFormat Def

```
def supportsFormat( format : String ) : Boolean
```

SetFormat Def

```
def setFormat( format : String,
               options : JsonOptions
             ) : Unit
```

Def WithFormat

```
def withFormat( format : String,
                options : JsonOptions = JsonOptions.empty
              ) : DataSink
```

Def setAccumulableSize

```
def setAccumulableSize( size : Int ) : Unit
```

Def getOutputError RecordsAccumulable

```
def getOutputErrorRecordsAccumulable : Accumulable[List[OutputError], OutputError]
```

Bingkai Def errorsAsDynamic

```
def errorsAsDynamicFrame : DynamicFrame
```

DataSink objek

```
object DataSink
```

Def RecordMetrics

```
def recordMetrics( frame : DynamicFrame,
                   ctxt : String
                 ) : DynamicFrame
```

AWS GlueSifat skala DataSource

Package: com.amazonaws.services.glue

Antarmuka tingkat tinggi untuk memproduksi sebuah `DynamicFrame`.

```
trait DataSource {

  def getDynamicFrame : DynamicFrame

  def getDynamicFrame( minPartitions : Int,
                       targetPartitions : Int
                     ) : DynamicFrame
  def getDataFrame : DataFrame

  /** @param num: the number of records for sampling.
    * @param options: optional parameters to control sampling behavior. Current
 available parameter for Amazon S3 sources in options:
    *  1. maxSamplePartitions: the maximum number of partitions the sampling will
 read.
    *  2. maxSampleFilesPerPartition: the maximum number of files the sampling will
 read in one partition.
```

```
    */
  def getSampleDynamicFrame(num:Int, options: JsonOptions = JsonOptions.empty):
 DynamicFrame

  def glueContext : GlueContext

  def setFormat( format : String,
                 options : String
               ) : Unit

  def setFormat( format : String,
                 options : JsonOptions
               ) : Unit

  def supportsFormat( format : String ) : Boolean

  def withFormat( format : String,
                  options : JsonOptions = JsonOptions.empty
                ) : DataSource
}
```

AWS GlueAPI Scala DynamicFrame

Package: com.amazonaws.services.glue

Daftar Isi

- Def Dihitung getSchemaIf
- Def isSchemaComputed
- Def javaToPython
- Def bergabung
- Def peta
- Def mergeDynamicFrames
- PrintSkema Cetak Def
- ReComputesChema Def
- Def relasialisasi
- Def RenameField
- Def repartisi
- Def ResolveChoice
- Skema Def
- SelectField Def
- SelectFields Def
- Pertunjukan def
- Def SederhanaDDBJSON
- Keran Def
- Def SplitFields
- Def SplitRows
- Def stageErrorsCount
- Def ToDF
- Def membuka kotak
- Def tidak bersarang
- Def Unnestddbjson
- Def withFrameSchema
- Def withName
- Def withTransformationContext

- DynamicFrame Objeknya
  - Def berlaku

- Def emptyDynamicFrame
- Def dariPythonRDD
- Def IgnoreErrors
- Def InlineErrors
- Kesalahan Def newFrameWith

AWS GlueKelas scala DynamicFrame

Package: com.amazonaws.services.glue

```
class DynamicFrame extends Serializable with Logging  (
           val glueContext : GlueContext,
           _records : RDD[DynamicRecord],
           val name : String = s"",
           val transformationContext : String = DynamicFrame.UNDEFINED,
           callSite : CallSite = CallSite("Not provided", ""),
           stageThreshold : Long = 0,
           totalThreshold : Long = 0,
           prevErrors : => Long = 0,
           errorExpr : => Unit = {} )
```

Sebuah `DynamicFrame` adalah koleksi terdistribusi dari objek DynamicRecord yang mendeskripsi secara mandiri.

`DynamicFrame` dirancang untuk menyediakan model data yang fleksibel untuk operasi ETL (ekstrak, transformasi, dan muat). Mereka tidak mengharuskan membuat sebuah skema, dan Anda dapat menggunakannya untuk membaca dan melakukan transformasi pada data yang berisi nilai dan jenis yang berantakan atau tidak konsisten. Sebuah skema dapat dihitung berdasarkan permintaan untuk operasi-operasi yang membutuhkannya.

`DynamicFrame` menyediakan berbagai transformasi untuk pembersihan data dan ETL. Mereka juga mendukung konversi ke dan dari DataFrames SparkSQL untuk diintegrasikan dengan kode yang ada dan banyak operasi analitik yang menyediakan. DataFrames

Parameter berikut dibagi di banyak AWS Glue transformasi yang membangun s`DynamicFrame`:

- `transformationContext` — Pengidentifikasi untuk `DynamicFrame` ini. `transformationContext` digunakan sebagai kunci untuk status bookmark tugas yang tetap ada di seluruh eksekusi.

- `callSite` — Menyediakan informasi konteks untuk pelaporan kesalahan. Nilai-nilai ini secara otomatis ditetapkan ketika memanggil dari Python.
- `stageThreshold` — Jumlah maksimum catatan kesalahan yang diizinkan dari komputasi `DynamicFrame` ini sebelum melemparkan pengecualian, tidak termasuk catatan yang ada dalam `DynamicFrame` sebelumnya.
- `totalThreshold` — Jumlah maksimum total catatan kesalahan sebelum pengecualian dilemparkan, termasuk yang dari bingkai sebelumnya.

ErrorsCount Val

```
val errorsCount
```

Jumlah catatan kesalahan dalam `DynamicFrame` ini. Ini termasuk kesalahan dari operasi sebelumnya.

Def ApplyMapping

```
def applyMapping( mappings : Seq[Product4[String, String, String, String]],
                  caseSensitive : Boolean = true,
                  transformationContext : String = "",
                  callSite : CallSite = CallSite("Not provided", ""),
                  stageThreshold : Long = 0,
                  totalThreshold : Long = 0
                ) : DynamicFrame
```

- `mappings` — Urutan pemetaan untuk membangun `DynamicFrame` baru.
- `caseSensitive` — Apakah akan memperlakukan kolom sumber sebagai kolom yang peka huruf besar dan kecil. Mengatur ini ke false mungkin membantu ketika mengintegrasikan dengan penyimpanan yang peka huruf besar dan kecil seperti Katalog Data Glue AWS.

Memilih, memproyeksikan, dan melemparkan kolom berdasarkan urutan pemetaan.

Setiap pemetaan terdiri dari kolom sumber dan jenis serta kolom target dan jenis. Pemetaan dapat ditentukan sebagai empat tupel (`source_path`, `source_type`, `target_path`, `target_type`) atau objek MappingSpec yang berisi informasi yang sama.

Selain menggunakan pemetaan untuk proyeksi dan transmisi sederhana, Anda dapat menggunakannya untuk melakukan nest atau membuka nest pada bidang dengan memisahkan komponen dari path dengan tanda '`.`' (titik).

Sebagai contoh, anggaplah Anda memiliki `DynamicFrame` dengan skema berikut.

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address: struct
  |    |-- state: string
  |    |-- zip: int
}}}
```

Anda dapat membuat panggilan berikut untuk membuka sarang dari bidang `state` dan `zip`.

```
{{{
  df.applyMapping(
   Seq(("name", "string", "name", "string"),
       ("age", "int", "age", "int"),
       ("address.state", "string", "state", "string"),
       ("address.zip", "int", "zip", "int")))
}}}
```

Skema yang dihasilkan adalah sebagai berikut.

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- state: string
  |-- zip: int
}}}
```

Anda juga dapat menggunakan `applyMapping` untuk melakukan nest kembali pada kolom. Sebagai contoh, berikut ini membalikkan transformasi sebelumnya dan membuat sebuah struct bernama `address` dalam target.

```
{{{
  df.applyMapping(
```

```
    Seq(("name", "string", "name", "string"),
        ("age", "int", "age", "int"),
        ("state", "string", "address.state", "string"),
        ("zip", "int", "address.zip", "int")))
 }}}
```

Nama kolom yang berisi karakter '`.`' (titik) dapat dikutip dengan menggunakan karakter backtick (`` ` ``).

> **Note**
>
> Saat ini, Anda tidak dapat menggunakan metode `applyMapping` untuk memetakan kolom yang bersarang di bawah array.

Def assertErrorThreshold

```
def assertErrorThreshold : Unit
```

Tindakan yang memaksa perhitungan dan memverifikasi bahwa jumlah catatan kesalahan berada di bawah `stageThreshold` dan `totalThreshold`. Melempar pengecualian jika salah satu kondisi gagal.

Hitungan def

```
lazy
def count
```

Mengembalikan jumlah elemen dalam `DynamicFrame` ini.

DropField Def

```
def dropField( path : String,
               transformationContext : String = "",
               callSite : CallSite = CallSite("Not provided", ""),
               stageThreshold : Long = 0,
               totalThreshold : Long = 0
             ) : DynamicFrame
```

Mengembalikan `DynamicFrame` baru dengan kolom tertentu dihapus.

DropFields Def

```
def dropFields( fieldNames : Seq[String],   // The column names to drop.
                transformationContext : String = "",
                callSite : CallSite = CallSite("Not provided", ""),
                stageThreshold : Long = 0,
                totalThreshold : Long = 0
              ) : DynamicFrame
```

Mengembalikan `DynamicFrame` baru dengan kolom-kolom tertentu dihapus.

Anda dapat menggunakan metode ini untuk menghapus kolom bersarang, termasuk yang ada dalam array, tetapi tidak untuk membuang elemen array tertentu.

DropNulls Def

```
def dropNulls( transformationContext : String = "",
               callSite : CallSite = CallSite("Not provided", ""),
               stageThreshold : Long = 0,
               totalThreshold : Long = 0 )
```

Mengembalikan `DynamicFrame` baru dengan semua kolom nol dihapus.

> **Note**
>
> Ini hanya menghapus kolom tipe `NullType`. Nilai-nilai nol individu dalam kolom lain tidak dihapus atau dimodifikasi.

Bingkai Def errorsAsDynamic

```
def errorsAsDynamicFrame
```

Mengembalikan `DynamicFrame` baru yang berisi catatan kesalahan dari `DynamicFrame`.

Filter Def

```
def filter( f : DynamicRecord => Boolean,
            errorMsg : String = "",
            transformationContext : String = "",
            callSite : CallSite = CallSite("Not provided"),
            stageThreshold : Long = 0,
```

```
            totalThreshold : Long = 0
          ) : DynamicFrame
```

Membangun `DynamicFrame` baru yang hanya berisi catatan-catatan yang untuknya fungsi 'f' mengembalikan `true`. Fungsi filter 'f' seharusnya tidak mengubah catatan masukan.

Def getName

```
def getName : String
```

Mengembalikan nama dari `DynamicFrame` ini.

Def getNumPartitions

```
def getNumPartitions
```

Mengembalikan jumlah partisi dalam `DynamicFrame` ini.

Def Dihitung getSchemaIf

```
def getSchemaIfComputed : Option[Schema]
```

Mengembalikan skema jika ia sudah dikomputasi. Tidak memindai data jika skema belum dikomputasi.

Def isSchemaComputed

```
def isSchemaComputed : Boolean
```

Mengembalikan `true` jika skema telah dikomputasi untuk `DynamicFrame` ini, atau `false` jika tidak. Jika metode ini mengembalikan false, maka memanggil metode `schema` akan mengharuskan pemberian lain atas catatan dalam `DynamicFrame` ini.

Def javaToPython

```
def javaToPython : JavaRDD[Array[Byte]]
```

Def bergabung

```
def join( keys1 : Seq[String],
```

```
          keys2 : Seq[String],
          frame2 : DynamicFrame,
          transformationContext : String = "",
          callSite : CallSite = CallSite("Not provided", ""),
          stageThreshold : Long = 0,
          totalThreshold : Long = 0
        ) : DynamicFrame
```

- `keys1` — Kolom dalam `DynamicFrame` ini yang akan digunakan untuk penggabungan.
- `keys2` — Kolom dalam `frame2` yang akan digunakan untuk penggabungan. Harus memiliki panjang yang sama seperti `keys1`.
- `frame2` — `DynamicFrame` yang akan digabungkan padanya.

Mengembalikan hasil dari pelaksanaan equijoin dengan `frame2` menggunakan kunci yang ditentukan.

Def peta

```
def map( f : DynamicRecord => DynamicRecord,
         errorMsg : String = "",
         transformationContext : String = "",
         callSite : CallSite = CallSite("Not provided", ""),
         stageThreshold : Long = 0,
         totalThreshold : Long = 0
       ) : DynamicFrame
```

Mengembalikan `DynamicFrame` baru yang dibangun dengan menerapkan fungsi 'f' tertentu untuk setiap catatan dalam `DynamicFrame`.

Metode ini menyalin setiap catatan sebelum menerapkan fungsi yang ditentukan, sehingga aman untuk mengubah catatan. Jika fungsi pemetaan melempar pengecualian pada catatan tertentu, maka catatan yang ditandai sebagai kesalahan, dan jejak tumpukan disimpan sebagai sebuah kolom dalam catatan kesalahan.

Def mergeDynamicFrames

```
def mergeDynamicFrames( stageDynamicFrame: DynamicFrame,  primaryKeys: Seq[String],
 transformationContext: String = "",
                         options: JsonOptions = JsonOptions.empty, callSite: CallSite =
 CallSite("Not provided"),
```

```
                        stageThreshold: Long = 0, totalThreshold: Long = 0):
DynamicFrame
```

- `stageDynamicFrame` — Pentahapan `DynamicFrame` yang akan digabungkan.
- `primaryKeys` — Daftar bidang kunci primer untuk mencocokkan catatan dari sumber dan pentahapan `DynamicFrame`.
- `transformationContext` — Sebuah string unik yang digunakan untuk mengambil metadata tentang transformasi saat ini (opsional).
- `options` — Sebuah string pasangan nama-nilai JSON yang memberikan informasi tambahan untuk transformasi ini.
- `callSite` — Digunakan untuk menyediakan informasi konteks untuk pelaporan kesalahan.
- `stageThreshold` — Sebuah `Long`. Jumlah kesalahan dalam transformasi yang ditentukan yang memerlukan pengolahan untuk membersihkan kesalahan.
- `totalThreshold` — Sebuah `Long`. Jumlah kesalahan hingga dan termasuk dalam transformasi yang memerlukan pengolahan untuk membersihkan kesalahan.

Menggabungkan `DynamicFrame` ini dengan pentahapan `DynamicFrame` berdasarkan kunci primer yang ditentukan untuk mengidentifikasi catatan. Catatan duplikat (catatan dengan kunci primer yang sama) tidak di-deduplikasi. Jika tidak ada catatan yang cocok dalam bingkai pentahapan, semua catatan (termasuk duplikat) akan dipertahankan dari sumber. Jika bingkai pementasan memiliki catatan yang cocok, catatan dari bingkai pementasan menimpa catatan di sumber. AWS Glue

`DynamicFrame` yang dikembalikan berisi catatan A dalam kasus berikut:

1. Jika `A` ada di bingkai sumber dan bingkai pentahapan, maka `A` dalam bingkai pentahapan akan dikembalikan.
2. Jika `A` ada dalam tabel sumber dan `A.primaryKeys` tidak ada di `stagingDynamicFrame` (berarti `A` tidak diperbarui dalam tabel pentahapan).

Bingkai sumber dan bingkai pentahapan tidak perlu memiliki skema yang sama.

Example

```
val mergedFrame: DynamicFrame = srcFrame.mergeDynamicFrames(stageFrame, Seq("id1",
 "id2"))
```

PrintSkema Cetak Def

```
def printSchema : Unit
```

Mencetak skema `DynamicFrame` ini ke `stdout` dalam format yang dapat dibaca manusia.

ReComputesChema Def

```
def recomputeSchema : Schema
```

Memaksakan komputasi ulang skema. Hal ini memerlukan pemindaian data, tapi mungkin akan "mengencangkan" skema jika ada beberapa bidang dalam skema saat ini yang tidak ada dalam data.

Mengembalikan skema yang telah dikomputasi ulang.

Def relasialisasi

```
def relationalize( rootTableName : String,
                   stagingPath : String,
                   options : JsonOptions = JsonOptions.empty,
                   transformationContext : String = "",
                   callSite : CallSite = CallSite("Not provided"),
                   stageThreshold : Long = 0,
                   totalThreshold : Long = 0
                 ) : Seq[DynamicFrame]
```

- `rootTableName` — Nama yang digunakan untuk `DynamicFrame` dasar dalam output. `DynamicFrame` yang dibuat oleh array berputar dimulai dengan ini sebagai prefiks.
- `stagingPath` — Path Amazon Simple Storage Service (Amazon S3) untuk menulis data menengah.
- `options` — Menghubungkan opsi dan konfigurasi. Saat ini tidak digunakan.

Meratakan semua struktur bersarang dan memutar array ke dalam tabel terpisah.

Anda dapat menggunakan operasi ini untuk mempersiapkan data yang sangat bersarang untuk penyerapan ke dalam basis data relasional. Struct bersarang diratakan dengan cara yang sama seperti transformasi Tidak bersarang. Selain itu, array yang diputar ke dalam tabel terpisah dengan masing-masing elemen array yang menjadi sebuah baris. Sebagai contoh, anggaplah Anda memiliki `DynamicFrame` dengan data berikut.

```
 {"name": "Nancy", "age": 47, "friends": ["Fred", "Lakshmi"]}
 {"name": "Stephanie", "age": 28, "friends": ["Yao", "Phil", "Alvin"]}
 {"name": "Nathan", "age": 54, "friends": ["Nicolai", "Karen"]}
```

Jalankan kode berikut.

```
{{{
  df.relationalize("people", "s3:/my_bucket/my_path", JsonOptions.empty)
}}}
```

Hal ini menghasilkan dua tabel. Tabel pertama bernama "orang" dan isinya adalah sebagai berikut.

```
{{{
  {"name": "Nancy", "age": 47, "friends": 1}
  {"name": "Stephanie", "age": 28, "friends": 2}
  {"name": "Nathan", "age": 54, "friends": 3)
}}}
```

Di sini, array teman-teman telah diganti dengan kunci penggabungan yang dihasilkan secara otomatis. Sebuah tabel terpisah bernama `people.friends` dibuat dengan isi sebagai berikut.

```
{{{
  {"id": 1, "index": 0, "val": "Fred"}
  {"id": 1, "index": 1, "val": "Lakshmi"}
  {"id": 2, "index": 0, "val": "Yao"}
  {"id": 2, "index": 1, "val": "Phil"}
  {"id": 2, "index": 2, "val": "Alvin"}
  {"id": 3, "index": 0, "val": "Nicolai"}
  {"id": 3, "index": 1, "val": "Karen"}
}}}
```

Dalam tabel ini, '`id`' adalah sebuah kunci penggabungan yang mengidentifikasi yang mencatat asal elemen array, '`index`' mengacu pada posisi dalam array asli, dan '`val`' adalah entri array yang sebenarnya.

Metode `relationalize` mengembalikan deret `DynamicFrame` yang dibuat dengan menerapkan proses ini secara rekursif pada semua array.

> **Note**
>
> AWS GluePustaka secara otomatis menghasilkan kunci gabungan untuk tabel baru. Untuk memastikan bahwa kunci penggabungan bersifat unik di seluruh eksekusi tugas, Anda harus mengaktifkan bookmark tugas.

Def RenameField

```
def renameField( oldName : String,
                 newName : String,
                 transformationContext : String = "",
                 callSite : CallSite = CallSite("Not provided", ""),
                 stageThreshold : Long = 0,
                 totalThreshold : Long = 0
               ) : DynamicFrame
```

- `oldName` — Nama asli kolom.
- `newName` — Nama baru kolom.

Mengembalikan sebuah `DynamicFrame` baru dengan kolom tertentu yang diganti namanya.

Anda dapat menggunakan metode ini untuk mengganti nama bidang bersarang. Misalnya, kode berikut akan mengubah nama `state` menjadi `state_code` dalam struct alamat.

```
{{{
  df.renameField("address.state", "address.state_code")
}}}
```

Def repartisi

```
def repartition( numPartitions : Int,
                 transformationContext : String = "",
                 callSite : CallSite = CallSite("Not provided", ""),
                 stageThreshold : Long = 0,
                 totalThreshold : Long = 0
               ) : DynamicFrame
```

Mengembalikan sebuah `DynamicFrame` baru dengan partisi `numPartitions`.

Def ResolveChoice

```
def resolveChoice( specs : Seq[Product2[String, String]] = Seq.empty[ResolveSpec],
                   choiceOption : Option[ChoiceOption] = None,
                   database : Option[String] = None,
                   tableName : Option[String] = None,
                   transformationContext : String = "",
                   callSite : CallSite = CallSite("Not provided", ""),
                   stageThreshold : Long = 0,
                   totalThreshold : Long = 0
                 ) : DynamicFrame
```

- `choiceOption` — Sebuah tindakan yang akan diterapkan ke semua kolom `ChoiceType` yang tidak tercantum dalam urutan spesifikasi.
- `database` — Basis data Katalog Data yang akan digunakan dengan tindakan `match_catalog`.
- `tableName` — Tabel Katalog Data yang akan digunakan dengan tindakan `match_catalog`.

Mengembalikan sebuah `DynamicFrame` baru dengan mengganti satu atau beberapa `ChoiceType` dengan tipe yang lebih spesifik.

Ada dua cara untuk menggunakan `resolveChoice`. Yang pertama adalah menentukan urutan kolom tertentu dan cara mengubahnya. Hal ini ditentukan sebagai tupel yang terdiri dari pasangan (kolom, tindakan).

Berikut ini adalah tindakan yang mungkin:

- `cast:type` — Upaya untuk mengubah semua nilai ke jenis tertentu.
- `make_cols` — Mengkonversi setiap jenis yang berbeda menjadi kolom dengan nama `columnName_type`.
- `make_struct` — Mengkonversi kolom menjadi struct dengan kunci untuk setiap jenis yang berbeda.
- `project:type` — Mempertahankan hanya nilai-nilai dari jenis tertentu saja.

Mode lain untuk `resolveChoice` adalah untuk menentukan resolusi tunggal untuk semua `ChoiceType`. Anda dapat menggunakan ini dalam kasus di mana daftar `ChoiceType` lengkap tidak diketahui sebelum eksekusi. Selain tindakan-tindakan yang tercantum sebelumnya, mode ini juga mendukung tindakan berikut:

- `match_catalog` — Upaya untuk mengubah setiap `ChoiceType` menjadi jenis yang sesuai dalam tabel katalog yang ditentukan.

Contoh:

Ubah kolom `user.id` dengan mengubahnya menjadi int, dan membuat bidang `address` hanya mempertahankan struct saja.

```
{{{
  df.resolveChoice(specs = Seq(("user.id", "cast:int"), ("address", "project:struct")))
}}}
```

Ubah semua `ChoiceType` dengan mengkonversi setiap pilihan menjadi kolom terpisah.

```
{{{
  df.resolveChoice(choiceOption = Some(ChoiceOption("make_cols")))
}}}
```

Ubah semua `ChoiceType` dengan mengubahnya menjadi jenis dalam tabel katalog yang ditentukan.

```
{{{
  df.resolveChoice(choiceOption = Some(ChoiceOption("match_catalog")),
                   database = Some("my_database"),
                   tableName = Some("my_table"))
}}}
```

Skema Def

```
def schema : Schema
```

Mengembalikan skema `DynamicFrame` ini.

Skema yang dikembalikan dijamin mengandung setiap bidang yang ada dalam catatan di `DynamicFrame` ini. Namun dalam sejumlah kecil kasus, di mana skema mungkin juga berisi bidang tambahan. Anda dapat menggunakan metode Tidak bersarang untuk "mengencangkan" skema berdasarkan catatan dalam `DynamicFrame`.

SelectField Def

```
def selectField( fieldName : String,
```

```
                transformationContext : String = "",
                callSite : CallSite = CallSite("Not provided", ""),
                stageThreshold : Long = 0,
                totalThreshold : Long = 0
              ) : DynamicFrame
```

Mengembalikan satu bidang sebagai sebuah `DynamicFrame`.

SelectFields Def

```
def selectFields( paths : Seq[String],
                  transformationContext : String = "",
                  callSite : CallSite = CallSite("Not provided", ""),
                  stageThreshold : Long = 0,
                  totalThreshold : Long = 0
                ) : DynamicFrame
```

- `paths` — Urutan nama kolom yang akan dipilih.

Mengembalikan sebuah `DynamicFrame` baru yang berisi kolom yang ditentukan.

> **Note**
>
> Anda hanya dapat menggunakan metode `selectFields` untuk memilih kolom tingkat atas. Anda dapat menggunakan metode applyMapping untuk memilih kolom bersarang.

Pertunjukan def

```
def show( numRows : Int = 20 ) : Unit
```

- numRows — Jumlah baris yang akan dicetak.

Mencetak baris dari `DynamicFrame` ini dalam format JSON.

Def SederhanaDDBJSON

Ekspor DynamoDB dengan AWS Glue konektor ekspor DynamoDB menghasilkan file JSON dari struktur bersarang tertentu. Untuk informasi selengkapnya, lihat Objek data. `simplifyDDBJson`

Menyederhanakan kolom bersarang dalam jenis data ini, dan mengembalikan yang baru disederhanakan. DynamicFrame DynamicFrame Jika ada beberapa jenis atau tipe Peta yang terdapat dalam tipe Daftar, elemen dalam Daftar tidak akan disederhanakan. Metode ini hanya mendukung data dalam format JSON ekspor DynamoDB. Pertimbangkan `unnest` untuk melakukan perubahan serupa pada jenis data lainnya.

```
def simplifyDDBJson() : DynamicFrame
```

Metode ini tidak mengambil parameter apa pun.

Contoh masukan

Pertimbangkan skema berikut yang dihasilkan oleh ekspor DynamoDB:

```
root
|-- Item: struct
|    |-- parentMap: struct
|    |    |-- M: struct
|    |    |    |-- childMap: struct
|    |    |    |    |-- M: struct
|    |    |    |    |    |-- appName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- packageName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- updatedAt: struct
|    |    |    |    |    |    |-- N: string
|    |-- strings: struct
|    |    |-- SS: array
|    |    |    |-- element: string
|    |-- numbers: struct
|    |    |-- NS: array
|    |    |    |-- element: string
|    |-- binaries: struct
|    |    |-- BS: array
|    |    |    |-- element: string
|    |-- isDDBJson: struct
|    |    |-- BOOL: boolean
|    |-- nullValue: struct
|    |    |-- NULL: boolean
```

Contoh kode

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContextimport scala.collection.JavaConverters._

object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.export" -> "ddb",
        "dynamodb.tableArn" -> "ddbTableARN",
        "dynamodb.s3.bucket" -> "exportBucketLocation",
        "dynamodb.s3.prefix" -> "exportBucketPrefix",
        "dynamodb.s3.bucketOwner" -> "exportBucketAccountID",
      ))
    ).getDynamicFrame()

    val simplified = dynamicFrame.simplifyDDBJson()
    simplified.printSchema()

    Job.commit()
  }

}
```

Contoh Output

`simplifyDDBJson`Transformasi akan menyederhanakan ini menjadi:

```
root
|-- parentMap: struct
|    |-- childMap: struct
|    |    |-- appName: string
|    |    |-- packageName: string
|    |    |-- updatedAt: string
```

```
|-- strings: array
|    |-- element: string
|-- numbers: array
|    |-- element: string
|-- binaries: array
|    |-- element: string
|-- isDDBJson: boolean
|-- nullValue: null
```

Keran Def

```
def spigot( path : String,
            options : JsonOptions = new JsonOptions("{}"),
            transformationContext : String = "",
            callSite : CallSite = CallSite("Not provided"),
            stageThreshold : Long = 0,
            totalThreshold : Long = 0
          ) : DynamicFrame
```

Transformasi passthrough yang mengembalikan catatan yang sama tetapi menulis subset dari catatan sebagai efek samping.

- `path` — Path di Amazon S3 untuk menulis output, dalam bentuk `s3://bucket//path`.
- `options` — Peta `JsonOptions` opsional yang menjelaskan perilaku pengambilan sampel.

Mengembalikan sebuah `DynamicFrame` yang berisi catatan yang sama seperti yang satu ini.

Secara default, menulis 100 catatan yang berubah-ubah ke lokasi yang ditentukan oleh `path`. Anda dapat menyesuaikan perilaku ini dengan menggunakan peta `options`. Kunci yang valid meliputi yang berikut ini:

- `topk` — Menentukan jumlah total catatan yang ditulis. Secara default, nilainya adalah 100.
- `prob` — Menentukan probabilitas (dalam desimal) bahwa catatan individu sudah disertakan. Default-nya adalah 1.

Misalnya, panggilan berikut akan mengambil sampel dari set data dengan memilih setiap catatan dengan probabilitas 20 persen dan berhenti setelah 200 catatan telah ditulis.

```
{{{
```

```
  df.spigot("s3://my_bucket/my_path", JsonOptions(Map("topk" -&gt; 200, "prob" -&gt;
 0.2)))
}}}
```

Def SplitFields

```
def splitFields( paths : Seq[String],
                 transformationContext : String = "",
                 callSite : CallSite = CallSite("Not provided", ""),
                 stageThreshold : Long = 0,
                 totalThreshold : Long = 0
               ) : Seq[DynamicFrame]
```

- `paths` — Path yang akan disertakan dalam `DynamicFrame` pertama.

Mengembalikan deret dua `DynamicFrame`. `DynamicFrame` yang pertama berisi path yang ditentukan, dan yang kedua berisi semua kolom lainnya.

Contoh

Contoh ini mengambil DynamicFrame dibuat dari `persons` tabel dalam `legislators` database di Katalog Data AWS Glue dan membagi DynamicFrame menjadi dua, dengan bidang yang ditentukan masuk ke bidang pertama DynamicFrame dan yang tersisa menjadi yang kedua DynamicFrame. Contoh kemudian memilih yang pertama DynamicFrame dari hasilnya.

```
val InputFrame = glueContext.getCatalogSource(database="legislators",
 tableName="persons",
transformationContext="InputFrame").getDynamicFrame()

val SplitField_collection = InputFrame.splitFields(paths=Seq("family_name", "name",
 "links.note",
"links.url", "gender", "image", "identifiers.scheme", "identifiers.identifier",
 "other_names.lang",
"other_names.note", "other_names.name"), transformationContext="SplitField_collection")

val ResultFrame = SplitField_collection(0)
```

Def SplitRows

```
def splitRows( paths : Seq[String],
```

```
    values : Seq[Any],
    operators : Seq[String],
    transformationContext : String,
    callSite : CallSite,
    stageThreshold : Long,
    totalThreshold : Long
  ) : Seq[DynamicFrame]
```

Membagi baris berdasarkan predikat yang membandingkan kolom dengan konstanta.

- `paths` — Kolom yang digunakan untuk perbandingan.
- `values` — Nilai konstanta yang digunakan untuk perbandingan.
- `operators` — Operator yang digunakan untuk perbandingan.

Mengembalikan deret dua `DynamicFrame`. Yang pertama berisi baris dengan predikat true yang kedua berisi baris dengan predikat false.

Predikat ditentukan dengan menggunakan tiga urutan: '`paths`' berisi nama kolom (mungkin bersarang), '`values`' berisi nilai-nilai konstanta yang akan dibandingkan, dan '`operators`' berisi operator yang akan digunakan untuk perbandingan. Ketiga urutan harus sama panjangnya: operator ke-n digunakan untuk membandingkan kolom ke-n dengan nilai ke-n.

Setiap operator harus berupa salah satu dari "`!`=", "=", "<=", "<", ">=", atau ">".

Sebagai contoh, panggilan berikut akan membagi `DynamicFrame` sehingga bingkai output pertama akan berisi catatan dari orang-orang yang berusia di atas 65 dari Amerika Serikat, dan yang kedua akan berisi semua catatan lainnya.

```
{{{
  df.splitRows(Seq("age", "address.country"), Seq(65, "USA"), Seq("&gt;=", "="))
}}}
```

Def stageErrorsCount

```
def stageErrorsCount
```

Mengembalikan jumlah catatan kesalahan yang dibuat saat melakukan komputasi pada `DynamicFrame` ini. Ini tidak termasuk kesalahan dari operasi sebelumnya yang dilewatkan ke `DynamicFrame` ini sebagai masukan.

Def ToDF

```
def toDF( specs : Seq[ResolveSpec] = Seq.empty[ResolveSpec] ) : DataFrame
```

Mengkonversi `DynamicFrame` ini ke `DataFrame` Apache Spark SQL dengan skema dan catatan yang sama.

> **Note**
>
> Karena `DataFrame` tidak mendukung `ChoiceType`, jadi metode ini secara otomatis mengkonversi kolom `ChoiceType` menjadi `StructType`. Untuk informasi selengkapnya dan opsi pengubahan pilihan, lihat resolveChoice.

Def membuka kotak

```
def unbox( path : String,
           format : String,
           optionString : String = "{}",
           transformationContext : String = "",
           callSite : CallSite = CallSite("Not provided"),
           stageThreshold : Long = 0,
           totalThreshold : Long = 0
         ) : DynamicFrame
```

- `path` — Kolom yang akan diurai. Harus berupa string atau biner.
- `format` — Format yang akan digunakan untuk penguraian.
- `optionString` — Pilihan untuk memberikan ke format, seperti pemisah CSV.

Mengurai string atau kolom biner yang tertanam sesuai dengan format yang ditentukan. Kolom yang diurai disarangkan di bawah struct dengan nama kolom asli.

Misalnya, anggaplah Anda memiliki file CSV dengan kolom JSON yang tertanam.

```
name, age, address
Sally, 36, {"state": "NE", "city": "Omaha"}
...
```

Setelah penguraian awal, Anda akan mendapatkan `DynamicFrame` dengan skema berikut.

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address: string
}}}
```

Anda dapat memanggil unbox pada kolom alamat untuk mengurai komponen tertentu.

```
{{{
  df.unbox("address", "json")
}}}
```

hal ini akan memberi kita sebuah DynamicFrame dengan skema berikut.

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address: struct
  |    |-- state: string
  |    |-- city: string
}}}
```

Def tidak bersarang

```
def unnest( transformationContext : String = "",
            callSite : CallSite = CallSite("Not Provided"),
            stageThreshold : Long = 0,
            totalThreshold : Long = 0
          ) : DynamicFrame
```

Mengembalikan sebuah DynamicFrame baru dengan semua struktur bersarang yang sudah diratakan. Nama dibangun menggunakan karakter '.' (titik).

Sebagai contoh, anggaplah Anda memiliki DynamicFrame dengan skema berikut.

```
{{{
  root
  |-- name: string
```

```
  |-- age: int
  |-- address: struct
  |    |-- state: string
  |    |-- city: string
}}}
```

Panggilan berikut membuka sarang struct alamat.

```
{{{
  df.unnest()
}}}
```

Skema yang dihasilkan adalah sebagai berikut.

```
{{{
  root
  |-- name: string
  |-- age: int
  |-- address.state: string
  |-- address.city: string
}}}
```

Metode ini juga akan membuka struct bersarang dalam array. Tetapi karena alasan riwayat, nama-nama bidang tersebut didahului dengan nama array yang dilampirkan dan "`.val`".

Def Unnestddbjson

```
unnestDDBJson(transformationContext : String = "",
            callSite : CallSite = CallSite("Not Provided"),
            stageThreshold : Long = 0,
            totalThreshold : Long = 0): DynamicFrame
```

Unnests kolom bersarang di a `DynamicFrame` yang secara khusus dalam struktur DynamoDB JSON, dan mengembalikan unnested baru. `DynamicFrame` Kolom yang terdiri dari array tipe struct tidak akan di-unnested. Perhatikan bahwa ini adalah jenis transformasi unnesting tertentu yang berperilaku berbeda dari `unnest` transformasi biasa dan mengharuskan data sudah berada dalam struktur DynamoDB JSON. Untuk informasi selengkapnya, lihat DynamoDB JSON.

Misalnya, skema pembacaan ekspor dengan struktur DynamoDB JSON mungkin terlihat seperti berikut:

```
root
|-- Item: struct
|    |-- ColA: struct
|    |    |-- S: string
|    |-- ColB: struct
|    |    |-- S: string
|    |-- ColC: struct
|    |    |-- N: string
|    |-- ColD: struct
|    |    |-- L: array
|    |    |    |-- element: null
```

`unnestDDBJson()`Transformasi akan mengubah ini menjadi:

```
root
|-- ColA: string
|-- ColB: string
|-- ColC: string
|-- ColD: array
|    |-- element: null
```

Contoh kode berikut menunjukkan cara menggunakan konektor ekspor AWS Glue DynamoDB, memanggil DynamoDB JSON unnest, dan mencetak jumlah partisi:

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType = "dynamodb",
```

```
      options = JsonOptions(Map(
        "dynamodb.export" -> "ddb",
        "dynamodb.tableArn" -> "<test_source>",
        "dynamodb.s3.bucket" -> "<bucket name>",
        "dynamodb.s3.prefix" -> "<bucket prefix>",
        "dynamodb.s3.bucketOwner" -> "<account_id of bucket>",
      ))
    ).getDynamicFrame()

    val unnested = dynamicFrame.unnestDDBJson()
    print(unnested.getNumPartitions())

    Job.commit()
  }

}
```

Def withFrameSchema

```
def withFrameSchema( getSchema : () => Schema ) : DynamicFrame
```

- `getSchema` — Sebuah fungsi yang mengembalikan skema yang akan digunakan. Ditentukan sebagai fungsi nol-parameter untuk menunda komputasi yang berpotensi mahal.

Menetapkan skema dari `DynamicFrame` ini dengan nilai yang ditentukan. Hal ini terutama digunakan secara internal untuk menghindari komputasi ulang yang mahal pada skema. Skema yang dimasukkan harus berisi semua kolom yang ada dalam data.

Def withName

```
def withName( name : String ) : DynamicFrame
```

- `name` — Nama baru yang akan digunakan.

Mengembalikan salinan dari `DynamicFrame` ini dengan nama baru.

Def withTransformationContext

```
def withTransformationContext( ctx : String ) : DynamicFrame
```

Mengembalikan salinan dari `DynamicFrame` ini dengan konteks transformasi yang ditentukan.

DynamicFrame Objeknya

Package: com.amazonaws.services.glue

```
object DynamicFrame
```

Def berlaku

```
def apply( df : DataFrame,
           glueContext : GlueContext
         ) : DynamicFrame
```

Def emptyDynamicFrame

```
def emptyDynamicFrame( glueContext : GlueContext ) : DynamicFrame
```

Def dariPythonRDD

```
def fromPythonRDD( rdd : JavaRDD[Array[Byte]],
                   glueContext : GlueContext
                 ) : DynamicFrame
```

Def IgnoreErrors

```
def ignoreErrors( fn : DynamicRecord => DynamicRecord ) : DynamicRecord
```

Def InlineErrors

```
def inlineErrors( msg : String,
                  callSite : CallSite
                ) : (DynamicRecord => DynamicRecord)
```

Kesalahan Def newFrameWith

```
def newFrameWithErrors( prevFrame : DynamicFrame,
```

```
                        rdd : RDD[DynamicRecord],
                        name : String = "",
                        transformationContext : String = "",
                        callSite : CallSite,
                        stageThreshold : Long,
                        totalThreshold : Long
                      ) : DynamicFrame
```

AWS GlueKelas scala DynamicRecord

Topik

- Def AddField
- DropField Def
- Def SetError
- Def isError
- Def GetError
- Def ClearError
- Def menulis
- Def ReadFields
- Klon def
- Skema Def
- Def GetRoot
- Def ToJSON
- Def getFieldNode
- Def GetField
- Kode Hash Def
- Def sama dengan
- DynamicRecord objek
- RecordTraverser sifat

Package: com.amazonaws.services.glue

```
class DynamicRecord extends Serializable with Writable with Cloneable
```

Sebuah `DynamicRecord` adalah struktur data self-describing yang mewakili deretan data dalam set data yang sedang diproses. Ia bersifat self-describing dalam arti bahwa Anda bisa mendapatkan skema dari baris yang diwakili oleh `DynamicRecord` dengan memeriksa catatan itu sendiri. Sebuah `DynamicRecord` mirip dengan `Row` di Apache Spark.

Def AddField

```
def addField( path : String,
              dynamicNode : DynamicNode
            ) : Unit
```

Menambahkan sebuah DynamicNode ke path yang ditentukan.

- `path` — Path untuk bidang yang akan ditambahkan.
- `dynamicNode` — DynamicNode yang akan ditambahkan pada path yang ditentukan.

DropField Def

```
def dropField(path: String, underRename: Boolean = false): Option[DynamicNode]
```

Menghapus DynamicNode dari path yang ditentukan dan mengembalikan simpul yang dibuang jika tidak ada array di path yang ditentukan.

- `path` — Path ke bidang yang akan dibuang.
- `underRename` — BETUL jika `dropField` dipanggil sebagai bagian dari penggantian nama transformasi, atau SALAH jika sebaliknya (secara default SALAH).

Mengembalikan `scala.Option Option` (DynamicNode).

Def SetError

```
def setError( error : Error )
```

Menetapkan catatan ini sebagai catatan kesalahan, sebagaimana yang ditentukan oleh parameter `error`.

Mengembalikan `DynamicRecord`.

Def isError

```
def isError
```

Memeriksa apakah catatan ini adalah catatan kesalahan.

Def GetError

```
def getError
```

Mengambil `Error` jika catatan adalah catatan kesalahan. Mengembalikan `scala.Some` Some (Kesalahan) jika catatan ini adalah catatan kesalahan, atau sebaliknya `scala.None`.

Def ClearError

```
def clearError
```

Atur `Error` ke `scala.None.None` .

Def menulis

```
override def write( out : DataOutput ) : Unit
```

Def ReadFields

```
override def readFields( in : DataInput ) : Unit
```

Klon def

```
override def clone : DynamicRecord
```

Kloning catatan ini ke `DynamicRecord` yang baru dan kembalikan.

Skema Def

```
def schema
```

Ambil `Schema` dengan memeriksa catatan.

Def GetRoot

```
def getRoot : ObjectNode
```

Ambil `ObjectNode` akar untuk catatan.

Def ToJSON

```
def toJson : String
```

Ambil string JSON untuk catatan.

Def getFieldNode

```
def getFieldNode( path : String ) : Option[DynamicNode]
```

Dapatkan nilai bidang di `path` yang ditentukan sebagai pilihan `DynamicNode`.

Mengembalikan `scala.Some` Some (DynamicNode) jika bidang ada, atau sebaliknya `scala.None.None`.

Def GetField

```
def getField( path : String ) : Option[Any]
```

Dapatkan nilai bidang di `path` yang ditentukan sebagai pilihan `DynamicNode`.

Mengembalikan `scala.Some` Some (nilai).

Kode Hash Def

```
override def hashCode : Int
```

Def sama dengan

```
override def equals( other : Any )
```

DynamicRecord objek

```
object DynamicRecord
```

Def berlaku

```
def apply( row : Row,
           schema : SparkStructType )
```

Terapkan metode untuk mengkonversi Apache Spark SQL Row menjadi DynamicRecord.

- row — Sebuah Spark SQL Row.
- schema — Schema dari baris tersebut.

Mengembalikan DynamicRecord.

RecordTraverser sifat

```
trait RecordTraverser {
  def nullValue(): Unit
  def byteValue(value: Byte): Unit
  def binaryValue(value: Array[Byte]): Unit
  def booleanValue(value: Boolean): Unit
  def shortValue(value: Short) : Unit
  def intValue(value: Int) : Unit
  def longValue(value: Long) : Unit
  def floatValue(value: Float): Unit
  def doubleValue(value: Double): Unit
  def decimalValue(value: BigDecimal): Unit
  def stringValue(value: String): Unit
  def dateValue(value: Date): Unit
  def timestampValue(value: Timestamp): Unit
  def objectStart(length: Int): Unit
  def objectKey(key: String): Unit
  def objectEnd(): Unit
  def mapStart(length: Int): Unit
  def mapKey(key: String): Unit
  def mapEnd(): Unit
  def arrayStart(length: Int): Unit
  def arrayEnd(): Unit
}
```

AWS GlueAPI Scala GlueContext

Package: com.amazonaws.services.glue

```
class GlueContext extends SQLContext(sc) (
          @transient val sc : SparkContext,
          val defaultSourcePartitioner : PartitioningStrategy )
```

`GlueContext` adalah titik masuk untuk membaca dan menulis DynamicFrame dari dan ke Amazon Simple Storage Service (Amazon S3), Katalog Data Glue AWS , JDBC, dan sebagainya. Kelas ini menyediakan fungsi utilitas untuk membuat objek DataSource sifat dan DataSink yang pada gilirannya dapat digunakan untuk membaca dan menulis `DynamicFrame`.

Anda juga dapat menggunakan `GlueContext` untuk menetapkan target jumlah partisi (default 20) di `DynamicFrame` jika jumlah partisi yang dibuat dari sumber kurang dari ambang batas minimum untuk partisi (default 10).

Kolom def addIngestionTime

```
def addIngestionTimeColumns(
         df : DataFrame,
         timeGranularity : String = "") : dataFrame
```

Menambahkan kolom waktu penyerapan seperti `ingest_year`, `ingest_month`, `ingest_day`, `ingest_hour`, `ingest_minute` ke input `DataFrame`. Fungsi ini secara otomatis dihasilkan dalam skrip yang dihasilkan oleh AWS Glue saat Anda menentukan tabel Katalog Data dengan Amazon S3 sebagai target. Fungsi ini secara otomatis memperbarui partisi dengan kolom waktu penyerapan pada tabel output. Hal ini memungkinkan data output dipartisi secara otomatis pada waktu penyerapan tanpa memerlukan kolom waktu penyerapan eksplisit dalam data input.

- `dataFrame` — `dataFrame` yang akan ditambahi dengan kolom waktu penyerapan.
- `timeGranularity` — Kedetailan dari kolom waktu. Nilai yang benar adalah "`day`", "`hour`" dan "`minute`". Misalnya, jika "`hour`" diberikan dalam fungsi, maka `dataFrame` asli akan memiliki kolom waktu "`ingest_year`", "`ingest_month`", "`ingest_day`", dan "`ingest_hour`" yang ditambahkan.

Mengembalikan bingkai data setelah menambahkan kolom kedetailan waktu.

Contoh:

```
glueContext.addIngestionTimeColumns(dataFrame, "hour")
```

def createDataFrame FromOptions

```
def createDataFrameFromOptions( connectionType : String,
                          connectionOptions : JsonOptions,
                          transformationContext : String = "",
                          format : String = null,
                          formatOptions : JsonOptions = JsonOptions.empty
                        ) : DataSource
```

Mengembalikan sebuah `DataFrame` dibuat dengan koneksi dan format yang ditentukan. Gunakan fungsi ini hanya dengan sumber streaming AWS Glue.

- `connectionType`— Jenis koneksi streaming. Nilai yang valid mencakup `kinesis` dan `kafka`.
- `connectionOptions`— Opsi koneksi, yang berbeda untuk Kinesis dan Kafka. Anda dapat menemukan daftar semua opsi koneksi untuk setiap sumber data streaming di Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark. Perhatikan perbedaan berikut dalam pilihan koneksi streaming:
  - Sumber streaming Kinesis memerlukan `streamARN`, `startingPosition`, `inferSchema`, dan `classification`.
  - Sumber streaming Kafka membutuhkan `connectionName`, `topicName`, `startingOffsets`, `inferSchema`, dan `classification`.
- `transformationContext`— Konteks transformasi yang akan digunakan (opsional).
- `format`- Spesifikasi format (opsional). Ini digunakan untuk Amazon S3 atau AWS Glue koneksi yang mendukung berbagai format. Untuk informasi tentang format yang didukung, lihat Opsi format data untuk input dan output untuk Spark AWS Glue
- `formatOptions`— Opsi format untuk format yang ditentukan. Untuk informasi tentang pilihan format yang didukung, lihat Opsi format data.

Contoh untuk sumber streaming Amazon Kinesis:

```
val data_frame_datasource0 =
glueContext.createDataFrameFromOptions(transformationContext = "datasource0",
 connectionType = "kinesis",
connectionOptions = JsonOptions("""{"streamName": "example_stream", "startingPosition":
 "TRIM_HORIZON", "inferSchema": "true", "classification": "json"}}""""))
```

Contoh untuk sumber streaming Kafka:

```
val data_frame_datasource0 =
glueContext.createDataFrameFromOptions(transformationContext = "datasource0",
 connectionType = "kafka",
connectionOptions = JsonOptions("""{"connectionName": "example_connection",
 "topicName": "example_topic", "startingPosition": "earliest", "inferSchema": "false",
 "classification": "json", "schema":"`column1` STRING, `column2` STRING"}""" ))
```

forEachBatch

`forEachBatch(frame, batch_function, options)`

Menerapkan `batch_function` yang diberikan ke setiap batch mikro yang dibaca dari sumber Streaming.

- `frame`— Yang DataFrame berisi batch mikro saat ini.
- `batch_function` — Sebuah fungsi yang akan diterapkan untuk setiap batch mikro.
- `options` — Kumpulan pasangan kunci-nilai yang menyimpan informasi tentang cara memproses batch mikro. Opsi-opsi berikut diperlukan:
  - `windowSize` — Jumlah waktu yang diperlukan untuk pemrosesan setiap batch.
  - `checkpointLocation` — Lokasi di mana pos pemeriksaan disimpan untuk tugas ETL streaming.
  - `batchMaxRetries` — Jumlah waktu maksimum untuk mengulang mencoba batch sekali lagi jika gagal. Nilai default-nya adalah 3. Opsi ini hanya dapat dikonfigurasi untuk Glue versi 2.0 dan di atasnya.

Contoh:

```
glueContext.forEachBatch(data_frame_datasource0, (dataFrame: Dataset[Row], batchId:
 Long) =>
   {
      if (dataFrame.count() > 0)
        {
          val datasource0 = DynamicFrame(glueContext.addIngestionTimeColumns(dataFrame,
 "hour"), glueContext)
          // @type: DataSink
          // @args: [database = "tempdb", table_name = "fromoptionsoutput",
 stream_batch_time = "100 seconds",
          //      stream_checkpoint_location = "s3://from-options-testing-eu-central-1/
fromOptionsOutput/checkpoint/",
```

```
          //      transformation_ctx = "datasink1"]
          // @return: datasink1
          // @inputs: [frame = datasource0]
          val options_datasink1 = JsonOptions(
             Map("partitionKeys" -> Seq("ingest_year", "ingest_month","ingest_day",
 "ingest_hour"),
             "enableUpdateCatalog" -> true))
          val datasink1 = glueContext.getCatalogSink(
             database = "tempdb",
             tableName = "fromoptionsoutput",
             redshiftTmpDir = "",
             transformationContext = "datasink1",
             additionalOptions = options_datasink1).writeDynamicFrame(datasource0)
        }
   }, JsonOptions("""{"windowSize" : "100 seconds",
         "checkpointLocation" : "s3://from-options-testing-eu-central-1/
fromOptionsOutput/checkpoint/"}"""))
```

def getCatalogSink

```
def getCatalogSink( database : String,
        tableName : String,
        redshiftTmpDir : String = "",
        transformationContext : String = ""
        additionalOptions: JsonOptions = JsonOptions.empty,
        catalogId: String = null
) : DataSink
```

Membuat sebuah DataSink yang menulis ke lokasi yang ditentukan dalam tabel yang didefinisikan dalam Katalog Data.

- `database` — Nama basis data dalam Katalog Data.
- `tableName` — Nama tabel dalam Katalog Data.
- `redshiftTmpDir` — Direktori pentahapan sementara yang akan digunakan dengan data sink tertentu. Diatur ke kosong secara default.
- `transformationContext` — Konteks transformasi yang dikaitkan dengan sink yang akan digunakan oleh bookmark tugas. Diatur ke kosong secara default.
- `additionalOptions`— Opsi tambahan yang disediakan untukAWS Glue.
- `catalogId` — ID katalog (ID akun) dari Katalog Data yang sedang diakses. Bila nol, maka ID akun default pemanggil yang akan digunakan.

Mengembalikan `DataSink`.

def getCatalogSource

```
def getCatalogSource( database : String,
                      tableName : String,
                      redshiftTmpDir : String = "",
                      transformationContext : String = ""
                      pushDownPredicate : String = " "
                      additionalOptions: JsonOptions = JsonOptions.empty,
                      catalogId: String = null
                    ) : DataSource
```

Membuat sebuah DataSource sifat yang membaca data dari tabel definisi dalam Katalog Data.

- `database` — Nama basis data dalam Katalog Data.
- `tableName` — Nama tabel dalam Katalog Data.
- `redshiftTmpDir` — Direktori pentahapan sementara yang akan digunakan dengan data sink tertentu. Diatur ke kosong secara default.
- `transformationContext` — Konteks transformasi yang dikaitkan dengan sink yang akan digunakan oleh bookmark tugas. Diatur ke kosong secara default.
- `pushDownPredicate` — Memfilter partisi tanpa harus mencantumkan dan membaca semua file dalam set data Anda. Untuk informasi selengkapnya, lihat Pra-penyaringan menggunakan predikat pushdown.
- `additionalOptions` — Kumpulan pasangan nama-nilai opsional. Opsi yang mungkin adalah opsi-opsi yang tercantum dalam Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark kecuali `endpointUrl`, `streamName`, `bootstrap.servers`, `security.protocol`, `topicName`, `classification`, dan `delimiter`. Opsi lain yang didukung adalah`catalogPartitionPredicate`:

  `catalogPartitionPredicate`— Anda dapat meneruskan ekspresi katalog untuk memfilter berdasarkan kolom indeks. Ini mendorong penyaringan ke sisi server. Untuk informasi selengkapnya, lihat Indeks AWS Glue Partisi. Perhatikan itu `push_down_predicate` dan `catalogPartitionPredicate` gunakan sintaks yang berbeda. Yang pertama menggunakan sintaks standar Spark SQL dan yang kemudian menggunakan parser JSQL.
- `catalogId` — ID katalog (ID akun) dari Katalog Data yang sedang diakses. Bila nol, maka ID akun default pemanggil yang akan digunakan.

Mengembalikan `DataSource`.

Contoh untuk sumber streaming

```
val data_frame_datasource0 = glueContext.getCatalogSource(
    database = "tempdb",
    tableName = "test-stream-input",
    redshiftTmpDir = "",
    transformationContext = "datasource0",
    additionalOptions = JsonOptions("""{
        "startingPosition": "TRIM_HORIZON", "inferSchema": "false"}""")
    ).getDataFrame()
```

def getJDBCSink

```
def getJDBCSink( catalogConnection : String,
                 options : JsonOptions,
                 redshiftTmpDir : String = "",
                 transformationContext : String = "",
                 catalogId: String = null
               ) : DataSink
```

Membuat sebuah DataSink yang menulis ke basis data JDBC yang ditentukan dalam objek `Connection` dalam Katalog Data. Objek `Connection` memiliki informasi untuk terhubung ke sbuah sink JDBC, termasuk URL, nama pengguna, kata sandi, VPC, subnet, dan grup keamanan.

- `catalogConnection` — Nama koneksi dalam Katalog Data yang berisi URL JDBC yang akan ditulisi.
- `options` — Sebuah string pasangan nama-nilai JSON yang memberikan informasi tambahan yang diperlukan untuk menulis ke penyimpanan data JDBC. Hal ini mencakup:
  - dbtable (wajib) — Nama tabel JDBC. Untuk penyimpanan data JDBC yang mendukung skema dalam basis data, tentukan `schema.table-name`. Jika skema tidak disediakan, maka skema "publik" default digunakan. Contoh berikut menunjukkan parameter pilihan yang mengarahkan ke skema bernama `test` dan sebuah tabel bernama `test_table` dalam basis data `test_db`.

    ```
    options = JsonOptions("""{"dbtable": "test.test_table", "database": "test_db"}""")
    ```
  - database (wajib) — Nama basis data JDBC.
  - Setiap pilihan tambahan diberikan langsung ke penulis JDBC SparkSQL. Untuk informasi selengkapnya, lihat Sumber data Redshift untuk Spark.

- `redshiftTmpDir` — Sebuah direktori pentahapan sementara yang akan digunakan dengan data sink tertentu. Diatur ke kosong secara default.
- `transformationContext` — Konteks transformasi yang dikaitkan dengan sink yang akan digunakan oleh bookmark tugas. Diatur ke kosong secara default.
- `catalogId` — ID katalog (ID akun) dari Katalog Data yang sedang diakses. Bila nol, maka ID akun default pemanggil yang akan digunakan.

Kode contoh:

```
getJDBCSink(catalogConnection = "my-connection-name", options =
 JsonOptions("""{"dbtable": "my-jdbc-table", "database": "my-jdbc-db"}"""),
 redshiftTmpDir = "", transformationContext = "datasink4")
```

Mengembalikan `DataSink`.

def getSink

```
def getSink( connectionType : String,
             connectionOptions : JsonOptions,
             transformationContext : String = ""
           ) : DataSink
```

Membuat file DataSink yang menulis data ke tujuan seperti Amazon Simple Storage Service (Amazon S3), JDBC, atau Glue Data Catalog, atau AWS aliran data Apache Kafka atau Amazon Kinesis.

- `connectionType` — Jenis koneksi. Lihat the section called “Parameter koneksi”.
- `connectionOptions` — Sebuah string pasangan nama-nilai JSON yang memberikan informasi tambahan untuk membangun koneksi dengan data sink. Lihat the section called “Parameter koneksi”.
- `transformationContext` — Konteks transformasi yang dikaitkan dengan sink yang akan digunakan oleh bookmark tugas. Diatur ke kosong secara default.

Mengembalikan `DataSink`.

Format def getSinkWith

```
def getSinkWithFormat( connectionType : String,
```

```
                        options : JsonOptions,
                        transformationContext : String = "",
                        format : String = null,
                        formatOptions : JsonOptions = JsonOptions.empty
                      ) : DataSink
```

Membuat DataSink yang menulis data ke tujuan seperti Amazon S3, JDBC, atau Katalog Data, atau aliran data Apache Kafka atau Amazon Kinesis. Juga menetapkan format untuk data yang akan ditulis ke tujuan.

- `connectionType` — Jenis koneksi. Lihat the section called “Parameter koneksi”.
- `options` — Sebuah string pasangan nama-nilai JSON yang memberikan informasi tambahan untuk membangun sebuah koneksi dengan data sink. Lihat the section called “Parameter koneksi”.
- `transformationContext` — Konteks transformasi yang dikaitkan dengan sink yang akan digunakan oleh bookmark tugas. Diatur ke kosong secara default.
- `format` — Format data yang akan ditulis ke tujuan.
- `formatOptions` — Sebuah string pasangan nama-nilai JSON yang menyediakan opsi tambahan untuk memformat data di tujuan. Lihat Opsi format data.

Mengembalikan `DataSink`.

def getSource

```
def getSource( connectionType : String,
               connectionOptions : JsonOptions,
               transformationContext : String = ""
               pushDownPredicate
             ) : DataSource
```

Membuat DataSource sifat yang membaca data dari sumber seperti Amazon S3, JDBC, atau Glue AWS Data Catalog. Juga mendukung sumber data streaming Kafka dan Kinesis.

- `connectionType` — Jenis sumber data. Lihat the section called “Parameter koneksi”.
- `connectionOptions` — Sebuah string pasangan nama-nilai JSON yang memberikan informasi tambahan untuk membangun sebuah koneksi dengan sumber data. Untuk informasi selengkapnya, lihat the section called “Parameter koneksi”.

Sumber streaming Kinesis memerlukan opsi koneksi berikut: `streamARN`, `startingPosition`, `inferSchema`, dan `classification`.

Sumber streaming Kafka membutuhkan pilihan koneksi berikut: `connectionName`, `topicName`, `startingOffsets`, `inferSchema`, dan `classification`.

- `transformationContext` — Konteks transformasi yang dikaitkan dengan sink yang akan digunakan oleh bookmark tugas. Diatur ke kosong secara default.
- `pushDownPredicate` — Predikat pada kolom partisi.

Mengembalikan `DataSource`.

Contoh untuk sumber streaming Amazon Kinesis:

```
val kinesisOptions = jsonOptions()
data_frame_datasource0 = glueContext.getSource("kinesis",
 kinesisOptions).getDataFrame()

private def jsonOptions(): JsonOptions = {
    new JsonOptions(
      s"""{"streamARN": "arn:aws:kinesis:eu-central-1:123456789012:stream/
fromOptionsStream",
         |"startingPosition": "TRIM_HORIZON",
         |"inferSchema": "true",
         |"classification": "json"}""".stripMargin)
}
```

Contoh untuk sumber streaming Kafka:

```
val kafkaOptions = jsonOptions()
val data_frame_datasource0 = glueContext.getSource("kafka",
 kafkaOptions).getDataFrame()

private def jsonOptions(): JsonOptions = {
    new JsonOptions(
      s"""{"connectionName": "ConfluentKafka",
         |"topicName": "kafka-auth-topic",
         |"startingOffsets": "earliest",
         |"inferSchema": "true",
         |"classification": "json"}""".stripMargin)
 }
```

Format def getSourceWith

```
def getSourceWithFormat( connectionType : String,
                         options : JsonOptions,
                         transformationContext : String = "",
                         format : String = null,
                         formatOptions : JsonOptions = JsonOptions.empty
                       ) : DataSource
```

Membuat DataSource sifat yang membaca data dari sumber seperti Amazon S3, JDBC, atau AWS Glue Data Catalog, dan juga menetapkan format data yang disimpan dalam sumber.

- `connectionType` — Jenis sumber data. Lihat the section called “Parameter koneksi”.
- `options` — Sebuah string pasangan nama-nilai JSON yang memberikan informasi tambahan untuk membangun sebuah koneksi dengan sumber data. Lihat the section called “Parameter koneksi”.
- `transformationContext` — Konteks transformasi yang dikaitkan dengan sink yang akan digunakan oleh bookmark tugas. Diatur ke kosong secara default.
- `format` — Format data yang disimpan pada sumber. Saat `connectionType` adalah "s3", Anda juga dapat menentukan `format`. Bisa berupa “avro”, “csv”, “groklog”, “ion”, “json”, “xml”, “parquet”, atau “orc”, salah satunya.
- `formatOptions` — Sebuah string pasangan nama-nilai JSON yang menyediakan opsi tambahan untuk mengurai data di sumber. Lihat Opsi format data.

Mengembalikan `DataSource`.

Contoh

Buat DynamicFrame dari sumber data yang merupakan file nilai yang dipisahkan koma (CSV) di Amazon S3:

```
val datasource0 = glueContext.getSourceWithFormat(
    connectionType="s3",
    options =JsonOptions(s"""{"paths": [ "s3://csv/nycflights.csv"]}"""),
    transformationContext = "datasource0",
    format = "csv",
    formatOptions=JsonOptions(s"""{"withHeader":"true","separator": ","}""")
    ).getDynamicFrame()
```

Buat DynamicFrame dari sumber data yang merupakan PostgreSQL menggunakan koneksi JDBC:

```
val datasource0 = glueContext.getSourceWithFormat(
    connectionType="postgresql",
    options =JsonOptions(s"""{
      "url":"jdbc:postgresql://databasePostgres-1.rds.amazonaws.com:5432/testdb",
      "dbtable": "public.company",
      "redshiftTmpDir":"",
      "user":"username",
      "password":"password123"
    }"""),
    transformationContext = "datasource0").getDynamicFrame()
```

Buat DynamicFrame dari sumber data yang merupakan MySQL menggunakan koneksi JDBC:

```
 val datasource0 = glueContext.getSourceWithFormat(
    connectionType="mysql",
    options =JsonOptions(s"""{
      "url":"jdbc:mysql://databaseMysql-1.rds.amazonaws.com:3306/testdb",
      "dbtable": "athenatest_nycflights13_csv",
      "redshiftTmpDir":"",
      "user":"username",
      "password":"password123"
    }"""),
    transformationContext = "datasource0").getDynamicFrame()
```

def getSparkSession

```
def getSparkSession : SparkSession
```

Mendapat `SparkSession` objek yang terkait dengan ini GlueContext. Gunakan SparkSession objek ini untuk mendaftarkan tabel dan UDF untuk digunakan dengan `DataFrame` dibuat dari DynamicFrames.

Mengembalikan SparkSession.

def StartTransaksi

```
def startTransaction(readOnly: Boolean):String
```

Mulai transaksi baru. Secara internal memanggil Lake Formation StartTransaction API.

- `readOnly`— (Boolean) Menunjukkan apakah transaksi ini harus dibaca saja atau dibaca dan ditulis. Penulisan yang dibuat menggunakan ID transaksi hanya-baca akan ditolak. Transaksi read-only tidak perlu dilakukan.

Mengembalikan ID transaksi.

def CommitTransaction

```
def commitTransaction(transactionId: String, waitForCommit: Boolean): Boolean
```

Upaya untuk melakukan transaksi yang ditentukan. `commitTransaction`dapat kembali sebelum transaksi selesai dilakukan. Secara internal memanggil Lake Formation CommitTransaction API.

- `transactionId`— (String) Transaksi untuk melakukan.
- `waitForCommit`— (Boolean) Menentukan apakah `commitTransaction` pengembalian segera. Nilai default-nya adalah betul. Jika salah, `commitTransaction` polling dan menunggu sampai transaksi dilakukan. Jumlah waktu tunggu dibatasi hingga 1 menit menggunakan backoff eksponensial dengan maksimal 6 upaya coba lagi.

Mengembalikan Boolean untuk menunjukkan apakah komit dilakukan atau tidak.

def batalkan Transaksi

```
def cancelTransaction(transactionId: String): Unit
```

Upaya untuk membatalkan transaksi yang ditentukan. Secara internal memanggil Lake Formation CancelTransactionAPI.

- `transactionId`— (String) Transaksi untuk membatalkan.

Mengembalikan `TransactionCommittedException` pengecualian jika transaksi sebelumnya dilakukan.

def this

```
def this( sc : SparkContext,
          minPartitions : Int,
          targetPartitions : Int )
```

Membuat sebuah objek `GlueContext` menggunakan `SparkContext` yang ditentukan, partisi minimum, dan partisi target.

- `sc`—`SparkContext`.
- `minPartitions` — Jumlah partisi minimum.
- `targetPartitions` — Jumlah partisi target.

Mengembalikan `GlueContext`.

def this

```
def this( sc : SparkContext )
```

Membuat sebuah objek `GlueContext` dengan `SparkContext` yang disediakan. Menetapkan partisi minimum ke 10 dan partisi target ke 20.

- `sc`—`SparkContext`.

Mengembalikan `GlueContext`.

def this

```
def this( sparkContext : JavaSparkContext )
```

Membuat sebuah objek `GlueContext` dengan `JavaSparkContext` yang disediakan. Menetapkan partisi minimum ke 10 dan partisi target ke 20.

- `sparkContext`—`JavaSparkContext`.

Mengembalikan `GlueContext`.

MappingSpec

Package: com.amazonaws.services.glue

MappingSpec kelas kasus

```
case class MappingSpec( sourcePath: SchemaPath,
```

```
                        sourceType: DataType,
                        targetPath: SchemaPath,
                        targetType: DataTyp
                       ) extends Product4[String, String, String, String] {
  override def _1: String = sourcePath.toString
  override def _2: String = ExtendedTypeName.fromDataType(sourceType)
  override def _3: String = targetPath.toString
  override def _4: String = ExtendedTypeName.fromDataType(targetType)
}
```

- `sourcePath` — `SchemaPath` dari bidang sumber.
- `sourceType` — `DataType` dari bidang sumber.
- `targetPath` — `SchemaPath` dari bidang target.
- `targetType` — `DataType` dari bidang target.

Sebuah `MappingSpec` menentukan pemetaan dari path sumber dan tipe data sumber ke path target dan tipe data target. Nilai di path sumber dalam bingkai sumber muncul dalam bingkai target di path target. Jenis sumber data diubah ke tipe data target.

Ia meluas dari `Product4` sehingga Anda dapat menangani `Product4` di antarmuka `applyMapping` Anda.

MappingSpec objek

```
object MappingSpec
```

Objek `MappingSpec` memiliki anggota-anggota berikut:

Val orderingByTarget

```
val orderingByTarget: Ordering[MappingSpec]
```

Def berlaku

```
def apply( sourcePath : String,
           sourceType : DataType,
           targetPath : String,
           targetType : DataType
```

```
        ) : MappingSpec
```

Membuat sebuah `MappingSpec`.

- `sourcePath` — Sebuah representasi string dari path sumber.
- `sourceType` — `DataType` sumber.
- `targetPath` — Sebuah representasi string dari path target.
- `targetType` — `DataType` target.

Mengembalikan `MappingSpec`.

Def berlaku

```
def apply( sourcePath : String,
           sourceTypeString : String,
           targetPath : String,
           targetTypeString : String
         ) : MappingSpec
```

Membuat sebuah `MappingSpec`.

- `sourcePath` — Sebuah representasi string dari path sumber.
- `sourceType` — Sebuah representasi string dari tipe data sumber.
- `targetPath` — Sebuah representasi string dari path target.
- `targetType` — Sebuah representasi string dari tipe data target.

Mengembalikan a MappingSpec.

Def berlaku

```
def apply( product : Product4[String, String, String, String] ) : MappingSpec
```

Membuat sebuah `MappingSpec`.

- `product` — `Product4` dari path sumber, tipe data sumber, path target, dan tipe data target.

Mengembalikan `MappingSpec`.

AWS GlueAPI Scala ResolveSpec

Topik

Package: com.amazonaws.services.glue

ResolveSpec objek

ResolveSpec

```
object ResolveSpec
```

Def

```
def apply( path : String,
           action : String
         ) : ResolveSpec
```

Membuat sebuah `ResolveSpec`.

- `path` — Sebuah representasi string dari bidang pilihan yang perlu diubah.
- `action` — Sebuah tindakan resolusi. Tindakan tersebut dapat berupa salah satu dari yang berikut ini: `Project`, `KeepAsStruct`, atau `Cast`.

Mengembalikan `ResolveSpec`.

Def

```
def apply( product : Product2[String, String] ) : ResolveSpec
```

Membuat sebuah `ResolveSpec`.

- `product` — `Product2` dari: path sumber, tindakan resolusi.

Mengembalikan `ResolveSpec`.

ResolveSpec kelas kasus

```
case class ResolveSpec extends Product2[String, String]  (
           path : SchemaPath,
           action : String )
```

Membuat sebuah `ResolveSpec`.

- `path` — `SchemaPath` dari bidang pilihan yang perlu diubah.
- `action` — Sebuah tindakan resolusi. Tindakan tersebut dapat berupa salah satu dari yang berikut ini: `Project`, `KeepAsStruct`, atau `Cast`.

ResolveSpec metode def

```
def _1 : String
```

```
def _2 : String
```

AWS GlueAPI Scala ArrayNode

Package: com.amazonaws.services.glue.types

ArrayNode kelas kasus

ArrayNode

```
case class ArrayNode extends DynamicNode  (
           value : ArrayBuffer[DynamicNode] )
```

ArrayNode metode def

```
def add( node : DynamicNode )
```

```
def clone
```

```
def equals( other : Any )
```

```
def get( index : Int ) : Option[DynamicNode]
```

```
def getValue
```

```
def hashCode : Int
```

```
def isEmpty : Boolean
```

```
def nodeType
```

```
def remove( index : Int )
```

```
def this
```

```
def toIterator : Iterator[DynamicNode]
```

```
def toJson : String
```

```
def update( index : Int,
            node : DynamicNode )
```

AWS GlueAPI Scala BinaryNode

Package: com.amazonaws.services.glue.types

BinaryNode kelas kasus

BinaryNode

```
case class BinaryNode extends ScalarNode(value, TypeCode.BINARY)  (
           value : Array[Byte] )
```

BinaryNode bidang val

- `ordering`

BinaryNode metode def

```
def clone
```

```
def equals( other : Any )
```

```
def hashCode : Int
```

AWS GlueAPI Scala BooleanNode

Package: com.amazonaws.services.glue.types

BooleanNode kelas kasus

BooleanNode

```
case class BooleanNode extends ScalarNode(value, TypeCode.BOOLEAN)  (
           value : Boolean )
```

BooleanNode bidang val

- `ordering`

BooleanNode metode def

```
def equals( other : Any )
```

AWS GlueAPI Scala ByteNode

Package: com.amazonaws.services.glue.types

ByteNode kelas kasus

ByteNode

```
case class ByteNode extends ScalarNode(value, TypeCode.BYTE)  (
           value : Byte )
```

ByteNode bidang val

- `ordering`

ByteNode metode def

```
def equals( other : Any )
```

AWS GlueAPI Scala DateNode

Package: com.amazonaws.services.glue.types

DateNode kelas kasus

DateNode

```
case class DateNode extends ScalarNode(value, TypeCode.DATE)  (
           value : Date )
```

DateNode bidang val

- `ordering`

DateNode metode def

```
def equals( other : Any )
```

```
def this( value : Int )
```

AWS GlueAPI Scala DecimalNode

Package: com.amazonaws.services.glue.types

DecimalNode kelas kasus

DecimalNode

```
case class DecimalNode extends ScalarNode(value, TypeCode.DECIMAL)  (
```

```
            value : BigDecimal )
```

DecimalNode bidang val

- `ordering`

DecimalNode metode def

```
def equals( other : Any )
```

```
def this( value : Decimal )
```

AWS GlueAPI Scala DoubleNode

Package: com.amazonaws.services.glue.types

DoubleNode kelas kasus

DoubleNode

```
case class DoubleNode extends ScalarNode(value, TypeCode.DOUBLE)  (
            value : Double )
```

DoubleNode bidang val

- `ordering`

DoubleNode metode def

```
def equals( other : Any )
```

AWS GlueAPI Scala DynamicNode

Topik

- DynamicNode kelas
- DynamicNode objek

Package: com.amazonaws.services.glue.types

DynamicNode kelas

DynamicNode

```
class DynamicNode extends Serializable with Cloneable
```

DynamicNode metode def

```
def getValue : Any
```

Dapatkan nilai biasa dan mengikat catatan saat ini:

```
def nodeType : TypeCode
```

```
def toJson : String
```

Metode untuk debug:

```
def toRow( schema : Schema,
           options : Map[String, ResolveOption]
         ) : Row
```

```
def typeName : String
```

DynamicNode objek

DynamicNode

```
object DynamicNode
```

DynamicNode metode def

```
def quote( field : String,
           useQuotes : Boolean
         ) : String
```

```
def quote( node : DynamicNode,
           useQuotes : Boolean
```

```
        ) : String
```

EvaluateDataQuality kelas

AWSKualitas Data Glue dalam rilis pratinjau untuk AWS Glue dan dapat berubah sewaktu-waktu.

Package: com.amazonaws.services.glue.dq

```
object EvaluateDataQuality
```

Def berlaku

```
def apply(frame: DynamicFrame,
            ruleset: String,
            publishingOptions: JsonOptions = JsonOptions.empty): DynamicFrame
```

Mengevaluasi aturan kualitas data terhadap`DynamicFrame`, dan mengembalikan yang baru `DynamicFrame` dengan hasil evaluasi. Untuk mempelajari lebih lanjut tentang Kualitas Data AWS Glue, lihatAWS Glue Kualitas Data.

- `frame`— `DynamicFrame` Yang ingin Anda evaluasi kualitas data.
- `ruleset`— Aturan Bahasa Definisi Kualitas Data (DQDL) dalam format string. Untuk mempelajari lebih lanjut tentang DQDL, lihat panduan. Referensi Bahasa Definisi Kualitas Data (DQDL)
- `publishingOptions`— Kamus yang menentukan opsi berikut untuk mempublikasikan hasil evaluasi dan metrik:
  - `dataQualityEvaluationContext`— String yang menentukan namespace di mana AWS Glue harus mempublikasikan Amazon CloudWatch metrik dan hasil kualitas data. Metrik agregat muncul di CloudWatch, sementara hasil lengkap muncul di antarmuka AWS Glue Studio.
    - Wajib: Tidak
    - Nilai default: `default_context`
  - `enableDataQualityCloudWatchMetrics`— Menentukan apakah hasil evaluasi kualitas data harus dipublikasikan ke CloudWatch. Anda menentukan namespace untuk metrik menggunakan opsi. `dataQualityEvaluationContext`
    - Wajib: Tidak
    - Nilai default: Salah

- `enableDataQualityResultsPublishing`— Menentukan apakah hasil kualitas data harus terlihat pada tab Kualitas Data di antarmuka AWS Glue Studio.
  - Wajib: Tidak
  - Nilai default: Benar
- `resultsS3Prefix`— Menentukan lokasi Amazon S3 di mana AWS Glue dapat menulis hasil evaluasi kualitas data.
  - Wajib: Tidak
  - Nilai default: “” (string kosong)

Contoh

Contoh kode berikut menunjukkan bagaimana mengevaluasi kualitas data untuk `DynamicFrame` sebelum melakukan `SelectFields` transformasi. Skrip memverifikasi bahwa semua aturan kualitas data lulus sebelum mencoba transformasi.

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
import com.amazonaws.services.glue.dq.EvaluateDataQuality

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    // @params: [JOB_NAME]
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    // Create DynamicFrame with data
    val Legislators_Area = glueContext.getCatalogSource(database="legislators",
 tableName="areas_json", transformationContext="S3bucket_node1").getDynamicFrame()

    // Define data quality ruleset
    val DQ_Ruleset = """
      Rules = [ColumnExists "id"]
```

```
    """

    // Evaluate data quality
    val DQ_Results = EvaluateDataQuality.apply(frame=Legislators_Area,
 ruleset=DQ_Ruleset, publishingOptions=JsonOptions("""{"dataQualityEvaluationContext":
 "Legislators_Area", "enableDataQualityMetrics": "true",
 "enableDataQualityResultsPublishing": "true"}"""))
    assert(DQ_Results.filter(_.getField("Outcome").contains("Failed")).count == 0,
 "Failing DQ rules for Legislators_Area caused the job to fail.")

    // Script generated for node Select Fields
    val SelectFields_Results = Legislators_Area.selectFields(paths=Seq("id", "name"),
 transformationContext="Legislators_Area")

    Job.commit()
  }
}
```

AWS GlueAPI Scala FloatNode

Package: com.amazonaws.services.glue.types

FloatNode kelas kasus

FloatNode

```
case class FloatNode extends ScalarNode(value, TypeCode.FLOAT)  (
           value : Float )
```

FloatNode bidang val

- `ordering`

FloatNode metode def

```
def equals( other : Any )
```

FillMissingValues kelas

Package: com.amazonaws.services.glue.ml

```
object FillMissingValues
```

Def berlaku

```
def apply(frame: DynamicFrame,
          missingValuesColumn: String,
          outputColumn: String = "",
          transformationContext: String = "",
          callSite: CallSite = CallSite("Not provided", ""),
          stageThreshold: Long = 0,
          totalThreshold: Long = 0): DynamicFrame
```

Mengisi nilai-nilai yang hilang dari bingkai dinamis dalam kolom yang ditentukan dan mengembalikan bingkai baru dengan perkiraan dalam sebuah kolom baru. Untuk baris tanpa nilai yang hilang, nilai kolom yang ditentukan diduplikasi ke kolom baru tersebut.

- `frame`— DynamicFrame Di mana untuk mengisi nilai yang hilang. Diperlukan.
- `missingValuesColumn` — Kolom yang berisi nilai-nilai yang hilang (nilai `null` dan string kosong). Diperlukan.
- `outputColumn` — Nama kolom baru yang akan berisi perkiraan nilai untuk semua baris yang nilainya hilang. Opsional; default-nya adalah nilai `missingValuesColumn` dengan sufiks `"_filled"`.
- `transformationContext` — Sebuah string unik yang digunakan untuk mengidentifikasi informasi status (opsional).
- `callSite` — Digunakan untuk menyediakan informasi konteks untuk pelaporan kesalahan. (opsional).
- `stageThreshold` — Jumlah maksimum kesalahan yang dapat terjadi dalam transformasi sebelum kesalahan keluar (opsional; default-nya adalah nol).
- `totalThreshold` — Jumlah maksimum kesalahan yang dapat terjadi secara keseluruhan sebelum kesalahan keluar (opsional; default-nya adalah nol).

Mengembalikan bingkai dinamis baru dengan satu kolom tambahan yang berisi perkiraan untuk baris dengan nilai-nilai yang hilang dan nilai sekarang untuk baris lainnya.

FindMatches kelas

Package: com.amazonaws.services.glue.ml

```
object FindMatches
```

Def berlaku

```
def apply(frame: DynamicFrame,
            transformId: String,
            transformationContext: String = "",
            callSite: CallSite = CallSite("Not provided", ""),
            stageThreshold: Long = 0,
            totalThreshold: Long = 0,
            enforcedMatches: DynamicFrame = null): DynamicFrame,
   computeMatchConfidenceScores: Boolean
```

Temukan kecocokan dalam bingkai input dan kembalikan bingkai baru dengan kolom baru yang berisi ID unik per grup kecocokan.

- `frame`— Tempat DynamicFrame untuk menemukan kecocokan. Diperlukan.
- `transformId`— ID unik yang terkait dengan FindMatches transformasi untuk diterapkan pada bingkai input. Diperlukan.
- `transformationContext` — Pengidentifikasi untuk `DynamicFrame` ini. `transformationContext` digunakan sebagai kunci untuk status bookmark tugas yang tetap ada di seluruh eksekusi. Tidak wajib.
- `callSite` — Digunakan untuk menyediakan informasi konteks untuk pelaporan kesalahan. Nilai-nilai ini secara otomatis ditetapkan ketika memanggil dari Python. Tidak wajib.
- `stageThreshold` — Jumlah maksimum catatan kesalahan yang diizinkan dari komputasi `DynamicFrame` ini sebelum melemparkan pengecualian, tidak termasuk catatan yang ada dalam `DynamicFrame` sebelumnya. Tidak wajib. Default-nya adalah nol.
- `totalThreshold` — Jumlah maksimum total catatan kesalahan sebelum pengecualian dilemparkan, termasuk yang dari bingkai sebelumnya. Tidak wajib. Default-nya adalah nol.
- `enforcedMatches` — Bingkai untuk kecocokan yang diberlakukan. Tidak wajib. Defaultnya adalah `null`.
- `computeMatchConfidenceScores`— Nilai Boolean yang menunjukkan apakah akan menghitung skor kepercayaan untuk setiap kelompok catatan yang cocok. Tidak wajib. Default-nya adalah salah.

Mengembalikan sebuah bingkai dinamis baru dengan pengidentifikasi unik yang ditetapkan untuk setiap grup catatan yang cocok.

FindIncrementalMatches kelas

Package: com.amazonaws.services.glue.ml

```
object FindIncrementalMatches
```

Def berlaku

```
apply(existingFrame: DynamicFrame,
            incrementalFrame: DynamicFrame,
            transformId: String,
            transformationContext: String = "",
            callSite: CallSite = CallSite("Not provided", ""),
            stageThreshold: Long = 0,
            totalThreshold: Long = 0,
            enforcedMatches: DynamicFrame = null): DynamicFrame,
   computeMatchConfidenceScores: Boolean
```

Menemukan kecocokan di seluruh bingkai yang ada dan bingkai tambahan dan mengembalikan bingkai baru dengan kolom yang berisi ID unik per grup kecocokan.

- `existingframe` — Sebuah bingkai yang ada yang telah ditetapkan ID kecocokan untuk setiap grup. Wajib.
- `incrementalframe` — Sebuah bingkai tambahan yang digunakan untuk menemukan kecocokan berdasarkan bingkai yang ada. Diperlukan.
- `transformId`— ID unik yang terkait dengan FindIncrementalMatches transformasi untuk diterapkan pada frame input. Diperlukan.
- `transformationContext` — Pengidentifikasi untuk `DynamicFrame` ini. `transformationContext` digunakan sebagai kunci untuk status bookmark tugas yang tetap ada di seluruh eksekusi. Tidak wajib.
- `callSite` — Digunakan untuk menyediakan informasi konteks untuk pelaporan kesalahan. Nilai-nilai ini secara otomatis ditetapkan ketika memanggil dari Python. Tidak wajib.
- `stageThreshold` — Jumlah maksimum catatan kesalahan yang diizinkan dari komputasi `DynamicFrame` ini sebelum melemparkan pengecualian, tidak termasuk catatan yang ada dalam `DynamicFrame` sebelumnya. Tidak wajib. Default-nya adalah nol.
- `totalThreshold` — Jumlah maksimum total catatan kesalahan sebelum pengecualian dilemparkan, termasuk yang dari bingkai sebelumnya. Tidak wajib. Default-nya adalah nol.

- `enforcedMatches` — Bingkai untuk kecocokan yang diberlakukan. Tidak wajib. Defaultnya adalah `null`.
- `computeMatchConfidenceScores`— Nilai Boolean yang menunjukkan apakah akan menghitung skor kepercayaan untuk setiap kelompok catatan yang cocok. Tidak wajib. Default-nya adalah salah.

Mengembalikan sebuah bingkai dinamis baru dengan pengidentifikasi unik yang ditetapkan untuk setiap grup catatan yang cocok.

AWS GlueAPI Scala IntegerNode

Package: com.amazonaws.services.glue.types

IntegerNode kelas kasus

IntegerNode

```
case class IntegerNode extends ScalarNode(value, TypeCode.INT)  (
           value : Int )
```

IntegerNode bidang val

- `ordering`

IntegerNode metode def

```
def equals( other : Any )
```

AWS GlueAPI Scala LongNode

Package: com.amazonaws.services.glue.types

LongNode kelas kasus

LongNode

```
case class LongNode extends ScalarNode(value, TypeCode.LONG)  (
           value : Long )
```

LongNode bidang val

- `ordering`

LongNode metode def

```
def equals( other : Any )
```

AWS GlueAPI Scala MapLikeNode

Package: com.amazonaws.services.glue.types

MapLikeNode kelas

MapLikeNode

```
class MapLikeNode extends DynamicNode  (
          value : mutable.Map[String, DynamicNode] )
```

MapLikeNode metode def

```
def clear : Unit
```

```
def get( name : String ) : Option[DynamicNode]
```

```
def getValue
```

```
def has( name : String ) : Boolean
```

```
def isEmpty : Boolean
```

```
def put( name : String,
         node : DynamicNode
       ) : Option[DynamicNode]
```

```
def remove( name : String ) : Option[DynamicNode]
```

```
def toIterator : Iterator[(String, DynamicNode)]
```

```
def toJson : String
```

```
def toJson( useQuotes : Boolean ) : String
```

Contoh: Mengingat JSON ini:

```
{"foo": "bar"}
```

Jika `useQuotes == true`, `toJson` menghasilkan `{"foo": "bar"}`. Jika `useQuotes == false`, `toJson` menghasilkan `{foo: bar}` @return.

AWS GlueAPI Scala MapNode

Package: com.amazonaws.services.glue.types

MapNode kelas kasus

MapNode

```
case class MapNode extends MapLikeNode(value)  (
           value : mutable.Map[String, DynamicNode] )
```

MapNode metode def

```
def clone
```

```
def equals( other : Any )
```

```
def hashCode : Int
```

```
def nodeType
```

```
def this
```

AWS GlueAPI Scala NullNode

Topik

Package: com.amazonaws.services.glue.types

NullNode kelas

NullNode

```
class NullNode
```

NullNode objek kasus

NullNode

```
case object NullNode extends NullNode
```

AWS GlueAPI Scala ObjectNode

Topik

Package: com.amazonaws.services.glue.types

ObjectNode objek

ObjectNode

```
object ObjectNode
```

ObjectNode metode def

```
def apply( frameKeys : Set[String],
           v1 : mutable.Map[String, DynamicNode],
```

```
            v2 : mutable.Map[String, DynamicNode],
            resolveWith : String
          ) : ObjectNode
```

ObjectNode kelas kasus

ObjectNode

```
case class ObjectNode extends MapLikeNode(value)  (
            val value : mutable.Map[String, DynamicNode] )
```

ObjectNode metode def

```
def clone
```

```
def equals( other : Any )
```

```
def hashCode : Int
```

```
def nodeType
```

```
def this
```

AWS GlueAPI Scala ScalarNode

Topik

- ScalarNode kelas
- ScalarNode objek

Package: com.amazonaws.services.glue.types

ScalarNode kelas

ScalarNode

```
class ScalarNode extends DynamicNode  (
            value : Any,
```

```
            scalarType : TypeCode )
```

ScalarNode metode def

```
def compare( other : Any,
             operator : String
           ) : Boolean
```

```
def getValue
```

```
def hashCode : Int
```

```
def nodeType
```

```
def toJson
```

ScalarNode objek

ScalarNode

```
object ScalarNode
```

ScalarNode metode def

```
def apply( v : Any ) : DynamicNode
```

```
def compare( tv : Ordered[T],
             other : T,
             operator : String
           ) : Boolean
```

```
def compareAny( v : Any,
                y : Any,
                o : String )
```

```
def withEscapedSpecialCharacters( jsonToEscape : String ) : String
```

AWS GlueAPI Scala ShortNode

Package: com.amazonaws.services.glue.types

ShortNode kelas kasus

ShortNode

```
case class ShortNode extends ScalarNode(value, TypeCode.SHORT)  (
           value : Short )
```

ShortNode bidang val

- `ordering`

ShortNode metode def

```
def equals( other : Any )
```

AWS GlueAPI Scala StringNode

Package: com.amazonaws.services.glue.types

StringNode kelas kasus

StringNode

```
case class StringNode extends ScalarNode(value, TypeCode.STRING)  (
           value : String )
```

StringNode bidang val

- `ordering`

StringNode metode def

```
def equals( other : Any )
```

```
def this( value : UTF8String )
```

AWS GlueAPI Scala TimestampNode

Package: com.amazonaws.services.glue.types

TimestampNode kelas kasus

TimestampNode

```
case class TimestampNode extends ScalarNode(value, TypeCode.TIMESTAMP)  (
           value : Timestamp )
```

TimestampNode bidang val

- `ordering`

TimestampNode metode def

```
def equals( other : Any )
```

```
def this( value : Long )
```

AWS GlueAPI Scala GlueArgParser

Package: com.amazonaws.services.glue.util

Objek GlueArgParser

GlueArgParser

```
object GlueArgParser
```

Ini sangat konsisten dengan versi Python dari `utils.getResolvedOptions` dalam paket `AWSGlueDataplanePython`.

GlueArgParser metode def

```
def getResolvedOptions( args : Array[String],
                        options : Array[String]
```

```
                    ) : Map[String, String]
```

```
def initParser( userOptionsSet : mutable.Set[String] ) : ArgumentParser
```

Example Mengambil argumen yang diteruskan ke pekerjaan

Untuk mengambil argumen pekerjaan, Anda dapat menggunakan `getResolvedOptions` metode ini. Perhatikan contoh berikut, yang mengambil argumen pekerjaan bernama`aws_region`.

```
val args = GlueArgParser.getResolvedOptions(sysArgs,
 Seq("JOB_NAME","aws_region").toArray)
Job.init(args("JOB_NAME"), glueContext, args.asJava)
val region = args("aws_region")
println(region)
```

AWS GlueAPI pekerjaan Scala

Package: com.amazonaws.services.glue.util

Objek Job

Job

```
object Job
```

Metode Job def

```
def commit
```

```
def init( jobName : String,
          glueContext : GlueContext,
          args : java.util.Map[String, String] = Map[String, String]().asJava
        ) : this.type
```

```
def init( jobName : String,
          glueContext : GlueContext,
          endpoint : String,
```

```
          args : java.util.Map[String, String]
        ) : this.type
```

```
def isInitialized
```

```
def reset
```

```
def runId
```

Fitur dan optimasi untuk pemrograman AWS Glue untuk skrip Spark ETL

Bagian berikut menjelaskan teknik dan nilai yang berlaku umum AWS Glue untuk pemrograman Spark ETL (ekstrak, transformasi, dan muat) dalam bahasa apa pun.

Topik

- Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark
- Opsi format data untuk input dan output untuk Spark AWS Glue
- AWSDukungan Glue Data Catalog untuk pekerjaan Spark SQL
- Menggunakan bookmark pekerjaan
- Menggunakan Deteksi Data Sensitif di luar AWS Glue Studio
- AWS GlueVisual Job API

Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark

Dalam AWS Glue untuk Spark, berbagai PySpark dan metode Scala dan transformasi menentukan jenis koneksi menggunakan parameter. `connectionType` Mereka menentukan pilihan koneksi menggunakan parameter `connectionOptions` atau `options`.

Parameter `connectionType` dapat mengambil nilai yang ditunjukkan dalam tabel berikut. Nilai parameter `connectionOptions` (atau `options`) untuk setiap jenis didokumentasikan di bagian berikut. Kecuali jika dinyatakan lain, parameter berlaku saat koneksi digunakan sebagai sumber atau sink.

Untuk kode contoh yang menunjukkan pengaturan dan menggunakan opsi koneksi, lihat beranda untuk setiap jenis koneksi.

`connectionType`	Terhubung ke
dinamodb	Basis data Amazon DynamoDB
kinesis	Amazon Kinesis Data Streams
s3	Amazon S3
documentdb	Basis data Amazon DocumentDB (dengan kompatibilitas MongoDB)
opensearch	OpenSearch Layanan Amazon.
pergeseran merah	Basis data Amazon Redshift
kafka	Kafka atau Amazon Managed Streaming for Apache Kafka
azurecosmos	Azure Cosmos untuk NoSQL.
azuresql	SQL Azure.
bigquery	Google BigQuery.
mongodb	Database MongoDB, termasuk MongoDB Atlas.
sqlserver	Basis data Microsoft SQL Server (lihat Koneksi JDBC)
mysql	Basis data MySQL (lihat Koneksi JDBC)
oracle	Basis data Oracle (lihat Koneksi JDBC)
postgresql	Basis data PostgreSQL (lihat Koneksi JDBC)
saphana	GETAH HANA.
kepingan salju	Danau data kepingan salju
teradata	Pandang Teradata.
vertica	Vertika.
kebiasaan. *	Penyimpanan data Spark, Athena, atau JDBC (lihat Nilai ConnectionType kustom dan AWS Marketplace

`connectionType`	Terhubung ke
pasar. *	Penyimpanan data Spark, Athena, atau JDBC (lihat Nilai ConnectionType kustom dan AWS Marketplace)

Koneksi DynamoDB

Anda dapat menggunakan AWS Glue for Spark untuk membaca dan menulis ke tabel di DynamoDB di Glue. AWS Anda terhubung ke DynamoDB menggunakan izin IAM yang dilampirkan ke pekerjaan Glue Anda. AWS AWS Gluemendukung penulisan data ke tabel DynamoDB AWS akun lain. Untuk informasi selengkapnya, lihat the section called “Akses lintas akun Lintas wilayah ke tabel DynamoDB”.

Selain konektor AWS Glue DynamoDB ETL, Anda dapat membaca dari DynamoDB menggunakan konektor ekspor DynamoDB, yang memanggil permintaan DynamoDB dan menyimpannya di lokasi Amazon S3 yang Anda suplai, dalam `ExportTableToPointInTime` format DynamoDB JSON. AWS Gluekemudian membuat DynamicFrame objek dengan membaca data dari lokasi ekspor Amazon S3.

Penulis DynamoDB tersedia AWS Glue dalam versi 1.0 atau versi yang lebih baru. Konektor ekspor AWS Glue DynamoDB tersedia AWS Glue dalam versi 2.0 atau versi yang lebih baru.

Untuk informasi selengkapnya tentang DynamoDB, lihat dokumentasi Amazon DynamoDB.

> **Note**
>
> Pembaca DynamoDB ETL tidak mendukung filter atau predikat pushdown.

Mengkonfigurasi koneksi DynamoDB

Untuk terhubung ke DynamoDB AWS dari Glue, berikan peran IAM yang terkait dengan izin pekerjaan Glue AWS Anda untuk berinteraksi dengan DynamoDB. Untuk informasi selengkapnya tentang izin yang diperlukan untuk membaca atau menulis dari DynamoDB, lihat Tindakan, sumber daya, dan kunci kondisi untuk DynamoDB dalam dokumentasi IAM.

Dalam situasi berikut, Anda mungkin memerlukan konfigurasi tambahan:

- Saat menggunakan konektor ekspor DynamoDB, Anda harus mengkonfigurasi IAM sehingga pekerjaan Anda dapat meminta ekspor tabel DynamoDB. Selain itu, Anda perlu mengidentifikasi bucket Amazon S3 untuk ekspor dan memberikan izin yang sesuai di IAM agar DynamoDB menulis kepadanya, dan agar tugas Glue Anda AWS dapat membacanya. Untuk informasi lebih lanjut, lihat Minta ekspor tabel di DynamoDB.
- Jika pekerjaan AWS Glue Anda memiliki persyaratan konektivitas VPC Amazon tertentu, gunakan jenis koneksi `NETWORK` AWS Glue untuk menyediakan opsi jaringan. Karena akses ke DynamoDB diotorisasi oleh IAM, tidak perlu jenis koneksi AWS Glue DynamoDB.

Membaca dari dan menulis ke DynamoDB

Contoh kode berikut menunjukkan cara membaca dari (melalui konektor ETL) dan menulis ke tabel DynamoDB. Contoh-contoh tersebut menunjukkan pembacaan dari satu tabel dan penulisan ke tabel lain.

Python

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame.from_options(
    connection_type="dynamodb",
    connection_options={"dynamodb.input.tableName": test_source,
        "dynamodb.throughput.read.percent": "1.0",
        "dynamodb.splits": "100"
    }
)
print(dyf.getNumPartitions())

glue_context.write_dynamic_frame_from_options(
    frame=dyf,
    connection_type="dynamodb",
    connection_options={"dynamodb.output.tableName": test_sink,
```

```
        "dynamodb.throughput.write.percent": "1.0"
    }
)

job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.input.tableName" -> test_source,
        "dynamodb.throughput.read.percent" -> "1.0",
        "dynamodb.splits" -> "100"
      ))
    ).getDynamicFrame()

    print(dynamicFrame.getNumPartitions())

    val dynamoDbSink: DynamoDbDataSink =  glueContext.getSinkWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.output.tableName" -> test_sink,
        "dynamodb.throughput.write.percent" -> "1.0"
      ))
    ).asInstanceOf[DynamoDbDataSink]
```

```
    dynamoDbSink.writeDynamicFrame(dynamicFrame)

    Job.commit()
  }

}
```

Menggunakan konektor ekspor DynamoDB

Konektor ekspor berkinerja lebih baik daripada konektor ETL ketika ukuran tabel DynamoDB lebih besar dari 80 GB. Selain itu, mengingat permintaan ekspor dilakukan di luar proses Spark dalam suatu AWS Glue pekerjaan, Anda dapat mengaktifkan penskalaan otomatis pekerjaan AWS Glue untuk menghemat penggunaan DPU selama permintaan ekspor. Dengan konektor ekspor, Anda juga tidak perlu mengonfigurasi jumlah split untuk paralelisme pelaksana Spark atau persentase pembacaan throughput DynamoDB.

> **Note**
>
> DynamoDB memiliki persyaratan khusus untuk memanggil permintaan. `ExportTableToPointInTime` Untuk informasi selengkapnya, lihat Meminta ekspor tabel di DynamoDB. Misalnya, Point-in-Time-Restore (PITR) perlu diaktifkan di atas meja untuk menggunakan konektor ini. Konektor DynamoDB juga mendukung AWS KMS enkripsi untuk ekspor DynamoDB ke Amazon S3. Menyediakan konfigurasi keamanan Anda dalam konfigurasi pekerjaan AWS Glue memungkinkan AWS KMS enkripsi untuk ekspor DynamoDB. Kunci KMS harus berada di Wilayah yang sama dengan bucket Amazon S3. Perhatikan bahwa biaya tambahan untuk ekspor DynamoDB dan biaya penyimpanan Amazon S3 berlaku. Data yang diekspor di Amazon S3 tetap ada setelah pekerjaan selesai sehingga Anda dapat menggunakannya kembali tanpa ekspor DynamoDB tambahan. Persyaratan untuk menggunakan konektor ini adalah point-in-time pemulihan (PITR) diaktifkan untuk tabel.
> Konektor DynamoDB ETL atau konektor ekspor tidak mendukung filter atau predikat pushdown untuk diterapkan pada sumber DynamoDB.

Contoh kode berikut menunjukkan cara membaca dari (melalui konektor ekspor) dan mencetak jumlah partisi.

Python

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame.from_options(
    connection_type="dynamodb",
    connection_options={
        "dynamodb.export": "ddb",
        "dynamodb.tableArn": test_source,
        "dynamodb.s3.bucket": bucket_name,
        "dynamodb.s3.prefix": bucket_prefix,
        "dynamodb.s3.bucketOwner": account_id_of_bucket,
    }
)
print(dyf.getNumPartitions())

job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
```

```
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType = "dynamodb",
      options = JsonOptions(Map(
        "dynamodb.export" -> "ddb",
        "dynamodb.tableArn" -> test_source,
        "dynamodb.s3.bucket" -> bucket_name,
        "dynamodb.s3.prefix" -> bucket_prefix,
        "dynamodb.s3.bucketOwner" -> account_id_of_bucket,
      ))
    ).getDynamicFrame()

    print(dynamicFrame.getNumPartitions())

    Job.commit()
  }

}
```

Contoh-contoh ini menunjukkan bagaimana melakukan pembacaan dari (melalui konektor ekspor) dan mencetak jumlah partisi dari tabel Katalog Data AWS Glue yang memiliki dynamodb klasifikasi:

Python

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dynamicFrame = glue_context.create_dynamic_frame.from_catalog(
        database=catalog_database,
        table_name=catalog_table_name,
        additional_options={
            "dynamodb.export": "ddb",
            "dynamodb.s3.bucket": s3_bucket,
```

```
            "dynamodb.s3.prefix": s3_bucket_prefix
        }
    )
print(dynamicFrame.getNumPartitions())

job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamoDbDataSink
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {

  def main(sysArgs: Array[String]): Unit = {
    val glueContext = new GlueContext(SparkContext.getOrCreate())
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    val dynamicFrame = glueContext.getCatalogSource(
        database = catalog_database,
        tableName = catalog_table_name,
        additionalOptions = JsonOptions(Map(
            "dynamodb.export" -> "ddb",
            "dynamodb.s3.bucket" -> s3_bucket,
            "dynamodb.s3.prefix" -> s3_bucket_prefix
        ))
    ).getDynamicFrame()
    print(dynamicFrame.getNumPartitions())
)
```

Menyederhanakan penggunaan DynamoDB ekspor JSON

Ekspor DynamoDB dengan AWS Glue konektor ekspor DynamoDB menghasilkan file JSON dari struktur bersarang tertentu. Untuk informasi selengkapnya, lihat Objek data. AWS Gluememasok

DynamicFrame transformasi, yang dapat membuat struktur seperti itu menjadi easier-to-use bentuk untuk aplikasi hilir.

Transformasi dapat dipanggil dengan salah satu dari dua cara. Anda dapat mengatur opsi koneksi `"dynamodb.simplifyDDBJson"` dengan nilai `"true"` saat memanggil metode untuk membaca dari DynamoDB. Anda juga dapat memanggil transformasi sebagai metode yang tersedia secara independen di perpustakaan AWS Glue.

Pertimbangkan skema berikut yang dihasilkan oleh ekspor DynamoDB:

```
root
|-- Item: struct
|    |-- parentMap: struct
|    |    |-- M: struct
|    |    |    |-- childMap: struct
|    |    |    |    |-- M: struct
|    |    |    |    |    |-- appName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- packageName: struct
|    |    |    |    |    |    |-- S: string
|    |    |    |    |    |-- updatedAt: struct
|    |    |    |    |    |    |-- N: string
|    |-- strings: struct
|    |    |-- SS: array
|    |    |    |-- element: string
|    |-- numbers: struct
|    |    |-- NS: array
|    |    |    |-- element: string
|    |-- binaries: struct
|    |    |-- BS: array
|    |    |    |-- element: string
|    |-- isDDBJson: struct
|    |    |-- BOOL: boolean
|    |-- nullValue: struct
|    |    |-- NULL: boolean
```

`simplifyDDBJson`Transformasi akan menyederhanakan ini menjadi:

```
root
|-- parentMap: struct
|    |-- childMap: struct
|    |    |-- appName: string
```

```
|    |    |-- packageName: string
|    |    |-- updatedAt: string
|-- strings: array
|    |-- element: string
|-- numbers: array
|    |-- element: string
|-- binaries: array
|    |-- element: string
|-- isDDBJson: boolean
|-- nullValue: null
```

> **Note**
>
> `simplifyDDBJson`tersedia dalam AWS Glue 3.0 dan versi yang lebih baru. `unnestDDBJson`Transformasi ini juga tersedia untuk menyederhanakan DynamoDB ekspor JSON. Kami mendorong pengguna untuk beralih ke `simplifyDDBJson` dari`unnestDDBJson`.

Mengkonfigurasi paralleisme dalam operasi DynamoDB

Untuk meningkatkan kinerja, Anda dapat menyetel parameter tertentu yang tersedia untuk konektor DynamoDB. Tujuan Anda saat menyetel parameter paralleisme adalah memaksimalkan penggunaan pekerja Glue yang disediakan. AWS Kemudian, jika Anda membutuhkan lebih banyak kinerja, kami sarankan Anda untuk meningkatkan skala pekerjaan Anda dengan meningkatkan jumlah DPU.

Anda dapat mengubah paralelisme dalam operasi baca DynamoDB menggunakan parameter saat menggunakan konektor ETL. `dynamodb.splits` Saat membaca dengan konektor ekspor, Anda tidak perlu mengkonfigurasi jumlah split untuk paralelisme pelaksana Spark. Anda dapat mengubah paralelisme dalam operasi penulisan DynamoDB dengan. `dynamodb.output.numParallelTasks`

Membaca dengan konektor DynamoDB ETL

Kami menyarankan Anda untuk menghitung `dynamodb.splits` berdasarkan jumlah maksimum pekerja yang ditetapkan dalam konfigurasi pekerjaan Anda dan `numSlots` perhitungan berikut. Jika penskalaan otomatis, jumlah aktual pekerja yang tersedia dapat berubah di bawah batas itu. Untuk informasi selengkapnya tentang pengaturan jumlah maksimum pekerja, lihat Jumlah pekerja (`NumberOfWorkers`) dithe section called “Mengkonfigurasi properti pekerjaan Spark”.

- `numExecutors = NumberOfWorkers - 1`

Untuk konteks, satu eksekutor dicadangkan untuk driver Spark; pelaksana lain digunakan untuk memproses data.

- `numSlotsPerExecutor =`

  AWS Glue 3.0 and later versions

  - 4 jika `WorkerType` adalah `G.1X`
  - 8 jika `WorkerType` adalah `G.2X`
  - 16 jika `WorkerType` adalah `G.4X`
  - 32 jika `WorkerType` adalah `G.8X`

  AWS Glue 2.0 and legacy versions

  - 8 jika `WorkerType` adalah `G.1X`
  - 16 jika `WorkerType` adalah `G.2X`
- `numSlots = numSlotsPerExecutor * numExecutors`

Kami sarankan Anda mengatur `dynamodb.splits` ke jumlah slot yang tersedia,`numSlots`.

Menulis ke DynamoDB

`dynamodb.output.numParallelTasks`Parameter ini digunakan untuk menentukan WCU per tugas Spark, menggunakan perhitungan berikut:

```
permittedWcuPerTask = ( TableWCU * dynamodb.throughput.write.percent ) / 
dynamodb.output.numParallelTasks
```

Penulis DynamoDB akan berfungsi paling baik jika konfigurasi secara akurat mewakili jumlah tugas Spark yang ditulis ke DynamoDB. Dalam beberapa kasus, Anda mungkin perlu mengganti perhitungan default untuk meningkatkan kinerja penulisan. Jika Anda tidak menentukan parameter ini, tugas WCU per Spark yang diizinkan akan dihitung secara otomatis dengan rumus berikut:

-
  - `numPartitions = dynamicframe.getNumPartitions()`
  - `numSlots`(seperti yang dijelaskan sebelumnya di bagian ini)
  - `numParallelTasks = min(numPartitions, numSlots)`
- Contoh 1 DPU = 10, = Standar. WorkerType Input DynamicFrame memiliki 100 partisi RDD.
  - `numPartitions = 100`
  - `numExecutors = (10 - 1) * 2 - 1 = 17`

  - `numSlots = 4 * 17 = 68`
  - `numParallelTasks = min(100, 68) = 68`
- Contoh 2 DPU = 10, = Standar. WorkerType Input DynamicFrame memiliki 20 partisi RDD.
  - `numPartitions = 20`
  - `numExecutors = (10 - 1) * 2 - 1 = 17`
  - `numSlots = 4 * 17 = 68`
  - `numParallelTasks = min(20, 68) = 20`

> **Note**
>
> Pekerjaan pada versi AWS Glue lama dan yang menggunakan pekerja Standar memerlukan metode yang berbeda untuk menghitung jumlah slot. Jika Anda perlu menyesuaikan kinerja pekerjaan ini, kami sarankan Anda beralih ke versi AWS Glue yang didukung.

Referensi opsi koneksi DynamoDB

Mengkhususkan koneksi ke Amazon DynamoDB.

Pilihan koneksi berbeda untuk koneksi sumber dan koneksi sink.

“ConnectionType”: “dynamodb” dengan konektor ETL sebagai sumber

Gunakan opsi koneksi berikut `"connectionType": "dynamodb"` sebagai sumber, saat menggunakan konektor AWS Glue DynamoDB ETL:

- `"dynamodb.input.tableName"`: (Wajib) Tabel DynamoDB tempat untuk membaca.
- `"dynamodb.throughput.read.percent"`: (Opsional) Persentase unit kapasitas baca (RCU) yang akan digunakan. Default diatur ke "0,5". Nilai yang dapat diterima adalah dari "0,1" hingga "1,5", inklusif.
  - `0.5`mewakili tingkat baca default, artinya AWS Glue akan mencoba mengkonsumsi setengah dari kapasitas baca tabel. Jika Anda meningkatkan nilai di atas`0.5`, AWS Glue meningkatkan tingkat permintaan; mengurangi nilai di bawah ini `0.5` mengurangi tingkat permintaan baca. (Tingkat baca sebenarnya akan bervariasi, tergantung pada faktor-faktor seperti apakah ada distribusi kunci seragam dalam tabel DynamoDB.)

  - Ketika tabel DynamoDB dalam mode on-demandAWS Glue, menangani kapasitas baca tabel sebagai 40000. Untuk mengekspor tabel besar, sebaiknya ubah tabel DynamoDB Anda ke mode sesuai permintaan.
- `"dynamodb.splits"`: (Opsional) Mendefinisikan berapa banyak bagian partisi yang kita buat untuk tabel DynamoDB ini saat membaca. Default diatur ke "1". Nilai yang dapat diterima adalah dari "1" hingga "1.000.000", inklusif.

  1 mewakili tidak ada paralelisme. Kami sangat menyarankan Anda menentukan nilai yang lebih besar untuk performa yang lebih baik dengan menggunakan rumus di bawah ini. Untuk informasi selengkapnya tentang pengaturan nilai dengan tepat, lihatthe section called “Paralelisme DynamoDB”.
- `"dynamodb.sts.roleArn"`: (Opsional) ARN IAM role yang akan diambil untuk akses lintas-akun. Parameter ini tersedia dalam AWS Glue 1.0 atau yang lebih baru.
- `"dynamodb.sts.roleSessionName"`: (Opsional) Nama sesi STS. Default diatur ke "glue-dynamodb-read-sts-session”. Parameter ini tersedia dalam AWS Glue 1.0 atau yang lebih baru.

“ConnectionType”: “dynamodb” dengan konektor ekspor DynamoDB AWS Glue sebagai sumber

Gunakan opsi koneksi berikut dengan “ConnectionType”: “dynamodb” sebagai sumber, saat menggunakan konektor ekspor AWS Glue DynamoDB, yang hanya tersedia untuk versi 2.0 dan seterusnya: AWS Glue

- `"dynamodb.export"`: (Diperlukan) Nilai string:
  - Jika diatur untuk ddb mengaktifkan konektor ekspor AWS Glue DynamoDB di mana yang `ExportTableToPointInTimeRequest` baru akan dipanggil selama pekerjaan. AWS Glue Ekspor baru akan dihasilkan dengan lokasi yang dilewatkan dari `dynamodb.s3.bucket` dan`dynamodb.s3.prefix`.
  - Jika diatur untuk `s3` mengaktifkan konektor ekspor AWS Glue DynamoDB tetapi melewatkan pembuatan ekspor DynamoDB baru dan sebagai gantinya menggunakan dan `dynamodb.s3.bucket` sebagai lokasi Amazon S3 dari ekspor sebelumnya dari tabel `dynamodb.s3.prefix` tersebut.
- `"dynamodb.tableArn"`: (Wajib) Tabel DynamoDB tempat untuk membaca.
- `"dynamodb.unnestDDBJson"`: (Opsional) Default: false. Nilai yang valid: boolean. Jika disetel ke true, melakukan transformasi unnest dari struktur DynamoDB JSON yang hadir dalam ekspor. Ini adalah kesalahan untuk mengatur `"dynamodb.unnestDDBJson"` dan `"dynamodb.simplifyDDBJson"` menjadi true pada saat yang sama. Di AWS Glue 3.0 dan

versi yang lebih baru, kami sarankan Anda menggunakan `"dynamodb.simplifyDDBJson"` untuk perilaku yang lebih baik saat menyederhanakan jenis Peta DynamoDB. Untuk informasi selengkapnya, lihat the section called “Menyederhanakan penggunaan DynamoDB ekspor JSON”.

- `"dynamodb.simplifyDDBJson"`: (Opsional) Default: false. Nilai yang valid: boolean. Jika disetel ke true, melakukan transformasi untuk menyederhanakan skema struktur DynamoDB JSON yang hadir dalam ekspor. Ini memiliki tujuan yang sama dengan `"dynamodb.unnestDDBJson"` opsi tetapi memberikan dukungan yang lebih baik untuk jenis Peta DynamoDB atau bahkan jenis Peta bersarang di tabel DynamoDB Anda. Opsi ini tersedia di AWS Glue 3.0 dan versi yang lebih baru. Ini adalah kesalahan untuk mengatur `"dynamodb.unnestDDBJson"` dan `"dynamodb.simplifyDDBJson"` menjadi true pada saat yang sama. Untuk informasi selengkapnya, lihat the section called “Menyederhanakan penggunaan DynamoDB ekspor JSON”.
- `"dynamodb.s3.bucket"`: (Opsional) Menunjukkan lokasi bucket Amazon S3 di mana proses DynamoDB akan dilakukan`ExportTableToPointInTime`. Format file untuk ekspor adalah DynamoDB JSON.
  - `"dynamodb.s3.prefix"`: (Opsional) Menunjukkan lokasi awalan Amazon S3 di dalam bucket Amazon S3 tempat pemuatan DynamoDB akan disimpan. `ExportTableToPointInTime` Jika tidak `dynamodb.s3.bucket` ada `dynamodb.s3.prefix` atau ditentukan, nilai-nilai ini akan default ke lokasi Direktori Sementara yang ditentukan dalam konfigurasi AWS Glue pekerjaan. Untuk informasi lebih lanjut, lihat Parameter Khusus yang Digunakan oleh AWS Glue.
  - `"dynamodb.s3.bucketOwner"`: Menunjukkan pemilik bucket yang diperlukan untuk akses Amazon S3 lintas akun.
- `"dynamodb.sts.roleArn"`: (Opsional) ARN peran IAM akan diasumsikan untuk akses lintas akun dan/atau akses lintas wilayah untuk tabel DynamoDB. Catatan: ARN peran IAM yang sama akan digunakan untuk mengakses lokasi Amazon S3 yang ditentukan untuk permintaan tersebut. `ExportTableToPointInTime`
- `"dynamodb.sts.roleSessionName"`: (Opsional) Nama sesi STS. Default diatur ke "glue-dynamodb-read-sts-session”.
- `"dynamodb.exportTime"`(Opsional) Nilai yang valid: string yang mewakili instan ISO-8601. A point-in-time di mana ekspor harus dilakukan.
- `"dynamodb.sts.region"`: (Diperlukan jika melakukan panggilan lintas wilayah menggunakan titik akhir regional) Wilayah yang menghosting tabel DynamoDB yang ingin Anda baca.

“ConnectionType”: “dynamodb” dengan konektor ETL sebagai wastafel

Gunakan opsi koneksi berikut dengan `"connectionType": "dynamodb"` sebagai sink:

- `"dynamodb.output.tableName"`: (Wajib) Tabel DynamoDB tempat untuk menulis.
- `"dynamodb.throughput.write.percent"`: (Opsional) Persentase unit kapasitas tulis (WCU) yang akan digunakan. Default diatur ke "0,5". Nilai yang dapat diterima adalah dari "0,1" hingga "1,5", inklusif.
  - `0.5`mewakili tingkat tulis default, yang berarti bahwa AWS Glue akan mencoba untuk mengkonsumsi setengah dari kapasitas tulis tabel. Jika Anda meningkatkan nilai di atas 0,5, AWS Glue meningkatkan tingkat permintaan; menurunkan nilai di bawah 0,5 mengurangi tingkat permintaan tulis. (Tingkat tulis sebenarnya akan bervariasi, tergantung pada faktor-faktor seperti apakah ada distribusi kunci seragam dalam tabel DynamoDB).
  - Ketika tabel DynamoDB dalam mode on-demandAWS Glue, menangani kapasitas tulis tabel sebagai. `40000` Untuk mengimpor tabel besar, sebaiknya ubah tabel DynamoDB Anda ke mode sesuai permintaan.
- `"dynamodb.output.numParallelTasks"`: (Opsional) Mendefinisikan berapa banyak tugas paralel yang menulis ke DynamoDB pada saat yang sama. Digunakan untuk menghitung WCU permisif untuk setiap tugas Spark. Dalam kebanyakan kasus, AWS Glue akan menghitung default yang masuk akal untuk nilai ini. Untuk informasi selengkapnya, lihat the section called “Paralelisme DynamoDB”.
- `"dynamodb.output.retry"`: (Opsional) Mendefinisikan berapa banyak percobaan ulang yang kita lakukan ketika ada `ProvisionedThroughputExceededException` dari DynamoDB. Default diatur ke "10".
- `"dynamodb.sts.roleArn"`: (Opsional) ARN IAM role yang akan diambil untuk akses lintas-akun.
- `"dynamodb.sts.roleSessionName"`: (Opsional) Nama sesi STS. Default diatur ke "glue-dynamodb-write-sts-session”.

Akses lintas akun Lintas wilayah ke tabel DynamoDB

Tugas ETL AWS Glue mendukung akses lintas wilayah dan akses lintas akun ke tabel DynamoDB. AWS Glue Tugas ETL mendukung pembacaan data dari tabel DynamoDB dari Akun AWS yang lain, dan menulis data ke tabel DynamoDB dari Akun AWS yang lain. AWS Glue juga mendukung pembacaan dari tabel DynamoDB di wilayah lain, dan menulis ke tabel DynamoDB di wilayah lain. Bagian ini memberikan petunjuk tentang pengaturan akses, dan menyediakan contoh skrip.

Prosedur dalam bagian ini me-referensi tutorial IAM untuk menciptakan IAM role dan memberikan akses ke peran. Tutorial ini juga membahas asumsi peran, tapi di sini Anda malah akan menggunakan sebuah skrip tugas untuk mengambil peran dalam AWS Glue. Tutorial ini juga berisi

informasi tentang praktik-praktik lintas akun umum. Untuk informasi selengkapnya, lihat Tutorial: Delegasi Akses Lintas Akun AWS Menggunakan IAM Role dalam Panduan Pengguna IAM.

Membuat peran

Ikuti langkah 1 dalam tutorial untuk membuat IAM role di akun A. Ketika menentukan izin peran, Anda dapat memilih untuk melampirkan kebijakan yang ada seperti `AmazonDynamoDBReadOnlyAccess`, atau `AmazonDynamoDBFullAccess` untuk mengizinkan peran untuk membaca/menulis dari dan ke DynamoDB. Contoh berikut menunjukkan cara membuat sebuah peran bernama `DynamoDBCrossAccessRole`, dengan kebijakan izin `AmazonDynamoDBFullAccess`.

Berikan akses ke peran tersebut

Ikuti langkah 2 dalam tutorial di Panduan Pengguna IAM untuk mengizinkan akun B untuk beralih ke peran yang baru dibuat. Contoh berikut membuat sebuah kebijakan baru dengan pernyataan berikut:

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
        "Resource": "<DynamoDBCrossAccessRole's ARN>"
    }
}
```

Kemudian, Anda dapat melampirkan kebijakan ini ke grup/peran/pengguna yang ingin Anda gunakan untuk mengakses DynamoDB.

Asumsikan peran dalam skrip AWS Glue pekerjaan

Sekarang, Anda dapat log in masuk ke akun B dan membuat sebuah tugas AWS Glue. Untuk membuat sebuah tugas, lihat petunjuk di Mengkonfigurasi properti pekerjaan untuk pekerjaan Spark di AWS Glue.

Dalam skrip tugas tersebut, Anda perlu menggunakan parameter `dynamodb.sts.roleArn` untuk mengambil peran `DynamoDBCrossAccessRole`. Dengan asumsi peran ini akan memungkinkan Anda untuk mendapatkan kredensial sementara, yang perlu digunakan untuk mengakses DynamoDB di akun B. Tinjau skrip contoh ini.

Untuk pembacaan lintas akun di seluruh wilayah (konektor ETL):

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame_from_options(
    connection_type="dynamodb",
    connection_options={
    "dynamodb.region": "us-east-1",
    "dynamodb.input.tableName": "test_source",
    "dynamodb.sts.roleArn": "<DynamoDBCrossAccessRole's ARN>"
    }
)
dyf.show()
job.commit()
```

Untuk pembacaan lintas akun di seluruh wilayah (konektor ELT):

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame_from_options(
    connection_type="dynamodb",
    connection_options={
        "dynamodb.export": "ddb",
        "dynamodb.tableArn": "<test_source ARN>",
        "dynamodb.sts.roleArn": "<DynamoDBCrossAccessRole's ARN>"
    }
)
```

```
dyf.show()
job.commit()
```

Untuk pembacaan dan penulisan lintas akun di seluruh wilayah:

```
import sys
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job
from awsglue.utils import getResolvedOptions

args = getResolvedOptions(sys.argv, ["JOB_NAME"])
glue_context= GlueContext(SparkContext.getOrCreate())
job = Job(glue_context)
job.init(args["JOB_NAME"], args)

dyf = glue_context.create_dynamic_frame_from_options(
    connection_type="dynamodb",
    connection_options={
        "dynamodb.region": "us-east-1",
        "dynamodb.input.tableName": "test_source"
    }
)
dyf.show()

glue_context.write_dynamic_frame_from_options(
    frame=dyf,
    connection_type="dynamodb",
    connection_options={
        "dynamodb.region": "us-west-2",
        "dynamodb.output.tableName": "test_sink",
        "dynamodb.sts.roleArn": "<DynamoDBCrossAccessRole's ARN>"
    }
)

job.commit()
```

Koneksi Kinesis

Anda dapat membaca dan menulis ke aliran data Amazon Kinesis menggunakan informasi yang disimpan dalam tabel Katalog Data, atau dengan memberikan informasi untuk mengakses aliran data secara langsung. Anda dapat membaca informasi dari Kinesis menjadi Spark DataFrame, lalu mengubahnya menjadi Glue. AWS DynamicFrame Anda dapat menulis DynamicFrames ke Kinesis

dalam format JSON. Jika Anda langsung mengakses aliran data, gunakan opsi ini untuk memberikan informasi tentang cara mengakses aliran data.

Jika Anda menggunakan `getCatalogSource` atau `create_data_frame_from_catalog` menggunakan catatan dari sumber streaming Kinesis, pekerjaan tersebut memiliki database Katalog Data dan informasi nama tabel, dan dapat menggunakannya untuk mendapatkan beberapa parameter dasar untuk membaca dari sumber streaming Kinesis. Jika Anda menggunakan`getSource`,`getSourceWithFormat`, `createDataFrameFromOptions` atau`create_data_frame_from_options`, Anda harus menentukan parameter dasar ini menggunakan opsi koneksi yang dijelaskan di sini.

Anda dapat menentukan opsi koneksi untuk Kinesis menggunakan argumen berikut untuk metode yang ditentukan di `GlueContext` kelas.

- Skala
  - `connectionOptions`: Gunakan dengan`getSource`,`createDataFrameFromOptions`, `getSink`
  - `additionalOptions`: Gunakan dengan`getCatalogSource`, `getCatalogSink`
  - `options`: Gunakan dengan`getSourceWithFormat`, `getSinkWithFormat`
- Python
  - `connection_options`: Gunakan dengan`create_data_frame_from_options`, `write_dynamic_frame_from_options`
  - `additional_options`: Gunakan dengan`create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog`
  - `options`: Gunakan dengan`getSource`, `getSink`

Untuk catatan dan batasan tentang pekerjaan Streaming ETL, konsultasikanthe section called “Streaming catatan dan batasan ETL”.

Konfigurasikan Kinesis

Untuk terhubung ke aliran data Kinesis dalam pekerjaan AWS Glue Spark, Anda memerlukan beberapa prasyarat:

- Jika membaca, tugas AWS Glue harus memiliki izin IAM tingkat akses Baca ke aliran data Kinesis.
- Jika menulis, pekerjaan AWS Glue harus memiliki izin IAM tingkat akses Tulis ke aliran data Kinesis.

Dalam kasus tertentu, Anda perlu mengkonfigurasi prasyarat tambahan:

- Jika pekerjaan AWS Glue Anda dikonfigurasi dengan koneksi jaringan Tambahan (biasanya untuk terhubung ke kumpulan data lain) dan salah satu koneksi tersebut menyediakan opsi Jaringan VPC Amazon, ini akan mengarahkan pekerjaan Anda untuk berkomunikasi melalui Amazon VPC. Dalam hal ini Anda juga perlu mengonfigurasi aliran data Kinesis Anda untuk berkomunikasi melalui Amazon VPC. Anda dapat melakukan ini dengan membuat titik akhir VPC antarmuka antara VPC Amazon dan aliran data Kinesis Anda. Untuk informasi selengkapnya, lihat Menggunakan Kinesis Data Streams dengan Titik Akhir VPC Antarmuka.
- Saat menentukan Amazon Kinesis Data Streams di akun lain, Anda harus menyiapkan peran dan kebijakan untuk mengizinkan akses lintas akun. Untuk informasi selengkapnya, lihat Contoh: Baca Dari Pengaliran Kinesis di Akun Berbeda.

Untuk informasi lebih lanjut tentang prasyarat pekerjaan Streaming ETL, lihat. the section called “Lowongan kerja Streaming ETL”

Contoh: Membaca dari aliran Kinesis

Contoh: Membaca dari aliran Kinesis

Digunakan bersama denganthe section called “forEachBatch”.

Contoh untuk sumber streaming Amazon Kinesis:

```
kinesis_options =
   { "streamARN": "arn:aws:kinesis:us-east-2:777788889999:stream/fromOptionsStream",
     "startingPosition": "TRIM_HORIZON",
     "inferSchema": "true",
     "classification": "json"
   }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kinesis",
 connection_options=kinesis_options)
```

Contoh: Menulis ke aliran Kinesis

Contoh: Membaca dari aliran Kinesis

Digunakan bersama denganthe section called “forEachBatch”.

Contoh untuk sumber streaming Amazon Kinesis:

```
kinesis_options =
   { "streamARN": "arn:aws:kinesis:us-east-2:777788889999:stream/fromOptionsStream",
     "startingPosition": "TRIM_HORIZON",
     "inferSchema": "true",
     "classification": "json"
   }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kinesis",
 connection_options=kinesis_options)
```

Referensi opsi koneksi Kinesis

Menetapkan opsi koneksi untuk Amazon Kinesis Data Streams.

Gunakan opsi koneksi berikut untuk sumber data streaming Kinesis:

- `"streamARN"`(Wajib) Digunakan untuk Baca/Tulis. ARN dari aliran data Kinesis.
- `"classification"`(Diperlukan untuk dibaca) Digunakan untuk Baca. Format file yang digunakan oleh data dalam catatan. Diperlukan kecuali disediakan melalui Katalog Data.
- `"streamName"`— (Opsional) Digunakan untuk Baca. Nama aliran data Kinesis untuk dibaca. Digunakan dengan`endpointUrl`.
- `"endpointUrl"`— (Opsional) Digunakan untuk Baca. Default: "https://kinesis.us-east-1.amazonaws.com”. AWS Titik akhir dari aliran Kinesis. Anda tidak perlu mengubah ini kecuali Anda terhubung ke wilayah khusus.
- `"partitionKey"`— (Opsional) Digunakan untuk Menulis. Kunci partisi Kinesis digunakan saat memproduksi catatan.
- `"delimiter"`(Opsional) Digunakan untuk Baca. Pemisah nilai yang digunakan saat `classification` CSV. Defaultnya adalah `","`.”
- `"startingPosition"`: (Opsional) Digunakan untuk Baca. Posisi awal dalam aliran data Kinesis untuk membaca data dari. Nilai yang mungkin adalah`"latest"`,, `"trim_horizon"``"earliest"`, atau string Timestamp dalam format UTC dalam pola `yyyy-mm-ddTHH:MM:SSZ` (di mana `Z` mewakili offset zona waktu UTC dengan +/-. Misalnya “2023-04-04T 08:00:00-04:00 “). Nilai default-nya adalah `"latest"`. Catatan: string Timestamp dalam Format UTC untuk hanya didukung untuk `"startingPosition"` AWS Glue versi 4.0 atau yang lebih baru.
- `"failOnDataLoss"`: (Opsional) Gagal pekerjaan jika ada pecahan aktif yang hilang atau kedaluwarsa. Nilai default-nya adalah `"false"`.

- `"awsSTSRoleARN"`: (Opsional) Digunakan untuk Baca/Tulis. Nama Sumber Daya Amazon (ARN) dari peran yang akan diasumsikan menggunakan AWS Security Token Service ()AWS STS. Peran ini harus memiliki izin untuk mendeskripsikan atau membaca operasi rekaman untuk aliran data Kinesis. Anda harus menggunakan parameter ini saat mengakses aliran data di akun yang berbeda. Digunakan bersama dengan`"awsSTSSessionName"`.
- `"awsSTSSessionName"`: (Opsional) Digunakan untuk Baca/Tulis. Pengidentifikasi untuk sesi dengan asumsi peran menggunakan. AWS STS Anda harus menggunakan parameter ini saat mengakses aliran data di akun yang berbeda. Digunakan bersama dengan`"awsSTSRoleARN"`.
- `"awsSTSEndpoint"`: (Opsional) AWS STS Titik akhir yang digunakan saat menghubungkan ke Kinesis dengan peran yang diasumsikan. Ini memungkinkan penggunaan AWS STS titik akhir regional dalam VPC, yang tidak dimungkinkan dengan titik akhir global default.
- `"maxFetchTimeInMs"`: (Opsional) Digunakan untuk Baca. Waktu maksimum yang dihabiskan untuk pelaksana pekerjaan untuk membaca catatan untuk batch saat ini dari aliran data Kinesis, ditentukan dalam milidetik (ms). Beberapa panggilan `GetRecords` API dapat dilakukan dalam waktu ini. Nilai default-nya adalah `1000`.
- `"maxFetchRecordsPerShard"`: (Opsional) Digunakan untuk Baca. Jumlah maksimum catatan yang diambil per pecahan dalam aliran data Kinesis per mikrobatch. Catatan: Klien dapat melampaui batas ini jika pekerjaan streaming telah membaca catatan tambahan dari Kinesis (dalam panggilan get-records yang sama). Jika `maxFetchRecordsPerShard` perlu ketat maka itu harus kelipatan`maxRecordPerRead`. Nilai default-nya adalah `100000`.
- `"maxRecordPerRead"`: (Opsional) Digunakan untuk Baca. Jumlah maksimum catatan untuk diambil dari aliran data Kinesis dalam `getRecords` setiap operasi. Nilai default-nya adalah `10000`.
- `"addIdleTimeBetweenReads"`: (Opsional) Digunakan untuk Baca. Menambahkan penundaan waktu antara dua operasi berturut-turut`getRecords`. Nilai default-nya adalah `"False"`. Opsi ini hanya dapat dikonfigurasi untuk Glue versi 2.0 dan di atasnya.
- `"idleTimeBetweenReadsInMs"`: (Opsional) Digunakan untuk Baca. Waktu tunda minimum antara dua `getRecords` operasi berturut-turut, ditentukan dalam ms. Nilai default-nya adalah `1000`. Opsi ini hanya dapat dikonfigurasi untuk Glue versi 2.0 dan di atasnya.
- `"describeShardInterval"`: (Opsional) Digunakan untuk Baca. Interval waktu minimum antara dua panggilan `ListShards` API untuk skrip Anda untuk mempertimbangkan resharding. Untuk informasi selengkapnya, lihat Strategi Penyerpihan Kembali di Panduan Developer Amazon Kinesis Data Streams. Nilai default-nya adalah `1s`.
- `"numRetries"`: (Opsional) Digunakan untuk Baca. Jumlah maksimum percobaan ulang untuk permintaan API Kinesis Data Streams. Nilai default-nya adalah `3`.

- `"retryIntervalMs"`: (Opsional) Digunakan untuk Baca. Periode waktu pendinginan (ditentukan dalam ms) sebelum mencoba kembali panggilan API Kinesis Data Streams. Nilai default-nya adalah `1000`.
- `"maxRetryIntervalMs"`: (Opsional) Digunakan untuk Baca. Periode waktu pendinginan maksimum (ditentukan dalam ms) antara dua percobaan ulang panggilan API Kinesis Data Streams. Nilai default-nya adalah `10000`.
- `"avoidEmptyBatches"`: (Opsional) Digunakan untuk Baca. Hindari membuat pekerjaan microbatch kosong dengan memeriksa data yang belum dibaca di aliran data Kinesis sebelum batch dimulai. Nilai default-nya adalah `"False"`.
- `"schema"`: (Diperlukan saat InferSchema disetel ke false) Digunakan untuk Baca. Skema yang digunakan untuk memproses muatan. Jika klasifikasi adalah `avro` skema yang disediakan harus dalam format skema Avro. Jika klasifikasi tidak, skema `avro` yang disediakan harus dalam format skema DDL.

  Berikut ini adalah contoh skema.

  Example in DDL schema format

  ```
  `column1` INT, `column2` STRING , `column3` FLOAT
  ```

  Example in Avro schema format

  ```
  {
    "type":"array",
    "items":
    {
      "type":"record",
      "name":"test",
      "fields":
      [
        {
          "name":"_id",
          "type":"string"
        },
        {
          "name":"index",
          "type":
          [
            "int",
            "string",
  ```

```
          "float"
        ]
      }
    ]
  }
}
```

- `"inferSchema"`: (Opsional) Digunakan untuk Baca. Nilai default adalah 'salah'. Jika disetel ke 'true', skema akan terdeteksi saat runtime dari payload di dalamnya. `foreachbatch`
- `"avroSchema"`: (Usang) Digunakan untuk Baca. Parameter yang digunakan untuk menentukan skema data Avro saat format Avro digunakan. Parameter ini sekarang tidak digunakan lagi. Gunakan parameter `schema`.
- `"addRecordTimestamp"`: (Opsional) Digunakan untuk Baca. Ketika opsi ini diatur ke 'true', output data akan berisi kolom tambahan bernama “__src_timestamp” yang menunjukkan waktu ketika catatan terkait diterima oleh aliran. Nilai default adalah 'salah'. Opsi ini didukung di AWS Glue versi 4.0 atau yang lebih baru.
- `"emitConsumerLagMetrics"`: (Opsional) Digunakan untuk Baca. Ketika opsi disetel ke 'true', untuk setiap batch, itu akan memancarkan metrik untuk durasi antara rekaman tertua yang diterima oleh aliran dan waktu tiba. AWS Glue CloudWatch Nama metriknya adalah “glue.driver.streaming. maxConsumerLagInMs”. Nilai default adalah 'salah'. Opsi ini didukung di AWS Glue versi 4.0 atau yang lebih baru.
- `"fanoutConsumerARN"`: (Opsional) Digunakan untuk Baca. ARN dari konsumen aliran Kinesis untuk aliran yang ditentukan dalam. `streamARN` Digunakan untuk mengaktifkan mode fan-out yang disempurnakan untuk koneksi Kinesis Anda. Untuk informasi lebih lanjut tentang mengonsumsi aliran Kinesis dengan fan-out yang ditingkatkan, lihat. the section called “Menggunakan fan-out yang disempurnakan dalam pekerjaan streaming Kinesis”
- `"recordMaxBufferedTime"`— (Opsional) Digunakan untuk Menulis. Default: 1000 (ms). Waktu maksimum sebuah rekaman di-buffer sambil menunggu untuk ditulis.
- `"aggregationEnabled"`— (Opsional) Digunakan untuk Menulis. Default: benar. Menentukan apakah catatan harus dikumpulkan sebelum mengirim mereka ke Kinesis.
- `"aggregationMaxSize"`— (Opsional) Digunakan untuk Menulis. Default: 51200 (byte). Jika catatan lebih besar dari batas ini, itu akan melewati agregator. Catatan Kinesis memberlakukan batas 50KB pada ukuran rekaman. Jika Anda menetapkan ini di luar 50KB, catatan kebesaran akan ditolak oleh Kinesis.
- `"aggregationMaxCount"`— (Opsional) Digunakan untuk Menulis. Standar: 4294967295. Jumlah maksimum item untuk dikemas ke dalam catatan agregat.

- `"producerRateLimit"`— (Opsional) Digunakan untuk Menulis. Default: 150 (%). Membatasi throughput per shard yang dikirim dari satu produsen (seperti pekerjaan Anda), sebagai persentase dari batas backend.
- `"collectionMaxCount"`— (Opsional) Digunakan untuk Menulis. Default: 500. Jumlah maksimum item untuk dikemas ke dalam PutRecords permintaan.
- `"collectionMaxSize"`— (Opsional) Digunakan untuk Menulis. Default: 5242880 (byte). Jumlah maksimum data untuk dikirim dengan PutRecords permintaan.

Menggunakan fan-out yang disempurnakan dalam pekerjaan streaming Kinesis

Konsumen fan-out yang ditingkatkan dapat menerima catatan dari aliran Kinesis dengan throughput khusus yang bisa lebih besar daripada konsumen biasa. Ini dilakukan dengan mengoptimalkan protokol transfer yang digunakan untuk memberikan data kepada konsumen Kinesis, seperti pekerjaan Anda. Untuk informasi lebih lanjut tentang Kinesis Enhanced Fan-Out, lihat dokumentasi Kinesis.

Dalam mode fan-out yang disempurnakan, opsi `maxRecordPerRead` dan `idleTimeBetweenReadsInMs` koneksi tidak lagi berlaku, karena parameter tersebut tidak dapat dikonfigurasi saat menggunakan fan-out yang disempurnakan. Opsi konfigurasi untuk percobaan ulang berfungsi seperti yang dijelaskan.

Gunakan prosedur berikut untuk mengaktifkan dan menonaktifkan fan-out yang disempurnakan untuk pekerjaan streaming Anda. Anda harus mendaftarkan pengguna streaming untuk setiap pekerjaan yang akan menggunakan data dari aliran Anda.

Untuk mengaktifkan peningkatan konsumsi fan-out pada pekerjaan Anda:

1. Daftarkan konsumen streaming untuk pekerjaan Anda menggunakan Kinesis API. Ikuti petunjuk untuk mendaftarkan konsumen dengan fan-out yang disempurnakan menggunakan Kinesis Data Streams API dalam dokumentasi Kinesis. Anda hanya perlu mengikuti langkah pertama - menelepon RegisterStreamConsumer. *`Permintaan Anda harus mengembalikan ARN, ConsumerARN.`*
2. Setel opsi koneksi `fanoutConsumerARN` ke *`consumerArn dalam argumen`* metode koneksi Anda.
3. Mulai ulang pekerjaan Anda.

Untuk menonaktifkan peningkatan konsumsi fan-out pada pekerjaan Anda:

1. Hapus opsi `fanoutConsumerARN` koneksi dari panggilan metode Anda.
2. Mulai ulang pekerjaan Anda.
3. Ikuti instruksi untuk membatalkan pendaftaran konsumen dalam dokumentasi Kinesis. Instruksi ini berlaku untuk konsol, tetapi juga dapat dicapai melalui Kinesis API. Untuk informasi lebih lanjut tentang deregistrasi konsumen streaming melalui Kinesis API, lihat di dokumentasi Kinesis. DeregisterStreamConsumer

Koneksi Amazon S3

Anda dapat menggunakan AWS Glue for Spark untuk membaca dan menulis file di Amazon S3. AWS Glue for Spark mendukung banyak format data umum yang disimpan di Amazon S3 di luar kotak, termasuk CSV, Avro, JSON, Orc, dan Parket. Untuk informasi selengkapnya tentang format data yang didukung, lihatthe section called “Opsi format data”. Setiap format data dapat mendukung serangkaian fitur AWS Glue yang berbeda. Konsultasikan halaman untuk format data Anda untuk spesifikasi dukungan fitur. Selain itu, Anda dapat membaca dan menulis file berversi yang disimpan dalam kerangka data lake Hudi, Iceberg, dan Delta Lake. Untuk informasi selengkapnya tentang kerangka data lake, lihatthe section called “Kerangka data lake”.

Dengan AWS Glue Anda dapat mempartisi objek Amazon S3 Anda ke dalam struktur folder saat menulis, lalu mengambilnya dengan partisi untuk meningkatkan kinerja menggunakan konfigurasi sederhana. Anda juga dapat mengatur konfigurasi untuk mengelompokkan file kecil bersama-sama saat mengubah data Anda untuk meningkatkan kinerja. Anda dapat membaca dan menulis `bzip2` dan `gzip` mengarsipkan di Amazon S3.

Topik

- Mengkonfigurasi koneksi S3
- Referensi opsi koneksi Amazon S3
- Sintaks koneksi usang untuk format data
- Tidak termasuk kelas penyimpanan Amazon S3
- Mengelola partisi untuk output ETL di AWS Glue
- Membaca file input dalam kelompok yang lebih besar
- Amazon VPC endpoint untuk Amazon S3

Mengkonfigurasi koneksi S3

Untuk terhubung ke Amazon S3 dalam pekerjaan AWS Glue with Spark, Anda memerlukan beberapa prasyarat:

- Pekerjaan AWS Glue harus memiliki izin IAM untuk bucket Amazon S3 yang relevan.

Dalam kasus tertentu, Anda perlu mengkonfigurasi prasyarat tambahan:

- Saat mengonfigurasi akses lintas akun, kontrol akses yang sesuai pada bucket Amazon S3.
- Untuk alasan keamanan, Anda dapat memilih untuk merutekan permintaan Amazon S3 Anda melalui Amazon VPC. Pendekatan ini dapat memperkenalkan tantangan bandwidth dan ketersediaan. Untuk informasi selengkapnya, lihat the section called “Amazon VPC endpoint untuk Amazon S3”.

Referensi opsi koneksi Amazon S3

Mengkhususkan koneksi ke Amazon S3.

Karena Amazon S3 mengelola file daripada tabel, selain menentukan properti koneksi yang disediakan dalam dokumen ini, Anda perlu menentukan konfigurasi tambahan tentang jenis file Anda. Anda menentukan informasi ini melalui opsi format data. Untuk informasi selengkapnya tentang opsi format, lihatthe section called “Opsi format data”. Anda juga dapat menentukan informasi ini dengan mengintegrasikan dengan Katalog Data AWS Glue.

Untuk contoh perbedaan antara opsi koneksi dan opsi format, pertimbangkan bagaimana the section called “create_dynamic_frame_from_options” metode ini mengambil`connection_type`,`connection_options`, `format` dan`format_options`. Bagian ini secara khusus membahas parameter yang disediakan untuk`connection_options`.

Gunakan opsi koneksi berikut dengan `"connectionType": "s3"`:

- `"paths"`: (Wajib) Daftar path Amazon S3 tempat untuk membaca.
- `"exclusions"`: (Opsional) Sebuah string yang berisi daftar JSON pola glob dengan gaya Unix untuk mengecualikan. Misalnya, `"[\"**.pdf\"]"` mengecualikan semua file PDF. Untuk informasi selengkapnya tentang sintaks glob yang AWS Glue mendukung, lihat Sertakan dan Kecualikan Pola.

- `"compressionType"`: atau `"compression"`: (Opsional) Menentukan cara kompresi data. Gunakan `"compressionType"` untuk sumber Amazon S3 dan `"compression"` untuk target Amazon S3. Hal ini umumnya tidak diperlukan jika data memiliki sebuah ekstensi file standar. Nilai yang mungkin adalah `"gzip"` dan`"bzip2"`). Format kompresi tambahan dapat didukung untuk format tertentu. Untuk spesifikasi dukungan fitur, lihat halaman format data.
- `"groupFiles"`: (Opsional) Pengelompokan file dalam grup diaktifkan secara default ketika input berisi lebih dari 50.000 file. Untuk mengaktifkan pengelompokan dalam grup dengan jumlah file kurang dari 50.000, tetapkan parameter ini ke `"inPartition"`. Untuk menonaktifkan pengelompokan dalam grup ketika ada lebih dari 50.000 file, tetapkan parameter ini ke `"none"`.
- `"groupSize"`: (Opsional) Ukuran grup target dalam byte. Default-nya dihitung berdasarkan ukuran input data dan ukuran klaster Anda. Ketika ada kurang dari 50.000 file input, `"groupFiles"` harus diatur ke `"inPartition"` agar ini berlaku.
- `"recurse"`: (Opsional) Jika diatur ke BETUL, secara rekursif akan membaca file di semua subdirektori pada path yang ditentukan.
- `"maxBand"`: (Opsional, lanjutan) Opsi ini mengontrol durasi dalam milidetik yang setelahnya pencantuman `s3` kemungkinan akan konsisten. File dengan stempel waktu modifikasi berada dalam `maxBand` milidetik terakhir dilacak secara khusus ketika menggunakan `JobBookmarks` untuk memperhitungkan eventual consistency Amazon S3. Sebagian besar pengguna tidak perlu mengatur opsi ini. Default-nya adalah 900000 milidetik, atau 15 menit.
- `"maxFilesInBand"`: (Opsional, lanjutan) Opsi ini menentukan jumlah maksimum file yang akan disimpan dari `maxBand` detik terakhir. Jika jumlah ini terlampaui, file tambahan akan dilewati dan hanya diproses dalam eksekusi tugas berikutnya. Sebagian besar pengguna tidak perlu mengatur opsi ini.
- `"isFailFast"`: (Opsional) Opsi ini menentukan apakah pekerjaan AWS Glue ETL melempar pengecualian penguraian pembaca. Jika diatur ke `true`, maka tugas akan di fail-fast jika empat kali percobaan tugas Spark gagal untuk mengurai data dengan benar.
- `"catalogPartitionPredicate"`: (Opsional) Digunakan untuk Baca. Isi dari `WHERE` klausa SQL. Digunakan saat membaca dari tabel Katalog Data dengan jumlah partisi yang sangat besar. Mengambil partisi yang cocok dari indeks Katalog Data. Digunakan dengan`push_down_predicate`, opsi pada the section called “create_dynamic_frame_from_catalog” metode (dan metode serupa lainnya). Untuk informasi selengkapnya, lihat the section called “Katalog predikat partisi”.
- `"partitionKeys"`: (Opsional) Digunakan untuk Menulis. Sebuah array string label kolom. AWS Glue akan mempartisi data Anda seperti yang ditentukan oleh konfigurasi ini. Untuk informasi selengkapnya, lihat the section called “Menulis partisi”.

- `"excludeStorageClasses"`: (Opsional) Digunakan untuk Baca. Array string yang menentukan kelas penyimpanan Amazon S3. AWS Glue akan mengecualikan objek Amazon S3 berdasarkan konfigurasi ini. Untuk informasi selengkapnya, lihat the section called “Tidak termasuk kelas penyimpanan Amazon S3”.

Sintaks koneksi usang untuk format data

Format data tertentu dapat diakses menggunakan sintaks jenis koneksi tertentu. Sintaks ini sudah usang. Kami menyarankan Anda menentukan format Anda menggunakan jenis `s3` koneksi dan opsi format yang disediakan the section called “Opsi format data” sebagai gantinya.

“ConnectionType”: “ConnectionType”

Mengkhususkan koneksi ke file yang disimpan di Amazon S3 dalam format file Apache Hive Optimized Row Columnar (ORC).

Gunakan opsi koneksi berikut dengan `"connectionType": "orc"`:

- `paths`: (Wajib) Daftar path Amazon S3 tempat untuk membaca.
- (Pasangan pilihan nama/pasangan lainnya): Setiap opsi tambahan, termasuk opsi format, diberikan langsung ke SparkSQL `DataSource`.

"connectionType": "parquet"

Mengkhususkan koneksi ke file yang disimpan di Amazon S3 dalam format file Apache Parquet.

Gunakan opsi koneksi berikut dengan `"connectionType": "parquet"`:

- `paths`: (Wajib) Daftar path Amazon S3 tempat untuk membaca.
- (Pasangan pilihan nama/pasangan lainnya): Setiap opsi tambahan, termasuk opsi format, diberikan langsung ke SparkSQL `DataSource`.

Tidak termasuk kelas penyimpanan Amazon S3

Jika Anda menjalankan tugas AWS Glue ETL yang membaca file atau partisi dari Amazon Simple Storage Service (Amazon S3), Anda dapat mengecualikan beberapa jenis kelas penyimpanan Amazon S3.

Kelas penyimpanan berikut tersedia di Amazon S3:

- `STANDARD` — Untuk penyimpanan data yang sering diakses untuk tujuan umum.
- `INTELLIGENT_TIERING` — Untuk data dengan pola akses yang berubah-ubah atau tidak diketahui.
- `STANDARD_IA` dan `ONEZONE_IA` — Untuk data yang tahan lama, namun tidak banyak diakses.
- `GLACIER`, `DEEP_ARCHIVE`, dan `REDUCED_REDUNDANCY` — Untuk arsip jangka panjang dan pelestarian digital.

Untuk informasi selengkapnya, lihat Kelas Penyimpanan Amazon S3 dalam Panduan Developer Amazon S3.

Contoh dalam bagian ini menunjukkan cara mengecualikan kelas penyimpanan `GLACIER` dan `DEEP_ARCHIVE`. Kelas-kelas ini memungkinkan Anda untuk mencantumkan file, tetapi tidak akan membiarkan Anda membaca file kecuali mereka dipulihkan. (Untuk informasi lebih lanjut, lihat Mengembalikan Objek yang Diarsipkan di Panduan Developer Amazon S3.)

Dengan menggunakan pengecualian kelas penyimpanan, Anda dapat memastikan bahwa AWS Glue pekerjaan Anda akan bekerja pada tabel yang memiliki partisi di seluruh tingkatan kelas penyimpanan ini. Tanpa pengecualian, tugas yang membaca data dari tingkatan ini gagal dengan kesalahan berikut: Amazons3Exception: operasi ini tidak berlaku untuk kelas penyimpanan objek.

Ada berbagai cara untuk memfilter kelas penyimpanan Amazon S3. AWS Glue

Topik

- Tidak termasuk kelas penyimpanan Amazon S3 saat membuat Bingkai Dinamis
- Tidak termasuk kelas penyimpanan Amazon S3 pada tabel Katalog Data

Tidak termasuk kelas penyimpanan Amazon S3 saat membuat Bingkai Dinamis

Untuk mengecualikan kelas penyimpanan Amazon S3 saat membuat bingkai dinamis, gunakan `excludeStorageClasses` di. `additionalOptions` AWS Gluesecara otomatis menggunakan `Lister` implementasi Amazon S3 sendiri untuk membuat daftar dan mengecualikan file yang sesuai dengan kelas penyimpanan yang ditentukan.

Contoh Python dan Scala berikut menunjukkan bagaimana cara untuk mengecualikan kelas penyimpanan `GLACIER` dan `DEEP_ARCHIVE` saat membuat sebuah bingkai dinamis.

Contoh Python:

```
glueContext.create_dynamic_frame.from_catalog(
    database = "my_database",
    tableName = "my_table_name",
    redshift_tmp_dir = "",
    transformation_ctx = "my_transformation_context",
    additional_options = {
        "excludeStorageClasses" : ["GLACIER", "DEEP_ARCHIVE"]
    }
)
```

Contoh scala:

```
val* *df = glueContext.getCatalogSource(
    nameSpace, tableName, "", "my_transformation_context",
    additionalOptions = JsonOptions(
        Map("excludeStorageClasses" -> List("GLACIER", "DEEP_ARCHIVE"))
    )
).getDynamicFrame()
```

Tidak termasuk kelas penyimpanan Amazon S3 pada tabel Katalog Data

Anda dapat menentukan pengecualian kelas penyimpanan yang akan digunakan oleh pekerjaan AWS Glue ETL sebagai parameter tabel di AWS Glue Data Catalog. Anda dapat menyertakan parameter ini di operasi `CreateTable` dengan menggunakan AWS Command Line Interface (AWS CLI) atau secara pemrograman menggunakan API. Untuk informasi lebih lanjut, lihat Struktur Tabel dan CreateTable.

Anda juga dapat menentukan kelas penyimpanan yang dikecualikan di AWS Glue konsol.

Untuk mengecualikan kelas penyimpanan Amazon S3 (konsol)

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Pada panel navigasi di sebelah kiri, pilih Tabel.
3. Pilih nama tabel dalam daftar, kemudian pilih Edit tabel.
4. Di Properti tabel, tambahkan **`excludeStorageClasses`** sebagai kunci dan **`[\"GLACIER\", \"DEEP_ARCHIVE\"]`** sebagai nilai.
5. Pilih Apply (Terapkan).

Mengelola partisi untuk output ETL di AWS Glue

Pemartisian merupakan teknik penting untuk mengatur set data sehingga mereka dapat di-kueri secara efisien. Ia mengatur data dalam sebuah struktur direktori hirarkis berdasarkan nilai-nilai yang berbeda dari satu atau beberapa kolom.

Misalnya, Anda dapat memutuskan untuk melakukan partisi pada log aplikasi Anda di Amazon Simple Storage Service (Amazon S3) berdasarkan tanggal, dirinci berdasarkan tahun, bulan, dan hari. File yang sesuai dengan data satu hari, kemudian ditempatkan dengan prefiks seperti `s3://my_bucket/logs/year=2018/month=01/day=23/`. Sistem seperti Amazon Athena, Amazon Redshift Spectrum, dan AWS Glue sekarang dapat menggunakan partisi ini untuk memfilter data berdasarkan nilai partisi tanpa harus membaca semua data yang mendasarinya dari Amazon S3.

Crawler tidak hanya menyimpulkan jenis file dan skema, crawler juga secara otomatis mengidentifikasi struktur partisi set data Anda ketika mereka mengisi Katalog Data Glue AWS. Kolom partisi yang dihasilkan tersedia untuk kueri dalam pekerjaan AWS Glue ETL atau mesin kueri seperti Amazon Athena.

Setelah melakukan perayapan pada tabel, Anda dapat melihat partisi yang dibuat crawler. Di AWS Glue konsol, pilih Tabel di panel navigasi kiri. Pilih tabel yang dibuat oleh crawler, lalu pilih Lihat partisi.

Untuk path yang dipartisi dengan gaya Apache Hive di `key=val`, crawler secara otomatis mengisi nama kolom menggunakan nama kunci. Jika tidak, ia menggunakan nama default seperti `partition_0`, `partition_1`, dan sebagainya. Anda dapat mengubah nama default di konsol. Untuk melakukannya, arahkan ke tabel. Periksa apakah indeks ada di bawah tab Indeks. Jika itu masalahnya, Anda perlu menghapusnya untuk melanjutkan (Anda dapat membuatnya kembali menggunakan nama kolom baru setelahnya). Kemudian, pilih Edit Skema, dan ubah nama kolom partisi di sana.

Dalam skrip ETL Anda, Anda kemudian dapat mem-filter pada kolom partisi. Karena informasi partisi disimpan dalam Katalog Data, gunakan panggilan API `from_catalog` untuk menyertakan kolom partisi di `DynamicFrame`. Misalnya, gunakan `create_dynamic_frame.from_catalog` sebagai ganti dari `create_dynamic_frame.from_options`.

Partisi adalah teknik optimasi yang mengurangi pemindaian data. Untuk informasi lebih lanjut tentang proses mengidentifikasi kapan teknik ini tepat, lihat Mengurangi jumlah pemindaian data dalam Panduan Pekerjaan AWS Glue for Apache Spark Praktik Terbaik untuk Penyetelan Kinerja di AWS Panduan Preskriptif.

Pra-penyaringan menggunakan predikat pushdown

Dalam banyak kasus, Anda dapat menggunakan predikat pushdown untuk mem-filter pada partisi tanpa harus mencantumkan dan membaca semua file dalam set data Anda. Alih-alih membaca seluruh kumpulan data dan kemudian memfilter di a DynamicFrame, Anda dapat menerapkan filter langsung pada metadata partisi di Katalog Data. Kemudian Anda hanya membuat daftar dan membaca apa yang sebenarnya Anda butuhkan ke dalam DynamicFrame.

Misalnya, dengan Python, Anda bisa menulis yang berikut ini.

```
glue_context.create_dynamic_frame.from_catalog(
    database = "my_S3_data_set",
    table_name = "catalog_data_table",
    push_down_predicate = my_partition_predicate)
```

Ini menciptakan sebuah DynamicFrame yang memuat hanya partisi dalam Katalog Data yang memenuhi ekspresi predikat. Tergantung pada seberapa kecil subset data yang Anda muat, hal ini dapat menghemat banyak waktu pemrosesan.

Ekspresi predikat dapat berupa ekspresi Boolean yang didukung oleh Spark SQL. Apa pun yang Anda bisa masukkan ke dalam klausul `WHERE` dalam kueri Spark SQL akan bisa digunakan. Misalnya, ekspresi predikat `pushDownPredicate = "(year=='2017' and month=='04')"` memuat hanya partisi dalam Katalog Data yang memiliki `year` sama dengan 2017 dan `month` sama dengan 04. Untuk informasi lebih lanjut, lihat dokumentasi Apache Spark SQL, dan khususnya, referensi fungsi Scala SQL.

Pemfilteran sisi server menggunakan predikat partisi katalog

`push_down_predicate`Opsi ini diterapkan setelah mencantumkan semua partisi dari katalog dan sebelum mencantumkan file dari Amazon S3 untuk partisi tersebut. Jika Anda memiliki banyak partisi untuk sebuah tabel, daftar partisi katalog masih dapat menimbulkan overhead waktu tambahan. Untuk mengatasi overhead ini, Anda dapat menggunakan pemangkasan partisi sisi server dengan `catalogPartitionPredicate` opsi yang menggunakan indeks partisi di Katalog Data Glue. AWS Ini membuat pemfilteran partisi jauh lebih cepat ketika Anda memiliki jutaan partisi dalam satu tabel. Anda dapat menggunakan keduanya `push_down_predicate` dan `catalogPartitionPredicate` `additional_options` bersama-sama jika Anda `catalogPartitionPredicate` memerlukan sintaks predikat yang belum didukung dengan indeks partisi katalog.

Python:

```
dynamic_frame = glueContext.create_dynamic_frame.from_catalog(
    database=dbname,
    table_name=tablename,
    transformation_ctx="datasource0",
    push_down_predicate="day>=10 and customer_id like '10%'",
    additional_options={"catalogPartitionPredicate":"year='2021' and month='06'"}
)
```

Scala:

```
val dynamicFrame = glueContext.getCatalogSource(
    database = dbname,
    tableName = tablename,
    transformationContext = "datasource0",
    pushDownPredicate="day>=10 and customer_id like '10%'",
    additionalOptions = JsonOptions("""{
        "catalogPartitionPredicate": "year='2021' and month='06'"}""")
    ).getDynamicFrame()
```

> **Note**
>
> `push_down_predicate`Dan `catalogPartitionPredicate` menggunakan sintaks yang berbeda. Yang pertama menggunakan sintaks standar Spark SQL dan yang kemudian menggunakan parser JSQL.

Menulis partisi

Secara default, a tidak DynamicFrame dipartisi saat ditulis. Semua file output ditulis di tingkat atas dari path output yang ditentukan. Sampai saat ini, satu-satunya cara untuk menulis DynamicFrame menjadi partisi adalah mengubahnya menjadi Spark SQL DataFrame sebelum menulis.

Namun, DynamicFrames sekarang mendukung partisi asli menggunakan urutan kunci, menggunakan `partitionKeys` opsi saat Anda membuat wastafel. Sebagai contoh, kode Python berikut menuliskan set data ke Amazon S3 dalam format Parquet, ke direktori yang dipartisi oleh bidang jenis. Dari sana, Anda dapat memproses partisi ini dengan menggunakan sistem lain, seperti Amazon Athena.

```
glue_context.write_dynamic_frame.from_options(
    frame = projectedEvents,
```

```
    connection_type = "s3",
    connection_options = {"path": "$outpath", "partitionKeys": ["type"]},
    format = "parquet")
```

Membaca file input dalam kelompok yang lebih besar

Anda dapat mengatur properti tabel Anda untuk memungkinkan tugas ETL AWS Glue untuk mengelompokkan file ketika dibaca dari penyimpanan data Amazon S3. Properti ini memungkinkan setiap tugas ETL untuk membaca sekelompok file input ke dalam partisi di memori tunggal, ini sangat berguna ketika ada sejumlah besar file kecil di Penyimpanan data Amazon S3 Anda. Ketika Anda mengatur properti tertentu, Anda menginstruksikan AWS Glue untuk mengelompokkan file dalam partisi data Amazon S3 dan mengatur ukuran grup yang akan dibaca. Anda juga dapat mengatur pilihan ini ketika membaca dari penyimpanan data Amazon S3 dengan metode `create_dynamic_frame.from_options`.

Untuk mengaktifkan pengelompokan file untuk sebuah tabel, Anda mengatur pasangan nilai-kunci di bidang parameter struktur tabel Anda. Gunakan notasi JSON untuk menetapkan nilai untuk bidang parameter tabel Anda. Untuk informasi lebih lanjut tentang mengedit properti tabel, lihat Melihat dan mengedit detail tabel.

Anda dapat menggunakan metode ini untuk mengaktifkan pengelompokan untuk tabel di Katalog Data dengan menyimpan data Amazon S3.

groupFiles

Atur groupFiles ke `inPartition` untuk mengaktifkan pengelompokan file dalam partisi data Amazon S3. AWS Glue secara otomatis memungkinkan pengelompokan jika ada lebih dari 50.000 file input, seperti dalam contoh berikut.

```
'groupFiles': 'inPartition'
```

groupSize

Atur groupSize untuk ukuran target grup dalam byte. Properti groupSize bersifat opsional, jika tidak disediakan, AWS Glue menghitung ukuran untuk menggunakan semua core CPU di klaster sekaligus masih mengurangi jumlah keseluruhan tugas ETL dan partisi di memori.

Sebagai contoh, berikut ini menetapkan ukuran grup ke 1 MB.

```
'groupSize': '1048576'
```

Perhatikan bahwa `groupsize` harus diatur dengan hasil perhitungan. Sebagai contoh 1024 * 1024 = 1048576.

rekursi

Atur rekursi ke `True` untuk secara rekursif membaca file di semua subdirektori saat menentukan `paths` sebagai array path. Anda tidak perlu mengatur recurse if `paths` adalah array kunci objek di Amazon S3, atau jika format input parquet/orc, seperti pada contoh berikut.

```
'recurse':True
```

Jika Anda membaca dari Amazon S3 secara langsung menggunakan metode `create_dynamic_frame.from_options`, tambahkan opsi koneksi ini. Sebagai contoh, upaya berikut untuk mengelompokkan file ke dalam grup 1 MB.

```
df = glueContext.create_dynamic_frame.from_options("s3", {'paths': ["s3://s3path/"],
 'recurse':True, 'groupFiles': 'inPartition', 'groupSize': '1048576'}, format="json")
```

> **Note**
>
> `groupFiles`didukung untuk DynamicFrames dibuat dari format data berikut: csv, ion, GrokLog, json, dan xml. Opsi ini tidak didukung untuk avro, parket, dan orc.

Amazon VPC endpoint untuk Amazon S3

Untuk alasan keamanan, banyak pelanggan AWS menjalankan aplikasinya dalam lingkungan Amazon Virtual Private Cloud (Amazon VPC). Dengan Amazon VPC, Anda dapat meluncurkan instans Amazon EC2 ke virtual private cloud, yang secara logis terisolasi dari jaringan lain—termasuk internet publik. Dengan Amazon VPC, Anda dapat mengontrol rentang alamat IP, subnet, tabel perutean, gateway jaringan, dan pengaturan keamanan.

> **Note**
>
> Jika Anda membuat akun AWS setelah 2013-12-04, Anda sudah memiliki VPC default di setiap Wilayah AWS. Anda dapat segera mulai menggunakan VPC default Anda tanpa perlu konfigurasi tambahan.
> Untuk informasi lebih lanjut, lihat VPC dan Subnet Default Anda di Panduan Pengguna Amazon VPC.

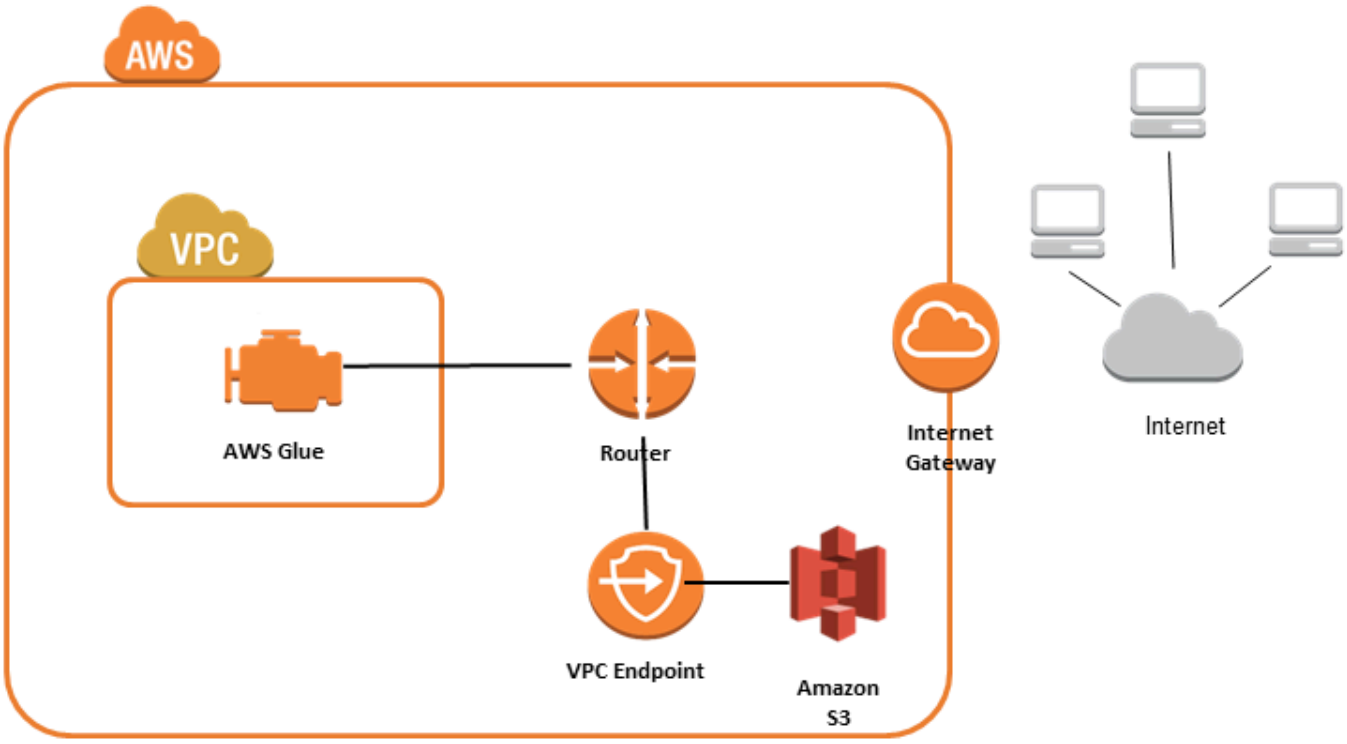
Banyak pelanggan memiliki masalah privasi dan keamanan yang sah tentang cara mengirim dan menerima data di internet publik. Para pelanggan dapat mengatasi masalah ini dengan menggunakan virtual private network (VPN) untuk merutekan semua lalu lintas jaringan Amazon S3 melalui infrastruktur jaringan perusahaan mereka sendiri. Namun, pendekatan ini dapat memunculkan tantangan bandwidth dan ketersediaan.

VPC endpoint untuk Amazon S3 dapat meringankan tantangan ini. Sebuah VPC endpoint untuk Amazon S3 memungkinkan AWS Glue untuk menggunakan alamat IP privat guna mengakses Amazon S3 tanpa terbuka ke internet publik. AWS Glue tidak memerlukan alamat IP privat, dan Anda tidak memerlukan sebuah gateway internet, perangkat NAT, atau virtual private gateway di VPC Anda. Anda menggunakan kebijakan titik akhir guna mengendalikan akses ke Amazon S3. Lalu lintas antara VPC Anda dan layanan AWS tidak meninggalkan jaringan Amazon.

Ketika Anda membuat VPC endpoint untuk Amazon S3, setiap permintaan untuk titik akhir Amazon S3 dalam Wilayah (misalnya, s3.us-west-2.amazonaws.com) dirutekan ke titik akhir Amazon S3 privat dalam jaringan Amazon. Anda tidak perlu memodifikasi aplikasi Anda yang berjalan di instans Amazon EC2 di VPC Anda— nama titik akhir tetap sama, tetapi rute ke Amazon S3 tetap sepenuhnya dalam jaringan Amazon, dan tidak mengakses internet publik.

Untuk informasi selengkapnya tentang VPC endpoint, lihat VPC Endpoint dalam Panduan Pengguna Amazon VPC.

Diagram berikut menunjukkan bagaimana AWS Glue dapat menggunakan VPC endpoint untuk mengakses Amazon S3.



Untuk menyiapkan akses untuk Amazon S3

1. Masuk ke AWS Management Console dan buka konsol Amazon VPC di https://console.aws.amazon.com/vpc/.
2. Pada panel navigasi kiri, pilih Titik Akhir.
3. Pilih Buat titik akhir, dan ikuti langkah-langkah untuk membuat VPC endpoint akhir Amazon S3 jenis Gateway.

Koneksi Amazon DocumentDB

Anda dapat menggunakan AWS Glue for Spark untuk membaca dan menulis ke tabel di Amazon DocumentDB. Anda dapat terhubung ke Amazon DocumentDB menggunakan kredensil yang disimpan melalui koneksi Glue. AWS Secrets Manager AWS

Untuk informasi selengkapnya tentang Amazon DocumentDB, lihat dokumentasi Amazon DocumentDB.

> **Note**
>
> Cluster elastis Amazon DocumentDB saat ini tidak didukung saat menggunakan konektor Glue. AWS Untuk informasi selengkapnya tentang cluster elastis, lihat Menggunakan cluster elastis Amazon DocumentDB.

Membaca dan menulis ke koleksi Amazon DocumentDB

> **Note**
>
> Bila Anda membuat tugas ETL yang menghubungkan ke Amazon DocumentDB, untuk properti tugas `Connections`, Anda harus mengkhususkan sebuah objek koneksi yang menentukan virtual private cloud (VPC) di mana Amazon DocumentDB berjalan. Untuk objek koneksi, jenis koneksinya harus `JDBC`, dan `JDBC URL` harus `mongo://<DocumentDB_host>:27017`.

> **Note**
>
> Sampel kode ini dikembangkan untuk AWS Glue 3.0. Untuk bermigrasi ke AWS Glue 4.0, konsultasikanthe section called “MongoDB”. `uri`Parameter telah berubah.

> **Note**
>
> Saat menggunakan Amazon DocumentDB`retryWrites`, harus disetel ke false dalam situasi tertentu, seperti saat dokumen yang ditulis menentukan. `_id` Untuk informasi lebih lanjut, lihat Perbedaan Fungsional dengan MongoDB di dokumentasi Amazon DocumentDB.

Skrip Python berikut menunjukkan menggunakan jenis koneksi dan opsi koneksi untuk membaca dan menulis ke Amazon DocumentDB.

```
import sys
```

```
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext, SparkConf
from awsglue.context import GlueContext
from awsglue.job import Job
import time

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

job = Job(glueContext)
job.init(args['JOB_NAME'], args)

output_path = "s3://some_bucket/output/" + str(time.time()) + "/"
documentdb_uri = "mongodb://<mongo-instanced-ip-address>:27017"
documentdb_write_uri = "mongodb://<mongo-instanced-ip-address>:27017"

read_docdb_options = {
    "uri": documentdb_uri,
    "database": "test",
    "collection": "coll",
    "username": "username",
    "password": "1234567890",
    "ssl": "true",
    "ssl.domain_match": "false",
    "partitioner": "MongoSamplePartitioner",
    "partitionerOptions.partitionSizeMB": "10",
    "partitionerOptions.partitionKey": "_id"
}

write_documentdb_options = {
    "retryWrites": "false",
    "uri": documentdb_write_uri,
    "database": "test",
    "collection": "coll",
    "username": "username",
    "password": "pwd"
}

# Get DynamicFrame from  DocumentDB
```

```
dynamic_frame2 =
 glueContext.create_dynamic_frame.from_options(connection_type="documentdb",

 connection_options=read_docdb_options)

# Write DynamicFrame to MongoDB and DocumentDB
glueContext.write_dynamic_frame.from_options(dynamic_frame2,
 connection_type="documentdb",

 connection_options=write_documentdb_options)

job.commit()
```

Skrip Scala berikut menunjukkan menggunakan jenis koneksi dan opsi koneksi untuk membaca dan menulis ke Amazon DocumentDB.

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamicFrame
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
  val DOC_URI: String = "mongodb://<mongo-instanced-ip-address>:27017"
  val DOC_WRITE_URI: String = "mongodb://<mongo-instanced-ip-address>:27017"
  lazy val documentDBJsonOption = jsonOptions(DOC_URI)
  lazy val writeDocumentDBJsonOption = jsonOptions(DOC_WRITE_URI)
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    // Get DynamicFrame from DocumentDB
    val resultFrame2: DynamicFrame = glueContext.getSource("documentdb",
 documentDBJsonOption).getDynamicFrame()

    // Write DynamicFrame to DocumentDB
    glueContext.getSink("documentdb", writeJsonOption).writeDynamicFrame(resultFrame2)
```

```
    Job.commit()
  }

  private def jsonOptions(uri: String): JsonOptions = {
    new JsonOptions(
      s"""{"uri": "${uri}",
         |"database":"test",
         |"collection":"coll",
         |"username": "username",
         |"password": "pwd",
         |"ssl":"true",
         |"ssl.domain_match":"false",
         |"partitioner": "MongoSamplePartitioner",
         |"partitionerOptions.partitionSizeMB": "10",
         |"partitionerOptions.partitionKey": "_id"}""".stripMargin)
  }
}
```

Referensi opsi koneksi Amazon DocumentDB

Mengkhususkan koneksi ke Amazon DocumentDB (dengan kompatibilitas MongoDB).

Pilihan koneksi berbeda untuk koneksi sumber dan koneksi sink.

“ConnectionType”: “Documentdb” sebagai sumber

Gunakan opsi koneksi berikut dengan `"connectionType": "documentdb"` sebagai sumber:

- `"uri"`: (Wajib) Host Amazon DocumentDB tempat untuk membaca, diformat sebagai `mongodb://<host>:<port>`.
- `"database"`: (Wajib) Basis data Amazon DocumentDB tempat untuk membaca.
- `"collection"`: (Wajib) Koleksi Amazon DocumentDB tempat untuk membaca.
- `"username"`: (Wajib) Nama pengguna Amazon DocumentDB.
- `"password"`: (Wajib) Kata sandi Amazon DocumentDB.
- `"ssl"`: (Wajib jika menggunakan SSL) Jika koneksi Anda menggunakan SSL, maka Anda harus menyertakan opsi ini dengan nilai `"true"`.
- `"ssl.domain_match"`: (Wajib jika menggunakan SSL) Jika koneksi Anda menggunakan SSL, maka Anda harus menyertakan opsi ini dengan nilai `"false"`.

- `"batchSize"`: (Opsional): Jumlah dokumen yang akan dikembalikan per batch, digunakan dalam kursor batch internal.
- `"partitioner"`: (Opsional): Nama kelas pemartisi untuk membaca input data dari Amazon DocumentDB. Konektor menyediakan pemartisi berikut:
  - `MongoDefaultPartitioner`(default) (Tidak didukung di AWS Glue 4.0)
  - `MongoSamplePartitioner`(Tidak didukung di AWS Glue 4.0)
  - `MongoShardedPartitioner`
  - `MongoSplitVectorPartitioner`
  - `MongoPaginateByCountPartitioner`
  - `MongoPaginateBySizePartitioner`(Tidak didukung di AWS Glue 4.0)
- `"partitionerOptions"` (Opsional): Opsi untuk pemartisi yang ditunjuk. Opsi berikut didukung untuk setiap pemartisi:
  - `MongoSamplePartitioner`: `partitionKey`, `partitionSizeMB`, `samplesPerPartition`
  - `MongoShardedPartitioner`: `shardkey`
  - `MongoSplitVectorPartitioner`: `partitionKey`, partitionSizeMB
  - `MongoPaginateByCountPartitioner`: `partitionKey`, `numberOfPartitions`
  - `MongoPaginateBySizePartitioner`: `partitionKey`, partitionSizeMB

  Untuk informasi lebih lanjut tentang opsi ini, lihat Konfigurasi Partisi dalam dokumentasi MongoDB.

“ConnectionType”: “Documentdb” sebagai wastafel

Gunakan opsi koneksi berikut dengan `"connectionType": "documentdb"` sebagai sink:

- `"uri"`: (Wajib) Host Amazon DocumentDB tempat untuk menulis, diformat sebagai `mongodb://<host>:<port>`.
- `"database"`: (Wajib) Basis data Amazon DocumentDB tempat untuk menulis.
- `"collection"`: (Wajib) Koleksi Amazon DocumentDB tempat untuk menulis.
- `"username"`: (Wajib) Nama pengguna Amazon DocumentDB.
- `"password"`: (Wajib) Kata sandi Amazon DocumentDB.
- `"extendedBsonTypes"`: (Opsional) Jika `true`, memungkinkan jenis BSON diperpanjang saat menulis data ke Amazon DocumentDB. Defaultnya adalah `true`.

- `"replaceDocument"`: (Opsional) Jika `true`, menggantikan seluruh dokumen ketika menyimpan set data yang berisi bidang `_id`. Jika `false`, hanya bidang dalam dokumen yang cocok dengan bidang dalam set data saja yang diperbarui. Defaultnya adalah `true`.
- `"maxBatchSize"`: (Opsional): Ukuran batch maksimum untuk operasi massal saat menyimpan data. Default-nya adalah 512.
- `"retryWrites"`: (Opsional): Secara otomatis mencoba kembali operasi penulisan tertentu satu kali jika AWS Glue menemukan kesalahan jaringan.

OpenSearch Koneksi layanan

Anda dapat menggunakan AWS Glue for Spark untuk membaca dan menulis ke tabel di OpenSearch Service in AWS Glue 4.0 dan versi yang lebih baru. Anda dapat menentukan apa yang harus dibaca dari OpenSearch Layanan dengan OpenSearch kueri. Anda terhubung ke OpenSearch Layanan menggunakan kredensi otentikasi dasar HTTP yang disimpan melalui koneksi GlueAWS. AWS Secrets Manager Fitur ini tidak kompatibel dengan OpenSearch Layanan tanpa server.

Untuk informasi selengkapnya tentang OpenSearch Layanan Amazon, lihat dokumentasi OpenSearch Layanan Amazon.

Mengkonfigurasi koneksi OpenSearch Layanan

Untuk terhubung ke OpenSearch Layanan dari AWS Glue, Anda harus membuat dan menyimpan kredensi OpenSearch Layanan Anda secara AWS Secrets Manager rahasia, lalu mengaitkan rahasia itu dengan koneksi OpenSearch AWS Glue Layanan.

Prasyarat:

- Identifikasi titik akhir domain, *`AoSendPoint`* dan port, *`AoSport`* yang ingin Anda baca, atau buat sumber daya dengan mengikuti petunjuk dalam dokumentasi Layanan Amazon. OpenSearch Untuk informasi selengkapnya tentang membuat domain, lihat Membuat dan mengelola domain OpenSearch Layanan Amazon di dokumentasi OpenSearch Layanan Amazon.

  Titik akhir domain OpenSearch Layanan Amazon akan memiliki formulir default berikut, https:// search - *`domainName`* -. *`unstructuredIdContent wilayah`* .es.amazonaws.com. Untuk informasi selengkapnya tentang mengidentifikasi titik akhir domain Anda, lihat Membuat dan mengelola domain OpenSearch Layanan Amazon di dokumentasi OpenSearch Layanan Amazon.

  `Identifikasi atau hasilkan kredensi otentikasi dasar HTTP, AOSuser, dan AOSPassWord untuk domain Anda.`

Untuk mengkonfigurasi koneksi ke OpenSearch Layanan:

1. DiAWS Secrets Manager, buat rahasia menggunakan kredensyal OpenSearch Layanan Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

   - *Saat memilih **pasangan kunci/nilai**, buat pasangan untuk kunci opensearch.net.http.auth.user dengan nilai AOSUSER.*
   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci opensearch.net.http.auth.pass dengan nilai AOSPassword.*

2. Di konsol AWS Glue, buat koneksi dengan mengikuti langkah-langkah dithe section called "Menambahkan AWS Glue koneksi". Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk penggunaan masa depan di GlueAWS.

   - Saat memilih jenis Koneksi, pilih OpenSearch Layanan.
   - Saat memilih titik akhir Domain, berikan *AoSendPoint*.
   - Saat memilih port, sediakan *AOsport*.
   - Saat memilih AWSSecret, berikan *secretName*.

Setelah membuat koneksi AWS Glue OpenSearch Service, Anda harus melakukan langkah-langkah berikut sebelum menjalankan pekerjaan AWS Glue Anda:

- Berikan peran IAM yang terkait dengan izin pekerjaan AWS Glue Anda untuk membaca *secretName*.
- Dalam konfigurasi pekerjaan AWS Glue Anda, berikan *ConnectionName* sebagai koneksi jaringan Tambahan.

Membaca dari indeks OpenSearch Layanan

Prasyarat:

- Indeks OpenSearch Layanan yang ingin Anda baca, *AOsIndex*.
- Koneksi AWS Glue OpenSearch Service yang dikonfigurasi untuk memberikan informasi autentikasi dan lokasi jaringan. Untuk mendapatkan ini, selesaikan langkah-langkah dalam

prosedur sebelumnya, Untuk mengkonfigurasi koneksi ke OpenSearch Layanan. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.

Contoh ini membaca indeks dari Amazon OpenSearch Service. Anda harus memberikan `pushdown` parameternya.

Sebagai contoh:

```
opensearch_read = glueContext.create_dynamic_frame.from_options(
    connection_type="opensearch",
    connection_options={
        "connectionName": "connectionName",
        "opensearch.resource": "aosIndex",
        "pushdown": "true",
    }
)
```

Anda juga dapat memberikan string kueri untuk memfilter hasil yang dikembalikan dalam file Anda DynamicFrame. Anda perlu mengkonfigurasi`opensearch.query`.

`opensearch.query`*dapat mengambil parameter kueri URL string QueryString atau kueri DSL JSON objek QueryObject.* Untuk informasi selengkapnya tentang kueri DSL, lihat Kueri DSL di dokumentasi. OpenSearch Untuk menyediakan string parameter kueri URL, tambahkan ?q= ke kueri Anda, seperti yang Anda lakukan di URL yang sepenuhnya memenuhi syarat. Untuk menyediakan objek DSL kueri, string melarikan diri dari objek JSON sebelum menyediakannya.

Sebagai contoh:

```
            queryObject = "{ "query": { "multi_match": { "query": "Sample", "fields":
[ "sample" ] } } }"
            queryString = "?q=queryString"

            opensearch_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="opensearch",
    connection_options={
        "connectionName": "connectionName",
        "opensearch.resource": "aosIndex",
        "opensearch.query": queryString,
        "pushdown": "true",
    }
```

```
)
```

Untuk informasi selengkapnya tentang cara membuat kueri di luar sintaks spesifiknya, lihat sintaks string kueri dalam dokumentasi. OpenSearch

Saat membaca dari OpenSearch koleksi yang berisi data tipe array, Anda harus menentukan bidang mana yang merupakan tipe array dalam panggilan metode Anda menggunakan `opensearch.read.field.as.array.include` parameter.

Misalnya, saat membaca dokumen berikut, Anda akan menemukan bidang `genre` dan `actor` array:

```
{
    "_index": "movies",
    "_id": "2",
    "_version": 1,
    "_seq_no": 0,
    "_primary_term": 1,
    "found": true,
    "_source": {
        "director": "Frankenheimer, John",
        "genre": [
            "Drama",
            "Mystery",
            "Thriller",
            "Crime"
        ],
        "year": 1962,
        "actor": [
            "Lansbury, Angela",
            "Sinatra, Frank",
            "Leigh, Janet",
            "Harvey, Laurence",
            "Silva, Henry",
            "Frees, Paul",
            "Gregory, James",
            "Bissell, Whit",
            "McGiver, John",
            "Parrish, Leslie",
            "Edwards, James",
            "Flowers, Bess",
            "Dhiegh, Khigh",
            "Payne, Julie",
            "Kleeb, Helen",
```

```
            "Gray, Joe",
            "Nalder, Reggie",
            "Stevens, Bert",
            "Masters, Michael",
            "Lowell, Tom"
        ],
        "title": "The Manchurian Candidate"
    }
}
```

Dalam hal ini, Anda akan menyertakan nama-nama bidang tersebut dalam panggilan metode Anda. Sebagai contoh:

```
"opensearch.read.field.as.array.include": "genre,actor"
```

Jika bidang array Anda bersarang di dalam struktur dokumen Anda, lihat itu menggunakan notasi titik:. `"genre,actor,foo.bar.baz"` Ini akan menentukan array yang `baz` disertakan dalam dokumen sumber Anda melalui dokumen tertanam `foo` yang berisi dokumen tertanam`bar`.

Menulis ke tabel OpenSearch Layanan

Contoh ini menulis informasi dari *`DynamicFrame`* ke OpenSearch Service yang sudah ada DynamicFrame. Jika indeks sudah memiliki informasi, AWS Glue akan menambahkan data dari Anda DynamicFrame. Anda harus memberikan `pushdown` parameternya.

Prasyarat:

- Meja OpenSearch layanan yang ingin Anda tulis. Anda akan memerlukan informasi identifikasi untuk tabel. Mari kita sebut ini *`TableName`*.
- Koneksi AWS Glue OpenSearch Service yang dikonfigurasi untuk memberikan informasi autentikasi dan lokasi jaringan. Untuk mendapatkan ini, selesaikan langkah-langkah dalam prosedur sebelumnya, Untuk mengkonfigurasi koneksi ke OpenSearch Layanan. Anda akan memerlukan nama koneksi AWS Glue, *`ConnectionName`*.

Sebagai contoh:

```
glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="opensearch",
    connection_options={
```

```
      "connectionName": "connectionName",
      "opensearch.resource": "aosIndex",
    },
)
```

OpenSearch Referensi opsi koneksi layanan

- `connectionName` — Diperlukan. Digunakan untuk Baca/Tulis. Nama koneksi AWS Glue OpenSearch Service yang dikonfigurasi untuk memberikan informasi autentikasi dan lokasi jaringan ke metode koneksi Anda.
- `opensearch.resource` — Diperlukan. Digunakan untuk Baca/Tulis. Nilai Valid: nama OpenSearch indeks. Nama indeks metode koneksi Anda akan berinteraksi dengan.
- `opensearch.query`— Digunakan untuk Baca. Nilai Valid: String lolos dari JSON atau, ketika string ini dimulai dengan?, bagian pencarian URL. OpenSearch Kueri yang memfilter apa yang harus diambil saat membaca. Untuk informasi lebih lanjut tentang penggunaan parameter ini, lihat bagian sebelumnyathe section called “Baca dari OpenSearch Layanan”.
- `pushdown`— Diperlukan jika. Digunakan untuk Baca. Nilai Valid: boolean. Menginstruksikan Spark untuk meneruskan kueri baca OpenSearch sehingga database hanya mengembalikan dokumen yang relevan.
- `opensearch.read.field.as.array.include`— Diperlukan jika membaca data tipe array. Digunakan untuk Baca. Nilai Valid: daftar nama bidang yang dipisahkan koma. Menentukan bidang untuk membaca sebagai array dari OpenSearch dokumen. Untuk informasi lebih lanjut tentang penggunaan parameter ini, lihat bagian sebelumnyathe section called “Baca dari OpenSearch Layanan”.

Koneksi Redshift

Anda dapat menggunakan AWS Glue for Spark untuk membaca dan menulis ke tabel di database Amazon Redshift. Saat menghubungkan ke database Amazon Redshift, AWS Glue memindahkan data melalui Amazon S3 untuk mencapai throughput maksimum, menggunakan Amazon Redshift SQL dan perintah. `COPY` `UNLOAD` Di AWS Glue 4.0 dan yang lebih baru, Anda dapat menggunakan integrasi Amazon Redshift untuk Apache Spark untuk membaca dan menulis dengan pengoptimalan dan fitur khusus untuk Amazon Redshift di luar yang tersedia saat menghubungkan melalui versi sebelumnya.

Pelajari tentang bagaimana AWS Glue mempermudah pengguna Amazon Redshift untuk bermigrasi ke AWS Glue untuk integrasi data tanpa server dan ETL.

Mengkonfigurasi koneksi Redshift

Untuk menggunakan cluster Amazon Redshift di AWS Glue, Anda memerlukan beberapa prasyarat:

- Direktori Amazon S3 untuk digunakan untuk penyimpanan sementara saat membaca dari dan menulis ke database.
- VPC Amazon yang memungkinkan komunikasi antara kluster Amazon Redshift, pekerjaan Glue, dan direktori Amazon AWS S3 Anda.
- Izin IAM yang sesuai pada tugas AWS Glue dan cluster Amazon Redshift.

Mengkonfigurasi peran IAM

Siapkan peran untuk cluster Amazon Redshift

Cluster Amazon Redshift Anda harus dapat membaca dan menulis ke Amazon S3 agar dapat berintegrasi dengan AWS pekerjaan Glue. Untuk mengizinkannya, Anda dapat mengaitkan peran IAM dengan cluster Amazon Redshift yang ingin Anda sambungkan. Peran Anda harus memiliki kebijakan yang mengizinkan baca dari dan tulis ke direktori sementara Amazon S3 Anda. Peran Anda harus memiliki hubungan kepercayaan yang memungkinkan `redshift.amazonaws.com` layanan untuk`AssumeRole`.

Untuk mengaitkan peran IAM dengan Amazon Redshift

1. Prasyarat: Bucket atau direktori Amazon S3 yang digunakan untuk penyimpanan sementara file.
2. Identifikasi izin Amazon S3 mana yang dibutuhkan cluster Amazon Redshift Anda. Saat memindahkan data ke dan dari klaster Amazon Redshift, AWS Glue jobs mengeluarkan pernyataan COPY dan UNLOAD terhadap Amazon Redshift. Jika pekerjaan Anda mengubah tabel di Amazon Redshift, AWS Glue juga akan mengeluarkan pernyataan CREATE LIBRARY. Untuk informasi tentang izin Amazon S3 tertentu yang diperlukan Amazon Redshift untuk menjalankan pernyataan ini, lihat dokumentasi Amazon Redshift: Amazon Redshift: Izin untuk mengakses Sumber Daya lainnya. AWS
3. Di konsol IAM, buat kebijakan IAM dengan izin yang diperlukan. Untuk informasi selengkapnya tentang membuat kebijakan Membuat kebijakan IAM.
4. Di konsol IAM, buat hubungan peran dan kepercayaan yang memungkinkan Amazon Redshift untuk mengambil peran tersebut. Ikuti petunjuk dalam dokumentasi IAM Untuk membuat peran untuk AWS layanan (konsol)

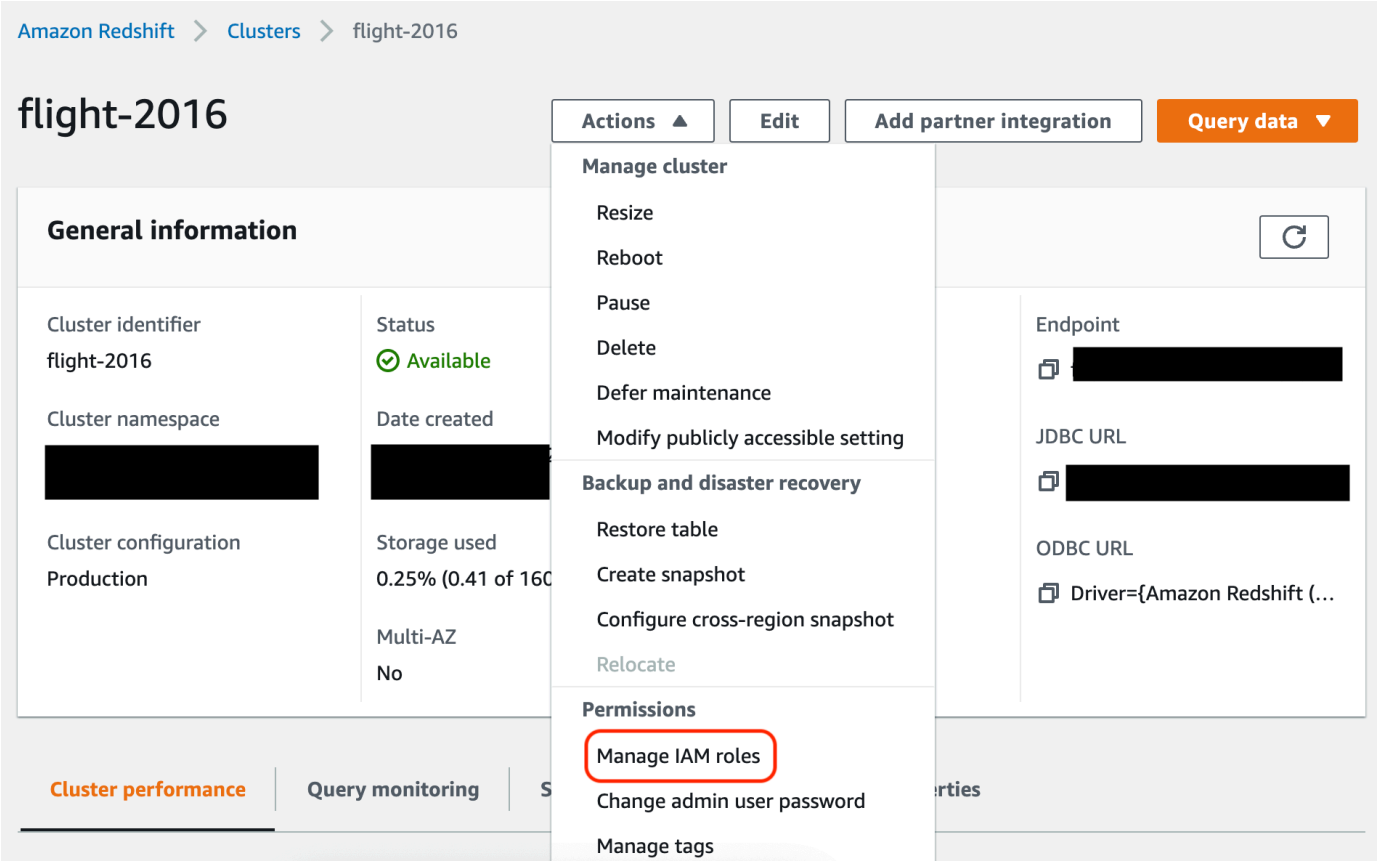
    - Ketika diminta untuk memilih kasus penggunaan AWS layanan, pilih “Redshift - Customizable”.

- Saat diminta untuk melampirkan kebijakan, pilih kebijakan yang telah Anda tetapkan sebelumnya.

> **Note**
>
> Untuk informasi selengkapnya tentang mengonfigurasi peran untuk Amazon Redshift, lihat Mengotorisasi Amazon Redshift untuk AWS mengakses layanan lain atas nama Anda dalam dokumentasi Amazon Redshift.

5. Di konsol Amazon Redshift, kaitkan peran tersebut dengan cluster Amazon Redshift Anda. Ikuti petunjuk dalam dokumentasi Amazon Redshift.

   Pilih opsi yang disorot di konsol Amazon Redshift untuk mengonfigurasi pengaturan ini:



> **Note**
>
> Secara default, pekerjaan AWS Glue meneruskan kredenal sementara Amazon Redshift yang dibuat menggunakan peran yang Anda tentukan untuk menjalankan pekerjaan. Kami

tidak menyarankan menggunakan kredensyal ini. Untuk tujuan keamanan, kredensyal ini kedaluwarsa setelah 1 jam.

Siapkan peran untuk pekerjaan AWS Glue

Pekerjaan AWS Glue membutuhkan peran untuk mengakses bucket Amazon S3. Anda tidak memerlukan izin IAM untuk klaster Amazon Redshift, akses Anda dikendalikan oleh konektivitas di Amazon VPC dan kredenal database Anda.

Siapkan Amazon VPC

Untuk menyiapkan akses untuk penyimpanan data Amazon Redshift

1. Masuk ke AWS Management Console dan buka konsol Amazon Redshift di https://console.aws.amazon.com/redshiftv2/.
2. Di panel navigasi sebelah kiri, pilih Klaster.
3. Pilih nama klaster yang ingin Anda akses dari AWS Glue.
4. Di bagian Properti Klaster, pilih grup keamanan di Grup keamanan VPC untuk mengizinkan AWS Glue untuk digunakan. Catat nama grup keamanan yang Anda pilih untuk referensi di masa mendatang. Memilih grup keamanan membuka daftar Grup Keamanan konsol Amazon EC2.
5. Memilih grup keamanan untuk memodifikasi dan menavigasi ke tab Inbound.
6. Tambahkan aturan self-referencing untuk mengizinkan komponen AWS Glue untuk berkomunikasi. Secara khusus, tambahkan atau konfirmasi bahwa ada aturan Jenis `All TCP`, Protokol adalah `TCP`, Rentang Port mencakup semua port, dan yang Sumber adalah nama grup keamanan yang sama seperti ID Grup.

   Aturan inbound terlihat serupa dengan yang berikut ini:

Jenis	Protokol	Rentang port	Sumber
Semua TCP	TCP	0–65535	database-security-group

   Misalnya:

7. Menambahkan aturan untuk lalu lintas outbound juga. Baik dengan membuka lalu lintas outbound ke semua port, misalnya:

Jenis	Protokol	Rentang Port	Tujuan
Semua Lalu Lintas	SEMUA	SEMUA	0.0.0.0/0

Atau membuat aturan self-referencing di mana Jenis `All TCP`, Protokol adalah `TCP`, Rentang Port mencakup semua port, dan yang Tujuan adalah nama grup keamanan yang sama seperti ID Grup. Jika menggunakan VPC endpoint Amazon S3, maka tambahkan juga aturan HTTPS untuk akses Amazon S3. *`S3- prefix-list-id`* diperlukan dalam aturan grup keamanan untuk mengizinkan lalu lintas dari VPC ke titik akhir VPC Amazon S3.

Misalnya:

Jenis	Protokol	Rentang Port	Tujuan
Semua TCP	TCP	0–65535	*`kelompok keamanan`*
HTTPS	TCP	443	*`s3- prefix-list-id`*

Mengatur AWS Glue

Anda perlu membuat koneksi AWS Glue Data Catalog yang menyediakan informasi koneksi Amazon VPC.

Untuk mengonfigurasi konektivitas Amazon Redshift Amazon VPC ke Glue AWS di konsol

1. Buat koneksi Katalog Data dengan mengikuti langkah-langkah di:the section called "Menambahkan AWS Glue koneksi". Setelah membuat koneksi, simpan nama koneksi, *`connectionName`*, untuk langkah selanjutnya.

   - Saat memilih jenis Koneksi, pilih Amazon Redshift.
   - Saat memilih klaster Redshift, pilih klaster Anda berdasarkan nama.
   - Berikan informasi koneksi default untuk pengguna Amazon Redshift di klaster Anda.

- Pengaturan VPC Amazon Anda akan dikonfigurasi secara otomatis.

> **Note**
>
> Anda harus menyediakan VPC Amazon Anda `PhysicalConnectionRequirements` secara manual saat membuat koneksi Amazon Redshift melalui SDK. AWS

2. Dalam konfigurasi pekerjaan AWS Glue Anda, berikan *`ConnectionName`* sebagai koneksi jaringan Tambahan.

Contoh: Membaca dari tabel Amazon Redshift

Anda dapat membaca dari cluster Amazon Redshift dan lingkungan tanpa server Amazon Redshift.

Prasyarat: Tabel Amazon Redshift yang ingin Anda baca. Ikuti langkah-langkah di bagian sebelumnya the section called “Konfigurasikan Redshift” setelah itu Anda harus memiliki URI Amazon S3 untuk direktori sementara, *`temp-s3-dir`* dan peran IAM,, (dalam akun). *`rs-role-name`* *`role-account-id`*

Using the Data Catalog

Prasyarat Tambahan: Database Katalog Data dan Tabel untuk tabel Amazon Redshift yang ingin Anda baca. Untuk informasi selengkapnya tentang Katalog Data, lihatPenemuan dan katalogisasi data. Setelah membuat entri untuk tabel Amazon Redshift Anda, Anda akan mengidentifikasi koneksi Anda dengan *`redshift-dc-database-name`*dan. *`redshift-table-name`*

Konfigurasi: Dalam opsi fungsi Anda, Anda akan mengidentifikasi Tabel Katalog Data Anda dengan `table_name` parameter `database` dan. Anda akan mengidentifikasi direktori sementara Amazon S3 Anda dengan. `redshift_tmp_dir` Anda juga akan memberikan *`rs-role-name`*menggunakan `aws_iam_role` kunci dalam `additional_options` parameter.

```
 glueContext.create_dynamic_frame.from_catalog(
    database = "redshift-dc-database-name",
    table_name = "redshift-table-name",
    redshift_tmp_dir = args["temp-s3-dir"],
    additional_options = {"aws_iam_role": "arn:aws:iam::role-account-id:role/rs-
role-name"})
```

Connecting directly

Prasyarat Tambahan: Anda akan memerlukan nama tabel Amazon Redshift Anda (. *redshift-table-name* Anda akan memerlukan informasi koneksi JDBC untuk cluster Amazon Redshift yang menyimpan tabel itu. Anda akan memberikan informasi koneksi Anda dengan *host*, *port*, *redshift-database-name*, *nama pengguna* dan *kata sandi*.

Anda dapat mengambil informasi koneksi dari konsol Amazon Redshift saat bekerja dengan cluster Amazon Redshift. Saat menggunakan Amazon Redshift tanpa server, lihat Menghubungkan ke Amazon Redshift Tanpa Server di dokumentasi Amazon Redshift.

Konfigurasi: Dalam opsi fungsi Anda, Anda akan mengidentifikasi parameter koneksi Anda dengan`url`,`dbtable`, `user` dan`password`. Anda akan mengidentifikasi direktori sementara Amazon S3 Anda dengan. `redshift_tmp_dir` Anda dapat menentukan peran IAM Anda menggunakan `aws_iam_role` saat Anda menggunakan`from_options`. Sintaksnya mirip dengan menghubungkan melalui Katalog Data, tetapi Anda meletakkan parameter di `connection_options` peta.

Ini adalah praktik yang buruk untuk membuat hardcode kata sandi ke dalam skrip AWS Glue. Pertimbangkan untuk menyimpan kata sandi Anda AWS Secrets Manager dan mengambilnya di skrip Anda dengan SDK for Python (Boto3).

```
my_conn_options = {
    "url": "jdbc:redshift://host:port/redshift-database-name",
    "dbtable": "redshift-table-name",
    "user": "username",
    "password": "password",
    "redshiftTmpDir": args["temp-s3-dir"],
    "aws_iam_role": "arn:aws:iam::account id:role/rs-role-name"
}

df = glueContext.create_dynamic_frame.from_options("redshift", my_conn_options)
```

Contoh: Menulis ke tabel Amazon Redshift

Anda dapat menulis ke cluster Amazon Redshift dan lingkungan tanpa server Amazon Redshift.

Prasyarat: Cluster Amazon Redshift dan ikuti langkah-langkah di bagian sebelumnya the section called “Konfigurasikan Redshift” setelah itu Anda harus memiliki URI Amazon S3 untuk direktori

sementara, *temp-s3-dir* dan peran IAM,, (dalam akun). *rs-role-namerole-account-id* Anda juga akan membutuhkan konten `DynamicFrame` yang ingin Anda tulis ke database.

Using the Data Catalog

Prasyarat Tambahan Database Katalog Data untuk klaster Amazon Redshift dan tabel yang ingin Anda tulis. Untuk informasi selengkapnya tentang Katalog Data, lihatPenemuan dan katalogisasi data. Anda akan mengidentifikasi koneksi Anda dengan *redshift-dc-database-name*dan tabel target dengan *redshift-table-name*.

Konfigurasi: Dalam opsi fungsi Anda, Anda akan mengidentifikasi Database Katalog Data Anda dengan `database` parameter, lalu berikan tabel dengan`table_name`. Anda akan mengidentifikasi direktori sementara Amazon S3 Anda dengan. `redshift_tmp_dir` Anda juga akan memberikan *rs-role-name*menggunakan `aws_iam_role` kunci dalam `additional_options` parameter.

```
 glueContext.write_dynamic_frame.from_catalog(
    frame = input dynamic frame,
    database = "redshift-dc-database-name",
    table_name = "redshift-table-name",
    redshift_tmp_dir = args["temp-s3-dir"],
    additional_options = {"aws_iam_role": "arn:aws:iam::account-id:role/rs-role-
name"})
```

Connecting through a AWS Glue connection

Anda dapat terhubung ke Amazon Redshift secara langsung menggunakan metode ini`write_dynamic_frame.from_options`. Namun, daripada memasukkan detail koneksi Anda langsung ke skrip Anda, Anda dapat mereferensikan detail koneksi yang disimpan dalam koneksi Katalog Data dengan `from_jdbc_conf` metode ini. Anda dapat melakukan ini tanpa merayapi atau membuat tabel Katalog Data untuk database Anda. Untuk informasi selengkapnya tentang koneksi Katalog Data, lihatMenghubungkan ke data.

Prasyarat Tambahan: Koneksi Katalog Data untuk database Anda, tabel Amazon Redshift yang ingin Anda baca

Konfigurasi: Anda akan mengidentifikasi koneksi Katalog Data Anda dengan *dc-connection-name*. Anda akan mengidentifikasi database dan tabel Amazon Redshift Anda dengan *redshift-table-name*dan. *redshift-database-name* Anda akan memberikan informasi

koneksi Katalog Data Anda dengan `catalog_connection` dan informasi Amazon Redshift Anda dengan `dbtable` dan. `database` Sintaksnya mirip dengan menghubungkan melalui Katalog Data, tetapi Anda meletakkan parameter di `connection_options` peta.

```
my_conn_options = {
    "dbtable": "redshift-table-name",
    "database": "redshift-database-name",
    "aws_iam_role": "arn:aws:iam::role-account-id:role/rs-role-name"
}

glueContext.write_dynamic_frame.from_jdbc_conf(
    frame = input dynamic frame,
    catalog_connection = "dc-connection-name",
    connection_options = my_conn_options,
    redshift_tmp_dir = args["temp-s3-dir"])
```

Referensi opsi koneksi Amazon Redshift

Opsi koneksi dasar yang digunakan untuk semua koneksi AWS Glue JDBC untuk mengatur informasi sepertiurl, `user` dan `password` konsisten di semua jenis JDBC. Untuk informasi selengkapnya tentang parameter JDBC standar, lihat. the section called “Parameter koneksi JDBC”

Jenis koneksi Amazon Redshift membutuhkan beberapa opsi koneksi tambahan:

- `"redshiftTmpDir"`: (Wajib) Jalur Amazon S3 tempat data sementara dapat dipentaskan saat menyalin dari database.
- `"aws_iam_role"`: (Opsional) ARN untuk peran IAM. Pekerjaan AWS Glue akan meneruskan peran ini ke klaster Amazon Redshift untuk memberikan izin klaster yang diperlukan untuk menyelesaikan instruksi dari pekerjaan tersebut.

Opsi koneksi tambahan tersedia di AWS Glue 4.0+

Anda juga dapat meneruskan opsi untuk konektor Amazon Redshift baru melalui opsi koneksi AWS Glue. Untuk daftar lengkap opsi konektor yang didukung, lihat bagian parameter Spark SQL di integrasi Amazon Redshift untuk Apache Spark.

Untuk kenyamanan Anda, kami mengulangi opsi baru tertentu di sini:

Nama	Wajib	Default	Deskripsi
autopushdown	Tidak	BETUL	Menerapkan pushdown predikat dan kueri dengan menangkap dan menganalisis rencana logis Spark untuk operasi SQL. Operasi diterjemahkan ke dalam kueri SQL, dan kemudian dijalankan di Amazon Redshift untuk meningkatkan kinerja.
autopushdown.s3_result_cache	Tidak	SALAH	Cache kueri SQL untuk membongkar data untuk pemetaan jalur Amazon S3 di memori sehingga kueri yang sama tidak perlu dijalankan lagi dalam sesi Spark yang sama. Hanya didukung saat `autopushdown` diaktifkan.
unload_s3_format	Tidak	PARQUET	PARQUET - Membongkar hasil kueri dalam format Parket. TEKS - Membongkar hasil kueri dalam

Nama	Wajib	Default	Deskripsi
			format teks yang dibatasi pipa.
sse_kms_key	Tidak	N/A	Kunci AWS SSE-KMS untuk digunakan untuk enkripsi selama `UNLOAD` operasi alih-alih enkripsi default untuk. AWS
ekstrakoopiopsi	Tidak	N/A	Daftar opsi tambahan untuk ditambahkan ke perintah Amazon `COPY` Redshift saat memuat data, `TRUNCATECOLUMNS` seperti `MAXERROR` n atau (untuk opsi lain lihat COPY: Parameter opsional). Perhatikan bahwa karena opsi ini ditambahkan ke akhir `COPY` perintah, hanya opsi yang masuk akal di akhir perintah yang dapat digunakan. Itu harus mencakup sebagian besar kasus penggunaan yang mungkin.

Nama	Wajib	Default	Deskripsi
csvnullstring (eksperimental)	Tidak	NULL	Nilai String untuk menulis untuk nulls saat menggunakan CSV. `tempformat` Ini harus menjadi nilai yang tidak muncul dalam data aktual Anda.

Parameter baru ini dapat digunakan dengan cara-cara berikut.

Opsi baru untuk peningkatan kinerja

Konektor baru memperkenalkan beberapa opsi peningkatan kinerja baru:

- `autopushdown`: Diaktifkan secara default.
- `autopushdown.s3_result_cache`: Dinonaktifkan secara default.
- `unload_s3_format`: secara `PARQUET` default.

Untuk informasi tentang penggunaan opsi ini, lihat Integrasi Amazon Redshift untuk Apache Spark. Kami menyarankan agar Anda tidak mengaktifkan `autopushdown.s3_result_cache` ketika Anda memiliki operasi baca dan tulis campuran karena hasil cache mungkin berisi informasi basi. Opsi `unload_s3_format` ini diatur ke secara `PARQUET` default untuk `UNLOAD` perintah, untuk meningkatkan kinerja dan mengurangi biaya penyimpanan. Untuk menggunakan perilaku default `UNLOAD` perintah, setel ulang opsi ke`TEXT`.

Opsi enkripsi baru untuk membaca

Secara default, data dalam folder sementara yang AWS Glue digunakan saat membaca data dari tabel Amazon Redshift dienkripsi menggunakan enkripsi. `SSE-S3` Untuk menggunakan kunci terkelola pelanggan from AWS Key Management Service (AWS KMS) untuk mengenkripsi data Anda, Anda dapat mengatur dari `("sse_kms_key" # kmsKey)` mana KSMKey adalah ID kunci AWS KMS, bukan opsi `("extraunloadoptions" # s"ENCRYPTED KMS_KEY_ID '$kmsKey'")` pengaturan lama di versi 3.0. AWS Glue

```
datasource0 = glueContext.create_dynamic_frame.from_catalog(
  database = "database-name",
  table_name = "table-name",
  redshift_tmp_dir = args["TempDir"],
  additional_options = {"sse_kms_key":"<KMS_KEY_ID>"},
  transformation_ctx = "datasource0"
)
```

Mendukung URL JDBC berbasis IAM

Konektor baru mendukung URL JDBC berbasis IAM sehingga Anda tidak perlu memasukkan pengguna/kata sandi atau rahasia. Dengan URL JDBC berbasis IAM, konektor menggunakan peran runtime pekerjaan untuk mengakses sumber data Amazon Redshift.

Langkah 1: Lampirkan kebijakan minimal yang diperlukan berikut ke peran runtime AWS Glue pekerjaan Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "redshift:GetClusterCredentials",
            "Resource": [
                "arn:aws:redshift:<region>:<account>:dbgroup:<cluster name>/*",
                "arn:aws:redshift:*:<account>:dbuser:*/*",
                "arn:aws:redshift:<region>:<account>:dbname:<cluster name>/<database
 name>"
            ]
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": "redshift:DescribeClusters",
            "Resource": "*"
        }
    ]
}
```

Langkah 2: Gunakan URL JDBC berbasis IAM sebagai berikut. Tentukan opsi baru `DbUser` dengan nama pengguna Amazon Redshift yang terhubung dengan Anda.

```
conn_options = {
     // IAM-based JDBC URL
    "url": "jdbc:redshift:iam://<cluster name>:<region>/<database name>",
    "dbtable": dbtable,
    "redshiftTmpDir": redshiftTmpDir,
    "aws_iam_role": aws_iam_role,
    "DbUser": "<Redshift User name>" // required for IAM-based JDBC URL
    }

redshift_write = glueContext.write_dynamic_frame.from_options(
    frame=dyf,
    connection_type="redshift",
    connection_options=conn_options
)

redshift_read = glueContext.create_dynamic_frame.from_options(
    connection_type="redshift",
    connection_options=conn_options
)
```

> **Note**
>
> A `DynamicFrame` saat ini hanya mendukung URL JDBC berbasis IAM dengan a dalam `DbUser` alur kerja. `GlueContext.create_dynamic_frame.from_options`

Migrasi dari AWS Glue versi 3.0 ke versi 4.0

Di AWS Glue 4.0, pekerjaan ETL memiliki akses ke konektor Amazon Redshift Spark baru dan driver JDBC baru dengan opsi dan konfigurasi yang berbeda. Konektor dan driver Amazon Redshift baru ditulis dengan mempertimbangkan kinerja, dan menjaga konsistensi transaksional data Anda. Produk-produk ini didokumentasikan dalam dokumentasi Amazon Redshift. Untuk informasi selengkapnya, lihat:

- Integrasi Amazon Redshift untuk Apache Spark
- Driver Amazon Redshift JDBC, versi 2.1

Pembatasan nama tabel/kolom dan pengidentifikasi

Konektor dan driver Amazon Redshift Spark yang baru memiliki persyaratan yang lebih terbatas untuk nama tabel Redshift. Untuk informasi selengkapnya, lihat Nama dan pengidentifikasi untuk menentukan nama tabel Amazon Redshift Anda. Alur kerja bookmark pekerjaan mungkin tidak berfungsi dengan nama tabel yang tidak cocok dengan aturan dan karakter tertentu, seperti spasi.

Jika Anda memiliki tabel lama dengan nama yang tidak sesuai dengan aturan Nama dan pengenal dan melihat masalah dengan bookmark (pekerjaan memproses ulang data tabel Amazon Redshift lama), sebaiknya ganti nama nama tabel Anda. Untuk informasi selengkapnya, lihat contoh ALTER TABLE.

Perubahan tempformat default di Dataframe

Konektor Spark AWS Glue versi 3.0 default `tempformat` ke CSV saat menulis ke Amazon Redshift. Agar konsisten, di AWS Glue versi 3.0, `DynamicFrame` masih default untuk digunakan. `tempformat` `CSV` Jika sebelumnya Anda pernah menggunakan Spark Dataframe API secara langsung dengan konektor Amazon Redshift Spark, Anda dapat secara eksplisit menyetel ke CSV di opsi/. `tempformat` `DataframeReader` `Writer` Jika tidak, `tempformat` default ke konektor `AVRO` Spark baru.

Perubahan perilaku: petakan tipe data Amazon Redshift REAL ke Spark tipe data FLOAT, bukan DOUBLE

Di AWS Glue versi 3.0, Amazon Redshift `REAL` dikonversi ke tipe Spark `DOUBLE`. Konektor Amazon Redshift Spark yang baru telah memperbarui perilaku sehingga jenis Amazon Redshift dikonversi ke, dan kembali dari, `REAL` tipe Spark. `FLOAT` Jika Anda memiliki kasus penggunaan lama di mana Anda masih ingin jenis Amazon `REAL` Redshift dipetakan ke tipe `DOUBLE` Spark, Anda dapat menggunakan solusi berikut:

- Untuk a`DynamicFrame`, petakan `Float` tipe ke `Double` tipe dengan`DynamicFrame.ApplyMapping`. Untuk a`Dataframe`, Anda perlu menggunakan`cast`.

Contoh kode:

```
dyf_cast = dyf.apply_mapping([('a', 'long', 'a', 'long'), ('b', 'float', 'b',
 'double')])
```

Koneksi Kafka

Mengkhususkan koneksi ke klaster Kafka atau klaster Amazon Managed Streaming for Apache Kafka.

Anda dapat membaca dan menulis ke aliran data Kafka menggunakan informasi yang disimpan dalam tabel Katalog Data, atau dengan memberikan informasi untuk langsung mengakses aliran data. Anda dapat membaca informasi dari Kafka menjadi Spark DataFrame, lalu mengubahnya menjadi Glue AWS . DynamicFrame Anda dapat menulis DynamicFrames ke Kafka dalam format JSON. Jika Anda langsung mengakses aliran data, gunakan opsi ini untuk memberikan informasi tentang cara mengakses aliran data.

Jika Anda menggunakan `getCatalogSource` atau `create_data_frame_from_catalog` menggunakan catatan dari sumber streaming Kafka, `getCatalogSink` atau `write_dynamic_frame_from_catalog` untuk menulis catatan ke Kafka, dan pekerjaan tersebut memiliki database Katalog Data dan informasi nama tabel, dan dapat menggunakannya untuk mendapatkan beberapa parameter dasar untuk membaca dari sumber streaming Kafka. Jika Anda menggunakan`getSource`,`getCatalogSink`,`getSourceWithFormat`,`getSinkWithFormat`, `createDataFrameFromOptions` atau`create_data_frame_from_options`, atau`write_dynamic_frame_from_catalog`, Anda harus menentukan parameter dasar ini menggunakan opsi koneksi yang dijelaskan di sini.

Anda dapat menentukan opsi koneksi untuk Kafka menggunakan argumen berikut untuk metode yang ditentukan di `GlueContext` kelas.

- Skala
  - `connectionOptions`: Gunakan dengan`getSource`,`createDataFrameFromOptions`, `getSink`
  - `additionalOptions`: Gunakan dengan`getCatalogSource`, `getCatalogSink`
  - `options`: Gunakan dengan`getSourceWithFormat`, `getSinkWithFormat`
- Python
  - `connection_options`: Gunakan dengan`create_data_frame_from_options`, `write_dynamic_frame_from_options`
  - `additional_options`: Gunakan dengan`create_data_frame_from_catalog`, `write_dynamic_frame_from_catalog`
  - `options`: Gunakan dengan`getSource`, `getSink`

Untuk catatan dan batasan tentang streaming pekerjaan ETL, konsultasikanthe section called “Streaming catatan dan batasan ETL”.

Konfigurasikan Kafka

Tidak ada AWS prasyarat untuk menghubungkan ke aliran Kafka yang tersedia melalui internet.

Anda dapat membuat koneksi AWS Glue Kafka untuk mengelola kredenal koneksi Anda. Untuk informasi selengkapnya, lihat the section called “Membuat koneksi untuk aliran data Kafka”. Dalam konfigurasi pekerjaan AWS Glue Anda, berikan *connectionName* sebagai koneksi jaringan Tambahan, lalu, dalam panggilan metode Anda, berikan *connectionName* ke parameter. `connectionName`

Dalam kasus tertentu, Anda perlu mengkonfigurasi prasyarat tambahan:

- Jika menggunakan Amazon Managed Streaming for Apache Kafka dengan autentikasi IAM, Anda memerlukan konfigurasi IAM yang sesuai.
- Jika menggunakan Amazon Managed Streaming for Apache Kafka dalam Amazon VPC, Anda memerlukan konfigurasi Amazon VPC yang sesuai. Anda perlu membuat koneksi AWS Glue yang menyediakan informasi koneksi Amazon VPC. Anda akan memerlukan konfigurasi pekerjaan Anda untuk menyertakan koneksi AWS Glue sebagai koneksi jaringan Tambahan.

Untuk informasi lebih lanjut tentang prasyarat pekerjaan Streaming ETL, lihat. the section called “Lowongan kerja Streaming ETL”

Contoh: Membaca dari aliran Kafka

Digunakan bersama dengantthe section called “forEachBatch”.

Contoh untuk sumber streaming Kafka:

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "startingOffsets": "earliest",
      "inferSchema": "true",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.create_data_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

Contoh: Menulis ke aliran Kafka

Contoh untuk menulis ke Kafka:

Contoh dengan `getSink` metode:

```
data_frame_datasource0 =
glueContext.getSink(
 connectionType="kafka",
 connectionOptions={
  JsonOptions("""{
   "connectionName": "ConfluentKafka",
   "classification": "json",
   "topic": "kafka-auth-topic",
   "typeOfData": "kafka"}
  """)},
 transformationContext="dataframe_ApacheKafka_node1711729173428")
 .getDataFrame()
```

Contoh dengan `write_dynamic_frame.from_options` metode:

```
kafka_options =
    { "connectionName": "ConfluentKafka",
      "topicName": "kafka-auth-topic",
      "classification": "json"
    }
data_frame_datasource0 =
 glueContext.write_dynamic_frame.from_options(connection_type="kafka",
 connection_options=kafka_options)
```

Referensi opsi koneksi Kafka

Saat membaca, gunakan opsi koneksi berikut dengan`"connectionType": "kafka"`:

- `"bootstrap.servers"` (Wajib) Daftar URL server bootstrap, misalnya, sebagai `b-1.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094`. Opsi ini harus ditentukan dalam panggilan API atau didefinisikan dalam metadata tabel dalam Katalog Data.
- `"security.protocol"`(Wajib) Protokol yang digunakan untuk berkomunikasi dengan broker. Nilai yang mungkin adalah `"SSL"` atau `"PLAINTEXT"`.
- `"topicName"`(Wajib) Daftar topik yang dipisahkan koma untuk berlangganan. Anda harus menentukan satu dan hanya satu dari`"topicName"`, `"assign"` atau`"subscribePattern"`.
- `"assign"`: (Diperlukan) String JSON yang menentukan spesifik `TopicPartitions` untuk dikonsumsi. Anda harus menentukan satu dan hanya satu dari`"topicName"`, `"assign"` atau`"subscribePattern"`.

Contoh: '{"topicA”: [0,1], "topiCB”: [2,4]}'

- `"subscribePattern"`: (Wajib) Sebuah string regex Java yang mengidentifikasi daftar topik untuk berlangganan. Anda harus menentukan satu dan hanya satu dari`"topicName"`, `"assign"` atau`"subscribePattern"`.

  Contoh: 'topik. *'

- `"classification"`(Wajib) Format file yang digunakan oleh data dalam catatan. Diperlukan kecuali disediakan melalui Katalog Data.
- `"delimiter"`(Opsional) Pemisah nilai yang digunakan saat `classification` CSV. Defaultnya adalah `",`.”
- `"startingOffsets"`: (Opsional) Posisi awal dalam topik Kafka tempat untuk membaca data. Nilai yang mungkin adalah `"earliest"` atau `"latest"`. Nilai default-nya adalah `"latest"`.
- `"startingTimestamp"`: (Opsional, hanya didukung untuk AWS Glue versi 4.0 atau yang lebih baru) Stempel waktu catatan dalam topik Kafka untuk membaca data dari. Nilai yang mungkin adalah string Timestamp dalam format UTC dalam pola `yyyy-mm-ddTHH:MM:SSZ` (di mana `Z` mewakili zona waktu UTC offset dengan +/-. Misalnya: “2023-04-04T 08:00:00-04:00 “).

  Catatan: Hanya satu dari 'startingOffsets' atau 'startingTimeStamp' yang dapat hadir dalam daftar Opsi Koneksi skrip streaming AWS Glue, termasuk kedua properti ini akan mengakibatkan kegagalan pekerjaan.

- `"endingOffsets"`: (Opsional) Titik akhir ketika kueri batch berakhir. Nilai yang mungkin adalah `"latest"` atau string JSON yang menentukan sebuah ending offset untuk setiap `TopicPartition`.

  Untuk string JSON, formatnya adalah `{"topicA":{"0":23,"1":-1},"topicB":{"0":-1}}`. Nilai `-1` sebagai offset mewakili `"latest"`.

- `"pollTimeoutMs"`: (Opsional) Waktu habis dalam milidetik untuk memeriksa status data dari Kafka di pelaksana tugas Spark. Nilai default-nya adalah `512`.
- `"numRetries"`: (Opsional) Jumlah percobaan sebelum gagal untuk mengambil offset Kafka. Nilai default-nya adalah `3`.
- `"retryIntervalMs"`: (Opsional) Waktu dalam milidetik yang digunakan untuk menunggu sebelum mencoba kembali untuk mengambil offset Kafka. Nilai default-nya adalah `10`.
- `"maxOffsetsPerTrigger"`: (Opsional) Batas tingkat pada jumlah maksimum offset yang diproses untuk setiap interval pemicu. Jumlah total offset yang ditentukan dibagi secara

proporsional di seluruh `topicPartitions` dengan volume yang berbeda. Nilai default-nya adalah nol, yang berarti bahwa konsumen membaca semua offset sampai diketahui offset terbaru.

- `"minPartitions"`: (Opsional) Jumlah minimum partisi yang diinginkan yang akan dibaca dari Kafka. Nilai default-nya adalah nol, yang berarti bahwa jumlah partisi spark sama dengan jumlah partisi Kafka.
- `"includeHeaders"`: (Opsional) Apakah akan menyertakan header Kafka. Ketika opsi diatur ke “true”, output data akan berisi kolom tambahan bernama “glue_streaming_kafka_headers” dengan tipe. `Array[Struct(key: String, value: String)]` Nilai default adalah “false”. Opsi ini tersedia dalam AWS Glue versi 3.0 atau yang lebih baru.
- `"schema"`: (Diperlukan saat InferSchema disetel ke false) Skema yang digunakan untuk memproses muatan. Jika klasifikasi adalah `avro` skema yang disediakan harus dalam format skema Avro. Jika klasifikasi tidak, skema `avro` yang disediakan harus dalam format skema DDL.

  Berikut ini adalah contoh skema.

  Example in DDL schema format

  ```
  'column1' INT, 'column2' STRING , 'column3' FLOAT
  ```

  Example in Avro schema format

  ```
  {
  "type":"array",
  "items":
  {
  "type":"record",
  "name":"test",
  "fields":
  [
    {
      "name":"_id",
      "type":"string"
    },
    {
      "name":"index",
      "type":
      [
        "int",
        "string",
        "float"
  ```

```
    ]
  }
]
}
}
```

- `"inferSchema"`: (Opsional) Nilai default adalah 'salah'. Jika disetel ke 'true', skema akan terdeteksi saat runtime dari payload di dalamnya. `foreachbatch`
- `"avroSchema"`: (Usang) Parameter digunakan untuk menentukan skema data Avro saat format Avro digunakan. Parameter ini sekarang tidak digunakan lagi. Gunakan parameter `schema`.
- `"addRecordTimestamp"`: (Opsional) Ketika opsi ini diatur ke 'true', output data akan berisi kolom tambahan bernama “__src_timestamp” yang menunjukkan waktu ketika catatan terkait diterima oleh topik. Nilai default adalah 'salah'. Opsi ini didukung di AWS Glue versi 4.0 atau yang lebih baru.
- `"emitConsumerLagMetrics"`: (Opsional) Ketika opsi disetel ke 'true', untuk setiap batch, itu akan memancarkan metrik untuk durasi antara catatan tertua yang diterima oleh topik dan waktu tiba. AWS Glue CloudWatch Nama metriknya adalah “glue.driver.streaming. maxConsumerLagInMs”. Nilai default adalah 'salah'. Opsi ini didukung di AWS Glue versi 4.0 atau yang lebih baru.

Saat menulis, gunakan opsi koneksi berikut dengan`"connectionType": "kafka"`:

- `"connectionName"`(Wajib) Nama sambungan AWS Glue yang digunakan untuk menghubungkan ke cluster Kafka (mirip dengan sumber Kafka).
- `"topic"`(Wajib) Jika kolom topik ada maka nilainya digunakan sebagai topik saat menulis baris yang diberikan ke Kafka, kecuali jika opsi konfigurasi topik disetel. Artinya, opsi `topic` konfigurasi mengesampingkan kolom topik.
- `"partition"`(Opsional) Jika nomor partisi yang valid ditentukan, itu `partition` akan digunakan saat mengirim catatan.

  Jika tidak ada partisi yang `key` ditentukan tetapi ada, partisi akan dipilih menggunakan hash kunci.

  Jika tidak `partition` ada `key` atau tidak ada, partisi akan dipilih berdasarkan partisi lengket perubahan tersebut ketika setidaknya batch.size byte diproduksi ke partisi.
- `"key"`(Opsional) Digunakan untuk partisi jika nol`partition`.
- `"classification"`(Opsional) Format file yang digunakan oleh data dalam catatan. Kami hanya mendukung JSON, CSV dan Avro.

Dengan format Avro, kami dapat menyediakan AvroSchema khusus untuk diserialisasikan, tetapi perhatikan bahwa ini perlu disediakan pada sumber untuk deserialisasi juga. Lain, secara default menggunakan Apache AvroSchema untuk serialisasi.

Selain itu, Anda dapat menyempurnakan wastafel Kafka sesuai kebutuhan dengan memperbarui parameter konfigurasi produsen Kafka. Perhatikan bahwa tidak ada daftar izin pada opsi koneksi, semua pasangan nilai kunci tetap ada di wastafel apa adanya.

Namun, ada daftar kecil opsi penolakan yang tidak akan berlaku. Untuk informasi selengkapnya, lihat konfigurasi khusus Kafka.

Koneksi Azure Cosmos DB

Anda dapat menggunakan AWS Glue for Spark untuk membaca dan menulis ke wadah yang ada di Azure Cosmos DB menggunakan NoSQL API di Glue 4.0 dan versi yang lebih baru. AWS Anda dapat menentukan apa yang harus dibaca dari Azure Cosmos DB dengan kueri SQL. Anda terhubung ke Azure Cosmos DB menggunakan Azure Cosmos DB Key yang disimpan melalui AWS Secrets Manager koneksi Glue. AWS

Untuk informasi selengkapnya tentang Azure Cosmos DB untuk NoSQL, lihat dokumentasi Azure.

Mengkonfigurasi koneksi Azure Cosmos DB

Untuk terhubung ke Azure Cosmos DB dari AWS Glue, Anda harus membuat dan menyimpan Kunci Azure Cosmos DB Anda secara AWS Secrets Manager rahasia, lalu mengaitkan rahasia itu dengan koneksi Glue Azure Cosmos DB. AWS

Prasyarat:

- Di Azure, Anda perlu mengidentifikasi atau membuat Azure Cosmos DB Key untuk digunakan oleh GlueAWS,. `cosmosKey` Untuk informasi selengkapnya, lihat Akses aman ke data di Azure Cosmos DB di dokumentasi Azure.

Untuk mengonfigurasi koneksi ke Azure Cosmos DB:

1. DiAWS Secrets Manager, buat rahasia menggunakan Azure Cosmos DB Key Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *`secretName`* untuk langkah selanjutnya.

- *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci spark.cosmos.accountKey dengan nilai CosmosKey.*

2. Di konsol AWS Glue, buat koneksi dengan mengikuti langkah-langkah dithe section called "Menambahkan AWS Glue koneksi". Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk penggunaan masa depan di GlueAWS.

   - Saat memilih jenis Koneksi, pilih Azure Cosmos DB.
   - Saat memilih AWSSecret, berikan *secretName*.

Setelah membuat koneksi AWS Glue Azure Cosmos DB, Anda harus melakukan langkah-langkah berikut sebelum menjalankan pekerjaan AWS Glue Anda:

- Berikan peran IAM yang terkait dengan izin pekerjaan AWS Glue Anda untuk membaca *secretName*.
- Dalam konfigurasi pekerjaan AWS Glue Anda, berikan *ConnectionName* sebagai koneksi jaringan Tambahan.

Membaca dari Azure Cosmos DB untuk wadah NoSQL

Prasyarat:

- Azure Cosmos DB untuk wadah NoSQL yang ingin Anda baca. Anda akan memerlukan informasi identifikasi untuk wadah.

  Sebuah Azure Cosmos untuk wadah NoSQL diidentifikasi oleh database dan wadahnya. Anda harus memberikan nama database, *CosmosDBName*, dan container *cosmosContainerName*, saat menghubungkan ke Azure Cosmos for NoSQL API.
- Koneksi AWS Glue Azure Cosmos DB yang dikonfigurasi untuk memberikan informasi autentikasi dan lokasi jaringan. Untuk mendapatkan ini, selesaikan langkah-langkah dalam prosedur sebelumnya, Untuk mengkonfigurasi koneksi ke Azure Cosmos DB. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.

Misalnya:

```
azurecosmos_read = glueContext.create_dynamic_frame.from_options(
    connection_type="azurecosmos",
```

```
    connection_options={
    "connectionName": connectionName,
    "spark.cosmos.database": cosmosDBName,
    "spark.cosmos.container": cosmosContainerName,
    }
)
```

Anda juga dapat memberikan kueri SELECT SQL, untuk memfilter hasil yang dikembalikan ke Anda DynamicFrame. Anda perlu mengkonfigurasi`query`.

Misalnya:

```
azurecosmos_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="azurecosmos",
    connection_options={
        "connectionName": "connectionName",
        "spark.cosmos.database": cosmosDBName,
        "spark.cosmos.container": cosmosContainerName,
        "spark.cosmos.read.customQuery": "query"
    }
)
```

Menulis ke Azure Cosmos DB untuk wadah NoSQL

Contoh ini menulis informasi dari *DynamicFrame* yang sudah ada DynamicFrame ke Azure Cosmos DB. Jika wadah sudah memiliki informasi, AWS Glue akan menambahkan data dari Anda DynamicFrame. Jika informasi dalam wadah memiliki skema yang berbeda dari informasi yang Anda tulis, Anda akan mengalami kesalahan.

Prasyarat:

- Tabel Azure Cosmos DB yang ingin Anda tulis. Anda akan memerlukan informasi identifikasi untuk wadah. Anda harus membuat wadah sebelum memanggil metode koneksi.

  Sebuah Azure Cosmos untuk wadah NoSQL diidentifikasi oleh database dan wadahnya. Anda harus memberikan nama database, *CosmosDBName*, dan container *cosmosContainerName*, saat menghubungkan ke Azure Cosmos for NoSQL API.
- Koneksi AWS Glue Azure Cosmos DB yang dikonfigurasi untuk memberikan informasi autentikasi dan lokasi jaringan. Untuk mendapatkan ini, selesaikan langkah-langkah dalam prosedur sebelumnya, Untuk mengkonfigurasi koneksi ke Azure Cosmos DB. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.

Misalnya:

```
azurecosmos_write = glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="azurecosmos",
    connection_options={
    "connectionName": connectionName,
    "spark.cosmos.database": cosmosDBName,
    "spark.cosmos.container": cosmosContainerName
)
```

Referensi opsi koneksi Azure Cosmos DB

- `connectionName` — Diperlukan. Digunakan untuk Baca/tulis Nama koneksi AWS Glue Azure Cosmos DB yang dikonfigurasi untuk memberikan informasi autentikasi dan lokasi jaringan ke metode koneksi Anda.
- `spark.cosmos.database` — Diperlukan. Digunakan untuk Baca/tulis Nilai yang Valid: nama database. Azure Cosmos DB untuk nama database NoSQL.
- `spark.cosmos.container` — Diperlukan. Digunakan untuk Baca/tulis Nilai Valid: nama wadah. Azure Cosmos DB untuk nama wadah NoSQL.
- `spark.cosmos.read.customQuery`— Digunakan untuk Baca. Nilai Valid: PILIH kueri SQL. Kueri khusus untuk memilih dokumen yang akan dibaca.

Koneksi Azure SQL

Anda dapat menggunakan AWS Glue for Spark untuk membaca dan menulis ke tabel di Azure SQL Managed Instances di AWS Glue 4.0 dan versi yang lebih baru. Anda dapat menentukan apa yang harus dibaca dari Azure SQL dengan query SQL. Anda terhubung ke Azure SQL menggunakan kredensyal pengguna dan kata sandi yang disimpan melalui AWS Secrets Manager koneksi Glue. AWS

Untuk informasi lebih lanjut tentang Azure SQL, lihat dokumentasi Azure SQL.

Mengkonfigurasi koneksi Azure SQL

Untuk terhubung ke Azure SQL dari AWS Glue, Anda harus membuat dan menyimpan kredensyal Azure SQL Anda secara AWS Secrets Manager rahasia, lalu mengaitkan rahasia itu dengan koneksi Azure SQL Glue. AWS

Untuk mengkonfigurasi koneksi ke Azure SQL:

1. DiAWS Secrets Manager, buat rahasia menggunakan kredensyal Azure SQL Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci user dengan nilai AzuresQLUsername.*
   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci password dengan nilai AzuresQLPassword.*

2. Di konsol AWS Glue, buat koneksi dengan mengikuti langkah-langkah dithe section called "Menambahkan AWS Glue koneksi". Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk penggunaan masa depan di GlueAWS.

   - Saat memilih jenis Koneksi, pilih Azure SQL.
   - Saat menyediakan URL Azure SQL, berikan URL endpoint JDBC.

     URL harus dalam format:
     `jdbc:sqlserver://`*databaseServerName*`:`*databasePort*`;databaseName=`*azuresqlDBnam*

     AWSGlue membutuhkan properti URL berikut:

     - `databaseName`— Database default di Azure SQL untuk terhubung ke.

     Untuk informasi selengkapnya tentang URL JDBC untuk Instans Terkelola Azure SQL, lihat dokumentasi Microsoft.

   - Saat memilih AWSSecret, berikan *secretName*.

Setelah membuat koneksi AWS Glue Azure SQL, Anda harus melakukan langkah-langkah berikut sebelum menjalankan pekerjaan AWS Glue Anda:

- Berikan peran IAM yang terkait dengan izin pekerjaan AWS Glue Anda untuk membaca *secretName*.
- Dalam konfigurasi pekerjaan AWS Glue Anda, berikan *ConnectionName* sebagai koneksi jaringan Tambahan.

Membaca dari tabel Azure SQL

Prasyarat:

- Tabel Azure SQL yang ingin Anda baca. *Anda akan memerlukan informasi identifikasi untuk tabel, DatabaseName dan TableIdentifier.*

  Tabel Azure SQL diidentifikasi oleh database, skema, dan nama tabelnya. Anda harus memberikan nama database dan nama tabel saat menghubungkan ke Azure SQL. Anda juga harus memberikan skema jika bukan default, “publik”. Database disediakan melalui properti URL di *connectionName*, skema dan nama tabel melalui. `dbtable`
- Koneksi AWS Glue Azure SQL yang dikonfigurasi untuk memberikan informasi autentikasi. Selesaikan langkah-langkah dalam prosedur sebelumnya, Untuk mengkonfigurasi koneksi ke Azure SQL untuk mengkonfigurasi informasi autentikasi Anda. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.

Misalnya:

```
azuresql_read_table = glueContext.create_dynamic_frame.from_options(
    connection_type="azuresql",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableIdentifier"
    }
)
```

Anda juga dapat memberikan kueri SELECT SQL, untuk memfilter hasil yang dikembalikan ke Anda DynamicFrame. Anda perlu mengkonfigurasi`query`.

Misalnya:

```
azuresql_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="azuresql",
    connection_options={
        "connectionName": "connectionName",
        "query": "query"
    }
)
```

Menulis ke tabel Azure SQL

Contoh ini menulis informasi dari *DynamicFrame* yang sudah ada DynamicFrame ke Azure SQL. Jika tabel sudah memiliki informasi, AWS Glue akan menambahkan data dari Anda DynamicFrame.

Prasyarat:

- Tabel Azure SQL yang ingin Anda tulis. *Anda akan memerlukan informasi identifikasi untuk tabel, DatabaseName dan TableIdentifier.*

  Tabel Azure SQL diidentifikasi oleh database, skema, dan nama tabelnya. Anda harus memberikan nama database dan nama tabel saat menghubungkan ke Azure SQL. Anda juga harus memberikan skema jika bukan default, “publik”. Database disediakan melalui properti URL di *connectionName*, skema dan nama tabel melalui. `dbtable`
- Informasi autentikasi Azure SQL. Selesaikan langkah-langkah dalam prosedur sebelumnya, Untuk mengkonfigurasi koneksi ke Azure SQL untuk mengkonfigurasi informasi autentikasi Anda. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.

Misalnya:

```
azuresql_write = glueContext.write_dynamic_frame.from_options(
    connection_type="azuresql",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableIdentifier"
    }
)
```

Referensi opsi koneksi Azure SQL

- `connectionName` — Diperlukan. Digunakan untuk Baca/tulis. Nama koneksi AWS Glue Azure SQL dikonfigurasi untuk memberikan informasi autentikasi ke metode koneksi Anda.
- `databaseName`— Digunakan untuk Membaca/Menulis. Nilai yang Valid: Nama database Azure SQL. Nama basis data di Azure SQL untuk terhubung.
- `dbtable`— Diperlukan untuk menulis, diperlukan untuk membaca kecuali `query` disediakan. Digunakan untuk Baca/tulis. Nilai Valid: Nama tabel Azure SQL, atau kombinasi nama skema/tabel yang dipisahkan periode. Digunakan untuk menentukan tabel dan skema yang mengidentifikasi tabel untuk terhubung ke. Skema default adalah “public”. Jika tabel Anda berada dalam skema non-default, berikan informasi ini dalam formulir. *schemaName*.*tableName*

- `query`— Digunakan untuk Baca. Sebuah query Transact-SQL SELECT mendefinisikan apa yang harus diambil ketika membaca dari Azure SQL. Untuk informasi selengkapnya, lihat dokumentasi Microsoft.

BigQuery koneksi

Anda dapat menggunakan AWS Glue for Spark untuk membaca dan menulis ke tabel di Google BigQuery di AWS Glue 4.0 dan versi yang lebih baru. Anda dapat membaca dari BigQuery dengan kueri Google SQL. Anda terhubung BigQuery menggunakan kredensional yang disimpan AWS Secrets Manager melalui koneksi AWS Glue.

Untuk informasi selengkapnya tentang Google BigQuery, lihat BigQuery situs web Google Cloud.

Mengkonfigurasi koneksi BigQuery

Untuk terhubung ke Google BigQuery dari AWS Glue, Anda harus membuat dan menyimpan kredensi Google Cloud Platform Anda secara AWS Secrets Manager rahasia, lalu mengaitkan rahasia itu dengan koneksi Google BigQuery AWS Glue.

Untuk mengkonfigurasi koneksi ke BigQuery:

1. Di Google Cloud Platform, buat dan identifikasi sumber daya yang relevan:
   - Buat atau identifikasi proyek GCP yang berisi BigQuery tabel yang ingin Anda sambungkan.
   - Aktifkan BigQuery API. Untuk informasi selengkapnya, lihat Menggunakan BigQuery Storage Read API untuk membaca data tabel.
2. Di Google Cloud Platform, buat dan ekspor kredensional akun layanan:

   Anda dapat menggunakan panduan BigQuery kredensial untuk mempercepat langkah ini: Buat kredensial.

   Untuk membuat akun layanan di GCP, ikuti tutorial yang tersedia di Buat akun layanan.
   - Saat memilih proyek, pilih proyek yang berisi BigQuery tabel Anda.
   - Saat memilih peran IAM GCP untuk akun layanan Anda, tambahkan atau buat peran yang akan memberikan izin yang sesuai untuk menjalankan BigQuery pekerjaan untuk membaca, menulis, atau membuat tabel. BigQuery

Untuk membuat kredensi untuk akun layanan Anda, ikuti tutorial yang tersedia di Buat kunci akun layanan.

- Saat memilih jenis kunci, pilih JSON.

Anda seharusnya sudah mengunduh file JSON dengan kredensi untuk akun layanan Anda. Itu terlihat serupa dengan yang berikut ini:

```
{
  "type": "service_account",
  "project_id": "*****",
  "private_key_id": "*****",
  "private_key": "*****",
  "client_email": "*****",
  "client_id": "*****",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "*****",
  "universe_domain": "googleapis.com"
}
```

3. base64 menyandikan file kredensi yang Anda unduh. Pada AWS CloudShell sesi atau serupa, Anda dapat melakukan ini dari baris perintah dengan menjalankan`cat` *`credentialsFile.json`* `| base64 -w 0`. Pertahankan output dari perintah ini, *`CredentialString`*.
4. DiAWS Secrets Manager, buat rahasia menggunakan kredensi Google Cloud Platform Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *`secretName`* untuk langkah selanjutnya.

   - *Saat memilih pasangan* ***kunci/nilai, buat pasangan*** *untuk kunci credentials dengan nilai CredentialString.*
5. Dalam AWS Glue Data Catalog, buat koneksi dengan mengikuti langkah-langkah dithe section called “Menambahkan AWS Glue koneksi”. Setelah membuat koneksi, simpan nama koneksi, *`connectionName`*, untuk langkah selanjutnya.

   - Saat memilih jenis Koneksi, pilih Google BigQuery.

- Saat memilih AWSSecret, berikan *secretName*.

6. Berikan peran IAM yang terkait dengan izin pekerjaan AWS Glue Anda untuk membaca *secretName*.
7. Dalam konfigurasi pekerjaan AWS Glue Anda, berikan *ConnectionName* sebagai koneksi jaringan Tambahan.

Membaca dari BigQuery tabel

Prasyarat:

- BigQuery Meja yang ingin Anda baca. Anda akan memerlukan nama BigQuery tabel dan dataset, dalam formulir`[dataset].[table]`. Mari kita sebut ini *TableName*.
- Proyek penagihan untuk BigQuery tabel. Anda akan membutuhkan nama proyek, *ParentProject*. Jika tidak ada proyek induk penagihan, gunakan proyek yang berisi tabel.
- BigQuery informasi autentikasi. Selesaikan langkah-langkahnya Untuk mengelola kredensi koneksi Anda dengan AWS Glue untuk mengonfigurasi informasi autentikasi Anda. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.

Misalnya:

```
bigquery_read = glueContext.create_dynamic_frame.from_options(
    connection_type="bigquery",
    connection_options={
        "connectionName": "connectionName",
        "parentProject": "parentProject",
        "sourceType": "table",
        "table": "tableName",
    }
```

Anda juga dapat memberikan kueri, untuk memfilter hasil yang dikembalikan ke Anda DynamicFrame. Anda perlu mengkonfigurasi`query`,`sourceType`, `viewsEnabled` dan`materializationDataset`.

Misalnya:

Prasyarat tambahan:

Anda perlu membuat atau mengidentifikasi BigQuery kumpulan data, *MaterializationDataset, tempat BigQuery dapat menulis tampilan terwujud* untuk kueri Anda.

Anda harus memberikan izin IAM GCP yang sesuai ke akun layanan Anda untuk membuat tabel di MaterializationDataset.

```
glueContext.create_dynamic_frame.from_options(
            connection_type="bigquery",
            connection_options={
                "connectionName": "connectionName",
                "materializationDataset": materializationDataset,
                "parentProject": "parentProject",
                "viewsEnabled": "true",
                "sourceType": "query",
                "query": "select * from bqtest.test"
            }
        )
```

Menulis ke BigQuery tabel

Contoh ini menulis langsung ke BigQuery layanan. BigQuery juga mendukung metode penulisan "tidak langsung". Untuk informasi selengkapnya tentang mengonfigurasi penulisan tidak langsung, lihatthe section called "Menggunakan penulisan tidak langsung dengan Google BigQuery".

Prasyarat:

- BigQuery Meja yang ingin Anda tulis. Anda akan memerlukan nama BigQuery tabel dan dataset, dalam formulir`[dataset].[table]`. Anda juga dapat memberikan nama tabel baru yang akan dibuat secara otomatis. Mari kita sebut ini *TableName*.
- Proyek penagihan untuk BigQuery tabel. Anda akan membutuhkan nama proyek, *ParentProject*. Jika tidak ada proyek induk penagihan, gunakan proyek yang berisi tabel.
- BigQuery informasi autentikasi. Selesaikan langkah-langkahnya Untuk mengelola kredensi koneksi Anda dengan AWS Glue untuk mengonfigurasi informasi autentikasi Anda. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.

Misalnya:

```
bigquery_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="bigquery",
```

```
    connection_options={
        "connectionName": "connectionName",
        "parentProject": "parentProject",
        "writeMethod": "direct",
        "table": "tableName",
    }
)
```

BigQuery referensi opsi koneksi

- `project`— Default: default akun layanan Google Cloud. Digunakan untuk Baca/Tulis. Nama proyek Google Cloud yang terkait dengan tabel Anda.
- `table`— (Wajib) Digunakan untuk Baca/Tulis. Nama BigQuery tabel Anda dalam format`[[project:]dataset.]`.
- `dataset`— Diperlukan ketika tidak ditentukan melalui `table` opsi. Digunakan untuk Baca/Tulis. Nama dataset yang berisi BigQuery tabel Anda.
- `parentProject`— Default: default akun layanan Google Cloud. Digunakan untuk Baca/Tulis. Nama proyek Google Cloud yang terkait dengan `project` digunakan untuk penagihan.
- `sourceType`— Digunakan untuk Baca. Diperlukan saat membaca. Nilai yang Valid:`table`, `query` Menginformasikan AWS Glue apakah Anda akan membaca berdasarkan tabel atau kueri.
- `materializationDataset`— Digunakan untuk Baca. Nilai Valid: string. Nama BigQuery dataset yang digunakan untuk menyimpan materialisasi untuk tampilan.
- `viewsEnabled`— Digunakan untuk Baca. Default: false. Nilai Valid: true, false. Mengkonfigurasi apakah BigQuery akan menggunakan tampilan.
- `query`— Digunakan untuk Baca. Digunakan saat `viewsEnabled` itu benar. Kueri GoogleSQL DQL.
- `temporaryGcsBucket`Digunakan untuk menulis. Diperlukan saat `writeMethod` diatur ke default (`indirect`). Nama bucket Google Cloud Storage yang digunakan untuk menyimpan bentuk perantara data Anda saat menulis ke BigQuery.
- `writeMethod`— Default:`indirect`. Nilai yang Valid:`direct`,`indirect`. Digunakan untuk menulis. Menentukan metode yang digunakan untuk menulis data Anda.
  - Jika disetel ke`direct`, konektor Anda akan menulis menggunakan BigQuery Storage Write API.
  - Jika disetel ke`indirect`, konektor Anda akan menulis ke Google Cloud Storage, lalu mentransfernya ke BigQuery menggunakan operasi pemuatan. Akun layanan Google Cloud Anda akan memerlukan izin GCS yang sesuai.

Menggunakan penulisan tidak langsung dengan Google BigQuery

Contoh ini menggunakan penulisan tidak langsung, yang menulis data ke Google Cloud Storage dan menyalinnya ke Google BigQuery.

Prasyarat:

Anda akan memerlukan bucket Google Cloud Storage sementara, *TemporaryBucket*.

Peran GCP IAM untuk akun layanan GCP AWS Glue akan memerlukan izin GCS yang sesuai untuk mengakses TemporaryBucket.

Konfigurasi Tambahan:

Untuk mengonfigurasi penulisan tidak langsung dengan BigQuery:

1. Menilai the section called “Mengkonfigurasi BigQuery” dan menemukan atau men-download ulang file JSON kredensi GCP Anda. Identifikasi *SecretName*, AWS Secrets Manager rahasia untuk koneksi Google BigQuery AWS Glue yang digunakan dalam pekerjaan Anda.
2. Unggah file JSON kredensional Anda ke lokasi Amazon S3 yang aman dengan tepat. Pertahankan path ke file, *s3secretpath* untuk langkah-langkah masa depan.
3. Edit *secretName*, menambahkan kunci. `spark.hadoop.google.cloud.auth.service.account.json.keyfile` Tetapkan nilainya ke *s3secretpath*.
4. *Berikan izin IAM Amazon S3 pekerjaan AWS Glue Anda untuk mengakses s3secretpath.*

Sekarang Anda dapat memberikan lokasi bucket GCS sementara ke metode penulisan Anda. Anda tidak perlu menyediakan`writeMethod`, seperti yang secara `indirect` historis default.

```
bigquery_write = glueContext.write_dynamic_frame.from_options(
    frame=frameToWrite,
    connection_type="bigquery",
    connection_options={
        "connectionName": "connectionName",
        "parentProject": "parentProject",
        "temporaryGcsBucket": "temporaryBucket",
        "table": "tableName",
    }
```

```
)
```

Koneksi JDBC

Jenis database tertentu, biasanya relasional, mendukung koneksi melalui standar JDBC. Untuk informasi selengkapnya tentang JDBC, lihat dokumentasi Java JDBC API. AWS Glue secara native mendukung koneksi ke database tertentu melalui konektor JDBC mereka - perpustakaan JDBC disediakan dalam pekerjaan Glue Spark. AWS Saat menghubungkan ke tipe database ini menggunakan pustaka AWS Glue, Anda memiliki akses ke serangkaian opsi standar.

Nilai ConnectionType JDBC meliputi yang berikut:

- `"connectionType": "sqlserver"`: Mengkhususkan koneksi ke Microsoft SQL Server.
- `"connectionType": "mysql"`: Mengkhususkan koneksi ke basis data MySQL.
- `"connectionType": "oracle"`: Mengkhususkan koneksi ke basis data Oracle.
- `"connectionType": "postgresql"`: Mengkhususkan koneksi ke basis data PostgreSQL.
- `"connectionType": "redshift"`: Mengkhususkan koneksi ke basis data Amazon Redshift. Untuk informasi selengkapnya, lihat the section called “Koneksi Redshift”.

Tabel berikut mencantumkan versi driver JDBC yang AWS Glue mendukung.

Produk	Versi driver JDBC untuk Glue 4.0	Versi driver JDBC untuk Glue 3.0	Versi driver JDBC untuk Glue 0.9, 1.0, 2.0
Microsoft SQL Server	9.4.0	7.x	6.x
MySQL	8.0.23	8.0.23	5.1
Basis data Oracle	21.7	21.1	11.2
PostgreSQL	42.3.6	42.2.18	42.1.x
MongoDB	4.7.2	4.0.0	2.0.0
Pergeseran Merah Amazon*	redshift-jdbc42-2.1.0.16	redshift-jdbc41-1.2.12.1017	redshift-jdbc41-1.2.12.1017

* Untuk jenis koneksi Amazon Redshift, semua pasangan nama/nilai opsi lain yang disertakan dalam opsi koneksi untuk koneksi JDBC, termasuk opsi pemformatan, diteruskan langsung ke SparkSQL yang mendasarinya. DataSource Dalam pekerjaan AWS Glue with Spark di AWS Glue 4.0 dan versi yang lebih baru, konektor asli AWS Glue untuk Amazon Redshift menggunakan integrasi Amazon Redshift untuk Apache Spark. Untuk informasi selengkapnya, lihat integrasi Amazon Redshift untuk Apache Spark. Di versi sebelumnya, lihat Sumber data Amazon Redshift untuk Spark.

Untuk mengonfigurasi VPC Amazon agar tersambung ke penyimpanan data Amazon RDS menggunakan JDBC, lihat. the section called “Menyiapkan Amazon VPC untuk terhubung ke penyimpanan data Amazon RDS”

> **Note**
>
> AWSGlue jobs hanya dikaitkan dengan satu subnet selama menjalankan. Ini dapat memengaruhi kemampuan Anda untuk terhubung ke beberapa sumber data melalui pekerjaan yang sama. Perilaku ini tidak terbatas pada sumber JDBC.

Topik

- Referensi opsi koneksi JDBC
- Gunakan SampleQuery
- Gunakan driver JDBC khusus
- Membaca dari tabel JDBC secara paralel
- Menyiapkan Amazon VPC untuk koneksi JDBC ke penyimpanan data Amazon RDS AWS Glue

Referensi opsi koneksi JDBC

Jika Anda sudah memiliki koneksi AWS Glue JDBC yang ditentukan, Anda dapat menggunakan kembali properti konfigurasi yang ditentukan di dalamnya, seperti: url, pengguna, dan kata sandi; jadi Anda tidak perlu menentukannya dalam kode sebagai opsi koneksi. Fitur ini tersedia dalam AWS Glue 3.0 dan versi yang lebih baru. Untuk melakukannya, gunakan properti koneksi berikut:

- `"useConnectionProperties"`: Setel ke “true” untuk menunjukkan Anda ingin menggunakan konfigurasi dari koneksi.
- `"connectionName"`: Masukkan nama koneksi untuk mengambil konfigurasi dari, koneksi harus ditentukan di wilayah yang sama dengan pekerjaan.

Gunakan opsi koneksi ini dengan koneksi JDBC:

- `"url"`: (Wajib) URL JDBC untuk basis data.
- `"dbtable"`: (Wajib) Tabel database untuk dibaca. Untuk penyimpanan data JDBC yang mendukung skema dalam basis data, tentukan `schema.table-name`. Jika skema tidak disediakan, maka skema "publik" default digunakan.
- `"user"`: (Wajib) Nama pengguna yang akan digunakan saat terhubung.
- `"password"`: (Wajib) Kata sandi yang akan digunakan saat terhubung.
- (Opsional) Opsi berikut memungkinkan Anda untuk memberikan driver JDBC kustom. Gunakan opsi ini jika Anda harus menggunakan driver yang AWS Glue tidak mendukung secara asli.

  Tugas ETL dapat menggunakan versi driver JDBC yang berbeda untuk sumber data dan target, bahkan jika sumber dan target adalah produk basis data yang sama. Hal ini memungkinkan Anda untuk memigrasi data antara sumber dan target basis data dengan versi yang berbeda. Untuk menggunakan opsi ini, Anda harus terlebih dahulu mengunggah file JAR dari driver JDBC ke Amazon S3.

  - `"customJdbcDriverS3Path"`: Jalur Amazon S3 dari driver JDBC khusus.
  - `"customJdbcDriverClassName"`: Nama kelas driver JDBC.
- `"bulkSize"`: (Opsional) Digunakan untuk mengkonfigurasi sisipan paralel untuk mempercepat beban massal ke target JDBC. Tentukan nilai integer untuk tingkat paralelisme yang akan digunakan saat menulis atau memasukkan data. Opsi ini berguna untuk meningkatkan kinerja penulisan ke dalam database seperti Arch User Repository (AUR).
- `"hashfield"`(Opsional) String, digunakan untuk menentukan nama kolom dalam tabel JDBC yang akan digunakan untuk membagi data menjadi partisi saat membaca dari tabel JDBC secara paralel. Berikan “hashfield” ATAU “hashexpression”. Untuk informasi selengkapnya, lihat the section called “Membaca dari JDBC secara paralel”.
- `"hashexpression"`(Opsional) Klausa pilih SQL mengembalikan bilangan bulat. Digunakan untuk membagi data dalam tabel JDBC menjadi partisi saat membaca dari tabel JDBC secara paralel. Berikan “hashfield” ATAU “hashexpression”. Untuk informasi selengkapnya, lihat the section called “Membaca dari JDBC secara paralel”.
- `"hashpartitions"`(Opsional) Sebuah bilangan bulat positif. Digunakan untuk menentukan jumlah pembacaan paralel dari tabel JDBC saat membaca dari tabel JDBC secara paralel. Default: 7. Untuk informasi selengkapnya, lihat the section called “Membaca dari JDBC secara paralel”.
- `"sampleQuery"`: (Opsional) Pernyataan kueri SQL kustom. Digunakan untuk menentukan subset informasi dalam tabel untuk mengambil sampel isi tabel. Ketika dikonfigurasi tanpa memperhatikan

data Anda, itu bisa kurang efisien daripada DynamicFrame metode, menyebabkan batas waktu atau kehabisan kesalahan memori. Untuk informasi selengkapnya, lihat the section called “Gunakan SampleQuery”.

- `"enablePartitioningForSampleQuery"`: (Opsional) Sebuah boolean. Default: false. Digunakan untuk mengaktifkan membaca dari tabel JDBC secara paralel saat menentukan. `sampleQuery` Jika disetel ke true, **`sampleQuery`** harus diakhiri dengan “where” atau “and” agar AWS Glue menambahkan kondisi partisi. Untuk informasi selengkapnya, lihat the section called “Gunakan SampleQuery”.
- `"sampleSize"`: (Opsional) Sebuah bilangan bulat positif. Membatasi jumlah baris yang dikembalikan oleh kueri sampel. Bekerja hanya ketika `enablePartitioningForSampleQuery` itu benar. Jika partisi tidak diaktifkan, Anda harus langsung menambahkan `"limit x"` `sampleQuery` untuk membatasi ukuran. Untuk informasi selengkapnya, lihat the section called “Gunakan SampleQuery”.

Gunakan SampleQuery

Bagian ini menjelaskan cara menggunakan`sampleQuery`, `sampleSize` dan`enablePartitioningForSampleQuery`.

`sampleQuery`dapat menjadi cara yang efisien untuk mengambil sampel beberapa baris dataset Anda. Secara default, kueri dijalankan oleh seorang eksekutor tunggal. Ketika dikonfigurasi tanpa memperhatikan data Anda, itu bisa kurang efisien daripada DynamicFrame metode, menyebabkan batas waktu atau kehabisan kesalahan memori. Menjalankan SQL pada database yang mendasarinya sebagai bagian dari pipeline ETL Anda umumnya hanya diperlukan untuk tujuan kinerja. Jika Anda mencoba untuk melihat pratinjau beberapa baris dataset Anda, pertimbangkan untuk menggunakanthe section called “show”. Jika Anda mencoba untuk mengubah dataset Anda menggunakan SQL, pertimbangkan the section called “toDF” untuk menggunakan untuk mendefinisikan transformasi SparkSQL terhadap data Anda dalam formulir. DataFrame

Meskipun kueri Anda dapat memanipulasi berbagai tabel, `dbtable` tetap diperlukan.

Menggunakan SampleQuery untuk mengambil sampel tabel Anda

Saat menggunakan perilaku SampleQuery default untuk mengambil sampel data Anda, AWS Glue tidak mengharapkan throughput yang substansif, sehingga menjalankan kueri Anda pada satu eksekutor. Untuk membatasi data yang Anda berikan dan tidak menyebabkan masalah kinerja, kami sarankan Anda memberikan SQL dengan `LIMIT` klausa.

Example Gunakan SampleQuery tanpa partisi

Contoh kode berikut menunjukkan bagaimana menggunakan `sampleQuery` tanpa partisi.

```
//A full sql query statement.
val query = "select name from $tableName where age > 0 limit 1"
val connectionOptions = JsonOptions(Map(
    "url" -> url,
    "dbtable" -> tableName,
    "user" -> user,
    "password" -> password,
    "sampleQuery" -> query ))
val dyf = glueContext.getSource("mysql", connectionOptions)
          .getDynamicFrame()
```

Menggunakan SampleQuery terhadap kumpulan data yang lebih besar

Jika Anda membaca kumpulan data besar, Anda mungkin perlu mengaktifkan partisi JDBC untuk menanyakan tabel secara paralel. Untuk informasi selengkapnya, lihat the section called “Membaca dari JDBC secara paralel”. Untuk digunakan `sampleQuery` dengan partisi JDBC, atur ke true. `enablePartitioningForSampleQuery` Mengaktifkan fitur ini mengharuskan Anda untuk membuat beberapa perubahan pada Anda`sampleQuery`.

Saat menggunakan partisi JDBC dengan`sampleQuery`, kueri Anda harus diakhiri dengan “where” atau “and” agar AWS Glue menambahkan kondisi partisi.

Jika Anda ingin membatasi hasil SampleQuery Anda saat membaca dari tabel JDBC secara paralel, atur `"sampleSize"` parameter daripada menentukan klausa. `LIMIT`

Example Gunakan SampleQuery dengan partisi JDBC

Contoh kode berikut menunjukkan bagaimana menggunakan `sampleQuery` dengan partisi JDBC.

```
//note that the query should end with "where" or "and" if use with JDBC partitioning.
val query = "select name from $tableName where age > 0 and"

//Enable JDBC partitioning by setting hashfield.
//to use sampleQuery with partitioning, set enablePartitioningForSampleQuery.
//use sampleSize to limit the size of returned data.
val connectionOptions = JsonOptions(Map(
    "url" -> url,
    "dbtable" -> tableName,
    "user" -> user,
```

```
    "password" -> password,
    "hashfield" -> primaryKey,
    "sampleQuery" -> query,
    "enablePartitioningForSampleQuery" -> true,
    "sampleSize" -> "1" ))
val dyf = glueContext.getSource("mysql", connectionOptions)
          .getDynamicFrame()
```

Catatan dan Pembatasan:

Contoh kueri tidak dapat digunakan bersama dengan bookmark pekerjaan. Status bookmark akan diabaikan ketika konfigurasi untuk keduanya disediakan.

Gunakan driver JDBC khusus

Contoh kode berikut ini menunjukkan cara membaca dan menulis ke basis data JDBC dengan driver JDBC kustom. Mereka menunjukkan membaca dari satu versi produk database, dan menulis ke versi yang lebih baru dari produk yang sama.

Python

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext, SparkConf
from awsglue.context import GlueContext
from awsglue.job import Job
import time
from pyspark.sql.types import StructType, StructField, IntegerType, StringType

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

# Construct JDBC connection options
connection_mysql5_options = {
    "url": "jdbc:mysql://<jdbc-host-name>:3306/db",
    "dbtable": "test",
    "user": "admin",
    "password": "pwd"}

connection_mysql8_options = {
    "url": "jdbc:mysql://<jdbc-host-name>:3306/db",
```

```
    "dbtable": "test",
    "user": "admin",
    "password": "pwd",
    "customJdbcDriverS3Path": "s3://DOC-EXAMPLE-BUCKET/mysql-connector-
java-8.0.17.jar",
    "customJdbcDriverClassName": "com.mysql.cj.jdbc.Driver"}

connection_oracle11_options = {
    "url": "jdbc:oracle:thin:@//<jdbc-host-name>:1521/ORCL",
    "dbtable": "test",
    "user": "admin",
    "password": "pwd"}

connection_oracle18_options = {
    "url": "jdbc:oracle:thin:@//<jdbc-host-name>:1521/ORCL",
    "dbtable": "test",
    "user": "admin",
    "password": "pwd",
    "customJdbcDriverS3Path": "s3://DOC-EXAMPLE-BUCKET/ojdbc10.jar",
    "customJdbcDriverClassName": "oracle.jdbc.OracleDriver"}

# Read from JDBC databases with custom driver
df_mysql8 = glueContext.create_dynamic_frame.from_options(connection_type="mysql",

 connection_options=connection_mysql8_options)

# Read DynamicFrame from MySQL 5 and write to MySQL 8
df_mysql5 = glueContext.create_dynamic_frame.from_options(connection_type="mysql",

 connection_options=connection_mysql5_options)
glueContext.write_from_options(frame_or_dfc=df_mysql5, connection_type="mysql",
                               connection_options=connection_mysql8_options)

# Read DynamicFrame from Oracle 11 and write to Oracle 18
df_oracle11 =
 glueContext.create_dynamic_frame.from_options(connection_type="oracle",

 connection_options=connection_oracle11_options)
glueContext.write_from_options(frame_or_dfc=df_oracle11, connection_type="oracle",
                               connection_options=connection_oracle18_options)
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamicFrame
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._


object GlueApp {
  val MYSQL_5_URI: String = "jdbc:mysql://<jdbc-host-name>:3306/db"
  val MYSQL_8_URI: String = "jdbc:mysql://<jdbc-host-name>:3306/db"
  val ORACLE_11_URI: String = "jdbc:oracle:thin:@//<jdbc-host-name>:1521/ORCL"
  val ORACLE_18_URI: String = "jdbc:oracle:thin:@//<jdbc-host-name>:1521/ORCL"

  // Construct JDBC connection options
  lazy val mysql5JsonOption = jsonOptions(MYSQL_5_URI)
  lazy val mysql8JsonOption = customJDBCDriverJsonOptions(MYSQL_8_URI, "s3://DOC-
EXAMPLE-BUCKET/mysql-connector-java-8.0.17.jar", "com.mysql.cj.jdbc.Driver")
  lazy val oracle11JsonOption = jsonOptions(ORACLE_11_URI)
  lazy val oracle18JsonOption = customJDBCDriverJsonOptions(ORACLE_18_URI, "s3://
DOC-EXAMPLE-BUCKET/ojdbc10.jar", "oracle.jdbc.OracleDriver")

  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    // Read from JDBC database with custom driver
    val df_mysql8: DynamicFrame = glueContext.getSource("mysql",
 mysql8JsonOption).getDynamicFrame()

    // Read DynamicFrame from MySQL 5 and write to MySQL 8
    val df_mysql5: DynamicFrame = glueContext.getSource("mysql",
 mysql5JsonOption).getDynamicFrame()
    glueContext.getSink("mysql", mysql8JsonOption).writeDynamicFrame(df_mysql5)

    // Read DynamicFrame from Oracle 11 and write to Oracle 18
```

```
    val df_oracle11: DynamicFrame = glueContext.getSource("oracle",
 oracle11JsonOption).getDynamicFrame()
    glueContext.getSink("oracle", oracle18JsonOption).writeDynamicFrame(df_oracle11)

    Job.commit()
  }

  private def jsonOptions(url: String): JsonOptions = {
    new JsonOptions(
      s"""{"url": "${url}",
         |"dbtable":"test",
         |"user": "admin",
         |"password": "pwd"}""".stripMargin)
  }

  private def customJDBCDriverJsonOptions(url: String, customJdbcDriverS3Path:
 String, customJdbcDriverClassName: String): JsonOptions = {
    new JsonOptions(
      s"""{"url": "${url}",
         |"dbtable":"test",
         |"user": "admin",
         |"password": "pwd",
         |"customJdbcDriverS3Path": "${customJdbcDriverS3Path}",
         |"customJdbcDriverClassName" :
 "${customJdbcDriverClassName}"}""".stripMargin)
  }
}
```

Membaca dari tabel JDBC secara paralel

Anda dapat mengatur properti dari tabel JDBC Anda untuk memungkinkan AWS Glue membaca data secara paralel. Ketika Anda mengatur properti tertentu, Anda menginstruksikan AWS Glue untuk menjalankan kueri SQL paralel terhadap partisi logis data Anda. Anda dapat mengontrol pemartisian dengan menetapkan bidang hash atau ekspresi hash. Anda juga dapat mengontrol jumlah baca paralel yang digunakan untuk mengakses data Anda.

Membaca dari tabel JDBC secara paralel adalah teknik optimasi yang dapat meningkatkan kinerja. Untuk informasi lebih lanjut tentang proses mengidentifikasi kapan teknik ini tepat, lihat Kurangi jumlah pemindaian data dalam Praktik terbaik untuk penyetelan AWS Glue kinerja untuk panduan pekerjaan Apache Spark pada AWS Panduan Preskriptif.

Untuk memungkinkan baca paralel, Anda dapat mengatur pasangan nilai-kunci di bidang parameter struktur tabel Anda. Gunakan notasi JSON untuk menetapkan nilai untuk bidang parameter tabel Anda. Untuk informasi lebih lanjut tentang mengedit properti tabel, lihat Melihat dan mengedit detail tabel. Anda juga dapat mengaktifkan baca paralel ketika Anda memanggil metode ETL (extract, transform, dan load) `create_dynamic_frame_from_options` dan `create_dynamic_frame_from_catalog`. Untuk informasi lebih lanjut tentang menentukan opsi dalam metode ini, lihat from_options dan from_catalog.

Anda dapat menggunakan metode ini untuk tabel JDBC, yaitu, sebagian besar tabel yang data dasar-nya merupakan penyimpanan data JDBC. Properti ini diabaikan saat membaca tabel Amazon Redshift dan tabel Amazon S3.

hashfield

Atur `hashfield` dengan nama kolom dalam tabel JDBC yang akan digunakan untuk membagi data menjadi partisi. Untuk hasil terbaik, kolom ini harus memiliki distribusi nilai yang genap untuk menyebarkan data antar partisi. Kolom ini dapat berupa tipe data apa pun. AWS Glue menghasilkan kueri yang tidak bertumpang tindih yang berjalan secara paralel untuk membaca data yang dipartisi oleh kolom ini. Misalnya, jika data Anda didistribusikan secara merata berdasarkan bulan, Anda dapat menggunakan kolom `month` untuk membaca setiap bulan data secara paralel.

```
'hashfield': 'month'
```

AWS Glue membuat sebuah kueri untuk meng-hash nilai bidang ke nomor partisi dan menjalankan kueri untuk semua partisi secara paralel. Untuk menggunakan kueri Anda sendiri untuk melakukan partisi pada pembacaan tabel, berikan `hashexpression`, bukan sebuah `hashfield`.

hashexpression

Atur `hashexpression` dengan ekspresi SQL (sesuai dengan tata bahasa mesin basis data JDBC) yang mengembalikan bilangan bulat. Sebuah ekspresi sederhana adalah nama dari setiap kolom numerik dalam tabel. AWS Glue menghasilkan kueri SQL untuk membaca data JDBC secara paralel menggunakan `hashexpression` di `WHERE` untuk melakukan partisi data.

Misalnya, gunakan kolom numerik `customerID` untuk membaca data yang dipartisi berdasarkan nomor pelanggan.

```
'hashexpression': 'customerID'
```

Untuk membuat AWS Glue mengontrol partisi, berikan `hashfield`, bukan sebuah `hashexpression`.

hashpartitions

Atur `hashpartitions` dengan jumlah baca paralel tabel JDBC. Jika properti ini tidak diatur, maka nilai default 7 akan digunakan.

Sebagai contoh, tetapkan jumlah baca paralel ke 5 sehingga AWS Glue membaca data Anda dengan lima kueri (atau lebih sedikit).

```
'hashpartitions': '5'
```

Menyiapkan Amazon VPC untuk koneksi JDBC ke penyimpanan data Amazon RDS AWS Glue

Saat menggunakan JDBC untuk terhubung ke database di Amazon RDS, Anda perlu melakukan pengaturan tambahan. Untuk mengaktifkan AWS Glue komponen untuk berkomunikasi dengan Amazon RDS, Anda harus mengatur akses ke penyimpanan data Amazon RDS Anda di Amazon VPC. Untuk mengaktifkan AWS Glue untuk berkomunikasi antara komponen-komponennya, tentukan grup keamanan dengan aturan inbound self-referencing untuk semua port TCP. Dengan membuat aturan referensi diri, Anda dapat membatasi sumber ke grup keamanan yang sama di VPC. Aturan referensi diri tidak akan membuka VPC ke semua jaringan. Grup keamanan default untuk VPC Anda mungkin sudah memiliki aturan self-referencing inbound untuk semua lalu lintas.

Untuk mengatur akses antara penyimpanan data AWS Glue dan Amazon RDS

1. Masuk ke AWS Management Console dan buka konsol Amazon RDS di https://console.aws.amazon.com/rds/.
2. Di konsol Amazon RDS, identifikasi grup keamanan yang digunakan untuk mengontrol akses ke database Amazon RDS Anda.

   Di panel navigasi kiri, pilih Databases, lalu pilih instance yang ingin Anda sambungkan dari daftar di panel utama.

Di halaman detail database, temukan grup keamanan VPC di tab Konektivitas & keamanan.

3. Berdasarkan arsitektur jaringan Anda, identifikasi grup keamanan terkait mana yang terbaik untuk dimodifikasi untuk memungkinkan akses ke layanan AWS Glue. Simpan namanya, *database-security-group*untuk referensi masa depan. Jika tidak ada grup keamanan yang sesuai, ikuti petunjuk untuk Menyediakan akses ke instans DB Anda di VPC Anda dengan membuat grup keamanan dalam dokumentasi Amazon RDS.
4. Masuk ke AWS Management Console dan buka konsol VPC Amazon di https://console.aws.amazon.com/vpc/.
5. Di konsol VPC Amazon, identifikasi cara memperbarui. *database-security-group*

   Di panel navigasi kiri, pilih Grup keamanan, lalu pilih *database-security-group*dari daftar di panel utama.
6. Identifikasi ID grup keamanan untuk *database-security-group*, *database-sg-id*. Simpan untuk referensi masa depan.

   Di halaman detail grup keamanan, temukan ID grup keamanan.
7. Ubah aturan masuk untuk *database-security-group*, tambahkan aturan referensi diri untuk memungkinkan AWS Glue komponen berkomunikasi. Secara khusus, tambahkan atau konfirmasikan bahwa ada aturan di mana Type adalah`All TCP`, Protocol adalah`TCP`, Port Range mencakup semua port, dan Source adalah *database-sg-id*. Verifikasi bahwa grup keamanan yang Anda masukkan untuk Sumber sama dengan grup keamanan yang Anda edit.

   Di halaman detail grup keamanan, pilih Edit aturan masuk.

   Aturan inbound terlihat serupa dengan ini:

Jenis	Protokol	Rentang port	Sumber
Semua TCP	TCP	0–65535	*database-sg-id*

8. Tambahkan aturan untuk lalu lintas keluar.

   Di halaman detail grup keamanan, pilih Edit aturan keluar.

   Jika grup keamanan Anda mengizinkan semua lalu lintas keluar, Anda tidak perlu aturan terpisah. Misalnya:

Jenis	Protokol	Rentang Port	Tujuan
Semua Lalu Lintas	SEMUA	SEMUA	0.0.0.0/0

Jika arsitektur jaringan Anda dirancang untuk membatasi lalu lintas keluar, buat aturan keluar berikut:

Buat aturan referensi mandiri di mana Type adalah`All TCP`, Protocol adalah`TCP`, Port Range mencakup semua port, dan Destination adalah. *`database-sg-id`* Verifikasi bahwa grup keamanan yang Anda masukkan untuk Tujuan sama dengan grup keamanan yang Anda edit.

Jika menggunakan titik akhir VPC Amazon S3, tambahkan aturan HTTPS untuk mengizinkan lalu lintas dari VPC ke Amazon S3. *`Buat aturan di mana` **`Type`** `adalah`HTTPS`, **`Protocol`** `adalah`TCP`, **`Port Range`** `adalah 443 dan` **`Destination`** `adalah ID dari daftar awalan terkelola untuk titik akhir gateway Amazon S3, s3-. prefix-list-id`* Untuk informasi selengkapnya tentang daftar awalan dan titik akhir gateway Amazon S3, lihat titik akhir Gateway untuk Amazon S3 dalam dokumentasi Amazon VPC.

Misalnya:

Jenis	Protokol	Rentang Port	Tujuan
Semua TCP	TCP	0–65535	*`database-sg-id`*
HTTPS	TCP	443	*`s3- prefix-list-id`*

Koneksi MongoDB

Anda dapat menggunakan AWS Glue for Spark untuk membaca dan menulis ke tabel di MongoDB dan AWS MongoDB Atlas di Glue 4.0 dan versi yang lebih baru. Anda dapat terhubung ke MongoDB menggunakan kredensional nama pengguna dan kata sandi yang disimpan melalui koneksi Glue. AWS Secrets Manager AWS

Untuk informasi lebih lanjut tentang MongoDB, lihat dokumentasi MongoDB.

Mengkonfigurasi koneksi MongoDB

Untuk terhubung ke MongoDB AWS dari Glue, Anda akan memerlukan kredensi MongoDB Anda, MongoDbuser dan MongoDBPass.

Untuk terhubung ke MongoDB AWS dari Glue, Anda mungkin memerlukan beberapa prasyarat:

- Jika instans MongoDB Anda ada di VPC Amazon, konfigurasikan Amazon VPC untuk memungkinkan pekerjaan Glue AWS Anda berkomunikasi dengan instans MongoDB tanpa lalu lintas melintasi internet publik.

  Di Amazon VPC, identifikasi atau buat grup VPC, Subnet, dan Keamanan yang akan digunakan AWS Glue saat menjalankan pekerjaan. Selain itu, Anda perlu memastikan Amazon VPC dikonfigurasi untuk mengizinkan lalu lintas jaringan antara instans MongoDB Anda dan lokasi ini. Berdasarkan tata letak jaringan Anda, ini mungkin memerlukan perubahan pada aturan grup keamanan, ACL Jaringan, Gateway NAT, dan koneksi Peering.

Anda kemudian dapat melanjutkan untuk mengkonfigurasi AWS Glue untuk digunakan dengan MongoDB.

Untuk mengkonfigurasi koneksi ke MongoDB:

1. Secara opsional, diAWS Secrets Manager, buat rahasia menggunakan kredensyal MongoDB Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci username dengan nilai MongoDbuser.*

     *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci password dengan nilai MongoDBPass.*

2. Di konsol AWS Glue, buat koneksi dengan mengikuti langkah-langkah dithe section called “Menambahkan AWS Glue koneksi”. Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk penggunaan masa depan di GlueAWS.

   - Saat memilih jenis Koneksi, pilih MongoDB atau MongoDB Atlas.
   - Saat memilih URL MongoDB atau URL MongoDB Atlas, berikan nama host instance MongoDB Anda.

URL MongoDB disediakan dalam format.
`mongodb://mongoHost:mongoPort/mongoDBname`

URL Atlas MongoDB disediakan dalam format. `mongodb+srv://mongoHost:mongoPort/mongoDBname`

Menyediakan database default untuk koneksi, *MongoDBName* adalah opsional.

- Jika Anda memilih untuk membuat rahasia Secrets Manager, pilih jenis AWS Secrets Manager Credential.

  Kemudian, di AWSSecret berikan *secretName*.

- *Jika Anda memilih untuk memberikan **Nama Pengguna dan kata sandi**, berikan MongoDbuser dan MongoDBPass.*

3. Dalam situasi berikut, Anda mungkin memerlukan konfigurasi tambahan:

   - Untuk instance MongoDB yang dihosting di VPC Amazon AWS
     - Anda harus memberikan informasi koneksi Amazon VPC ke koneksi AWS Glue yang menentukan kredenal keamanan MongoDB Anda. Saat membuat atau memperbarui koneksi Anda, atur grup VPC, Subnet, dan Keamanan dalam opsi Jaringan.

Setelah membuat koneksi AWS Glue MongoDB, Anda harus melakukan tindakan berikut sebelum memanggil metode koneksi Anda:

- Jika Anda memilih untuk membuat rahasia Secrets Manager, berikan peran IAM yang terkait dengan izin pekerjaan AWS Glue Anda untuk membaca *secretName*.
- Dalam konfigurasi pekerjaan AWS Glue Anda, berikan *ConnectionName* sebagai koneksi jaringan Tambahan.

Untuk menggunakan koneksi AWS Glue MongoDB Anda di AWS Glue for Spark, berikan opsi dalam panggilan `connectionName` metode koneksi Anda. Atau, Anda dapat mengikuti langkah-langkah the section called “Melakukan integrasi dengan MongoDB” untuk menggunakan koneksi bersama dengan Katalog Data AWS Glue.

Membaca dari MongoDB menggunakan koneksi Glue AWS

Prasyarat:

- Koleksi MongoDB yang ingin Anda baca. Anda akan memerlukan informasi identifikasi untuk koleksi.

  Koleksi MongoDB diidentifikasi oleh nama database dan nama koleksi, MongoDBName, MongoDBCollection.

- Koneksi AWS Glue MongoDB dikonfigurasi untuk memberikan informasi autentikasi. Selesaikan langkah-langkah dalam prosedur sebelumnya, Untuk mengkonfigurasi koneksi ke MongoDB untuk mengkonfigurasi informasi autentikasi Anda. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.

Misalnya:

```
mongodb_read = glueContext.create_dynamic_frame.from_options(
    connection_type="mongodb",
    connection_options={
        "connectionName": "connectionName",
        "database": "mongodbName",
        "collection": "mongodbCollection",
        "partitioner":
 "com.mongodb.spark.sql.connector.read.partitioner.SinglePartitionPartitioner",
        "partitionerOptions.partitionSizeMB": "10",
        "partitionerOptions.partitionKey": "_id",
        "disableUpdateUri": "false",
    }
)
```

Menulis ke tabel MongoDB

Contoh ini menulis informasi dari *DynamicFrame* yang sudah ada DynamicFrame ke MongoDB.

Prasyarat:

- Koleksi MongoDB yang ingin Anda tulis. Anda akan memerlukan informasi identifikasi untuk koleksi.

  Koleksi MongoDB diidentifikasi oleh nama database dan nama koleksi, MongoDBName, MongoDBCollection.

- Koneksi AWS Glue MongoDB dikonfigurasi untuk memberikan informasi autentikasi. Selesaikan langkah-langkah dalam prosedur sebelumnya, Untuk mengkonfigurasi koneksi ke MongoDB untuk

mengkonfigurasi informasi autentikasi Anda. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.

Misalnya:

```
glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="mongodb",
    connection_options={
        "connectionName": "connectionName",
        "database": "mongodbName",
        "collection": "mongodbCollection",
        "disableUpdateUri": "false",
        "retryWrites": "false",
    },
)
```

Membaca dan menulis ke tabel MongoDB

Contoh ini menulis informasi dari *DynamicFrame* yang sudah ada DynamicFrame ke MongoDB.

Prasyarat:

- Koleksi MongoDB yang ingin Anda baca. Anda akan memerlukan informasi identifikasi untuk koleksi.

  Koleksi MongoDB yang ingin Anda tulis. Anda akan memerlukan informasi identifikasi untuk koleksi.

  Koleksi MongoDB diidentifikasi oleh nama database dan nama koleksi, MongoDBName, MongoDBCollection.
- *Informasi autentikasi MongoDB, MongoDbuser dan MongoDBPassword.*

Misalnya:

Python

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext, SparkConf
```

```
from awsglue.context import GlueContext
from awsglue.job import Job
import time

## @params: [JOB_NAME]
args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

job = Job(glueContext)
job.init(args['JOB_NAME'], args)

output_path = "s3://some_bucket/output/" + str(time.time()) + "/"
mongo_uri = "mongodb://<mongo-instanced-ip-address>:27017"
mongo_ssl_uri = "mongodb://<mongo-instanced-ip-address>:27017"
write_uri = "mongodb://<mongo-instanced-ip-address>:27017"

read_mongo_options = {
    "uri": mongo_uri,
    "database": "mongodbName",
    "collection": "mongodbCollection",
    "username": "mongodbUsername",
    "password": "mongodbPassword",
    "partitioner": "MongoSamplePartitioner",
    "partitionerOptions.partitionSizeMB": "10",
    "partitionerOptions.partitionKey": "_id"}

ssl_mongo_options = {
    "uri": mongo_ssl_uri,
    "database": "mongodbName",
    "collection": "mongodbCollection",
    "ssl": "true",
    "ssl.domain_match": "false"
}

write_mongo_options = {
    "uri": write_uri,
    "database": "mongodbName",
    "collection": "mongodbCollection",
    "username": "mongodbUsername",
    "password": "mongodbPassword",
}
```

```
# Get DynamicFrame from MongoDB
dynamic_frame =
 glueContext.create_dynamic_frame.from_options(connection_type="mongodb",

 connection_options=read_mongo_options)

# Write DynamicFrame to MongoDB
glueContext.write_dynamic_frame.from_options(dynamicFrame,
 connection_type="mongodb", connection_options=write_mongo_options)

job.commit()
```

Scala

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.DynamicFrame
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
  val DEFAULT_URI: String = "mongodb://<mongo-instanced-ip-address>:27017"
  val WRITE_URI: String = "mongodb://<mongo-instanced-ip-address>:27017"
  lazy val defaultJsonOption = jsonOptions(DEFAULT_URI)
  lazy val writeJsonOption = jsonOptions(WRITE_URI)
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)

    // Get DynamicFrame from MongoDB
    val dynamicFrame: DynamicFrame = glueContext.getSource("mongodb",
 defaultJsonOption).getDynamicFrame()

    // Write DynamicFrame to MongoDB
    glueContext.getSink("mongodb", writeJsonOption).writeDynamicFrame(dynamicFrame)
```

```
    Job.commit()
  }

  private def jsonOptions(uri: String): JsonOptions = {
    new JsonOptions(
      s"""{"uri": "${uri}",
         |"database":"mongodbName",
         |"collection":"mongodbCollection",
         |"username": "mongodbUsername",
         |"password": "mongodbPassword",
         |"ssl":"true",
         |"ssl.domain_match":"false",
         |"partitioner": "MongoSamplePartitioner",
         |"partitionerOptions.partitionSizeMB": "10",
         |"partitionerOptions.partitionKey": "_id"}""".stripMargin)
  }
}
```

Referensi opsi koneksi MongoDB

Mengkhususkan koneksi ke MongoDB. Pilihan koneksi berbeda untuk koneksi sumber dan koneksi sink.

Properti koneksi ini dibagi antara koneksi sumber dan sink:

- `connectionName`— Digunakan untuk Membaca/Menulis. Nama koneksi AWS Glue MongoDB dikonfigurasi untuk memberikan informasi auth dan jaringan ke metode koneksi Anda. Ketika koneksi AWS Glue dikonfigurasi seperti yang dijelaskan di bagian sebelumnyathe section called “Mengkonfigurasi MongoDB”,, menyediakan `connectionName` akan menggantikan kebutuhan untuk menyediakan`"uri"`, `"username"` dan opsi `"password"` koneksi.
- `"uri"`: (Wajib) Host MongoDB tempat untuk membaca, diformat sebagai `mongodb://<host>:<port>`. Digunakan dalam versi AWS Glue sebelum AWS Glue 4.0.
- `"connection.uri"`: (Wajib) Host MongoDB tempat untuk membaca, diformat sebagai `mongodb://<host>:<port>`. Digunakan dalam AWS Glue 4.0 dan versi yang lebih baru.
- `"username"`: (Wajib) Nama pengguna MongoDB.
- `"password"`: (Wajib) Kata sandi MongoDB.
- `"database"`: (Wajib) Basis data MongoDB untuk dibaca. Pilihan ini juga bisa diberikan di `additional_options` ketika memanggil `glue_context.create_dynamic_frame_from_catalog` dalam skrip tugas Anda.

- `"collection"`: (Wajib) Kumpulan MongoDB untuk dibaca. Pilihan ini juga bisa diberikan di `additional_options` ketika memanggil `glue_context.create_dynamic_frame_from_catalog` dalam skrip tugas Anda.

“ConnectionType”: “mongodb” sebagai sumber

Gunakan opsi koneksi berikut dengan `"connectionType": "mongodb"` sebagai sumber:

- `"ssl"`: (Opsional) Jika `true`, maka memulai koneksi SSL. Defaultnya adalah `false`.
- `"ssl.domain_match"`: (Opsional) Jika `true` dan `ssl` adalah `true`, maka pemeriksaan kecocokan domain dilakukan. Defaultnya adalah `true`.
- `"batchSize"`: (Opsional): Jumlah dokumen yang akan dikembalikan per batch, digunakan dalam kursor batch internal.
- `"partitioner"`: (Opsional): Nama kelas pemartisi untuk membaca input data dari MongoDB. Konektor menyediakan pemartisi berikut:
  - `MongoDefaultPartitioner`(default) (Tidak didukung di AWS Glue 4.0)
  - `MongoSamplePartitioner`(Memerlukan MongoDB 3.2 atau yang lebih baru) (Tidak didukung di AWS Glue 4.0)
  - `MongoShardedPartitioner`(Tidak didukung di AWS Glue 4.0)
  - `MongoSplitVectorPartitioner`(Tidak didukung di AWS Glue 4.0)
  - `MongoPaginateByCountPartitioner`(Tidak didukung di AWS Glue 4.0)
  - `MongoPaginateBySizePartitioner`(Tidak didukung di AWS Glue 4.0)
  - `com.mongodb.spark.sql.connector.read.partitioner.SinglePartitionPartitioner`
  - `com.mongodb.spark.sql.connector.read.partitioner.ShardedPartitioner`
  - `com.mongodb.spark.sql.connector.read.partitioner.PaginateIntoPartitionsPartit`
- `"partitionerOptions"` (Opsional): Opsi untuk pemartisi yang ditunjuk. Opsi berikut didukung untuk setiap pemartisi:
  - `MongoSamplePartitioner`: `partitionKey`, `partitionSizeMB`, `samplesPerPartition`
  - `MongoShardedPartitioner`: `shardkey`
  - `MongoSplitVectorPartitioner`: `partitionKey`, `partitionSizeMB`
  - `MongoPaginateByCountPartitioner`: `partitionKey`, `numberOfPartitions`
  - `MongoPaginateBySizePartitioner`: `partitionKey`, `partitionSizeMB`

Untuk informasi lebih lanjut tentang opsi ini, lihat Konfigurasi Partisi dalam dokumentasi MongoDB.

“ConnectionType”: “mongodb” sebagai wastafel

Gunakan opsi koneksi berikut dengan `"connectionType": "mongodb"` sebagai sink:

- `"ssl"`: (Opsional) Jika `true`, maka memulai koneksi SSL. Defaultnya adalah `false`.
- `"ssl.domain_match"`: (Opsional) Jika `true` dan `ssl` adalah `true`, maka pemeriksaan kecocokan domain dilakukan. Defaultnya adalah `true`.
- `"extendedBsonTypes"`: (Opsional) Jika `true`, memungkinkan jenis BSON yang diperpanjang saat menulis data ke MongoDB. Defaultnya adalah `true`.
- `"replaceDocument"`: (Opsional) Jika `true`, menggantikan seluruh dokumen ketika menyimpan set data yang berisi bidang `_id`. Jika `false`, hanya bidang dalam dokumen yang cocok dengan bidang dalam set data saja yang diperbarui. Defaultnya adalah `true`.
- `"maxBatchSize"`: (Opsional): Ukuran batch maksimum untuk operasi massal saat menyimpan data. Default-nya adalah 512.
- `"retryWrites"`: (Opsional): Secara otomatis mencoba kembali operasi penulisan tertentu satu kali jika AWS Glue menemukan kesalahan jaringan.

Koneksi SAP HANA

Anda dapat menggunakan AWS Glue for Spark untuk membaca dari dan menulis ke tabel di SAP HANA di AWS Glue 4.0 dan versi yang lebih baru. Anda dapat menentukan apa yang harus dibaca dari SAP HANA dengan query SQL. Anda terhubung ke SAP HANA menggunakan kredensyal JDBC yang disimpan melalui AWS Secrets Manager koneksi Glue AWS SAP HANA.

Untuk informasi lebih lanjut tentang SAP HANA JDBC, lihat dokumentasi SAP HANA.

Mengkonfigurasi koneksi SAP HANA

Untuk terhubung ke SAP HANA dari AWS Glue, Anda harus membuat dan menyimpan kredensi SAP HANA Anda secara AWS Secrets Manager rahasia, kemudian mengaitkan rahasia itu dengan koneksi SAP HANA Glue. AWS Anda perlu mengonfigurasi konektivitas jaringan antara layanan SAP HANA dan AWS Glue.

Untuk terhubung ke SAP HANA, Anda mungkin memerlukan beberapa prasyarat:

- Jika layanan SAP HANA Anda berada dalam VPC Amazon, konfigurasikan Amazon VPC untuk memungkinkan pekerjaan AWS Glue Anda berkomunikasi dengan layanan SAP HANA tanpa lalu lintas melintasi internet publik.

  Di Amazon VPC, identifikasi atau buat grup VPC, Subnet, dan Keamanan yang akan digunakan AWS Glue saat menjalankan pekerjaan. Selain itu, Anda perlu memastikan Amazon VPC dikonfigurasi untuk mengizinkan lalu lintas jaringan antara titik akhir SAP HANA Anda dan lokasi ini. Pekerjaan Anda perlu membuat koneksi TCP dengan port SAP HANA JDBC Anda. Untuk informasi selengkapnya tentang port SAP HANA, lihat dokumentasi SAP HANA. Berdasarkan tata letak jaringan Anda, ini mungkin memerlukan perubahan pada aturan grup keamanan, ACL Jaringan, Gateway NAT, dan koneksi Peering.
- Tidak ada prasyarat tambahan jika titik akhir SAP HANA Anda dapat diakses melalui internet.

Untuk mengkonfigurasi koneksi ke SAP HANA:

1. DiAWS Secrets Manager, buat rahasia menggunakan kredensyal SAP HANA Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci user dengan nilai saphanaUsername.*
   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci password dengan nilai saphanaPassword.*
2. Di konsol AWS Glue, buat koneksi dengan mengikuti langkah-langkah dithe section called “Menambahkan AWS Glue koneksi”. Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk penggunaan masa depan di GlueAWS.

   - Saat memilih jenis Koneksi, pilih SAP HANA.
   - Saat memberikan URL SAP HANA, berikan URL untuk instance Anda.

     URL SAP HANA JDBC ada dalam bentuk `jdbc:sap://`*saphanaHostname*`:`*saphanaPort*`/?`*databaseName*`=`*saphanaDBname*`,`*Paramete*

     AWSGlue membutuhkan parameter URL JDBC berikut:

     - `databaseName`— Database default di SAP HANA untuk terhubung ke.
   - Saat memilih AWSSecret, berikan *secretName*.

Setelah membuat koneksi AWS Glue SAP HANA, Anda harus melakukan langkah-langkah berikut sebelum menjalankan pekerjaan AWS Glue Anda:

- Berikan peran IAM yang terkait dengan izin pekerjaan AWS Glue Anda untuk membaca *secretName*.
- Dalam konfigurasi pekerjaan AWS Glue Anda, berikan *ConnectionName* sebagai koneksi jaringan Tambahan.

Membaca dari tabel SAP HANA

Prasyarat:

- Meja SAP HANA yang ingin Anda baca. Anda akan memerlukan informasi identifikasi untuk tabel.

  Sebuah tabel dapat ditentukan dengan nama tabel SAP HANA dan nama skema, dalam formulir. *schemaName*.*tableName* Nama skema dan pemisah "." tidak diperlukan jika tabel dalam skema default, "publik". Panggil *TableIdentifier* ini. Perhatikan bahwa database disediakan sebagai parameter URL JDBC di. `connectionName`
- Koneksi AWS Glue SAP HANA yang dikonfigurasi untuk memberikan informasi autentikasi. Selesaikan langkah-langkah dalam prosedur sebelumnya, Untuk mengkonfigurasi koneksi ke SAP HANA untuk mengkonfigurasi informasi autentikasi Anda. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.

Misalnya:

```
saphana_read_table = glueContext.create_dynamic_frame.from_options(
    connection_type="saphana",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableIdentifier",
    }
)
```

Anda juga dapat memberikan kueri SELECT SQL, untuk memfilter hasil yang dikembalikan ke Anda DynamicFrame. Anda perlu mengkonfigurasi`query`.

Misalnya:

```
saphana_read_query = glueContext.create_dynamic_frame.from_options(
```

```
    connection_type="saphana",
    connection_options={
        "connectionName": "connectionName",
        "query": "query"
    }
)
```

Menulis ke tabel SAP HANA

Contoh ini menulis informasi dari *DynamicFrame yang sudah ada DynamicFrame ke SAP* HANA. Jika tabel sudah memiliki informasi, AWS Glue akan error.

Prasyarat:

- Meja SAP HANA yang ingin Anda tulis.

  Sebuah tabel dapat ditentukan dengan nama tabel SAP HANA dan nama skema, dalam formulir. *schemaName*.*tableName* Nama skema dan pemisah “.” tidak diperlukan jika tabel dalam skema default, “publik”. Panggil *TableIdentifier* ini. Perhatikan bahwa database disediakan sebagai parameter URL JDBC di. connectionName
- Informasi autentikasi SAP HANA. Selesaikan langkah-langkah dalam prosedur sebelumnya, Untuk mengkonfigurasi koneksi ke SAP HANA untuk mengkonfigurasi informasi autentikasi Anda. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.

Misalnya:

```
options = {
    "connectionName": "connectionName",
    "dbtable": 'tableIdentifier'
}

    saphana_write = glueContext.write_dynamic_frame.from_options(
        frame=dynamicFrame,
        connection_type="saphana",
        connection_options=options
)
```

Referensi opsi koneksi SAP HANA

- connectionName — Diperlukan. Digunakan untuk Baca/tulis. Nama koneksi AWS Glue SAP HANA dikonfigurasi untuk memberikan informasi auth dan jaringan ke metode koneksi Anda.

- `databaseName`— Digunakan untuk Membaca/Menulis. Nilai Valid: nama database di SAP HANA. Nama database untuk connect.
- `dbtable`— Diperlukan untuk menulis, diperlukan untuk membaca kecuali `query` disediakan. Digunakan untuk Baca/tulis. Nilai Valid: isi dari klausa SAP HANA SQL FROM. Mengidentifikasi tabel di SAP HANA untuk terhubung ke. Anda juga dapat memberikan SQL lain selain nama tabel, seperti subquery. Untuk informasi lebih lanjut, lihat klausa Dari dalam dokumentasi SAP HANA.
- `query`— Digunakan untuk Baca. Kueri SAP HANA SQL SELECT yang mendefinisikan apa yang harus diambil saat membaca dari SAP HANA.

Koneksi kepingan salju

Anda dapat menggunakan AWS Glue for Spark untuk membaca dan menulis ke tabel di Snowflake di AWS Glue 4.0 dan versi yang lebih baru. Anda dapat membaca dari Snowflake dengan query SQL. Anda dapat terhubung ke Snowflake menggunakan pengguna dan kata sandi. Anda dapat merujuk ke kredensil Snowflake yang disimpan melalui AWS Secrets Manager Katalog Data GlueAWS. Katalog Data Kredensil Kepingan Salju untuk AWS Glue for Spark disimpan secara terpisah dari kredenal Kepingan Salju Katalog Data untuk perayap. Anda harus memilih `SNOWFLAKE` jenis koneksi dan bukan jenis koneksi yang `JDBC` dikonfigurasi untuk terhubung ke Snowflake.

Untuk informasi lebih lanjut tentang Snowflake, lihat situs web Snowflake. Untuk informasi selengkapnya tentang SnowflakeAWS, lihat Snowflake Data Warehouse di Amazon Web Services.

Mengkonfigurasi koneksi Snowflake

Tidak ada AWS prasyarat untuk menghubungkan ke database Snowflake yang tersedia melalui internet.

Secara opsional, Anda dapat melakukan konfigurasi berikut untuk mengelola kredensil koneksi Anda dengan Glue. AWS

Untuk mengelola kredensi koneksi Anda dengan Glue AWS

1. *Di Snowflake, buat pengguna, SnowFlakeUser dan kata sandi, SnowFlakePassword.*
2. DiAWS Secrets Manager, buat rahasia menggunakan kredensi Snowflake Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

- Saat memilih pasangan kunci/nilai, buat pasangan untuk *SnowFlakeUser* dengan kuncinya. `sfUser`
- Saat memilih pasangan kunci/nilai, buat pasangan untuk *SnowFlakePassword* dengan kuncinya. `sfPassword`
- Saat memilih pasangan kunci/nilai, Anda dapat memberikan kunci kepada gudang Snowflake Anda. `sfWarehouse`

3. Dalam AWS Glue Data Catalog, buat koneksi dengan mengikuti langkah-langkah dithe section called “Menambahkan AWS Glue koneksi”. Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk langkah selanjutnya.

   - Saat memilih jenis Koneksi, pilih Snowflake.
   - Saat memilih URL Snowflake, berikan URL instance Snowflake Anda. URL akan menggunakan nama host dalam formulir*account_identifier*`.snowflakecomputing.com`.
   - Saat memilih AWSSecret, berikan *secretName*.

4. Dalam konfigurasi pekerjaan AWS Glue Anda, berikan *ConnectionName* sebagai koneksi jaringan Tambahan.

Dalam situasi berikut, Anda mungkin memerlukan yang berikut:

- Untuk Snowflake yang dihosting AWS di VPC Amazon
  - Anda akan memerlukan konfigurasi VPC Amazon yang sesuai untuk Snowflake. Untuk informasi selengkapnya tentang cara mengonfigurasi VPC Amazon Anda, lihat AWS PrivateLink& Snowflake di dokumentasi Snowflake.
  - Anda akan memerlukan konfigurasi Amazon VPC yang sesuai untuk GlueAWS. the section called “Titik akhir VPC (AWS PrivateLink)”.
  - Anda perlu membuat koneksi AWS Glue Data Catalog yang menyediakan informasi koneksi Amazon VPC (selain id AWS Secrets Manager rahasia yang mendefinisikan kredenal keamanan Snowflake Anda). URL Anda akan berubah saat menggunakanAWS PrivateLink, seperti yang dijelaskan dalam dokumentasi Snowflake yang ditautkan di item sebelumnya.
  - Anda akan memerlukan konfigurasi pekerjaan Anda termasuk koneksi Katalog Data sebagai koneksi jaringan tambahan.

Membaca dari tabel Snowflake

Prasyarat: Meja Kepingan Salju yang ingin Anda baca. *Anda akan membutuhkan nama tabel Snowflake, TableName. Anda akan memerlukan url Snowflake Anda SnowFlakeUrl, nama pengguna SnowFlakeUser dan kata sandi SnowFlakePassword. Jika pengguna Snowflake Anda tidak memiliki set namespace default, Anda akan memerlukan nama database Snowflake, DatabaseName dan nama skema SchemaName. Selain itu, jika pengguna Snowflake Anda tidak memiliki set gudang default, Anda akan memerlukan nama gudang WarehouseName.*

Sebagai contoh:

Prasyarat Tambahan: *Selesaikan langkah-langkah Untuk mengelola kredensil koneksi Anda dengan AWS Glue untuk mengkonfigurasi SnowFlakeUrl, SnowFlakeUsername dan SnowFlakePassword.* Untuk meninjau langkah-langkah ini, lihatthe section called “Mengkonfigurasi Snowflake”, bagian sebelumnya. Untuk memilih yang mana Koneksi jaringan tambahan untuk terhubung, kami akan menggunakan `connectionName` parameter.

```
snowflake_read = glueContext.create_dynamic_frame.from_options(
  connection_type="snowflake",
  connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName",
        "sfDatabase": "databaseName",
        "sfSchema": "schemaName",
        "sfWarehouse": "warehouseName",
    }
)
```

Selain itu, Anda dapat menggunakan `query` parameter `autopushdown` dan untuk membaca sebagian dari tabel Snowflake. Ini bisa jauh lebih efisien daripada memfilter hasil Anda setelah dimuat ke Spark. Pertimbangkan contoh di mana semua penjualan disimpan dalam tabel yang sama, tetapi Anda hanya perlu menganalisis penjualan dari toko tertentu pada hari libur. Jika informasi tersebut disimpan dalam tabel, Anda dapat menggunakan predikat pushdown untuk mengambil hasil sebagai berikut:

```
snowflake_node = glueContext.create_dynamic_frame.from_options(
    connection_type="snowflake",
    connection_options={
        "autopushdown": "on",
```

```
        "query": "select * from sales where store='1' and IsHoliday='TRUE'",
        "connectionName": "snowflake-glue-conn",
        "sfDatabase": "databaseName",
        "sfSchema": "schemaName",
        "sfWarehouse": "warehouseName",
    }
)
```

Menulis ke tabel Snowflake

Prasyarat: Database Snowflake yang ingin Anda tulis. Anda akan memerlukan nama tabel saat ini atau yang diinginkan, *TableName*. *Anda akan memerlukan url Snowflake Anda SnowFlakeUrl, nama pengguna SnowFlakeUser dan kata sandi SnowFlakePassword. Jika pengguna Snowflake Anda tidak memiliki set namespace default, Anda akan memerlukan nama database Snowflake, DatabaseName dan nama skema SchemaName. Selain itu, jika pengguna Snowflake Anda tidak memiliki set gudang default, Anda akan memerlukan nama gudang WarehouseName.*

Sebagai contoh:

Prasyarat Tambahan: *Selesaikan langkah-langkah Untuk mengelola kredensil koneksi Anda dengan AWS Glue untuk mengkonfigurasi SnowFlakeUrl, SnowFlakeUsername dan SnowFlakePassword.* Untuk meninjau langkah-langkah ini, lihatthe section called “Mengkonfigurasi Snowflake”, bagian sebelumnya. Untuk memilih yang mana Koneksi jaringan tambahan untuk terhubung, kami akan menggunakan `connectionName` parameter.

```
glueContext.write_dynamic_frame.from_options(
    connection_type="snowflake",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName",
        "sfDatabase": "databaseName",
        "sfSchema": "schemaName",
        "sfWarehouse": "warehouseName",
    },
)
```

Referensi opsi koneksi kepingan salju

Jenis koneksi Snowflake mengambil opsi koneksi berikut:

Anda dapat mengambil beberapa parameter di bagian ini dari koneksi Katalog Data (`sfUrl`,,`sfPassword`)`sfUser`, dalam hal ini Anda tidak diharuskan untuk menyediakannya. Anda dapat melakukan ini dengan memberikan parameter`connectionName`.

Anda dapat mengambil beberapa parameter di bagian ini dari AWS Secrets Manager rahasia (`sfUser`,`sfPassword`), dalam hal ini Anda tidak diharuskan untuk menyediakannya. Rahasianya harus menyediakan konten di bawah `sfUser` dan `sfPassword` kunci. Anda dapat melakukan ini dengan memberikan parameter`secretId`.

Parameter berikut digunakan secara umum saat menghubungkan ke Snowflake.

- `sfDatabase`— Diperlukan jika default pengguna tidak diatur di Snowflake. Digunakan untuk Baca/Tulis. Database yang akan digunakan untuk sesi setelah menghubungkan.
- `sfSchema`— Diperlukan jika default pengguna tidak diatur di Snowflake. Digunakan untuk Baca/Tulis. Skema yang digunakan untuk sesi setelah menghubungkan.
- `sfWarehouse`— Diperlukan jika default pengguna tidak diatur di Snowflake. Digunakan untuk Baca/Tulis. Gudang virtual default untuk digunakan untuk sesi setelah menghubungkan.
- `sfRole`— Diperlukan jika default pengguna tidak diatur di Snowflake. Digunakan untuk Baca/Tulis. Peran keamanan default yang akan digunakan untuk sesi setelah menghubungkan.
- `sfUrl`— (Wajib) Digunakan untuk Baca/Tulis. Menentukan nama host untuk akun Anda dalam format berikut:. *`account_identifier`*`.snowflakecomputing.com` Untuk informasi selengkapnya tentang pengenal akun, lihat Pengidentifikasi Akun di dokumentasi Snowflake.
- `sfUser`— (Wajib) Digunakan untuk Baca/Tulis. Nama login untuk pengguna Snowflake.
- `sfPassword`— (Diperlukan kecuali `pem_private_key` disediakan) Digunakan untuk Baca/Tulis. Kata sandi untuk pengguna Snowflake.
- `dbtable`— Diperlukan saat bekerja dengan tabel lengkap. Digunakan untuk Baca/Tulis. Nama tabel yang akan dibaca atau tabel tempat data ditulis. Saat membaca, semua kolom dan catatan diambil.
- `pem_private_key`— Digunakan untuk Membaca/Menulis. String kunci pribadi yang dikodekan b64 yang tidak terenkripsi. Kunci pribadi untuk pengguna Snowflake. Adalah umum untuk menyalin ini dari file PEM. Untuk informasi selengkapnya, lihat Autentikasi pasangan kunci dan rotasi pasangan kunci dalam dokumentasi Snowflake.
- `query`— Diperlukan saat membaca dengan kueri. Digunakan untuk Baca. Kueri (`SELECT`pernyataan) yang tepat untuk dijalankan

Opsi berikut digunakan untuk mengonfigurasi perilaku tertentu selama proses menghubungkan ke Snowflake.

- `preactions`— Digunakan untuk Membaca/Menulis. Nilai Valid: Titik koma dipisahkan daftar pernyataan SQL sebagai String. Pernyataan SQL dijalankan sebelum data ditransfer antara AWS Glue dan Snowflake. Jika pernyataan berisi`%s`, diganti dengan nama tabel direferensikan untuk operasi. `%s`
- `postactions`— Digunakan untuk Membaca/Menulis. Pernyataan SQL dijalankan setelah data ditransfer antara AWS Glue dan Snowflake. Jika pernyataan berisi`%s`, diganti dengan nama tabel direferensikan untuk operasi. `%s`
- `autopushdown`— Default:`"on"`. Nilai yang Valid:`"on"`,`"off"`. Parameter ini mengontrol apakah pushdown kueri otomatis diaktifkan. Jika pushdown diaktifkan, maka ketika kueri dijalankan di Spark, jika bagian dari kueri dapat “didorong ke bawah” ke server Snowflake, itu didorong ke bawah. Ini meningkatkan kinerja beberapa kueri. Untuk informasi tentang apakah kueri Anda dapat ditekan ke bawah, lihat Pushdown di dokumentasi Snowflake.

Selain itu, beberapa opsi yang tersedia pada konektor Snowflake Spark mungkin didukung di Glue. AWS Untuk informasi selengkapnya tentang opsi yang tersedia di konektor Snowflake Spark, lihat Mengatur Opsi Konfigurasi untuk Konektor dalam dokumentasi Snowflake.

Keterbatasan konektor kepingan salju

Menghubungkan ke Snowflake dengan AWS Glue for Spark tunduk pada batasan berikut.

- Konektor ini tidak mendukung bookmark pekerjaan. Untuk informasi selengkapnya tentang bookmark pekerjaan, lihatthe section called “Melacak data yang diproses menggunakan bookmark pekerjaan”.
- Konektor ini tidak mendukung Snowflake membaca dan menulis melalui tabel di Katalog Data AWS Glue menggunakan metode `create_dynamic_frame.from_catalog` dan`write_dynamic_frame.from_catalog`.
- Konektor ini tidak mendukung koneksi ke Snowflake dengan kredensil selain pengguna dan kata sandi.
- Konektor ini tidak didukung dalam pekerjaan streaming.
- Konektor ini mendukung query berbasis `SELECT` pernyataan saat mengambil informasi (seperti dengan `query` parameter). Jenis kueri lainnya (seperti`SHOW`,`DESC`, atau pernyataan DHTML) tidak didukung.

- Snowflake membatasi ukuran teks kueri (yaitu pernyataan SQL) yang dikirimkan melalui klien Snowflake hingga 1 MB per pernyataan. Untuk detail selengkapnya, lihat Batas Ukuran Teks Kueri.

Koneksi Teradata Vantage

Anda dapat menggunakan AWS Glue for Spark untuk membaca dan menulis ke tabel yang ada di Teradata Vantage di AWS Glue 4.0 dan versi yang lebih baru. Anda dapat menentukan apa yang harus dibaca dari Teradata dengan query SQL. Anda dapat terhubung ke Teradata menggunakan kredenal nama pengguna dan kata sandi yang disimpan melalui AWS Secrets Manager koneksi Glue. AWS

Untuk informasi lebih lanjut tentang Teradata, lihat dokumentasi Teradata

Mengkonfigurasi koneksi Teradata

Untuk terhubung ke Teradata dari AWS Glue, Anda harus membuat dan menyimpan kredenal Teradata Anda secara AWS Secrets Manager rahasia, lalu mengaitkan rahasia itu dengan koneksi Glue Teradata. AWS Jika instans Teradata Anda ada di VPC Amazon, Anda juga perlu memberikan opsi jaringan ke koneksi AWS Glue Teradata Anda.

Untuk terhubung ke Teradata dari AWS Glue, Anda mungkin memerlukan beberapa prasyarat:

- Jika Anda mengakses lingkungan Teradata Anda melalui Amazon VPC, konfigurasikan Amazon VPC untuk memungkinkan pekerjaan AWS Glue Anda berkomunikasi dengan lingkungan Teradata. Kami tidak menyarankan mengakses lingkungan Teradata melalui internet publik.

  Di Amazon VPC, identifikasi atau buat grup VPC, Subnet, dan Keamanan yang akan digunakan AWS Glue saat menjalankan pekerjaan. Selain itu, Anda perlu memastikan Amazon VPC dikonfigurasi untuk mengizinkan lalu lintas jaringan antara instans Teradata Anda dan lokasi ini. Pekerjaan Anda perlu membuat koneksi TCP dengan port klien Teradata Anda. Untuk informasi selengkapnya tentang port Teradata, lihat dokumentasi Teradata.

  Berdasarkan tata letak jaringan Anda, konektivitas VPC yang aman mungkin memerlukan perubahan di Amazon VPC dan layanan jaringan lainnya. Untuk informasi lebih lanjut tentang AWS konektivitas, lihat Opsi AWS Konektivitas di dokumentasi Teradata.

Untuk mengkonfigurasi koneksi AWS Glue Teradata:

1. *Dalam konfigurasi Teradata Anda, identifikasi atau buat pengguna dan kata sandi AWS Glue akan terhubung dengan, Teradatauser dan TeradataPassWord.* Untuk informasi lebih lanjut, lihat Ikhtisar Keamanan Vantage di dokumentasi Teradata.
2. Di AWS Secrets Manager, buat rahasia menggunakan kredenal Teradata Anda. Untuk membuat rahasia di Secrets Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci user dengan nilai TeradataUsername.*
   - *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci password dengan nilai TeradataPassword.*
3. Di konsol AWS Glue, buat koneksi dengan mengikuti langkah-langkah dithe section called “Menambahkan AWS Glue koneksi”. Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk langkah selanjutnya.

   - Saat memilih jenis Koneksi, pilih Teradata.
   - Saat memberikan URL JDBC, berikan URL untuk instance Anda. Anda juga dapat membuat hardcode parameter koneksi yang dipisahkan koma tertentu di URL JDBC Anda. URL harus sesuai dengan format berikut: `jdbc:teradata://`*teradataHostname*`/`*ParameterName*`=`*ParameterValue*`,`*ParameterName*

     Parameter URL yang didukung meliputi:

     - `DATABASE`— nama database pada host untuk mengakses secara default.
     - `DBS_PORT`— port database, digunakan saat berjalan pada port yang tidak standar.
   - Saat memilih jenis Credential, pilih AWS Secrets Manager, lalu atur AWS Secret ke *secretName*.
4. Dalam situasi berikut, Anda mungkin memerlukan konfigurasi tambahan:

   - Untuk instans Teradata yang dihosting di VPC AWS Amazon
     - Anda harus memberikan informasi koneksi Amazon VPC ke koneksi AWS Glue yang menentukan kredenal keamanan Teradata Anda. Saat membuat atau memperbarui koneksi Anda, atur grup VPC, Subnet, dan Keamanan dalam opsi Jaringan.

Setelah membuat koneksi AWS Glue Teradata, Anda harus melakukan langkah-langkah berikut sebelum memanggil metode koneksi Anda.

- Berikan peran IAM yang terkait dengan izin pekerjaan AWS Glue Anda untuk membaca *secretName*.
- Dalam konfigurasi pekerjaan AWS Glue Anda, berikan *ConnectionName* sebagai koneksi jaringan Tambahan.

Membaca dari Teradata

Prasyarat:

- Tabel Teradata yang ingin Anda baca. Anda akan membutuhkan nama tabel, *TableName*.
- Koneksi AWS Glue Teradata yang dikonfigurasi untuk memberikan informasi autentikasi. Selesaikan langkah-langkahnya Untuk mengonfigurasi koneksi ke Teradata untuk mengonfigurasi informasi autentikasi Anda. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.

Sebagai contoh:

```
teradata_read_table = glueContext.create_dynamic_frame.from_options(
    connection_type="teradata",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName"
    }
)
```

Anda juga dapat memberikan kueri SELECT SQL, untuk memfilter hasil yang dikembalikan ke Anda DynamicFrame. Anda perlu mengkonfigurasi`query`.

Sebagai contoh:

```
teradata_read_query = glueContext.create_dynamic_frame.from_options(
    connection_type="teradata",
    connection_options={
        "connectionName": "connectionName",
        "query": "query"
    }
)
```

Menulis ke tabel Teradata

Prasyarat: *Tabel Teradata yang ingin Anda tulis, TableName.* Anda harus membuat tabel sebelum memanggil metode koneksi.

Sebagai contoh:

```
teradata_write = glueContext.write_dynamic_frame.from_options(
    connection_type="teradata",
    connection_options={
        "connectionName": "connectionName",
        "dbtable": "tableName"
    }
)
```

Referensi opsi koneksi Teradata

- `connectionName` — Diperlukan. Digunakan untuk Baca/Tulis. Nama koneksi AWS Glue Teradata yang dikonfigurasi untuk memberikan informasi autentikasi dan jaringan ke metode koneksi Anda.
- `dbtable`— Diperlukan untuk menulis, diperlukan untuk membaca kecuali `query` disediakan. Digunakan untuk Baca/Tulis. Nama tabel metode koneksi Anda akan berinteraksi dengan.
- `query`— Digunakan untuk Baca. Kueri SELECT SQL yang mendefinisikan apa yang harus diambil saat membaca dari Teradata.

Koneksi Vertica

Anda dapat menggunakan AWS Glue for Spark untuk membaca dan menulis ke tabel di Vertica di AWS Glue 4.0 dan versi yang lebih baru. Anda dapat menentukan apa yang harus dibaca dari Vertica dengan query SQL. Anda terhubung ke Vertica menggunakan kredensyal nama pengguna dan kata sandi yang disimpan melalui koneksi GlueAWS. AWS Secrets Manager

Untuk informasi lebih lanjut tentang Vertica, lihat dokumentasi Vertica.

Mengkonfigurasi koneksi Vertica

Untuk terhubung ke Vertica dari AWS Glue, Anda harus membuat dan menyimpan kredensyal Vertica Anda secara AWS Secrets Manager rahasia, lalu mengaitkan rahasia itu dengan koneksi Vertica Glue. AWS Jika instans Vertica Anda ada di VPC Amazon, Anda juga perlu memberikan opsi jaringan ke koneksi AWS Glue Vertica Anda. Anda memerlukan bucket atau folder Amazon S3 untuk digunakan untuk penyimpanan sementara saat membaca dari dan menulis ke database.

Untuk terhubung ke Vertica dari AWS Glue, Anda memerlukan beberapa prasyarat:

- *Bucket atau folder Amazon S3 yang akan digunakan untuk penyimpanan sementara saat membaca dari dan menulis ke database, dirujuk oleh Temps3Path.*

  > **Note**
  >
  > *Saat menggunakan Vertica dalam pratinjau data pekerjaan AWS Glue, file sementara mungkin tidak dihapus secara otomatis dari Temps3Path.* Untuk memastikan penghapusan file sementara, langsung akhiri sesi pratinjau data dengan memilih Akhiri sesi di panel pratinjau data.
  > Jika Anda tidak dapat menjamin sesi pratinjau data berakhir secara langsung, pertimbangkan untuk menyetel konfigurasi Siklus Hidup Amazon S3 untuk menghapus data lama. Sebaiknya hapus data yang lebih lama dari 49 jam, berdasarkan runtime pekerjaan maksimum ditambah margin. Untuk informasi selengkapnya tentang mengonfigurasi Siklus Hidup Amazon S3, lihat Mengelola siklus hidup penyimpanan di dokumentasi Amazon S3.

- Kebijakan IAM dengan izin yang sesuai ke jalur Amazon S3 yang dapat Anda kaitkan dengan peran pekerjaan AWS Glue Anda.
- Jika instans Vertica Anda ada di VPC Amazon, konfigurasikan Amazon VPC untuk memungkinkan pekerjaan AWS Glue Anda berkomunikasi dengan instans Vertica tanpa lalu lintas melintasi internet publik.

  Di Amazon VPC, identifikasi atau buat grup VPC, Subnet, dan Keamanan yang akan digunakan AWS Glue saat menjalankan pekerjaan. Selain itu, Anda perlu memastikan Amazon VPC dikonfigurasi untuk mengizinkan lalu lintas jaringan antara instans Vertica Anda dan lokasi ini. Pekerjaan Anda perlu membuat koneksi TCP dengan port klien Vertica Anda, (default 5433). Berdasarkan tata letak jaringan Anda, ini mungkin memerlukan perubahan pada aturan grup keamanan, ACL Jaringan, Gateway NAT, dan koneksi Peering.

Anda kemudian dapat melanjutkan untuk mengkonfigurasi AWS Glue untuk digunakan dengan Vertica.

Untuk mengonfigurasi koneksi ke Vertica:

1. *DiAWS Secrets Manager, buat rahasia menggunakan kredensyal Vertica Anda, VerticaUsername dan VerticaPassWord.* Untuk membuat rahasia di Secrets

Manager, ikuti tutorial yang tersedia di Buat AWS Secrets Manager rahasia dalam AWS Secrets Manager dokumentasi. Setelah membuat rahasia, simpan nama Rahasia, *secretName* untuk langkah selanjutnya.

- *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci user dengan nilai VerticaUsername.*
- *Saat memilih pasangan **kunci/nilai, buat pasangan** untuk kunci password dengan nilai VerticaPassWord.*

2. Di konsol AWS Glue, buat koneksi dengan mengikuti langkah-langkah dithe section called "Menambahkan AWS Glue koneksi". Setelah membuat koneksi, simpan nama koneksi, *connectionName*, untuk langkah selanjutnya.

   - Saat memilih jenis Koneksi, pilih Vertica.
   - Saat memilih Vertica Host, berikan nama host instalasi Vertica Anda.
   - Saat memilih Port Vertica, port instalasi Vertica Anda tersedia.
   - Saat memilih AWSSecret, berikan *secretName*.

3. Dalam situasi berikut, Anda mungkin memerlukan konfigurasi tambahan:

   - Untuk instance Vertica yang dihosting AWS di VPC Amazon
     - Berikan informasi koneksi Amazon VPC ke koneksi AWS Glue yang menentukan kredensyal keamanan Vertica Anda. Saat membuat atau memperbarui koneksi Anda, atur grup VPC, Subnet, dan Keamanan dalam opsi Jaringan.

Setelah membuat koneksi AWS Glue Vertica, Anda perlu melakukan langkah-langkah berikut sebelum memanggil metode koneksi Anda.

- *Berikan peran IAM yang terkait dengan izin pekerjaan AWS Glue Anda ke Temps3Path.*
- Berikan peran IAM yang terkait dengan izin pekerjaan AWS Glue Anda untuk membaca *secretName*.
- Dalam konfigurasi pekerjaan AWS Glue Anda, berikan *ConnectionName* sebagai koneksi jaringan Tambahan.

Membaca dari Vertica

Prasyarat:

- Tabel Vertica yang ingin Anda baca. *Anda akan memerlukan nama database Vertica, dbName dan nama tabel, TableName.*
- Koneksi AWS Glue Vertica dikonfigurasi untuk memberikan informasi autentikasi. Selesaikan langkah-langkah dalam prosedur sebelumnya, Untuk mengkonfigurasi koneksi ke Vertica untuk mengkonfigurasi informasi autentikasi Anda. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.
- Bucket atau folder Amazon S3 yang akan digunakan untuk penyimpanan sementara, yang disebutkan sebelumnya. Anda akan membutuhkan nama, *Temps3Path*. Anda harus terhubung ke lokasi ini menggunakan `s3a` protokol.

Misalnya:

```
dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="vertica",
    connection_options={
        "connectionName": "connectionName",
        "staging_fs_url": "s3a://tempS3Path",
        "db": "dbName",
        "table": "tableName",
    }
)
```

Anda juga dapat memberikan kueri SELECT SQL, untuk memfilter hasil yang dikembalikan ke Anda DynamicFrame atau untuk mengakses kumpulan data dari beberapa tabel.

Misalnya:

```
dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="vertica",
    connection_options={
        "connectionName": "connectionName",
        "staging_fs_url": "s3a://tempS3Path",
        "db": "dbName",
        "query": "select * FROM tableName",
    },
)
```

Menulis ke tabel Vertica

Contoh ini menulis informasi dari *DynamicFrame* ke DynamicFrame Vertica yang sudah ada. Jika tabel sudah memiliki informasi, AWS Glue akan menambahkan data dari Anda DynamicFrame.

Prasyarat:

- Nama tabel saat ini atau yang diinginkan, *TableName*, yang ingin Anda tulis. Anda juga akan memerlukan nama database Vertica yang sesuai, *DBName*.
- Koneksi AWS Glue Vertica dikonfigurasi untuk memberikan informasi autentikasi. Selesaikan langkah-langkah dalam prosedur sebelumnya, Untuk mengkonfigurasi koneksi ke Vertica untuk mengkonfigurasi informasi autentikasi Anda. Anda akan memerlukan nama koneksi AWS Glue, *ConnectionName*.
- Bucket atau folder Amazon S3 yang akan digunakan untuk penyimpanan sementara, yang disebutkan sebelumnya. Anda akan membutuhkan nama, *Temps3Path*. Anda harus terhubung ke lokasi ini menggunakan `s3a` protokol.

Misalnya:

```
glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="vertica",
    connection_options={
        "connectionName": "connectionName",
        "staging_fs_url": "s3a://tempS3Path",
        "db": "dbName",
        "table": "tableName",
    }
)
```

Referensi opsi koneksi Vertica

- `connectionName` — Diperlukan. Digunakan untuk Baca/Tulis. Nama koneksi AWS Glue Vertica dikonfigurasi untuk memberikan informasi autentikasi dan jaringan ke metode koneksi Anda.
- db — Diperlukan. Digunakan untuk Baca/Tulis. Nama database di Vertica metode koneksi Anda akan berinteraksi dengan.
- `dbSchema`— Diperlukan jika diperlukan untuk mengidentifikasi tabel Anda. Digunakan untuk Baca/Tulis. Default: `public`. Nama skema metode koneksi Anda akan berinteraksi dengan.

- `table`— Diperlukan untuk menulis, diperlukan untuk membaca kecuali `query` disediakan. Digunakan untuk Baca/Tulis. Nama tabel metode koneksi Anda akan berinteraksi dengan.
- `query`— Digunakan untuk Baca. Kueri SELECT SQL yang mendefinisikan apa yang harus diambil saat membaca dari Teradata.
- `staging_fs_url` — Diperlukan. Digunakan untuk Baca/Tulis. Nilai Valid: `s3a` URL. URL bucket atau folder Amazon S3 yang akan digunakan untuk penyimpanan sementara.

Nilai ConnectionType kustom dan AWS Marketplace

Sumber daya yang dimaksud meliputi:

- `"connectionType": "marketplace.athena"`: Mengkhususkan koneksi ke penyimpanan data Amazon Athena. Koneksi menggunakan sebuah konektor dari AWS Marketplace.
- `"connectionType": "marketplace.spark"`: Mengkhususkan koneksi ke Penyimpanan data Apache Spark. Koneksi menggunakan sebuah konektor dari AWS Marketplace.
- `"connectionType": "marketplace.jdbc"`: Mengkhususkan koneksi ke Penyimpanan data JDBC. Koneksi menggunakan sebuah konektor dari AWS Marketplace.
- `"connectionType": "custom.athena"`: Mengkhususkan koneksi ke penyimpanan data Amazon Athena. Koneksi menggunakan konektor khusus yang Anda unggahAWS Glue Studio.
- `"connectionType": "custom.spark"`: Mengkhususkan koneksi ke Penyimpanan data Apache Spark. Koneksi menggunakan konektor khusus yang Anda unggahAWS Glue Studio.
- `"connectionType": "custom.jdbc"`: Mengkhususkan koneksi ke Penyimpanan data JDBC. Koneksi menggunakan konektor khusus yang Anda unggahAWS Glue Studio.

Pilihan koneksi untuk jenis custom.jdbc atau marketplace.jdbc

- `className` — String, wajib, nama kelas driver.
- `connectionName` — String, wajib, nama koneksi yang dikaitkan dengan konektor.
- `url` — String, wajib, URL JDBC dengan placeholder (`${}`) yang digunakan untuk membangun koneksi ke sumber data. Placeholder `${secretKey}` diganti dengan rahasia dari nama yang sama di AWS Secrets Manager. Lihat dokumentasi penyimpanan data untuk informasi lebih lanjut tentang cara membangun URL.
- `secretId` atau `user/password` — String, wajib, yang digunakan untuk mengambil kredensial untuk URL.

- `dbTable` atau `query` — String, wajib, tabel atau kueri SQL tempat untuk mendapatkan data. Anda dapat menentukan salah satu dari `dbTable` atau `query`, bukan keduanya.
- `partitionColumn` — String, opsional, nama kolom integer yang digunakan untuk pemartisian. Opsi ini bekerja hanya ketika ia disertakan dengan `lowerBound`, `upperBound`, dan `numPartitions`. Pilihan ini bekerja dengan cara yang sama seperti pada pembaca Spark SQL JDBC. Untuk informasi selengkapnya, lihat JDBC Ke Database Lain di Apache Spark SQL, dan Panduan Datasets. DataFrames

  Nilai `lowerBound` dan `upperBound` digunakan untuk menentukan langkah partisi, bukan untuk menyaring baris dalam tabel. Semua baris dalam tabel dipartisi dan dikembalikan.

  > **Note**
  >
  > Bila menggunakan sebuah kueri bukan nama sebuah tabel, maka Anda harus memvalidasi bahwa kueri bekerja dengan syarat pemartisian yang ditentukan. Misalnya:
  >
  > - Jika format kueri Anda adalah `"SELECT col1 FROM table1"`, maka uji kueri dengan menambahkan klausul `WHERE` pada akhir kueri yang menggunakan kolom partisi.
  > - Jika format kueri Anda adalah `"SELECT col1 FROM table1 WHERE col2=val"`, maka uji kueri dengan memperluas klausul `WHERE` dengan `AND` dan ekspresi yang menggunakan kolom partisi.

- `lowerBound` — Integer, opsional, nilai minimum `partitionColumn` yang digunakan untuk memutuskan langkah partisi.
- `upperBound` — Integer, opsional, nilai maksimum `partitionColumn` yang digunakan untuk memutuskan langkah partisi.
- `numPartitions` — Integer, opsional, jumlah partisi. Nilai ini, bersama dengan `lowerBound` (inklusif) dan `upperBound` (eksklusif), membentuk langkah partisi untuk ekspresi klausul `WHERE` yang dihasilkan yang digunakan untuk membagi `partitionColumn`.

  > **Important**
  >
  > Hati-hati dengan jumlah partisi karena terlalu banyak partisi dapat menyebabkan masalah pada sistem basis data eksternal Anda.

- `filterPredicate` — String, opsional, klausul syarat ekstra untuk mem-filter data dari sumber. Misalnya:

```
BillingCity='Mountain View'
```

Saat menggunakan sebuah kueri bukan sebuah nama tabel, Anda harus memvalidasi bahwa kueri bekerja dengan `filterPredicate` yang ditentukan. Misalnya:

- Jika format kueri Anda adalah `"SELECT col1 FROM table1"`, maka uji kueri dengan menambahkan klausul `WHERE` pada akhir kueri yang menggunakan predikat filter.
- Jika format kueri Anda adalah `"SELECT col1 FROM table1 WHERE col2=val"`, maka uji kueri dengan memperluas klausul `WHERE` dengan `AND` dan ekspresi yang menggunakan predikat filter.

- `dataTypeMapping` — Kamus, pemetaan jenis data kustom, opsional, yang membangun pemetaan dari jenis data JDBC ke jenis data Glue. Misalnya, opsi `"dataTypeMapping":{"FLOAT":"STRING"}` memetakan bidang data tipe JDBC `FLOAT` ke dalam `String` tipe Java dengan memanggil `ResultSet.getString()` metode driver, dan menggunakannya untuk membangun AWS Glue catatan. Objek `ResultSet` dilaksanakan oleh masing-masing driver, sehingga perilaku bersifat spesifik untuk driver yang Anda gunakan. Lihat dokumentasi untuk driver JDBC Anda untuk memahami bagaimana driver melakukan konversi.
- Tipe AWS Glue data yang didukung saat ini adalah:
  - TANGGAL
  - STRING
  - TIMESTAMP
  - INT
  - MENGAMBANG
  - LONG
  - BIGDECIMAL
  - BYTE
  - SHORT
  - DOBEL

  Jenis data JDBC yang didukung adalah Java8 java.sql.types.

  Pemetaan tipe data default (dari JDBC ke) adalah: AWS Glue

  - DATE -> DATE
  - VARCHAR -> STRING

- CHAR -> STRING
- LONGNVARCHAR -> STRING
- TIMESTAMP -> TIMESTAMP
- INTEGER -> INT
- FLOAT -> FLOAT
- REAL -> FLOAT
- BIT -> BOOLEAN
- BOOLEAN -> BOOLEAN
- BIGINT -> LONG
- DECIMAL -> BIGDECIMAL
- NUMERIC -> BIGDECIMAL
- TINYINT -> SHORT
- SMALLINT -> SHORT
- DOUBLE -> DOUBLE

Jika Anda menggunakan pemetaan tipe data kustom dengan opsi `dataTypeMapping`, maka Anda dapat menimpa pemetaan tipe data default. Hanya tipe data JDBC yang tercantum dalam pilihan `dataTypeMapping` yang terpengaruh; pemetaan default digunakan untuk semua jenis data JDBC lainnya. Anda dapat menambahkan pemetaan untuk jenis data JDBC tambahan, jika diperlukan. Jika tipe data JDBC tidak disertakan dalam pemetaan default atau pemetaan khusus, maka tipe data akan dikonversi ke tipe data secara default. AWS Glue `STRING`

Contoh kode Python berikut ini menunjukkan cara membaca ke basis data JDBC dengan driver JDBC AWS Marketplace. Ia menunjukkan membaca dari basis data dan menulis ke sebuah lokasi S3.

```
    import sys
    from awsglue.transforms import *
    from awsglue.utils import getResolvedOptions
    from pyspark.context import SparkContext
    from awsglue.context import GlueContext
    from awsglue.job import Job

    ## @params: [JOB_NAME]
    args = getResolvedOptions(sys.argv, ['JOB_NAME'])
```

```
    sc = SparkContext()
    glueContext = GlueContext(sc)
    spark = glueContext.spark_session
    job = Job(glueContext)
    job.init(args['JOB_NAME'], args)
    ## @type: DataSource
    ## @args: [connection_type = "marketplace.jdbc", connection_options =
     {"dataTypeMapping":{"INTEGER":"STRING"},"upperBound":"200","query":"select id,
       name, department from department where id < 200","numPartitions":"4",
       "partitionColumn":"id","lowerBound":"0","connectionName":"test-connection-
jdbc"},
        transformation_ctx = "DataSource0"]
    ## @return: DataSource0
    ## @inputs: []
    DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type =
      "marketplace.jdbc", connection_options = {"dataTypeMapping":{"INTEGER":"STRING"},
      "upperBound":"200","query":"select id, name, department from department where
       id < 200","numPartitions":"4","partitionColumn":"id","lowerBound":"0",
       "connectionName":"test-connection-jdbc"}, transformation_ctx = "DataSource0")
    ## @type: ApplyMapping
    ## @args: [mappings = [("department", "string", "department", "string"), ("name",
 "string",
     "name", "string"), ("id", "int", "id", "int")], transformation_ctx =
 "Transform0"]
    ## @return: Transform0
    ## @inputs: [frame = DataSource0]
    Transform0 = ApplyMapping.apply(frame = DataSource0, mappings = [("department",
 "string",
     "department", "string"), ("name", "string", "name", "string"), ("id", "int",
 "id", "int")],
      transformation_ctx = "Transform0")
    ## @type: DataSink
    ## @args: [connection_type = "s3", format = "json", connection_options = {"path":
     "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0"]
    ## @return: DataSink0
    ## @inputs: [frame = Transform0]
    DataSink0 = glueContext.write_dynamic_frame.from_options(frame = Transform0,
      connection_type = "s3", format = "json", connection_options = {"path":
      "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0")
    job.commit()
```

Pilihan koneksi untuk tipe custom.athena atau marketplace.athena

- `className` — String, wajib, nama kelas driver. Bila Anda menggunakan Athena-CloudWatch konektor, nilai parameter ini adalah awalan dari nama kelas (misalnya,). `"com.amazonaws.athena.connectors"` CloudWatch Konektor Athena terdiri dari dua kelas: pengendali metadata dan penangan rekaman. Jika Anda menyediakan prefiks umum di sini, maka API memuat kelas yang benar berdasarkan prefiks itu.
- `tableName`— String, diperlukan, nama aliran CloudWatch log untuk dibaca. Snippet kode ini menggunakan nama tampilan khusus `all_log_streams`, yang berarti bahwa bingkai data dinamis yang dikembalikan akan berisi data dari semua pengaliran log dalam grup log.
- `schemaName`— String, diperlukan, nama grup CloudWatch log untuk dibaca. Sebagai contoh, `/aws-glue/jobs/output`.
- `connectionName` — String, wajib, nama koneksi yang dikaitkan dengan konektor.

Untuk opsi tambahan untuk konektor ini, lihat file README CloudWatch Konektor Amazon Athena aktif. GitHub

Kode Python contoh berikut ini menunjukkan bagaimana membaca dari penyimpanan data Athena menggunakan konektor AWS Marketplace. Ia menunjukkan membaca dari Athena dan menulis ke sebuah lokasi S3.

```
    import sys
    from awsglue.transforms import *
    from awsglue.utils import getResolvedOptions
    from pyspark.context import SparkContext
    from awsglue.context import GlueContext
    from awsglue.job import Job

    ## @params: [JOB_NAME]
    args = getResolvedOptions(sys.argv, ['JOB_NAME'])

    sc = SparkContext()
    glueContext = GlueContext(sc)
    spark = glueContext.spark_session
    job = Job(glueContext)
    job.init(args['JOB_NAME'], args)
    ## @type: DataSource
    ## @args: [connection_type = "marketplace.athena", connection_options =
     {"tableName":"all_log_streams","schemaName":"/aws-glue/jobs/output",
```

```
      "connectionName":"test-connection-athena"}, transformation_ctx = "DataSource0"]
   ## @return: DataSource0
   ## @inputs: []
   DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type =
     "marketplace.athena", connection_options = {"tableName":"all_log_streams",,
     "schemaName":"/aws-glue/jobs/output","connectionName":
     "test-connection-athena"}, transformation_ctx = "DataSource0")
   ## @type: ApplyMapping
   ## @args: [mappings = [("department", "string", "department", "string"), ("name",
"string",
     "name", "string"), ("id", "int", "id", "int")], transformation_ctx =
"Transform0"]
   ## @return: Transform0
   ## @inputs: [frame = DataSource0]
   Transform0 = ApplyMapping.apply(frame = DataSource0, mappings = [("department",
"string",
     "department", "string"), ("name", "string", "name", "string"), ("id", "int",
"id", "int")],
      transformation_ctx = "Transform0")
   ## @type: DataSink
   ## @args: [connection_type = "s3", format = "json", connection_options = {"path":
    "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0"]
   ## @return: DataSink0
   ## @inputs: [frame = Transform0]
   DataSink0 = glueContext.write_dynamic_frame.from_options(frame = Transform0,
     connection_type = "s3", format = "json", connection_options = {"path":
     "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0")
   job.commit()
```

Pilihan koneksi untuk jenis custom.spark atau marketplace.spark

- `className` — String, wajib, nama kelas konektor.
- `secretId` — String, opsional, yang digunakan untuk mengambil kredensial untuk koneksi konektor.
- `connectionName` — String, wajib, nama koneksi yang dikaitkan dengan konektor.
- Pilihan lain tergantung pada penyimpanan data. Misalnya, opsi OpenSearch konfigurasi dimulai dengan awalan`es`, seperti yang dijelaskan dalam dokumentasi Elasticsearch for Apache Hadoop. Koneksi Spark ke Snowflake menggunakan pilihan seperti `sfUser` dan `sfPassword`, seperti yang dijelaskan dalam dokumentasi Menggunakan Konektor Spark di panduan Menghubungkan ke Snowflake.

Contoh kode Python berikut menunjukkan cara membaca dari penyimpanan OpenSearch data menggunakan koneksi`marketplace.spark`.

```
   import sys
   from awsglue.transforms import *
   from awsglue.utils import getResolvedOptions
   from pyspark.context import SparkContext
   from awsglue.context import GlueContext
   from awsglue.job import Job

   ## @params: [JOB_NAME]
   args = getResolvedOptions(sys.argv, ['JOB_NAME'])

   sc = SparkContext()
   glueContext = GlueContext(sc)
   spark = glueContext.spark_session
   job = Job(glueContext)
   job.init(args['JOB_NAME'], args)
   ## @type: DataSource
   ## @args: [connection_type = "marketplace.spark", connection_options =
{"path":"test",
     "es.nodes.wan.only":"true","es.nodes":"https://<AWS endpoint>",
     "connectionName":"test-spark-es","es.port":"443"}, transformation_ctx =
"DataSource0"]
   ## @return: DataSource0
   ## @inputs: []
   DataSource0 = glueContext.create_dynamic_frame.from_options(connection_type =
     "marketplace.spark", connection_options = {"path":"test","es.nodes.wan.only":
     "true","es.nodes":"https://<AWS endpoint>","connectionName":
     "test-spark-es","es.port":"443"}, transformation_ctx = "DataSource0")
   ## @type: DataSink
   ## @args: [connection_type = "s3", format = "json", connection_options = {"path":
        "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0"]
   ## @return: DataSink0
   ## @inputs: [frame = DataSource0]
   DataSink0 = glueContext.write_dynamic_frame.from_options(frame = DataSource0,
      connection_type = "s3", format = "json", connection_options = {"path":
      "s3://<S3 path>/", "partitionKeys": []}, transformation_ctx = "DataSink0")
   job.commit()
```

Opsi umum

Opsi di bagian ini disediakan sebagai`connection_options`, tetapi tidak secara khusus berlaku untuk satu konektor.

Parameter berikut digunakan secara umum saat mengkonfigurasi bookmark. Mereka mungkin berlaku untuk alur kerja Amazon S3 atau JDBC. Untuk informasi selengkapnya, lihat the section called “Menggunakan bookmark pekerjaan”.

- `jobBookmarkKeys`— Array nama kolom.
- `jobBookmarkKeysSortOrder`— String mendefinisikan cara membandingkan nilai berdasarkan urutan pengurutan. Nilai yang valid:`"asc"`,`"desc"`.
- `useS3ListImplementation`— Digunakan untuk mengelola kinerja memori saat mencantumkan konten bucket Amazon S3. Untuk informasi selengkapnya, lihat Optimalkan manajemen memori di AWS Glue.

Opsi format data untuk input dan output untuk Spark AWS Glue

Halaman ini menawarkan informasi tentang dukungan fitur dan parameter konfigurasi untuk format data yang didukung oleh AWS Glue for Spark. Lihat berikut ini untuk deskripsi penggunaan dan penerapan informasi ini.

Dukungan fitur di seluruh format data di AWS Glue

Setiap format data dapat mendukung fitur AWS Glue yang berbeda. Fitur umum berikut mungkin atau mungkin tidak didukung berdasarkan jenis format Anda. Lihat dokumentasi untuk format data Anda untuk memahami cara memanfaatkan fitur kami untuk memenuhi kebutuhan Anda.

Baca	AWSGlue dapat mengenali dan menafsirkan format data ini tanpa sumber daya tambahan, seperti konektor.
Tulis	AWSGlue dapat menulis data dalam format ini tanpa sumber daya tambahan. Anda dapat menyertakan pustaka pihak ketiga dalam pekerjaan Anda dan menggunakan fungsi Apache Spark standar untuk menulis data, seperti yang Anda lakukan di lingkungan Spark

	lainnya. Untuk informasi selengkapnya tentang menyertakan pustaka, lihatthe section called “Pustaka Python”.
Streamin dibaca	AWSGlue dapat mengenali dan menafsirkan format data ini dari Apache Kafka, Amazon Managed Streaming for Apache Kafka atau aliran pesan Amazon Kinesis. Kami mengharapkan aliran untuk menyajikan data dalam format yang konsisten, sehingga mereka dibaca sebagai`DataFrames` .
Kelompoan file kecil	AWSGlue dapat mengelompokkan file bersama-sama untuk pekerjaan batch yang dikirim ke setiap node saat melakukan transformasi AWS Glue. Ini secara signifikan dapat meningkatkan kinerja untuk beban kerja yang melibatkan sejumlah besar file kecil. Untuk informasi selengkapnya, lihat the section called “Mengelompokkan file masukan”.
Bookmar tugas	AWSGlue dapat melacak kemajuan transformasi yang melakukan pekerjaan yang sama pada kumpulan data yang sama di seluruh pekerjaan yang dijalankan dengan bookmark pekerjaan . Ini dapat meningkatkan kinerja untuk beban kerja yang melibatkan kumpulan data di mana pekerjaan hanya perlu dilakukan pada data baru sejak pekerjaan terakhir dijalankan. Untuk informasi selengkapnya, lihat the section called “Melacak data yang diproses menggunakan bookmark pekerjaan”.

Parameter yang digunakan untuk berinteraksi dengan format data di AWS Glue

Jenis koneksi AWS Glue tertentu mendukung beberapa `format` jenis, mengharuskan Anda menentukan informasi tentang format data Anda dengan `format_options` objek saat menggunakan metode sepertiGlueContext.write_dynamic_frame.from_options.

- `s3`— Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue:Parameter koneksi S3. Anda juga dapat melihat dokumentasi untuk metode yang memfasilitasi jenis koneksi ini: the section called “create_dynamic_frame_from_options” dan dengan Python dan the section called “write_dynamic_frame_from_options” metode the section called “getSourceWithFormat” Scala yang sesuai dan. the section called “getSinkWithFormat”
- `kinesis`— Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue:Parameter koneksi Kinesis. Anda juga dapat melihat dokumentasi untuk metode yang memfasilitasi jenis koneksi ini: the section called “create_data_frame_from_options” dan metode Scala yang sesuai. the section called “createDataFrameFromOptions”
- `kafka`— Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue:Parameter koneksi Kafka. Anda juga dapat melihat dokumentasi untuk metode yang memfasilitasi jenis koneksi ini: the section called “create_data_frame_from_options” dan metode Scala yang sesuai. the section called “createDataFrameFromOptions”

Beberapa jenis koneksi tidak memerlukan`format_options`. Misalnya, dalam penggunaan normal, koneksi JDBC ke database relasional mengambil data dalam format data tabular yang konsisten. Oleh karena itu, membaca dari koneksi JDBC tidak memerlukan. `format_options`

Beberapa metode untuk membaca dan menulis data dalam lem tidak diperlukan`format_options`. Misalnya, menggunakan `GlueContext.create_dynamic_frame.from_catalog` dengan AWS Glue crawler. Crawler menentukan bentuk data Anda. Saat menggunakan crawler, pengklasifikasi AWS Glue akan memeriksa data Anda untuk membuat keputusan cerdas tentang cara merepresentasikan format data Anda. Kemudian akan menyimpan representasi data Anda di AWS Glue Data Catalog, yang dapat digunakan dalam skrip AWS Glue ETL untuk mengambil data Anda dengan metode tersebut`GlueContext.create_dynamic_frame.from_catalog`. Crawler menghapus kebutuhan untuk menentukan informasi secara manual tentang format data Anda.

Untuk pekerjaan yang mengakses tabel yang AWS Lake Formation diatur, AWS Glue mendukung membaca dan menulis semua format yang didukung oleh tabel yang diatur Lake Formation. Untuk daftar format yang didukung saat ini untuk tabel yang AWS Lake Formation diatur, lihat Catatan dan Pembatasan untuk Tabel yang Diatur dalam Panduan AWS Lake FormationPengembang.

> **Note**
>
> Untuk menulis Apache Parquet, AWS Glue ETL hanya mendukung penulisan ke tabel yang diatur dengan menentukan opsi untuk jenis penulis Parket kustom yang dioptimalkan untuk Dynamic Frames. Saat menulis ke tabel yang diatur dengan `parquet` format, Anda harus menambahkan kunci `useGlueParquetWriter` dengan nilai `true` dalam parameter tabel.

Topik

Menggunakan format CSV di AWS Glue

AWSGlue mengambil data dari sumber dan menulis data ke target yang disimpan dan diangkut dalam berbagai format data. Jika data Anda disimpan atau diangkut dalam format data CSV, dokumen ini memperkenalkan fitur yang tersedia untuk menggunakan data Anda di Glue. AWS

AWSGlue mendukung penggunaan format comma-separated value (CSV). Format ini adalah format data berbasis baris minimal. CSV sering tidak sepenuhnya sesuai dengan standar, tetapi Anda dapat merujuk ke RFC 4180 dan RFC 7111 untuk informasi lebih lanjut.

Anda dapat menggunakan AWS Glue untuk membaca CSV dari Amazon S3 dan dari sumber streaming serta menulis CSV ke Amazon S3. Anda dapat membaca dan menulis `bzip` dan `gzip` mengarsipkan yang berisi file CSV dari S3. Anda mengonfigurasi perilaku kompresi pada Parameter koneksi S3 alih-alih dalam konfigurasi yang dibahas di halaman ini.

Tabel berikut menunjukkan fitur AWS Glue umum mana yang mendukung opsi format CSV.

Baca	Tulis	Streaming dibaca	Kelompokkan file kecil	Bookmark tugas
Didukung	Didukung	Didukung	Didukung	Didukung

Contoh: Baca file CSV atau folder dari S3

Prasyarat: Anda akan memerlukan jalur S3 (`s3path`) ke file CSV atau folder yang ingin Anda baca.

Konfigurasi: Dalam opsi fungsi Anda, tentukan`format="csv"`. Dalam Anda`connection_options`, gunakan `paths` kunci untuk menentukan`s3path`. Anda dapat mengonfigurasi bagaimana pembaca berinteraksi dengan S3 di. `connection_options` Untuk detailnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue:Parameter koneksi S3. Anda dapat mengonfigurasi bagaimana pembaca menafsirkan file CSV di file Anda. `format_options` Untuk detailnya, lihat Referensi Konfigurasi CSV.

Skrip AWS Glue ETL berikut menunjukkan proses membaca file CSV atau folder dari S3.

Kami menyediakan pembaca CSV kustom dengan pengoptimalan kinerja untuk alur kerja umum melalui kunci konfigurasi. `optimizePerformance` Untuk menentukan apakah pembaca ini tepat untuk beban kerja Anda, lihatthe section called “Menggunakan pembaca CSV yang dioptimalkan”.

Python

Untuk contoh ini, gunakan metode create_dynamic_frame.from_options.

```
# Example: Read CSV from S3
# For show, we handle a CSV with a header row.  Set the withHeader option.
# Consider whether optimizePerformance is right for your workflow.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
```

```
    format="csv",
    format_options={
        "withHeader": True,
        # "optimizePerformance": True,
    },
)
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
dataFrame = spark.read\
    .format("csv")\
    .option("header", "true")\
    .load("s3://s3path")
```

Scala

Untuk contoh ini, gunakan operasi getSourceWithFormat.

```
// Example: Read CSV from S3
// For show, we handle a CSV with a header row.  Set the withHeader option.
// Consider whether optimizePerformance is right for your workflow.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      formatOptions=JsonOptions("""{"withHeader": true}"""),
      connectionType="s3",
      format="csv",
      options=JsonOptions("""{"paths": ["s3://s3path"], "recurse": true}""")
    ).getDynamicFrame()
  }
}
```

Anda juga dapat menggunakan DataFrames dalam script (`org.apache.spark.sql.DataFrame`).

```
val dataFrame = spark.read
  .option("header","true")
  .format("csv")
  .load("s3://s3path“)
```

Contoh: Tulis file dan folder CSV ke S3

Prasyarat: Anda akan memerlukan initialized DataFrame (`dataFrame`) atau a (). DynamicFrame `dynamicFrame` Anda juga akan membutuhkan jalur output S3 yang Anda harapkan,`s3path`.

Konfigurasi: Dalam opsi fungsi Anda, tentukan`format="csv"`. Dalam Anda`connection_options`, gunakan `paths` kunci untuk menentukan`s3path`. Anda dapat mengonfigurasi bagaimana penulis berinteraksi dengan S3 di. `connection_options` Untuk detailnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue:Parameter koneksi S3. Anda dapat mengonfigurasi bagaimana operasi Anda menulis konten file Anda`format_options`. Untuk detailnya, lihat Referensi Konfigurasi CSV. Skrip AWS Glue ETL berikut menunjukkan proses penulisan file CSV dan folder ke S3.

Python

Untuk contoh ini, gunakan metode write_dynamic_frame.from_options.

```
# Example: Write CSV to S3
# For show, customize how we write string type values.  Set quoteChar to -1 so our
 values are not quoted.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    connection_options={"path": "s3://s3path"},
    format="csv",
    format_options={
        "quoteChar": -1,
    },
)
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
dataFrame.write\
    .format("csv")\
    .option("quote", None)\
    .mode("append")\
    .save("s3://s3path")
```

Scala

Untuk contoh ini, gunakan metode getSinkWithFormat.

```
// Example: Write CSV to S3
// For show, customize how we write string type values. Set quoteChar to -1 so our
 values are not quoted.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
        connectionType="s3",
        options=JsonOptions("""{"path": "s3://s3path"}"""),
        format="csv"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Anda juga dapat menggunakan DataFrames dalam script (`org.apache.spark.sql.DataFrame`).

```
dataFrame.write
    .format("csv")
    .option("quote", null)
    .mode("Append")
    .save("s3://s3path")
```

Referensi konfigurasi CSV

Anda dapat menggunakan yang berikut ini di `format_options` mana pun pustaka AWS Glue menentukan`format="csv"`:

- `separator`—Menentukan karakter pembatas. Defaultnya adalah koma, tetapi karakter lain dapat ditentukan.
  - Jenis: Teks, Default: `","`
- `escaper`- Menentukan karakter yang akan digunakan untuk melarikan diri. Opsi ini hanya digunakan saat membaca file CSV, bukan menulis. Jika diaktifkan, karakter yang segera mengikuti digunakan apa adanya, kecuali untuk satu set kecil pelarian terkenal (`\n`,, `\r\t`, dan`\0`).
  - Jenis: Teks, Default: tidak ada
- `quoteChar`- Menentukan karakter yang akan digunakan untuk mengutip. Defaultnya adalah kutipan ganda. Atur ini ke `-1` untuk menonaktifkan pengutipan seluruhnya.
  - Jenis: Teks, Default: `'"'`
- `multiLine`- Menentukan apakah catatan tunggal dapat menjangkau beberapa baris. Hal ini dapat terjadi ketika bidang berisi karakter baris baru yang dikutip. Anda harus menetapkan opsi ini ke `True` jika ada catatan yang mencakup beberapa baris. Mengaktifkan `multiLine` dapat menurunkan kinerja karena membutuhkan pemisahan file yang lebih hati-hati saat mengurai.
  - Jenis: Boolean, Default: `false`
- `withHeader`- Menentukan apakah untuk memperlakukan baris pertama sebagai header. Opsi ini dapat digunakan dalam kelas `DynamicFrameReader`.
  - Jenis: Boolean, Default: `false`
- `writeHeader`- Menentukan apakah akan menulis header untuk output. Opsi ini dapat digunakan dalam kelas `DynamicFrameWriter`.
  - Jenis: Boolean, Default: `true`
- `skipFirst`- Menentukan apakah akan melewati baris data pertama.
  - Jenis: Boolean, Default: `false`
- `optimizePerformance`— Menentukan apakah akan menggunakan pembaca CSV SIMD canggih bersama dengan format memori kolumnar berbasis Apache Arrow. Hanya tersedia dalam AWS Glue 3.0+.
  - Jenis: Boolean, Default: `false`
- `strictCheckForQuoting`— Saat menulis CSV, Glue dapat menambahkan tanda kutip ke nilai yang ditafsirkan sebagai string. Ini dilakukan untuk mencegah ambiguitas dalam apa yang tertulis.

Untuk menghemat waktu ketika memutuskan apa yang akan ditulis, Glue dapat mengutip dalam situasi tertentu di mana kutipan tidak diperlukan. Mengaktifkan pemeriksaan ketat akan melakukan perhitungan yang lebih intensif dan hanya akan mengutip jika benar-benar diperlukan. Hanya tersedia dalam AWS Glue 3.0+.

- Jenis: Boolean, Default: `false`

Optimalkan kinerja baca dengan pembaca SIMD CSV vektor

AWS Glueversi 3.0 menambahkan pembaca CSV yang dioptimalkan yang secara signifikan dapat mempercepat kinerja pekerjaan secara keseluruhan dibandingkan dengan pembaca CSV berbasis baris.

Pembaca yang dioptimalkan:

- Menggunakan instruksi CPU SIMD untuk membaca dari disk
- Segera menulis catatan ke memori dalam format kolom (Apache Arrow)
- Membagi catatan menjadi batch

Ini menghemat waktu pemrosesan ketika catatan akan di-batch atau dikonversi ke format kolumnar nanti. Beberapa contoh adalah ketika mengubah skema atau mengambil data berdasarkan kolom.

Untuk menggunakan pembaca yang dioptimalkan, atur `"optimizePerformance"` ke `true` dalam properti `format_options` or table.

```
glueContext.create_dynamic_frame.from_options(
    frame = datasource1,
    connection_type = "s3",
    connection_options = {"paths": ["s3://s3path"]},
    format = "csv",
    format_options={
        "optimizePerformance": True,
        "separator": ","
        },
    transformation_ctx = "datasink2")
```

Keterbatasan untuk pembaca CSV vektor

Perhatikan batasan berikut dari pembaca CSV vektor:

- Itu tidak mendukung opsi `multiLine` dan `escaper` format. Ini menggunakan default `escaper` karakter kutipan ganda. `'"'` Ketika opsi ini disetel, AWS Glue secara otomatis kembali menggunakan pembaca CSV berbasis baris.
- Itu tidak mendukung pembuatan DynamicFrame dengan ChoiceType.
- Itu tidak mendukung pembuatan catatan kesalahan DynamicFrame dengan.
- Itu tidak mendukung membaca file CSV dengan karakter multibyte seperti karakter Jepang atau China.

Menggunakan format Parket di AWS Glue

AWSGlue mengambil data dari sumber dan menulis data ke target yang disimpan dan diangkut dalam berbagai format data. Jika data Anda disimpan atau diangkut dalam format data Parket, dokumen ini memperkenalkan fitur yang tersedia untuk menggunakan data Anda di Glue. AWS

AWSGlue mendukung menggunakan format Parket. Format ini adalah format data berbasis kolom yang berorientasi kinerja. Untuk pengenalan format oleh otoritas standar lihat, Apache Parquet Documentation Overview.

Anda dapat menggunakan AWS Glue untuk membaca file Parket dari Amazon S3 dan dari sumber streaming serta menulis file Parket ke Amazon S3. Anda dapat membaca dan menulis `bzip` dan `gzip` mengarsipkan yang berisi file Parket dari S3. Anda mengonfigurasi perilaku kompresi pada Parameter koneksi S3 alih-alih dalam konfigurasi yang dibahas di halaman ini.

Tabel berikut menunjukkan fitur AWS Glue umum mana yang mendukung opsi format Parket.

Baca	Tulis	Streaming dibaca	Kelompokkan file kecil	Bookmark tugas
Didukung	Didukung	Didukung	Tidak didukung	Didukung *

* Didukung dalam AWS Glue versi 1.0+

Contoh: Baca file atau folder Parket dari S3

Prasyarat: Anda akan memerlukan jalur S3 (`s3path`) ke file atau folder Parket yang ingin Anda baca.

Konfigurasi: Dalam opsi fungsi Anda, tentukan`format="parquet"`. Dalam Anda`connection_options`, gunakan `paths` kunci untuk menentukan Anda`s3path`.

Anda dapat mengonfigurasi bagaimana pembaca berinteraksi dengan S3 di file. `connection_options` Untuk detailnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue:Parameter koneksi S3.

Anda dapat mengonfigurasi bagaimana pembaca menafsirkan file Parket di file Anda. `format_options` Untuk detailnya, lihat Referensi Konfigurasi Parket.

Skrip AWS Glue ETL berikut menunjukkan proses membaca file Parket atau folder dari S3:

Python

Untuk contoh ini, gunakan metode create_dynamic_frame.from_options.

```
# Example: Read Parquet from S3

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type = "s3",
    connection_options = {"paths": ["s3://s3path/"]},
    format = "parquet"
)
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
dataFrame = spark.read.parquet("s3://s3path/")
```

Scala

Untuk contoh ini, gunakan metode getSourceWithFormat.

```
// Example: Read Parquet from S3

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext
```

```
object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType="s3",
      format="parquet",
      options=JsonOptions("""{"paths": ["s3://s3path"]}""")
    ).getDynamicFrame()
  }
}
```

Anda juga dapat menggunakan DataFrames dalam script (`org.apache.spark.sql.DataFrame`).

```
spark.read.parquet("s3://s3path/")
```

Contoh: Tulis file dan folder Parket ke S3

Prasyarat: Anda akan memerlukan initialized DataFrame () atau ()`dataFrame`. DynamicFrame `dynamicFrame` Anda juga akan membutuhkan jalur output S3 yang Anda harapkan,`s3path`.

Konfigurasi: Dalam opsi fungsi Anda, tentukan`format="parquet"`. Dalam Anda`connection_options`, gunakan `paths` kunci untuk menentukan`s3path`.

Anda selanjutnya dapat mengubah cara penulis berinteraksi dengan S3 di. `connection_options` Untuk detailnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue:Parameter koneksi S3. Anda dapat mengonfigurasi bagaimana operasi Anda menulis konten file Anda`format_options`. Untuk detailnya, lihat Referensi Konfigurasi Parket.

Skrip AWS Glue ETL berikut menunjukkan proses penulisan file dan folder Parket ke S3.

Kami menyediakan penulis Parket khusus dengan pengoptimalan kinerja untuk DynamicFrames, melalui kunci konfigurasi. `useGlueParquetWriter` Untuk menentukan apakah penulis ini tepat untuk beban kerja Anda, lihat Glue Parquet Writer.

Python

Untuk contoh ini, gunakan metode write_dynamic_frame.from_options.

```
# Example: Write Parquet to S3
# Consider whether useGlueParquetWriter is right for your workflow.

from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    format="parquet",
    connection_options={
        "path": "s3://s3path",
    },
    format_options={
        # "useGlueParquetWriter": True,
    },
)
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
df.write.parquet("s3://s3path/")
```

Scala

Untuk contoh ini, gunakan metode getSinkWithFormat.

```
// Example: Write Parquet to S3
// Consider whether useGlueParquetWriter is right for your workflow.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
```

```
        connectionType="s3",
        options=JsonOptions("""{"path": "s3://s3path"}"""),
        format="parquet"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Anda juga dapat menggunakan DataFrames dalam script (`org.apache.spark.sql.DataFrame`).

```
df.write.parquet("s3://s3path/")
```

Referensi konfigurasi parket

Anda dapat menggunakan yang berikut ini di `format_options` mana pun pustaka AWS Glue menentukan`format="parquet"`:

- `useGlueParquetWriter`— Menentukan penggunaan penulis Parket kustom yang memiliki optimasi kinerja untuk alur kerja. DynamicFrame Untuk detail penggunaan, lihat Glue Parquet Writer.
  - Jenis: Boolean, Default: `false`
- `compression`- Menentukan codec kompresi yang digunakan. Nilai sepenuhnya kompatibel dengan`org.apache.parquet.hadoop.metadata.CompressionCodecName`.
  - Jenis: Teks Terhitung, Default: `"snappy"`
  - Nilai:`"uncompressed"`,`"snappy"`,`"gzip"`, dan `"lzo"`
- `blockSize`- Menentukan ukuran dalam byte dari kelompok baris yang disangga dalam memori. Anda menggunakan ini untuk tuning kinerja. Ukuran harus dibagi persis menjadi sejumlah megabyte.
  - Jenis: Numerik, Default: `134217728`
  - Nilai defaultnya sama dengan 128 MB.
- `pageSize`- Menentukan ukuran dalam byte halaman. Anda menggunakan ini untuk tuning kinerja. Halaman adalah unit terkecil yang harus dibaca sepenuhnya untuk mengakses satu catatan.
  - Jenis: Numerik, Default: `1048576`
  - Nilai defaultnya sama dengan 1 MB.

> **Note**
>
> Selain itu, opsi apa pun yang diterima oleh kode SparkSQL yang mendasarinya dapat diteruskan ke format ini melalui parameter peta. `connection_options` Misalnya, Anda dapat mengatur konfigurasi Spark seperti MergeSchema untuk pembaca AWS Glue Spark untuk menggabungkan skema untuk semua file.

Optimalkan kinerja menulis dengan AWS Glue Parquet writer

> **Note**
>
> Penulis AWS Glue Parquet secara historis telah diakses melalui jenis `glueparquet` format. Pola akses ini tidak lagi dianjurkan. Sebagai gantinya, gunakan `parquet` tipe dengan `useGlueParquetWriter` diaktifkan.

Penulis AWS Glue Parquet memiliki peningkatan kinerja yang memungkinkan penulisan file Parket lebih cepat. Penulis tradisional menghitung skema sebelum menulis. Format Parket tidak menyimpan skema dengan cara yang dapat diambil dengan cepat, jadi ini mungkin memakan waktu. Dengan penulis AWS Glue Parquet, skema pra-komputasi tidak diperlukan. Penulis menghitung dan memodifikasi skema secara dinamis, saat data masuk.

Perhatikan batasan berikut saat Anda menentukan`useGlueParquetWriter`:

- Penulis hanya mendukung evolusi skema (seperti menambahkan atau menghapus kolom), tetapi tidak mengubah jenis kolom, seperti dengan`ResolveChoice`.
- Penulis tidak mendukung penulisan kosong DataFrames —misalnya, untuk menulis file khusus skema. Saat mengintegrasikan dengan Katalog Data AWS Glue dengan pengaturan`enableUpdateCatalog=True`, mencoba menulis kosong tidak DataFrame akan memperbarui Katalog Data. Ini akan menghasilkan pembuatan tabel di Katalog Data tanpa skema.

Jika transformasi Anda tidak memerlukan batasan ini, menyalakan penulis AWS Glue Parquet akan meningkatkan kinerja.

Menggunakan format XHTML di AWS Glue

AWSGlue mengambil data dari sumber dan menulis data ke target yang disimpan dan diangkut dalam berbagai format data. Jika data Anda disimpan atau diangkut dalam format data XHTML, dokumen ini memperkenalkan fitur yang tersedia untuk menggunakan data Anda di Glue. AWS

AWSGlue mendukung menggunakan format XHTML. Format ini mewakili struktur data yang sangat dapat dikonfigurasi dan didefinisikan secara kaku yang tidak berbasis baris atau kolom. XML-nya sangat terstandarisasi. Untuk pengenalan format oleh otoritas standar, lihat XMLEssentials.

Anda dapat menggunakan AWS Glue untuk membaca file XHTML dari Amazon S3, `bzip` serta `gzip` dan arsip yang berisi file XHTML. Anda mengonfigurasi perilaku kompresi pada Parameter koneksi S3 alih-alih dalam konfigurasi yang dibahas di halaman ini.

Tabel berikut menunjukkan fitur AWS Glue umum mana yang mendukung opsi format XHTML.

Baca	Tulis	Streaming dibaca	Kelompokkan file kecil	Bookmark tugas
Didukung	Tidak didukung	Tidak didukung	Didukung	Didukung

Contoh: Baca XMLdari S3

Pembaca XHTML mengambil nama tag XHTML. Ini memeriksa elemen dengan tag itu dalam inputnya untuk menyimpulkan skema dan mengisi a dengan nilai yang DynamicFrame sesuai. Fungsionalitas AWS Glue XMLberperilaku mirip dengan Sumber Data XMLuntuk Apache Spark. Anda mungkin bisa mendapatkan wawasan seputar perilaku dasar dengan membandingkan pembaca ini dengan dokumentasi proyek itu.

Prasyarat: Anda akan memerlukan jalur S3 (`s3path`) ke file atau folder XHTML yang ingin Anda baca, dan beberapa informasi tentang file XMLmu. Anda juga akan memerlukan tag untuk elemen XHTML yang ingin Anda baca,`xmlTag`.

Konfigurasi: Dalam opsi fungsi Anda, tentukan`format="xml"`. Dalam Anda`connection_options`, gunakan `paths` kunci untuk menentukan`s3path`. Anda dapat mengonfigurasi lebih lanjut bagaimana pembaca berinteraksi dengan S3 di file. `connection_options` Untuk detailnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue:Parameter koneksi S3. Dalam Anda`format_options`, gunakan `rowTag` kunci untuk menentukan`xmlTag`. Anda dapat mengkonfigurasi lebih lanjut bagaimana

pembaca menafsirkan file XHTML di file Anda. `format_options` Untuk detailnya, lihat Referensi Konfigurasi XHTML.

Berikut skrip AWS Glue ETL menunjukkan proses membaca file XHTML atau folder dari S3.

Python

Untuk contoh ini, gunakan metode create_dynamic_frame.from_options.

```
# Example: Read XML from S3
# Set the rowTag option to configure the reader.

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="xml",
    format_options={"rowTag": "xmlTag"},
)
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
dataFrame = spark.read\
    .format("xml")\
    .option("rowTag", "xmlTag")\
    .load("s3://s3path")
```

Scala

Untuk contoh ini, gunakan operasi getSourceWithFormat.

```
// Example: Read XML from S3
// Set the rowTag option to configure the reader.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.GlueContext
import org.apache.spark.sql.SparkSession
```

```
val glueContext = new GlueContext(SparkContext.getOrCreate())
val sparkSession: SparkSession = glueContext.getSparkSession

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val dynamicFrame = glueContext.getSourceWithFormat(
      formatOptions=JsonOptions("""{"rowTag": "xmlTag"}"""),
      connectionType="s3",
      format="xml",
      options=JsonOptions("""{"paths": ["s3://s3path"], "recurse": true}""")
    ).getDynamicFrame()
}
```

Anda juga dapat menggunakan DataFrames dalam script (`org.apache.spark.sql.DataFrame`).

```
val dataFrame = spark.read
  .option("rowTag", "xmlTag")
  .format("xml")
  .load("s3://s3path“)
```

Referensi konfigurasi XML

Anda dapat menggunakan yang berikut ini di `format_options` mana pun pustaka AWS Glue menentukan`format="xml"`:

- `rowTag`- Menentukan tag XMLdalam file untuk memperlakukan sebagai baris. Tag baris tidak dapat berupa penutupan-sendiri.
  - Jenis: Teks, Diperlukan
- `encoding`- Menentukan pengkodean karakter. Ini bisa berupa nama atau alias Charset yang didukung oleh lingkungan runtime kami. Kami tidak membuat jaminan khusus seputar dukungan pengkodean, tetapi pengkodean utama harus berfungsi.
  - Jenis: Teks, Default: `"UTF-8"`
- `excludeAttribute`- Menentukan apakah Anda ingin mengecualikan atribut dalam elemen atau tidak.
  - Jenis: Boolean, Default: `false`

- `treatEmptyValuesAsNulls`- Menentukan apakah untuk memperlakukan ruang putih sebagai nilai null.
  - Jenis: Boolean, Default: `false`
- `attributePrefix`— Awalan untuk atribut untuk membedakannya dari teks elemen anak. Prefiks ini digunakan untuk nama bidang.
  - Jenis: Teks, Default: `"_"`
- `valueTag`— Tag yang digunakan untuk nilai ketika ada atribut dalam elemen yang tidak memiliki anak.
  - Jenis: Teks, Default: `"_VALUE"`
- `ignoreSurroundingSpaces`- Menentukan apakah ruang putih yang mengelilingi nilai-nilai harus diabaikan.
  - Jenis: Boolean, Default: `false`
- `withSchema`— Berisi skema yang diharapkan, dalam situasi di mana Anda ingin mengganti skema yang disimpulkan. Jika Anda tidak menggunakan opsi ini, AWS Glue simpulkan skema dari data XHTML.
  - Jenis: Teks, Default: Tidak berlaku
  - Nilainya harus berupa objek JSON yang mewakili a`StructType`.

Tentukan skema XMLnya secara manual

Contoh skema XHTML manual

Ini adalah contoh penggunaan pilihan format `withSchema` untuk menentukan skema untuk data XML.

```
from awsglue.gluetypes import *

schema = StructType([
  Field("id", IntegerType()),
  Field("name", StringType()),
  Field("nested", StructType([
    Field("x", IntegerType()),
    Field("y", StringType()),
    Field("z", ChoiceType([IntegerType(), StringType()]))
  ]))
])
```

```
datasource0 = create_dynamic_frame_from_options(
    connection_type,
    connection_options={"paths": ["s3://xml_bucket/someprefix"]},
    format="xml",
    format_options={"withSchema": json.dumps(schema.jsonValue())},
    transformation_ctx = ""
)
```

Menggunakan format Avro di AWS Glue

AWSGlue mengambil data dari sumber dan menulis data ke target yang disimpan dan diangkut dalam berbagai format data. Jika data Anda disimpan atau diangkut dalam format data Avro, dokumen ini memperkenalkan fitur yang tersedia untuk menggunakan data Anda di Glue. AWS

AWSGlue mendukung menggunakan format Avro. Format ini adalah format data berbasis baris yang berorientasi kinerja. Untuk pengenalan format oleh otoritas standar lihat, Apache Avro 1.8.2 Dokumentasi.

Anda dapat menggunakan AWS Glue untuk membaca file Avro dari Amazon S3 dan dari sumber streaming serta menulis file Avro ke Amazon S3. Anda dapat membaca dan menulis `bzip2` dan `gzip` mengarsipkan yang berisi file Avro dari S3. Selain itu, Anda dapat menulis`deflate`,`snappy`, dan `xz` arsip yang berisi file Avro. Anda mengonfigurasi perilaku kompresi pada Parameter koneksi S3 alih-alih dalam konfigurasi yang dibahas di halaman ini.

Tabel berikut menunjukkan operasi AWS Glue umum mana yang mendukung opsi format Avro.

Baca	Tulis	Streaming dibaca	Kelompokkan file kecil	Bookmark tugas
Didukung	Didukung	Didukung *	Tidak didukung	Didukung

* Didukung dengan batasan. Untuk informasi selengkapnya, lihat the section called “Catatan dan batasan untuk sumber streaming Avro”.

Contoh: Baca file atau folder Avro dari S3

Prasyarat: Anda akan memerlukan jalur S3 (`s3path`) ke file atau folder Avro yang ingin Anda baca.

Konfigurasi: Dalam opsi fungsi Anda, tentukan`format="avro"`. Dalam Anda`connection_options`, gunakan `paths` kunci untuk menentukan`s3path`. Anda dapat

mengonfigurasi bagaimana pembaca berinteraksi dengan S3 di file. `connection_options` Untuk detailnya, lihat Opsi format data untuk input dan output ETL di GlueAWS:. the section called "Parameter koneksi S3" Anda dapat mengonfigurasi bagaimana pembaca menafsirkan file Avro di file Anda. `format_options` Untuk detailnya, lihat Referensi Konfigurasi Avro.

Skrip AWS Glue ETL berikut menunjukkan proses membaca file atau folder Avro dari S3:

Python

Untuk contoh ini, gunakan metode create_dynamic_frame.from_options.

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="avro"
)
```

Scala

Untuk contoh ini, gunakan operasi getSourceWithFormat.

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.GlueContext
import org.apache.spark.sql.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType="s3",
      format="avro",
      options=JsonOptions("""{"paths": ["s3://s3path"]}""")
    ).getDynamicFrame()
  }
```

Contoh: Tulis file dan folder Avro ke S3

Prasyarat: Anda akan memerlukan initialized DataFrame () atau ()`dataFrame`. DynamicFrame `dynamicFrame` Anda juga akan membutuhkan jalur output S3 yang Anda harapkan,`s3path`.

Konfigurasi: Dalam opsi fungsi Anda, tentukan`format="avro"`. Dalam Anda`connection_options`, gunakan `paths` kunci untuk menentukan Anda`s3path`. Anda selanjutnya dapat mengubah cara penulis berinteraksi dengan S3 di. `connection_options` Untuk detailnya, lihat Opsi format data untuk input dan output ETL di GlueAWS:. the section called "Parameter koneksi S3" Anda dapat mengubah cara penulis menafsirkan file Avro di file Anda. `format_options` Untuk detailnya, lihat Referensi Konfigurasi Avro.

Skrip AWS Glue ETL berikut menunjukkan proses penulisan file Avro atau folder ke S3.

Python

Untuk contoh ini, gunakan metode write_dynamic_frame.from_options.

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    format="avro",
    connection_options={
        "path": "s3://s3path"
    }
)
```

Scala

Untuk contoh ini, gunakan metode getSinkWithFormat.

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
```

```
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
      connectionType="s3",
      options=JsonOptions("""{"path": "s3://s3path"}"""),
      format="avro"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Referensi konfigurasi Avro

Anda dapat menggunakan `format_options` nilai-nilai berikut di mana pun pustaka AWS Glue menentukan`format="avro"`:

- `version` — Menentukan versi format pembaca/penulis Apache Avro yang didukung. Default-nya adalah "1.7". Anda dapat menentukan `format_options={"version": “1.8”}` untuk mengaktifkan jenis membaca dan menulis logis Avro. Untuk informasi lebih lanjut, lihat Spesifikasi Apache Avro 1.7.7 dan Spesifikasi Apache Avro 1.8.2.

  Konektor Apache Avro 1.8 mendukung konversi tipe logis berikut:

Untuk pembaca: tabel ini menunjukkan konversi antara tipe data Avro (tipe logis dan tipe primitif Avro) dan tipe AWS Glue `DynamicFrame` data untuk pembaca Avro 1.7 dan 1.8.

Tipe Data Avro: Tipe Logis	Tipe Data Avro: Jenis Primitif Avro	GlueDynamicFrame Tipe Data: Pembaca Avro 1.7	GlueDynamicFrame Tipe Data: Pembaca Avro 1.8
Decimal	byte	BINER	Decimal
Decimal	tetap	BINER	Decimal
Tanggal	int	INT	Tanggal
Waktu (milidetik)	int	INT	INT

Tipe Data Avro: Tipe Logis	Tipe Data Avro: Jenis Primitif Avro	GlueDynamicFrame Tipe Data: Pembaca Avro 1.7	GlueDynamicFrame Tipe Data: Pembaca Avro 1.8
Waktu (mikrodetik)	long	LONG	LONG
Timestamp (milidetik)	long	LONG	Timestamp
Timestamp (mikrodetik)	long	LONG	LONG
Durasi (bukan tipe logis)	tetap 12	BINER	BINER

Untuk penulis: tabel ini menunjukkan konversi antara tipe AWS Glue `DynamicFrame` data dan tipe data Avro untuk Avro writer 1.7 dan 1.8.

AWS Glue**`DynamicFrame`** Tipe Data	Tipe Data Avro: Penulis Avro 1.7	Tipe Data Avro: Penulis Avro 1.8
Decimal	String	decimal
Tanggal	String	tanggal
Timestamp	String	timestamp-micros

Dukungan Avro Spark DataFrame

Untuk menggunakan Avro dari Spark DataFrame API, Anda perlu menginstal plugin Spark Avro untuk versi Spark yang sesuai. Versi Spark yang tersedia di pekerjaan Anda ditentukan oleh versi AWS Glue Anda. Untuk informasi selengkapnya tentang versi Spark, lihatthe section called “Versi AWS Glue”. Plugin ini dikelola oleh Apache, kami tidak membuat jaminan dukungan khusus.

Di AWS Glue 2.0 - gunakan versi 2.4.3 dari plugin Spark Avro. Anda dapat menemukan JAR ini di Maven Central, lihat org.apache.spark:spark-avro_2. 12:2.4 .3.

Di AWS Glue 3.0 - gunakan versi 3.1.1 dari plugin Spark Avro. Anda dapat menemukan JAR ini di Maven Central, lihat org.apache.spark:spark-avro_2. 12:3.1 .1.

Untuk memasukkan JAR tambahan dalam pekerjaan AWS Glue ETL, gunakan parameter `--extra-jars` pekerjaan. Untuk informasi selengkapnya tentang parameter pekerjaan, lihatthe section called "Parameter Tugas". Anda juga dapat mengonfigurasi parameter ini di fileAWS Management Console.

Menggunakan format GrokLog di Glue AWS

AWSGlue mengambil data dari sumber dan menulis data ke target yang disimpan dan diangkut dalam berbagai format data. Jika data Anda disimpan atau diangkut dalam format plaintext yang terstruktur secara longgar, dokumen ini memperkenalkan fitur yang tersedia untuk menggunakan data Anda dalam pola Glue AWS through Grok.

AWSGlue mendukung menggunakan pola Grok. Pola Grok mirip dengan grup pengambilan ekspresi reguler. Mereka mengenali pola urutan karakter dalam file plaintext dan memberi mereka jenis dan tujuan. Di AWS Glue, tujuan utamanya adalah membaca log. Untuk pengenalan Grok oleh penulis, lihat Referensi Logstash: Plugin filter Grok.

Baca	Tulis	Streaming dibaca	Kelompokkan file kecil	Bookmark tugas
Didukung	Tidak Berlaku	Didukung	Didukung	Tidak didukung

Referensi konfigurasi GrokLog

Anda dapat menggunakan nilai `format_options` berikut dengan `format="grokLog"`:

- `logFormat` — Menentukan pola Grok yang cocok dengan format log.
- `customPatterns` — Menentukan pola Grok tambahan yang digunakan di sini.
- `MISSING` — Menentukan sinyal untuk digunakan dalam mengidentifikasi nilai-nilai yang hilang. Defaultnya adalah `'-'`.
- `LineCount` — Menentukan jumlah baris dalam setiap catatan log. Default-nya adalah `'1'`, dan saat ini hanya catatan baris-tunggal yang didukung.
- `StrictMode` — Nilai Boolean yang menentukan apakah mode ketat diaktifkan. Dalam mode ketat, pembaca tidak melakukan konversi jenis otomatis atau pemulihan. Nilai default-nya adalah `"false"`.

Menggunakan format Ion di AWS Glue

AWSGlue mengambil data dari sumber dan menulis data ke target yang disimpan dan diangkut dalam berbagai format data. Jika data Anda disimpan atau diangkut dalam format data Ion, dokumen ini memperkenalkan fitur yang tersedia untuk menggunakan data Anda di GlueAWS.

AWSGlue mendukung menggunakan format Ion. Format ini mewakili struktur data (yang tidak berbasis baris atau kolom) dalam representasi biner dan teks biasa yang dapat dipertukarkan. Untuk pengenalan format oleh penulis, lihat Amazon Ion. (Untuk informasi lebih lanjut, lihat Spesifikasi Amazon Ion.)

Anda dapat menggunakan AWS Glue untuk membaca file Ion dari Amazon S3. Anda dapat membaca `bzip` dan `gzip` mengarsipkan yang berisi file Ion dari S3. Anda mengonfigurasi perilaku kompresi pada Parameter koneksi S3 alih-alih dalam konfigurasi yang dibahas di halaman ini.

Tabel berikut menunjukkan operasi AWS Glue umum mana yang mendukung opsi format Ion.

Baca	Tulis	Streaming dibaca	Kelompokkan file kecil	Bookmark tugas
Didukung	Tidak didukung	Tidak didukung	Didukung	Tidak didukung

Contoh: Baca file dan folder Ion dari S3

Prasyarat: Anda akan memerlukan jalur S3 (`s3path`) ke file Ion atau folder yang ingin Anda baca.

Konfigurasi: Dalam opsi fungsi Anda, tentukan`format="json"`. Dalam Anda`connection_options`, gunakan `paths` kunci untuk menentukan Anda`s3path`. Anda dapat mengonfigurasi bagaimana pembaca berinteraksi dengan S3 di file. `connection_options` Untuk detailnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue:the section called “Parameter koneksi S3”.

Skrip AWS Glue ETL berikut menunjukkan proses membaca file Ion atau folder dari S3:

Python

Untuk contoh ini, gunakan metode create_dynamic_frame.from_options.

```
# Example: Read ION from S3
```

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="ion"
)
```

Scala

Untuk contoh ini, gunakan operasi getSourceWithFormat.

```
// Example: Read ION from S3

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.GlueContext
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType="s3",
      format="ion",
      options=JsonOptions("""{"paths": ["s3://s3path"], "recurse": true}""")
    ).getDynamicFrame()
  }
}
```

Referensi konfigurasi ion

Tidak ada nilai `format_options` untuk `format="ion"`.

Menggunakan format JSON di AWS Glue

AWS Glue mengambil data dari sumber dan menulis data ke target yang disimpan dan diangkut dalam berbagai format data. Jika data Anda disimpan atau diangkut dalam format data JSON,

dokumen ini memperkenalkan Anda ke fitur yang tersedia untuk menggunakan data Anda di Glue. AWS

AWS Glue mendukung menggunakan format JSON. Format ini mewakili struktur data dengan bentuk yang konsisten tetapi konten fleksibel, yang tidak berbasis baris atau kolom. JSON didefinisikan oleh standar paralel yang dikeluarkan oleh beberapa otoritas, salah satunya adalah ECMA-404. Untuk pengenalan format oleh sumber yang sering direferensikan, lihat Memperkenalkan JSON.

Anda dapat menggunakan AWS Glue untuk membaca file JSON dari Amazon S3, `bzip` serta file JSON `gzip` terkompresi. Anda mengonfigurasi perilaku kompresi pada Parameter koneksi S3 alih-alih dalam konfigurasi yang dibahas di halaman ini.

Baca	Tulis	Streaming dibaca	Kelompokkan file kecil	Bookmark tugas	
Didukung	Didukung	Didukung	Didukung	Didukung	

Contoh: Baca file atau folder JSON dari S3

Prasyarat: Anda akan memerlukan jalur S3 (`s3path`) ke file atau folder JSON yang ingin Anda baca.

Konfigurasi: Dalam opsi fungsi Anda, tentukan`format="json"`. Dalam Anda`connection_options`, gunakan `paths` kunci untuk menentukan Anda`s3path`. Anda dapat lebih lanjut mengubah bagaimana operasi baca Anda akan melintasi s3 dalam opsi koneksi, berkonsultasi untuk detailnya. the section called “Parameter koneksi S3” Anda dapat mengonfigurasi bagaimana pembaca menafsirkan file JSON di file Anda. `format_options` Untuk detailnya, lihat Referensi Konfigurasi JSON.

Berikut skrip AWS Glue ETL menunjukkan proses membaca file JSON atau folder dari S3:

Python

Untuk contoh ini, gunakan metode create_dynamic_frame.from_options.

```
# Example: Read JSON from S3
# For show, we handle a nested JSON file that we can limit with the JsonPath
 parameter
# For show, we also handle a JSON where a single entry spans multiple lines
# Consider whether optimizePerformance is right for your workflow.
```

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)
spark = glueContext.spark_session

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="json",
    format_options={
        "jsonPath": "$.id",
        "multiline": True,
        # "optimizePerformance": True, -> not compatible with jsonPath, multiline
    }
)
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
dataFrame = spark.read\
    .option("multiLine", "true")\
    .json("s3://s3path")
```

Scala

Untuk contoh ini, gunakan operasi getSourceWithFormat.

```
// Example: Read JSON from S3
// For show, we handle a nested JSON file that we can limit with the JsonPath
 parameter
// For show, we also handle a JSON where a single entry spans multiple lines
// Consider whether optimizePerformance is right for your workflow.

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
```

```
    val dynamicFrame = glueContext.getSourceWithFormat(
      formatOptions=JsonOptions("""{"jsonPath": "$.id", "multiline": true,
 "optimizePerformance":false}"""),
      connectionType="s3",
      format="json",
      options=JsonOptions("""{"paths": ["s3://s3path"], "recurse": true}""")
    ).getDynamicFrame()
  }
}
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
val dataFrame = spark.read
    .option("multiLine", "true")
    .json("s3://s3path")
```

Contoh: Tulis file dan folder JSON ke S3

Prasyarat: Anda akan memerlukan initialized DataFrame () atau ()`dataFrame`. DynamicFrame `dynamicFrame` Anda juga akan membutuhkan jalur output S3 yang Anda harapkan,`s3path`.

Konfigurasi: Dalam opsi fungsi Anda, tentukan`format="json"`. Dalam Anda`connection_options`, gunakan `paths` kunci untuk menentukan`s3path`. Anda selanjutnya dapat mengubah cara penulis berinteraksi dengan S3 di. `connection_options` Untuk detailnya, lihat Opsi format data untuk input dan output ETL di Glue AWS :. the section called “Parameter koneksi S3” Anda dapat mengonfigurasi bagaimana penulis menafsirkan file JSON di file Anda. `format_options` Untuk detailnya, lihat Referensi Konfigurasi JSON.

Berikut skrip AWS Glue ETL menunjukkan proses penulisan file JSON atau folder dari S3:

Python

Untuk contoh ini, gunakan metode write_dynamic_frame.from_options.

```
# Example: Write JSON to S3

from pyspark.context import SparkContext
from awsglue.context import GlueContext
```

```
sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    connection_options={"path": "s3://s3path"},
    format="json"
)
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
df.write.json("s3://s3path/")
```

Scala

Untuk contoh ini, gunakan metode getSinkWithFormat.

```
// Example: Write JSON to S3

import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
        connectionType="s3",
        options=JsonOptions("""{"path": "s3://s3path"}"""),
        format="json"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
df.write.json("s3://s3path")
```

Referensi konfigurasi Json

Anda dapat menggunakan nilai `format_options` berikut dengan `format="json"`:

- `jsonPath`— JsonPathEkspresi yang mengidentifikasi objek yang akan dibaca ke dalam catatan. Hal ini sangat berguna ketika sebuah file berisi catatan bersarang di dalam array luar. Misalnya, JsonPath ekspresi berikut menargetkan `id` bidang objek JSON.

  ```
  format="json", format_options={"jsonPath": "$.id"}
  ```

- `multiLine` — Nilai Boolean yang menentukan apakah satu catatan dapat memiliki panjang hingga beberapa baris. Hal ini dapat terjadi ketika bidang berisi karakter baris baru yang dikutip. Anda harus menetapkan opsi ini ke `"true"` jika ada catatan yang mencakup beberapa baris. Nilai default-nya adalah `"false"`, yang memungkinkan untuk pemecahan file yang lebih agresif selama penguraian.
- `optimizePerformance`— Nilai Boolean yang menentukan apakah akan menggunakan pembaca SIMD JSON tingkat lanjut bersama dengan format memori kolumnar berbasis Apache Arrow. Hanya tersedia di AWS Glue 3.0. Tidak kompatibel dengan `multiLine` atau`jsonPath`. Menyediakan salah satu dari opsi tersebut akan menginstruksikan AWS Glue untuk kembali ke pembaca standar.
- `withSchema`— Nilai String yang menentukan skema tabel dalam format yang dijelaskan dalam. the section called “Tentukan skema XHTML” Hanya digunakan dengan `optimizePerformance` saat membaca dari koneksi non-Katalog.

Menggunakan pembaca SIMD JSON vektor dengan format kolom Apache Arrow

AWS Glueversi 3.0 menambahkan pembaca vektor untuk data JSON. Ini melakukan 2x lebih cepat dalam kondisi tertentu, dibandingkan dengan pembaca standar. Pembaca ini dilengkapi dengan batasan tertentu yang harus diperhatikan pengguna sebelum digunakan, didokumentasikan di bagian ini.

Untuk menggunakan pembaca yang dioptimalkan, atur `"optimizePerformance"` ke True di properti `format_options` or table. Anda juga perlu menyediakan `withSchema` kecuali membaca dari katalog. `withSchema`mengharapkan masukan seperti yang dijelaskan dalam the section called “Tentukan skema XHTML”

```
// Read from S3 data source
```

```
glueContext.create_dynamic_frame.from_options(
    connection_type = "s3",
    connection_options = {"paths": ["s3://s3path"]},
    format = "json",
    format_options={
        "optimizePerformance": True,
        "withSchema": SchemaString
        })

// Read from catalog table
glueContext.create_dynamic_frame.from_catalog(
    database = database,
    table_name = table,
    additional_options = {
    // The vectorized reader for JSON can read your schema from a catalog table
 property.
        "optimizePerformance": True,
        })
```

Untuk informasi lebih lanjut tentang bangunan a *SchemaString*di perpustakaan AWS Glue, lihatthe section called “Tipe”.

Keterbatasan untuk pembaca CSV vektor

Perhatikan batasan berikut:

- Elemen JSON dengan objek bersarang atau nilai array tidak didukung. Jika disediakan, AWS Glue akan kembali ke pembaca standar.
- Skema harus disediakan, baik dari Katalog atau dengan`withSchema`.
- Tidak kompatibel dengan `multiLine` atau`jsonPath`. Menyediakan salah satu dari opsi tersebut akan menginstruksikan AWS Glue untuk kembali ke pembaca standar.
- Menyediakan catatan masukan yang tidak sesuai dengan skema masukan akan menyebabkan pembaca gagal.
- Catatan kesalahan tidak akan dibuat.
- File JSON dengan karakter multi-byte (seperti karakter Jepang atau China) tidak didukung.

Menggunakan format ORC di AWS Glue

AWSGlue mengambil data dari sumber dan menulis data ke target yang disimpan dan diangkut dalam berbagai format data. Jika data Anda disimpan atau diangkut dalam format data ORC, dokumen ini memperkenalkan fitur yang tersedia untuk menggunakan data Anda di Glue. AWS

AWSGlue mendukung menggunakan format ORC. Format ini adalah format data berbasis kolom yang berorientasi kinerja. Untuk pengenalan format oleh otoritas standar lihat, Apache Orc.

Anda dapat menggunakan AWS Glue untuk membaca file ORC dari Amazon S3 dan dari sumber streaming serta menulis file ORC ke Amazon S3. Anda dapat membaca dan menulis `bzip` dan `gzip` mengarsipkan yang berisi file ORC dari S3. Anda mengonfigurasi perilaku kompresi pada Parameter koneksi S3 alih-alih dalam konfigurasi yang dibahas di halaman ini.

Tabel berikut menunjukkan operasi AWS Glue umum mana yang mendukung opsi format ORC.

Baca	Tulis	Streaming dibaca	Kelompokkan file kecil	Bookmark tugas
Didukung	Didukung	Didukung	Tidak didukung	Didukung *

* Didukung dalam AWS Glue versi 1.0+

Contoh: Baca file atau folder ORC dari S3

Prasyarat: Anda akan memerlukan jalur S3 (`s3path`) ke file atau folder ORC yang ingin Anda baca.

Konfigurasi: Dalam opsi fungsi Anda, tentukan`format="orc"`. Dalam Anda`connection_options`, gunakan `paths` kunci untuk menentukan Anda`s3path`. Anda dapat mengonfigurasi bagaimana pembaca berinteraksi dengan S3 di file. `connection_options` Untuk detailnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue:the section called “Parameter koneksi S3”.

Berikut skrip AWS Glue ETL menunjukkan proses membaca file ORC atau folder dari S3:

Python

Untuk contoh ini, gunakan metode create_dynamic_frame.from_options.

```
from pyspark.context import SparkContext
```

```
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

dynamicFrame = glueContext.create_dynamic_frame.from_options(
    connection_type="s3",
    connection_options={"paths": ["s3://s3path"]},
    format="orc"
)
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
dataFrame = spark.read\
    .orc("s3://s3path")
```

Scala

Untuk contoh ini, gunakan operasi getSourceWithFormat.

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.GlueContext
import org.apache.spark.sql.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dynamicFrame = glueContext.getSourceWithFormat(
      connectionType="s3",
      format="orc",
      options=JsonOptions("""{"paths": ["s3://s3path"]}""")
    ).getDynamicFrame()
  }
}
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
val dataFrame = spark.read
    .orc("s3://s3path")
```

Contoh: Tulis file dan folder ORC ke S3

Prasyarat: Anda akan memerlukan initialized DataFrame () atau ()`dataFrame`. DynamicFrame `dynamicFrame` Anda juga akan membutuhkan jalur output S3 yang Anda harapkan,`s3path`.

Konfigurasi: Dalam opsi fungsi Anda, tentukan`format="orc"`. Dalam opsi koneksi Anda, gunakan `paths` kunci untuk menentukan`s3path`. Anda selanjutnya dapat mengubah cara penulis berinteraksi dengan S3 di. `connection_options` Untuk detailnya, lihat Opsi format data untuk input dan output ETL di GlueAWS:. the section called “Parameter koneksi S3” Contoh kode berikut menunjukkan proses:

Python

Untuk contoh ini, gunakan metode write_dynamic_frame.from_options.

```
from pyspark.context import SparkContext
from awsglue.context import GlueContext

sc = SparkContext.getOrCreate()
glueContext = GlueContext(sc)

glueContext.write_dynamic_frame.from_options(
    frame=dynamicFrame,
    connection_type="s3",
    format="orc",
    connection_options={
        "path": "s3://s3path"
    }
)
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
df.write.orc("s3://s3path/")
```

Scala

Untuk contoh ini, gunakan metode getSinkWithFormat.

```
import com.amazonaws.services.glue.util.JsonOptions
import com.amazonaws.services.glue.{DynamicFrame, GlueContext}
import org.apache.spark.SparkContext
```

```
object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    glueContext.getSinkWithFormat(
      connectionType="s3",
      options=JsonOptions("""{"path": "s3://s3path"}"""),
      format="orc"
    ).writeDynamicFrame(dynamicFrame)
  }
}
```

Anda juga dapat menggunakan DataFrames dalam script (`pyspark.sql.DataFrame`).

```
df.write.orc("s3://s3path/")
```

Referensi konfigurasi ORC

Tidak ada nilai `format_options` untuk `format="orc"`. Namun demikian, setiap pilihan yang diterima oleh kode SparkSQL yang mendasari dapat diteruskan ke sana dengan cara parameter peta `connection_options`.

Menggunakan kerangka kerja data lake dengan pekerjaan AWS Glue ETL

Kerangka kerja data lake sumber terbuka menyederhanakan pemrosesan data tambahan untuk file yang Anda simpan di danau data yang dibangun di Amazon S3. AWS Glue3.0 dan yang lebih baru mendukung kerangka kerja data lake sumber terbuka berikut:

- Apache Hudi
- Yayasan Linux Delta Lake
- Gunung Es Apache

Kami menyediakan dukungan asli untuk kerangka kerja ini sehingga Anda dapat membaca dan menulis data yang Anda simpan di Amazon S3 dengan cara yang konsisten secara transaksional. Tidak perlu menginstal konektor terpisah atau menyelesaikan langkah konfigurasi tambahan untuk menggunakan kerangka kerja ini dalam pekerjaan AWS Glue ETL.

Saat Anda mengelola kumpulan data melaluiAWS Glue Data Catalog, Anda dapat menggunakan AWS Glue metode untuk membaca dan menulis tabel danau data dengan Spark. DataFrames Anda juga dapat membaca dan menulis data Amazon S3 menggunakan Spark API DataFrame .

Dalam video ini, Anda dapat mempelajari dasar-dasar cara kerja Apache Hudi, Apache Iceberg, dan Delta Lake. Anda akan melihat cara menyisipkan, memperbarui, dan menghapus data di danau data Anda dan cara kerja masing-masing kerangka kerja ini.

Topik

Batasan

Pertimbangkan batasan berikut sebelum Anda menggunakan kerangka kerja data lake dengan AWS Glue.

- AWS Glue `GlueContext`Metode berikut untuk DynamicFrame tidak mendukung membaca dan menulis tabel kerangka data lake. Gunakan `GlueContext` metode untuk DataFrame atau Spark DataFrame API sebagai gantinya.
  - `GlueContext`Metode berikut untuk tidak DynamicFrame didukung dengan kontrol izin Lake Formation:
    - `create_dynamic_frame.from_catalog`
    - `write_dynamic_frame.from_catalog`
    - `getDynamicFrame`
    - `writeDynamicFrame`
  - `GlueContext`Metode berikut ini DataFrame didukung dengan kontrol izin Lake Formation:
    - `create_data_frame.from_catalog`
    - `write_data_frame.from_catalog`
    - `getDataFrame`
    - `writeDataFrame`
- Pengelompokan file kecil tidak didukung.

- Bookmark Job tidak didukung.
- Apache Hudi 0.10.1 untuk AWS Glue 3.0 tidak mendukung tabel Hudi Merge on Read (MoR).
- `ALTER TABLE … RENAME TO`tidak tersedia untuk Apache Iceberg 0.13.1 untuk 3.0. AWS Glue

Batasan untuk tabel format danau data yang dikelola oleh izin Lake Formation

Format data lake terintegrasi dengan AWS Glue ETL melalui izin Lake Formation. Membuat DynamicFrame penggunaan `create_dynamic_frame` tidak didukung. Untuk informasi selengkapnya, lihat contoh berikut ini:

- Contoh: Membaca dan menulis tabel Gunung Es dengan kontrol izin Lake Formation
- Contoh: Membaca dan menulis tabel Hudi dengan kontrol izin Lake Formation
- Contoh: Membaca dan menulis tabel Delta Lake dengan kontrol izin Lake Formation

> **Note**
>
> Integrasi dengan AWS Glue ETL melalui izin Lake Formation untuk Apache Hudi, Apache Iceberg, dan Delta Lake hanya didukung di versi 4.0. AWS Glue

Apache Iceberg memiliki integrasi terbaik dengan AWS Glue ETL melalui izin Lake Formation. Ini mendukung hampir semua operasi dan termasuk dukungan SQL.

Hudi mendukung sebagian besar operasi dasar dengan pengecualian operasi administratif. Hal ini karena pilihan ini umumnya dilakukan melalui penulisan dataframes dan ditentukan melalui. `additional_options` Anda perlu menggunakan AWS Glue API DataFrames untuk membuat operasi Anda karena SparkSQL tidak didukung.

Delta Lake hanya mendukung pembacaan dan penambahan dan penimpaan data tabel. Delta Lake membutuhkan penggunaan perpustakaan mereka sendiri untuk dapat melakukan berbagai tugas seperti pembaruan.

Fitur berikut tidak tersedia untuk tabel Iceberg yang dikelola oleh izin Lake Formation.

- Pemadatan menggunakan AWS Glue ETL
- Dukungan Spark SQL melalui ETL AWS Glue

Berikut ini adalah batasan tabel Hudi yang dikelola oleh izin Lake Formation:

- Penghapusan file yatim piatu

Berikut ini adalah batasan tabel Delta Lake yang dikelola oleh izin Lake Formation:

- Semua fitur selain memasukkan dan membaca dari tabel Delta Lake.

Menggunakan kerangka Hudi di Glue AWS

AWSGlue 3.0 dan yang lebih baru mendukung kerangka Apache Hudi untuk data lake. Hudi adalah kerangka penyimpanan danau data open-source yang menyederhanakan pemrosesan data tambahan dan pengembangan pipa data. Topik ini mencakup fitur yang tersedia untuk menggunakan data Anda di AWS Glue saat Anda mengangkut atau menyimpan data Anda dalam tabel Hudi. Untuk mempelajari lebih lanjut tentang Hudi, lihat dokumentasi resmi Apache Hudi.

Anda dapat menggunakan AWS Glue untuk melakukan operasi baca dan tulis pada tabel Hudi di Amazon S3, atau bekerja dengan tabel Hudi menggunakan Katalog Data AWS Glue. Operasi tambahan termasuk insert, update, dan semua operasi Apache Spark juga didukung.

> **Note**
>
> Apache Hudi 0.10.1 untuk AWS Glue 3.0 tidak mendukung tabel Hudi Merge on Read (MoR).

Tabel berikut mencantumkan versi Hudi yang disertakan dalam setiap versi AWS Glue.

AWSVersi Glue	Versi Hudi yang didukung
4.0	0.12.1
3.0	0.10.1

Untuk mempelajari lebih lanjut tentang framework data lake yang didukung AWS Glue, lihatMenggunakan kerangka kerja data lake dengan pekerjaan AWS Glue ETL.

Mengaktifkan Hudi

Untuk mengaktifkan Hudi for AWS Glue, selesaikan tugas berikut:

- Tentukan `hudi` sebagai nilai untuk parameter `--datalake-formats` pekerjaan. Untuk informasi selengkapnya, lihat AWS Glueparameter pekerjaan.
- Buat kunci bernama `--conf` untuk pekerjaan AWS Glue Anda, dan atur ke nilai berikut. Atau, Anda dapat mengatur konfigurasi berikut menggunakan `SparkConf` skrip Anda. Pengaturan ini membantu Apache Spark menangani tabel Hudi dengan benar.

```
spark.serializer=org.apache.spark.serializer.KryoSerializer --conf
 spark.sql.hive.convertMetastoreParquet=false
```

- Dukungan izin Lake Formation untuk Hudi diaktifkan secara default untuk AWS Glue 4.0. Tidak diperlukan konfigurasi tambahan untuk membaca/menulis ke tabel Hudi yang terdaftar di Lake Formation. Untuk membaca tabel Hudi yang terdaftar, peran IAM AWS Glue job harus memiliki izin SELECT. Untuk menulis ke tabel Hudi terdaftar, peran IAM pekerjaan AWS Glue harus memiliki izin SUPER. Untuk mempelajari lebih lanjut tentang mengelola izin Lake Formation, lihat Memberikan dan mencabut izin pada sumber daya Katalog Data.

Menggunakan versi Hudi yang berbeda

Untuk menggunakan versi Hudi yang tidak didukung AWS Glue, tentukan file Hudi JAR Anda sendiri menggunakan parameter `--extra-jars` pekerjaan. Jangan sertakan `hudi` sebagai nilai untuk parameter `--datalake-formats` pekerjaan.

Contoh: Tulis tabel Hudi ke Amazon S3 dan daftarkan di Katalog Data AWS Glue

Contoh skrip ini menunjukkan cara menulis tabel Hudi ke Amazon S3 dan mendaftarkan tabel ke Katalog Data AWS Glue. Contoh menggunakan alat Hudi Hive Sync untuk mendaftarkan tabel.

> **Note**
>
> Contoh ini mengharuskan Anda untuk mengatur parameter `--enable-glue-datacatalog` pekerjaan untuk menggunakan Katalog Data AWS Glue sebagai metastore Apache Spark Hive. Untuk mempelajari selengkapnya, lihat AWS Glueparameter pekerjaan.

Python

```
# Example: Create a Hudi table from a DataFrame
# and register the table to Glue Data Catalog
```

```
additional_options={
    "hoodie.table.name": "<your_table_name>",
    "hoodie.datasource.write.storage.type": "COPY_ON_WRITE",
    "hoodie.datasource.write.operation": "upsert",
    "hoodie.datasource.write.recordkey.field": "<your_recordkey_field>",
    "hoodie.datasource.write.precombine.field": "<your_precombine_field>",
    "hoodie.datasource.write.partitionpath.field": "<your_partitionkey_field>",
    "hoodie.datasource.write.hive_style_partitioning": "true",
    "hoodie.datasource.hive_sync.enable": "true",
    "hoodie.datasource.hive_sync.database": "<your_database_name>",
    "hoodie.datasource.hive_sync.table": "<your_table_name>",
    "hoodie.datasource.hive_sync.partition_fields": "<your_partitionkey_field>",
    "hoodie.datasource.hive_sync.partition_extractor_class":
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
    "hoodie.datasource.hive_sync.use_jdbc": "false",
    "hoodie.datasource.hive_sync.mode": "hms",
    "path": "s3://<s3Path/>"
}

dataFrame.write.format("hudi") \
    .options(**additional_options) \
    .mode("overwrite") \
    .save()
```

Scala

```
// Example: Example: Create a Hudi table from a DataFrame
// and register the table to Glue Data Catalog

val additionalOptions = Map(
  "hoodie.table.name" -> "<your_table_name>",
  "hoodie.datasource.write.storage.type" -> "COPY_ON_WRITE",
  "hoodie.datasource.write.operation" -> "upsert",
  "hoodie.datasource.write.recordkey.field" -> "<your_recordkey_field>",
  "hoodie.datasource.write.precombine.field" -> "<your_precombine_field>",
  "hoodie.datasource.write.partitionpath.field" -> "<your_partitionkey_field>",
  "hoodie.datasource.write.hive_style_partitioning" -> "true",
  "hoodie.datasource.hive_sync.enable" -> "true",
  "hoodie.datasource.hive_sync.database" -> "<your_database_name>",
  "hoodie.datasource.hive_sync.table" -> "<your_table_name>",
  "hoodie.datasource.hive_sync.partition_fields" -> "<your_partitionkey_field>",
  "hoodie.datasource.hive_sync.partition_extractor_class" ->
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
```

```
  "hoodie.datasource.hive_sync.use_jdbc" -> "false",
  "hoodie.datasource.hive_sync.mode" -> "hms",
  "path" -> "s3://<s3Path/>")

dataFrame.write.format("hudi")
  .options(additionalOptions)
  .mode("append")
  .save()
```

Contoh: Membaca tabel Hudi dari Amazon S3 menggunakan Katalog Data AWS Glue

Contoh ini membaca tabel Hudi yang Anda buat Contoh: Tulis tabel Hudi ke Amazon S3 dan daftarkan di Katalog Data AWS Glue dari Amazon S3.

> **Note**
>
> Contoh ini mengharuskan Anda untuk mengatur parameter `--enable-glue-datacatalog` pekerjaan untuk menggunakan Katalog Data AWS Glue sebagai metastore Apache Spark Hive. Untuk mempelajari selengkapnya, lihat AWS Glueparameter pekerjaan.

Python

Untuk contoh ini, gunakan `GlueContext.create_data_frame.from_catalog()` metode ini.

```
# Example: Read a Hudi table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

dataFrame = glueContext.create_data_frame.from_catalog(
    database = "<your_database_name>",
    table_name = "<your_table_name>"
)
```

Scala

Untuk contoh ini, gunakan getCatalogSourcemetode ini.

```
// Example: Read a Hudi table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apache.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)

    val dataFrame = glueContext.getCatalogSource(
      database = "<your_database_name>",
      tableName = "<your_table_name>"
    ).getDataFrame()
  }
}
```

Contoh: Perbarui dan masukkan **`DataFrame`** ke dalam tabel Hudi di Amazon S3

Contoh ini menggunakan AWS Glue Data Catalog untuk menyisipkan DataFrame ke dalam tabel Hudi yang Anda buat. Contoh: Tulis tabel Hudi ke Amazon S3 dan daftarkan di Katalog Data AWS Glue

> **Note**
>
> Contoh ini mengharuskan Anda untuk mengatur parameter `--enable-glue-datacatalog` pekerjaan untuk menggunakan Katalog Data AWS Glue sebagai metastore Apache Spark Hive. Untuk mempelajari selengkapnya, lihat AWS Glueparameter pekerjaan.

Python

Untuk contoh ini, gunakan `GlueContext.write_data_frame.from_catalog()` metode ini.

```
# Example: Upsert a Hudi table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)
```

```
glueContext.write_data_frame.from_catalog(
    frame = dataFrame,
    database = "<your_database_name>",
    table_name = "<your_table_name>",
    additional_options={
        "hoodie.table.name": "<your_table_name>",
        "hoodie.datasource.write.storage.type": "COPY_ON_WRITE",
        "hoodie.datasource.write.operation": "upsert",
        "hoodie.datasource.write.recordkey.field": "<your_recordkey_field>",
        "hoodie.datasource.write.precombine.field": "<your_precombine_field>",
        "hoodie.datasource.write.partitionpath.field": "<your_partitionkey_field>",
        "hoodie.datasource.write.hive_style_partitioning": "true",
        "hoodie.datasource.hive_sync.enable": "true",
        "hoodie.datasource.hive_sync.database": "<your_database_name>",
        "hoodie.datasource.hive_sync.table": "<your_table_name>",
        "hoodie.datasource.hive_sync.partition_fields": "<your_partitionkey_field>",
        "hoodie.datasource.hive_sync.partition_extractor_class":
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
        "hoodie.datasource.hive_sync.use_jdbc": "false",
        "hoodie.datasource.hive_sync.mode": "hms"
    }
)
```

Scala

Untuk contoh ini, gunakan getCatalogSinkmetode ini.

```
// Example: Upsert a Hudi table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.util.JsonOptions
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    glueContext.getCatalogSink("<your_database_name>", "<your_table_name>",
      additionalOptions = JsonOptions(Map(
        "hoodie.table.name" -> "<your_table_name>",
        "hoodie.datasource.write.storage.type" -> "COPY_ON_WRITE",
        "hoodie.datasource.write.operation" -> "upsert",
        "hoodie.datasource.write.recordkey.field" -> "<your_recordkey_field>",
```

```
        "hoodie.datasource.write.precombine.field" -> "<your_precombine_field>",
        "hoodie.datasource.write.partitionpath.field" ->
 "<your_partitionkey_field>",
        "hoodie.datasource.write.hive_style_partitioning" -> "true",
        "hoodie.datasource.hive_sync.enable" -> "true",
        "hoodie.datasource.hive_sync.database" -> "<your_database_name>",
        "hoodie.datasource.hive_sync.table" -> "<your_table_name>",
        "hoodie.datasource.hive_sync.partition_fields" ->
 "<your_partitionkey_field>",
        "hoodie.datasource.hive_sync.partition_extractor_class" ->
 "org.apache.hudi.hive.MultiPartKeysValueExtractor",
        "hoodie.datasource.hive_sync.use_jdbc" -> "false",
        "hoodie.datasource.hive_sync.mode" -> "hms"
      )))
      .writeDataFrame(dataFrame, glueContext)
  }
}
```

Contoh: Baca Tabel Hudi dari Amazon S3 menggunakan Spark

Contoh ini membaca tabel Hudi dari Amazon S3 menggunakan Spark API. DataFrame

Python

```
# Example: Read a Hudi table from S3 using a Spark DataFrame

dataFrame = spark.read.format("hudi").load("s3://<s3path/>")
```

Scala

```
// Example: Read a Hudi table from S3 using a Spark DataFrame

val dataFrame = spark.read.format("hudi").load("s3://<s3path/>")
```

Contoh: Tabel Hudi ke Amazon S3 menggunakan Spark

Contoh ini menulis tabel Hudi ke Amazon S3 menggunakan Spark.

Python

```
# Example: Write a Hudi table to S3 using a Spark DataFrame
```

```
dataFrame.write.format("hudi") \
    .options(**additional_options) \
    .mode("overwrite") \
    .save("s3://<s3Path/>)
```

Scala

```
// Example: Write a Hudi table to S3 using a Spark DataFrame

dataFrame.write.format("hudi")
  .options(additionalOptions)
  .mode("overwrite")
  .save("s3://<s3path/>")
```

Contoh: Membaca dan menulis tabel Hudi dengan kontrol izin Lake Formation

Contoh ini membaca dan menulis tabel Hudi dengan kontrol izin Lake Formation.

1. Buat tabel Hudi dan daftarkan di Lake Formation.
   a. Untuk mengaktifkan kontrol izin Lake Formation, Anda harus terlebih dahulu mendaftarkan tabel jalur Amazon S3 di Lake Formation. Untuk informasi selengkapnya, lihat Mendaftarkan lokasi Amazon S3. Anda dapat mendaftarkannya baik dari konsol Lake Formation atau dengan menggunakan AWS CLI:

      ```
      aws lakeformation register-resource --resource-arn arn:aws:s3:::<s3-bucket>/<s3-
      folder> --use-service-linked-role --region <REGION>
      ```

      Setelah Anda mendaftarkan lokasi Amazon S3, tabel AWS Glue apa pun yang menunjuk ke lokasi (atau lokasi turunannya) akan mengembalikan nilai `IsRegisteredWithLakeFormation` parameter sebagai true dalam panggilan. `GetTable`

   b. Buat tabel Hudi yang menunjuk ke jalur Amazon S3 terdaftar melalui API kerangka data Spark:

      ```
      hudi_options = {
          'hoodie.table.name': table_name,
          'hoodie.datasource.write.storage.type': 'COPY_ON_WRITE',
          'hoodie.datasource.write.recordkey.field': 'product_id',
          'hoodie.datasource.write.table.name': table_name,
          'hoodie.datasource.write.operation': 'upsert',
      ```

```
    'hoodie.datasource.write.precombine.field': 'updated_at',
    'hoodie.datasource.write.hive_style_partitioning': 'true',
    'hoodie.upsert.shuffle.parallelism': 2,
    'hoodie.insert.shuffle.parallelism': 2,
    'path': <S3_TABLE_LOCATION>,
    'hoodie.datasource.hive_sync.enable': 'true',
    'hoodie.datasource.hive_sync.database': database_name,
    'hoodie.datasource.hive_sync.table': table_name,
    'hoodie.datasource.hive_sync.use_jdbc': 'false',
    'hoodie.datasource.hive_sync.mode': 'hms'
}

df_products.write.format("hudi")  \
    .options(**hudi_options)  \
    .mode("overwrite")  \
    .save()
```

2. Berikan izin Formasi Lake untuk peran IAM pekerjaan AWS Glue. Anda dapat memberikan izin dari konsol Lake Formation, atau menggunakan AWS CLI. Untuk informasi selengkapnya, lihat Memberikan izin tabel menggunakan konsol Lake Formation dan metode sumber daya bernama
3. Baca tabel Hudi yang terdaftar di Lake Formation. Kodenya sama dengan membaca tabel Hudi yang tidak terdaftar. Perhatikan bahwa peran IAM AWS Glue job harus memiliki izin SELECT agar pembacaan berhasil.

```
val dataFrame = glueContext.getCatalogSource(
     database = "<your_database_name>",
     tableName = "<your_table_name>"
   ).getDataFrame()
```

4. Tulis ke tabel Hudi yang terdaftar di Lake Formation. Kode ini sama dengan menulis ke tabel Hudi yang tidak terdaftar. Perhatikan bahwa peran IAM AWS Glue job harus memiliki izin SUPER agar penulisan berhasil.

```
glueContext.getCatalogSink("<your_database_name>", "<your_table_name>",
      additionalOptions = JsonOptions(Map(
        "hoodie.table.name" -> "<your_table_name>",
        "hoodie.datasource.write.storage.type" -> "COPY_ON_WRITE",
        "hoodie.datasource.write.operation" -> "<write_operation>",
        "hoodie.datasource.write.recordkey.field" -> "<your_recordkey_field>",
        "hoodie.datasource.write.precombine.field" -> "<your_precombine_field>",
        "hoodie.datasource.write.partitionpath.field" -> "<your_partitionkey_field>",
        "hoodie.datasource.write.hive_style_partitioning" -> "true",
```

```
        "hoodie.datasource.hive_sync.enable" -> "true",
        "hoodie.datasource.hive_sync.database" -> "<your_database_name>",
        "hoodie.datasource.hive_sync.table" -> "<your_table_name>",
        "hoodie.datasource.hive_sync.partition_fields" ->
"<your_partitionkey_field>",
        "hoodie.datasource.hive_sync.partition_extractor_class" ->
"org.apache.hudi.hive.MultiPartKeysValueExtractor",
        "hoodie.datasource.hive_sync.use_jdbc" -> "false",
        "hoodie.datasource.hive_sync.mode" -> "hms"
      )))
      .writeDataFrame(dataFrame, glueContext)
```

Menggunakan kerangka Delta Lake di AWS Glue

AWSGlue 3.0 dan yang lebih baru mendukung kerangka Linux Foundation Delta Lake. Delta Lake adalah kerangka penyimpanan data lake sumber terbuka yang membantu Anda melakukan transaksi ACID, menskalakan penanganan metadata, dan menyatukan streaming dan pemrosesan data batch. Topik ini mencakup fitur yang tersedia untuk menggunakan data Anda di AWS Glue saat Anda mengangkut atau menyimpan data Anda di tabel Delta Lake. Untuk mempelajari lebih lanjut tentang Danau Delta, lihat dokumentasi resmi Danau Delta.

Anda dapat menggunakan AWS Glue untuk melakukan operasi baca dan tulis pada tabel Delta Lake di Amazon S3, atau bekerja dengan tabel Delta Lake menggunakan AWS Glue Data Catalog. Operasi tambahan seperti insert, update, dan Table batch read and write juga didukung. Saat Anda menggunakan tabel Delta Lake, Anda juga memiliki opsi untuk menggunakan metode dari perpustakaan Delta Lake Python seperti. `DeltaTable.forPath` Untuk informasi lebih lanjut tentang perpustakaan Delta Lake Python, lihat dokumentasi Python Delta Lake.

Tabel berikut mencantumkan versi Delta Lake yang disertakan dalam setiap versi AWS Glue.

AWSVersi Glue	Versi Delta Lake yang didukung
4.0	2.1.0
3.0	1.0.0

Untuk mempelajari lebih lanjut tentang framework data lake yang didukung AWS Glue, lihatMenggunakan kerangka kerja data lake dengan pekerjaan AWS Glue ETL.

Mengaktifkan Delta Lake untuk Glue AWS

Untuk mengaktifkan Delta Lake for AWS Glue, selesaikan tugas-tugas berikut:

- Tentukan `delta` sebagai nilai untuk parameter `--datalake-formats` pekerjaan. Untuk informasi selengkapnya, lihat AWS Glueparameter pekerjaan.
- Buat kunci bernama `--conf` untuk pekerjaan AWS Glue Anda, dan atur ke nilai berikut. Atau, Anda dapat mengatur konfigurasi berikut menggunakan `SparkConf` skrip Anda. Pengaturan ini membantu Apache Spark menangani tabel Delta Lake dengan benar.

```
spark.sql.extensions=io.delta.sql.DeltaSparkSessionExtension --conf
 spark.sql.catalog.spark_catalog=org.apache.spark.sql.delta.catalog.DeltaCatalog --
conf
 spark.delta.logStore.class=org.apache.spark.sql.delta.storage.S3SingleDriverLogStore
```

- Dukungan izin Lake Formation untuk tabel Delta diaktifkan secara default untuk AWS Glue 4.0. Tidak diperlukan konfigurasi tambahan untuk membaca/menulis ke tabel Delta yang terdaftar di Lake Formation. Untuk membaca tabel Delta terdaftar, peran IAM AWS Glue job harus memiliki izin SELECT. Untuk menulis ke tabel Delta terdaftar, peran IAM AWS Glue job harus memiliki izin SUPER. Untuk mempelajari lebih lanjut tentang mengelola izin Lake Formation, lihat Memberikan dan mencabut izin pada sumber daya Katalog Data.

Menggunakan versi Delta Lake yang berbeda

Untuk menggunakan versi danau Delta yang tidak didukung AWS Glue, tentukan file JAR Delta Lake Anda sendiri menggunakan parameter `--extra-jars` pekerjaan. Jangan sertakan `delta` sebagai nilai untuk parameter `--datalake-formats` pekerjaan. Untuk menggunakan pustaka Delta Lake Python dalam kasus ini, Anda harus menentukan file JAR perpustakaan menggunakan parameter pekerjaan`--extra-py-files`. Pustaka Python dikemas dalam file JAR Delta Lake.

Contoh: Tulis tabel Delta Lake ke Amazon S3 dan daftarkan ke Katalog Data AWS Glue

Skrip AWS Glue ETL berikut menunjukkan cara menulis tabel Delta Lake ke Amazon S3 dan mendaftarkan tabel ke Katalog Data Glue. AWS

Python

```
# Example: Create a Delta Lake table from a DataFrame
# and register the table to Glue Data Catalog
```

```
additional_options = {
    "path": "s3://<s3Path>"
}
dataFrame.write \
    .format("delta") \
    .options(**additional_options) \
    .mode("append") \
    .partitionBy("<your_partitionkey_field>") \
    .saveAsTable("<your_database_name>.<your_table_name>")
```

Scala

```
// Example: Example: Create a Delta Lake table from a DataFrame
// and register the table to Glue Data Catalog

val additional_options = Map(
  "path" -> "s3://<s3Path>"
)
dataFrame.write.format("delta")
  .options(additional_options)
  .mode("append")
  .partitionBy("<your_partitionkey_field>")
  .saveAsTable("<your_database_name>.<your_table_name>")
```

Contoh: Baca tabel Delta Lake dari Amazon S3 menggunakan Katalog Data AWS Glue

Skrip AWS Glue ETL berikut membaca tabel Delta Lake yang Anda buat. Contoh: Tulis tabel Delta Lake ke Amazon S3 dan daftarkan ke Katalog Data AWS Glue

Python

Untuk contoh ini, gunakan metode create_data_frame.from_catalog.

```
# Example: Read a Delta Lake table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

df = glueContext.create_data_frame.from_catalog(
```

```
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Scala

Untuk contoh ini, gunakan getCatalogSourcemetode ini.

```
// Example: Read a Delta Lake table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val df = glueContext.getCatalogSource("<your_database_name>",
 "<your_table_name>",
      additionalOptions = additionalOptions)
      .getDataFrame()
  }
}
```

Contoh: Masukkan **`DataFrame`** ke dalam tabel Delta Lake di Amazon S3 menggunakan AWS Glue Data Catalog

Contoh ini menyisipkan data ke dalam tabel Delta Lake yang Anda buat. Contoh: Tulis tabel Delta Lake ke Amazon S3 dan daftarkan ke Katalog Data AWS Glue

> **Note**
>
> Contoh ini mengharuskan Anda untuk mengatur parameter `--enable-glue-datacatalog` pekerjaan untuk menggunakan Katalog Data AWS Glue sebagai metastore Apache Spark Hive. Untuk mempelajari selengkapnya, lihat AWS Glueparameter pekerjaan.

Python

Untuk contoh ini, gunakan metode write_data_frame.from_catalog.

```
# Example: Insert into a Delta Lake table in S3 using Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

glueContext.write_data_frame.from_catalog(
    frame=dataFrame,
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Scala

Untuk contoh ini, gunakan getCatalogSinkmetode ini.

```
// Example: Insert into a Delta Lake table in S3 using Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    glueContext.getCatalogSink("<your_database_name>", "<your_table_name>",
      additionalOptions = additionalOptions)
      .writeDataFrame(dataFrame, glueContext)
  }
}
```

Contoh: Membaca tabel Delta Lake dari Amazon S3 menggunakan Spark API

Contoh ini membaca tabel Delta Lake dari Amazon S3 menggunakan Spark API.

Python

```
# Example: Read a Delta Lake table from S3 using a Spark DataFrame
```

```
dataFrame = spark.read.format("delta").load("s3://<s3path/>")
```

Scala

```
// Example: Read a Delta Lake table from S3 using a Spark DataFrame

val dataFrame = spark.read.format("delta").load("s3://<s3path/>")
```

Contoh: Tulis tabel Delta Lake ke Amazon S3 menggunakan Spark

Contoh ini menulis tabel Delta Lake ke Amazon S3 menggunakan Spark.

Python

```
# Example: Write a Delta Lake table to S3 using a Spark DataFrame

dataFrame.write.format("delta") \
    .options(**additional_options) \
    .mode("overwrite") \
    .partitionBy("<your_partitionkey_field>")
    .save("s3://<s3Path>")
```

Scala

```
// Example: Write a Delta Lake table to S3 using a Spark DataFrame

dataFrame.write.format("delta")
  .options(additionalOptions)
  .mode("overwrite")
  .partitionBy("<your_partitionkey_field>")
  .save("s3://<s3path/>")
```

Contoh: Membaca dan menulis tabel Delta Lake dengan kontrol izin Lake Formation

Contoh ini membaca dan menulis tabel Danau Delta dengan kontrol izin Lake Formation.

1. Buat tabel Delta dan daftarkan di Lake Formation
   a. Untuk mengaktifkan kontrol izin Lake Formation, Anda harus terlebih dahulu mendaftarkan tabel jalur Amazon S3 di Lake Formation. Untuk informasi selengkapnya, lihat Mendaftarkan

lokasi Amazon S3. Anda dapat mendaftarkannya baik dari konsol Lake Formation atau dengan menggunakan AWS CLI:

```
aws lakeformation register-resource --resource-arn arn:aws:s3:::<s3-bucket>/<s3-folder> --use-service-linked-role --region <REGION>
```

Setelah Anda mendaftarkan lokasi Amazon S3, tabel AWS Glue apa pun yang menunjuk ke lokasi (atau lokasi turunannya) akan mengembalikan nilai `IsRegisteredWithLakeFormation` parameter sebagai true dalam panggilan. `GetTable`

b. Buat tabel Delta yang menunjuk ke jalur Amazon S3 terdaftar melalui Spark:

> **Note**
>
> Berikut ini adalah contoh Python.

```
dataFrame.write \
 .format("delta") \
 .mode("overwrite") \
 .partitionBy("<your_partitionkey_field>") \
 .save("s3://<the_s3_path>")
```

Setelah data ditulis ke Amazon S3, gunakan crawler AWS Glue untuk membuat tabel katalog Delta baru. Untuk informasi selengkapnya, lihat Memperkenalkan dukungan tabel Delta Lake asli dengan crawler AWS Glue.

Anda juga dapat membuat tabel secara manual melalui AWS Glue `CreateTable` API.

2. Berikan izin Formasi Lake untuk peran IAM pekerjaan AWS Glue. Anda dapat memberikan izin dari konsol Lake Formation, atau menggunakan AWS CLI. Untuk informasi selengkapnya, lihat Memberikan izin tabel menggunakan konsol Lake Formation dan metode sumber daya bernama
3. Baca tabel Delta yang terdaftar di Lake Formation. Kode ini sama dengan membaca tabel Delta yang tidak terdaftar. Perhatikan bahwa peran IAM AWS Glue job harus memiliki izin SELECT agar pembacaan berhasil.

```
# Example: Read a Delta Lake table from Glue Data Catalog

df = glueContext.create_data_frame.from_catalog(
    database="<your_database_name>",
```

```
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

4. Tulis ke tabel Delta yang terdaftar di Lake Formation. Kode ini sama dengan menulis ke tabel Delta yang tidak terdaftar. Perhatikan bahwa peran IAM AWS Glue job harus memiliki izin SUPER agar penulisan berhasil.

   Secara default AWS Glue menggunakan `Append` sebagai SaveMode. Anda dapat mengubahnya dengan mengatur opsi SaveMode di. `additional_options` Untuk informasi tentang dukungan SaveMode di tabel Delta, lihat Menulis ke tabel.

```
glueContext.write_data_frame.from_catalog(
    frame=dataFrame,
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Menggunakan kerangka Iceberg di Glue AWS

AWSGlue 3.0 dan yang lebih baru mendukung kerangka Apache Iceberg untuk data lake. Iceberg menyediakan format tabel kinerja tinggi yang bekerja seperti tabel SQL. Topik ini mencakup fitur yang tersedia untuk menggunakan data Anda di AWS Glue saat Anda mengangkut atau menyimpan data Anda dalam tabel Gunung Es. Untuk mempelajari lebih lanjut tentang Iceberg, lihat dokumentasi resmi Apache Iceberg.

Anda dapat menggunakan AWS Glue untuk melakukan operasi baca dan tulis pada tabel Iceberg di Amazon S3, atau bekerja dengan tabel Iceberg menggunakan AWS Glue Data Catalog. Operasi tambahan termasuk insert, update, dan semua Spark Queries Spark Write juga didukung.

> **Note**
>
> `ALTER TABLE … RENAME TO`tidak tersedia untuk Apache Iceberg 0.13.1 untuk Glue 3.0. AWS

Tabel berikut mencantumkan versi Iceberg yang disertakan dalam setiap versi AWS Glue.

AWSVersi Glue	Versi Iceberg yang Didukung
4.0	1.0.0
3.0	0.13.1

Untuk mempelajari lebih lanjut tentang framework data lake yang didukung AWS Glue, lihatMenggunakan kerangka kerja data lake dengan pekerjaan AWS Glue ETL.

Mengaktifkan kerangka Iceberg

Untuk mengaktifkan Iceberg for AWS Glue, selesaikan tugas berikut:

- Tentukan `iceberg` sebagai nilai untuk parameter `--datalake-formats` pekerjaan. Untuk informasi selengkapnya, lihat AWS Glueparameter pekerjaan.
- Buat kunci bernama `--conf` untuk pekerjaan AWS Glue Anda, dan atur ke nilai berikut. Atau, Anda dapat mengatur konfigurasi berikut menggunakan `SparkConf` skrip Anda. Pengaturan ini membantu Apache Spark menangani tabel Iceberg dengan benar.

```
spark.sql.extensions=org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions
--conf spark.sql.catalog.glue_catalog=org.apache.iceberg.spark.SparkCatalog
--conf spark.sql.catalog.glue_catalog.warehouse=s3://<your-warehouse-dir>/
--conf spark.sql.catalog.glue_catalog.catalog-
impl=org.apache.iceberg.aws.glue.GlueCatalog
--conf spark.sql.catalog.glue_catalog.io-impl=org.apache.iceberg.aws.s3.S3FileIO
```

  Jika Anda membaca atau menulis ke tabel Gunung Es yang terdaftar di Lake Formation, tambahkan konfigurasi berikut untuk mengaktifkan dukungan Lake Formation. Perhatikan bahwa hanya AWS Glue 4.0 yang mendukung tabel Gunung Es yang terdaftar di Lake Formation:

```
--conf spark.sql.catalog.glue_catalog.glue.lakeformation-enabled=true
--conf spark.sql.catalog.glue_catalog.glue.id=<table-catalog-id>
```

  Jika Anda menggunakan AWS Glue 3.0 dengan Iceberg 0.13.1, Anda harus mengatur konfigurasi tambahan berikut untuk menggunakan pengelola kunci Amazon DynamoDB untuk memastikan transaksi atom. AWS Glue 4.0 menggunakan penguncian optimis secara default. Untuk informasi selengkapnya, lihat AWSIntegrasi Gunung Es di dokumentasi Apache Iceberg resmi.

```
--conf spark.sql.catalog.glue_catalog.lock-
impl=org.apache.iceberg.aws.glue.DynamoLockManager
--conf spark.sql.catalog.glue_catalog.lock.table=<your-dynamodb-table-name>
```

Menggunakan versi Iceberg yang berbeda

Untuk menggunakan versi Iceberg yang tidak didukung AWS Glue, tentukan file Iceberg JAR Anda sendiri menggunakan parameter pekerjaan. `--extra-jars` Jangan sertakan `iceberg` sebagai nilai untuk `--datalake-formats` parameter.

Mengaktifkan enkripsi untuk tabel Iceberg

> **Note**
>
> Tabel gunung es memiliki mekanisme sendiri untuk mengaktifkan enkripsi sisi server. Anda harus mengaktifkan konfigurasi ini selain konfigurasi keamanan AWS Glue.

Untuk mengaktifkan enkripsi sisi server pada tabel Iceberg, tinjau panduan dari dokumentasi Iceberg.

Contoh: Tulis tabel Gunung Es ke Amazon S3 dan daftarkan ke Katalog Data Glue AWS

Contoh skrip ini menunjukkan cara menulis tabel Iceberg ke Amazon S3. Contoh menggunakan Iceberg AWS Integrations untuk mendaftarkan tabel ke AWS Glue Data Catalog.

Python

```
# Example: Create an Iceberg table from a DataFrame
# and register the table to Glue Data Catalog

dataFrame.createOrReplaceTempView("tmp_<your_table_name>")

query = f"""
CREATE TABLE glue_catalog.<your_database_name>.<your_table_name>
USING iceberg
TBLPROPERTIES ("format-version"="2")
AS SELECT * FROM tmp_<your_table_name>
"""
spark.sql(query)
```

Scala

```
// Example: Example: Create an Iceberg table from a DataFrame
// and register the table to Glue Data Catalog

dataFrame.createOrReplaceTempView("tmp_<your_table_name>")

val query = """CREATE TABLE glue_catalog.<your_database_name>.<your_table_name>
USING iceberg
TBLPROPERTIES ("format-version"="2")
AS SELECT * FROM tmp_<your_table_name>
"""
spark.sql(query)
```

Atau, Anda dapat menulis tabel Iceberg ke Amazon S3 dan Katalog Data menggunakan metode Spark.

Prasyarat: Anda perlu menyediakan katalog untuk perpustakaan Iceberg untuk digunakan. Saat menggunakan AWS Glue Data Catalog, AWS Glue membuatnya mudah. Katalog Data AWS Glue sudah dikonfigurasi sebelumnya untuk digunakan oleh pustaka Spark sebagai. `glue_catalog` *Tabel Katalog Data diidentifikasi oleh DatabaseName dan TableName.* Untuk informasi selengkapnya tentang Katalog Data AWS Glue, lihatPenemuan dan katalogisasi data.

Jika Anda tidak menggunakan Katalog Data AWS Glue, Anda harus menyediakan katalog melalui API Spark. Untuk informasi selengkapnya, lihat Konfigurasi Percikan di dokumentasi Gunung Es.

Contoh ini menulis tabel Iceberg ke Amazon S3 dan Katalog Data menggunakan Spark.

Python

```
# Example: Write an Iceberg table to S3 on the Glue Data Catalog

# Create (equivalent to CREATE TABLE AS SELECT)
dataFrame.writeTo("glue_catalog.databaseName.tableName") \
    .tableProperty("format-version", "2") \
    .create()

# Append (equivalent to INSERT INTO)
dataFrame.writeTo("glue_catalog.databaseName.tableName") \
    .tableProperty("format-version", "2") \
    .append()
```

Scala

```
// Example: Write an Iceberg table to S3 on the Glue Data Catalog

// Create (equivalent to CREATE TABLE AS SELECT)
dataFrame.writeTo("glue_catalog.databaseName.tableName")
    .tableProperty("format-version", "2")
    .create()

// Append (equivalent to INSERT INTO)
dataFrame.writeTo("glue_catalog.databaseName.tableName")
    .tableProperty("format-version", "2")
    .append()
```

Contoh: Membaca tabel Gunung Es dari Amazon S3 menggunakan Katalog Data Glue AWS

Contoh ini membaca tabel Iceberg yang Anda buat. Contoh: Tulis tabel Gunung Es ke Amazon S3 dan daftarkan ke Katalog Data Glue AWS

Python

Untuk contoh ini, gunakan `GlueContext.create_data_frame.from_catalog()` metode ini.

```
# Example: Read an Iceberg table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

df = glueContext.create_data_frame.from_catalog(
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Scala

Untuk contoh ini, gunakan getCatalogSourcemetode ini.

```
// Example: Read an Iceberg table from Glue Data Catalog
```

```
import com.amazonaws.services.glue.GlueContext
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val df = glueContext.getCatalogSource("<your_database_name>",
 "<your_table_name>",
      additionalOptions = additionalOptions)
      .getDataFrame()
  }
}
```

Contoh: Masukkan **`DataFrame`** ke dalam tabel Iceberg di Amazon S3 menggunakan Glue Data Catalog AWS

Contoh ini menyisipkan data ke dalam tabel Iceberg yang Anda buat. Contoh: Tulis tabel Gunung Es ke Amazon S3 dan daftarkan ke Katalog Data Glue AWS

> **Note**
>
> Contoh ini mengharuskan Anda untuk mengatur parameter `--enable-glue-datacatalog` pekerjaan untuk menggunakan Katalog Data AWS Glue sebagai metastore Apache Spark Hive. Untuk mempelajari selengkapnya, lihat AWS Glueparameter pekerjaan.

Python

Untuk contoh ini, gunakan `GlueContext.write_data_frame.from_catalog()` metode ini.

```
# Example: Insert into an Iceberg table from Glue Data Catalog

from awsglue.context import GlueContext
from pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

glueContext.write_data_frame.from_catalog(
```

```
    frame=dataFrame,
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Scala

Untuk contoh ini, gunakan getCatalogSinkmetode ini.

```
// Example: Insert into an Iceberg table from Glue Data Catalog

import com.amazonaws.services.glue.GlueContext
import org.apacke.spark.SparkContext

object GlueApp {
  def main(sysArgs: Array[String]): Unit = {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    glueContext.getCatalogSink("<your_database_name>", "<your_table_name>",
      additionalOptions = additionalOptions)
      .writeDataFrame(dataFrame, glueContext)
  }
}
```

Contoh: Baca tabel Iceberg dari Amazon S3 menggunakan Spark

Prasyarat: Anda perlu menyediakan katalog untuk perpustakaan Iceberg untuk digunakan. Saat menggunakan AWS Glue Data Catalog, AWS Glue membuatnya mudah. Katalog Data AWS Glue sudah dikonfigurasi sebelumnya untuk digunakan oleh pustaka Spark sebagai. `glue_catalog` *Tabel Katalog Data diidentifikasi oleh DatabaseName dan TableName.* Untuk informasi selengkapnya tentang Katalog Data AWS Glue, lihatPenemuan dan katalogisasi data.

Jika Anda tidak menggunakan Katalog Data AWS Glue, Anda harus menyediakan katalog melalui API Spark. Untuk informasi selengkapnya, lihat Konfigurasi Percikan di dokumentasi Gunung Es.

Contoh ini membaca tabel Iceberg di Amazon S3 dari Katalog Data menggunakan Spark.

Python

```
# Example: Read an Iceberg table on S3 as a DataFrame from the Glue Data Catalog
```

```
dataFrame = spark.read.format("iceberg").load("glue_catalog.databaseName.tableName")
```

Scala

```
// Example: Read an Iceberg table on S3 as a DataFrame from the Glue Data Catalog

val dataFrame =
 spark.read.format("iceberg").load("glue_catalog.databaseName.tableName")
```

Contoh: Membaca dan menulis tabel Gunung Es dengan kontrol izin Lake Formation

Contoh ini membaca dan menulis tabel Gunung Es dengan kontrol izin Lake Formation.

1. Buat tabel Gunung Es dan daftarkan di Lake Formation:

   a. Untuk mengaktifkan kontrol izin Lake Formation, Anda harus terlebih dahulu mendaftarkan tabel jalur Amazon S3 di Lake Formation. Untuk informasi selengkapnya, lihat Mendaftarkan lokasi Amazon S3. Anda dapat mendaftarkannya baik dari konsol Lake Formation atau dengan menggunakan AWS CLI:

   ```
   aws lakeformation register-resource --resource-arn arn:aws:s3:::<s3-bucket>/<s3-
   folder> --use-service-linked-role --region <REGION>
   ```

   Setelah Anda mendaftarkan lokasi Amazon S3, tabel AWS Glue apa pun yang menunjuk ke lokasi (atau lokasi turunan mana pun) akan mengembalikan nilai `IsRegisteredWithLakeFormation` parameter sebagai true dalam panggilan. `GetTable`

   b. Buat tabel Iceberg yang menunjuk ke jalur terdaftar melalui Spark SQL:

   > **Note**
   >
   > Berikut ini adalah contoh Python.

   ```
   dataFrame.createOrReplaceTempView("tmp_<your_table_name>")

   query = f"""
   CREATE TABLE glue_catalog.<your_database_name>.<your_table_name>
   USING iceberg
   AS SELECT * FROM tmp_<your_table_name>
   ```

```
"""
spark.sql(query)
```

Anda juga dapat membuat tabel secara manual melalui AWS Glue `CreateTable` API. Untuk informasi selengkapnya, lihat Membuat tabel Apache Iceberg.

2. Berikan izin Formasi Danau untuk peran IAM pekerjaan. Anda dapat memberikan izin dari konsol Lake Formation, atau menggunakan AWS CLI. Untuk informasi selengkapnya, lihat: https:// docs.aws.amazon.com/lake-formation/latest/dg/ granting-table-permissions .html
3. Baca tabel Gunung Es yang terdaftar di Lake Formation. Kode ini sama dengan membaca tabel Iceberg yang tidak terdaftar. Perhatikan bahwa peran IAM AWS Glue job Anda harus memiliki izin SELECT agar pembacaan berhasil.

```
# Example: Read an Iceberg table from the AWS Glue Data Catalog
from awsglue.context import GlueContextfrom pyspark.context import SparkContext

sc = SparkContext()
glueContext = GlueContext(sc)

df = glueContext.create_data_frame.from_catalog(
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

4. Tulis ke meja Gunung Es yang terdaftar di Lake Formation. Kode ini sama dengan menulis ke tabel Iceberg yang tidak terdaftar. Perhatikan bahwa peran IAM AWS Glue job Anda harus memiliki izin SUPER agar penulisan berhasil.

```
glueContext.write_data_frame.from_catalog(
    frame=dataFrame,
    database="<your_database_name>",
    table_name="<your_table_name>",
    additional_options=additional_options
)
```

Referensi konfigurasi bersama

Anda dapat menggunakan `format_options` nilai berikut dengan jenis format apa pun.

- `attachFilename`— Sebuah string dalam format yang sesuai untuk digunakan sebagai nama kolom. Jika Anda memberikan opsi ini, nama file sumber untuk catatan akan ditambahkan ke catatan. Nilai parameter akan digunakan sebagai nama kolom.
- `attachTimestamp`— Sebuah string dalam format yang sesuai untuk digunakan sebagai nama kolom. Jika Anda memberikan opsi ini, waktu modifikasi file sumber untuk catatan akan ditambahkan ke catatan. Nilai parameter akan digunakan sebagai nama kolom.

AWSDukungan Glue Data Catalog untuk pekerjaan Spark SQL

Katalog Data Glue AWS adalah katalog yang kompatibel dengan metastore Apache Hive. Anda dapat mengonfigurasi AWS Glue pekerjaan dan titik akhir pengembangan untuk menggunakan Katalog Data sebagai metastore Apache Hive eksternal. Anda kemudian dapat langsung menjalankan kueri Apache Spark SQL terhadap tabel yang disimpan dalam Katalog Data. AWS Glueframe dinamis terintegrasi dengan Katalog Data secara default. Namun, dengan fitur ini, tugas Spark SQL dapat mulai menggunakan Katalog Data sebagai metastore Hive eksternal.

Fitur ini memerlukan akses jaringan ke titik akhir AWS Glue API. Untuk AWS Glue pekerjaan dengan koneksi yang terletak di subnet pribadi, Anda harus mengonfigurasi titik akhir VPC atau gateway NAT untuk menyediakan akses jaringan. Untuk informasi tentang mengonfigurasi titik akhir VPC, lihat. Menyiapkan akses jaringan ke penyimpanan data Untuk membuat gateway NAT, lihat Gateway NAT di Panduan Pengguna Amazon VPC.

Anda dapat mengonfigurasi AWS Glue pekerjaan dan titik akhir pengembangan dengan menambahkan `"--enable-glue-datacatalog": ""` argumen ke argumen pekerjaan dan argumen titik akhir pengembangan masing-masing. Dengan memberikan argumen ini akan menetapkan konfigurasi tertentu di Spark yang memungkinkannya untuk mengakses Katalog Data sebagai metastore Hive eksternal. Ini juga memungkinkan dukungan Hive di `SparkSession` objek yang dibuat di titik akhir AWS Glue pekerjaan atau pengembangan.

Untuk mengaktifkan akses Katalog Data, berikan tanda centang pada kotak centang Gunakan Katalog Data Glue AWS sebagai Metastore Hive di grup Opsi katalog pada Tambahkan tugas atau Tambahkan titik akhir di konsol. Perhatikan bahwa IAM role yang digunakan untuk tugas atau titik akhir pengembangan harus memiliki izin `glue:CreateDatabase`. Sebuah basis data dengan nama "`default`" dibuat dalam Katalog Data jika tidak ada.

Mari kita lihat contoh bagaimana Anda dapat menggunakan fitur ini dalam tugas Spark SQL Anda. Contoh berikut mengasumsikan bahwa Anda telah melakukan perayapan pada set data legislator AS yang tersedia di `s3://awsglue-datasets/examples/us-legislators`.

Untuk membuat serialisasi/deserialisasi data dari tabel yang didefinisikan dalam Katalog Data AWS Glue, Spark SQL membutuhkan kelas Hive SerDe untuk format yang ditentukan dalam Katalog Data AWS Glue di classpath pekerjaan percikan.

SerDes untuk format umum tertentu didistribusikan olehAWS Glue. Berikut ini adalah link Amazon S3 untuk ini:

- JSON
- XML
- Grok

Tambahkan JSON SerDe sebagai JAR tambahan ke titik akhir pengembangan. Untuk pekerjaan, Anda dapat menambahkan SerDe menggunakan `--extra-jars` argumen di bidang argumen. Untuk informasi selengkapnya, lihat AWS Glueparameter pekerjaan.

Berikut adalah contoh masukan JSON untuk membuat titik akhir pengembangan dengan Katalog Data yang diaktifkan untuk Spark SQL.

```
{
    "EndpointName": "Name",
    "RoleArn": "role_ARN",
    "PublicKey": "public_key_contents",
    "NumberOfNodes": 2,
    "Arguments": {
      "--enable-glue-datacatalog": ""
    },
    "ExtraJarsS3Path": "s3://crawler-public/json/serde/json-serde.jar"
}
```

Sekarang lakukan kueri pada tabel yang dibuat dari set data legislator AS dengan menggunakan Spark SQL.

```
>>> spark.sql("use legislators")
DataFrame[]
>>> spark.sql("show tables").show()
+-----------+-----------------+-----------+
|   database|        tableName|isTemporary|
+-----------+-----------------+-----------+
```

```
|legislators|        areas_json|      false|
|legislators|    countries_json|      false|
|legislators|       events_json|      false|
|legislators|  memberships_json|      false|
|legislators|organizations_json|      false|
|legislators|      persons_json|      false|
+-----------+------------------+-----------+
>>> spark.sql("describe memberships_json").show()
+--------------------+---------+-----------------+
|            col_name|data_type|          comment|
+--------------------+---------+-----------------+
|             area_id|   string|from deserializer|
|     on_behalf_of_id|   string|from deserializer|
|     organization_id|   string|from deserializer|
|                role|   string|from deserializer|
|           person_id|   string|from deserializer|
|legislative_perio...|   string|from deserializer|
|          start_date|   string|from deserializer|
|            end_date|   string|from deserializer|
+--------------------+---------+-----------------+
```

Jika SerDe kelas untuk format tidak tersedia di classpath pekerjaan, Anda akan melihat kesalahan yang mirip dengan berikut ini.

```
>>> spark.sql("describe memberships_json").show()

Caused by: MetaException(message:java.lang.ClassNotFoundException Class
 org.openx.data.jsonserde.JsonSerDe not found)
    at
 org.apache.hadoop.hive.metastore.MetaStoreUtils.getDeserializer(MetaStoreUtils.java:399)
    at
 org.apache.hadoop.hive.ql.metadata.Table.getDeserializerFromMetaStore(Table.java:276)
    ... 64 more
```

Untuk hanya melihat `organization_id` yang berbeda dari tabel `memberships`, jalankan kueri SQL berikut.

```
>>> spark.sql("select distinct organization_id from memberships_json").show()
+--------------------+
|     organization_id|
+--------------------+
|d56acebe-8fdc-47b...|
|8fa6c3d2-71dc-478...|
```

```
+--------------------+
```

Jika Anda perlu melakukan hal yang sama dengan bingkai dinamis, jalankan yang berikut.

```
>>> memberships = glueContext.create_dynamic_frame.from_catalog(database="legislators",
 table_name="memberships_json")
>>> memberships.toDF().createOrReplaceTempView("memberships")
>>> spark.sql("select distinct organization_id from memberships").show()
+--------------------+
|     organization_id|
+--------------------+
|d56acebe-8fdc-47b...|
|8fa6c3d2-71dc-478...|
+--------------------+
```

Sementara DynamicFrames dioptimalkan untuk operasi ETL, mengaktifkan Spark SQL untuk mengakses Katalog Data secara langsung menyediakan cara ringkas untuk menjalankan pernyataan SQL yang kompleks atau port aplikasi yang ada.

Menggunakan bookmark pekerjaan

AWS Glueuntuk Spark menggunakan bookmark pekerjaan untuk melacak data yang telah diproses. Untuk ringkasan fitur bookmark pekerjaan dan apa yang didukungnya, lihatthe section called “Melacak data yang diproses menggunakan bookmark pekerjaan”. Saat memprogram AWS Glue pekerjaan dengan bookmark, Anda memiliki akses ke fleksibilitas yang tidak tersedia dalam pekerjaan visual.

- Saat membaca dari JDBC, Anda dapat menentukan kolom yang akan digunakan sebagai tombol bookmark dalam skrip Anda. AWS Glue
- Anda dapat memilih mana `transformation_ctx` yang akan diterapkan pada setiap panggilan metode.

Selalu panggil *`job.init`* di awal skrip dan *`job.commit`* di akhir skrip dengan parameter yang dikonfigurasi dengan tepat. Kedua fungsi ini menginisialisasi layanan bookmark dan memperbarui perubahan status ke layanan. Bookmark tidak akan berfungsi tanpa memanggilnya.

Tentukan tombol bookmark

Untuk alur kerja JDBC, bookmark melacak baris mana yang telah dibaca pekerjaan Anda dengan membandingkan nilai bidang kunci dengan nilai yang ditandai. Ini tidak perlu atau berlaku untuk

alur kerja Amazon S3. Saat menulis AWS Glue skrip tanpa editor visual, Anda dapat menentukan kolom mana yang akan dilacak dengan bookmark. Anda juga dapat menentukan beberapa kolom. Kesenjangan dalam urutan nilai diizinkan saat menentukan kunci bookmark yang ditentukan pengguna.

> **Warning**
>
> Jika tombol bookmark yang ditentukan pengguna digunakan, masing-masing tombol tersebut harus benar-benar meningkat atau menurun secara monoton. Saat memilih bidang tambahan untuk kunci majemuk, bidang untuk konsep seperti “versi minor” atau “nomor revisi” tidak memenuhi kriteria ini, karena nilainya digunakan kembali di seluruh kumpulan data Anda.

Anda dapat menentukan `jobBookmarkKeys` dan dengan `jobBookmarkKeysSortOrder` cara berikut:

- `create_dynamic_frame.from_catalog`— Gunakan`additional_options`.
- `create_dynamic_frame.from_options`— Gunakan`connection_options`.

Konteks transformasi

Banyak metode frame AWS Glue PySpark dinamis menyertakan parameter opsional bernama`transformation_ctx`, yang merupakan pengidentifikasi unik untuk instance operator ETL. Parameter `transformation_ctx` digunakan untuk mengidentifikasi informasi status dalam bookmark tugas untuk operator tertentu. Secara khusus, AWS Glue menggunakan `transformation_ctx` untuk mengindeks kunci ke status bookmark.

> **Warning**
>
> Ini `transformation_ctx` berfungsi sebagai kunci untuk mencari status bookmark untuk sumber tertentu dalam skrip Anda. Agar bookmark berfungsi dengan baik, Anda harus selalu menjaga sumber dan yang terkait `transformation_ctx` konsisten. Mengubah properti sumber atau mengganti nama `transformation_ctx` dapat membuat bookmark sebelumnya tidak valid dan pemfilteran berbasis stempel waktu mungkin tidak menghasilkan hasil yang benar.

Agar bookmark tugas bekerja dengan benar, aktifkan parameter bookmark tugas dan atur parameter `transformation_ctx`. Jika Anda tidak memberikan parameter `transformation_ctx`, maka bookmark tugas tidak akan diaktifkan untuk bingkai dinamis atau tabel yang digunakan dalam metode tersebut. Sebagai contoh, jika Anda memiliki tugas ETL yang membaca dan menggabungkan dua sumber Amazon S3, maka Anda dapat memilih untuk memberikan parameter `transformation_ctx` hanya untuk metode-metode yang Anda ingin aktifkan bookmark-nya. Jika Anda menyetel ulang bookmark tugas untuk sebuah tugas, hal itu akan me-reset semua transformasi yang dikaitkan dengan tugas tersebut terlepas dari `transformation_ctx` yang digunakan.

Untuk informasi lebih lanjut tentang kelas `DynamicFrameReader`, lihat DynamicFrameReader kelas. Untuk informasi selengkapnya tentang PySpark ekstensi, lihatAWS Glue PySpark referensi ekstensi.

Contoh-contoh

Example

Berikut ini adalah contoh skrip yang dihasilkan untuk sumber data Amazon S3. Bagian dari skrip yang diperlukan untuk menggunakan bookmark pekerjaan ditampilkan dalam huruf miring. Untuk informasi selengkapnya tentang elemen-elemen ini lihat API GlueContext kelas, dan API DynamicFrameWriter kelas.

```
# Sample Script
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ['JOB_NAME'])
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)

datasource0 = glueContext.create_dynamic_frame.from_catalog(
    database = "database",
    table_name = "relatedqueries_csv",
    transformation_ctx = "datasource0"
)
```

```
applymapping1 = ApplyMapping.apply(
    frame = datasource0,
    mappings = [("col0", "string", "name", "string"), ("col1", "string", "number",
 "string")],
    transformation_ctx = "applymapping1"
)

datasink2 = glueContext.write_dynamic_frame.from_options(
    frame = applymapping1,
    connection_type = "s3",
    connection_options = {"path": "s3://input_path"},
    format = "json",
    transformation_ctx = "datasink2"
)


job.commit()
```

Example

Berikut ini adalah contoh skrip yang dihasilkan untuk sumber JDBC. Tabel sumber adalah tabel karyawan dengan kolom empno sebagai kunci primer. Meskipun secara default tugas menggunakan kunci primer berurutan sebagai kunci bookmark jika tidak ada kunci bookmark yang ditentukan, karena empno belum tentu secara berurutan—mungkin ada celah dalam nilai—maka ia tidak memenuhi syarat sebagai kunci bookmark default. Oleh karena itu, skrip secara eksplisit menetapkan empno sebagai kunci bookmark. Bagian kode itu ditampilkan dalam huruf miring.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

args = getResolvedOptions(sys.argv, ['JOB_NAME'])

sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args['JOB_NAME'], args)
```

```
datasource0 = glueContext.create_dynamic_frame.from_catalog(
    database = "hr",
    table_name = "emp",
    transformation_ctx = "datasource0",
    additional_options = {"jobBookmarkKeys":["empno"],"jobBookmarkKeysSortOrder":"asc"}
)

applymapping1 = ApplyMapping.apply(
    frame = datasource0,
    mappings = [("ename", "string", "ename", "string"), ("hrly_rate", "decimal(38,0)",
 "hrly_rate", "decimal(38,0)"), ("comm", "decimal(7,2)", "comm", "decimal(7,2)"),
 ("hiredate", "timestamp", "hiredate", "timestamp"), ("empno", "decimal(5,0)", "empno",
 "decimal(5,0)"), ("mgr", "decimal(5,0)", "mgr", "decimal(5,0)"), ("photo", "string",
 "photo", "string"), ("job", "string", "job", "string"), ("deptno", "decimal(3,0)",
 "deptno", "decimal(3,0)"), ("ssn", "decimal(9,0)", "ssn", "decimal(9,0)"), ("sal",
 "decimal(7,2)", "sal", "decimal(7,2)")],
    transformation_ctx = "applymapping1"
)

datasink2 = glueContext.write_dynamic_frame.from_options(
    frame = applymapping1,
    connection_type = "s3",
    connection_options = {"path": "s3://hr/employees"},
    format = "csv",
    transformation_ctx = "datasink2"
)

job.commit()
```

Menggunakan Deteksi Data Sensitif di luar AWS Glue Studio

AWS Glue Studio memungkinkan Anda mendeteksi data sensitif, namun Anda juga dapat menggunakan fungsi Deteksi Data Sensitif di luar AWS Glue Studio.

Untuk daftar lengkap tipe data sensitif terkelola, lihat Tipe data terkelola.

Mendeteksi Deteksi Data Sensitif menggunakan tipe PII AWS Terkelola

AWS Glue menyediakan dua API dalam pekerjaan AWS Glue ETL. Ini adalah `detect()` dan`classifyColumns()`:

```
  detect(frame: DynamicFrame,
```

```
      entityTypesToDetect: Seq[String],
      outputColumnName: String = "DetectedEntities",
      detectionSensitivity: String = "LOW"): DynamicFrame

  detect(frame: DynamicFrame,
      detectionParameters: JsonOptions,
      outputColumnName: String = "DetectedEntities",
      detectionSensitivity: String = "LOW"): DynamicFrame

   classifyColumns(frame: DynamicFrame,
      entityTypesToDetect: Seq[String],
      sampleFraction: Double = 0.1,
      thresholdFraction: Double = 0.1,
      detectionSensitivity: String = "LOW")
```

Anda dapat menggunakan `detect()` API untuk mengidentifikasi tipe PII AWS Terkelola dan jenis entitas kustom. Kolom baru secara otomatis dibuat dengan hasil deteksi. `classifyColumns()`API mengembalikan peta di mana kunci adalah nama kolom dan nilai adalah daftar tipe entitas yang terdeteksi. `SampleFraction`menunjukkan fraksi data yang akan diambil sampel saat memindai entitas PII sedangkan `ThresholdFraction` menunjukkan fraksi data yang harus dipenuhi agar kolom diidentifikasi sebagai data PII.

Deteksi tingkat baris

Dalam contoh, pekerjaan melakukan tindakan berikut menggunakan `classifyColumns()` API `detect()` dan:

- membaca data dari Amazon S3 ember dan mengubahnya menjadi DynamicFrame
- mendeteksi contoh “Email” dan “Kartu Kredit” di DynamicFrame
- mengembalikan DynamicFrame dengan nilai asli ditambah satu kolom yang mencakup hasil deteksi untuk setiap baris
- menulis DynamicFrame yang dikembalikan di jalur lain Amazon S3

```
 import com.amazonaws.services.glue.GlueContext
 import com.amazonaws.services.glue.MappingSpec
 import com.amazonaws.services.glue.errors.CallSite
 import com.amazonaws.services.glue.util.GlueArgParser
```

```
 import com.amazonaws.services.glue.util.Job
 import com.amazonaws.services.glue.util.JsonOptions
 import org.apache.spark.SparkContext
 import scala.collection.JavaConverters._
 import com.amazonaws.services.glue.ml.EntityDetector

 object GlueApp {
   def main(sysArgs: Array[String]) {
     val spark: SparkContext = new SparkContext()
     val glueContext: GlueContext = new GlueContext(spark)
     val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
     Job.init(args("JOB_NAME"), glueContext, args.asJava)
     val frame=
glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar": "\"",
"withHeader": true, "separator": ","}"""), connectionType="s3", format="csv",
options=JsonOptions("""{"paths": ["s3://pathToSource"], "recurse": true}"""),
transformationContext="AmazonS3_node1650160158526").getDynamicFrame()

     val frameWithDetectedPII = EntityDetector.detect(frame, Seq("EMAIL",
"CREDIT_CARD"))

     glueContext.getSinkWithFormat(connectionType="s3",
options=JsonOptions("""{"path": "s3://pathToOutput/", "partitionKeys": []}"""),
transformationContext="someCtx",
format="json").writeDynamicFrame(frameWithDetectedPII)

     Job.commit()
   }
 }
```

Deteksi tingkat baris dengan tindakan berbutir halus

Dalam contoh, tugas melakukan tindakan berikut menggunakan `detect()` API:

- membaca data dari bucket Amazon S3 dan mengubahnya menjadi DynamicFrame
- mendeteksi tipe data sensitif untuk “USA_PTIN”, “BANK_ACCOUNT”, “USA_SSN”, “USA_PASSPORT_NUMBER”, dan “PHONE_NUMBER” di DynamicFrame
- mengembalikan DynamicFrame dengan nilai bertopeng yang dimodifikasi ditambah satu kolom yang mencakup hasil deteksi untuk setiap baris
- menulis DynamicFrame yang dikembalikan di jalur Amazon S3 lainnya

Berbeda dengan `detect()` API di atas, ini menggunakan tindakan berbutir halus untuk dideteksi oleh tipe entitas. Untuk informasi selengkapnya, lihat Parameter deteksi untuk digunakan `detect()`.

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
import com.amazonaws.services.glue.ml.EntityDetector

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
    val frame =
 glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar": "\"",
 "withHeader": true, "separator": ","}"""), connectionType="s3", format="csv",
 options=JsonOptions("""{"paths": ["s3://pathToSource"], "recurse": true}"""),
 transformationContext="AmazonS3_node_source").getDynamicFrame()

    val detectionParameters = JsonOptions(
      """
        {
          "USA_DRIVING_LICENSE": [{
            "action": "PARTIAL_REDACT",
            "sourceColumns": ["Driving License"],
            "actionOptions": {
              "matchPattern": "[0-9]",
              "redactChar": "*"
            }
          }],
          "BANK_ACCOUNT": [{
            "action": "DETECT",
            "sourceColumns": ["*"]
          }],
          "USA_SSN": [{
            "action": "SHA256_HASH",
            "sourceColumns": ["SSN"]
```

```
          }],
          "IP_ADDRESS": [{
            "action": "REDACT",
            "sourceColumns": ["IP Address"],
            "actionOptions": {"redactText": "*****"}
          }],
          "PHONE_NUMBER": [{
            "action": "PARTIAL_REDACT",
            "sourceColumns": ["Phone Number"],
            "actionOptions": {
              "numLeftCharsToExclude": 1,
              "numRightCharsToExclude": 0,
              "redactChar": "*"
            }
          }]
        }
      """
    )

    val frameWithDetectedPII = EntityDetector.detect(frame, detectionParameters,
 "DetectedEntities", "HIGH")

    glueContext.getSinkWithFormat(connectionType="s3", options=JsonOptions("""{"path":
 "s3://pathToOutput/", "partitionKeys": []}"""),
 transformationContext="AmazonS3_node_target",
 format="json").writeDynamicFrame(frameWithDetectedPII)

    Job.commit()
  }
}
```

Deteksi tingkat kolom

Dalam contoh, tugas melakukan tindakan berikut menggunakan `classifyColumns()` API:

- membaca data dari bucket Amazon S3 dan mengubahnya menjadi DynamicFrame
- mendeteksi contoh “Email” dan “Kartu Kredit” di DynamicFrame
- atur parameter ke sampel 100% kolom, tandai entitas sebagai terdeteksi jika berada di 10% sel, dan memiliki sensitivitas “RENDAH”
- mengembalikan peta di mana kunci adalah nama kolom dan nilai daftar jenis entitas terdeteksi

- menulis DynamicFrame yang dikembalikan di jalur Amazon S3 lainnya

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
import com.amazonaws.services.glue.DynamicFrame
import com.amazonaws.services.glue.ml.EntityDetector

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
    val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
    Job.init(args("JOB_NAME"), glueContext, args.asJava)
    val frame =
 glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar":
 "\"", "withHeader": true, "separator": ",", "optimizePerformance": false}"""),
 connectionType="s3", format="csv", options=JsonOptions("""{"paths": ["s3://
pathToSource"], "recurse": true}"""), transformationContext="frame").getDynamicFrame()

    import glueContext.sparkSession.implicits._

    val detectedDataFrame = EntityDetector.classifyColumns(
        frame,
        entityTypesToDetect = Seq("CREDIT_CARD", "PHONE_NUMBER"),
        sampleFraction = 1.0,
        thresholdFraction = 0.1,
        detectionSensitivity = "LOW"
    )
    val detectedDF = (detectedDataFrame).toSeq.toDF("columnName", "entityTypes")
    val DetectSensitiveData_node = DynamicFrame(detectedDF, glueContext)

    glueContext.getSinkWithFormat(connectionType="s3", options=JsonOptions("""{"path":
 "s3://pathToOutput", "partitionKeys": []}"""), transformationContext="someCtx",
 format="json").writeDynamicFrame(DetectSensitiveData_node)

    Job.commit()
  }
```

```
}
```

Mendeteksi Deteksi Data Sensitif menggunakan tipe AWS CustomEntityType PII

Anda dapat menentukan entitas kustom melalui AWS Studio. Namun, untuk menggunakan fitur ini di luar AWS Studio, Anda harus terlebih dahulu menentukan jenis entitas kustom dan kemudian menambahkan jenis entitas kustom yang ditentukan ke daftar`entityTypesToDetect`.

Jika Anda memiliki tipe data sensitif tertentu dalam data Anda (seperti 'ID Karyawan'), Anda dapat membuat entitas kustom dengan memanggil API. `CreateCustomEntityType()` Contoh berikut mendefinisikan jenis entitas kustom 'EMPLOYEE_ID' ke `CreateCustomEntityType()` API dengan parameter permintaan:

```
{
    "name": "EMPLOYEE_ID",
    "regexString": "\d{4}-\d{3}",
    "contextWords": ["employee"]
}
```

Kemudian, ubah pekerjaan untuk menggunakan tipe data sensitif kustom baru dengan menambahkan tipe entitas kustom (EMPLOYEE_ID) ke API: `EntityDetector()`

```
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._
import com.amazonaws.services.glue.ml.EntityDetector

object GlueApp {
  def main(sysArgs: Array[String]) {
    val spark: SparkContext = new SparkContext()
    val glueContext: GlueContext = new GlueContext(spark)
```

```
      val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
      Job.init(args("JOB_NAME"), glueContext, args.asJava)
      val frame=
glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar": "\"",
"withHeader": true, "separator": ","}"""), connectionType="s3", format="csv",
options=JsonOptions("""{"paths": ["s3://pathToSource"], "recurse": true}"""),
transformationContext="AmazonS3_node1650160158526").getDynamicFrame()

      val frameWithDetectedPII = EntityDetector.detect(frame, Seq("EMAIL",
"CREDIT_CARD", "EMPLOYEE_ID"))

      glueContext.getSinkWithFormat(connectionType="s3",
options=JsonOptions("""{"path": "s3://pathToOutput/", "partitionKeys": []}"""),
transformationContext="someCtx",
format="json").writeDynamicFrame(frameWithDetectedPII)

      Job.commit()
    }
 }
```

> **Note**
>
> Jika tipe data sensitif kustom didefinisikan dengan nama yang sama dengan tipe entitas terkelola yang ada, maka tipe data sensitif kustom akan mengambil preseden dan menimpa logika tipe entitas terkelola.

Parameter deteksi untuk digunakan **`detect()`**

Metode ini digunakan untuk mendeteksi entitas dalam a DynamicFrame. Ia mengembalikan baru DataFrame dengan nilai asli dan kolom tambahan outputColumnName yang memiliki metadata deteksi PII. Penyamaran khusus dapat dilakukan setelah DynamicFrame ini dikembalikan dalam AWS Glue skrip, atau detect () dengan API tindakan berbutir halus dapat digunakan sebagai gantinya.

```
detect(frame: DynamicFrame,
       entityTypesToDetect: Seq[String],
       outputColumnName: String = "DetectedEntities",
       detectionSensitivity: String = "LOW"): DynamicFrame
```

Parameter:

- frame — (type:`DynamicFrame`) Input DynamicFrame yang berisi data yang akan diproses.
- entityTypesToDetect — (type:`[Seq[String]`) Daftar tipe entitas yang akan dideteksi. Dapat berupa Jenis Entitas Terkelola atau Jenis Entitas Kustom.
- outputColumnName— (type:`String`, default: "DetectedEntities“) Nama kolom tempat entitas yang terdeteksi akan disimpan. Jika tidak disediakan, nama kolom default adalah "DetectedEntities”.
- DetectionSensitivity — (type:`String`, options: “LOW” atau “HIGH”, default: “LOW”) Menentukan sensitivitas proses deteksi. Opsi yang valid adalah “RENDAH” atau “TINGGI”. Jika tidak disediakan, sensitivitas default diatur ke “RENDAH”.

`outputColumnName`pengaturan:

Nama kolom tempat entitas yang terdeteksi akan disimpan. Jika tidak disediakan, nama kolom default adalah "DetectedEntities”. Untuk setiap baris di kolom keluaran, kolom tambahan menyertakan peta nama kolom ke metadata entitas yang terdeteksi dengan pasangan nilai kunci berikut:

- EntityType - Jenis entitas yang terdeteksi.
- start — Posisi awal entitas yang terdeteksi dalam data asli.
- akhir — Posisi akhir dari entitas yang terdeteksi dalam data asli.
- ActionUsed — Tindakan yang dilakukan pada entitas yang terdeteksi (misalnya, “DETECT,” “REDACT,” “PARTIAL_REDACT,” “SHA256_HASH”).

Contoh:

```
{
   "DetectedEntities":{
      "SSN Col":[
         {
            "entityType":"USA_SSN",
            "actionUsed":"DETECT",
            "start":4,
            "end":15
         }
      ],
      "Random Data col":[
         {
            "entityType":"BANK_ACCOUNT",
```

```
            "actionUsed":"PARTIAL_REDACT",
            "start":4,
            "end":13
         },
         {
            "entityType":"IP_ADDRESS",
            "actionUsed":"REDACT",
            "start":4,
            "end":13
         }
      ]
   }
}
```

Parameter Deteksi untuk **`detect()`** dengan tindakan berbutir halus

Metode ini digunakan untuk mendeteksi entitas dalam DynamicFrame menggunakan parameter tertentu. Ini mengembalikan yang baru DataFrame dengan nilai asli diganti dengan data sensitif bertopeng dan kolom tambahan `outputColumnName` yang memiliki metadata deteksi PII.

```
detect(frame: DynamicFrame,
       detectionParameters: JsonOptions,
       outputColumnName: String = "DetectedEntities",
       detectionSensitivity: String = "LOW"): DynamicFrame
```

Parameter:

- frame — (type:`DynamicFrame`): Input DynamicFrame yang berisi data yang akan diproses.
- DetectionParameters — (type:`JsonOptions`): Opsi JSON yang menentukan parameter untuk proses deteksi.
- outputColumnName— (type:`String`, default: "DetectedEntities“): Nama kolom tempat entitas yang terdeteksi akan disimpan. Jika tidak disediakan, nama kolom default adalah "DetectedEntities”.
- DetectionSensitivity — (type:`String`, options: “LOW” atau “HIGH”, default: “LOW”): Menentukan sensitivitas proses deteksi. Opsi yang valid adalah “RENDAH” atau “TINGGI”. Jika tidak disediakan, sensitivitas default diatur ke “RENDAH”.

Pengaturan `detectionParameters`

Jika tidak ada pengaturan yang disertakan, nilai default akan digunakan.

- action — (type:`String`, options: “DETECT”, “REDACT”, “PARTIAL_REDACT”, “SHA256_HASH”) Menentukan tindakan yang akan dilakukan pada entitas. Wajib. Perhatikan bahwa tindakan yang melakukan masking (semua kecuali “DETECT”) hanya dapat melakukan satu tindakan per kolom. Ini adalah tindakan pencegahan untuk menutupi entitas yang bersatu.
- SourceColumns — (type:`List[String]`, default: [“*”]) Daftar nama kolom sumber untuk melakukan deteksi pada entitas. Default ke [“*”] jika tidak ada. Naikkan `IllegalArgumentException` jika nama kolom yang tidak valid digunakan.
- sourceColumnsToKecualikan — (tipe:`List[String]`) Daftar nama kolom sumber untuk melakukan deteksi pada entitas. Gunakan salah satu `sourceColumns` atau`sourceColumnsToExclude`. Naikkan `IllegalArgumentException` jika nama kolom yang tidak valid digunakan.
- ActionOptions - Opsi tambahan berdasarkan tindakan yang ditentukan:
  - Untuk “DETECT” dan “SHA256_HASH”, tidak ada opsi yang diizinkan.
  - Untuk “REDACT”:
    - RedactText — (type:`String`, default: “*****”) Teks untuk menggantikan entitas yang terdeteksi.
  - Untuk “PARTIAL_REDACT”:
    - RedactChar — (type:`String`, default: “*”) Karakter untuk menggantikan setiap karakter yang terdeteksi dalam entitas.
    - MatchPattern — (type:`String`) Pola Regex untuk redaksi parsi. Tidak dapat digabungkan dengan numLeftChars ToExclude atau`numRightCharsToExclude`.
    - numLeftCharsToExclude— (type:`String, integer`) Jumlah karakter kiri untuk dikecualikan. Tidak dapat digabungkan dengan MatchPattern, tetapi dapat digunakan dengan. `numRightCharsToExclude`
    - numRightCharsToExclude— (type:`String, integer`) Jumlah karakter yang tepat untuk dikecualikan. Tidak dapat digabungkan dengan MatchPattern, tetapi dapat digunakan dengan. `numRightCharsToExclude`

Pengaturan `outputColumnName`

Lihat outputColumnName setelan

Parameter Deteksi untuk **`classifyColumns()`**

Metode ini digunakan untuk mendeteksi entitas dalam a DynamicFrame. Ia mengembalikan peta di mana kunci adalah nama kolom dan nilai-nilai daftar jenis entitas terdeteksi. Custom masking dapat dilakukan setelah ini dikembalikan dalam AWS Glue script.

```
classifyColumns(frame: DynamicFrame,
                entityTypesToDetect: Seq[String],
                sampleFraction: Double = 0.1,
                thresholdFraction: Double = 0.1,
                detectionSensitivity: String = "LOW")
```

Parameter:

- frame — (type:`DynamicFrame`) Input DynamicFrame yang berisi data yang akan diproses.
- entityTypesToDetect — (type:`Seq[String]`) Daftar tipe entitas yang akan dideteksi. Dapat berupa Jenis Entitas Terkelola atau Jenis Entitas Kustom.
- SampleFraction — (type:`Double`, default: 10%) Fraksi data yang akan diambil sampel saat memindai entitas PII.
- ThresholdFraction — (type:`Double`, default: 10%): Fraksi data yang harus dipenuhi agar kolom dapat diidentifikasi sebagai data PII.
- DetectionSensitivity — (type:`String`, options: “LOW” atau “HIGH”, default: “LOW”) Menentukan sensitivitas proses deteksi. Opsi yang valid adalah “RENDAH” atau “TINGGI”. Jika tidak disediakan, sensitivitas default diatur ke “RENDAH”.

Jenis Data Sensitif Terkelola

Entitas global

Tipe data	Kategori	Deskripsi
PERSON_NAME	Universal	Nama orang tersebut.
Email	Personal	Alamat emailnya.
IP_ALAMAT	Komputer	Alamat IP

Tipe data	Kategori	Deskripsi
ALAMAT MAC_	Personal	Alamat MAC.

Tipe data AS

Tipe data	Deskripsi
BANK_ACCOUNT	Nomor rekening bank. Namun, tidak spesifik untuk negara atau wilayah, hanya format akun AS dan Kanada yang terdeteksi.
KARTU KREDIT	Nomor kartu kredit.
TELEPON_NOMOR	Nomor teleponnya. Namun, tidak spesifik untuk suatu negara atau wilayah, hanya nomor telepon AS dan Kanada yang terdeteksi saat ini.
USA_ATIN	Nomor Identifikasi Wajib Pajak Adopsi AS yang dikeluarkan oleh Internal Revenue Service.
USA_CPT_CODE	Kode CPT (khusus AS).
USA_DEA_NUMBER	Nomor DEA (khusus AS).
USA_DRIVING_LICENSE	Nomor SIM (khusus AS).
USA_HCPCS_CODE	Kode HCPCS (khusus AS).
USA_HEALTH_INSURANCE_CLAIM_NUMBER	Nomor Klaim Asuransi Kesehatan (khusus AS).
USA_ITIN	ITIN (untuk orang atau entitas AS).
USA_MEDICARE_BENEFICIARY_IDENTIFIER	Pengidentifikasi Penerima Medicare (khusus AS).

Tipe data	Deskripsi
USA_NATIONAL_DRUG_CODE	Kode NDC (khusus AS).
USA_NATIONAL_PROVIDER_IDENTIFIER	Nomor Pengenal Penyedia Nasional (khusus AS).
USA_PASSPORT_NUMBER	Nomor paspor (untuk orang AS).
USA_PTIN	Nomor Identifikasi Pajak Preparer AS yang dikeluarkan oleh Internal Revenue Service.
USA_SSN	Nomor jaminan sosial (untuk orang AS).

Tipe data Argentina

Tipe data	Deskripsi
ARGENTINA_TAX_IDENTIFICATIO N_NUMBER	Nomor Identifikasi Pajak Argentina. Juga dikenal sebagai CUIT atau CUIL.

Jenis data Australia

Tipe data	Deskripsi
AUSTRALIA_BUSINESS_NUMBER	Nomor Bisnis Australia (ABN). Pengenal unik yang dikeluarkan oleh Australian Business Register (ABR) untuk mengidentifikasi bisnis kepada pemerintah dan masyarakat.
AUSTRALIA_COMPANY_NUMBER	Nomor Perusahaan Australia (ACN). Pengident ifikasi unik yang dikeluarkan oleh Komisi Sekuritas dan Investasi Australia.
AUSTRALIA_DRIVING_LICENSE	Nomor SIM untuk Australia.

Tipe data	Deskripsi
AUSTRALIA_MEDICARE_NUMBER	Nomor Medicare Australia. Pengenal pribadi yang dikeluarkan oleh Komisi Asuransi Kesehatan Australia.
AUSTRALIA_PASSPORT_NUMBER	Nomor paspor Australia.
AUSTRALIA_TAX_FILE_NUMBER	Nomor Berkas Pajak Australia (TFN). Dikeluarkan oleh Australian Taxation Office (ATO) kepada wajib pajak (perorangan, perusahaan, dll) untuk transaksi pajak.

Tipe data Austria

Tipe data	Deskripsi
AUSTRIA_DRIVING_LICENSE	Nomor SIM (khusus Austria).
AUSTRIA_PASSPORT_NUMBER	Nomor paspor (khusus Austria).
AUSTRIA_SSN	Nomor jaminan sosial (untuk orang Austria).
AUSTRIA_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Austria).
AUSTRIA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Austria).

Tipe data Balkan

Tipe data	Deskripsi
BOSNIA_UNIQUE_MASTER_CITIZEN_NUMBER	Nomor warga negara master unik (JMBG) untuk warga negara Bosnia-Herzegovina.
KOSOVO_UNIQUE_MASTER_CITIZEN_NUMBER	Nomor warga master unik (JMBG) untuk Kosovo.

Tipe data	Deskripsi
MACEDONIA_UNIQUE_MASTER_CITIZEN_NUMBER	Nomor warga negara master unik untuk Makedonia.
MONTENEGRO_UNIQUE_MASTER_CITIZEN_NUMBER	Nomor warga negara master unik (JMBG) untuk Montenegro.
SERBIA_UNIQUE_MASTER_CITIZEN_NUMBER	Nomor warga negara master unik (JMBG) untuk Serbia.
SERBIA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Serbia).
VOJVODINA_UNIQUE_MASTER_CITIZEN_NUMBER	Nomor warga master unik (JMBG) untuk Vojvodina.

Tipe data Belgia

Tipe data	Deskripsi
BELGIUM_DRIVING_LICENSE	Nomor SIM (khusus Belgia).
BELGIUM_NATIONAL_IDENTIFICATION_NUMBER	Nomor Nasional Belgia (BNN).
BELGIUM_PASSPORT_NUMBER	Nomor paspor (khusus Belgia).
BELGIUM_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Belgia).
BELGIUM_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Belgia).

Tipe data Brazil

Tipe data	Deskripsi
BRASIL_BANK_ACCOUNT	Nomor rekening bank (khusus Brasil).

Tipe data	Deskripsi
BRAZIL_NATIONAL_IDENTIFICATION_NUMBER	Pengidentifikasi nasional (khusus Brasil).
BRAZIL_NATIONAL_REGISTRY_OF_LEGAL_ENTITIES_NUMBER	Nomor identifikasi yang dikeluarkan untuk perusahaan (khusus Brasil), juga dikenal sebagai CNPJ.
BRAZIL_NATURAL_PERSON_REGISTRY_NUMBER	Nomor Pendaftaran Orang Alami, juga dikenal sebagai CPF.

Tipe data Bulgaria

Tipe data	Deskripsi
BULGARIA_DRIVING_LICENSE	Nomor SIM (khusus Bulgaria).
BULGARIA_UNIFORM_CIVIL_NUMBER	Nomor Sipil Terpadu (EGN) yang berfungsi sebagai nomor identifikasi nasional.
BULGARIA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Bulgaria).

Tipe data Kanada

Tipe data	Deskripsi
CANADA_DRIVING_LICENSE	Nomor SIM (khusus Kanada).
CANADA_GOVERNMENT_IDENTIFICATION_CARD_NUMBER	Pengidentifikasi nasional (khusus Kanada).
CANADA_PASSPORT_NUMBER	Nomor paspor (khusus Kanada).
CANADA_PERMANENT_RESIDENCE_NUMBER	Nomor tempat tinggal permanen (nomor kartu PR).

Tipe data	Deskripsi
CANADA_PERSONAL_HEALTH_NUMBER	Pengidentifikasi unik untuk perawatan kesehatan (nomor PHN).
CANADA_SOCIAL_INSURANCE_NUMBER	Nomor asuransi sosial (SIN) di Kanada.

Tipe data Chili

Tipe data	Deskripsi
CHILE_DRIVING_LICENSE	Nomor SIM (khusus Chili).
CHILE_NATIONAL_IDENTIFICATION_NUMBER	Pengidentifikasi nasional Chili, juga dikenal sebagai RUT atau RUN.

Tipe data China, Hong Kong, Makau, dan Taiwan

Tipe data	Deskripsi
CHINA_IDENTIFIKASI_	Pengidentifikasi China.
CHINA_LICENSE_PLATE_NUMBER	Nomor SIM (khusus China).
CHINA_MAINLAND_TRAVEL_PERMIT_ID_HONG_KONG_MACAU	Izin Perjalanan Daratan untuk Penduduk Hong Kong dan Makau.
CHINA_MAINLAND_TRAVEL_PERMIT_ID_TAIWAN	Izin Perjalanan Daratan untuk Penduduk Taiwan dikeluarkan oleh Pemerintah Republik Rakyat Tiongkok (RRC).
CHINA_PASSPORT_NUMBER	Nomor paspor (khusus China).
CHINA_PHONE_NUMBER	Nomor telepon (khusus China).
HONG_KONG_IDENTITY_CARD	Dokumen identitas resmi yang dikeluarkan oleh Departemen Imigrasi Hong Kong.

Tipe data	Deskripsi
MACAU_RESIDENT_IDENTITY_CARD	Kartu Identitas Penduduk Macau atau BIR adalah kartu identitas resmi yang dikeluarkan oleh Biro Layanan Identifikasi Makau.
TAIWAN_NATIONAL_IDENTIFICATION_NUMBER	Pengidentifikasi nasional (khusus Taiwan).
TAIWAN_PASSPORT_NUMBER	Nomor paspor (khusus Taiwan).

Tipe data Kolombia

Tipe data	Deskripsi
KOLOMBIA_PERSONAL_IDENTIFICATION_NUMBER	Pengidentifikasi unik yang diberikan kepada orang Kolombia saat lahir.
KOLOMBIA_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Kolombia).

Tipe data Kroasia

Tipe data	Deskripsi
CROATIA_DRIVING_LICENSE	Nomor SIM (khusus Kroasia).
CROATIA_IDENTITY_NUMBER	Pengidentifikasi nasional (khusus Kroasia).
CROATIA_PASSPORT_NUMBER	Nomor paspor (khusus Kroasia).
CROATIA_PERSONAL_IDENTIFICATION_NUMBER	Nomor pengenal pribadi (OIB).

Tipe data Siprus

Tipe data	Deskripsi
CYPRUS_DRIVING_LICENSE	Nomor SIM (khusus Siprus).
CYPRUS_NATIONAL_IDENTIFICATION_NUMBER	Kartu identitas Siprus.
CYPRUS_PASSPORT_NUMBER	Nomor paspor (khusus Siprus).
CYPRUS_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Siprus).
CYPRUS_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Siprus).

Tipe data Ceko

Tipe data	Deskripsi
CZECHIA_DRIVING_LICENSE	Nomor SIM (khusus Ceko).
CZECHIA_PERSONAL_IDENTIFICATION_NUMBER	Nomor pengenal pribadi (khusus Ceko).
CZECHIA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Ceko).

Tipe data Denmark

Tipe data	Deskripsi
DENMARK_DRIVING_LICENSE	Nomor SIM (khusus Denmark).
DENMARK_PERSONAL_IDENTIFICATION_NUMBER	Nomor pengenal pribadi (khusus Denmark).
DENMARK_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Denmark).
DENMARK_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Denmark).

Tipe data Estonia

Tipe data	Deskripsi
ESTONIA_DRIVING_LICENSE	Nomor SIM (khusus Estonia).
ESTONIA_PASSPORT_NUMBER	Nomor paspor (khusus Estonia).
ESTONIA_PERSONAL_IDENTIFICATION_CODE	Nomor pengenal pribadi (khusus Estonia).
ESTONIA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Estonia).

Tipe data Finlandia

Tipe data	Deskripsi
FINLAND_DRIVING_LICENSE	Nomor SIM (khusus Finlandia).
FINLAND_HEALTH_INSURANCE_NUMBER	Nomor asuransi kesehatan (khusus Finlandia).
FINLAND_NATIONAL_IDENTIFICATION_NUMBER	Nomor pengenal nasional (khusus Finlandia).
FINLAND_PASSPORT_NUMBER	Nomor paspor (khusus Finlandia).
FINLAND_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Finlandia).

Tipe data Prancis

Tipe data	Deskripsi
FRANCE_BANK_ACCOUNT	Nomor rekening bank (khusus Prancis).
FRANCE_DRIVING_LICENSE	Nomor SIM (khusus Prancis).
FRANCE_HEALTH_INSURANCE_NUMBER	Nomor asuransi kesehatan Prancis.

Tipe data	Deskripsi
FRANCE_INSEE_CODE	Jaminan sosial Prancis, SSN, atau nomor NIR.
FRANCE_NATIONAL_IDENTIFICATION_NUMBER	Nomor pengenal nasional Prancis (CNI).
FRANCE_PASSPORT_NUMBER	Nomor paspor (khusus Prancis).
FRANCE_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Prancis).
FRANCE_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Prancis).

Tipe data Jerman

Tipe data	Deskripsi
GERMANY_BANK_ACCOUNT	Nomor rekening bank (khusus Jerman).
GERMANY_DRIVING_LICENSE	Nomor SIM (khusus Jerman).
JERMAN_PASSPORT_NUMBER	Nomor paspor (khusus Jerman).
GERMANY_PERSONAL_IDENTIFICATION_NUMBER	Nomor identifikasi pribadi (khusus Jerman).
GERMANY_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Jerman).
GERMANY_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Jerman).

Tipe data Yunani

Tipe data	Deskripsi
GREECE_DRIVING_LICENSE	Nomor SIM (khusus Yunani).
GREECE_PASSPORT_NUMBER	Nomor paspor (khusus Yunani).

Tipe data	Deskripsi
GREECE_SSN	Nomor jaminan sosial (untuk orang Yunani).
GREECE_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Yunani).
GREECE_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Yunani).

Tipe data Hongaria

Tipe data	Deskripsi
HUNGARY_DRIVING_LICENSE	Nomor SIM (khusus Hongaria).
HUNGARY_PASSPORT_NUMBER	Nomor paspor (khusus Hongaria).
HUNGARIA_SSN	Nomor jaminan sosial (untuk orang Hongaria).
HUNGARY_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Hongaria).
HUNGARY_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Hongaria).

Tipe data Islandia

Tipe data	Deskripsi
ICELAND_NATIONAL_IDENTIFICATION_NUMBER	Pengidentifikasi nasional (khusus Islandia).
ICELAND_PASSPORT_NUMBER	Nomor paspor (khusus Islandia).
ICELAND_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Islandia).

Tipe data India

Tipe data	Deskripsi
INDIA_AADHAAR_NUMBER	Nomor identifikasi Aadhaar yang dikeluarkan oleh Otoritas Identifikasi Unik India.
INDIA_PERMANENT_ACCOUNT_NUMBER	Nomor Rekening Permanen India (PAN).

Tipe data Indonesia

Tipe data	Deskripsi
INDONESIA_IDENTITY_CARD_NUMBER	Pengenal nasional (khusus Indonesia).

Tipe data Irlandia

Tipe data	Deskripsi
IRELAND_DRIVING_LICENSE	Nomor SIM (khusus Irlandia).
IRLANDIA_PASSPORT_NUMBER	Nomor paspor (khusus Irlandia).
IRELAND_PERSONAL_PUBLIC_SERVICE_NUMBER	Nomor layanan publik pribadi Irlandia (PPS).
IRRELAND_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Irlandia).
IRELAND_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Irlandia).

Tipe data Israel

Tipe data	Deskripsi
ISRAEL_IDENTIFICATION_NUMBER	Pengidentifikasi nasional (khusus Israel).

Tipe data Italia

Tipe data	Deskripsi
ITALY_BANK_ACCOUNT	Nomor rekening bank (khusus Italia).
ITALY_DRIVING_LICENSE	Nomor SIM (khusus Italia).
ITALY_FISCAL_CODE	Nomor pengenal, juga dikenal sebagai Italian Codice Fiscale.
ITALY_PASSPORT_NUMBER	Nomor paspor (khusus Italia).
ITALY_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Italia).

Tipe data Jepang

Tipe data	Deskripsi
AKUN JAPAN_BANK_	Rekening bank Jepang.
JAPAN_DRIVING_LICENSE	Nomor SIM untuk Jepang
JAPAN_MY_NUMBER	Pengenal unik untuk warga negara Jepang atau perusahaan yang digunakan untuk administrasi pajak, administrasi jaminan sosial, dan tanggap bencana
JAPAN_PASSPORT_NUMBER	Nomor passort Jepang.

Tipe data Korea

Tipe data	Deskripsi
KOREA_PASSPORT_NUMBER	Nomor paspor (khusus Korea).
KOREA_RESIDENCE_REGISTRATIO N_NUMBER_FOR_CITIZENS	Nomor pendaftar tempat tinggal Korea untuk penduduk.

Tipe data	Deskripsi
KOREA_RESIDENCE_REGISTRATIO N_NUMBER_FOR_FOREIGNERS	Nomor pendaftar tempat tinggal Korea untuk orang asing.

Tipe data Latvia

Tipe data	Deskripsi
LATVIA_DRIVING_LICENSE	Nomor SIM (khusus Latvia).
LATVIA_PASSPORT_NUMBER	Nomor paspor (khusus Latvia).
LATVIA_PERSONAL_IDENTIFICAT ION_NUMBER	Nomor pengenal pribadi (khusus Latvia).
LATVIA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Latvia).

Tipe data Liechtenstein

Tipe data	Deskripsi
LIECHTENSTEIN_NATIONAL_IDENTIFICATIO N_NUMBER	Pengidentifikasi nasional (khusus Liechtens tein).
LIECHTENSTEIN_PASSPORT_NUMBER	Nomor paspor (khusus Liechtenstein).
LIECHTENSTEIN_TAX_IDENTIFIC ATION_NUMBER	Nomor identifikasi pajak (khusus Liechtenstein).

Tipe data Lithuania

Tipe data	Deskripsi
LITHUANIA_DRIVING_LICENSE	Nomor SIM (khusus Lituania).

Tipe data	Deskripsi
LITHUANIA_PERSONAL_IDENTIFICATION_NUMBER	Nomor pengenal pribadi (khusus Lituania).
LITHUANIA_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Lituania).
LITHUANIA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Lituania).

Tipe data Luksemburg

Tipe data	Deskripsi
LUXEMBOURG_DRIVING_LICENSE	Nomor SIM (khusus Luksemburg).
LUXEMBOURG_NATIONAL_INDIVIDUAL_NUMBER	Pengidentifikasi nasional (khusus Luksemburg).
LUXEMBOURG_PASSPORT_NUMBER	Nomor paspor (khusus Luksemburg).
LUXEMBOURG_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Luksemburg).
LUXEMBOURG_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Luksemburg).

Tipe data Malaysia

Tipe data	Deskripsi
MALAYSIA_MYKAD_NUMBER	Pengenal nasional (khusus Malaysia).
MALAYSIA_PASSPORT_NUMBER	Nomor paspor (khusus Malaysia).

Tipe data Malta

Tipe data	Deskripsi
MALTA_DRIVING_LICENSE	Nomor SIM (khusus Malta).
MALTA_NATIONAL_IDENTIFICATION_NUMBER	Pengidentifikasi nasional (khusus Malta).
MALTA_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Malta).
MALTA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Malta).

Tipe data Meksiko

Tipe data	Deskripsi
MEXICO_CLABE_NUMBER	Meksiko CLABE (Clave Bancaria Estandarizada) nomor bank).
MEXICO_DRIVING_LICENSE	Nomor SIM (khusus Meksiko).
MEXICO_PASSPORT_NUMBER	Nomor paspor (khusus Meksiko).
MEXICO_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Meksiko).
MEXICO_UNIQUE_POPULATION_REGISTRY_CODE	Kode identitas unik Clave Única de Registro de Población (CURP) untuk Meksiko.

Tipe data Belanda

Tipe data	Deskripsi
BELANDA_CITIZEN_SERVICE_NUMBER	Nomor warga negara Belanda (BSN, burgerservicenummer).
BELANDA_DRIVING_LICENSE	Nomor SIM (khusus Belanda).
BELANDA_PASSPORT_NUMBER	Nomor paspor (khusus Belanda).

Tipe data	Deskripsi
BELANDA_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Belanda).
BELANDA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Belanda).
NETHERLANDS_BANK_ACCOUNT	Nomor rekening bank (khusus Belanda).

Tipe data Selandia Baru

Tipe data	Deskripsi
NEW_ZEALAND_DRIVING_LICENSE	Nomor SIM (khusus Selandia Baru).
NEW_ZEALAND_NATIONAL_HEALTH_INDEX_NUMBER	Nomor indeks kesehatan nasional Selandia Baru.
NEW_ZEALAND_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak, juga dikenal sebagai nomor pendapatan pedalaman (khusus Selandia Baru).

Tipe data Norwegia

Tipe data	Deskripsi
NORWAY_BIRTH_NUMBER	Nomor identitas nasional Norwegia.
NORWAY_DRIVING_LICENSE	Nomor SIM (khusus Norwegia).
NORWAY_HEALTH_INSURANCE_NUMBER	Nomor asuransi kesehatan Norwegia.
NORWAY_NATIONAL_IDENTIFICATION_NUMBER	Nomor pengenal nasional (khusus Norwegia).
NORWAY_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Norwegia).

Tipe data Filipina

Tipe data	Deskripsi
FILIPINA_DRIVING_LICENSE	Nomor SIM (khusus Filipina).
FILIPINA_PASSPORT_NUMBER	Nomor paspor (khusus Filipina).

Tipe data Polandia

Tipe data	Deskripsi
POLAND_DRIVING_LICENSE	Nomor SIM (khusus Polandia).
POLAND_IDENTIFICATION_NUMBER	Pengidentifikasi Polandia.
POLAND_PASSPORT_NUMBER	Nomor paspor (khusus Polandia).
POLAND_REGON_NUMBER	Nomor pengenal REGON, juga dikenal sebagai Nomor Identifikasi Statistik.
POLAND_SSN	Nomor jaminan sosial (untuk orang Polandia).
POLAND_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Polandia).
POLAND_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Polandia).

Tipe data Portugal

Tipe data	Deskripsi
PORTUGAL_DRIVING_LICENSE	Nomor SIM (khusus Portugal).
PORTUGAL_NATIONAL_IDENTIFIC ATION_NUMBER	Nomor pengenal nasional (khusus Portugal).
PORTUGAL_PASSPORT_NUMBER	Nomor paspor (khusus Portugal).
PORTUGAL_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Portugal).

Tipe data	Deskripsi
PORTUGAL_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Portugal).

Tipe data Rumania

Tipe data	Deskripsi
ROMANIA_DRIVING_LICENSE	Nomor SIM (khusus Rumania).
RUMANIA_NUMERICAL_PERSONAL_CODE	Nomor pengenal pribadi (khusus Rumania).
RUMANIA_PASSPORT_NUMBER	Nomor paspor (khusus Rumania).
RUMANIA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Rumania).

Tipe data Singapura

Tipe data	Deskripsi
SINGAPORE_DRIVING_LICENSE	Nomor SIM (khusus Singapura).
SINGAPORE_NATIONAL_REGISTRY _IDENTIFICATION_NUMBER	Kartu identitas pendaftaran nasional untuk Singapura.
SINGAPORE_PASSPORT_NUMBER	Nomor paspor (khusus Singapura).
SINGAPORE_UNIQUE_ENTITY_NUMBER	Nomor Entitas Unik untuk Singapura.

Tipe data Slovakia

Tipe data	Deskripsi
SLOVAKIA_DRIVING_LICENSE	Nomor SIM (khusus Slovakia).
SLOVAKIA_NATIONAL_IDENTIFIC ATION_NUMBER	Nomor pengenal nasional (khusus Slovakia).

Tipe data	Deskripsi
SLOVAKIA_PASSPORT_NUMBER	Nomor paspor (khusus Slovakia).
SLOVAKIA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Slovakia).

Tipe data Slovenia

Tipe data	Deskripsi
SLOVENIA_DRIVING_LICENSE	Nomor SIM (khusus Slovenia).
SLOVENIA_PASSPORT_NUMBER	Nomor paspor (khusus Slovenia).
SLOVENIA_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak (khusus Slovenia).
SLOVENIA_UNIQUE_MASTER_CITIZEN_NUMBER	Nomor warga negara master unik (JMBG) untuk warga negara Slovenia.
SLOVENIA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Slovenia).

Tipe data Afrika Selatan

Tipe data	Deskripsi
SOUTH_AFRICA_PERSONAL_IDENTIFICATION_NUMBER	Nomor pengenal pribadi (khusus South Sfrica).

Tipe data Spanyol

Tipe data	Deskripsi
SPAIN_BANK_ACCOUNT	Nomor rekening bank (khusus Spanyol).
SPANYOL_DNI	Kartu identitas nasional (Documento Nacional de Identidad) Spanyol.

Tipe data	Deskripsi
SPAIN_DRIVING_LICENSE	Nomor SIM (khusus Spanyol).
SPAIN_NIE	Nomor identitas orang asing (khusus Spanyol), juga dikenal sebagai NIE.
SPANYOL_NIF	Nomor identifikasi pajak (khusus Spanyol), juga dikenal sebagai NIF.
SPAIN_PASSPORT_NUMBER	Nomor paspor (khusus Spanyol).
SPAIN_SSN	Nomor jaminan sosial (untuk orang Spanyol).
SPAIN_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Spanyol).

Tipe data Sri Lanka

Tipe data	Deskripsi
SRI_LANKA_NATIONAL_IDENTIFICATION_NUMBER	Pengidentifikasi nasional (khusus Sri Lanka).

Tipe data Swedia

Tipe data	Deskripsi
SWEDEN_DRIVING_LICENSE	Nomor SIM (khusus Swedia).
SWEDEN_PASSPORT_NUMBER	Nomor paspor (khusus Swedia).
SWEDEN_PERSONAL_IDENTIFICATION_NUMBER	Nomor pengenal nasional (khusus Swedia).
SWEDEN_TAX_IDENTIFICATION_NUMBER	Nomor identifikasi pajak Swedia (personnummer).
SWEDEN_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Swedia).

Tipe data Swiss

Tipe data	Deskripsi
SWITZERLAND_AHV	Nomor jaminan sosial untuk orang Swiss (AHV).
SWITZERLAND_HEALTH_INSURANCE_NUMBER	Nomor asuransi kesehatan Swiss.
SWITZERLAND_PASSPORT_NUMBER	Nomor paspor (khusus Swiss).
SWITZERLAND_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Swiss).

Tipe data Thailand

Tipe data	Deskripsi
THAILAND_PASSPORT_NUMBER	Nomor paspor (khusus Thailand).
THAILAND_PERSONAL_IDENTIFICATION_NUMBER	Nomor pengenal pribadi (khusus Thailand).

Tipe data Turki

Tipe data	Deskripsi
TURKEY_NATIONAL_IDENTIFICATION_NUMBER	Nomor pengenal nasional (khusus Turki).
TURKI_PASSPORT_NUMBER	Nomor paspor (khusus Turki).
TURKEY_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Turki).

Tipe data Ukraina

Tipe data	Deskripsi
UKRAINE_INDIVIDUAL_IDENTIFICATION_NUMBER	Pengidentifikasi unik (khusus Ukraina).
UKRAINE_PASSPORT_NUMBER_DOMESTIC	Nomor paspor domestik (khusus Ukraina).
UKRAINE_PASSPORT_NUMBER_INTERNATIONAL	Nomor paspor internasional (khusus Ukraina).

Tipe data Uni Emirat Arab (UEA)

Tipe data	Deskripsi
UNITED_ARAB_EMIRATES_PERSONAL_NUMBER	Nomor pengenal pribadi (khusus UEA).

Tipe data Inggris

Tipe data	Deskripsi
UK_BANK_ACCOUNT	Rekening bank Inggris Raya (Inggris).
UK_BANK_SORT_CODE	Kode sortir bank Inggris Raya (Inggris). Kode sortir adalah kode bank yang digunakan untuk merutekan transfer uang antar bank di negara masing-masing melalui organisasi izin masing-masing.
UK_DRIVING_LICENSE	Nomor SIM untuk Inggris Raya dan Irlandia Utara (khusus Inggris)
UK_ELECTORAL_ROLL_NUMBER	Electoral Roll Number (ERN) adalah nomor identifikasi yang dikeluarkan untuk seseorang untuk pendaftaran pemilu Inggris. Format

Tipe data	Deskripsi
	nomor ini ditentukan oleh Standar Pemerintah Inggris dari Kantor Kabinet Inggris.
UK_NATIONAL_HEALTH_SERVICE_NUMBER	Nomor National Health Service (NHS) adalah nomor unik yang dialokasikan untuk pengguna terdaftar layanan kesehatan masyarakat di Inggris.
UK_NATIONAL_INSURANCE_NUMBER	Nomor Asuransi Nasional (NINO) adalah nomor yang digunakan di Inggris (Inggris) untuk mengidentifikasi seseorang untuk program asuransi nasional atau sistem jaminan sosial. Nomor ini kadang-kadang disebut sebagai NI No atau NINO.
UK_PASSPORT_NUMBER	Nomor paspor Inggris Raya (Inggris).
UK_UNIQUE_TAXPAYER_REFERENCE_NUMBER	Nomor Referensi Wajib Pajak Unik (UTR) Inggris Raya (Inggris). Pengidentifikasi yang digunakan oleh pemerintah Inggris untuk mengelola sistem perpajakan.
UK_VALUE_ADDED_TAX	PPN adalah pajak konsumsi yang ditanggung oleh konsumen akhir. PPN dibayarkan untuk setiap transaksi dalam proses manufaktur dan distribusi. Untuk Inggris, nomor PPN dikeluarkan oleh kantor PPN untuk wilayah tempat bisnis tersebut didirikan.
UK_PHONE_NUMBER	Nomor telepon Inggris Raya (UK).

Tipe data Venezuela

Tipe data	Deskripsi
VENEZUELA_DRIVING_LICENSE	Nomor SIM (khusus Venezuela).

Tipe data	Deskripsi
VENEZUELA_NATIONAL_IDENTIFICATION_NUMBER	Nomor pengenal nasional (khusus Venezuela).
VENEZUELA_VALUE_ADDED_TAX	Pajak Pertambahan Nilai (khusus Venezuela).

Menggunakan deteksi data sensitif berbutir halus

> **Note**
>
> Tindakan berbutir halus hanya tersedia di AWS Glue 3.0 dan 4.0. Ini termasuk AWS Glue Studio pengalaman. Perubahan log audit persisten juga tidak tersedia di 2.0.
> Semua pekerjaan visual AWS Glue Studio 3.0 dan 4.0 akan memiliki skrip yang dibuat yang secara otomatis menggunakan API tindakan berbutir halus.

Transformasi Detect Sensitive Data menyediakan kemampuan untuk mendeteksi, menutupi, atau menghapus entitas yang Anda tentukan, atau ditentukan sebelumnya oleh AWS Glue. Tindakan berbutir halus lebih lanjut memungkinkan Anda menerapkan tindakan spesifik per entitas. Manfaat tambahan meliputi:

- Peningkatan kinerja sebagai tindakan diterapkan segera setelah data terdeteksi.
- Pilihan untuk menyertakan atau mengecualikan kolom tertentu.
- Kemampuan untuk menggunakan masking sebagian. Ini memungkinkan Anda untuk menutupi entitas data sensitif yang terdeteksi sebagian, daripada menutupi seluruh string. Kedua parameter sederhana dengan offset dan regex didukung.

Berikut ini adalah cuplikan kode API deteksi data sensitif dan tindakan halus yang digunakan dalam pekerjaan sampel yang direferensikan di bagian berikutnya.

Deteksi API — tindakan berbutir halus menggunakan parameter baru: `detectionParameters`

```
def detect(
    frame: DynamicFrame,
    detectionParameters: JsonOptions,
    outputColumnName: String = "DetectedEntities",
    detectionSensitivity: String = "LOW"
```

```
): DynamicFrame = {}
```

Menggunakan API Deteksi Data Sensitif dengan tindakan halus

API deteksi data sensitif menggunakan deteksi menganalisis data yang diberikan, menentukan apakah baris atau kolom adalah Jenis Entitas Data Sensitif, dan akan menjalankan tindakan yang ditentukan oleh pengguna untuk setiap jenis Entitas.

Menggunakan API deteksi dengan tindakan berbutir halus

Gunakan API deteksi dan tentukan `outputColumnName` dan `detectionParameters`.

```
   object GlueApp {
     def main(sysArgs: Array[String]) {

       val spark: SparkContext = new SparkContext()
       val glueContext: GlueContext = new GlueContext(spark)

       // @params: [JOB_NAME]
       val args = GlueArgParser.getResolvedOptions(sysArgs, Seq("JOB_NAME").toArray)
       Job.init(args("JOB_NAME"), glueContext, args.asJava)

       // Script generated for node S3 bucket. Creates DataFrame from data stored in
S3.
       val S3bucket_node1 =
glueContext.getSourceWithFormat(formatOptions=JsonOptions("""{"quoteChar":
"\"", "withHeader": true, "separator": ",", "optimizePerformance": false}"""),
connectionType="s3", format="csv", options=JsonOptions("""{"paths":
["s3://189657479688-ddevansh-pii-test-bucket/tiny_pii.csv"], "recurse": true}"""),
transformationContext="S3bucket_node1").getDynamicFrame()

       // Script generated for node Detect Sensitive Data. Will run detect API for the
DataFrame
       // detectionParameter contains information on which EntityType are being
detected
       // and what actions are being applied to them when detected.
       val DetectSensitiveData_node2 = EntityDetector.detect(
           frame = S3bucket_node1,
           detectionParameters = JsonOptions(
            """
               {
```

```
                    "PHONE_NUMBER": [
                        {
                            "action": "PARTIAL_REDACT",
                            "actionOptions": {
                                "numLeftCharsToExclude": "3",
                                "numRightCharsToExclude": "4",
                                "redactChar": "#"
                            },
                            "sourceColumnsToExclude": [ "Passport No", "DL NO#" ]
                        }
                    ],
                    "USA_PASSPORT_NUMBER": [
                        {
                            "action": "SHA256_HASH",
                            "sourceColumns": [ "Passport No" ]
                        }
                    ],
                    "USA_DRIVING_LICENSE": [
                        {
                            "action": "REDACT",
                            "actionOptions": {
                                "redactText": "USA_DL"
                            },
                            "sourceColumns": [ "DL NO#" ]
                        }
                    ]

                }
            """
            ),
            outputColumnName = "DetectedEntities"
        )

        // Script generated for node S3 bucket. Store Results of detect to S3 location
        val S3bucket_node3 = glueContext.getSinkWithFormat(connectionType="s3",
 options=JsonOptions("""{"path": "s3://189657479688-ddevansh-pii-test-bucket/
test-output/", "partitionKeys": []}"""), transformationContext="S3bucket_node3",
 format="json").writeDynamicFrame(DetectSensitiveData_node2)

        Job.commit()
      }
```

Skrip di atas akan membuat DataFrame dari lokasi di Amazon S3 dan kemudian akan menjalankan API. `detect` Karena `detect` API memerlukan bidang `detectionParameters` (peta nama entitas ke daftar semua pengaturan tindakan yang akan digunakan untuk entitas itu) diwakili oleh `JsonOptions` objek AWS Glue, itu juga akan memungkinkan kita untuk memperluas fungsionalitas API.

Untuk setiap tindakan yang ditentukan per entitas, masukkan daftar semua nama kolom untuk menerapkan kombinasi entitas/tindakan. Ini memungkinkan Anda untuk menyesuaikan entitas untuk mendeteksi setiap kolom dalam kumpulan data Anda dan melewati entitas yang Anda tahu tidak berada di kolom tertentu. Ini juga memungkinkan pekerjaan Anda menjadi lebih berkinerja dengan tidak melakukan deteksi yang tidak perlu memanggil entitas tersebut dan memungkinkan Anda untuk melakukan tindakan yang unik untuk setiap kolom dan kombinasi entitas.

Melihat lebih dekat`detectionParameters`, ada tiga jenis entitas dalam pekerjaan sampel. Ini adalah`Phone Number`,`USA_PASSPORT_NUMBER`, dan`USA_DRIVING_LICENSE`. Untuk masing-masing jenis entitas ini AWS Glue akan menjalankan tindakan yang berbeda yaitu`PARTIAL_REDACT`,`SHA256_HASH`,`REDACT`, dan`DETECT`. Masing-masing Jenis Entitas juga `sourceColumns` harus berlaku untuk dan/atau `sourceColumnsToExclude` jika terdeteksi.

> **Note**
>
> Hanya satu edit-in-place tindakan (`PARTIAL_REDACT`,`SHA256_HASH`, atau`REDACT`) yang dapat digunakan per kolom tetapi `DETECT` tindakan dapat digunakan dengan salah satu tindakan ini.

`detectionParameters`Bidang ini memiliki tata letak di bawah ini:

```
ENTITY_NAME -> List[Actions]
{
 "ENTITY_NAME": [{
  Action, // required
  ColumnSpecs,
  ActionOptionsMap
    }],
    "ENTITY_NAME2": [{
  ...
    }]
}
```

Jenis-jenis `actions` dan `actionOptions` tercantum di bawah ini:

```
DETECT
{
    # Required
    "action": "DETECT",
    # Optional, depending on action chosen
    "actionOptions": {
        // There are no actionOptions for DETECT
    },
    # 1 of below required, both can also used
    "sourceColumns": [
        "COL_1", "COL_2", ..., "COL_N"
    ],
    "sourceColumnsToExclude": [
        "COL_5"
    ]
}

SHA256_HASH
{
    # Required
    "action": "SHA256_HASH",
    # Required or optional, depending on action chosen
    "actionOptions": {
        // There are no actionOptions for SHA256_HASH
    },

    # 1 of below required, both can also used
    "sourceColumns": [
        "COL_1", "COL_2", ..., "COL_N"
    ],
    "sourceColumnsToExclude": [
        "COL_5"
    ]
}

REDACT
{
    # Required
    "action": "REDACT",
```

```
    # Required or optional, depending on action chosen
    "actionOptions": {
        // The text that is being replaced
        "redactText": "USA_DL"
    },

    # 1 of below required, both can also used
    "sourceColumns": [
        "COL_1", "COL_2", ..., "COL_N"
    ],
    "sourceColumnsToExclude": [
        "COL_5"
    ]
}

PARTIAL_REDACT
{
    # Required
    "action": "PARTIAL_REDACT",
    # Required or optional, depending on action chosen
    "actionOptions": {
        // number of characters to not redact from the left side
        "numLeftCharsToExclude": "3",
        // number of characters to not redact from the right side
        "numRightCharsToExclude": "4",
        // the partial redact will be made with this redacted character
        "redactChar": "#",
        // regex pattern for partial redaction
        "matchPattern": "[0-9]"
    },

    # 1 of below required, both can also used
    "sourceColumns": [
        "COL_1", "COL_2", ..., "COL_N"
    ],
    "sourceColumnsToExclude": [
        "COL_5"
    ]
}
```

Setelah skrip berjalan, hasilnya akan dikeluarkan ke lokasi Amazon S3 yang diberikan. Anda dapat melihat data Anda di Amazon S3 tetapi dengan jenis entitas yang dipilih menjadi sensitif berdasarkan tindakan yang dipilih. Dalam kasus ini, kita akan memiliki baris yang akan terlihat seperti ini:


```
{
    "Name": "Colby Schuster",
    "Address": "39041 Antonietta Vista, South Rodgerside, Nebraska 24151",
    "Car Owned": "Fiat",
    "Email": "Kitty46@gmail.com",
    "Company": "O'Reilly Group",
    "Job Title": "Dynamic Functionality Facilitator",
    "ITIN": "991-22-2906",
    "Username": "Cassandre.Kub43",
    "SSN": "914-22-2906",
    "DOB": "2020-08-27",
    "Phone Number": "1-2#######1718",
    "Bank Account No": "69741187",
    "Credit Card Number": "6441-6289-6867-2162-2711",
    "Passport No": "94f311e93a623c72ccb6fc46cf5f5b0265ccb42c517498a0f27fd4c43b47111e",
    "DL NO#": "USA_DL"
}
```


Dalam naskah di atas, sebagian `Phone Number` disunting dengan#. `Passport No`Itu diubah menjadi hash SHA256. `DL NO#` Itu terdeteksi sebagai nomor SIM AS dan disunting menjadi “USA_DL” seperti yang dinyatakan dalam. `detectionParameters`

> **Note**
>
> ClassifyColumns API tidak tersedia untuk digunakan dengan tindakan berbutir halus karena sifat API. API ini melakukan sampling kolom (disesuaikan oleh pengguna tetapi memiliki nilai default) untuk melakukan deteksi lebih cepat. Tindakan berbutir halus memerlukan iterasi atas setiap nilai karena alasan ini.

Log Audit Persisten

Fitur baru yang diperkenalkan dengan tindakan berbutir halus (tetapi juga tersedia saat menggunakan API normal) adalah adanya log audit persisten. Saat ini, menjalankan API deteksi menambahkan kolom tambahan (default ke `DetectedEntities` tetapi dapat disesuaikan

melaluioutputColumnName) parameter dengan metadata deteksi PII. Ini sekarang memiliki kunci metadata “ActionUsed”, yang merupakan salah satu dariDETECT,,,. PARTIAL_REDACT SHA256_HASH REDACT

```
"DetectedEntities": {
    "Credit Card Number": [
        {
            "entityType": "CREDIT_CARD",
            "actionUsed": "DETECT",
            "start": 0,
            "end": 19
        }
    ],
    "Phone Number": [
        {
            "entityType": "PHONE_NUMBER",
            "actionUsed": "REDACT",
            "start": 0,
            "end": 14
        }
    ]
}
```

Bahkan pelanggan yang menggunakan API tanpa tindakan halus seperti `detect(entityTypesToDetect, outputColumnName)` akan melihat log audit persisten ini di kerangka data yang dihasilkan.

Pelanggan yang menggunakan API dengan tindakan halus akan melihat semua tindakan, terlepas dari apakah tindakan tersebut disunting atau tidak. Contoh:

```
+--------------------+---------------
+-------------------------------------------------------------------------------------------------
+
| Credit Card Number  |  Phone Number  |
                                              DetectedEntities
                                                                  |
+--------------------+---------------
+-------------------------------------------------------------------------------------------------
+
| 622126741306XXXX    | +12#####7890   | {"Credit Card Number":
[{"entityType":"CREDIT_CARD","actionUsed":"PARTIAL_REDACT","start":0,"end":16}],"Phone
```

```
 Number":
[{"entityType":"PHONE_NUMBER","actionUsed":"PARTIAL_REDACT","start":0,"end":12}]}} |
| 6221 2674 1306 XXXX | +12#######7890 | {"Credit Card Number":
[{"entityType":"CREDIT_CARD","actionUsed":"PARTIAL_REDACT","start":0,"end":19}],"Phone
 Number":
[{"entityType":"PHONE_NUMBER","actionUsed":"PARTIAL_REDACT","start":0,"end":14}]}} |
| 6221-2674-1306-XXXX | 22#######7890  | {"Credit Card Number":
[{"entityType":"CREDIT_CARD","actionUsed":"PARTIAL_REDACT","start":0,"end":19}],"Phone
 Number":
[{"entityType":"PHONE_NUMBER","actionUsed":"PARTIAL_REDACT","start":0,"end":14}]}} |
+--------------------+---------------
+-------------------------------------------------------------------------------------------------
+
```

Jika Anda tidak ingin melihat DetectedEntitieskolom, Anda cukup menjatuhkan kolom tambahan dalam skrip khusus.

AWS GlueVisual Job API

AWS Gluemenyediakan API yang memungkinkan pelanggan membuat pekerjaan integrasi data menggunakan AWS Glue API dari objek JSON yang mewakili alur kerja langkah visual. Pelanggan kemudian dapat menggunakan editor visual AWS Glue Studio untuk bekerja dengan pekerjaan ini.

Untuk informasi selengkapnya tentang tipe data Visual Job API, lihat Visual Job API.

Topik

- Desain API dan API CRUD
- Mulai
- Keterbatasan pekerjaan visual

Desain API dan API CRUD

UpdateJob API CreateJob dan sekarang mendukung parameter opsional tambahan, codeGenConfiguration Node. Menyediakan struktur JSON yang tidak kosong untuk bidang ini akan mengakibatkan DAG terdaftar AWS Glue Studio untuk pekerjaan yang dibuat dan kode terkait yang dihasilkan. Nilai nol atau string kosong untuk bidang ini pada pembuatan pekerjaan akan diabaikan.

Pembaruan pada bidang codeGenConfiguration Nodes akan dilakukan melalui UpdateJob AWS Glue API dengan cara yang sama seperti CreateJob. Seluruh bidang harus ditentukan di UpdateJob

mana DAG telah diubah sesuai keinginan. Nilai nol yang diberikan akan diabaikan dan tidak ada pembaruan untuk DAG yang akan dilakukan. Struktur atau string yang kosong akan menyebabkan codeGenConfiguration Node disetel sebagai kosong dan DAG sebelumnya dihapus. GetJob API akan mengembalikan DAG jika ada. DeleteJob API juga akan menghapus DAG terkait.

Mulai

Untuk membuat pekerjaan, gunakan CreateJob tindakan. Input `CreateJob` permintaan akan memiliki bidang tambahan 'codeGenConfigurationNodes' di mana Anda bisa menentukan objek DAG di JSON.

Hal-hal yang perlu diingat:

- Bidang 'codeGenConfigurationNodes' adalah peta nodeId ke node.
- Setiap node dimulai dengan kunci yang mengidentifikasi jenis node apa itu.
- Hanya ada satu kunci yang ditentukan karena node hanya bisa dari satu jenis.
- Bidang input berisi node induk dari node saat ini.

Berikut ini adalah representasi JSON dari CreateJobinput.

```
{
  "node-1": {
    "S3CatalogSource": {
      "Table": "csvFormattedTable",
      "PartitionPredicate": "",
      "Name": "S3 bucket",
      "AdditionalOptions": {},
      "Database": "myDatabase"
    }
  },
  "node-3": {
    "S3DirectTarget": {
      "Inputs": ["node-2"],
      "PartitionKeys": [],
      "Compression": "none",
      "Format": "json",
      "SchemaChangePolicy": { "EnableUpdateCatalog": false },
      "Path": "",
      "Name": "S3 bucket"
    }
```

```
  },
  "node-2": {
    "ApplyMapping": {
      "Inputs": ["node-1"],
      "Name": "ApplyMapping",
      "Mapping": [
        {
          "FromType": "long",
          "ToType": "long",
          "Dropped": false,
          "ToKey": "myheader1",
          "FromPath": ["myheader1"]
        },
        {
          "FromType": "long",
          "ToType": "long",
          "Dropped": false,
          "ToKey": "myheader2",
          "FromPath": ["myheader2"]
        },
        {
          "FromType": "long",
          "ToType": "long",
          "Dropped": false,
          "ToKey": "myheader3",
          "FromPath": ["myheader3"]
        }
      ]
    }
  }
}
```

Memperbarui dan mendapatkan pekerjaan

Karena juga UpdateJobakan memiliki bidang 'codeGenConfigurationNodes', format input akan sama. Lihat UpdateJobAksi.

GetJobTindakan akan mengembalikan bidang 'codeGenConfigurationNodes' dalam format yang sama juga. Lihat GetJobAksi.

Keterbatasan pekerjaan visual

Karena parameter 'codeGenConfigurationNodes' telah ditambahkan ke API yang ada, batasan apa pun dalam API tersebut akan diwariskan. Selain itu, codeGenConfiguration Node dan beberapa node akan dibatasi ukurannya. Lihat Job Structure untuk informasi lebih lanjut.

Pemrograman skrip Ray

AWS Gluemembuatnya mudah untuk menulis dan menjalankan skrip Ray. Bagian ini menjelaskan kemampuan Ray yang didukung yang tersedia AWS Glue untuk Ray. Anda memprogram skrip Ray dengan Python.

Skrip kustom Anda harus kompatibel dengan versi Ray yang ditentukan oleh `Runtime` bidang dalam definisi pekerjaan Anda. Untuk informasi selengkapnya tentang `Runtime` API Pekerjaan, lihatthe section called “Tugas”. Untuk informasi tentang setiap lingkungan runtime, lihatthe section called “Lingkungan runtime Ray yang didukung”.

Topik

- Tutorial: Menulis skrip ETL AWS Glue untuk Ray
- Menggunakan Ray Core dan Ray Data AWS Glue untuk Ray
- Menyediakan file dan pustaka Python untuk pekerjaan Ray
- Menghubungkan ke data dalam pekerjaan Ray

Tutorial: Menulis skrip ETL AWS Glue untuk Ray

Ray memberi Anda kemampuan untuk menulis dan menskalakan tugas terdistribusi secara native dengan Python. AWS Glueuntuk Ray menawarkan lingkungan Ray tanpa server yang dapat Anda akses dari pekerjaan dan sesi interaktif (sesi interaktif Ray dalam pratinjau). Sistem AWS Glue pekerjaan menyediakan cara yang konsisten untuk mengelola dan menjalankan tugas Anda—sesuai jadwal, dari pemicu, atau dari konsol. AWS Glue

Menggabungkan AWS Glue alat-alat ini menciptakan rantai alat yang kuat yang dapat Anda gunakan untuk mengekstrak, mengubah, dan memuat beban kerja (ETL), kasus penggunaan populer untuk. AWS Glue Dalam tutorial ini, Anda akan mempelajari dasar-dasar menyusun solusi ini.

Kami juga mendukung penggunaan Spark AWS Glue untuk beban kerja ETL Anda. Untuk tutorial tentang menulis skrip AWS Glue untuk Spark, lihatthe section called “Tutorial: Menulis skrip Spark”.

Untuk informasi lebih lanjut tentang mesin yang tersedia, lihatthe section called “AWS Glueuntuk Spark dan AWS Glue untuk Ray”. Ray mampu menangani berbagai jenis tugas dalam analitik, pembelajaran mesin (ML), dan pengembangan aplikasi.

Dalam tutorial ini, Anda akan mengekstrak, mengubah, dan memuat kumpulan data CSV yang di-host di Amazon Simple Storage Service (Amazon S3). Anda akan mulai dengan New York City Taxi and Limousine Commission (TLC) Trip Record Data Dataset, yang disimpan dalam ember Amazon S3 publik. Untuk informasi selengkapnya tentang kumpulan data ini, lihat Registry of Open Data pada AWS.

Anda akan mengubah data Anda dengan transformasi yang telah ditentukan yang tersedia di perpustakaan Ray Data. Ray Data adalah pustaka persiapan kumpulan data yang dirancang oleh Ray dan disertakan secara default AWS Glue untuk lingkungan Ray. Untuk informasi selengkapnya tentang pustaka yang disertakan secara default, lihatthe section called “Modul disediakan dengan pekerjaan Ray”. Anda kemudian akan menulis data yang diubah ke bucket Amazon S3 yang Anda kontrol.

Prasyarat - Untuk tutorial ini, Anda memerlukan AWS akun dengan akses ke dan Amazon AWS Glue S3.

Langkah 1: Buat bucket di Amazon S3 untuk menyimpan data keluaran Anda

Anda akan memerlukan bucket Amazon S3 yang Anda kontrol untuk berfungsi sebagai wastafel untuk data yang dibuat dalam tutorial ini. Anda dapat membuat ember ini dengan prosedur berikut.

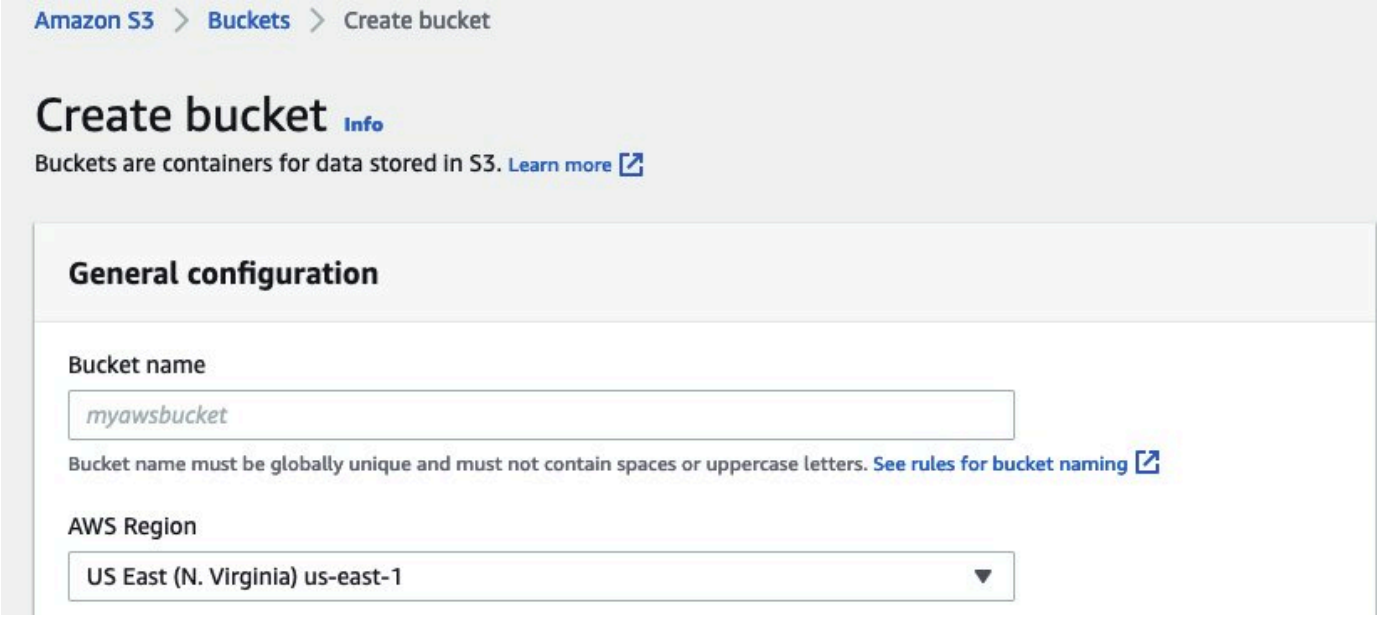
> **Note**
>
> Jika Anda ingin menulis data Anda ke bucket yang sudah ada yang Anda kontrol, Anda dapat melewati langkah ini. Perhatikan *`yourBucketName`*, nama bucket yang ada, untuk digunakan di langkah selanjutnya.

Untuk membuat bucket untuk output pekerjaan Ray Anda

- Buat bucket dengan mengikuti langkah-langkah dalam Membuat bucket di Panduan Pengguna Amazon S3.
  - Saat memilih nama ember, perhatikan *`yourBucketName`*, yang akan Anda rujuk di langkah selanjutnya.

- Untuk konfigurasi lain, pengaturan yang disarankan yang disediakan di konsol Amazon S3 akan berfungsi dengan baik dalam tutorial ini.

Sebagai contoh, kotak dialog pembuatan bucket mungkin terlihat seperti ini di konsol Amazon S3.



Langkah 2: Buat peran dan kebijakan IAM untuk pekerjaan Ray Anda

Pekerjaan Anda akan membutuhkan peran AWS Identity and Access Management (IAM) dengan yang berikut:

- Izin yang diberikan oleh kebijakan `AWSGlueServiceRole` terkelola. Ini adalah izin dasar yang diperlukan untuk menjalankan AWS Glue pekerjaan.
- `Read`izin tingkat akses untuk sumber daya `nyc-tlc/*` Amazon S3.
- `Write`izin tingkat akses untuk sumber daya *`yourBucketName`*`/*` Amazon S3.
- Hubungan kepercayaan yang memungkinkan kepala `glue.amazonaws.com` sekolah untuk mengambil peran.

Anda dapat membuat peran ini dengan prosedur berikut.

Untuk membuat peran IAM untuk pekerjaan Ray Anda AWS Glue

> **Note**
>
> Anda dapat membuat peran IAM dengan mengikuti banyak prosedur yang berbeda. Untuk informasi selengkapnya atau opsi tentang cara menyediakan sumber daya IAM, lihat AWS Identity and Access Managementdokumentasi.

1. Buat kebijakan yang menentukan izin Amazon S3 yang telah diuraikan sebelumnya dengan mengikuti langkah-langkah dalam Membuat kebijakan IAM (konsol) dengan editor visual di Panduan Pengguna IAM.
   - Saat memilih layanan, pilih Amazon S3.
   - Saat memilih izin untuk kebijakan Anda, lampirkan kumpulan tindakan berikut untuk sumber daya berikut (disebutkan sebelumnya):
     - Baca izin tingkat akses untuk sumber daya `nyc-tlc/*` Amazon S3.
     - Tulis izin tingkat akses untuk sumber daya *`yourBucketName`*`/*` Amazon S3.
   - Saat memilih nama kebijakan, perhatikan *`YourPolicyName`*, yang akan Anda rujuk di langkah selanjutnya.
2. Buat peran untuk pekerjaan Ray Anda AWS Glue dengan mengikuti langkah-langkah dalam Membuat peran untuk AWS layanan (konsol) di Panduan Pengguna IAM.
   - Saat memilih entitas AWS layanan tepercaya, pilih`Glue`. Ini secara otomatis akan mengisi hubungan kepercayaan yang diperlukan untuk pekerjaan Anda.
   - Saat memilih kebijakan untuk kebijakan izin, lampirkan kebijakan berikut:
     - `AWSGlueServiceRole`
     - *`YourPolicyName`*
   - Saat memilih nama peran, perhatikan *`YourRoleName`*, yang akan Anda rujuk di langkah selanjutnya.

Langkah 3: Buat dan jalankan pekerjaan AWS Glue untuk Ray

Pada langkah ini, Anda membuat AWS Glue pekerjaan menggunakanAWS Management Console, menyediakannya dengan skrip contoh, dan menjalankan pekerjaan. Ketika Anda membuat pekerjaan, itu menciptakan tempat di konsol bagi Anda untuk menyimpan, mengkonfigurasi, dan

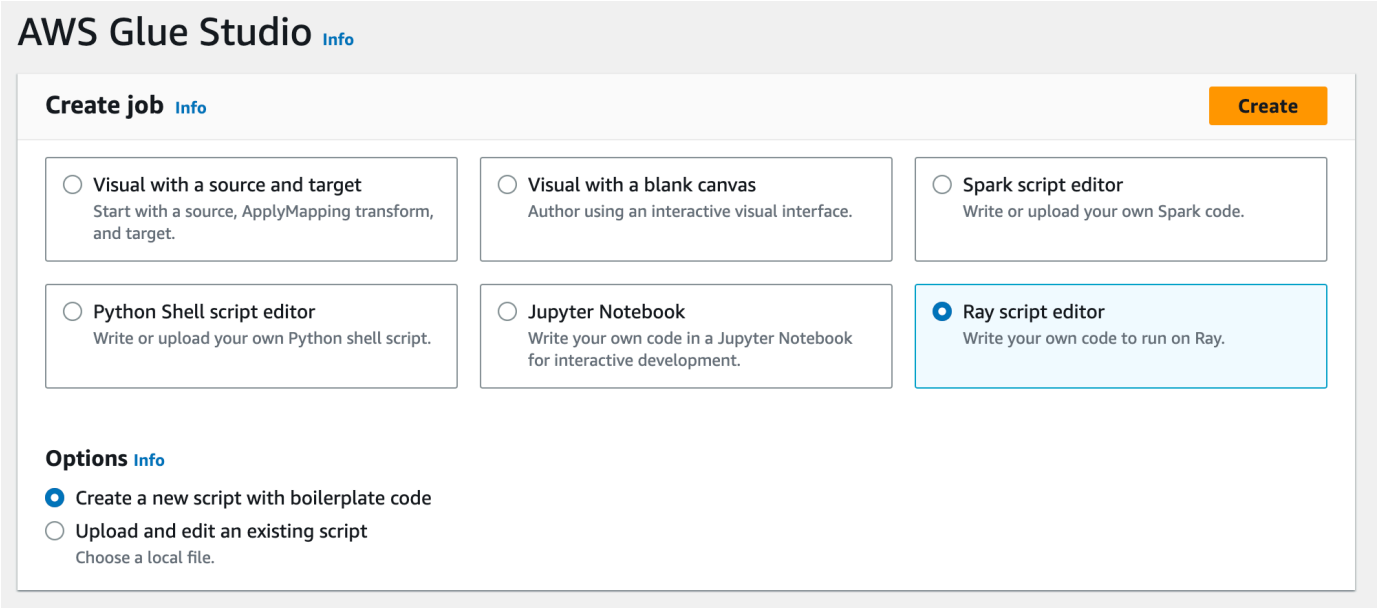
mengedit skrip Ray Anda. Untuk informasi selengkapnya tentang cara membuat tugas, lihat the section called “Masuk ke konsol”.

Dalam tutorial ini, kami membahas skenario ETL berikut: Anda ingin membaca catatan Januari 2022 dari kumpulan data Rekaman Perjalanan TLC Kota New York, menambahkan kolom baru (`tip_rate`) ke kumpulan data dengan menggabungkan data di kolom yang ada, lalu hapus sejumlah kolom yang tidak relevan dengan analisis Anda saat ini, dan kemudian Anda ingin menulis hasilnya. *yourBucketName* Skrip Ray berikut melakukan langkah-langkah ini:

```
import ray
import pandas
from ray import data

ray.init('auto')

ds = ray.data.read_csv("s3://nyc-tlc/opendata_repo/opendata_webconvert/yellow/
yellow_tripdata_2022-01.csv")

# Add the given new column to the dataset and show the sample record after adding a new
 column
ds = ds.add_column( "tip_rate", lambda df: df["tip_amount"] / df["total_amount"])

# Dropping few columns from the underlying Dataset
ds = ds.drop_columns(["payment_type", "fare_amount", "extra", "tolls_amount",
 "improvement_surcharge"])

ds.write_parquet("s3://yourBucketName/ray/tutorial/output/")
```

Untuk membuat dan menjalankan pekerjaan AWS Glue untuk Ray

1. DiAWS Management Console, navigasikan ke halaman AWS Glue arahan.
2. Di panel navigasi samping, pilih ETL Jobs.
3. Di Buat pekerjaan, pilih Editor skrip Ray, lalu pilih Buat, seperti pada ilustrasi berikut.



4. Tempelkan teks lengkap skrip ke panel Script, dan ganti teks yang ada.
5. Arahkan ke rincian Job dan atur properti IAM Role ke *YourRoleName*.
6. Pilih Simpan, lalu pilih Jalankan.

Langkah 4: Periksa output Anda

Setelah menjalankan AWS Glue pekerjaan Anda, Anda harus memvalidasi bahwa output sesuai dengan harapan skenario ini. Anda dapat melakukannya dengan prosedur berikut.

Untuk memvalidasi apakah pekerjaan Ray Anda berhasil

1. Di halaman detail pekerjaan, navigasikan ke Runs.
2. Setelah beberapa menit, Anda akan melihat run dengan status Run Succeeded.
3. Arahkan ke konsol Amazon S3 di https://console.aws.amazon.com/s3/ dan periksa. *yourBucketName* Anda akan melihat file yang ditulis ke bucket keluaran Anda.
4. Baca file Parket dan verifikasi isinya. Anda dapat melakukan ini dengan alat yang ada. Jika Anda tidak memiliki proses untuk memvalidasi file Parket, Anda dapat melakukannya di AWS Glue konsol dengan sesi AWS Glue interaktif, menggunakan Spark atau Ray (dalam pratinjau).

   Dalam sesi interaktif, Anda memiliki akses ke perpustakaan Ray Data, Spark, atau panda, yang disediakan secara default (berdasarkan pilihan mesin Anda). Untuk memverifikasi konten file, Anda dapat menggunakan metode pemeriksaan umum yang tersedia di pustaka tersebut—

metode seperti`count`,, dan. `schema show` Untuk informasi selengkapnya tentang sesi interaktif di konsol, lihat Menggunakan buku catatan dengan AWS Glue Studio dan AWS Glue.

Karena Anda telah mengkonfirmasi bahwa file telah ditulis ke bucket, Anda dapat mengatakan dengan pasti relatif bahwa jika output Anda memiliki masalah, mereka tidak terkait dengan konfigurasi IAM. Konfigurasikan sesi Anda *`yourRoleName`*untuk memiliki akses ke file yang relevan.

Jika Anda tidak melihat hasil yang diharapkan, periksa konten pemecahan masalah dalam panduan ini untuk mengidentifikasi dan memulihkan sumber kesalahan. Untuk menafsirkan status kesalahan job run, lihatthe section called “Status Job run”. Anda dapat menemukan konten pemecahan masalah di bagian ini. Pemecahan Masalah AWS Glue Untuk kesalahan spesifik yang terkait dengan pekerjaan Ray, lihat the section called “Memecahkan masalah kesalahan Ray” di bagian pemecahan masalah.

Langkah berikutnya

Anda sekarang telah melihat dan melakukan proses ETL menggunakan AWS Glue untuk Ray dari ujung ke ujung. Anda dapat menggunakan sumber daya berikut untuk memahami alat apa yang disediakan Ray AWS Glue untuk mengubah dan menafsirkan data Anda dalam skala besar.

- Untuk informasi selengkapnya tentang model tugas Ray, lihatthe section called “Menggunakan Ray Core dan Ray Data AWS Glue untuk Ray”. Untuk lebih banyak pengalaman dalam menggunakan tugas Ray, ikuti contoh dalam dokumentasi Ray Core. Lihat Ray Core: Ray Tutorials and Examples (2.4.0) dalam dokumentasi Ray.
- Untuk panduan tentang pustaka manajemen data yang tersedia AWS Glue untuk Ray, lihatthe section called “Menghubungkan ke data”. Untuk pengalaman lebih lanjut menggunakan Ray Data untuk mengubah dan menulis dataset, ikuti contoh dalam dokumentasi Ray Data. Lihat Data Sinar: Contoh (2.4.0).
- Untuk informasi selengkapnya tentang mengonfigurasi AWS Glue pekerjaan Ray, lihatthe section called “Bekerja dengan pekerjaan Ray”.
- Untuk informasi lebih lanjut tentang menulis AWS Glue skrip Ray, lanjutkan membaca dokumentasi di bagian ini.

Menggunakan Ray Core dan Ray Data AWS Glue untuk Ray

Ray adalah kerangka kerja untuk meningkatkan skrip Python dengan mendistribusikan karya di seluruh cluster. Anda dapat menggunakan Ray sebagai solusi untuk berbagai macam masalah, sehingga Ray menyediakan perpustakaan untuk mengoptimalkan tugas-tugas tertentu. DiAWS Glue, kami fokus menggunakan Ray untuk mengubah kumpulan data besar. AWS Gluemenawarkan dukungan untuk Ray Data dan bagian dari Ray Core untuk memfasilitasi tugas ini.

Apa itu Ray Core?

Langkah pertama membangun aplikasi terdistribusi adalah mengidentifikasi dan mendefinisikan pekerjaan yang dapat dilakukan secara bersamaan. Ray Core berisi bagian-bagian Ray yang Anda gunakan untuk menentukan tugas yang dapat dilakukan secara bersamaan. Ray menyediakan referensi dan informasi mulai cepat yang dapat Anda gunakan untuk mempelajari alat yang mereka sediakan. Untuk informasi lebih lanjut, lihat Apa itu Ray Core? dan Mulai Cepat Ray Core. Untuk informasi selengkapnya tentang mendefinisikan tugas bersamaan secara efektif di Ray, lihat Tips untuk pengguna pertama kali.

> **Tugas dan aktor Ray**
>
> AWS GlueUntuk dokumentasi Ray, kita mungkin merujuk pada tugas dan aktor, yang merupakan konsep inti dalam Ray.
> Ray menggunakan fungsi dan kelas Python sebagai blok bangunan dari sistem komputasi terdistribusi. Sama seperti ketika fungsi dan variabel Python menjadi “metode” dan “atribut” ketika digunakan di kelas, fungsi menjadi “tugas” dan kelas menjadi “aktor” ketika mereka digunakan di Ray untuk mengirim kode ke pekerja. Anda dapat mengidentifikasi fungsi dan kelas yang mungkin digunakan oleh Ray dengan `@ray.remote` anotasi.
> Tugas dan aktor dapat dikonfigurasi, mereka memiliki siklus hidup, dan mereka mengambil sumber daya komputasi sepanjang hidup mereka. Kode yang melempar kesalahan dapat ditelusuri kembali ke tugas atau aktor ketika Anda menemukan akar penyebab masalah. Dengan demikian, istilah-istilah ini mungkin muncul ketika Anda mempelajari cara mengonfigurasi, memantau, atau men-debug AWS Glue untuk pekerjaan Ray.
> Untuk mulai mempelajari cara menggunakan tugas dan aktor secara efektif untuk membangun aplikasi terdistribusi, lihat Konsep Utama dalam dokumen Ray.

Ray Core AWS Glue untuk Ray

AWS Glueuntuk lingkungan Ray mengelola pembentukan dan penskalaan cluster, serta mengumpulkan dan memvisualisasikan log. Karena kami mengelola masalah ini, akibatnya kami membatasi akses dan dukungan untuk API di Ray Core yang akan digunakan untuk mengatasi masalah ini di cluster sumber terbuka.

Di lingkungan `Ray2.4` runtime terkelola, kami tidak mendukung:

- CLI Inti Sinar
- CLI Negara Bagian Ray
- `ray.util.metrics`Metode utilitas metrik Prometheus:
  - Penghitung
  - Gauge
  - Histogram
- Alat debugging lainnya:
  - ray.util.pdb.set_trace
  - ray.util.inspect_serializability
  - garis waktu ray.

Apa itu Ray Data?

Saat Anda terhubung ke sumber data dan tujuan, menangani kumpulan data, dan memulai transformasi umum, Ray Data adalah metodologi langsung untuk menggunakan Ray untuk memecahkan masalah dalam mengubah kumpulan data Ray. Untuk informasi selengkapnya tentang penggunaan Ray Data, lihat Ray Datasets: Distributed Data Preprocessing.

Anda dapat menggunakan Ray Data atau alat lain untuk mengakses data Anda. Untuk informasi selengkapnya tentang mengakses data Anda di Ray, lihatthe section called “Menghubungkan ke data”.

Ray Data AWS Glue untuk Ray

Ray Data didukung dan disediakan secara default di lingkungan `Ray2.4` runtime terkelola. Untuk informasi selengkapnya tentang modul yang disediakan, lihatthe section called “Modul disediakan dengan pekerjaan Ray”.

Menyediakan file dan pustaka Python untuk pekerjaan Ray

Bagian ini memberikan informasi yang Anda butuhkan untuk menggunakan pustaka Python dengan AWS Glue pekerjaan Ray. Anda dapat menggunakan pustaka umum tertentu yang disertakan secara default di semua pekerjaan Ray. Anda juga dapat menyediakan pustaka Python Anda sendiri untuk pekerjaan Ray Anda.

Modul disediakan dengan pekerjaan Ray

Anda dapat melakukan alur kerja integrasi data dalam pekerjaan Ray dengan paket yang disediakan berikut. Paket-paket ini tersedia secara default di pekerjaan Ray.

AWS Glue version 4.0

Di AWS Glue 4.0, lingkungan Ray (`Ray2.4`runtime) menyediakan paket-paket berikut:

- boto3 == 1.26.133
- sinar == 2.4.0
- pyarrow == 11.0.0
- panda == 1.5.3
- numpy == 1.24.3
- fsspec == 2023.4.0

Daftar ini mencakup semua paket yang akan diinstal dengan`ray[data] == 2.4.0`. Ray Data didukung di luar kotak.

Menyediakan file untuk pekerjaan Ray Anda

Anda dapat memberikan file ke pekerjaan Ray Anda dengan `--working-dir` parameter. Berikan parameter ini dengan jalur ke file.zip yang dihosting di Amazon S3. Dalam file.zip, file Anda harus terkandung dalam satu direktori tingkat atas. Tidak ada file lain yang harus berada di tingkat atas.

File Anda akan didistribusikan ke setiap node Ray sebelum skrip Anda mulai berjalan. Pertimbangkan bagaimana hal ini dapat memengaruhi ruang disk yang tersedia untuk setiap node Ray. Ruang disk yang tersedia ditentukan oleh WorkerType set dalam konfigurasi pekerjaan. Jika Anda ingin memberikan data pekerjaan Anda dalam skala besar, mekanisme ini bukanlah solusi yang tepat. Untuk informasi selengkapnya tentang penyediaan data ke pekerjaan Anda, lihatthe section called “Menghubungkan ke data”.

File Anda akan dapat diakses seolah-olah direktori disediakan untuk Ray melalui `working_dir` parameter. Misalnya, untuk membaca file bernama `sample.txt` di direktori tingkat atas file.zip Anda, Anda dapat memanggil:

```
@ray.remote
def do_work():
    f = open("sample.txt", "r")
    print(f.read())
```

Untuk informasi selengkapnya`working_dir`, lihat dokumentasi Ray. Fitur ini berperilaku mirip dengan kemampuan asli Ray.

Modul Python tambahan untuk pekerjaan Ray

Modul tambahan dari PyPI

Pekerjaan Ray menggunakan Python Package Installer (pip3) untuk menginstal modul tambahan yang akan digunakan oleh skrip Ray. Anda dapat menggunakan `--pip-install` parameter dengan daftar modul Python yang dipisahkan koma untuk menambahkan modul baru atau mengubah versi modul yang ada.

Misalnya, untuk memperbarui atau menambahkan `scikit-learn` modul baru, gunakan pasangan kunci-nilai berikut:

`"--pip-install", "scikit-learn==0.21.3"`

Jika Anda memiliki modul khusus atau tambalan khusus, Anda dapat mendistribusikan pustaka Anda sendiri dari Amazon S3 dengan parameter. `--s3-py-modules` Sebelum mengunggah distribusi Anda, mungkin perlu dikemas ulang dan dibangun kembali. Ikuti pedoman di dalamthe section called “Termasuk kode Python dalam pekerjaan Ray”.

Distribusi khusus dari Amazon S3

Distribusi khusus harus mematuhi pedoman pengemasan Ray untuk dependensi. Anda dapat mengetahui cara membangun distribusi ini di bagian berikut. Untuk informasi selengkapnya tentang cara Ray mengatur dependensi, lihat Environment Dependencies dalam dokumentasi Ray.

Untuk menyertakan distribusi khusus setelah menilai kontennya, unggah yang dapat didistribusikan ke ember yang tersedia untuk peran IAM pekerjaan. Tentukan jalur Amazon S3 ke arsip zip Python dalam konfigurasi parameter Anda. Jika Anda menyediakan beberapa distributor, pisahkan dengan koma. Misalnya:

```
"--s3-py-modules", "s3://s3bucket/pythonPackage.zip"
```

Keterbatasan

Pekerjaan Ray tidak mendukung kompilasi kode asli di lingkungan kerja. Anda dapat dibatasi oleh ini jika dependensi Python Anda secara transitif bergantung pada kode asli yang dikompilasi. Pekerjaan Ray dapat menjalankan binari yang disediakan, tetapi mereka harus dikompilasi untuk Linux di ARM64. Ini berarti Anda mungkin dapat menggunakan isi `aarch64` `manylinux` roda. Anda dapat memberikan dependensi asli Anda dalam bentuk yang dikompilasi dengan mengemas ulang roda ke standar Ray. Biasanya, ini berarti menghapus `dist-info` folder sehingga hanya ada satu folder di root di arsip.

Anda tidak dapat memutakhirkan versi `ray` atau `ray[data]` menggunakan parameter ini. Untuk menggunakan versi baru Ray, Anda perlu mengubah bidang runtime pada pekerjaan Anda, setelah kami merilis dukungan untuk itu. Untuk informasi selengkapnya tentang versi Ray yang didukung, lihatthe section called “Versi AWS Glue”.

Termasuk kode Python dalam pekerjaan Ray

Yayasan Perangkat Lunak Python menawarkan perilaku standar untuk mengemas file Python untuk digunakan di berbagai runtime. Ray memperkenalkan batasan pada standar pengemasan yang harus Anda waspadai. AWS Gluetidak menentukan standar pengemasan di luar yang ditentukan untuk Ray. Petunjuk berikut memberikan panduan standar pada kemasan paket Python sederhana.

Package file Anda dalam `.zip` arsip. Direktori harus berada di root arsip. Seharusnya tidak ada file lain di tingkat root arsip, atau ini dapat menyebabkan perilaku yang tidak terduga. Direktori root adalah paket, dan namanya digunakan untuk merujuk ke kode Python Anda saat mengimpornya.

Jika Anda memberikan distribusi dalam formulir ini ke pekerjaan Ray`--s3-py-modules`, Anda akan dapat mengimpor kode Python dari paket Anda dalam skrip Ray Anda.

Paket Anda dapat menyediakan satu modul Python dengan beberapa file Python, atau Anda dapat mengemas banyak modul. Saat mengemas ulang dependensi, seperti pustaka dari PyPI, periksa file tersembunyi dan direktori metadata di dalam paket tersebut.

> **Warning**
>
> Perilaku OS tertentu membuat sulit untuk mengikuti instruksi pengemasan ini dengan benar.
>
> - OSX dapat menambahkan file tersembunyi seperti `__MACOSX` ke file zip Anda di tingkat atas.

- Windows dapat menambahkan file Anda ke folder di dalam zip secara otomatis, secara tidak sengaja membuat folder bersarang.

Prosedur berikut mengasumsikan Anda berinteraksi dengan file Anda di Amazon Linux 2 atau OS serupa yang menyediakan distribusi Info-ZIP `zip` dan utilitas. `zipinfo` Kami merekomendasikan menggunakan alat ini untuk mencegah perilaku yang tidak terduga.

Untuk mengemas file Python untuk digunakan di Ray

1. Buat direktori sementara dengan nama paket Anda, lalu konfirmasikan direktori kerja Anda adalah direktori induknya. Anda dapat melakukan ini dengan perintah berikut:

```
cd parent_directory
mkdir temp_dir
```

2. Salin file Anda ke direktori sementara, lalu konfirmasikan struktur direktori Anda. Isi direktori ini akan langsung diakses sebagai modul Python Anda. Anda dapat melakukan ini dengan perintah berikut:

```
ls -AR temp_dir
# my_file_1.py
# my_file_2.py
```

3. Kompres folder sementara Anda menggunakan zip. Anda dapat melakukan ini dengan perintah berikut:

```
zip -r zip_file.zip temp_dir
```

4. Konfirmasikan file Anda dikemas dengan benar. *`zip_file`*`.zip`sekarang harus ditemukan di direktori kerja Anda. Anda dapat memeriksanya dengan perintah berikut:

```
zipinfo -1 zip_file.zip
# temp_dir/
# temp_dir/my_file_1.py
# temp_dir/my_file_2.py
```

Untuk mengemas ulang paket Python untuk digunakan di Ray.

1. Buat direktori sementara dengan nama paket Anda, lalu konfirmasikan direktori kerja Anda adalah direktori induknya. Anda dapat melakukan ini dengan perintah berikut:

```
cd parent_directory
mkdir temp_dir
```

2. Dekompresi paket Anda dan salin isinya ke direktori sementara Anda. Hapus file yang terkait dengan standar kemasan Anda sebelumnya, hanya menyisakan konten modul. Konfirmasikan struktur file Anda terlihat benar dengan perintah berikut:

```
ls -AR temp_dir
# my_module
# my_module/__init__.py
# my_module/my_file_1.py
# my_module/my_submodule/__init__.py
# my_module/my_submodule/my_file_2.py
# my_module/my_submodule/my_file_3.py
```

3. Kompres folder sementara Anda menggunakan zip. Anda dapat melakukan ini dengan perintah berikut:

```
zip -r zip_file.zip temp_dir
```

4. Konfirmasikan file Anda dikemas dengan benar. *zip_file*`.zip`sekarang harus ditemukan di direktori kerja Anda. Anda dapat memeriksanya dengan perintah berikut:

```
zipinfo -1 zip_file.zip
# temp_dir/my_module/
# temp_dir/my_module/__init__.py
# temp_dir/my_module/my_file_1.py
# temp_dir/my_module/my_submodule/
# temp_dir/my_module/my_submodule/__init__.py
# temp_dir/my_module/my_submodule/my_file_2.py
# temp_dir/my_module/my_submodule/my_file_3.py
```

Menghubungkan ke data dalam pekerjaan Ray

AWS GluePekerjaan Ray dapat menggunakan beragam paket Python yang dirancang agar Anda dapat mengintegrasikan data dengan cepat. Kami menyediakan satu set dependensi minimal agar

tidak mengacaukan lingkungan Anda. Untuk informasi selengkapnya tentang apa yang disertakan secara default, lihatthe section called “Modul disediakan dengan pekerjaan Ray”.

> **Note**
>
> AWS Glueekstrak, transformasi, dan muat (ETL) menyediakan DynamicFrame abstraksi untuk merampingkan alur kerja ETL tempat Anda menyelesaikan perbedaan skema antar baris dalam kumpulan data Anda. AWS Glue ETL menyediakan fitur tambahan—bookmark pekerjaan dan pengelompokan file input. Saat ini kami tidak menyediakan fitur yang sesuai dalam pekerjaan Ray.
> AWS Glueuntuk Spark menyediakan dukungan langsung untuk menghubungkan ke format data tertentu, sumber dan sink. Di Ray, AWS SDK untuk panda dan pustaka pihak ketiga saat ini secara substansif mencakup kebutuhan itu. Anda perlu berkonsultasi dengan perpustakaan tersebut untuk memahami kemampuan apa yang tersedia.

AWS Glueuntuk integrasi Ray dengan Amazon VPC saat ini tidak tersedia. Sumber daya di Amazon VPC tidak akan dapat diakses tanpa rute umum. Untuk informasi selengkapnya tentang penggunaan AWS Glue dengan Amazon VPC, lihat. the section called “Titik akhir VPC (AWS PrivateLink)”

Pustaka umum untuk bekerja dengan data di Ray

Ray Data — Ray Data menyediakan metode untuk menangani format data umum, sumber dan sink. Untuk informasi selengkapnya tentang format dan sumber yang didukung di Ray Data, lihat Input/Output dalam dokumentasi Ray Data. Ray Data adalah pustaka berpendirian, bukan pustaka tujuan umum, untuk menangani kumpulan data.

Ray memberikan panduan tertentu seputar kasus penggunaan di mana Ray Data mungkin menjadi solusi terbaik untuk pekerjaan Anda. Untuk informasi selengkapnya, lihat Kasus penggunaan Ray dalam dokumentasi Ray.

AWSSDK untuk panda (awswrangler) - AWS SDK untuk panda adalah AWS produk yang memberikan solusi yang bersih dan teruji untuk membaca dan menulis ke AWS layanan saat transformasi Anda mengelola data dengan panda. DataFrames Untuk informasi selengkapnya tentang format dan sumber yang didukung di AWS SDK untuk panda, lihat Referensi API di dokumentasi AWS SDK untuk panda.

Untuk contoh cara membaca dan menulis data dengan AWS SDK untuk panda, lihat Mulai Cepat di dokumentasi AWS SDK untuk panda. AWSSDK untuk panda tidak menyediakan transformasi untuk data Anda. Ini hanya memberikan dukungan untuk membaca dan menulis dari sumber.

Modin - Modin adalah pustaka Python yang mengimplementasikan operasi panda umum dengan cara yang dapat didistribusikan. Untuk informasi lebih lanjut tentang Modin, lihat dokumentasi Modin. Modin sendiri tidak memberikan dukungan untuk membaca dan menulis dari sumber. Ini menyediakan implementasi terdistribusi dari transformasi umum. Modin didukung oleh AWS SDK untuk panda.

Saat Anda menjalankan Modin dan AWS SDK untuk panda bersama-sama di lingkungan Ray, Anda dapat melakukan tugas ETL umum dengan hasil kinerja. Untuk informasi selengkapnya tentang penggunaan Modin dengan AWS SDK untuk panda, lihat Pada skala di dokumentasi AWS SDK untuk panda.

Kerangka kerja lainnya — Untuk informasi selengkapnya tentang kerangka kerja yang didukung Ray, lihat Ekosistem Sinar dalam dokumentasi Ray. Kami tidak menyediakan dukungan untuk kerangka kerja lain di AWS Glue untuk Ray.

Menghubungkan ke data melalui Katalog Data

Mengelola data Anda melalui Katalog Data bersama dengan pekerjaan Ray didukung dengan AWS SDK untuk panda. Untuk informasi selengkapnya, lihat Glue Catalog di AWS situs SDK for pandas.

Menggunakan layanan ini dengan AWS SDK

AWS kit pengembangan perangkat lunak (SDK) tersedia untuk banyak bahasa pemrograman populer. Setiap SDK menyediakan API, contoh kode, dan dokumentasi yang memudahkan developer untuk membangun aplikasi dalam bahasa pilihan mereka.

Dokumentasi SDK	Contoh kode
AWS SDK for C++	AWS SDK for C++ contoh kode
AWS CLI	AWS CLI contoh kode
AWS SDK for Go	AWS SDK for Go contoh kode
AWS SDK for Java	AWS SDK for Java contoh kode
AWS SDK for JavaScript	AWS SDK for JavaScript contoh kode
AWS SDK for Kotlin	AWS SDK for Kotlin contoh kode
AWS SDK for .NET	AWS SDK for .NET contoh kode
AWS SDK for PHP	AWS SDK for PHP contoh kode
AWS Tools for PowerShell	Alat untuk contoh PowerShell kode
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) contoh kode
AWS SDK for Ruby	AWS SDK for Ruby contoh kode
AWS SDK for Rust	AWS SDK for Rust contoh kode
AWS SDK untuk SAP ABAP	AWS SDK untuk SAP ABAP contoh kode
AWS SDK for Swift	AWS SDK for Swift contoh kode

Untuk contoh khusus untuk layanan ini, lihat AWS Glue Contoh kode API menggunakan AWS SDK.

Ketersediaan contoh

Tidak dapat menemukan apa yang Anda butuhkan? Minta contoh kode menggunakan tautan Berikan umpan balik di bagian bawah halaman ini.

AWS Glue API

Bagian ini menjelaskan tipe data dan primitif yang digunakan oleh AWS Glue SDK dan Tools. Ada tiga cara umum untuk berinteraksi dengan AWS Glue pemrograman di luar AWS Management Console, masing-masing dengan dokumentasinya sendiri:

- Pustaka SDK Bahasa memungkinkan Anda mengakses AWS sumber daya dari bahasa pemrograman umum. Temukan informasi lebih lanjut di Alat untuk Dibangun AWS.
- AWS CLI Ini memungkinkan Anda untuk mengakses AWS sumber daya dari baris perintah. Temukan informasi lebih lanjut di Referensi AWS CLI Perintah.
- AWS CloudFormation memungkinkan Anda untuk menentukan satu set sumber AWS daya yang akan disediakan bersama secara konsisten. Temukan informasi lebih lanjut di AWS CloudFormation: referensi tipe AWS Glue sumber daya.

Dokumen bagian ini berbagi primitif secara independen dari SDK dan Alat ini. Alat menggunakan Referensi API AWS Glue Web untuk berkomunikasi dengan AWS.

Daftar Isi

- SerDeInfo struktur
- Struktur pesanan
- SkewedInfo struktur
- TableVersion struktur
- TableError struktur
- TableVersionError struktur
- SortCriterion struktur
- TableIdentifier struktur
- KeySchemaElement struktur
- PartitionIndex struktur
- PartitionIndexDescriptor struktur
- BackfillError struktur
- IcebergInput struktur
- OpenTableFormatInput struktur
- ViewDefinition struktur
- ViewDefinitionInput struktur
- ViewRepresentation struktur
- ViewRepresentationInput struktur
- Operasi
- CreateTable tindakan (Python: create_table)
- UpdateTable tindakan (Python: update_table)
- DeleteTable tindakan (Python: delete_table)
- BatchDeleteTable tindakan (Python: batch_delete_table)
- GetTable tindakan (Python: get_table)
- GetTables tindakan (Python: get_tables)
- GetTableVersion tindakan (Python: get_table_version)
- GetTableVersions tindakan (Python: get_table_versions)
- DeleteTableVersion tindakan (Python: delete_table_version)
- BatchDeleteTableVersion tindakan (Python: batch_delete_table_version)
- SearchTables tindakan (Python: search_tables)

- GetPartitionIndexes tindakan (Python: get_partition_indexes)
- CreatePartitionIndex tindakan (Python: create_partition_index)
- DeletePartitionIndex tindakan (Python: delete_partition_index)
- GetColumnStatisticsForTable tindakan (Python: get_column_statistics_for_table)
- UpdateColumnStatisticsForTable tindakan (Python: update_column_statistics_for_table)
- DeleteColumnStatisticsForTable tindakan (Python: delete_column_statistics_for_table)

- API partisi
  - Jenis data
  - Struktur partisi
  - PartitionInput struktur
  - PartitionSpecWithSharedStorageDescriptor struktur
  - PartitionListComposingSpec struktur
  - PartitionSpecProxy struktur
  - PartitionValueList struktur
  - Struktur segmen
  - PartitionError struktur
  - BatchUpdatePartitionFailureEntry struktur
  - BatchUpdatePartitionRequestEntry struktur
  - StorageDescriptor struktur
  - SchemaReference struktur
  - SerDeInfo struktur
  - SkewedInfo struktur
  - Operasi
  - CreatePartition tindakan (Python: create_partition)
  - BatchCreatePartition tindakan (Python: batch_create_partition)
  - UpdatePartition tindakan (Python: update_partition)
  - DeletePartition tindakan (Python: delete_partition)
  - BatchDeletePartition tindakan (Python: batch_delete_partition)
  - GetPartition tindakan (Python: get_partition)
  - GetPartitions tindakan (Python: get_partitions)

- Jenis Data
- Struktur pengklasifikasi
- GrokClassifier struktur
- Struktur XMLClassifier
- JsonClassifier struktur
- CsvClassifier struktur
- CreateGrokClassifierRequest struktur
- UpdateGrokClassifierRequest struktur
- struktur ClassifierRequest CreateXML
- struktur ClassifierRequest UpdateXML
- CreateJsonClassifierRequest struktur
- UpdateJsonClassifierRequest struktur
- CreateCsvClassifierRequest struktur
- UpdateCsvClassifierRequest struktur
- Operasi
- CreateClassifier tindakan (Python: create_classifier)
- DeleteClassifier tindakan (Python: delete_classifier)
- GetClassifier tindakan (Python: get_classifier)
- GetClassifiers tindakan (Python: get_classifiers)
- UpdateClassifier tindakan (Python: update_classifier)

- API Crawler
  - Jenis data
  - Struktur perayap
  - Struktur jadwal
  - CrawlerTargets struktur
  - Struktur S3Target
  - Struktur S3 DeltaCatalogTarget
  - Struktur S3 DeltaDirectTarget
  - JdbcTarget struktur
  - Struktur MongoDBTarget

- Struktur DynamodbTarget
- DeltaTarget struktur
- IcebergTarget struktur
- HudiTarget struktur
- CatalogTarget struktur
- CrawlerMetrics struktur
- CrawlerHistory struktur
- CrawlsFilter struktur
- SchemaChangePolicy struktur
- LastCrawlInfo struktur
- RecrawlPolicy struktur
- LineageConfiguration struktur
- LakeFormationConfiguration struktur
- Operasi
- CreateCrawler tindakan (Python: create_crawler)
- DeleteCrawler tindakan (Python: delete_crawler)
- GetCrawler tindakan (Python: get_crawler)
- GetCrawlers tindakan (Python: get_crawlers)
- GetCrawlerMetrics tindakan (Python: get_crawler_metrics)
- UpdateCrawler tindakan (Python: update_crawler)
- StartCrawler tindakan (Python: start_crawler)
- StopCrawler tindakan (Python: stop_crawler)
- BatchGetCrawlers tindakan (Python: batch_get_crawlers)
- ListCrawlers tindakan (Python: list_crawlers)
- ListCrawls tindakan (Python: list_crawls)

- API statistik kolom
  - Jenis Data
  - ColumnStatisticsTaskRun struktur
  - ColumnStatisticsTaskRunningException struktur
  - ColumnStatisticsTaskNotRunningException struktur

- Struktur JDBC ConnectorOptions
- StreamingDataPreviewOptions struktur
- AthenaConnectorSource struktur
- Struktur JDBC ConnectorSource
- SparkConnectorSource struktur
- CatalogSource struktur
- Struktur MySQL CatalogSource
- Struktur CatalogSource PostgreSQL
- Struktur CatalogSource OracleSQL
- Struktur ServerCatalogSource MicrosoftSQL
- CatalogKinesisSource struktur
- DirectKinesisSource struktur
- KinesisStreamingSourceOptions struktur
- CatalogKafkaSource struktur
- DirectKafkaSource struktur
- KafkaStreamingSourceOptions struktur
- RedshiftSource struktur
- AmazonRedshiftSource struktur
- AmazonRedshiftNodeData struktur
- AmazonRedshiftAdvancedOption struktur
- Struktur opsi
- Struktur S3 CatalogSource
- Struktur S3 SourceAdditionalOptions
- Struktur S3 CsvSource
- Struktur DirectJDBCSource
- Struktur S3 DirectSourceAdditionalOptions
- Struktur S3 JsonSource
- Struktur S3 ParquetSource
- Struktur S3 DeltaSource
- Struktur S3 CatalogDeltaSource

- CatalogDeltaSource struktur
- Struktur S3 HudiSource
- Struktur S3 CatalogHudiSource
- CatalogHudiSource struktur
- Struktur DynamoDB CatalogSource
- RelationalCatalogSource struktur
- Struktur JDBC ConnectorTarget
- SparkConnectorTarget struktur
- BasicCatalogTarget struktur
- Struktur MySQL CatalogTarget
- Struktur CatalogTarget PostgreSQL
- Struktur CatalogTarget OracleSQL
- Struktur ServerCatalogTarget MicrosoftSQL
- RedshiftTarget struktur
- AmazonRedshiftTarget struktur
- UpsertRedshiftTargetOptions struktur
- Struktur S3 CatalogTarget
- Struktur S3 GlueParquetTarget
- CatalogSchemaChangePolicy struktur
- Struktur S3 DirectTarget
- Struktur S3 HudiCatalogTarget
- Struktur S3 HudiDirectTarget
- Struktur S3 DeltaCatalogTarget
- Struktur S3 DeltaDirectTarget
- DirectSchemaChangePolicy struktur
- ApplyMapping struktur
- Struktur pemetaan
- SelectFields struktur
- DropFields struktur
- RenameField struktur

- Struktur keran
- Bergabunglah dengan struktur
- JoinColumn struktur
- SplitFields struktur
- SelectFromCollection struktur
- FillMissingValues struktur
- Struktur filter
- FilterExpression struktur
- FilterValue struktur
- CustomCode struktur
- Struktur SparkSQL
- SqlAlias struktur
- DropNullFields struktur
- NullCheckBoxList struktur
- NullValueField struktur
- Struktur tipe data
- Gabungkan struktur
- Struktur serikat
- Struktur Piidetection
- Struktur agregat
- DropDuplicates struktur
- GovernedCatalogTarget struktur
- GovernedCatalogSource struktur
- AggregateOperation struktur
- GlueSchema struktur
- GlueStudioSchemaColumn struktur
- GlueStudioColumn struktur
- DynamicTransform struktur
- TransformConfigParameter struktur
- EvaluateDataQuality struktur

- JobRun struktur
- Struktur pendahulu
- JobBookmarkEntry struktur
- BatchStopJobRunSuccessfulSubmission struktur
- BatchStopJobRunError struktur
- NotificationProperty struktur
- Operasi
- StartJobRun tindakan (Python: start_job_run)
- BatchStopJobRun tindakan (Python: batch_stop_job_run)
- GetJobRun tindakan (Python: get_job_run)
- GetJobRuns tindakan (Python: get_job_runs)
- GetJobBookmark tindakan (Python: get_job_bookmark)
- GetJobBookmarks tindakan (Python: get_job_bookmarks)
- ResetJobBookmark tindakan (Python: reset_job_bookmark)

- Pemicu
  - Jenis data
  - Struktur pemicu
  - TriggerUpdate struktur
  - Struktur predikat
  - Struktur kondisi
  - Struktur aksi
  - EventBatchingCondition struktur
  - Operasi
  - CreateTrigger tindakan (Python: create_trigger)
  - StartTrigger tindakan (Python: start_trigger)
  - GetTrigger tindakan (Python: get_trigger)
  - GetTriggers tindakan (Python: get_trigger)
  - UpdateTrigger tindakan (Python: update_trigger)
  - StopTrigger tindakan (Python: stop_trigger)
  - DeleteTrigger tindakan (Python: delete_trigger)

- BatchGetDevEndpoints tindakan (Python: batch_get_dev_endpoints)
- ListDevEndpoints tindakan (Python: list_dev_endpoints)

- Registri skema
  - Jenis data
  - RegistryId struktur
  - RegistryListItem struktur
  - MetadataInfo struktur
  - OtherMetadataValueListItem struktur
  - SchemaListItem struktur
  - SchemaVersionListItem struktur
  - MetadataKeyValuePair struktur
  - SchemaVersionErrorItem struktur
  - ErrorDetails struktur
  - SchemaVersionNumber struktur
  - SchemaId struktur
  - Operasi
  - CreateRegistry tindakan (Python: create_registry)
  - CreateSchema tindakan (Python: create_schema)
  - GetSchema tindakan (Python: get_schema)
  - ListSchemaVersions tindakan (Python: list_schema_versions)
  - GetSchemaVersion tindakan (Python: get_schema_version)
  - GetSchemaVersionsDiff tindakan (Python: get_schema_versions_diff)
  - ListRegistries tindakan (Python: list_registries)
  - ListSchemas tindakan (Python: list_schemas)
  - RegisterSchemaVersion tindakan (Python: register_schema_version)
  - UpdateSchema tindakan (Python: update_schema)
  - CheckSchemaVersionValidity tindakan (Python: check_schema_version_validity)
  - UpdateRegistry tindakan (Python: update_registry)
  - GetSchemaByDefinition tindakan (Python: get_schema_by_definition)
  - GetRegistry tindakan (Python: get_registry)

- PutSchemaVersionMetadata tindakan (Python: put_schema_version_metadata)
- QuerySchemaVersionMetadata tindakan (Python: query_schema_version_metadata)
- RemoveSchemaVersionMetadata tindakan (Python: remove_schema_version_metadata)
- DeleteRegistry tindakan (Python: delete_registry)
- DeleteSchema tindakan (Python: delete_schema)
- DeleteSchemaVersions tindakan (Python: delete_schema_versions)

- Alur Kerja
  - Jenis data
  - JobNodeDetails struktur
  - CrawlerNodeDetails struktur
  - TriggerNodeDetails struktur
  - Struktur merangkak
  - Struktur simpul
  - Struktur tepi
  - Struktur alur kerja
  - WorkflowGraph struktur
  - WorkflowRun struktur
  - WorkflowRunStatistics struktur
  - StartingEventBatchCondition struktur
  - Struktur cetak biru
  - BlueprintDetails struktur
  - LastActiveDefinition struktur
  - BlueprintRun struktur
  - Operasi
  - CreateWorkflow tindakan (Python: create_workflow)
  - UpdateWorkflow tindakan (Python: update_workflow)
  - DeleteWorkflow tindakan (Python: delete_workflow)
  - GetWorkflow tindakan (Python: get_workflow)
  - ListWorkflows tindakan (Python: list_workflows)
  - BatchGetWorkflows tindakan (Python: batch_get_workflows)

- GetWorkflowRun tindakan (Python: get_workflow_run)
- GetWorkflowRuns tindakan (Python: get_workflow_runs)
- GetWorkflowRunProperties tindakan (Python: get_workflow_run_properties)
- PutWorkflowRunProperties tindakan (Python: put_workflow_run_properties)
- CreateBlueprint tindakan (Python: create_blueprint)
- UpdateBlueprint tindakan (Python: update_blueprint)
- DeleteBlueprint tindakan (Python: delete_blueprint)
- ListBlueprints tindakan (Python: list_blueprints)
- BatchGetBlueprints tindakan (Python: batch_get_blueprints)
- StartBlueprintRun tindakan (Python: start_blueprint_run)
- GetBlueprintRun tindakan (Python: get_blueprint_run)
- GetBlueprintRuns tindakan (Python: get_blueprint_runs)
- StartWorkflowRun tindakan (Python: start_workflow_run)
- StopWorkflowRun tindakan (Python: stop_workflow_run)
- ResumeWorkflowRun tindakan (Python: resume_workflow_run)

- Profil penggunaan
  - Jenis data
  - ProfileConfiguration struktur
  - ConfigurationObject struktur
  - UsageProfileDefinition struktur
  - Operasi
  - CreateUsageProfile tindakan (Python: create_usage_profile)
  - GetUsageProfile tindakan (Python: get_usage_profile)
  - UpdateUsageProfile tindakan (Python: update_usage_profile)
  - DeleteUsageProfile tindakan (Python: delete_usage_profile)
  - ListUsageProfiles tindakan (Python: list_usage_profiles)
- API pembelajaran mesin
  - Jenis data
  - TransformParameters struktur
  - EvaluationMetrics struktur

- Struktur MLTransform
- FindMatchesParameters struktur
- FindMatchesMetrics struktur
- ConfusionMatrix struktur
- GlueTable struktur
- TaskRun struktur
- TransformFilterCriteria struktur
- TransformSortCriteria struktur
- TaskRunFilterCriteria struktur
- TaskRunSortCriteria struktur
- TaskRunProperties struktur
- FindMatchesTaskRunProperties struktur
- ImportLabelsTaskRunProperties struktur
- ExportLabelsTaskRunProperties struktur
- LabelingSetGenerationTaskRunProperties struktur
- SchemaColumn struktur
- TransformEncryption struktur
- UserDataEncryption Struktur ML
- ColumnImportance struktur
- Operasi
- Tindakan CreateMelTransform (Python: create_ml_transform)
- Tindakan updateMLTransform (Python: update_ml_transform)
- Tindakan DeletemlTransform (Python: delete_ml_transform)
- Tindakan getmlTransform (Python: get_ml_transform)
- Tindakan getmlTransforms (Python: get_ml_transforms)
- Tindakan ListmlTransforms (Python: list_ml_transforms)
- EvaluationTaskRun Tindakan startML (Python: start_ml_evaluation_task_run)
- LabelingSetGenerationTaskRun Tindakan startML (Python: start_ml_labeling_set_generation_task_run)

- TaskRun Tindakan GetMl (Python: get_ml_task_run)

- TagResource tindakan (Python: tag_resource)
- UntagResource tindakan (Python: untag_resource)
- GetTags tindakan (Python: get_tags)
- Tipe data umum
  - Struktur tag
  - DecimalNumber struktur
  - ErrorDetail struktur
  - PropertyPredicate struktur
  - ResourceUri struktur
  - ColumnStatistics struktur
  - ColumnStatisticsError struktur
  - ColumnError struktur
  - ColumnStatisticsData struktur
  - BooleanColumnStatisticsData struktur
  - DateColumnStatisticsData struktur
  - DecimalColumnStatisticsData struktur
  - DoubleColumnStatisticsData struktur
  - LongColumnStatisticsData struktur
  - StringColumnStatisticsData struktur
  - BinaryColumnStatisticsData struktur
  - Pola string
- Pengecualian
  - AccessDeniedException struktur
  - AlreadyExistsException struktur
  - ConcurrentModificationException struktur
  - ConcurrentRunsExceededException struktur
  - CrawlerNotRunningException struktur
  - CrawlerRunningException struktur
  - CrawlerStoppingException struktur
  - EntityNotFoundException struktur

- FederationSourceException struktur
- FederationSourceRetryableException struktur
- GlueEncryptionException struktur
- IdempotentParameterMismatchException struktur
- IllegalWorkflowStateException struktur
- InternalServiceException struktur
- InvalidExecutionEngineException struktur
- InvalidInputException struktur
- InvalidStateException struktur
- InvalidTaskStatusTransitionException struktur
- JobDefinitionErrorException struktur
- JobRunInTerminalStateException struktur
- JobRunInvalidStateTransitionException struktur
- JobRunNotInTerminalStateException struktur
- LateRunnerException struktur
- NoScheduleException struktur
- OperationTimeoutException struktur
- ResourceNotReadyException struktur
- ResourceNumberLimitExceededException struktur
- SchedulerNotRunningException struktur
- SchedulerRunningException struktur
- SchedulerTransitioningException struktur
- UnrecognizedRunnerException struktur
- ValidationException struktur
- VersionMismatchException struktur

API Keamanan di AWS Glue

API Keamanan menjelaskan tipe data keamanan, dan API yang terkait dengan keamanan di AWS Glue.

Jenis data

DataCatalogEncryptionSettings struktur

Berisi informasi konfigurasi untuk mempertahankan keamanan Katalog Data.

Bidang

- `EncryptionAtRest` — Sebuah objek EncryptionAtIstirahat.

  Menentukan encryption-at-rest konfigurasi untuk Data Catalog.
- `ConnectionPasswordEncryption` — Sebuah objek ConnectionPasswordEnkripsi.

  Ketika perlindungan kata sandi koneksi diaktifkan, Katalog Data menggunakan kunci yang disediakan pelanggan untuk mengenkripsi kata sandi sebagai bagian dari `CreateConnection` atau `UpdateConnection` dan menyimpannya dalam bidang `ENCRYPTED_PASSWORD` di properti koneksi. Anda dapat mengaktifkan enkripsi katalog atau enkripsi kata sandi saja.

EncryptionAtRest struktur

Menentukan encryption-at-rest konfigurasi untuk Data Catalog.

Bidang

- `CatalogEncryptionMode` – Wajib: String UTF-8 (nilai yang valid: `DISABLED` | `SSE-KMS="SSEKMS"` | `SSE-KMS-WITH-SERVICE-ROLE="SSEKMSWITHSERVICEROLE"`).

encryption-at-rest Mode untuk mengenkripsi data Katalog Data.

- `SseAwsKmsKeyId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID AWS KMS kunci yang digunakan untuk enkripsi saat istirahat.

- `CatalogEncryptionServiceRole` — String UTF-8, yang cocok dengan Custom string pattern #24.

  Peran yang AWS Glue mengasumsikan untuk mengenkripsi dan mendekripsi objek Katalog Data atas nama pemanggil.

ConnectionPasswordEncryption struktur

Struktur data yang digunakan oleh Katalog Data untuk mengenkripsi kata sandi sebagai bagian dari `CreateConnection` atau `UpdateConnection` dan menyimpannya di bidang `ENCRYPTED_PASSWORD` dalam properti koneksi. Anda dapat mengaktifkan enkripsi katalog atau enkripsi kata sandi saja.

Ketika `CreationConnection` permintaan tiba berisi kata sandi, Katalog Data pertama mengenkripsi kata sandi menggunakan kunci Anda AWS KMS . Kemudian ia mengenkripsi objek koneksi secara keseluruhan lagi jika enkripsi katalog juga diaktifkan.

Enkripsi ini mengharuskan Anda menetapkan izin AWS KMS kunci untuk mengaktifkan atau membatasi akses pada kunci kata sandi sesuai dengan persyaratan keamanan Anda. Misalnya, Anda mungkin ingin hanya administrator yang memiliki izin dekripsi pada kunci kata sandi.

Bidang

- `ReturnConnectionPasswordEncrypted` – Wajib: Boolean.

  Saat bendera `ReturnConnectionPasswordEncrypted` diatur ke "BETUL", kata sandi tetap dienkripsi dalam respons `GetConnection` dan `GetConnections`. Enkripsi ini berlaku secara independen dari enkripsi katalog.

- `AwsKmsKeyId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  AWS KMS Kunci yang digunakan untuk mengenkripsi kata sandi koneksi.

Jika perlindungan kata sandi koneksi diaktifkan, penelepon `CreateConnection` dan `UpdateConnection` memerlukan setidaknya `kms:Encrypt` izin pada AWS KMS kunci yang ditentukan, untuk mengenkripsi kata sandi sebelum menyimpannya di Katalog Data.

Anda dapat mengatur izin dekripsi untuk mengaktifkan atau membatasi akses pada kunci kata sandi sesuai dengan persyaratan keamanan Anda.

EncryptionConfiguration struktur

Menentukan konfigurasi enkripsi.

Bidang

- `S3Encryption` – Susunan objek S3Encryption.

  Konfigurasi enyption untuk data Amazon Simple Storage Service (Amazon S3).
- `CloudWatchEncryption` — Sebuah objek CloudWatchEnkripsi.

  Konfigurasi enkripsi untuk Amazon CloudWatch.
- `JobBookmarksEncryption` — Sebuah objek JobBookmarksEnkripsi.

  Konfigurasi enkripsi untuk bookmark tugas.

Struktur enkripsi S3

Menentukan bagaimana data Amazon Simple Storage Service (Amazon S3) harus dienkripsi.

Bidang

- `S3EncryptionMode` – String UTF-8 (nilai yang valid: `DISABLED` | `SSE-KMS="SSEKMS"` | `SSE-S3="SSES3"`).

  Mode enkripsi yang digunakan untuk data Amazon S3.
- `KmsKeyArn` — String UTF-8, yang cocok dengan Custom string pattern #25.

  Amazon Resource Name (ARN) dari kunci KMS yang akan digunakan untuk mengenkripsi data.

CloudWatchEncryption struktur

Menentukan bagaimana CloudWatch data Amazon harus dienkripsi.

Bidang

- `CloudWatchEncryptionMode` – String UTF-8 (nilai yang valid: `DISABLED` | `SSE-KMS="SSEKMS"`).

  Mode enkripsi yang digunakan untuk CloudWatch data.
- `KmsKeyArn` — String UTF-8, yang cocok dengan Custom string pattern #25.

  Amazon Resource Name (ARN) dari kunci KMS yang akan digunakan untuk mengenkripsi data.

JobBookmarksEncryption struktur

Menentukan bagaimana data bookmark tugas harus dienkripsi.

Bidang

- `JobBookmarksEncryptionMode` – String UTF-8 (nilai yang valid: `DISABLED` | `CSE-KMS="CSEKMS"`).

  Mode enkripsi yang digunakan untuk data bookmark tugas.
- `KmsKeyArn` — String UTF-8, yang cocok dengan Custom string pattern #25.

  Amazon Resource Name (ARN) dari kunci KMS yang akan digunakan untuk mengenkripsi data.

SecurityConfiguration struktur

Menentukan sebuah konfigurasi keamanan.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama konfigurasi keamanan.
- `CreatedTimeStamp` — Stempel waktu.

Waktu di mana konfigurasi keamanan ini dibuat.

- `EncryptionConfiguration` — Sebuah objek EncryptionConfiguration.

  Konfigurasi enkripsi yang dikaitkan dengan konfigurasi keamanan ini.

GluePolicy struktur

Sebuah struktur untuk mengembalikan kebijakan sumber daya.

Bidang

- `PolicyInJson`— UTF-8 string, setidaknya 2 byte panjang.

  Berisi dokumen kebijakan yang diminta, dalam format JSON.
- `PolicyHash` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Berisi nilai hash yang dikaitkan dengan kebijakan ini.
- `CreateTime` — Stempel waktu.

  Tanggal dan waktu kebijakan dibuat.
- `UpdateTime` — Stempel waktu.

  Tanggal dan waktu kebijakan terakhir diperbarui.

Operasi

- GetDataCatalogEncryptionSettings tindakan (Python: get_data_catalog_encryption_settings)
- PutDataCatalogEncryptionSettings tindakan (Python: put_data_catalog_encryption_settings)
- PutResourcePolicy tindakan (Python: put_resource_policy)
- GetResourcePolicy tindakan (Python: get_resource_policy)
- DeleteResourcePolicy tindakan (Python: delete_resource_policy)
- CreateSecurityConfiguration tindakan (Python: create_security_configuration)
- DeleteSecurityConfiguration tindakan (Python: delete_security_configuration)
- GetSecurityConfiguration tindakan (Python: get_security_configuration)

- GetSecurityConfigurations tindakan (Python: get_security_configurations)
- GetResourcePolicies tindakan (Python: get_resource_policies)

GetDataCatalogEncryptionSettings tindakan (Python: get_data_catalog_encryption_settings)

Mengambil konfigurasi keamanan untuk sebuah katalog yang ditentukan.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data yang akan diambil untuk konfigurasi keamanan. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.

Respons

- `DataCatalogEncryptionSettings` — Sebuah objek DataCatalogEncryptionSettings.

  Konfigurasi keamanan yang diminta.

Kesalahan

- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

PutDataCatalogEncryptionSettings tindakan (Python: put_data_catalog_encryption_settings)

Menetapkan konfigurasi keamanan untuk sebuah katalog yang ditentukan. Setelah konfigurasi diatur, enkripsi tertentu diterapkan untuk setiap penulisan katalog setelahnya.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data yang akan ditetapkan untuk konfigurasi keamanan. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DataCatalogEncryptionSettings` — Wajib: Sebuah objek DataCatalogEncryptionSettings.

  Konfigurasi keamanan yang akan ditetapkan.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

PutResourcePolicy tindakan (Python: put_resource_policy)

Menetapkan kebijakan sumber daya Katalog Data untuk kendali akses.

Permintaan

- `PolicyInJson`- Wajib: UTF-8 string, setidaknya 2 byte panjang.

  Berisi dokumen kebijakan yang akan ditetapkan, dalam format JSON.
- `ResourceArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Jangan gunakan . Untuk penggunaan internal saja.
- `PolicyHashCondition` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Nilai hash dikembalikan bila kebijakan sebelumnya diatur menggunakan `PutResourcePolicy`. Tujuannya adalah untuk mencegah modifikasi secara bersamaan pada kebijakan. Jangan gunakan parameter ini jika tidak ada kebijakan sebelumnya yang telah ditetapkan.

- `PolicyExistsCondition` – String UTF-8 (nilai yang valid: `MUST_EXIST` | `NOT_EXIST` | `NONE`).

  Nilai `MUST_EXIST` digunakan untuk memperbarui kebijakan. Nilai `NOT_EXIST` digunakan untuk membuat kebijakan baru. Jika nilai `NONE` atau nilai nol digunakan, maka panggilan tidak tergantung pada keberadaan sebuah kebijakan.

- `EnableHybrid` – String UTF-8 (nilai yang valid: `TRUE` | `FALSE`).

  Jika `'TRUE'`, menunjukkan bahwa Anda menggunakan kedua metode tersebut untuk memberikan akses lintas akun ke sumber daya Katalog Data:

  - Dengan langsung memperbarui kebijakan sumber daya dengan `PutResourePolicy`
  - Dengan menggunakan perintah Berikan izin di AWS Management Console.

  Harus diatur ke `'TRUE'` jika Anda telah menggunakan Konsol Manajemen untuk memberikan akses lintas akun, jika tidak, maka panggilan gagal. Default-nya adalah ‘SALAH’.

Respons

- `PolicyHash` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Sebuah hash dari kebijakan yang baru saja ditetapkan. Ini harus disertakan dalam panggilan berikutnya yang akan menimpa atau memperbarui kebijakan ini.

Kesalahan

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ConditionCheckFailureException`

GetResourcePolicy tindakan (Python: get_resource_policy)

Mengambil sebuah kebijakan sumber daya yang ditentukan.

Permintaan

- `ResourceArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  ARN AWS Glue sumber daya untuk mengambil kebijakan sumber daya. Jika tidak disediakan, maka kebijakan sumber daya Katalog Data akan dikembalikan. Gunakan `GetResourcePolicies` untuk melihat semua kebijakan sumber daya yang ada. Untuk informasi lebih lanjut lihat Menentukan ARN Sumber Daya AWS Glue.

Respons

- `PolicyInJson`— UTF-8 string, setidaknya 2 byte panjang.

  Berisi dokumen kebijakan yang diminta, dalam format JSON.
- `PolicyHash` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Berisi nilai hash yang dikaitkan dengan kebijakan ini.
- `CreateTime` — Stempel waktu.

  Tanggal dan waktu kebijakan dibuat.
- `UpdateTime` — Stempel waktu.

  Tanggal dan waktu kebijakan terakhir diperbarui.

Kesalahan

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

DeleteResourcePolicy tindakan (Python: delete_resource_policy)

Menghapus sebuah kebijakan yang ditentukan.

Permintaan

- `PolicyHashCondition` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nilai hash akan dikembalikan bila kebijakan ini ditetapkan.
- `ResourceArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  ARN AWS Glue sumber daya untuk kebijakan sumber daya yang akan dihapus.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ConditionCheckFailureException`

CreateSecurityConfiguration tindakan (Python: create_security_configuration)

Membuat sebuah konfigurasi keamanan baru. Konfigurasi keamanan adalah seperangkat properti keamanan yang dapat digunakan oleh AWS Glue. Anda dapat menggunakan sebuah konfigurasi keamanan untuk mengenkripsi data at rest. Untuk informasi tentang penggunaan konfigurasi keamanan di AWS Glue, lihat Mengenkripsi Data yang Ditulis oleh Crawler, Pekerjaan, dan Titik Akhir Pengembangan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama untuk konfigurasi keamanan yang baru.
- `EncryptionConfiguration` — Wajib: Sebuah objek EncryptionConfiguration.

  Konfigurasi enkripsi untuk konfigurasi keamanan yang baru.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama yang ditetapkan untuk konfigurasi keamanan yang baru.
- `CreatedTimestamp` — Stempel waktu.

  Waktu di mana konfigurasi keamanan baru dibuat.

Kesalahan

- `AlreadyExistsException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`

DeleteSecurityConfiguration tindakan (Python: delete_security_configuration)

Menghapus sebuah konfigurasi keamanan yang ditentukan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Nama konfigurasi keamanan yang akan dihapus.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetSecurityConfiguration tindakan (Python: get_security_configuration)

Mengambil sebuah konfigurasi keamanan yang ditentukan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama konfigurasi keamanan yang akan diambil.

Respons

- `SecurityConfiguration` — Sebuah objek SecurityConfiguration.

  Konfigurasi keamanan yang diminta.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetSecurityConfigurations tindakan (Python: get_security_configurations)

Mengambil daftar dari semua konfigurasi keamanan.

Permintaan

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah hasil maksimum yang akan dikembalikan.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Respons

- `SecurityConfigurations` – Susunan objek SecurityConfiguration.

  Sebuah daftar konfigurasi keamanan.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ada konfigurasi keamanan lainnya yang akan dikembalikan.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetResourcePolicies tindakan (Python: get_resource_policies)

Mengambil kebijakan sumber daya yang ditetapkan pada sumber daya individu AWS Resource Access Manager selama pemberian izin lintas akun. Juga mengambil kebijakan sumber daya Katalog Data.

Jika Anda mengaktifkan enkripsi metadata dalam pengaturan Katalog Data, dan Anda tidak memiliki izin pada AWS KMS kunci, operasi tidak dapat menampilkan kebijakan sumber daya Katalog Data.

Permintaan

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah permintaan kelanjutan.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Ukuran maksimum daftar yang akan dikembalikan.

Respons

- `GetResourcePoliciesResponseList` – Susunan objek GluePolicy.

  Sebuah daftar kebijakan sumber daya individu dan kebijakan sumber daya tingkat akun.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika daftar yang dikembalikan tidak berisi kebijakan sumber daya terakhir yang tersedia.

Kesalahan

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

API Katalog

API Katalog menjelaskan tipe data dan API yang terkait dengan bekerja dengan katalog di. AWS Glue

Topik

- API basis data
- Tabel API
- API partisi
- API Koneksi

- API Fungsi yang ditentukan pengguna
- Mengimpor Athena katalog ke AWS Glue

API basis data

API basis data menjelaskan tipe data basis data, dan termasuk API untuk membuat, menghapus, menemukan, memperbarui, dan mencantumkan basis data.

Jenis Data

- Struktur basis data
- DatabaseInput struktur
- PrincipalPermissions struktur
- DataLakePrincipal struktur
- DatabaseIdentifier struktur
- FederatedDatabase struktur

Struktur basis data

Objek `Database` merupakan pengelompokan logis tabel yang mungkin berada di metastore Hive atau RDBMS.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data. Untuk kompatibilitas Hive, nama ini diubah ke huruf kecil ketika disimpan.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi basis data.
- `LocationUri` — Pengenal sumber daya seragam (uri), dengan panjang tidak kurang dari 1 atau lebih dari 1024 byte, yang cocok dengan URI address multi-line string pattern.

  Lokasi basis data (misalnya, jalur HDFS).

- `Parameters` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah String kunci, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 512000 byte.

  Pasangan kunci-nilai ini menentukan parameter dan properti basis data.
- `CreateTime` — Stempel waktu.

  Waktu ketika metadata basis data dibuat dalam katalog.
- `CreateTableDefaultPermissions` – Susunan objek PrincipalPermissions.

  Menciptakan satu set izin default pada tabel untuk prinsipal utama. Digunakan olehAWS Lake Formation. Tidak digunakan dalam AWS Glue operasi normal.
- `TargetDatabase` — Sebuah objek DatabaseIdentifier.

  Struktur `DatabaseIdentifier` yang menggambarkan basis data target untuk penautan sumber daya.
- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat basis data berada.
- `FederatedDatabase` — Sebuah objek FederatedDatabase.

  `FederatedDatabase`Struktur yang mereferensikan entitas di luarAWS Glue Data Catalog.

DatabaseInput struktur

Struktur yang digunakan untuk membuat atau memperbarui basis data.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data. Untuk kompatibilitas Hive, nama ini diubah ke huruf kecil ketika disimpan.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

Deskripsi basis data.

- `LocationUri` — Pengenal sumber daya seragam (uri), dengan panjang tidak kurang dari 1 atau lebih dari 1024 byte, yang cocok dengan URI address multi-line string pattern.

  Lokasi basis data (misalnya, jalur HDFS).
- `Parameters` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah String kunci, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 512000 byte.

  Pasangan kunci-nilai ini menentukan parameter dan properti basis data.

  Pasangan kunci-nilai ini menentukan parameter dan properti basis data.
- `CreateTableDefaultPermissions` – Susunan objek PrincipalPermissions.

  Menciptakan satu set izin default pada tabel untuk prinsipal utama. Digunakan olehAWS Lake Formation. Tidak digunakan dalam AWS Glue operasi normal.
- `TargetDatabase` — Sebuah objek DatabaseIdentifier.

  Struktur `DatabaseIdentifier` yang menggambarkan basis data target untuk penautan sumber daya.
- `FederatedDatabase` — Sebuah objek FederatedDatabase.

  `FederatedDatabase`Struktur yang mereferensikan entitas di luarAWS Glue Data Catalog.

PrincipalPermissions struktur

Izin yang diberikan ke sebuah prinsipal utama.

Bidang

- `Principal` — Sebuah objek DataLakePrincipal.

  Prinsipal utama yang diberikan izin.
- `Permissions` – Susunan string UTF-8.

  Izin yang diberikan kepada prinsipal utama.

DataLakePrincipal struktur

AWS Lake Formation prinsipal.

Bidang

- `DataLakePrincipalIdentifier` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte.

  Pengidentifikasi untuk prinsipal utama AWS Lake Formation.

DatabaseIdentifier struktur

Struktur yang menggambarkan database target untuk menghubungkan sumber daya.

Bidang

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat basis data berada.
- `DatabaseName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog.
- `Region` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Wilayah basis data target.

FederatedDatabase struktur

Database yang menunjuk ke entitas di luarAWS Glue Data Catalog.

Bidang

- `Identifier` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 512 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik untuk database federasi.

- `ConnectionName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama koneksi ke metastore eksternal.

Operasi

- CreateDatabase tindakan (Python: create_database)
- UpdateDatabase tindakan (Python: update_database)
- DeleteDatabase tindakan (Python: delete_database)
- GetDatabase tindakan (Python: get_database)
- GetDatabases tindakan (Python: get_databases)

CreateDatabase tindakan (Python: create_database)

Menciptakan sebuah basis data baru dalam Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat basis data akan dibuat. Jika tidak ada yang disediakan, ID akun AWS digunakan secara default.
- `DatabaseInput` — Wajib: Sebuah objek DatabaseInput.

  Metadata untuk basis data.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Tag yang Anda tetapkan ke database.

Response

- Tidak ada parameter Respons.

Kesalahan

- `InvalidInputException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`
- `FederatedResourceAlreadyExistsException`

UpdateDatabase tindakan (Python: update_database)

Memperbarui definisi basis data yang ada dalam Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat metadata basis data berada. Jika tidak ada yang disediakan, ID akun AWS digunakan secara default.

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data yang akan diperbarui dalam katalog. Untuk kompatibilitas Hive, nama ini diubah ke huruf kecil.

- `DatabaseInput` — Wajib: Sebuah objek DatabaseInput.

  Sebuah objek `DatabaseInput` menentukan definisi baru dari metadata basis data dalam katalog.

Response

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`

DeleteDatabase tindakan (Python: delete_database)

Menghapus sebuah basis data yang ditentukan dari Katalog Data.

> **Note**
>
> Setelah menyelesaikan operasi ini, Anda tidak lagi memiliki akses ke tabel (dan semua versi tabel dan partisi yang mungkin merupakan milik tabel) dan fungsi yang ditetapkan pengguna dalam basis data yang dihapus. AWS Glue menghapus sumber daya "yatim piatu" secara asinkron pada waktu yang tepat, atas kebijaksanaan layanan.
> Untuk memastikan penghapusan langsung dari semua sumber daya terkait, sebelum memanggil `DeleteDatabase`, gunakan `DeleteTableVersion` atau `BatchDeleteTableVersion`, `DeletePartition` atau `BatchDeletePartition`, `DeleteUserDefinedFunction`, dan `DeleteTable` atau `BatchDeleteTable`, untuk menghapus sumber daya yang dimiliki oleh basis data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat basis data berada. Jika tidak ada yang disediakan, ID akun AWS digunakan secara default.

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data yang akan dihapus. Untuk kompatibilitas Hive, ini semua harus huruf kecil.

Response

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

GetDatabase tindakan (Python: get_database)

Mengambil definisi dari basis data yang ditentukan.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat basis data berada. Jika tidak ada yang disediakan, ID akun AWS digunakan secara default.

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data yang akan diambil. Untuk kompatibilitas Hive, nama ini harus huruf kecil.

Response

- `Database` — Sebuah objek Basis Data.

Definisi basis data yang ditentukan dalam Katalog Data.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `FederationSourceException`

GetDatabases tindakan (Python: get_databases)

Mengambil semua basis data yang didefinisikan dalam Katalog Data yang ditentukan.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat `Databases` akan diambil. Jika tidak ada yang disediakan, ID akun AWS digunakan secara default.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100.

  Jumlah maksimum basis data yang akan dikembalikan dalam satu respons.
- `ResourceShareType` – String UTF-8 (nilai yang valid: `FOREIGN` | `ALL` | `FEDERATED`).

  Memungkinkan Anda untuk menentukan apakah Anda ingin mencantumkan basis data yang dibagikan dengan akun Anda. Nilai yang diijinkan adalah`FEDERATED`, `FOREIGN` atau`ALL`.
  - Jika disetel ke`FEDERATED`, akan mencantumkan database federasi (merujuk entitas eksternal) yang dibagikan dengan akun Anda.
  - Jika diatur ke `FOREIGN`, akan mencantumkan basis data yang dibagikan dengan akun Anda.

- Jika diatur ke `ALL`, akan mencantumkan basis data yang dibagikan dengan akun Anda, serta basis data di akun lokal Anda.

Response

- `DatabaseList` – Wajib: Susunan objek Basis Data.

  Daftar objek `Database` dari katalog yang ditentukan.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan untuk pemberian nomor halaman untuk daftar token yang ditampilkan, dikembalikan jika segmen saat ini dari daftar tersebut bukan yang terakhir.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

Tabel API

API Tabel menjelaskan jenis data dan operasi yang dikaitkan dengan tabel.

Jenis data

- Struktur meja
- TableInput struktur
- FederatedTable struktur
- Struktur kolom
- StorageDescriptor struktur
- SchemaReference struktur
- SerDeInfo struktur
- Struktur pesanan
- SkewedInfo struktur

- TableVersion struktur
- TableError struktur
- TableVersionError struktur
- SortCriterion struktur
- TableIdentifier struktur
- KeySchemaElement struktur
- PartitionIndex struktur
- PartitionIndexDescriptor struktur
- BackfillError struktur
- IcebergInput struktur
- OpenTableFormatInput struktur
- ViewDefinition struktur
- ViewDefinitionInput struktur
- ViewRepresentation struktur
- ViewRepresentationInput struktur

Struktur meja

Mewakili kumpulan data terkait yang diatur dalam kolom dan baris.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `DatabaseName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data tempat metadata tabel berada. Untuk kompatibilitas Hive, ini semua harus huruf kecil.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi tabel.

- `Owner` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pemilik tabel.

- `CreateTime` — Stempel waktu.

  Waktu ketika definisi tabel dibuat dalam Katalog Data.

- `UpdateTime` — Stempel waktu.

  Terakhir kali tabel itu diperbarui.

- `LastAccessTime` — Stempel waktu.

  Terakhir kali tabel itu diakses. Ini biasanya diambil dari HDFS, dan mungkin tidak dapat diandalkan.

- `LastAnalyzedTime` — Stempel waktu.

  Terakhir kali statistik kolom dikomputasi untuk tabel ini.

- `Retention` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Waktu retensi untuk tabel ini.

- `StorageDescriptor` — Sebuah objek StorageDescriptor.

  Sebuah deskriptor penyimpanan yang berisi informasi tentang penyimpanan fisik tabel ini.

- `PartitionKeys` – Susunan objek Kolom.

  Daftar kolom yang digunakan sebagai dasar pemartisian. Hanya tipe primitif saja yang didukung sebagai kunci partisi.

  Bila Anda membuat sebuah tabel yang digunakan oleh Amazon Athena, dan Anda tidak menentukan `partitionKeys`, Anda harus setidaknya menetapkan nilai `partitionKeys` ke daftar kosong. Sebagai contoh:

  ```
  "PartitionKeys": []
  ```

- `ViewOriginalText` — String UTF-8, dengan panjang tidak lebih dari 409600 byte.

  Termasuk untuk kompatibilitas Apache Hive. Tidak digunakan dalam AWS Glue operasi normal. Jika tabelnya adalah`VIRTUAL_VIEW`, Athena konfigurasi tertentu dikodekan dalam base64.

- `ViewExpandedText` — String UTF-8, dengan panjang tidak lebih dari 409600 byte.

  Termasuk untuk kompatibilitas Apache Hive. Tidak digunakan dalam AWS Glue operasi normal.

- `TableType` — String UTF-8, dengan panjang tidak lebih dari 255 byte.

  Jenis tabel ini. AWS Glue akan membuat tabel dengan `EXTERNAL_TABLE` tipe. Layanan lain, seperti Athena, dapat membuat tabel dengan jenis tabel tambahan.

  AWS Glue jenis tabel terkait:

  EXTERNAL_TABLE

  Atribut kompatibel sarang - menunjukkan tabel terkelola non-HIVE.

  DIATUR

  Digunakan oleh AWS Lake Formation. Katalog AWS Glue Data mengerti`GOVERNED`.
- `Parameters` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah String kunci, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 512000 byte.

  Pasangan nilai kunci ini menentukan sifat yang dikaitkan dengan tabel.
- `CreatedBy` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Orang atau entitas yang telah membuat tabel.
- `IsRegisteredWithLakeFormation` – Boolean.

  Menunjukkan apakah tabel telah terdaftar AWS Lake Formation.
- `TargetTable` — Sebuah objek TableIdentifier.

  Sebuah struktur `TableIdentifier` yang menggambarkan tabel target untuk menghubungkan sumber daya.
- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat tabel berada.
- `VersionId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari versi tabel.

- `FederatedTable` — Sebuah objek FederatedTable.

  `FederatedTable`Struktur yang mereferensikan entitas di luar AWS Glue Data Catalog.

- `ViewDefinition` — Sebuah objek ViewDefinition.

  Struktur yang berisi semua informasi yang mendefinisikan tampilan, termasuk dialek atau dialek untuk tampilan, dan kueri.

- `IsMultiDialectView` – Boolean.

  Menentukan apakah tampilan mendukung dialek SQL dari satu atau lebih mesin query yang berbeda dan karena itu dapat dibaca oleh mesin tersebut.

TableInput struktur

Struktur yang digunakan untuk mendefinisikan tabel.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel. Untuk kompatibilitas Hive, nama ini diubah ke huruf kecil ketika disimpan.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi tabel.

- `Owner` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pemilik tabel. Termasuk untuk kompatibilitas Apache Hive. Tidak digunakan dalam AWS Glue operasi normal.

- `LastAccessTime` — Stempel waktu.

  Terakhir kali tabel itu diakses.

- `LastAnalyzedTime` — Stempel waktu.

  Terakhir kali statistik kolom dikomputasi untuk tabel ini.

- `Retention` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

Waktu retensi untuk tabel ini.

- `StorageDescriptor` — Sebuah objek StorageDescriptor.

  Sebuah deskriptor penyimpanan yang berisi informasi tentang penyimpanan fisik tabel ini.

- `PartitionKeys` – Susunan objek Kolom.

  Daftar kolom yang digunakan sebagai dasar pemartisian. Hanya tipe primitif saja yang didukung sebagai kunci partisi.

  Bila Anda membuat sebuah tabel yang digunakan oleh Amazon Athena, dan Anda tidak menentukan `partitionKeys`, Anda harus setidaknya menetapkan nilai `partitionKeys` ke daftar kosong. Sebagai contoh:

  `"PartitionKeys": []`

- `ViewOriginalText` — String UTF-8, dengan panjang tidak lebih dari 409600 byte.

  Termasuk untuk kompatibilitas Apache Hive. Tidak digunakan dalam AWS Glue operasi normal. Jika tabelnya adalah`VIRTUAL_VIEW`, Athena konfigurasi tertentu dikodekan dalam base64.

- `ViewExpandedText` — String UTF-8, dengan panjang tidak lebih dari 409600 byte.

  Termasuk untuk kompatibilitas Apache Hive. Tidak digunakan dalam AWS Glue operasi normal.

- `TableType` — String UTF-8, dengan panjang tidak lebih dari 255 byte.

  Jenis tabel ini. AWS Glue akan membuat tabel dengan `EXTERNAL_TABLE` tipe. Layanan lain, seperti Athena, dapat membuat tabel dengan jenis tabel tambahan.

  AWS Glue jenis tabel terkait:

  EXTERNAL_TABLE

  Atribut kompatibel sarang - menunjukkan tabel terkelola non-HIVE.

  DIATUR

  Digunakan oleh AWS Lake Formation. Katalog AWS Glue Data mengerti`GOVERNED`.

- `Parameters` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah String kunci, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 512000 byte.

Pasangan nilai kunci ini menentukan sifat yang dikaitkan dengan tabel.

- `TargetTable` — Sebuah objek TableIdentifier.

  Sebuah struktur `TableIdentifier` yang menggambarkan tabel target untuk menghubungkan sumber daya.

- `ViewDefinition` — Sebuah objek ViewDefinitionInput.

  Struktur yang berisi semua informasi yang mendefinisikan tampilan, termasuk dialek atau dialek untuk tampilan, dan kueri.

FederatedTable struktur

Tabel yang menunjuk ke entitas di luar AWS Glue Data Catalog.

Bidang

- `Identifier` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 512 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik untuk tabel federasi.

- `DatabaseIdentifier` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 512 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik untuk database federasi.

- `ConnectionName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama koneksi ke metastore eksternal.

Struktur kolom

Sebuah kolom di `Table`.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama `Column`.

- `Type` — String UTF-8, sepanjang tidak lebih dari 131072, yang cocok dengan Single-line string pattern.

  Jenis data dari `Column`.

- `Comment` — String komentar, sepanjang tidak lebih dari 255, yang cocok dengan Single-line string pattern.

  Sebuah komentar teks bentuk bebas.

- `Parameters` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah String kunci, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 512000 byte.

  Pasangan nilai kunci ini menentukan sifat yang dikaitkan dengan kolom.

StorageDescriptor struktur

Menjelaskan penyimpanan fisik data tabel.

Bidang

- `Columns` – Susunan objek Kolom.

  Daftar `Columns` dalam tabel.

- `Location` — String lokasi, sepanjang tidak lebih dari 2056, yang cocok dengan URI address multi-line string pattern.

  Lokasi fisik tabel. Secara default, ini mengambil bentuk lokasi gudang, yang diikuti oleh lokasi basis data di gudang, yang diikuti dengan nama tabel.

- `AdditionalLocations` – Susunan string UTF-8.

Daftar lokasi yang mengarah ke jalur tempat tabel Delta berada.

- `InputFormat` — String format, sepanjang tidak lebih dari 128, yang cocok dengan Single-line string pattern.

  Format input: `SequenceFileInputFormat` (biner), atau `TextInputFormat`, atau format kustom.

- `OutputFormat` — String format, sepanjang tidak lebih dari 128, yang cocok dengan Single-line string pattern.

  Format output: `SequenceFileOutputFormat` (biner), atau `IgnoreKeyTextOutputFormat`, atau format kustom.

- `Compressed` – Boolean.

  `True` jika data dalam tabel dikompresi, atau `False` jika tidak.

- `NumberOfBuckets` — Nomor (bilangan bulat).

  Harus ditentukan jika tabel berisi kolom dimensi.

- `SerdeInfo` — Sebuah objek SerDeInfo.

  Informasi serialisasi/deserialisasi (). SerDe

- `BucketColumns` – Susunan string UTF-8.

  Daftar kolom pengelompokan peredam, kolom pengklasteran, dan kolom pem-bucket-an dalam tabel.

- `SortColumns` – Susunan objek Order.

  Daftar yang menentukan urutan dari setiap bucket dalam tabel.

- `Parameters` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah String kunci, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 512000 byte.

  Properti yang disediakan pengguna dalam bentuk nilai kunci.

- `SkewedInfo` — Sebuah objek SkewedInfo.

  Informasi tentang nilai yang sering muncul di kolom (nilai kecenderungan).

- `StoredAsSubDirectories` – Boolean.

  `True` jika data tabel disimpan di subdirektori, atau `False` jika tidak.

- `SchemaReference` — Sebuah objek SchemaReference.

  Objek yang mereferensikan skema yang disimpan dalam AWS Glue Schema Registry.

  Saat membuat sebuah tabel, Anda dapat memberikan daftar kosong kolom untuk skema, dan sebaliknya menggunakan referensi skema.

SchemaReference struktur

Objek yang mereferensikan skema yang disimpan dalam AWS Glue Schema Registry.

Bidang

- `SchemaId` — Sebuah objek SchemaId.

  Struktur yang berisi bidang identitas skema. Baik ini atau `SchemaVersionId` harus disediakan.

- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  ID unik yang ditetapkan untuk sebuah versi skema. Baik ini atau `SchemaId` harus disediakan.

- `SchemaVersionNumber` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Nomor versi dari skema.

SerDeInfo struktur

Informasi tentang program serialisasi/deserialisasi (SerDe) yang berfungsi sebagai extractor dan loader.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama dari SerDe.

- `SerializationLibrary` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Biasanya kelas yang mengimplementasikan. SerDe Contohnya adalah `org.apache.hadoop.hive.serde2.columnar.ColumnarSerDe`.
- `Parameters` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah String kunci, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 512000 byte.

  Pasangan kunci-nilai ini menentukan parameter inisialisasi untuk. SerDe

Struktur pesanan

Menentukan urutan dari kolom yang diurutkan.

Bidang

- `Column` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama kolom.
- `SortOrder` — Wajib: Nomor (bilangan bulat), tidak lebih dari 1.

  Menunjukkan bahwa kolom diurutkan dalam urutan menaik (`==  1`), atau dalam urutan menurun (`==0`).

SkewedInfo struktur

Menentukan nilai menyimpang dalam sebuah tabel. Nilai menyimpang adalah nilai yang terjadi dengan frekuensi sangat tinggi.

Bidang

- `SkewedColumnNames` – Susunan string UTF-8.

  Daftar nama kolom yang berisi nilai-nilai menyimpang.
- `SkewedColumnValues` – Susunan string UTF-8.

Daftar nilai yang muncul sangat sering untuk dianggap menyimpang.

- `SkewedColumnValueLocationMaps` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

  Pemetaan nilai menyimpang untuk kolom yang berisi nilai tersebut.

TableVersion struktur

Menentukan versi dari sebuah tabel.

Bidang

- `Table` — Sebuah objek Tabel.

  Tabel yang dimaksud.

- `VersionId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nilai ID yang mengidentifikasi versi tabel ini. Sebuah `VersionId` adalah representasi string dari bilangan bulat. Setiap versi bertambah 1.

TableError struktur

Catatan kesalahan untuk operasi tabel.

Bidang

- `TableName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.

- `ErrorDetail` — Sebuah objek ErrorDetail.

  Detail tentang kesalahan.

TableVersionError struktur

Catatan kesalahan untuk operasi versi tabel.

Bidang

- `TableName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel yang dimaksud.
- `VersionId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nilai ID dari versi yang dimaksud. Sebuah `VersionID` adalah representasi string dari bilangan bulat. Setiap versi bertambah 1.
- `ErrorDetail` — Sebuah objek ErrorDetail.

  Detail tentang kesalahan.

SortCriterion struktur

Menentukan sebuah bidang yang akan dijadikan dasar urutan dan urutan pengurutan.

Bidang

- `FieldName` — String nilai, dengan panjang tidak lebih dari 1024 byte.

  Nama bidang yang akan dijadikan dasar pengurutan.
- `Sort` – String UTF-8 (nilai yang valid: `ASC="ASCENDING"` | `DESC="DESCENDING"`).

  Pengurutan naik atau turun.

TableIdentifier struktur

Sebuah struktur yang menggambarkan tabel target untuk menghubungkan sumber daya.

Bidang

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

ID dari Katalog Data tempat tabel berada.

- `DatabaseName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog yang berisi tabel target.

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel target.

- `Region` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Wilayah tabel target.

KeySchemaElement struktur

Pasangan kunci partisi yang terdiri dari nama dan tipe.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama kunci partisi.

- `Type` – Wajib: String UTF-8, sepanjang tidak lebih dari 131072 byte, yang cocok dengan Single-line string pattern.

  Jenis kunci partisi.

PartitionIndex struktur

Struktur untuk indeks partisi.

Bidang

- `Keys` — Wajib: Susunan string UTF-8, setidaknya 1 string.

  Kunci untuk indeks partisi.

- `IndexName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama indeks partisi.

PartitionIndexDescriptor struktur

Sebuah deskriptor untuk indeks partisi dalam sebuah tabel.

Bidang

- `IndexName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama indeks partisi.
- `Keys` — Wajib: Array dari objek KeySchemaElement, setidaknya 1 struktur.

  Daftar satu atau beberapa kunci, sebagai struktur `KeySchemaElement`, untuk indeks partisi.
- `IndexStatus` – Wajib: String UTF-8 (nilai yang valid: `CREATING` | `ACTIVE` | `DELETING` | `FAILED`).

  Status indeks partisi.

  Status yang mungkin muncul adalah:
  - CREATING: Indeks sedang dibuat. Ketika indeks dalam status CREATING, indeks atau tabelnya tidak dapat dihapus.
  - ACTIVE: Pembuatan indeks berhasil.
  - FAILED: Pembuatan indeks gagal.
  - DELETING: Indeks dihapus dari daftar indeks.
- `BackfillErrors` – Susunan objek BackfillError.

  Daftar kesalahan yang dapat terjadi saat mendaftarkan indeks partisi untuk tabel yang ada.

BackfillError struktur

Daftar kesalahan yang dapat terjadi saat mendaftarkan indeks partisi untuk tabel yang ada.

Kesalahan ini memberikan detail tentang mengapa pendaftaran indeks gagal dan menyediakan sejumlah partisi dalam respon dalam jumlah terbatas, sehingga Anda dapat memperbaiki partisi yang

salah dan mencoba mendaftarkan indeks lagi. Kumpulan kesalahan yang paling umum yang dapat terjadi dikategorikan sebagai berikut:

- EncryptedPartitionError: Partisi dienkripsi.
- InvalidPartitionTypeDataError: Nilai partisi tidak cocok dengan tipe data untuk kolom partisi itu.
- MissingPartitionValueError: Partisi dienkripsi.
- UnsupportedPartitionCharacterError: Karakter di dalam nilai partisi tidak didukung. Sebagai contoh: U+0000, U+0001, U+0002.
- InternalError: Kesalahan apa pun yang bukan milik kode kesalahan lainnya.

Bidang

- `Code` – String UTF-8 (nilai yang valid: `ENCRYPTED_PARTITION_ERROR` | `INTERNAL_ERROR` | `INVALID_PARTITION_TYPE_DATA_ERROR` | `MISSING_PARTITION_VALUE_ERROR` | `UNSUPPORTED_PARTITION_CHARACTER_ERROR`).

  Kode kesalahan untuk kesalahan yang terjadi saat mendaftarkan indeks partisi untuk tabel yang ada.

- `Partitions` – Susunan objek PartitionValueDaftar.

  Daftar sejumlah partisi dalam respon dalam jumlah terbatas.

IcebergInput struktur

Struktur yang mendefinisikan tabel metadata Apache Iceberg untuk dibuat dalam katalog.

Bidang

- `MetadataOperation` – Wajib: String UTF-8 (nilai yang valid: `CREATE`).

  Operasi metadata yang diperlukan. Hanya dapat diatur ke`CREATE`.

- `Version` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Versi tabel untuk tabel Iceberg. Secara default ke 2.

OpenTableFormatInput struktur

Struktur yang mewakili tabel format terbuka.

Bidang

- `IcebergInput` — Sebuah objek IcebergInput.

  Menentukan `IcebergInput` struktur yang mendefinisikan tabel metadata Apache Iceberg.

ViewDefinition struktur

Struktur yang berisi detail untuk representasi.

Bidang

- `IsProtected` – Boolean.

  Anda dapat menyetel flag ini sebagai true untuk menginstruksikan mesin agar tidak mendorong operasi yang disediakan pengguna ke dalam rencana logis tampilan selama perencanaan kueri. Namun, pengaturan bendera ini tidak menjamin bahwa mesin akan mematuhi. Lihat dokumentasi mesin untuk memahami jaminan yang diberikan, jika ada.

- `Definer` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 512 byte, yang cocok dengan Single-line string pattern.

  Penentu tampilan di SQL.

- `SubObjects` — Susunan string UTF-8, tidak lebih dari 10 string.

  Daftar tabel Amazon Resource Names (ARN).

- `Representations` — Susunan objek ViewRepresentation, tidak kurang dari 1 atau tidak lebih dari 1000 struktur.

  Daftar representasi.

ViewDefinitionInput struktur

Struktur yang berisi detail untuk membuat atau memperbarui AWS Glue tampilan.

Bidang

- `IsProtected` – Boolean.

  Anda dapat menyetel flag ini sebagai true untuk menginstruksikan mesin agar tidak mendorong operasi yang disediakan pengguna ke dalam rencana logis tampilan selama perencanaan kueri. Namun, pengaturan bendera ini tidak menjamin bahwa mesin akan mematuhi. Lihat dokumentasi mesin untuk memahami jaminan yang diberikan, jika ada.
- `Definer` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 512 byte, yang cocok dengan Single-line string pattern.

  Penentu tampilan di SQL.
- `Representations`— Array ViewRepresentationInput objek, tidak kurang dari 1 atau lebih dari 10 struktur.

  Daftar struktur yang berisi dialek tampilan, dan kueri yang mendefinisikan tampilan.
- `SubObjects` — Susunan string UTF-8, tidak lebih dari 10 string.

  Daftar ARN tabel dasar yang membentuk tampilan.

ViewRepresentation struktur

Struktur yang berisi dialek tampilan, dan kueri yang mendefinisikan tampilan.

Bidang

- `Dialect` – String UTF-8 (nilai yang valid: `REDSHIFT` | `ATHENA` | `SPARK`).

  Dialek mesin kueri.
- `DialectVersion` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte.

  Versi dialek mesin kueri. Misalnya, 3.0.0.
- `ViewOriginalText` — String UTF-8, dengan panjang tidak lebih dari 409600 byte.

  `SELECT`Permintaan yang diberikan oleh pelanggan selama`CREATE VIEW DDL`. SQL ini tidak digunakan selama query pada tampilan (`ViewExpandedText`digunakan sebagai gantinya). `ViewOriginalText`digunakan untuk kasus-kasus seperti `SHOW CREATE VIEW` di mana pengguna ingin melihat perintah DDL asli yang membuat tampilan.
- `ViewExpandedText` — String UTF-8, dengan panjang tidak lebih dari 409600 byte.

SQL diperluas untuk tampilan. SQL ini digunakan oleh mesin saat memproses kueri pada tampilan. Mesin dapat melakukan operasi selama pembuatan tampilan `ViewOriginalText` untuk diubah menjadi`ViewExpandedText`. Sebagai contoh:

  - Pengidentifikasi yang sepenuhnya memenuhi syarat: `SELECT * from table1 -> SELECT * from db1.table1`

- `ValidationConnection` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama koneksi yang akan digunakan untuk memvalidasi representasi spesifik dari tampilan.

- `IsStale` – Boolean.

  Dialek yang ditandai sebagai basi tidak lagi valid dan harus diperbarui sebelum dapat ditanyakan di mesin kueri masing-masing.

ViewRepresentationInput struktur

Struktur yang berisi detail representasi untuk memperbarui atau membuat tampilan Lake Formation.

Bidang

- `Dialect` – String UTF-8 (nilai yang valid: `REDSHIFT` | `ATHENA` | `SPARK`).

  Parameter yang menentukan jenis mesin dari representasi tertentu.

- `DialectVersion` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte.

  Parameter yang menentukan versi mesin representasi tertentu.

- `ViewOriginalText` — String UTF-8, dengan panjang tidak lebih dari 409600 byte.

  Sebuah string yang mewakili query SQL asli yang menggambarkan tampilan.

- `ValidationConnection` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama koneksi yang akan digunakan untuk memvalidasi representasi spesifik dari tampilan.

- `ViewExpandedText` — String UTF-8, dengan panjang tidak lebih dari 409600 byte.

  String yang mewakili kueri SQL yang menjelaskan tampilan dengan ARN sumber daya yang diperluas

Operasi

- CreateTable tindakan (Python: create_table)
- UpdateTable tindakan (Python: update_table)
- DeleteTable tindakan (Python: delete_table)
- BatchDeleteTable tindakan (Python: batch_delete_table)
- GetTable tindakan (Python: get_table)
- GetTables tindakan (Python: get_tables)
- GetTableVersion tindakan (Python: get_table_version)
- GetTableVersions tindakan (Python: get_table_versions)
- DeleteTableVersion tindakan (Python: delete_table_version)
- BatchDeleteTableVersion tindakan (Python: batch_delete_table_version)
- SearchTables tindakan (Python: search_tables)
- GetPartitionIndexes tindakan (Python: get_partition_indexes)
- CreatePartitionIndex tindakan (Python: create_partition_index)
- DeletePartitionIndex tindakan (Python: delete_partition_index)
- GetColumnStatisticsForTable tindakan (Python: get_column_statistics_for_table)
- UpdateColumnStatisticsForTable tindakan (Python: update_column_statistics_for_table)
- DeleteColumnStatisticsForTable tindakan (Python: delete_column_statistics_for_table)

CreateTable tindakan (Python: create_table)

Menciptakan sebuah definisi tabel baru dalam Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat `Table` dibuat. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Basis data katalog di mana tabel baru akan dibuat. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.

- `TableInput` — Wajib: Sebuah objek TableInput.

  Objek `TableInput` yang mendefinisikan tabel metadata yang akan dibuat dalam katalog.

- `PartitionIndexes` — Susunan objek PartitionIndex, tidak lebih dari 3 struktur.

  Daftar indeks partisi, struktur `PartitionIndex`, yang akan dibuat dalam tabel.

- `TransactionId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #16.

  ID transaksi.

- `OpenTableFormatInput` — Sebuah objek OpenTableFormatInput.

  Menentukan `OpenTableFormatInput` struktur saat membuat tabel format terbuka.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `AlreadyExistsException`
- `InvalidInputException`
- `EntityNotFoundException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ConcurrentModificationException`
- `ResourceNotReadyException`

UpdateTable tindakan (Python: update_table)

Memperbarui tabel metadata dalam Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat tabel berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.

- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat tabel berada. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.

- `TableInput` — Wajib: Sebuah objek TableInput.

  Objek `TableInput` yang diperbarui untuk menentukan tabel metadata dalam katalog.

- `SkipArchive` – Boolean.

  Secara default, `UpdateTable` selalu membuat versi diarsipkan dari tabel tersebut sebelum memperbaruinya. Namun, jika `skipArchive` diatur ke BETUL, `UpdateTable` tidak akan membuat versi yang diarsipkan.

- `TransactionId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #16.

  ID transaksi untuk memperbarui isi tabel.

- `VersionId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID versi untuk memperbarui isi tabel.

- `ViewUpdateAction` – String UTF-8 (nilai yang valid: `ADD` | `REPLACE` | `ADD_OR_REPLACE` | `DROP`).

  Operasi yang akan dilakukan saat memperbarui tampilan.

- `Force` – Boolean.

  Bendera yang dapat disetel ke true untuk mengabaikan deskriptor penyimpanan yang cocok dan persyaratan pencocokan subobject.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`
- `ResourceNumberLimitExceededException`
- `GlueEncryptionException`
- `ResourceNotReadyException`

DeleteTable tindakan (Python: delete_table)

Menghapus sebuah definisi tabel dari Katalog Data.

> **Note**
>
> Setelah menyelesaikan operasi ini, Anda tidak lagi memiliki akses ke versi tabel dan partisi yang merupakan milik tabel yang dihapus. AWS Glue menghapus sumber daya "yatim piatu" secara asinkron pada waktu yang tepat, atas kebijaksanaan layanan.
> Untuk memastikan penghapusan langsung dari semua sumber daya terkait, sebelum memanggil `DeleteTable`, gunakan `DeleteTableVersion` atau `BatchDeleteTableVersion`, dan `DeletePartition` atau `BatchDeletePartition`, untuk menghapus sumber daya yang dimiliki oleh tabel.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat tabel berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.

- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat tabel berada. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel yang akan dihapus. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.

- `TransactionId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #16.

  ID transaksi untuk menghapus isi tabel.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`
- `ResourceNotReadyException`

BatchDeleteTable tindakan (Python: batch_delete_table)

Menghapus beberapa tabel sekaligus.

> **Note**
>
> Setelah menyelesaikan operasi ini, Anda tidak lagi memiliki akses ke versi tabel dan partisi yang merupakan milik tabel yang dihapus. AWS Glue menghapus sumber daya "yatim piatu" secara asinkron pada waktu yang tepat, atas kebijaksanaan layanan.

Untuk memastikan penghapusan langsung dari semua sumber daya terkait, sebelum memanggil `BatchDeleteTable`, gunakan `DeleteTableVersion` atau `BatchDeleteTableVersion`, dan `DeletePartition` atau `BatchDeletePartition`, untuk menghapus sumber daya yang dimiliki oleh tabel.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat tabel berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog di mana tabel yang akan dihapus berada. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `TablesToDelete` — Wajib: Susunan string UTF-8, tidak lebih dari 100 string.

  Daftar tabel yang akan dihapus.
- `TransactionId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #16.

  ID transaksi untuk menghapus isi tabel.

Respons

- `Errors` – Susunan objek TableError.

  Daftar kesalahan yang ditemui saat berusaha untuk menghapus tabel tertentu.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`

- `OperationTimeoutException`
- `GlueEncryptionException`
- `ResourceNotReadyException`

GetTable tindakan (Python: get_table)

Mengambil definisi `Table` dalam Katalog Data untuk tabel yang ditentukan.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat tabel berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data dalam katalog tempat tabel berada. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel yang untuknya definisi akan diambil. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `TransactionId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #16.

  ID transaksi untuk membaca isi tabel.
- `QueryAsOfTime` — Stempel waktu.

  Waktu kapan harus membaca isi tabel. Jika tidak diatur, waktu komit transaksi terbaru akan digunakan. Tidak dapat ditentukan bersama dengan`TransactionId`.

Respons

- `Table` — Sebuah objek Tabel.

Objek `Table` yang mendefinisikan tabel yang ditentukan.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `ResourceNotReadyException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetTables tindakan (Python: get_tables)

Mengambil definisi dari beberapa atau semua tabel dalam `Database` yang ditentukan.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat tabel-tabel tersebut berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Basis data dalam katalog yang tabelnya akan dicantumkan. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `Expression` — String UTF-8, sepanjang tidak lebih dari 2048, yang cocok dengan Single-line string pattern.

  Sebuah pola ekspresi reguler. Jika ada, hanya tabel yang namanya cocok dengan pola saja yang dikembalikan.
- `NextToken` – String UTF-8.

Sebuah token kelanjutan, disertakan jika ini adalah sebuah panggilan kelanjutan.

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100.

  Jumlah maksimum tabel yang akan dikembalikan dalam satu respons tunggal.

- `TransactionId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #16.

  ID transaksi untuk membaca isi tabel.

- `QueryAsOfTime` — Stempel waktu.

  Waktu kapan harus membaca isi tabel. Jika tidak diatur, waktu komit transaksi terbaru akan digunakan. Tidak dapat ditentukan bersama dengan`TransactionId`.

Respons

- `TableList` – Susunan objek Tabel.

  Daftar objek `Table` yang diminta.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, ada jika segmen daftar saat ini bukan yang terakhir.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetTableVersion tindakan (Python: get_table_version)

Mengambil versi tertentu dari sebuah tabel.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat tabel-tabel tersebut berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Basis data dalam katalog tempat tabel berada. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `VersionId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nilai ID dari versi tabel yang akan diambil. Sebuah `VersionID` adalah representasi string dari bilangan bulat. Setiap versi bertambah 1.

Respons

- `TableVersion` — Sebuah objek TableVersion.

  Versi tabel yang diminta.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

GetTableVersions tindakan (Python: get_table_versions)

Mengambil daftar string yang mengidentifikasi versi yang tersedia dari sebuah tabel yang ditentukan.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat tabel-tabel tersebut berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Basis data dalam katalog tempat tabel berada. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini bukan panggilan pertama.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100.

  Jumlah maksimum versi tabel yang akan dikembalikan dalam satu respons.

Respons

- `TableVersions` – Susunan objek TableVersion.

  Daftar string yang mengidentifikasi versi yang tersedia dari sebuah tabel yang ditentukan.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika daftar versi yang tersedia tidak termasuk yang terakhir.

Kesalahan

- `EntityNotFoundException`

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeleteTableVersion tindakan (Python: delete_table_version)

Menghapus versi tertentu dari sebuah tabel.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat tabel-tabel tersebut berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Basis data dalam katalog tempat tabel berada. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `VersionId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari versi tabel yang akan dihapus. Sebuah `VersionID` adalah representasi string dari bilangan bulat. Setiap versi bertambah 1.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchDeleteTableVersion tindakan (Python: batch_delete_table_version)

Menghapus batch versi tertentu dari sebuah tabel.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat tabel-tabel tersebut berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Basis data dalam katalog tempat tabel berada. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel. Untuk kompatibilitas Hive, nama ini semuanya harus huruf kecil.
- `VersionIds` — Wajib: Susunan string UTF-8, tidak lebih dari 100 string.

  Daftar ID versi yang akan dihapus. Sebuah `VersionId` adalah representasi string dari bilangan bulat. Setiap versi bertambah 1.

Respons

- `Errors` – Susunan objek TableVersionError.

  Daftar kesalahan yang ditemui saat berusaha menghapus versi tabel yang ditentukan.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

SearchTables tindakan (Python: search_tables)

Mencari satu set tabel berdasarkan properti dalam metadata tabel dan pada basis data induk. Anda dapat mencari berdasarkan teks atau syarat filter.

Anda hanya bisa mendapatkan tabel yang dapat Anda akses berdasarkan kebijakan keamanan yang ditetapkan dalam Lake Formation. Anda memerlukan setidaknya akses baca-saja ke tabel yang untuknya akan dikembalikan. Jika Anda tidak memiliki akses ke semua kolom dalam tabel, maka kolom-kolom tersebut tidak akan dicari saat mengembalikan daftar tabel kembali kepada Anda. Jika Anda memiliki akses ke kolom tetapi tidak ke data dalam kolom, maka kolom tersebut dan metadata yang dikaitkan ke kolom tersebut akan dimasukkan dalam pencarian.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik, terdiri dari *`account_id`*.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, disertakan jika ini adalah sebuah panggilan kelanjutan.
- `Filters` – Susunan objek PropertyPredicate.

  Sebuah daftar pasangan nilai-kunci, dan sebuah pembanding digunakan untuk mem-filter hasil pencarian. Mengembalikan semua entitas yang cocok dengan predikat.

  Anggota `Comparator` dari struct `PropertyPredicate` hanya digunakan untuk bidang waktu, dan dapat dihilangkan untuk jenis bidang lainnya. Selain itu, ketika membandingkan nilai string, seperti ketika `Key=Name`, algoritme kecocokan fuzzy digunakan. Bidang `Key` (misalnya, nilai bidang `Name`) dibagi berdasarkan karakter tanda baca tertentu, misalnya, -, :, #, dll. menjadi token. Kemudian setiap token adalah dibandingkan secara exact-match dengan anggota `Value` dari

`PropertyPredicate`. Misalnya, jika `Key=Name` dan `Value=link`, tabel bernama `customer-link` dan `xx-link-yy` dikembalikan, tapi `xxlinkyy` tidak dikembalikan.

- `SearchText` — String nilai, dengan panjang tidak lebih dari 1024 byte.

  String yang digunakan untuk pencarian teks.

  Menentukan nilai dalam filter kutipan berdasarkan kecocokan persis dengan nilai.

- `SortCriteria` — Susunan objek SortCriterion, tidak lebih dari 1 struktur.

  Daftar kriteria untuk mengurutkan hasil berdasarkan nama bidang, dalam urutan menaik atau menurun.

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah maksimum tabel yang akan dikembalikan dalam satu respons tunggal.

- `ResourceShareType` – String UTF-8 (nilai yang valid: `FOREIGN` | `ALL` | `FEDERATED`).

  Memungkinkan Anda untuk menentukan apakah Anda ingin mencari tabel yang dibagi dengan akun Anda. Nilai yang diijinkan adalah `FOREIGN` atau `ALL`.

  - Jika diatur ke `FOREIGN`, akan mencari tabel yang dibagikan dengan akun Anda.
  - Jika diatur ke `ALL`, akan mencari tabel yang dibagikan dengan akun Anda, serta tabel di akun lokal Anda.

Respons

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, ada jika segmen daftar saat ini bukan yang terakhir.

- `TableList` – Susunan objek Tabel.

  Daftar objek `Table` yang diminta. Respons `SearchTables` hanya mengembalikan tabel yang Anda miliki aksesnya.

Kesalahan

- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

GetPartitionIndexes tindakan (Python: get_partition_indexes)

Mengambil indeks partisi yang dikaitkan dengan sebuah tabel.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID katalog tempat tabel berada.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Menentukan nama basis data tempat Anda ingin mengambil indeks partisi.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Menentukan nama tabel yang untuknya Anda ingin mengambil indeks partisi.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, disertakan jika ini adalah sebuah panggilan kelanjutan.

Respons

- `PartitionIndexDescriptorList` – Susunan objek PartitionIndexDescriptor.

  Daftar deskriptor indeks.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, ada jika segmen daftar saat ini bukan yang terakhir.

Kesalahan

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `EntityNotFoundException`
- `ConflictException`

CreatePartitionIndex tindakan (Python: create_partition_index)

Menciptakan sebuah indeks partisi tertentu dalam tabel yang ada.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID katalog tempat tabel berada.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Menentukan nama sebuah basis data di mana Anda ingin membuat sebuah indeks partisi.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Menentukan nama sebuah tabel di mana Anda ingin membuat sebuah indeks partisi.
- `PartitionIndex` — Wajib: Sebuah objek PartitionIndex.

  Menentukan struktur `PartitionIndex` untuk membuat sebuah indeks partisi dalam tabel yang ada.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `AlreadyExistsException`
- `InvalidInputException`
- `EntityNotFoundException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeletePartitionIndex tindakan (Python: delete_partition_index)

Menghapus sebuah indeks partisi tertentu dari tabel yang ada.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID katalog tempat tabel berada.

- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Menentukan nama sebuah basis data yang ingin Anda hapus indeks partisinya.

- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Menentukan nama sebuah tabel yang ingin Anda hapus indeks partisinya.

- `IndexName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama indeks partisi yang akan dihapus.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `EntityNotFoundException`
- `ConflictException`
- `GlueEncryptionException`

GetColumnStatisticsForTable tindakan (Python: get_column_statistics_for_table)

Mengambil statistik tabel kolom.

Izin Identity and Access Management (IAM) yang diperlukan untuk operasi ini adalah `GetTable`.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat partisi yang dimaksud berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat partisi berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel partisi.
- `ColumnNames` — Wajib: Susunan string UTF-8, tidak lebih dari 100 string.

  Daftar nama kolom.

Respons

- `ColumnStatisticsList` – Susunan objek ColumnStatistics.

  Daftar ColumnStatistics.
- `Errors` – Susunan objek ColumnError.

  Daftar ColumnStatistics yang gagal diambil.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`

- `OperationTimeoutException`
- `GlueEncryptionException`

UpdateColumnStatisticsForTable tindakan (Python: update_column_statistics_for_table)

Menciptakan atau memperbarui statistik tabel kolom.

Izin Identity and Access Management (IAM) yang diperlukan untuk operasi ini adalah `UpdateTable`.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat partisi yang dimaksud berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat partisi berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel partisi.
- `ColumnStatisticsList` — Wajib: Susunan objek ColumnStatistics, tidak lebih dari 25 struktur.

  Daftar statistik kolom.

Respons

- `Errors` – Susunan objek ColumnStatisticsError.

  Daftar ColumnStatisticsErrors.

Kesalahan

- `EntityNotFoundException`

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeleteColumnStatisticsForTable tindakan (Python: delete_column_statistics_for_table)

Mengambil statistik tabel kolom.

Izin Identity and Access Management (IAM) yang diperlukan untuk operasi ini adalah `DeleteTable`.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat partisi yang dimaksud berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat partisi berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel partisi.
- `ColumnName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama kolom.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

API partisi

API Partisi menjelaskan tipe data dan operasi yang digunakan untuk bekerja dengan partisi.

Jenis data

- Struktur partisi
- PartitionInput struktur
- PartitionSpecWithSharedStorageDescriptor struktur
- PartitionListComposingSpec struktur
- PartitionSpecProxy struktur
- PartitionValueList struktur
- Struktur segmen
- PartitionError struktur
- BatchUpdatePartitionFailureEntry struktur
- BatchUpdatePartitionRequestEntry struktur
- StorageDescriptor struktur
- SchemaReference struktur
- SerDeInfo struktur
- SkewedInfo struktur

Struktur partisi

Merupakan sepotong data tabel.

Bidang

- `Values` – Susunan string UTF-8.

Nilai-nilai partisi.

- `DatabaseName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat partisi akan dibuat.

- `TableName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel basis data tempat partisi akan dibuat.

- `CreationTime` — Stempel waktu.

  Waktu pembuatan partisi.

- `LastAccessTime` — Stempel waktu.

  Waktu saat partisi terakhir kali diakses.

- `StorageDescriptor` — Sebuah objek StorageDescriptor.

  Sediakan informasi tentang lokasi fisik di mana partisi disimpan.

- `Parameters` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah String kunci, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 512000 byte.

  Pasangan nilai kunci ini menentukan parameter partisi.

- `LastAnalyzedTime` — Stempel waktu.

  Terakhir kali statistik kolom dikomputasi untuk partisi ini.

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat partisi berada.

PartitionInput struktur

Struktur yang digunakan untuk membuat dan memperbarui partisi.

Bidang

- `Values` – Susunan string UTF-8.

  Nilai-nilai partisi. Meskipun parameter ini tidak diperlukan oleh SDK, Anda harus menentukan parameter ini untuk input yang valid.

  Nilai kunci untuk partisi baru harus dilewatkan sebagai array objek String yang harus dipesan dalam urutan yang sama seperti kunci partisi yang muncul di prefiks Amazon S3. Jika tidak, AWS Glue akan menambahkan nilai ke kunci yang salah.
- `LastAccessTime` — Stempel waktu.

  Waktu saat partisi terakhir kali diakses.
- `StorageDescriptor` — Sebuah objek StorageDescriptor.

  Sediakan informasi tentang lokasi fisik di mana partisi disimpan.
- `Parameters` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah String kunci, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 512000 byte.

  Pasangan nilai kunci ini menentukan parameter partisi.
- `LastAnalyzedTime` — Stempel waktu.

  Terakhir kali statistik kolom dikomputasi untuk partisi ini.

PartitionSpecWithSharedStorageDescriptor struktur

Sebuah spesifikasi partisi untuk partisi yang berbagi lokasi fisik.

Bidang

- `StorageDescriptor` — Sebuah objek StorageDescriptor.

  Informasi penyimpanan fisik bersama.
- `Partitions` – Susunan objek Partisi.

  Daftar partisi yang berbagi lokasi fisik ini.

PartitionListComposingSpec struktur

Daftar partisi terkait.

Bidang

- `Partitions` – Susunan objek Partisi.

  Daftar partisi dalam spesifikasi penyusunan.

PartitionSpecProxy struktur

Menyediakan path root untuk partisi yang ditentukan.

Bidang

- `DatabaseName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Katalog basis data di mana partisi berada.
- `TableName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel yang berisi partisi.
- `RootPath` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Path akar dari proxy untuk pengalamatan partisi.
- `PartitionSpecWithSharedSD` — Sebuah objek PartitionSpecWithSharedStorageDescriptor.

  Spesifikasi partisi yang berbagi lokasi penyimpanan fisik yang sama.
- `PartitionListComposingSpec` — Sebuah objek PartitionListComposingSpec.

  Menentukan daftar partisi.

PartitionValueList struktur

Berisi daftar nilai yang mendefinisikan partisi.

Bidang

- `Values` – Wajib: Susunan string UTF-8.

  Daftar nilai.

Struktur segmen

Mendefinisikan wilayah yang tidak bertumpang tindih dari partisi tabel ini, memungkinkan beberapa permintaan untuk dijalankan secara paralel.

Bidang

- `SegmentNumber` — Wajib: Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Nomor indeks berbasis-nol dari segmen. Sebagai contoh, jika jumlah segmen adalah 4, nilai `SegmentNumber` berkisar dari 0 sampai 3.
- `TotalSegments` — Wajib: Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 10.

  Jumlah total segmen.

PartitionError struktur

Berisi informasi tentang sebuah kesalahan partisi.

Bidang

- `PartitionValues` – Susunan string UTF-8.

  Nilai yang menentukan partisi.
- `ErrorDetail` — Sebuah objek ErrorDetail.

  Detail tentang kesalahan partisi.

BatchUpdatePartitionFailureEntry struktur

Berisi informasi tentang kesalahan partisi pembaruan batch.

Bidang

- `PartitionValueList` — Susunan string UTF-8, tidak lebih dari 100 string.

  Daftar nilai yang mendefinisikan partisi.
- `ErrorDetail` — Sebuah objek ErrorDetail.

  Detail tentang kesalahan partisi pembaruan batch.

BatchUpdatePartitionRequestEntry struktur

Struktur yang berisi nilai-nilai dan struktur yang digunakan untuk memperbarui partisi.

Bidang

- `PartitionValueList` — Wajib: Susunan string UTF-8, tidak lebih dari 100 string.

  Daftar nilai yang mendefinisikan partisi.
- `PartitionInput` — Wajib: Sebuah objek PartitionInput.

  Struktur yang digunakan untuk memperbarui sebuah partisi.

StorageDescriptor struktur

Menjelaskan penyimpanan fisik data tabel.

Bidang

- `Columns` – Susunan objek Kolom.

  Daftar `Columns` dalam tabel.
- `Location` — String lokasi, sepanjang tidak lebih dari 2056, yang cocok dengan URI address multi-line string pattern.

  Lokasi fisik tabel. Secara default, ini mengambil bentuk lokasi gudang, yang diikuti oleh lokasi basis data di gudang, yang diikuti dengan nama tabel.
- `AdditionalLocations` – Susunan string UTF-8.

  Daftar lokasi yang mengarah ke jalur tempat tabel Delta berada.

- `InputFormat` — String format, sepanjang tidak lebih dari 128, yang cocok dengan Single-line string pattern.

  Format input: `SequenceFileInputFormat` (biner), atau `TextInputFormat`, atau format kustom.

- `OutputFormat` — String format, sepanjang tidak lebih dari 128, yang cocok dengan Single-line string pattern.

  Format output: `SequenceFileOutputFormat` (biner), atau `IgnoreKeyTextOutputFormat`, atau format kustom.

- `Compressed` – Boolean.

  `True` jika data dalam tabel dikompresi, atau `False` jika tidak.

- `NumberOfBuckets` — Nomor (bilangan bulat).

  Harus ditentukan jika tabel berisi kolom dimensi.

- `SerdeInfo` — Sebuah objek SerDeInfo.

  Informasi serialisasi/deserialisasi (). SerDe

- `BucketColumns` – Susunan string UTF-8.

  Daftar kolom pengelompokan peredam, kolom pengklasteran, dan kolom pem-bucket-an dalam tabel.

- `SortColumns` – Susunan objek Order.

  Daftar yang menentukan urutan dari setiap bucket dalam tabel.

- `Parameters` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah String kunci, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 512000 byte.

  Properti yang disediakan pengguna dalam bentuk nilai kunci.

- `SkewedInfo` — Sebuah objek SkewedInfo.

  Informasi tentang nilai yang sering muncul di kolom (nilai kecenderungan).

- `StoredAsSubDirectories` – Boolean.

`True` jika data tabel disimpan di subdirektori, atau `False` jika tidak.

- `SchemaReference` — Sebuah objek SchemaReference.

  Objek yang mereferensikan skema yang disimpan dalam AWS Glue Schema Registry.

  Saat membuat sebuah tabel, Anda dapat memberikan daftar kosong kolom untuk skema, dan sebaliknya menggunakan referensi skema.

SchemaReference struktur

Objek yang mereferensikan skema yang disimpan dalam AWS Glue Schema Registry.

Bidang

- `SchemaId` — Sebuah objek SchemaId.

  Struktur yang berisi bidang identitas skema. Baik ini atau `SchemaVersionId` harus disediakan.
- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  ID unik yang ditetapkan untuk sebuah versi skema. Baik ini atau `SchemaId` harus disediakan.
- `SchemaVersionNumber` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Nomor versi dari skema.

SerDeInfo struktur

Informasi tentang program serialisasi/deserialisasi (SerDe) yang berfungsi sebagai extractor dan loader.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama dari SerDe.
- `SerializationLibrary` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Biasanya kelas yang mengimplementasikan. SerDe Contohnya adalah `org.apache.hadoop.hive.serde2.columnar.ColumnarSerDe`.

- `Parameters` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah String kunci, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 512000 byte.

  Pasangan kunci-nilai ini menentukan parameter inisialisasi untuk. SerDe

SkewedInfo struktur

Menentukan nilai menyimpang dalam sebuah tabel. Nilai menyimpang adalah nilai yang terjadi dengan frekuensi sangat tinggi.

Bidang

- `SkewedColumnNames` – Susunan string UTF-8.

  Daftar nama kolom yang berisi nilai-nilai menyimpang.
- `SkewedColumnValues` – Susunan string UTF-8.

  Daftar nilai yang muncul sangat sering untuk dianggap menyimpang.
- `SkewedColumnValueLocationMaps` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

  Pemetaan nilai menyimpang untuk kolom yang berisi nilai tersebut.

Operasi

- CreatePartition tindakan (Python: create_partition)
- BatchCreatePartition tindakan (Python: batch_create_partition)
- UpdatePartition tindakan (Python: update_partition)
- DeletePartition tindakan (Python: delete_partition)

- BatchDeletePartition tindakan (Python: batch_delete_partition)
- GetPartition tindakan (Python: get_partition)
- GetPartitions tindakan (Python: get_partitions)
- BatchGetPartition tindakan (Python: batch_get_partition)
- BatchUpdatePartition tindakan (Python: batch_update_partition)
- GetColumnStatisticsForPartition tindakan (Python: get_column_statistics_for_partition)
- UpdateColumnStatisticsForPartition tindakan (Python: update_column_statistics_for_partition)
- DeleteColumnStatisticsForPartition tindakan (Python: delete_column_statistics_for_partition)

CreatePartition tindakan (Python: create_partition)

Membuat sebuah partisi baru.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID AWS akun katalog tempat partisi akan dibuat.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data metadata di mana partisi akan dibuat.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel metadata di mana partisi akan dibuat.
- `PartitionInput` — Wajib: Sebuah objek PartitionInput.

  Struktur `PartitionInput` mendefinisikan partisi yang akan dibuat.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `InvalidInputException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `GlueEncryptionException`

BatchCreatePartition tindakan (Python: batch_create_partition)

Menciptakan satu atau beberapa partisi dalam operasi batch.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID katalog di mana partisi akan dibuat. Saat ini, ini harus menjadi ID AWS akun.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data metadata di mana partisi akan dibuat.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel metadata di mana partisi akan dibuat.
- `PartitionInputList` — Wajib: Susunan objek PartitionInput, tidak lebih dari 100 struktur.

  Daftar struktur `PartitionInput` yang menentukan partisi yang akan dibuat.

Respons

- `Errors` – Susunan objek PartitionError.

  Kesalahan yang dihadapi saat mencoba membuat partisi yang diminta.

Kesalahan

- `InvalidInputException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `GlueEncryptionException`

UpdatePartition tindakan (Python: update_partition)

Memperbarui sebuah partisi.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data di mana partisi yang akan diperbarui berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog di mana tabel yang dimaksud berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel tempat partisi yang akan diperbarui berada.
- `PartitionValueList` — Wajib: Susunan string UTF-8, tidak lebih dari 100 string.

  Daftar nilai kunci partisi yang menentukan partisi yang akan diperbarui.
- `PartitionInput` — Wajib: Sebuah objek PartitionInput.

  Objek partisi baru untuk memperbarui partisi.

Properti `Values` tidak dapat diubah. Jika Anda ingin mengubah nilai kunci partisi untuk sebuah partisi, hapus dan buat ulang partisi.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeletePartition tindakan (Python: delete_partition)

Menghapus partisi yang ditentukan.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data di mana partisi yang akan dihapus berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog di mana tabel yang dimaksud berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel yang berisi partisi yang akan dihapus.
- `PartitionValues` – Wajib: Susunan string UTF-8.

Nilai yang menentukan partisi.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchDeletePartition tindakan (Python: batch_delete_partition)

Menghapus satu atau beberapa partisi dalam sebuah operasi batch.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data di mana partisi yang akan dihapus berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog di mana tabel yang dimaksud berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel yang berisi partisi yang akan dihapus.
- `PartitionsToDelete` — Wajib: Susunan objek PartitionValueDaftar, tidak lebih dari 25 struktur.

  Daftar struktur `PartitionInput` yang menentukan partisi yang akan dihapus.

Respons

- `Errors` – Susunan objek PartitionError.

  Kesalahan yang ditemui saat mencoba menghapus partisi yang diminta.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetPartition tindakan (Python: get_partition)

Mengambil informasi tentang sebuah partisi yang ditentukan.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat partisi yang dimaksud berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat partisi berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel partisi.
- `PartitionValues` – Wajib: Susunan string UTF-8.

  Nilai yang menentukan partisi.

Respons

- `Partition` — Sebuah objek Partisi.

  Informasi yang diminta, dalam bentuk objek `Partition`.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`
- `FederationSourceException`
- `FederationSourceRetryableException`

GetPartitions tindakan (Python: get_partitions)

Mengambil informasi tentang partisi dalam sebuah tabel.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat partisi yang dimaksud berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat partisi berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel partisi.
- `Expression` — String predikat, sepanjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

Ekspresi yang menyaring partisi yang akan dikembalikan.

Ekspresi menggunakan sintaksis SQL yang mirip dengan klausul filter `WHERE` SQL. Pengurai pernyataan SQL JSQLParser mem-parsing ekspresi.

Operator: Berikut ini adalah operator yang dapat Anda gunakan dalam panggilan API `Expression`:

=

Periksa apakah nilai-nilai dari dua operan adalah sama; jika ya, maka syarat menjadi BETUL.

Contoh: Asumsikan 'variabel a' adalah 10 dan 'variabel b' adalah 20.

(a = b) tidak BETUL.

< >

Periksa apakah nilai-nilai dari dua operan adalah sama; jika nilainya tidak sama, maka syarat menjadi BETUL.

Contoh: (< > b) BETUL.

>

Periksa apakah nilai operan kiri lebih besar dari nilai operan kanan; jika ya, maka syarat menjadi BETUL.

Contoh: (a> b) tidak BETUL.

<

Periksa apakah nilai operan kiri lebih kecil dari nilai operan kanan; jika ya, maka syarat menjadi BETUL.

Contoh: (a < b) adalah BETUL.

>=

Periksa apakah nilai operan kiri lebih besar dari atau sama dengan nilai operan kanan; jika ya, maka syarat menjadi BETUL.

Contoh: (a> = b) tidak BETUL.

<=

Periksa apakah nilai operan kiri lebih kecil dari atau sama dengan nilai operan kanan; jika ya, maka syarat menjadi BETUL.

Contoh: (a <= b) benar.

DAN, ATAU, DI, ANTARA, SEPERTI, TIDAK, ADALAH NOL

Operator logistik.

Jenis Kunci Partisi yang Didukung: Berikut ini adalah kunci partisi yang didukung.

- `string`
- `date`
- `timestamp`
- `int`
- `bigint`
- `long`
- `tinyint`
- `smallint`
- `decimal`

Jika ada jenis kunci yang ditemui yang tidak valid, maka pengecualian dilemparkan.

Daftar berikut menunjukkan operator yang valid pada masing-masing jenis. Bila Anda menentukan sebuah crawler, jenis `partitionKey` dibuat sebagai `STRING`, agar kompatibel dengan partisi katalog.

Panggilan API Sampel:

Example

Tabel `twitter_partition` memiliki tiga partisi:

```
year = 2015
        year = 2016
        year = 2017
```

Example

Ambil partisi `year` sama dengan 2015

```
aws glue get-partitions --database-name dbname --table-name twitter_partition
        --expression "year*=*'2015'"
```

Example

Ambil partisi `year` antara 2016 dan 2018 (eksklusif)

```
aws glue get-partitions --database-name dbname --table-name twitter_partition
        --expression "year>'2016' AND year<'2018'"
```

Example

Ambil partisi `year` antara 2015 dan 2018 (inklusif). Panggilan API berikut setara satu sama lain:

```
aws glue get-partitions --database-name dbname --table-name twitter_partition
        --expression "year>='2015' AND year<='2018'"

        aws glue get-partitions --database-name dbname --table-name
 twitter_partition
        --expression "year BETWEEN 2015 AND 2018"

        aws glue get-partitions --database-name dbname --table-name
 twitter_partition
        --expression "year IN (2015,2016,2017,2018)"
```

Example

Filter partisi wildcard, di mana output panggilan berikut adalah partisi tahun=2017. Sebuah ekspresi reguler tidak didukung di `LIKE`.

```
aws glue get-partitions --database-name dbname --table-name twitter_partition
        --expression "year LIKE '%7'"
```

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini bukan panggilan pertama untuk mengambil partisi ini.

- `Segment` — Sebuah objek Segment.

  Segmen tabel partisi yang akan dipindai dalam permintaan ini.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah maksimum partisi yang akan dikembalikan dalam satu respons tunggal.
- `ExcludeColumnSchema` – Boolean.

  Ketika benar, menentukan tidak mengembalikan skema kolom partisi. Berguna bila Anda hanya tertarik pada atribut partisi lain seperti nilai partisi atau lokasi. Pendekatan ini menghindari masalah respons besar dengan tidak mengembalikan data duplikat.
- `TransactionId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #16.

  ID transaksi untuk membaca isi partisi.
- `QueryAsOfTime` — Stempel waktu.

  Waktu kapan harus membaca isi partisi. Jika tidak diatur, waktu komit transaksi terbaru akan digunakan. Tidak dapat ditentukan bersama dengan`TransactionId`.

Respons

- `Partitions` – Susunan objek Partisi.

  Daftar partisi yang diminta.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika daftar partisi yang dikembalikan tidak termasuk yang terakhir.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`
- `InvalidStateException`

- `ResourceNotReadyException`
- `FederationSourceException`
- `FederationSourceRetryableException`

BatchGetPartition tindakan (Python: batch_get_partition)

Mengambil partisi dalam sebuah permintaan batch.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat partisi yang dimaksud berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat partisi berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel partisi.
- `PartitionsToGet` — Wajib: Susunan objek PartitionValueDaftar, tidak lebih dari 1000 struktur.

  Daftar nilai partisi yang mengidentifikasi partisi yang akan diambil.

Respons

- `Partitions` – Susunan objek Partisi.

  Daftar partisi yang diminta.
- `UnprocessedKeys` — Susunan objek PartitionValueDaftar, tidak lebih dari 1000 struktur.

  Daftar nilai partisi dalam permintaan yang untuknya partisi tidak dikembalikan.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`
- `InvalidStateException`
- `FederationSourceException`
- `FederationSourceRetryableException`

BatchUpdatePartition tindakan (Python: batch_update_partition)

Memperbarui satu atau beberapa partisi dalam sebuah operasi batch.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID katalog di mana partisi akan diperbarui. Saat ini, ini harus menjadi ID AWS akun.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data metadata di mana partisi akan diperbarui.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel metadata di mana partisi akan diperbarui.
- `Entries` — Wajib: Susunan objek BatchUpdatePartitionRequestEntri, tidak kurang dari 1 atau lebih dari 100 struktur.

  Daftar dari hingga 100 objek `BatchUpdatePartitionRequestEntry` yang akan diperbarui.

Respons

- `Errors` – Susunan objek BatchUpdatePartitionFailureEntri.

Kesalahan yang ditemui saat mencoba memperbarui partisi yang diminta. Sebuah daftar objek `BatchUpdatePartitionFailureEntry`.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`

GetColumnStatisticsForPartition tindakan (Python: get_column_statistics_for_partition)

Mengambil statistik partisi kolom.

Izin Identity and Access Management (IAM) yang diperlukan untuk operasi ini adalah `GetPartition`.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat partisi yang dimaksud berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat partisi berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel partisi.
- `PartitionValues` – Wajib: Susunan string UTF-8.

  Daftar nilai partisi yang mengidentifikasi partisi.
- `ColumnNames` — Wajib: Susunan string UTF-8, tidak lebih dari 100 string.

Daftar nama kolom.

Respons

- `ColumnStatisticsList` – Susunan objek ColumnStatistics.

  Daftar ColumnStatistics yang gagal diambil.
- `Errors` – Susunan objek ColumnError.

  Terjadi kesalahan saat mengambil data statistik kolom.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

UpdateColumnStatisticsForPartition tindakan (Python: update_column_statistics_for_partition)

Membuat atau memperbarui statistik partisi kolom.

Izin Identity and Access Management (IAM) yang diperlukan untuk operasi ini adalah `UpdatePartition`.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat partisi yang dimaksud berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Nama basis data katalog tempat partisi berada.

- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel partisi.

- `PartitionValues` – Wajib: Susunan string UTF-8.

  Daftar nilai partisi yang mengidentifikasi partisi.

- `ColumnStatisticsList` — Wajib: Susunan objek ColumnStatistics, tidak lebih dari 25 struktur.

  Daftar statistik kolom.

Respons

- `Errors` – Susnan objek ColumnStatisticsError.

  Terjadi kesalahan saat memperbarui data statistik kolom.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeleteColumnStatisticsForPartition tindakan (Python: delete_column_statistics_for_partition)

Menghapus statistik kolom partisi dari sebuah kolom.

Izin Identity and Access Management (IAM) yang diperlukan untuk operasi ini adalah `DeletePartition`.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat partisi yang dimaksud berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat partisi berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel partisi.
- `PartitionValues` – Wajib: Susunan string UTF-8.

  Daftar nilai partisi yang mengidentifikasi partisi.
- `ColumnName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama kolomnya.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

API Koneksi

Connection API menjelaskan tipe data AWS Glue koneksi, dan API untuk membuat, menghapus, memperbarui, dan mencantumkan koneksi.

Jenis data

- Struktur koneksi
- ConnectionInput struktur
- PhysicalConnectionRequirements struktur
- GetConnectionsFilter struktur

Struktur koneksi

Mendefinisikan sebuah koneksi ke sumber data.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi koneksi.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi koneksi.
- `ConnectionType` – String UTF-8 (nilai yang valid: `JDBC` | `SFTP` | `MONGODB` | `KAFKA` | `NETWORK` | `MARKETPLACE` | `CUSTOM` | `SALESFORCE`).

  Jenis koneksi. Saat ini, SFTP tidak didukung.
- `MatchCriteria` — Susunan string UTF-8, tidak lebih dari 10 string.

  Daftar kriteria yang dapat digunakan dalam memilih koneksi ini.
- `ConnectionProperties` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 100 pasang.

  Setiap kunci adalah string UTF-8 (nilai yang valid: `HOST` | `PORT` | `USERNAME="USER_NAME"` | `PASSWORD` | `ENCRYPTED_PASSWORD` | `JDBC_DRIVER_JAR_URI` | `JDBC_DRIVER_CLASS_NAME`

| JDBC_ENGINE | JDBC_ENGINE_VERSION | CONFIG_FILES INSTANCE_ID
| JDBC_CONNECTION_URL | JDBC_ENFORCE_SSL | CUSTOM_JDBC_CERT |
SKIP_CUSTOM_JDBC_CERT_VALIDATION | CUSTOM_JDBC_CERT_STRING |
CONNECTION_URL | KAFKA_BOOTSTRAP_SERVERS | KAFKA_SSL_ENABLED
| KAFKA_CUSTOM_CERT | KAFKA_SKIP_CUSTOM_CERT_VALIDATION |
KAFKA_CLIENT_KEYSTORE | KAFKA_CLIENT_KEYSTORE_PASSWORD |
KAFKA_CLIENT_KEY_PASSWORD | ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD
| ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD | SECRET_ID | CONNECTOR_URL
| CONNECTOR_TYPE | CONNECTOR_CLASS_NAME | KAFKA_SASL_MECHANISM
| KAFKA_SASL_PLAIN_USERNAME | KAFKA_SASL_PLAIN_PASSWORD |
ENCRYPTED_KAFKA_SASL_PLAIN_PASSWORD KAFKA_SASL_SCRAM_USERNAME
KAFKA_SASL_SCRAM_PASSWORD | KAFKA_SASL_SCRAM_SECRETS_ARN |
ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD | KAFKA_SASL_GSSAPI_KEYTAB
KAFKA_SASL_GSSAPI_KRB5_CONF | KAFKA_SASL_GSSAPI_SERVICE |
KAFKA_SASL_GSSAPI_PRINCIPAL |ROLE_ARN).

Masing-masing kunci adalah sebuah string Nilai, dengan panjang tidak lebih dari 1024 byte.

Pasangan nilai-kunci ini menentukan parameter untuk koneksi:

- HOST - URI host: baik fully qualified domain name (FQDN) atau alamat IPv4 dari basis data host.
- PORT - Nomor port, antara 1024 dan 65535, port di mana host basis data mendengarkan untuk koneksi basis data.
- USER_NAME - Nama yang digunakan untuk log in ke basis data. Nilai string untuk USER_NAME adalah "USERNAME".
- PASSWORD - Sebuah kata sandi, jika salah satu digunakan, untuk nama pengguna.
- ENCRYPTED_PASSWORD - Bila Anda mengaktifkan perlindungan kata sandi koneksi dengan menetapkan ConnectionPasswordEncryption dalam pengaturan enkripsi Katalog Data, bidang ini menyimpan kata sandi terenkripsi.
- JDBC_DRIVER_JAR_URI - Path Amazon Simple Storage Service (Amazon S3) dari file JAR yang berisi driver JDBC yang akan digunakan.
- JDBC_DRIVER_CLASS_NAME - Nama kelas driver JDBC yang akan digunakan.
- JDBC_ENGINE - Nama mesin JDBC yang akan digunakan.
- JDBC_ENGINE_VERSION - Versi mesin JDBC yang akan digunakan.
- CONFIG_FILES - (Dicadangkan untuk digunakan di masa depan.)
- INSTANCE_ID - ID instans yang akan digunakan.

- JDBC_CONNECTION_URL - URL untuk menghubungkan ke sumber data JDBC.
- JDBC_ENFORCE_SSL - Sebuah string Boolean (BETUL, SALAH) yang menentukan apakah Lapisan Soket Aman (SSL) dengan pencocokan hostname ditegakkan untuk koneksi JDBC pada klien. Default-nya adalah salah.
- CUSTOM_JDBC_CERT- Lokasi Amazon S3 yang menentukan sertifikat root pelanggan. AWS Glue menggunakan sertifikat root ini untuk memvalidasi sertifikat pelanggan saat menghubungkan ke database pelanggan. AWS Glue hanya menangani sertifikat X.509. Sertifikat yang diberikan harus dikodekan-DER dan disediakan dalam format PEM encoding Base64.
- SKIP_CUSTOM_JDBC_CERT_VALIDATION- Secara default, ini`false`. AWS Glue memvalidasi algoritma Signature dan Subject Public Key Algorithm untuk sertifikat pelanggan. Algoritme yang diizinkan untuk Algoritme Tanda Tangan adalah SHA256withRSA, SHA384withRSA or SHA512withRSA. Untuk Algoritme Kunci Publik Subjek, panjang kunci paling tidak 2048. Anda dapat mengatur nilai properti ini ke `true` untuk melompati validasi sertifikat pelanggan AWS Glue.
- CUSTOM_JDBC_CERT_STRING- String sertifikat JDBC kustom yang digunakan untuk pencocokan domain atau pencocokan nama yang dibedakan untuk mencegah serangan. man-in-the-middle Dalam basis data Oracle, ini digunakan sebagai SSL_SERVER_CERT_DN; di Microsoft SQL Server, ini digunakan sebagai `hostNameInCertificate`.
- CONNECTION_URL - URL untuk menghubungkan ke sumber data umum (non-JDBC).
- SECRET_ID - ID rahasia yang digunakan untuk secret manager kredensialnya.
- CONNECTOR_URL - URL konektor untuk koneksi MARKETPLACE atau KUSTOM.
- CONNECTOR_TYPE - Jenis konektor untuk koneksi MARKETPLACE atau KUSTOM.
- CONNECTOR_CLASS_NAME - Nama kelas konektor untuk koneksi MARKETPLACE atau KUSTOM.
- KAFKA_BOOTSTRAP_SERVERS - Sebuah daftar yang dipisahkan koma dari pasangan host dan port yang merupakan alamat dari broker Apache Kafka dalam klaster Kafka yang akan terhubung dengan klien Kafka dan bootstrap itu sendiri.
- KAFKA_SSL_ENABLED - Baik untuk mengaktifkan atau menonaktifkan SSL pada koneksi Apache Kafka. Nilai default-nya adalah "BETUL".
- KAFKA_CUSTOM_CERT - URL Amazon S3 untuk file sertifikat CA privat (format.pem). Default-nya adalah string kosong.
- KAFKA_SKIP_CUSTOM_CERT_VALIDATION- Apakah akan melewatkan validasi file sertifikat CA atau tidak. AWS Glue memvalidasi untuk tiga algoritma: SHA256withRSA, SHA384withRSA dan SHA512WithRSA. Nilai default-nya adalah "SALAH".

- KAFKA_CLIENT_KEYSTORE - Lokasi Amazon S3 dari file keystore klien untuk autentikasi sisi klien Kafka (Opsional).
- KAFKA_CLIENT_KEYSTORE_PASSWORD - Kata sandi untuk mengakses keystore yang disediakan (Opsional).
- KAFKA_CLIENT_KEY_PASSWORD - Sebuah keystore dapat terdiri dari beberapa kunci, jadi ini adalah kata sandi untuk mengakses kunci klien yang akan digunakan dengan kunci sisi klien Kafka (Opsional).
- ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD- Versi terenkripsi dari kata sandi keystore klien Kafka (jika pengguna memiliki pengaturan kata sandi AWS Glue enkripsi yang dipilih).
- ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD- Versi terenkripsi dari kata sandi kunci klien Kafka (jika pengguna memiliki pengaturan kata sandi AWS Glue enkripsi yang dipilih).
- KAFKA_SASL_MECHANISM-"SCRAM-SHA-512","GSSAPI","AWS_MSK_IAM", atau"PLAIN". Ini adalah Mekanisme SASL yang didukung.
- KAFKA_SASL_PLAIN_USERNAME- Nama pengguna plaintext yang digunakan untuk mengautentikasi dengan mekanisme “PLAIN”.
- KAFKA_SASL_PLAIN_PASSWORD- Kata sandi plaintext yang digunakan untuk mengautentikasi dengan mekanisme “PLAIN”.
- ENCRYPTED_KAFKA_SASL_PLAIN_PASSWORD- Versi terenkripsi dari kata sandi Kafka SASL PLAIN (jika pengguna memiliki pengaturan kata sandi AWS Glue enkripsi yang dipilih).
- KAFKA_SASL_SCRAM_USERNAME- Nama pengguna plaintext yang digunakan untuk mengautentikasi dengan mekanisme “SCRAM-SHA-512".
- KAFKA_SASL_SCRAM_PASSWORD- Kata sandi plaintext yang digunakan untuk mengautentikasi dengan mekanisme “SCRAM-SHA-512".
- ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD- Versi terenkripsi dari kata sandi Kafka SASL SCRAM (jika pengguna memiliki pengaturan kata sandi AWS Glue enkripsi yang dipilih).
- KAFKA_SASL_SCRAM_SECRETS_ARN- Nama Sumber Daya Amazon dari AWS rahasia di Secrets Manager.
- KAFKA_SASL_GSSAPI_KEYTAB- Lokasi S3 dari file Kerberos`keytab`. Keytab menyimpan kunci jangka panjang untuk satu atau lebih prinsipal. Untuk informasi selengkapnya, lihat Dokumentasi MIT Kerberos: Keytab.
- KAFKA_SASL_GSSAPI_KRB5_CONF- Lokasi S3 dari file Kerberos`krb5.conf`. Krb5.conf menyimpan informasi konfigurasi Kerberos, seperti lokasi server KDC. Untuk informasi lebih lanjut, lihat Dokumentasi MIT Kerberos: krb5.conf.

  - `KAFKA_SASL_GSSAPI_SERVICE`- Nama layanan Kerberos, seperti yang diatur `sasl.kerberos.service.name` dalam Konfigurasi Kafka Anda.
  - `KAFKA_SASL_GSSAPI_PRINCIPAL`- Nama princial Kerberos yang digunakan oleh. AWS Glue Untuk informasi lebih lanjut, lihat Dokumentasi Kafka: Mengkonfigurasi Broker Kafka.
- `PhysicalConnectionRequirements` — Sebuah objek PhysicalConnectionPersyaratan.

  Persyaratan koneksi fisik, seperti virtual private cloud (VPC) dan`SecurityGroup`, yang diperlukan untuk membuat koneksi ini berhasil.
- `CreationTime` — Stempel waktu.

  Stempel waktu waktu definisi koneksi ini dibuat.
- `LastUpdatedTime` — Stempel waktu.

  Stempel waktu terakhir kali definisi koneksi diperbarui.
- `LastUpdatedBy` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengguna, grup, atau peran yang terakhir memperbarui definisi koneksi ini.
- `Status` – String UTF-8 (nilai yang valid: `READY` | `IN_PROGRESS` | `FAILED`).

  Status koneksi. Bisa menjadi salah satu dari:`READY`,`IN_PROGRESS`, atau`FAILED`.
- `StatusReason`— String UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 16384 byte.

  Alasan status koneksi.
- `LastConnectionValidationTime` — Stempel waktu.

  Stempel waktu saat koneksi ini terakhir divalidasi.
- `AuthenticationConfiguration` — Sebuah objek AuthenticationConfiguration.

  Properti otentikasi koneksi.

ConnectionInput struktur

Sebuah struktur yang digunakan untuk menentukan koneksi yang akan dibuat atau diperbarui.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama koneksi.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi koneksi.

- `ConnectionType`— Diperlukan: UTF-8 string (nilai valid: `JDBC` | `SFTP` | | `MONGODB` | `KAFKA` | `NETWORK` `MARKETPLACE` `CUSTOM` |`SALESFORCE`).

  Jenis koneksi. Saat ini, jenis-jenis berikut ini didukung:

  - `JDBC` - Menunjuk koneksi ke database melalui Java Database Connectivity (JDBC).

    `JDBC`Koneksi menggunakan yang berikut ini ConnectionParameters.

    - Diperlukan: Semua (`HOST`,`PORT`,`JDBC_ENGINE`) atau`JDBC_CONNECTION_URL`.
    - Diperlukan: Semua (`USERNAME`,`PASSWORD`) atau`SECRET_ID`.
    - Opsional:`JDBC_ENFORCE_SSL`,`CUSTOM_JDBC_CERT`,`CUSTOM_JDBC_CERT_STRING`,`SKIP_CUSTOM_` Parameter ini digunakan untuk mengkonfigurasi SSL dengan JDBC.

  - `KAFKA` - Menunjuk koneksi ke platform streaming Apache Kafka.

    `KAFKA`Koneksi menggunakan yang berikut ini ConnectionParameters.

    - Diperlukan:`KAFKA_BOOTSTRAP_SERVERS`.
    - Opsional:`KAFKA_SSL_ENABLED`,`KAFKA_CUSTOM_CERT`,`KAFKA_SKIP_CUSTOM_CERT_VALIDATION` Parameter ini digunakan untuk mengkonfigurasi SSL dengan`KAFKA`.
    - Opsional:`KAFKA_CLIENT_KEYSTORE`,`KAFKA_CLIENT_KEYSTORE_PASSWORD`,`KAFKA_CLIENT_KEY` Parameter ini digunakan untuk mengkonfigurasi konfigurasi klien TLS dengan SSL di. `KAFKA`
    - Opsional:`KAFKA_SASL_MECHANISM`. Dapat ditentukan sebagai`SCRAM-SHA-512`,`GSSAPI`, atau`AWS_MSK_IAM`.
    - Opsional:`KAFKA_SASL_SCRAM_USERNAME`,`KAFKA_SASL_SCRAM_PASSWORD`,`ENCRYPTED_KAFKA_S` Parameter ini digunakan untuk mengkonfigurasi otentikasi SASL/SCRAM-SHA-512 dengan. `KAFKA`
    - Opsional:`KAFKA_SASL_GSSAPI_KEYTAB`,`KAFKA_SASL_GSSAPI_KRB5_CONF`,`KAFKA_SASL_GSSAP` Parameter ini digunakan untuk mengkonfigurasi otentikasi SASL/GSSAPI dengan. `KAFKA`

- `MONGODB` - Menunjuk koneksi ke database dokumen MongoDB.

  `MONGODB`Koneksi menggunakan yang berikut ini ConnectionParameters.

  - Diperlukan:`CONNECTION_URL`.
  - Diperlukan: Semua (`USERNAME`,`PASSWORD`) atau`SECRET_ID`.
- `SALESFORCE`- Menentukan koneksi ke Salesforce menggunakan authencation OAuth.
  - Membutuhkan `AuthenticationConfiguration` anggota untuk dikonfigurasi.
- `NETWORK` - Menunjuk koneksi jaringan ke sumber data dalam lingkungan Amazon Virtual Private Cloud (Amazon VPC).

  `NETWORK`Koneksi tidak memerlukan ConnectionParameters. Sebaliknya, berikan a PhysicalConnectionRequirements.
- `MARKETPLACE`- Menggunakan pengaturan konfigurasi yang terdapat dalam konektor yang dibeli AWS Marketplace untuk membaca dan menulis ke penyimpanan data yang tidak didukung secara asli oleh AWS Glue.

  `MARKETPLACE`Koneksi menggunakan yang berikut ini ConnectionParameters.
  - Diperlukan:`CONNECTOR_TYPE`,`CONNECTOR_URL`,`CONNECTOR_CLASS_NAME`,`CONNECTION_URL`.
  - Diperlukan untuk `JDBC` `CONNECTOR_TYPE` koneksi: Semua (`USERNAME`,`PASSWORD`) atau`SECRET_ID`.
- `CUSTOM` - Menggunakan pengaturan konfigurasi yang terkandung dalam sebuah konektor kustom untuk membaca dari dan menulis ke penyimpanan data yang tidak didukung secara asli oleh AWS Glue.

  `SFTP` tidak didukung.

  Untuk informasi selengkapnya tentang cara opsional ConnectionProperties digunakan untuk mengonfigurasi fitur AWS Glue, lihat properti AWS Glue koneksi.

  Untuk informasi selengkapnya tentang cara opsional ConnectionProperties digunakan untuk mengonfigurasi fitur di AWS Glue Studio, lihat Menggunakan konektor dan koneksi.
- `MatchCriteria` — Susunan string UTF-8, tidak lebih dari 10 string.

  Daftar kriteria yang dapat digunakan dalam memilih koneksi ini.
- `ConnectionProperties` — Wajib: Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 100 pasang.

Setiap kunci adalah string UTF-8 (nilai yang valid: HOST | PORT | USERNAME="USER_NAME" | PASSWORD | ENCRYPTED_PASSWORD | JDBC_DRIVER_JAR_URI | JDBC_DRIVER_CLASS_NAME | JDBC_ENGINE | JDBC_ENGINE_VERSION | CONFIG_FILES INSTANCE_ID | JDBC_CONNECTION_URL | JDBC_ENFORCE_SSL | CUSTOM_JDBC_CERT | SKIP_CUSTOM_JDBC_CERT_VALIDATION | CUSTOM_JDBC_CERT_STRING | CONNECTION_URL | KAFKA_BOOTSTRAP_SERVERS | KAFKA_SSL_ENABLED | KAFKA_CUSTOM_CERT | KAFKA_SKIP_CUSTOM_CERT_VALIDATION | KAFKA_CLIENT_KEYSTORE | KAFKA_CLIENT_KEYSTORE_PASSWORD | KAFKA_CLIENT_KEY_PASSWORD | ENCRYPTED_KAFKA_CLIENT_KEYSTORE_PASSWORD | ENCRYPTED_KAFKA_CLIENT_KEY_PASSWORD | SECRET_ID | CONNECTOR_URL | CONNECTOR_TYPE | CONNECTOR_CLASS_NAME | KAFKA_SASL_MECHANISM | KAFKA_SASL_PLAIN_USERNAME | KAFKA_SASL_PLAIN_PASSWORD | ENCRYPTED_KAFKA_SASL_PLAIN_PASSWORD KAFKA_SASL_SCRAM_USERNAME KAFKA_SASL_SCRAM_PASSWORD | KAFKA_SASL_SCRAM_SECRETS_ARN | ENCRYPTED_KAFKA_SASL_SCRAM_PASSWORD | KAFKA_SASL_GSSAPI_KEYTAB KAFKA_SASL_GSSAPI_KRB5_CONF | KAFKA_SASL_GSSAPI_SERVICE | KAFKA_SASL_GSSAPI_PRINCIPAL |ROLE_ARN).

Masing-masing kunci adalah sebuah string Nilai, dengan panjang tidak lebih dari 1024 byte.

Pasangan nilai-kunci ini menentukan parameter untuk koneksi.

- `PhysicalConnectionRequirements` — Sebuah objek PhysicalConnectionPersyaratan.

  Persyaratan koneksi fisik, seperti virtual private cloud (VPC) dan`SecurityGroup`, yang diperlukan untuk berhasil membuat koneksi ini.

- `AuthenticationConfiguration` — Sebuah objek AuthenticationConfigurationMasukan.

  Properti otentikasi koneksi. Digunakan untuk koneksi Salesforce.

- `ValidateCredentials` – Boolean.

  Bendera untuk memvalidasi kredensyal selama membuat koneksi. Digunakan untuk koneksi Salesforce. Default benar.

PhysicalConnectionRequirements struktur

Aplikasi klien OAuth sebagai tanggapan GetConnection .

Bidang

- `SubnetId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID subnet yang digunakan oleh koneksi.
- `SecurityGroupIdList` — Susunan string UTF-8, tidak lebih dari 50 string.

  Daftar ID grup keamanan yang digunakan oleh koneksi.
- `AvailabilityZone` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Availability Zone koneksi.

GetConnectionsFilter struktur

Mem-filter definisi koneksi yang dikembalikan oleh operasi API `GetConnections`.

Bidang

- `MatchCriteria` — Susunan string UTF-8, tidak lebih dari 10 string.

  Sebuah kriteria string yang harus sesuai dengan kriteria yang dicatat dalam definisi koneksi untuk definisi koneksi yang akan dikembalikan.
- `ConnectionType` – String UTF-8 (nilai yang valid: `JDBC` | `SFTP` | `MONGODB` | `KAFKA` | `NETWORK` | `MARKETPLACE` | `CUSTOM` | `SALESFORCE`).

  Jenis koneksi yang akan dikembalikan. Saat ini, SFTP tidak didukung.

Operasi

- CreateConnection tindakan (Python: create_connection)
- DeleteConnection tindakan (Python: delete_connection)
- GetConnection tindakan (Python: get_connection)
- GetConnections tindakan (Python: get_connections)
- UpdateConnection tindakan (Python: update_connection)
- BatchDeleteConnection tindakan (Python: batch_delete_connection)

CreateConnection tindakan (Python: create_connection)

Menciptakan sebuah definisi koneksi baru dalam Katalog Data.

Koneksi yang digunakan untuk membuat sumber daya federasi memerlukan izin IAM`glue:PassConnection`.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat koneksi akan dibuat. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `ConnectionInput` — Wajib: Sebuah objek ConnectionInput.

  Sebuah objek `ConnectionInput` yang mendefinisikan koneksi yang akan dibuat.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Tag yang Anda tetapkan ke koneksi.

Respons

- `CreateConnectionStatus` – String UTF-8 (nilai yang valid: `READY` | `IN_PROGRESS` | `FAILED`).

  Status permintaan pembuatan koneksi. Permintaan dapat memakan waktu untuk jenis otentikasi tertentu, misalnya saat membuat koneksi OAuth dengan pertukaran token melalui VPC.

Kesalahan

- `AlreadyExistsException`
- `InvalidInputException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`

- `GlueEncryptionException`

DeleteConnection tindakan (Python: delete_connection)

Menghapus sebuah koneksi dari Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat koneksi berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `ConnectionName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama koneksi yang akan dihapus.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `OperationTimeoutException`

GetConnection tindakan (Python: get_connection)

Mengambil sebuah definisi koneksi dari Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat koneksi berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi koneksi yang akan diambil.

- `HidePassword` – Boolean.

  Memungkinkan Anda untuk mengambil metadata koneksi tanpa perlu mengembalikan kata sandi. Misalnya, AWS Glue konsol menggunakan bendera ini untuk mengambil koneksi, dan tidak menampilkan kata sandi. Tetapkan parameter ini ketika pemanggil mungkin tidak memiliki izin untuk menggunakan AWS KMS kunci untuk mendekripsi kata sandi, tetapi ia memiliki izin untuk mengakses properti koneksi lainnya.

Respons

- `Connection` — Sebuah objek Koneksi.

  Definisi koneksi yang diminta.

Kesalahan

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

GetConnections tindakan (Python: get_connections)

Mengambil sebuah daftar definisi koneksi dari Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat koneksi berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.

- `Filter` — Sebuah objek GetConnectionsFilter.

Filter yang mengontrol koneksi mana yang dikembalikan.

- `HidePassword` – Boolean.

  Memungkinkan Anda untuk mengambil metadata koneksi tanpa perlu mengembalikan kata sandi. Misalnya, AWS Glue konsol menggunakan bendera ini untuk mengambil koneksi, dan tidak menampilkan kata sandi. Tetapkan parameter ini ketika pemanggil mungkin tidak memiliki izin untuk menggunakan AWS KMS kunci untuk mendekripsi kata sandi, tetapi ia memiliki izin untuk mengakses properti koneksi lainnya.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah maksimum koneksi yang akan dikembalikan dalam satu respons.

Respons

- `ConnectionList` – Susunan objek Koneksi.

  Daftar definisi koneksi yang diminta.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika daftar koneksi yang dikembalikan tidak mencakup koneksi disaring yang terakhir.

Kesalahan

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

UpdateConnection tindakan (Python: update_connection)

Memperbarui sebuah definisi koneksi baru dalam Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat koneksi berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.
- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi koneksi yang akan diperbarui.
- `ConnectionInput` — Wajib: Sebuah objek ConnectionInput.

  Sebuah objek `ConnectionInput` yang mengubah koneksi yang dimaksud.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`
- `GlueEncryptionException`

BatchDeleteConnection tindakan (Python: batch_delete_connection)

Menghapus sebuah daftar definisi koneksi dari Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat koneksi berada. Jika tidak ada yang disediakan, ID AWS akun digunakan secara default.

- `ConnectionNameList` — Wajib: Susunan string UTF-8, tidak lebih dari 25 string.

  Daftar nama koneksi yang akan dihapus.

Respons

- `Succeeded` – Susunan string UTF-8.

  Daftar nama definisi koneksi yang sudah berhasil dihapus.
- `Errors` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah sebuah objek ErrorDetail.

  Peta nama koneksi yang tidak berhasil dihapus ke detail kesalahan.

Kesalahan

- `InternalServiceException`
- `OperationTimeoutException`

Konfigurasi otentikasi

- AuthenticationConfiguration struktur
- AuthenticationConfigurationInput struktur
- struktur OAuth2properties
- Struktur OAuth2 PropertiesInput
- Struktur OAuth2 ClientApplication
- AuthorizationCodeProperties struktur

AuthenticationConfiguration struktur

Struktur yang berisi konfigurasi otentikasi.

Bidang

- `AuthenticationType` – String UTF-8 (nilai yang valid: `BASIC` | `OAUTH2` | `CUSTOM`).

  Struktur yang berisi konfigurasi otentikasi.
- `SecretArn` — String UTF-8, yang cocok dengan Custom string pattern #11.

  Manajer rahasia ARN untuk menyimpan kredensyal.
- `OAuth2Properties` — Sebuah objek OAuth2properti.

  Properti untuk otentikasi OAuth2.

AuthenticationConfigurationInput struktur

Struktur yang berisi konfigurasi otentikasi dalam CreateConnection permintaan.

Bidang

- `AuthenticationType` – String UTF-8 (nilai yang valid: `BASIC` | `OAUTH2` | `CUSTOM`).

  Struktur yang berisi konfigurasi otentikasi dalam CreateConnection permintaan.
- `SecretArn` — String UTF-8, yang cocok dengan Custom string pattern #11.

  Manajer rahasia ARN untuk menyimpan kredensyal dalam permintaan. CreateConnection
- `OAuth2Properties` — Sebuah objek OAuth2 PropertiesInput.

  Properti untuk otentikasi OAuth2 dalam permintaan. CreateConnection

struktur OAuth2properties

Struktur yang berisi properti untuk otentikasi OAuth2.

Bidang

- `OAuth2GrantType` – String UTF-8 (nilai yang valid: `AUTHORIZATION_CODE` | `CLIENT_CREDENTIALS` | `JWT_BEARER`).

  Jenis hibah OAuth2. Sebagai contoh, `AUTHORIZATION_CODE`, `JWT_BEARER`, atau `CLIENT_CREDENTIALS`.

- `OAuth2ClientApplication` — Sebuah objek OAuth2 ClientApplication.

  Jenis aplikasi klien. Misalnya, AWS_MANAGED atau USER_MANAGED.

- `TokenUrl`— String UTF-8, panjangnya tidak lebih dari 256 byte, cocok dengan file. Custom string pattern #12

  URL server otentikasi penyedia, untuk menukar kode otorisasi untuk token akses.

- `TokenUrlParametersMap` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Setiap nilai adalah string UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 512 byte.

  Peta parameter yang ditambahkan ke `GET` permintaan token.

Struktur OAuth2 PropertiesInput

Struktur yang berisi properti untuk OAuth2 dalam permintaan. CreateConnection

Bidang

- `OAuth2GrantType` – String UTF-8 (nilai yang valid: `AUTHORIZATION_CODE` | `CLIENT_CREDENTIALS` | `JWT_BEARER`).

  Jenis hibah OAuth2 dalam permintaan. CreateConnection Sebagai contoh, `AUTHORIZATION_CODE`, `JWT_BEARER`, atau `CLIENT_CREDENTIALS`.

- `OAuth2ClientApplication` — Sebuah objek OAuth2 ClientApplication.

  Jenis aplikasi klien dalam CreateConnection permintaan. Misalnya, `AWS_MANAGED` atau `USER_MANAGED`.

- `TokenUrl`— String UTF-8, panjangnya tidak lebih dari 256 byte, cocok dengan file. Custom string pattern #12

  URL server otentikasi penyedia, untuk menukar kode otorisasi untuk token akses.

- `TokenUrlParametersMap` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

Setiap nilai adalah string UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 512 byte.

Peta parameter yang ditambahkan ke `GET` permintaan token.

- `AuthorizationCodeProperties` — Sebuah objek AuthorizationCodeProperti.

  Kumpulan properti yang diperlukan untuk jenis `AUTHORIZATION_CODE` hibah OAuth2.

Struktur OAuth2 ClientApplication

Aplikasi klien OAuth2 yang digunakan untuk koneksi.

Bidang

- `UserManagedClientApplicationClientId` — String UTF-8, sepanjang tidak lebih dari 2048, yang cocok dengan Custom string pattern #13.

  Aplikasi klien ClientID jika ada. ClientAppType `USER_MANAGED`
- `AWSManagedClientApplicationReference` — String UTF-8, sepanjang tidak lebih dari 2048, yang cocok dengan Custom string pattern #13.

  Referensi ke aplikasi klien sisi SaaS yang dikelola AWS .

AuthorizationCodeProperties struktur

Kumpulan properti yang diperlukan untuk alur kerja jenis `AUTHORIZATION_CODE` hibah OAuth2.

Bidang

- `AuthorizationCode`— String UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 4096 byte, cocok dengan file. Custom string pattern #13

  Kode otorisasi yang akan digunakan di bagian ketiga alur kerja `AUTHORIZATION_CODE` hibah. Ini adalah kode sekali pakai yang menjadi tidak valid setelah ditukar dengan token akses, sehingga dapat diterima untuk memiliki nilai ini sebagai parameter permintaan.
- `RedirectUri`— String UTF-8, panjangnya tidak lebih dari 512 byte, cocok dengan. Custom string pattern #14

  URI pengalihan tempat pengguna diarahkan oleh server otorisasi saat mengeluarkan kode otorisasi. URI selanjutnya digunakan ketika kode otorisasi ditukar dengan token akses.

API Fungsi yang ditentukan pengguna

User-defined Function API menjelaskan tipe AWS Glue data dan operasi yang digunakan dalam bekerja dengan fungsi.

Jenis Data

- UserDefinedFunction struktur
- UserDefinedFunctionInput struktur

UserDefinedFunction struktur

Merepresentasikan definisi fungsi yang ditetapkan pengguna Hive (UDF) yang setara.

Bidang

- `FunctionName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama fungsi.
- `DatabaseName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog yang berisi fungsi.
- `ClassName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Kelas Java yang berisi kode fungsi.
- `OwnerName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pemilik fungsi.
- `OwnerType` – String UTF-8 (nilai yang valid: `USER` | `ROLE` | `GROUP`).

  Jenis pemilik.
- `CreateTime` — Stempel waktu.

  Waktu saat fungsi dibuat.

- `ResourceUris` — Susunan objek ResourceUri, tidak lebih dari 1000 struktur.

  URI sumber daya untuk fungsi.
- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat fungsi berada.

UserDefinedFunctionInput struktur

Struktur yang digunakan untuk membuat atau memperbarui sebuah fungsi yang ditetapkan pengguna.

Bidang

- `FunctionName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama fungsi.
- `ClassName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Kelas Java yang berisi kode fungsi.
- `OwnerName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pemilik fungsi.
- `OwnerType` – String UTF-8 (nilai yang valid: `USER` | `ROLE` | `GROUP`).

  Jenis pemilik.
- `ResourceUris` — Susunan objek ResourceUri, tidak lebih dari 1000 struktur.

  URI sumber daya untuk fungsi.

Operasi

- CreateUserDefinedFunction tindakan (Python: create_user_defined_function)
- UpdateUserDefinedFunction tindakan (Python: update_user_defined_function)

- DeleteUserDefinedFunction tindakan (Python: delete_user_defined_function)
- GetUserDefinedFunction tindakan (Python: get_user_defined_function)
- GetUserDefinedFunctions tindakan (Python: get_user_defined_functions)

CreateUserDefinedFunction tindakan (Python: create_user_defined_function)

Menciptakan sebuah definisi fungsi baru dalam Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data tempat fungsi dibuat. Jika tidak ada yang disediakan, ID akun AWS digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat fungsi dibuat.
- `FunctionInput` — Wajib: Sebuah objek UserDefinedFunctionInput.

  Sebuah objek `FunctionInput` yang mendefinisikan fungsi yang akan dibuat dalam Katalog Data.

Response

- Tidak ada parameter Respons.

Kesalahan

- `AlreadyExistsException`
- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `GlueEncryptionException`

UpdateUserDefinedFunction tindakan (Python: update_user_defined_function)

Memperbarui definisi fungsi yang ada dalam Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog Data di mana fungsi yang akan diperbarui berada. Jika tidak ada yang disediakan, ID akun AWS digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog di mana fungsi yang akan diperbarui berada.
- `FunctionName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama fungsi.
- `FunctionInput` — Wajib: Sebuah objek UserDefinedFunctionInput.

  Sebuah objek `FunctionInput` yang mendefinisikan kembali fungsi yang ada dalam Katalog Data.

Response

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

DeleteUserDefinedFunction tindakan (Python: delete_user_defined_function)

Menghapus sebuah definisi fungsi yang ada dari Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data di mana fungsi yang akan dihapus berada. Jika tidak ada yang disediakan, ID akun AWS digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat fungsi berada.
- `FunctionName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi fungsi yang akan dihapus.

Response

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetUserDefinedFunction tindakan (Python: get_user_defined_function)

Mengambil definisi fungsi yang ditentukan dari Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data di mana fungsi yang akan diambil berada. Jika tidak ada yang disediakan, ID akun AWS digunakan secara default.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog tempat fungsi berada.
- `FunctionName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama fungsi.

Response

- `UserDefinedFunction` — Sebuah objek UserDefinedFunction.

  Definisi fungsi yang diminta.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `GlueEncryptionException`

GetUserDefinedFunctions tindakan (Python: get_user_defined_functions)

Mengambil beberapa definisi fungsi dari Katalog Data.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

ID dari Katalog Data di mana fungsi yang akan diambil berada. Jika tidak ada yang disediakan, ID akun AWS digunakan secara default.

- `DatabaseName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data katalog di mana fungsi berada. Jika tidak ada yang disediakan, maka fungsi dari semua basis data di katalog akan dikembalikan.

- `Pattern` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  String pola fungsi-nama opsional yang menyaring definisi fungsi yang dikembalikan.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100.

  Jumlah maksimum fungsi yang akan dikembalikan dalam satu respons.

Response

- `UserDefinedFunctions` – Susunan objek UserDefinedFunction.

  Daftar definisi fungsi yang diminta.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika daftar fungsi yang dikembalikan tidak termasuk fungsi yang diminta terakhir.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `GlueEncryptionException`

Mengimpor Athena katalog ke AWS Glue

Migration API menjelaskan tipe AWS Glue data dan operasi yang berkaitan dengan migrasi katalog Athena Data keAWS Glue.

Jenis Data

- CatalogImportStatus struktur

CatalogImportStatus struktur

Struktur yang berisi informasi status migrasi.

Bidang

- `ImportCompleted` – Boolean.

  `True` jika migrasi telah selesai, atau `False` jika sebaliknya.
- `ImportTime` — Stempel waktu.

  Waktu ketika migrasi dimulai.
- `ImportedBy` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama orang yang memulai migrasi.

Operasi

- ImportCatalogToGlue tindakan (Python: import_catalog_to_glue)
- GetCatalogImportStatus tindakan (Python: get_catalog_import_status)

ImportCatalogToGlue tindakan (Python: import_catalog_to_glue)

Mengimpor Katalog Data Amazon Athena yang ada ke AWS Glue.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

ID katalog yang akan diimpor. Saat ini, ini adalah ID akun AWS.

Response

- Tidak ada parameter Respons.

Kesalahan

- `InternalServiceException`
- `OperationTimeoutException`

GetCatalogImportStatus tindakan (Python: get_catalog_import_status)

Mengambil status dari sebuah operasi migrasi.

Permintaan

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID katalog yang akan dimigrasi. Saat ini, ini adalah ID akun AWS.

Response

- `ImportStatus` — Sebuah objek CatalogImportStatus.

  Status dari migrasi katalog yang ditentukan.

Kesalahan

- `InternalServiceException`
- `OperationTimeoutException`

API pengoptimal tabel

API pengoptimal tabel menjelaskan AWS Glue API untuk mengaktifkan pemadatan guna meningkatkan kinerja baca.

Jenis data

- TableOptimizer struktur
- TableOptimizerConfiguration struktur
- TableOptimizerRun struktur
- RunMetrics struktur
- BatchGetTableOptimizerEntry struktur
- BatchTableOptimizer struktur
- BatchGetTableOptimizerError struktur

TableOptimizer struktur

Berisi rincian tentang pengoptimal yang terkait dengan tabel.

Bidang

- `type` – String UTF-8 (nilai yang valid: `compaction="COMPACTION"`).

  Jenis pengoptimal tabel. Saat ini, satu-satunya nilai yang valid adalah`compaction`.
- `configuration` — Sebuah objek TableOptimizerConfiguration.

  `TableOptimizerConfiguration`Objek yang ditentukan saat membuat atau memperbarui pengoptimal tabel.
- `lastRun` — Sebuah objek TableOptimizerRun.

  Sebuah `TableOptimizerRun` objek yang mewakili run terakhir dari pengoptimal tabel.

TableOptimizerConfiguration struktur

Berisi rincian tentang konfigurasi pengoptimal tabel. Anda meneruskan konfigurasi ini saat membuat atau memperbarui pengoptimal tabel.

Bidang

- `roleArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 512 byte, yang cocok dengan Single-line string pattern.

  Peran yang diteruskan oleh pemanggil yang memberikan izin layanan untuk memperbarui sumber daya yang terkait dengan pengoptimal atas nama pemanggil.
- `enabled` – Boolean.

  Apakah optimasi tabel diaktifkan.

TableOptimizerRun struktur

Berisi detail untuk menjalankan pengoptimal tabel.

Bidang

- `eventType` – String UTF-8 (nilai yang valid: `starting="STARTING"` | `completed="COMPLETED"` | `failed="FAILED"` | `in_progress="IN_PROGRESS"`).

  Jenis peristiwa yang mewakili status menjalankan pengoptimal tabel.
- `startTimestamp` — Stempel waktu.

  Merupakan stempel waktu zaman di mana pekerjaan pemadatan dimulai dalam Lake Formation.
- `endTimestamp` — Stempel waktu.

  Merupakan stempel waktu zaman di mana pekerjaan pemadatan berakhir.
- `metrics` — Sebuah objek RunMetrics.

  `RunMetrics`Objek yang berisi metrik untuk menjalankan pengoptimal.
- `error` – String UTF-8.

  Kesalahan yang terjadi selama pengoptimal dijalankan.

RunMetrics struktur

Metrik untuk menjalankan pengoptimal.

Bidang

- `NumberOfBytesCompacted` – String UTF-8.

  Jumlah byte yang dihapus oleh pekerjaan pemadatan dijalankan.
- `NumberOfFilesCompacted` – String UTF-8.

  Jumlah file yang dihapus oleh pekerjaan pemadatan dijalankan.
- `NumberOfDpus` – String UTF-8.

  Jumlah jam DPU yang dikonsumsi oleh pekerjaan.
- `JobDurationInHour` – String UTF-8.

  Durasi pekerjaan dalam jam.

BatchGetTableOptimizerEntry struktur

Merupakan pengoptimal tabel untuk mengambil dalam operasi. `BatchGetTableOptimizer`

Bidang

- `catalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog tabel.
- `databaseName`— UTF-8 string, setidaknya 1 byte panjang.

  Nama basis data dalam katalog tempat tabel berada.
- `tableName`— String UTF-8, setidaknya 1 byte panjang.

  Nama tabel.
- `type` – String UTF-8 (nilai yang valid: `compaction="COMPACTION"`).

  Jenis pengoptimal tabel.

BatchTableOptimizer struktur

Berisi detail untuk salah satu pengoptimal tabel yang dikembalikan oleh operasi. `BatchGetTableOptimizer`

Bidang

- `catalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog tabel.
- `databaseName`— String UTF-8, setidaknya 1 byte panjang.

  Nama basis data dalam katalog tempat tabel berada.
- `tableName`— String UTF-8, setidaknya 1 byte panjang.

  Nama tabel.
- `tableOptimizer` — Sebuah objek TableOptimizer.

  `TableOptimizer`Objek yang berisi detail tentang konfigurasi dan proses terakhir dari pengoptimal tabel.

BatchGetTableOptimizerError struktur

Berisi rincian tentang salah satu kesalahan dalam daftar kesalahan yang dikembalikan oleh `BatchGetTableOptimizer` operasi.

Bidang

- `error` — Sebuah objek ErrorDetail.

  `ErrorDetail`Objek yang berisi kode dan rincian pesan tentang kesalahan.
- `catalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog tabel.
- `databaseName`— String UTF-8, setidaknya 1 byte panjang.

  Nama basis data dalam katalog tempat tabel berada.
- `tableName`— String UTF-8, setidaknya 1 byte panjang.

  Nama tabel.
- `type` – String UTF-8 (nilai yang valid: `compaction="COMPACTION"`).

Jenis pengoptimal tabel.

Operasi

- GetTableOptimizer tindakan (Python: get_table_optimizer)
- BatchGetTableOptimizer tindakan (Python: batch_get_table_optimizer)
- ListTableOptimizerRuns tindakan (Python: list_table_optimizer_runs)
- CreateTableOptimizer tindakan (Python: create_table_optimizer)
- DeleteTableOptimizer tindakan (Python: delete_table_optimizer)
- UpdateTableOptimizer tindakan (Python: update_table_optimizer)

GetTableOptimizer tindakan (Python: get_table_optimizer)

Mengembalikan konfigurasi semua pengoptimal yang terkait dengan tabel tertentu.

Permintaan

- `CatalogId`— Wajib: String id katalog, panjangnya tidak kurang dari 1 atau lebih dari 255 byte, cocok dengan file. Single-line string pattern

  ID Katalog tabel.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data dalam katalog tempat tabel berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel.
- `Type` – Wajib: String UTF-8 (nilai yang valid: `compaction="COMPACTION"`).

  Jenis pengoptimal tabel.

Respons

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog tabel.

- `DatabaseName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data dalam katalog tempat tabel berada.

- `TableName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel.

- `TableOptimizer` — Sebuah objek TableOptimizer.

  Pengoptimal terkait dengan tabel yang ditentukan.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `AccessDeniedException`
- `InternalServiceException`

BatchGetTableOptimizer tindakan (Python: batch_get_table_optimizer)

Mengembalikan konfigurasi untuk pengoptimal tabel tertentu.

Permintaan

- `Entries` – Wajib: Susunan objek BatchGetTableOptimizerEntry.

  Daftar `BatchGetTableOptimizerEntry` objek yang menentukan pengoptimal tabel untuk mengambil.

Respons

- `TableOptimizers` – Susunan objek BatchTableOptimizer.

  Daftar objek `BatchTableOptimizer`.
- `Failures` – Susunan objek BatchGetTableOptimizerError.

  Daftar kesalahan dari operasi.

Kesalahan

- `InternalServiceException`

ListTableOptimizerRuns tindakan (Python: list_table_optimizer_runs)

Daftar riwayat pengoptimal sebelumnya berjalan untuk tabel tertentu.

Permintaan

- `CatalogId`— Wajib: String id katalog, panjangnya tidak kurang dari 1 atau lebih dari 255 byte, cocok dengan file. Single-line string pattern

  ID Katalog tabel.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data dalam katalog tempat tabel berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel.
- `Type` – Wajib: String UTF-8 (nilai yang valid: `compaction="COMPACTION"`).

  Jenis pengoptimal tabel. Saat ini, satu-satunya nilai yang valid adalah`compaction`.
- `MaxResults` — Nomor (bilangan bulat).

  Jumlah maksimum pengoptimal berjalan untuk kembali pada setiap panggilan.
- `NextToken` – String UTF-8.

Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Respons

- `CatalogId` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Katalog tabel.

- `DatabaseName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data dalam katalog tempat tabel berada.

- `TableName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel.

- `NextToken` – String UTF-8.

  Token kelanjutan untuk paginasi daftar pengoptimal yang dikembalikan berjalan, dikembalikan jika segmen daftar saat ini bukan yang terakhir.

- `TableOptimizerRuns` – Susunan objek TableOptimizerRun.

  Daftar pengoptimal berjalan terkait dengan tabel.

Kesalahan

- `EntityNotFoundException`
- `AccessDeniedException`
- `InvalidInputException`
- `InternalServiceException`

CreateTableOptimizer tindakan (Python: create_table_optimizer)

Membuat pengoptimal tabel baru untuk fungsi tertentu. `compaction`adalah satu-satunya jenis pengoptimal yang didukung saat ini.

Permintaan

- `CatalogId`— Wajib: String id katalog, panjangnya tidak kurang dari 1 atau lebih dari 255 byte, cocok dengan file. Single-line string pattern

  ID Katalog tabel.
- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data dalam katalog tempat tabel berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel.
- `Type` – Wajib: String UTF-8 (nilai yang valid: `compaction="COMPACTION"`).

  Jenis pengoptimal tabel. Saat ini, satu-satunya nilai yang valid adalah`compaction`.
- `TableOptimizerConfiguration` — Wajib: Sebuah objek TableOptimizerConfiguration.

  Sebuah `TableOptimizerConfiguration` objek yang mewakili konfigurasi pengoptimal tabel.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `AccessDeniedException`
- `AlreadyExistsException`
- `InternalServiceException`

DeleteTableOptimizer tindakan (Python: delete_table_optimizer)

Menghapus pengoptimal dan semua metadata terkait untuk tabel. Optimalisasi tidak akan lagi dilakukan di atas meja.

Permintaan

- `CatalogId`— Wajib: String id katalog, panjangnya tidak kurang dari 1 atau lebih dari 255 byte, cocok dengan file. Single-line string pattern

  ID Katalog tabel.

- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data dalam katalog tempat tabel berada.

- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel.

- `Type` – Wajib: String UTF-8 (nilai yang valid: `compaction="COMPACTION"`).

  Jenis pengoptimal tabel.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `AccessDeniedException`
- `InternalServiceException`

UpdateTableOptimizer tindakan (Python: update_table_optimizer)

Memperbarui konfigurasi untuk pengoptimal tabel yang ada.

Permintaan

- `CatalogId`— Wajib: String id katalog, panjangnya tidak kurang dari 1 atau lebih dari 255 byte, cocok dengan file. Single-line string pattern

ID Katalog tabel.

- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data dalam katalog tempat tabel berada.

- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel.

- `Type` – Wajib: String UTF-8 (nilai yang valid: `compaction="COMPACTION"`).

  Jenis pengoptimal tabel. Saat ini, satu-satunya nilai yang valid adalah`compaction`.

- `TableOptimizerConfiguration` — Wajib: Sebuah objek TableOptimizerConfiguration.

  Sebuah `TableOptimizerConfiguration` objek yang mewakili konfigurasi pengoptimal tabel.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `AccessDeniedException`
- `InternalServiceException`

API Crawler dan Pengklasifikasi

API Crawler dan pengklasifikasi menjelaskan tentang jenis data AWS Glue crawler dan pengklasifikasi, dan termasuk API untuk membuat, menghapus, memperbarui, dan mencantumkan crawler atau pengklasifikasi.

Topik

- API pengklasifikasi

- API Crawler
- API statistik kolom
- API penjadwal perayap

API pengklasifikasi

API Pengklasifikasi menjelaskan tipe data AWS Glue pengklasifikasi, dan menyertakan API untuk membuat, menghapus, memperbarui, dan mencantumkan pengklasifikasi.

Jenis Data

- Struktur pengklasifikasi
- GrokClassifier struktur
- Struktur XMLClassifier
- JsonClassifier struktur
- CsvClassifier struktur
- CreateGrokClassifierRequest struktur
- UpdateGrokClassifierRequest struktur
- struktur ClassifierRequest CreateXML
- struktur ClassifierRequest UpdateXML
- CreateJsonClassifierRequest struktur
- UpdateJsonClassifierRequest struktur
- CreateCsvClassifierRequest struktur
- UpdateCsvClassifierRequest struktur

Struktur pengklasifikasi

Pengklasifikasi dipicu selama tugas melakukan perayapan. Sebuah pengklasifikasi memeriksa apakah file yang diberikan dalam format yang dapat ditanganinya. Jika ya, maka pengklasifikasi menciptakan sebuah skema dalam bentuk objek `StructType` yang cocok dengan format data tersebut.

Anda dapat menggunakan pengklasifikasi standar yang disediakan oleh AWS Glue, atau Anda dapat menulis pengklasifikasi Anda sendiri untuk mengkategorikan sumber data Anda dengan sebaik-baiknya dan menentukan skema yang sesuai untuk digunakan untuk mereka. Sebuah pengklasifikasi

dapat berupa pengklasifikasi `grok`, pengklasifikasi `XML`, pengklasifikasi `JSON`, atau pengklasifikasi `CSV` kustom, sebagaimana ditentukan dalam salah satu bidang di objek `Classifier`.

Bidang

- `GrokClassifier` — Sebuah objek GrokClassifier.

  Sebuah pengklasifikasi yang menggunakan `grok`.
- `XMLClassifier` — Sebuah objek XMLClassifier.

  Sebuah pengklasifikasi untuk konten XML.
- `JsonClassifier` — Sebuah objek JsonClassifier.

  Sebuah pengklasifikasi untuk konten JSON.
- `CsvClassifier` — Sebuah objek CsvClassifier.

  Pengklasifikasi untuk nilai yang dipisahkan koma (CSV).

GrokClassifier struktur

Pengklasifikasi yang menggunakan pola `grok`.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pengklasifikasi.
- `Classification` – Wajib: String UTF-8.

  Pengenal format data yang cocok dengan pengklasifikasi, seperti Twitter, JSON, log Omniture, dan sebagainya.
- `CreationTime` — Stempel waktu.

  Waktu pada daat pengklasifikasi ini didaftarkan.
- `LastUpdated` — Stempel waktu.

  Waktu pada saat pengklasifikasi ini terakhir diperbarui.
- `Version` — Nomor (panjang).

Versi dari pengklasifikasi ini.

- `GrokPattern` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 2048 byte, yang cocok dengan A Logstash Grok string pattern.

  Pola grok diterapkan ke penyimpanan data oleh pengklasifikasi ini. Untuk informasi selengkapnya, lihat pola bawaan dalam Menulis Pengklasifikasi Kustom.
- `CustomPatterns` — String UTF-8, sepanjang tidak lebih dari 16000, yang cocok dengan URI address multi-line string pattern.

  Pola grok kustom opsional ditentukan oleh pengklasifikasi ini. Untuk informasi selengkapnya, lihat pola kustom dalam Menulis Pengklasifikasi Kustom.

Struktur XMLClassifier

Sebuah pengklasifikasi untuk konten `XML`.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pengklasifikasi.
- `Classification` – Wajib: String UTF-8.

  Sebuah pengenal format data yang cocok dengan pengklasifikasi.
- `CreationTime` — Stempel waktu.

  Waktu pada daat pengklasifikasi ini didaftarkan.
- `LastUpdated` — Stempel waktu.

  Waktu pada saat pengklasifikasi ini terakhir diperbarui.
- `Version` — Nomor (panjang).

  Versi dari pengklasifikasi ini.
- `RowTag` – String UTF-8.

  Tag XML yang menunjuk elemen yang berisi setiap catatan dalam dokumen XML yang diurai. Ini tidak dapat mengidentifikasi elemen penutup diri (ditutup oleh `/>`). Elemen baris kosong

yang hanya berisi atribut dapat diurai selama itu berakhir dengan tag penutup (misalnya, `<row item_a="A" item_b="B"></row>` baik-baik saja, tapi `<row item_a="A" item_b="B" />` tidak).

JsonClassifier struktur

Sebuah pengklasifikasi untuk konten `JSON`.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pengklasifikasi.
- `CreationTime` — Stempel waktu.

  Waktu pada daat pengklasifikasi ini didaftarkan.
- `LastUpdated` — Stempel waktu.

  Waktu pada saat pengklasifikasi ini terakhir diperbarui.
- `Version` — Nomor (panjang).

  Versi dari pengklasifikasi ini.
- `JsonPath` – Wajib: String UTF-8.

  `JsonPath`String yang mendefinisikan data JSON untuk pengklasifikasi untuk mengklasifikasikan. AWS Gluemendukung subset dari JsonPath, seperti yang dijelaskan dalam Menulis JsonPath Pengklasifikasi Kustom.

CsvClassifier struktur

Pengklasifikasi untuk konten `CSV` kustom.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pengklasifikasi.

- `CreationTime` — Stempel waktu.

  Waktu pada daat pengklasifikasi ini didaftarkan.

- `LastUpdated` — Stempel waktu.

  Waktu pada saat pengklasifikasi ini terakhir diperbarui.

- `Version` — Nomor (panjang).

  Versi dari pengklasifikasi ini.

- `Delimiter` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 1 byte, yang cocok dengan Custom string pattern #10.

  Sebuah simbol kustom untuk menunjukkan apa yang memisahkan masing-masing entri kolom pada baris.

- `QuoteSymbol` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 1 byte, yang cocok dengan Custom string pattern #10.

  Sebuah simbol kustom untuk menunjukkan apa yang menggabungkan konten ke dalam satu nilai kolom tunggal. Ia harus berbeda dari pembatas kolom.

- `ContainsHeader` – String UTF-8 (nilai yang valid: `UNKNOWN` | `PRESENT` | `ABSENT`).

  Menunjukkan apakah file CSV berisi sebuah header.

- `Header` – Susunan string UTF-8.

  Sebuah daftar string yang mewakili nama kolom.

- `DisableValueTrimming` – Boolean.

  Menentukan tidak akan memotong nilai sebelum mengidentifikasi jenis nilai kolom. Nilai default-nya adalah `true`.

- `AllowSingleColumn` – Boolean.

  Memungkinkan pemrosesan file yang hanya berisi satu kolom.

- `CustomDatatypeConfigured` – Boolean.

  Mengaktifkan tipe data khusus untuk dikonfigurasi.

- `CustomDatatypes` – Susunan string UTF-8.

Daftar tipe data khusus termasuk “BINARY”, “BOOLEAN”, “DATE”, “DECIMAL”, “DOUBLE”, “FLOAT”, “INT”, “LONG”, “SHORT”, “STRING”, “TIMESTAMP”.

- `Serde` – String UTF-8 (nilai yang valid: `OpenCSVSerDe` | `LazySimpleSerDe` | `None`).

  Menetapkan SerDe untuk memproses CSV di classifier, yang akan diterapkan dalam Katalog Data. Nilai yang valid adalah `OpenCSVSerDe`, `LazySimpleSerDe`, dan `None`. Anda dapat menentukan `None` nilai saat Anda ingin crawler melakukan deteksi.

CreateGrokClassifierRequest struktur

Menentukan pengklasifikasi `grok` untuk `CreateClassifier` yang akan dibuat.

Bidang

- `Classification` – Wajib: String UTF-8.

  Pengidentifikasi format data yang cocok dengan pengklasifikasi, seperti Twitter, JSON, log Omniture, Amazon CloudWatch Logs, dan sebagainya.

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pengklasifikasi baru.

- `GrokPattern` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 2048 byte, yang cocok dengan A Logstash Grok string pattern.

  Pola grok yang digunakan oleh pengklasifikasi ini.

- `CustomPatterns` — String UTF-8, sepanjang tidak lebih dari 16000, yang cocok dengan URI address multi-line string pattern.

  Pola grok kustom opsional yang digunakan oleh pengklasifikasi ini.

UpdateGrokClassifierRequest struktur

Menentukan pengklasifikasi grok untuk memperbarui ketika diberikan ke `UpdateClassifier`.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama `GrokClassifier`.
- `Classification` – String UTF-8.

  Pengidentifikasi format data yang cocok dengan pengklasifikasi, seperti Twitter, JSON, log Omniture, Amazon CloudWatch Logs, dan sebagainya.
- `GrokPattern` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 2048 byte, yang cocok dengan A Logstash Grok string pattern.

  Pola grok yang digunakan oleh pengklasifikasi ini.
- `CustomPatterns` — String UTF-8, sepanjang tidak lebih dari 16000, yang cocok dengan URI address multi-line string pattern.

  Pola grok kustom opsional yang digunakan oleh pengklasifikasi ini.

struktur ClassifierRequest CreateXML

Menentukan pengklasifikasi XML untuk `CreateClassifier` yang akan dibuat.

Bidang

- `Classification` – Wajib: String UTF-8.

  Sebuah pengenal format data yang cocok dengan pengklasifikasi.
- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pengklasifikasi.
- `RowTag` – String UTF-8.

  Tag XML yang menunjuk elemen yang berisi setiap catatan dalam dokumen XML yang diurai. Ini tidak dapat mengidentifikasi elemen penutup diri (ditutup oleh `/>`). Elemen baris kosong yang hanya berisi atribut dapat diurai selama itu berakhir dengan tag penutup (misalnya, `<row item_a="A" item_b="B"></row>` baik-baik saja, tapi `<row item_a="A" item_b="B" />` tidak).

struktur ClassifierRequest UpdateXML

Menentukan pengklasifikasi XML yang akan diperbarui.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pengklasifikasi.

- `Classification` – String UTF-8.

  Sebuah pengenal format data yang cocok dengan pengklasifikasi.

- `RowTag` – String UTF-8.

  Tag XML yang menunjuk elemen yang berisi setiap catatan dalam dokumen XML yang diurai. Ini tidak dapat mengidentifikasi elemen penutup mandiri (ditutup oleh `/>`). Elemen baris kosong yang hanya berisi atribut dapat diurai selama diakhiri berakhir dengan tag penutup (misalnya, `<row item_a="A" item_b="B"></row>` tidak apa-apa, tapi `<row item_a="A" item_b="B" />` tidak boleh).

CreateJsonClassifierRequest struktur

Menentukan pengklasifikasi JSON untuk `CreateClassifier` yang akan dibuat.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pengklasifikasi.

- `JsonPath` – Wajib: String UTF-8.

  `JsonPath`String yang mendefinisikan data JSON untuk pengklasifikasi untuk mengklasifikasikan. AWS Gluemendukung subset dari JsonPath, seperti yang dijelaskan dalam Menulis JsonPath Pengklasifikasi Kustom.

UpdateJsonClassifierRequest struktur

Menentukan pengklasifikasi JSON yang akan diperbarui.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pengklasifikasi.
- `JsonPath` – String UTF-8.

  `JsonPath`String yang mendefinisikan data JSON untuk pengklasifikasi untuk mengklasifikasikan. AWS Gluemendukung subset dari JsonPath, seperti yang dijelaskan dalam Menulis JsonPath Pengklasifikasi Kustom.

CreateCsvClassifierRequest struktur

Menentukan pengklasifikasi CSV kustom untuk `CreateClassifier` yang akan dibuat.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pengklasifikasi.
- `Delimiter` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 1 byte, yang cocok dengan Custom string pattern #10.

  Sebuah simbol kustom untuk menunjukkan apa yang memisahkan masing-masing entri kolom pada baris.
- `QuoteSymbol` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 1 byte, yang cocok dengan Custom string pattern #10.

  Sebuah simbol kustom untuk menunjukkan apa yang menggabungkan konten ke dalam satu nilai kolom tunggal. Harus berbeda dari pembatas kolom.
- `ContainsHeader` – String UTF-8 (nilai yang valid: `UNKNOWN` | `PRESENT` | `ABSENT`).

  Menunjukkan apakah file CSV berisi sebuah header.

- `Header` – Susunan string UTF-8.

  Sebuah daftar string yang mewakili nama kolom.

- `DisableValueTrimming` – Boolean.

  Menentukan tidak akan memotong nilai sebelum mengidentifikasi jenis nilai kolom. Nilai default-nya adalah BETUL.

- `AllowSingleColumn` – Boolean.

  Memungkinkan pemrosesan file yang hanya berisi satu kolom.

- `CustomDatatypeConfigured` – Boolean.

  Mengaktifkan konfigurasi tipe data kustom.

- `CustomDatatypes` – Susunan string UTF-8.

  Membuat daftar tipe data kustom yang didukung.

- `Serde` – String UTF-8 (nilai yang valid: `OpenCSVSerDe` | `LazySimpleSerDe` | `None`).

  Menetapkan SerDe untuk memproses CSV di classifier, yang akan diterapkan dalam Katalog Data. Nilai yang valid adalah `OpenCSVSerDe`, `LazySimpleSerDe`, dan `None`. Anda dapat menentukan `None` nilai saat Anda ingin crawler melakukan deteksi.

UpdateCsvClassifierRequest struktur

Menentukan pengklasifikasi CSV kustom yang akan diperbarui.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pengklasifikasi.

- `Delimiter` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 1 byte, yang cocok dengan Custom string pattern #10.

  Sebuah simbol kustom untuk menunjukkan apa yang memisahkan masing-masing entri kolom pada baris.

- `QuoteSymbol` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 1 byte, yang cocok dengan Custom string pattern #10.

  Sebuah simbol kustom untuk menunjukkan apa yang menggabungkan konten ke dalam satu nilai kolom tunggal. Ia harus berbeda dari pembatas kolom.
- `ContainsHeader` – String UTF-8 (nilai yang valid: `UNKNOWN` | `PRESENT` | `ABSENT`).

  Menunjukkan apakah file CSV berisi sebuah header.
- `Header` – Susunan string UTF-8.

  Sebuah daftar string yang mewakili nama kolom.
- `DisableValueTrimming` – Boolean.

  Menentukan tidak akan memotong nilai sebelum mengidentifikasi jenis nilai kolom. Nilai default-nya adalah BETUL.
- `AllowSingleColumn` – Boolean.

  Memungkinkan pemrosesan file yang hanya berisi satu kolom.
- `CustomDatatypeConfigured` – Boolean.

  Menentukan konfigurasi tipe data kustom.
- `CustomDatatypes` – Susunan string UTF-8.

  Menentukan daftar tipe data kustom didukung.
- `Serde` – String UTF-8 (nilai yang valid: `OpenCSVSerDe` | `LazySimpleSerDe` | `None`).

  Menetapkan SerDe untuk memproses CSV di classifier, yang akan diterapkan dalam Katalog Data. Nilai yang valid adalah `OpenCSVSerDe`, `LazySimpleSerDe`, dan `None`. Anda dapat menentukan `None` nilai saat Anda ingin crawler melakukan deteksi.

Operasi

- CreateClassifier tindakan (Python: create_classifier)
- DeleteClassifier tindakan (Python: delete_classifier)
- GetClassifier tindakan (Python: get_classifier)
- GetClassifiers tindakan (Python: get_classifiers)
- UpdateClassifier tindakan (Python: update_classifier)

CreateClassifier tindakan (Python: create_classifier)

Menciptakan pengklasifikasi di akun pengguna. Bisa berupa sebuah `GrokClassifier`, sebuah `XMLClassifier`, sebuah `JsonClassifier`, atau `CsvClassifier`, tergantung pada bidang permintaan yang ada.

Permintaan

- `GrokClassifier` — Sebuah objek CreateGrokClassifierRequest.

  Sebuah objek `GrokClassifier` yang menentukan pengklasifikasi yang akan dibuat.
- `XMLClassifier` — Sebuah objek CreateXML ClassifierRequest.

  Sebuah objek `XMLClassifier` yang menentukan pengklasifikasi yang akan dibuat.
- `JsonClassifier` — Sebuah objek CreateJsonClassifierRequest.

  Sebuah objek `JsonClassifier` yang menentukan pengklasifikasi yang akan dibuat.
- `CsvClassifier` — Sebuah objek CreateCsvClassifierRequest.

  Sebuah objek `CsvClassifier` yang menentukan pengklasifikasi yang akan dibuat.

Response

- Tidak ada parameter Respons.

Kesalahan

- `AlreadyExistsException`
- `InvalidInputException`
- `OperationTimeoutException`

DeleteClassifier tindakan (Python: delete_classifier)

Menghapus sebuah pengklasifikasi dari Katalog Data.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Nama pengklasifikasi yang akan dihapus.

Response

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `OperationTimeoutException`

GetClassifier tindakan (Python: get_classifier)

Mengambil sebuah pengklasifikasi berdasarkan nama.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pengklasifikasi yang akan diambil.

Response

- `Classifier` — Sebuah objek Pengklasifikasi.

  Pengklasifikasi yang diminta.

Kesalahan

- `EntityNotFoundException`
- `OperationTimeoutException`

GetClassifiers tindakan (Python: get_classifiers)

Mencantumkan semua objek pengklasifikasi dalam Katalog Data.

Permintaan

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Ukuran daftar untuk yang akan dikembalikan (opsional).
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan opsional.

Response

- `Classifiers` – Susunan objek Pengklasifikasi.

  Daftar objek pengklasifikasi yang diminta.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan.

Kesalahan

- `OperationTimeoutException`

UpdateClassifier tindakan (Python: update_classifier)

Memodifikasi pengklasifikasi yang ada (sebuah `GrokClassifier`, sebuah `XMLClassifier`, sebuah `JsonClassifier`, atau `CsvClassifier`, tergantung pada bidang mana yang ada).

Permintaan

- `GrokClassifier` — Sebuah objek UpdateGrokClassifierRequest.

  Sebuah objek `GrokClassifier` dengan bidang yang diperbarui.
- `XMLClassifier` — Sebuah objek UpdateXML ClassifierRequest.

  Sebuah objek `XMLClassifier` dengan bidang yang diperbarui.
- `JsonClassifier` — Sebuah objek UpdateJsonClassifierRequest.

  Sebuah objek `JsonClassifier` dengan bidang yang diperbarui.
- `CsvClassifier` — Sebuah objek UpdateCsvClassifierRequest.

Sebuah objek `CsvClassifier` dengan bidang yang diperbarui.

Response

- Tidak ada parameter Respons.

Kesalahan

- `InvalidInputException`
- `VersionMismatchException`
- `EntityNotFoundException`
- `OperationTimeoutException`

API Crawler

Crawler API menjelaskan tipe data AWS Glue crawler, bersama dengan API untuk membuat, menghapus, memperbarui, dan mencantumkan crawler.

Jenis data

- Struktur perayap
- Struktur jadwal
- CrawlerTargets struktur
- Struktur S3Target
- Struktur S3 DeltaCatalogTarget
- Struktur S3 DeltaDirectTarget
- JdbcTarget struktur
- Struktur MongoDBTarget
- Struktur DynamodbTarget
- DeltaTarget struktur
- IcebergTarget struktur
- HudiTarget struktur

- CatalogTarget struktur
- CrawlerMetrics struktur
- CrawlerHistory struktur
- CrawlsFilter struktur
- SchemaChangePolicy struktur
- LastCrawlInfo struktur
- RecrawlPolicy struktur
- LineageConfiguration struktur
- LakeFormationConfiguration struktur

Struktur perayap

Menentukan sebuah program crawler yang meneliti sumber data dan menggunakan pengklasifikasi untuk mencoba menentukan skemanya. Jika berhasil, crawler mencatat metadata yang terkait sumber data di AWS Glue Data Catalog.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler.
- `Role` – String UTF-8.

  Amazon Resource Name (ARN) dari sebuah IAM role yang digunakan untuk mengakses sumber daya pelanggan, seperti data Amazon Simple Storage Service (Amazon S3).
- `Targets` — Sebuah objek CrawlerTargets.

  Sebuah sekumpulan target yang akan dilakukan perayapan padanya.
- `DatabaseName` – String UTF-8.

  Nama basis data tempat output crawler disimpan.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi crawler.

- `Classifiers` – Susunan string UTF-8.

  Daftar string UTF-8 yang menentukan pengklasifikasi kustom yang dikaitkan dengan crawler.

- `RecrawlPolicy` — Sebuah objek RecrawlPolicy.

  Sebuah kebijakan yang menentukan apakah akan melakukan perayapan pada seluruh set data lagi, atau hanya pada folder yang ditambahkan sejak crawler terakhir kali dijalankan.

- `SchemaChangePolicy` — Sebuah objek SchemaChangePolicy.

  Kebijakan yang menentukan perilaku pembaruan dan penghapusan untuk crawler.

- `LineageConfiguration` — Sebuah objek LineageConfiguration.

  Sebuah konfigurasi yang menentukan apakah garis keturunan data diaktifkan untuk crawler.

- `State` – String UTF-8 (nilai yang valid: `READY` | `RUNNING` | `STOPPING`).

  Menunjukkan apakah crawler sedang berjalan, atau apakah eksekusi-nya ditunda.

- `TablePrefix` — String UTF-8, dengan panjang tidak lebih dari 128 byte.

  Prefiks ditambahkan ke nama tabel yang dibuat.

- `Schedule` — Sebuah objek Jadwal.

  Untuk crawler terjadwal, jadwal pada saat crawler berjalan.

- `CrawlElapsedTime` — Nomor (panjang).

  Jika crawler berjalan, berisi total waktu yang berlalu sejak perayapan terakhir dimulai.

- `CreationTime` — Stempel waktu.

  Waktu saat crawler diciptakan.

- `LastUpdated` — Stempel waktu.

  Waktu saat crawler terakhir diperbarui.

- `LastCrawl` — Sebuah objek LastCrawlInfo.

  Status perayapan terakhir, dan kemungkinan kesalahan informasi jika terjadi kesalahan.

- `Version` — Nomor (panjang).

  Versi crawler.

- `Configuration` – String UTF-8.

Informasi konfigurasi crawler. String JSON berversi ini memungkinkan pengguna untuk menentukan aspek perilaku perayap. Untuk informasi selengkapnya, lihat Menyetel opsi konfigurasi crawler.

- `CrawlerSecurityConfiguration` — String UTF-8, dengan panjang tidak lebih dari 128 byte.

  Nama struktur `SecurityConfiguration` yang akan digunakan oleh crawler ini.

- `LakeFormationConfiguration` — Sebuah objek LakeFormationConfiguration.

  Menentukan apakah crawler harus menggunakan AWS Lake Formation kredensyal untuk crawler, bukan kredensyal peran IAM.

Struktur jadwal

Sebuah objek penjadwalan menggunakan pernyataan `cron` untuk menjadwalkan sebuah peristiwa.

Bidang

- `ScheduleExpression` – String UTF-8.

  Sebuah ekspresi `cron` yang digunakan untuk menentukan jadwal (lihat Jadwal Berbasis Waktu untuk Tugas dan Crawler. Sebagai contoh, untuk menjalankan sesuatu setiap hari pada 12:15 UTC, Anda harus menentukan `cron(15 12 * * ? *)`.

- `State` – String UTF-8 (nilai yang valid: `SCHEDULED` | `NOT_SCHEDULED` | `TRANSITIONING`).

  Status jadwal.

CrawlerTargets struktur

Menentukan data yang disimpan ke perayapan.

Bidang

- `S3Targets` – Susunan objek S3Target.

  Menentukan target Amazon Simple Storage Service (Amazon S3).

- `JdbcTargets` – Susunan objek JdbcTarget.

  Menentukan target JDBC.

- `MongoDBTargets` – Susunan objek MongoDbTarget.

  Menentukan target Amazon DocumentDB atau MongoDB.

- `DynamoDBTargets` – Susunan objek DynamoDBTarget.

  Menentukan target Amazon DynamoDB.

- `CatalogTargets` – Susunan objek CatalogTarget.

  Menentukan AWS Glue Data Catalog target.

- `DeltaTargets` – Susunan objek DeltaTarget.

  Menentukan target penyimpanan data Delta.

- `IcebergTargets` – Susunan objek IcebergTarget.

  Menentukan target penyimpanan data Apache Iceberg.

- `HudiTargets` – Susunan objek HudiTarget.

  Menentukan target penyimpanan data Apache Hudi.

Struktur S3Target

Menentukan penyimpanan data dalam Amazon Simple Storage Service (Amazon S3).

Bidang

- `Path` – String UTF-8.

  Path menuju target Amazon S3.

- `Exclusions` – Susunan string UTF-8.

  Daftar pola glob yang digunakan untuk mengecualikan dari perayapan. Untuk informasi selengkapnya, lihat: Tabel Katalog dengan Crawler.

- `ConnectionName` – String UTF-8.

  Nama koneksi yang memungkinkan tugas atau crawler untuk mengakses data di Amazon S3 dalam lingkungan Amazon Virtual Private Cloud (Amazon VPC).

- `SampleSize` — Nomor (bilangan bulat).

Menetapkan jumlah file di setiap folder daun yang akan di-crawl saat melakukan perayapan pada file sampel dalam set data. Jika tidak diatur, maka semua file di-crawl. Nilai yang valid adalah bilangan bulat antara 1 dan 249.

- `EventQueueArn` – String UTF-8.

  Amazon SQS ARN yang valid. Misalnya, `arn:aws:sqs:region:account:sqs`.

- `DlqEventQueueArn` – String UTF-8.

  SQS ARN surat mati Amazon yang valid. Misalnya, `arn:aws:sqs:region:account:deadLetterQueue`.

Struktur S3 DeltaCatalogTarget

Menentukan target yang menulis ke sumber data Delta Lake di Katalog AWS Glue Data.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.

- `PartitionKeys` – Susunan string UTF-8.

  Menentukan partisi asli menggunakan urutan kunci.

- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk menulis ke.

- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk menulis.

- `AdditionalOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

Menentukan pilihan koneksi tambahan untuk konektor.

- `SchemaChangePolicy` — Sebuah objek CatalogSchemaChangePolicy.

  Kebijakan yang menentukan perilaku update untuk crawler.

Struktur S3 DeltaDirectTarget

Menentukan target yang menulis ke sumber data Delta Lake di. Amazon S3

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.

- `PartitionKeys` – Susunan string UTF-8.

  Menentukan partisi asli menggunakan urutan kunci.

- `Path` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Jalur Amazon S3 dari sumber data Delta Lake Anda untuk menulis.

- `Compression` – Wajib: String UTF-8 (nilai yang valid: `uncompressed="UNCOMPRESSED"` | `snappy="SNAPPY"`).

  Menentukan bagaimana data dikompresi. Hal ini umumnya tidak diperlukan jika data memiliki sebuah ekstensi file standar. Nilai yang mungkin adalah `"gzip"` dan`"bzip"`).

- `Format` – Wajib: String UTF-8 (nilai yang valid: `json="JSON"` | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | `delta="DELTA"`).

  Menentukan format output data untuk target.

- `AdditionalOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

Menentukan pilihan koneksi tambahan untuk konektor.

- `SchemaChangePolicy` — Sebuah objek DirectSchemaChangePolicy.

  Kebijakan yang menentukan perilaku update untuk crawler.

JdbcTarget struktur

Menentukan penyimpanan data JDBC untuk perayapan.

Bidang

- `ConnectionName` – String UTF-8.

  Nama koneksi yang akan digunakan untuk menyambungkan ke target JDBC.
- `Path` – String UTF-8.

  Path target JDBC.
- `Exclusions` – Susunan string UTF-8.

  Daftar pola glob yang digunakan untuk mengecualikan dari perayapan. Untuk informasi selengkapnya, lihat: Tabel Katalog dengan Crawler.
- `EnableAdditionalMetadata` – Susunan string UTF-8.

  Tentukan nilai `RAWTYPES` atau `COMMENTS` untuk mengaktifkan metadata tambahan dalam respons tabel. `RAWTYPES`menyediakan tipe data tingkat asli. `COMMENTS`memberikan komentar yang terkait dengan kolom atau tabel dalam database.

  Jika Anda tidak memerlukan metadata tambahan, biarkan bidang kosong.

Struktur MongoDBTarget

Menentukan penyimpanan data Amazon DocumentDB atau MongoDB yang akan di-crawl.

Bidang

- `ConnectionName` – String UTF-8.

  Nama koneksi yang akan digunakan untuk menghubungkan ke target Amazon DocumentDB atau MongoDB.

- `Path` – String UTF-8.

  Path target Amazon DocumentDB atau MongoDB target (basis data/koleksi).

- `ScanAll` – Boolean.

  Menunjukkan apakah akan memindai semua catatan, atau mengambil sampel baris dari tabel. Memindai semua catatan dapat memakan waktu lama ketika tabel tersebut bukan merupakan tabel throughput tinggi.

  Sebuah nilai `true` berarti memindai semua catatan, sementara nilai `false` berarti mengambil sampel catatan. Jika tidak ada nilai yang ditentukan, nilai defaultnya menjadi `true`.

Struktur DynamodbTarget

Menentukan tabel Amazon DynamoDB untuk bergerak.

Bidang

- `Path` – String UTF-8.

  Nama dari tabel DynamoDB untuk bergerak.

- `scanAll` – Boolean.

  Menunjukkan apakah akan memindai semua catatan, atau mengambil sampel baris dari tabel. Memindai semua catatan dapat memakan waktu lama ketika tabel tersebut bukan merupakan tabel throughput tinggi.

  Sebuah nilai `true` berarti memindai semua catatan, sementara nilai `false` berarti mengambil sampel catatan. Jika tidak ada nilai yang ditentukan, nilai defaultnya menjadi `true`.

- `scanRate` — Nomor (ganda).

  Persentase unit kapasitas baca yang dikonfigurasi untuk digunakan oleh AWS Glue crawler. Unit kapasitas baca adalah istilah yang didefinisikan oleh DynamoDB, dan merupakan nilai numerik yang bertindak sebagai tingkat pembatar untuk jumlah baca yang dapat dilakukan pada tabel tersebut per detik.

  Nilai-nilai yang valid adalah nol atau nilai antara 0,1 sampai 1,5. Nilai nol digunakan ketika pengguna tidak memberikan nilai, dan default-nya menjadi 0,5 Unit Kapasitas Baca yang

dikonfigurasi (untuk tabel yang disediakan), atau maksimal 0,25 Unit Kapasitas Baca yang dikonfigurasi (untuk tabel yang menggunakan mode sesuai permintaan).

DeltaTarget struktur

Menentukan penyimpanan data Delta untuk merayapi satu atau lebih tabel Delta.

Bidang

- `DeltaTables` – Susunan string UTF-8.

  Daftar jalur Amazon S3 ke tabel Delta.

- `ConnectionName` – String UTF-8.

  Nama koneksi yang akan digunakan untuk terhubung ke target tabel Delta.

- `WriteManifest` – Boolean.

  Menentukan apakah akan menulis file manifes ke jalur tabel Delta.

- `CreateNativeDeltaTable` – Boolean.

  Menentukan apakah crawler akan membuat tabel asli, untuk memungkinkan integrasi dengan mesin kueri yang mendukung kueri log transaksi Delta secara langsung.

IcebergTarget struktur

Menentukan sumber data Apache Iceberg di mana tabel Iceberg disimpan dalam. Amazon S3

Bidang

- `Paths` – Susunan string UTF-8.

  Satu atau beberapa Amazon S3 jalur yang berisi folder metadata Iceberg sebagai. `s3://bucket/prefix`

- `ConnectionName` – String UTF-8.

  Nama koneksi yang digunakan untuk terhubung ke target Gunung Es.

- `Exclusions` – Susunan string UTF-8.

Daftar pola glob yang digunakan untuk mengecualikan dari perayapan. Untuk informasi selengkapnya, lihat: Tabel Katalog dengan Crawler.

- `MaximumTraversalDepth` — Nomor (bilangan bulat).

  Kedalaman maksimum Amazon S3 jalur yang dapat dilalui crawler untuk menemukan folder metadata Iceberg di jalur Anda. Amazon S3 Digunakan untuk membatasi waktu berjalan crawler.

HudiTarget struktur

Menentukan sumber data Apache Hudi.

Bidang

- `Paths` – Susunan string UTF-8.

  Sebuah array string Amazon S3 lokasi untuk Hudi, masing-masing menunjukkan folder root dengan mana file metadata untuk tabel Hudi berada. Folder Hudi mungkin terletak di folder anak dari folder root.

  Crawler akan memindai semua folder di bawah jalur untuk folder Hudi.

- `ConnectionName` – String UTF-8.

  Nama koneksi yang digunakan untuk terhubung ke target Hudi. Jika file Hudi Anda disimpan dalam bucket yang memerlukan otorisasi VPC, Anda dapat mengatur properti koneksi mereka di sini.

- `Exclusions` – Susunan string UTF-8.

  Daftar pola glob yang digunakan untuk mengecualikan dari perayapan. Untuk informasi selengkapnya, lihat: Tabel Katalog dengan Crawler.

- `MaximumTraversalDepth` — Nomor (bilangan bulat).

  Kedalaman maksimum Amazon S3 jalur yang dapat dilalui crawler untuk menemukan folder metadata Hudi di jalur Anda. Amazon S3 Digunakan untuk membatasi waktu berjalan crawler.

CatalogTarget struktur

Menentukan AWS Glue Data Catalog target.

Bidang

- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data yang akan disinkronkan.
- `Tables` — Wajib: Susunan string UTF-8, setidaknya 1 string.

  Daftar tabel yang akan disinkronkan.
- `ConnectionName` – String UTF-8.

  Nama sambungan untuk tabel Katalog Data yang didukung Amazon S3 menjadi target crawl saat menggunakan jenis `Catalog` koneksi yang dipasangkan dengan tipe Sambungan. `NETWORK`
- `EventQueueArn` – String UTF-8.

  Amazon SQS ARN yang valid. Misalnya, `arn:aws:sqs:region:account:sqs`.
- `DlqEventQueueArn` – String UTF-8.

  SQS ARN surat mati Amazon yang valid. Misalnya, `arn:aws:sqs:region:account:deadLetterQueue`.

CrawlerMetrics struktur

Metrik untuk sebuah crawler yang ditentukan.

Bidang

- `CrawlerName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler.
- `TimeLeftSeconds` — Nomor (ganda), tidak lebih dari Tidak Ada.

  Perkiraan waktu tersisa untuk menyelesaikan perayapan yang berjalan.
- `StillEstimating` – Boolean.

  BETUL jika crawler masih memperkirakan berapa lama waktu yang dibutuhkan untuk menyelesaikan eksekusi ini.
- `LastRuntimeSeconds` — Nomor (ganda), tidak lebih dari Tidak Ada.

Durasi eksekusi terbaru oleh crawler, dalam hitungan detik.

- `MedianRuntimeSeconds` — Nomor (ganda), tidak lebih dari Tidak Ada.

  Durasi median dari eksekusi crawler ini, dalam hitungan detik.

- `TablesCreated` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Jumlah tabel yang dibuat oleh crawler ini.

- `TablesUpdated` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Jumlah tabel yang diperbarui oleh crawler ini.

- `TablesDeleted` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Jumlah tabel yang dihapus oleh crawler ini.

CrawlerHistory struktur

Berisi informasi untuk menjalankan crawler.

Bidang

- `CrawlId` – String UTF-8.

  Pengenal UUID untuk setiap crawl.

- `State` – String UTF-8 (nilai yang valid: `RUNNING` | `COMPLETED` | `FAILED` | `STOPPED`).

  Keadaan merangkak.

- `StartTime` — Stempel waktu.

  Tanggal dan waktu saat perayapan dimulai.

- `EndTime` — Stempel waktu.

  Tanggal dan waktu di mana perayapan berakhir.

- `Summary` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Ringkasan run untuk crawl tertentu di JSON. Berisi tabel katalog dan partisi yang ditambahkan, diperbarui, atau dihapus.

- `ErrorMessage` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Jika terjadi kesalahan, pesan kesalahan terkait dengan crawl.

- `LogGroup` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 512 byte, yang cocok dengan Log group string pattern.

  Grup log yang dikaitkan dengan perayapan.

- `LogStream` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 512 byte, yang cocok dengan Log-stream string pattern.

  Pengaliran log yang dikaitkan dengan perayapan.

- `MessagePrefix` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Awalan untuk CloudWatch pesan tentang crawl ini.

- `DPUHour` — Nomor (ganda), tidak lebih dari Tidak Ada.

  Jumlah unit pemrosesan data (DPU) yang digunakan dalam jam untuk crawl.

CrawlsFilter struktur

Daftar bidang, pembanding, dan nilai yang dapat Anda gunakan untuk memfilter crawler berjalan untuk crawler tertentu.

Bidang

- `FieldName` – String UTF-8 (nilai yang valid: `CRAWL_ID` | `STATE` | `START_TIME` | `END_TIME` | `DPU_HOUR`).

  Kunci yang digunakan untuk memfilter crawler berjalan untuk crawler tertentu. Nilai yang valid untuk masing-masing nama bidang adalah:

  - `CRAWL_ID`: String yang mewakili identifier UUID untuk crawl.
  - `STATE`: Sebuah string yang mewakili status crawl.
  - `START_TIME`dan`END_TIME`: Stempel waktu zaman dalam milidetik.
  - `DPU_HOUR`: Jumlah jam unit pemrosesan data (DPU) yang digunakan untuk crawl.

- `FilterOperator` – String UTF-8 (nilai valid: `GT` | `GE` | `LT` | `LE` | `EQ` | `NE`).

Komparator didefinisikan yang beroperasi pada nilai. Operator yang tersedia adalah:

- `GT`: Lebih besar dari.
- `GE`: Lebih besar dari atau sama dengan.
- `LT`: Kurang dari.
- `LE`: Kurang dari atau sama dengan.
- `EQ`: Sama dengan.
- `NE`: Tidak sama dengan.

- `FieldValue` – String UTF-8.

  Nilai yang diberikan untuk perbandingan pada bidang crawl.

SchemaChangePolicy struktur

Kebijakan yang menentukan perilaku pembaruan dan penghapusan untuk perayap.

Bidang

- `UpdateBehavior` – String UTF-8 (nilai yang valid: `LOG` | `UPDATE_IN_DATABASE`).

  Perilaku pembaruan ketika crawler menemukan skema yang berubah.
- `DeleteBehavior` – String UTF-8 (nilai yang valid: `LOG` | `DELETE_FROM_DATABASE` | `DEPRECATE_IN_DATABASE`).

  Perilaku penghapusan saat crawler menemukan objek yang dihapus.

LastCrawlInfo struktur

Informasi status dan kesalahan tentang perayapan terbaru.

Bidang

- `Status` – String UTF-8 (nilai yang valid: `SUCCEEDED` | `CANCELLED` | `FAILED`).

  Status perayapan terakhir.
- `ErrorMessage` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

Jika terjadi kesalahan, informasi kesalahan tentang perayapan terakhir.

- `LogGroup` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 512 byte, yang cocok dengan Log group string pattern.

  Grup log untuk perayapan terakhir.

- `LogStream` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 512 byte, yang cocok dengan Log-stream string pattern.

  Pengaliran log untuk perayapan terakhir.

- `MessagePrefix` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Prefiks untuk pesan tentang perayapan ini.

- `StartTime` — Stempel waktu.

  Waktu saat perayapan dimulai.

RecrawlPolicy struktur

Saat melakukan perayapan pada sumber data Amazon S3 setelah perayapan pertama selesai, tentukan apakah akan melakukan perayapan pada seluruh set data lagi atau hanya pada folder yang ditambahkan sejak crawler terakhir kali dijalankan. Untuk informasi selengkapnya, lihat Perayapan Tambahan AWS Glue dalam panduan developer.

Bidang

- `RecrawlBehavior` – String UTF-8 (nilai yang valid: `CRAWL_EVERYTHING` | `CRAWL_NEW_FOLDERS_ONLY` | `CRAWL_EVENT_MODE`).

  Menentukan apakah akan melakukan perayapan pada seluruh set data lagi, atau hanya pada folder yang ditambahkan sejak crawler terakhir kali dijalankan.

  Sebuah nilai `CRAWL_EVERYTHING` menentukan untuk melakukan perayapan pada seluruh set data lagi.

  Sebuah nilai `CRAWL_NEW_FOLDERS_ONLY` menentukan untuk hanya melakukan perayapan pada folder yang ditambahkan sejak menjalankan crawler terakhir kali dijalankan.

Nilai `CRAWL_EVENT_MODE` menentukan crawling hanya perubahan yang diidentifikasi oleh peristiwa Amazon S3.

LineageConfiguration struktur

Menentukan pengaturan konfigurasi garis keturunan data untuk crawler tersebut.

Bidang

- `CrawlerLineageSettings` – String UTF-8 (nilai yang valid: `ENABLE` | `DISABLE`).

  Menentukan apakah garis keturunan data diaktifkan untuk crawler. Nilai yang valid adalah:

  - AKTIFKAN: mengaktifkan garis keturunan data untuk crawler
  - NONAKTIFKAN: menonaktifkan garis keturunan data untuk crawler

LakeFormationConfiguration struktur

Menentukan pengaturan AWS Lake Formation konfigurasi untuk crawler.

Bidang

- `UseLakeFormationCredentials` – Boolean.

  Menentukan apakah akan menggunakan AWS Lake Formation kredensyal untuk crawler, bukan kredensyal peran IAM.

- `AccountId`— String UTF-8, panjangnya tidak lebih dari 12 byte.

  Diperlukan untuk crawl lintas akun. Untuk crawl akun yang sama dengan data target, ini dapat dibiarkan sebagai null.

Operasi

- CreateCrawler tindakan (Python: create_crawler)
- DeleteCrawler tindakan (Python: delete_crawler)
- GetCrawler tindakan (Python: get_crawler)
- GetCrawlers tindakan (Python: get_crawlers)

- GetCrawlerMetrics tindakan (Python: get_crawler_metrics)
- UpdateCrawler tindakan (Python: update_crawler)
- StartCrawler tindakan (Python: start_crawler)
- StopCrawler tindakan (Python: stop_crawler)
- BatchGetCrawlers tindakan (Python: batch_get_crawlers)
- ListCrawlers tindakan (Python: list_crawlers)
- ListCrawls tindakan (Python: list_crawls)

CreateCrawler tindakan (Python: create_crawler)

Menciptakan sebuah crawler baru dengan target tertentu, peran, konfigurasi, dan jadwal opsional. Setidaknya satu target perayapan harus ditentukan, dalam bidang `s3Targets`, bidang `jdbcTargets`, atau bidang `DynamoDBTargets`.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler baru.
- `Role` – Wajib: String UTF-8.

  IAM role atau Amazon Resource Name (ARN) dari IAM role yang digunakan oleh crawler baru tersebut untuk mengakses sumber daya pelanggan.
- `DatabaseName` – String UTF-8.

  AWS Glue Database tempat hasil ditulis, seperti:`arn:aws:daylight:us-east-1::database/sometable/*`.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi crawler baru.
- `Targets` — Wajib: Sebuah objek CrawlerTargets.

  Sebuah daftar sekumpulan target yang akan dilakukan perayapan padanya.
- `Schedule` – String UTF-8.

Sebuah ekspresi `cron` yang digunakan untuk menentukan jadwal (lihat Jadwal Berbasis Waktu untuk Tugas dan Crawler. Sebagai contoh, untuk menjalankan sesuatu setiap hari pada 12:15 UTC, Anda harus menentukan `cron(15 12 * * ? *)`.

- `Classifiers` – Susunan string UTF-8.

  Daftar pengklasifikasi kustom yang didaftarkan oleh pengguna. Secara default, semua pengklasifikasi bawaan disertakan dalam sebuah perayapan, tetapi pengklasifikasi kustom ini selalu menimpa pengklasifikasi default untuk klasifikasi tertentu.

- `TablePrefix` — String UTF-8, dengan panjang tidak lebih dari 128 byte.

  Prefiks tabel yang digunakan untuk tabel katalog yang dibuat.

- `SchemaChangePolicy` — Sebuah objek SchemaChangePolicy.

  Kebijakan untuk perilaku pembaruan dan penghapusan crawler.

- `RecrawlPolicy` — Sebuah objek RecrawlPolicy.

  Sebuah kebijakan yang menentukan apakah akan melakukan perayapan pada seluruh set data lagi, atau hanya pada folder yang ditambahkan sejak crawler terakhir kali dijalankan.

- `LineageConfiguration` — Sebuah objek LineageConfiguration.

  Menentukan pengaturan konfigurasi garis keturunan data untuk crawler tersebut.

- `LakeFormationConfiguration` — Sebuah objek LakeFormationConfiguration.

  Menentukan pengaturan AWS Lake Formation konfigurasi untuk crawler.

- `Configuration` – String UTF-8.

  Informasi konfigurasi crawler. String JSON berversi ini memungkinkan pengguna untuk menentukan aspek perilaku perayap. Untuk informasi selengkapnya, lihat Menyetel opsi konfigurasi crawler.

- `CrawlerSecurityConfiguration` — String UTF-8, dengan panjang tidak lebih dari 128 byte.

  Nama struktur `SecurityConfiguration` yang akan digunakan oleh crawler ini.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

Tag untuk digunakan dengan permintaan crawler ini. Anda dapat menggunakan tag untuk membatasi akses ke crawler. Untuk informasi selengkapnya tentang AWS tag AWS Glue, lihat Tag AWS Glue di panduan pengembang.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `InvalidInputException`
- `AlreadyExistsException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`

DeleteCrawler tindakan (Python: delete_crawler)

Menghapus crawler yang ditentukan dari AWS Glue Data Catalog, kecuali status crawler. `RUNNING`

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler yang akan dihapus.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `CrawlerRunningException`

- `SchedulerTransitioningException`
- `OperationTimeoutException`

GetCrawler tindakan (Python: get_crawler)

Mengambil metadata untuk crawler yang ditentukan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler yang untuknya metadata diambil.

Respons

- `Crawler` — Sebuah objek Crawler.

  Metadata untuk crawler yang ditentukan.

Kesalahan

- `EntityNotFoundException`
- `OperationTimeoutException`

GetCrawlers tindakan (Python: get_crawlers)

Mengambil metadata untuk semua crawler yang didefinisikan dalam akun pelanggan.

Permintaan

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah crawler yang akan dikembalikan pada setiap panggilan.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah permintaan kelanjutan.

Respons

- `Crawlers` – Susunan objek Crawler.

  Daftar metadata crawler.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika daftar yang dikembalikan belum mencapai akhir yang didefinisikan dalam akun pelanggan ini.

Kesalahan

- `OperationTimeoutException`

GetCrawlerMetrics tindakan (Python: get_crawler_metrics)

Mengambil metrik tentang crawler yang ditentukan.

Permintaan

- `CrawlerNameList` — Susunan string UTF-8, tidak lebih dari 100 string.

  Daftar nama crawler yang akan diambil metriknya.

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Ukuran maksimum daftar yang akan dikembalikan.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Respons

- `CrawlerMetricsList` – Susunan objek CrawlerMetrics.

  Daftar metrik untuk crawler yang ditentukan.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika daftar yang dikembalikan tidak berisi metrik terakhir yang tersedia.

Kesalahan

- `OperationTimeoutException`

UpdateCrawler tindakan (Python: update_crawler)

Memperbarui sebuah crawler. Jika sebuah crawler sedang berjalan, Anda harus menghentikannya menggunakan `StopCrawler` sebelum memperbaruinya.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler baru.

- `Role` – String UTF-8.

  IAM role atau Amazon Resource Name (ARN) dari IAM role yang digunakan oleh crawler baru tersebut untuk mengakses sumber daya pelanggan.

- `DatabaseName` – String UTF-8.

  AWS Glue Database tempat hasil disimpan, seperti:`arn:aws:daylight:us-east-1::database/sometable/*`.

- `Description` — String UTF-8, sepanjang tidak lebih dari 2048, yang cocok dengan URI address multi-line string pattern.

  Deskripsi crawler baru.

- `Targets` — Sebuah objek CrawlerTargets.

  Daftar target yang akan di-crawl.

- `Schedule` – String UTF-8.

  Sebuah ekspresi `cron` yang digunakan untuk menentukan jadwal (lihat Jadwal Berbasis Waktu untuk Tugas dan Crawler. Sebagai contoh, untuk menjalankan sesuatu setiap hari pada 12:15 UTC, Anda harus menentukan `cron(15 12 * * ? *)`.

- `Classifiers` – Susunan string UTF-8.

Daftar pengklasifikasi kustom yang didaftarkan oleh pengguna. Secara default, semua pengklasifikasi bawaan disertakan dalam sebuah perayapan, tetapi pengklasifikasi kustom ini selalu menimpa pengklasifikasi default untuk klasifikasi tertentu.

- `TablePrefix` — String UTF-8, dengan panjang tidak lebih dari 128 byte.

  Prefiks tabel yang digunakan untuk tabel katalog yang dibuat.

- `SchemaChangePolicy` — Sebuah objek SchemaChangePolicy.

  Kebijakan untuk perilaku pembaruan dan penghapusan crawler.

- `RecrawlPolicy` — Sebuah objek RecrawlPolicy.

  Sebuah kebijakan yang menentukan apakah akan melakukan perayapan pada seluruh set data lagi, atau hanya pada folder yang ditambahkan sejak crawler terakhir kali dijalankan.

- `LineageConfiguration` — Sebuah objek LineageConfiguration.

  Menentukan pengaturan konfigurasi garis keturunan data untuk crawler tersebut.

- `LakeFormationConfiguration` — Sebuah objek LakeFormationConfiguration.

  Menentukan pengaturan AWS Lake Formation konfigurasi untuk crawler.

- `Configuration` – String UTF-8.

  Informasi konfigurasi crawler. String JSON berversi ini memungkinkan pengguna untuk menentukan aspek perilaku perayap. Untuk informasi selengkapnya, lihat Menyetel opsi konfigurasi crawler.

- `CrawlerSecurityConfiguration` — String UTF-8, dengan panjang tidak lebih dari 128 byte.

  Nama struktur `SecurityConfiguration` yang akan digunakan oleh crawler ini.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `InvalidInputException`
- `VersionMismatchException`

- `EntityNotFoundException`
- `CrawlerRunningException`
- `OperationTimeoutException`

StartCrawler tindakan (Python: start_crawler)

Memmulai sebuah perayapan menggunakan crawler yang ditentukan, terlepas dari apa yang dijadwalkan. Jika crawler sudah berjalan, mengembalikan file. CrawlerRunningException

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler yang akan dimulai.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `CrawlerRunningException`
- `OperationTimeoutException`

StopCrawler tindakan (Python: stop_crawler)

Jika crawler yang ditentukan sedang berjalan, berhenti melakukan perayapan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler yang akan dihentikan.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `CrawlerNotRunningException`
- `CrawlerStoppingException`
- `OperationTimeoutException`

BatchGetCrawlers tindakan (Python: batch_get_crawlers)

Mengembalikan daftar metadata sumber daya untuk daftar yang nama crawler yang ditentukan. Setelah memanggil operasi `ListCrawlers`, Anda dapat memanggil operasi ini untuk mengakses data yang Anda telah diberikan izinnya. Operasi ini mendukung semua izin IAM, termasuk syarat izin yang menggunakan tag.

Permintaan

- `CrawlerNames` — Wajib: Susunan string UTF-8, tidak lebih dari 100 string.

  Daftar nama crawler, mungkin nama yang dikembalikan oleh operasi `ListCrawlers`.

Respons

- `Crawlers` – Susunan objek Crawler.

  Daftar definisi crawler.
- `CrawlersNotFound` — Susunan string UTF-8, tidak lebih dari 100 string.

  Daftar nama crawler yang tidak ditemukan.

Kesalahan

- `InvalidInputException`
- `OperationTimeoutException`

ListCrawlers tindakan (Python: list_crawlers)

Mengambil nama semua sumber daya crawler di AWS akun ini, atau sumber daya dengan tag yang ditentukan. Operasi ini memungkinkan Anda melihat sumber daya yang tersedia di akun Anda, dan nama-namanya.

Operasi ini mengambil kolom `Tags` opsional, yang dapat Anda gunakan sebagai filter pada respon sehingga tag sumber daya dapat diambil sebagai sebuah grup. Jika Anda memilih untuk menggunakan pem-filter-an tag, maka hanya sumber daya dengan tag saja yang diambil.

Permintaan

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Ukuran maksimum daftar yang akan dikembalikan.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah permintaan kelanjutan.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Menentukan untuk mengembalikan hanya sumber daya ditandai saja.

Respons

- `CrawlerNames` — Susunan string UTF-8, tidak lebih dari 100 string.

  Nama dari semua crawler dalam akun, atau crawler dengan tag yang ditentukan.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika daftar yang dikembalikan tidak berisi metrik terakhir yang tersedia.

Kesalahan

- `OperationTimeoutException`

ListCrawls tindakan (Python: list_crawls)

Mengembalikan semua crawl dari crawler tertentu. Hanya mengembalikan crawl yang telah terjadi sejak tanggal peluncuran fitur riwayat crawler, dan hanya mempertahankan perayapan hingga 12 bulan. Perayapan yang lebih tua tidak akan dikembalikan.

Anda dapat menggunakan API ini untuk:

- Ambil semua crawl dari crawler tertentu.
- Ambil semua crawl crawler tertentu dalam hitungan terbatas.
- Ambil semua crawl crawler tertentu dalam rentang waktu tertentu.
- Ambil semua crawl crawler tertentu dengan status tertentu, ID crawl, atau nilai jam DPU.

Permintaan

- `CrawlerName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler yang menjalankan Anda ingin mengambil.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah hasil maksimum yang akan dikembalikan. Defaultnya adalah 20, dan maksimum adalah 100.
- `Filters` – Susunan objek CrawlsFilter.

  Memfilter crawl berdasarkan kriteria yang Anda tentukan dalam daftar `CrawlsFilter` objek.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Respons

- `Crawls` – Susunan objek CrawlerHistory.

  Daftar `CrawlerHistory` objek yang mewakili proses crawl yang memenuhi kriteria Anda.
- `NextToken` – String UTF-8.

Sebuah token kelanjutan untuk pemberian nomor halaman untuk daftar token yang ditampilkan, dikembalikan jika segmen saat ini dari daftar tersebut bukan yang terakhir.

Kesalahan

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`

API statistik kolom

API statistik kolom menjelaskan AWS Glue API untuk mengembalikan statistik pada kolom dalam tabel.

Jenis Data

- ColumnStatisticsTaskRun struktur
- ColumnStatisticsTaskRunningException struktur
- ColumnStatisticsTaskNotRunningException struktur
- ColumnStatisticsTaskStoppingException struktur

ColumnStatisticsTaskRun struktur

Objek yang menunjukkan detail statistik kolom yang dijalankan.

Bidang

- `CustomerId`String UTF-8, dengan panjang tidak lebih dari 12 byte.

  ID akun AWS.
- `ColumnStatisticsTaskRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi untuk tugas statistik kolom tertentu dijalankan.
- `DatabaseName` – String UTF-8.

Basis data tempat tabel berada.

- `TableName` – String UTF-8.

  Nama tabel tempat statistik kolom dihasilkan.

- `ColumnNameList` – Susunan string UTF-8.

  Daftar nama kolom. Jika tidak ada yang disediakan, semua nama kolom untuk tabel akan digunakan secara default.

- `CatalogID` — String id katalog, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat tabel berada. Jika tidak ada yang disediakan, ID akun AWS digunakan secara default.

- `Role` – String UTF-8.

  Peran IAM yang diasumsikan oleh layanan untuk menghasilkan statistik.

- `SampleSize`— Jumlah (ganda), tidak lebih dari 100.

  Persentase baris yang digunakan untuk menghasilkan statistik. Jika tidak ada yang disediakan, seluruh tabel akan digunakan untuk menghasilkan statistik.

- `SecurityConfiguration` — String UTF-8, dengan panjang tidak lebih dari 128 byte.

  Nama konfigurasi keamanan yang digunakan untuk mengenkripsi CloudWatch log untuk menjalankan tugas statistik kolom.

- `NumberOfWorkers` — Nomor (bilangan bulat), minimal 1.

  Jumlah pekerja yang digunakan untuk menghasilkan statistik kolom. Pekerjaan ini telah dikonfigurasi sebelumnya untuk skala otomatis hingga 25 instance.

- `WorkerType` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Jenis pekerja yang digunakan untuk menghasilkan statistik. Defaultnya adalah `g.1x`.

- `Status` – String UTF-8 (nilai yang valid: `STARTING` | `RUNNING` | `SUCCEEDED` | `FAILED` | `STOPPED`).

  Status tugas.

- `CreationTime` — Stempel waktu.

Waktu tugas ini dibuat.

- `LastUpdated` — Stempel waktu.

  Titik dalam waktu terakhir ketika tugas ini dimodifikasi.

- `StartTime` — Stempel waktu.

  Waktu mulai tugas.

- `EndTime` — Stempel waktu.

  Waktu akhir tugas.

- `ErrorMessage` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Pesan kesalahan untuk tugas.

- `DPUSeconds` — Nomor (ganda), tidak lebih dari Tidak Ada.

  Penggunaan DPU yang dihitung dalam hitungan detik untuk semua pekerja berskala otomatis.

ColumnStatisticsTaskRunningException struktur

Pengecualian dilemparkan saat Anda mencoba memulai pekerjaan lain saat menjalankan pekerjaan pembuatan statistik kolom.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

ColumnStatisticsTaskNotRunningException struktur

Pengecualian dilemparkan ketika Anda mencoba menghentikan tugas yang dijalankan ketika tidak ada tugas yang berjalan.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

ColumnStatisticsTaskStoppingException struktur

Pengecualian dilemparkan saat Anda mencoba menghentikan menjalankan tugas.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

Operasi

- StartColumnStatisticsTaskRun aksi (Python: start_column_statistics_task_run)
- GetColumnStatisticsTaskRun aksi (Python: get_column_statistics_task_run)
- GetColumnStatisticsTaskRuns tindakan (Python: get_column_statistics_task_runs)
- ListColumnStatisticsTaskRuns tindakan (Python: list_column_statistics_task_runs)
- StopColumnStatisticsTaskRun aksi (Python: stop_column_statistics_task_run)

StartColumnStatisticsTaskRun aksi (Python: start_column_statistics_task_run)

Memulai tugas statistik kolom yang dijalankan, untuk tabel dan kolom tertentu.

Permintaan

- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama basis data tempat tabel berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel untuk menghasilkan statistik.
- `ColumnNameList` – Susunan string UTF-8.

  Daftar nama kolom untuk menghasilkan statistik. Jika tidak ada yang disediakan, semua nama kolom untuk tabel akan digunakan secara default.
- `Role` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Peran IAM yang diasumsikan oleh layanan untuk menghasilkan statistik.

- `SampleSize`— Jumlah (ganda), tidak lebih dari 100.

  Persentase baris yang digunakan untuk menghasilkan statistik. Jika tidak ada yang disediakan, seluruh tabel akan digunakan untuk menghasilkan statistik.

- `CatalogID` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari Katalog Data tempat tabel berada. Jika tidak ada yang disediakan, ID akun AWS digunakan secara default.

- `SecurityConfiguration` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama konfigurasi keamanan yang digunakan untuk mengenkripsi CloudWatch log untuk menjalankan tugas statistik kolom.

Response

- `ColumnStatisticsTaskRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi tugas statistik kolom dijalankan.

Kesalahan

- `AccessDeniedException`
- `EntityNotFoundException`
- `ColumnStatisticsTaskRunningException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `InvalidInputException`

GetColumnStatisticsTaskRun aksi (Python: get_column_statistics_task_run)

Dapatkan metadata/informasi terkait untuk menjalankan tugas, dengan diberi ID task run.

Permintaan

- `ColumnStatisticsTaskRunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi untuk tugas statistik kolom tertentu dijalankan.

Response

- `ColumnStatisticsTaskRun` — Sebuah objek ColumnStatisticsTaskRun.

  Sebuah `ColumnStatisticsTaskRun` objek yang mewakili rincian statistik kolom berjalan.

Kesalahan

- `EntityNotFoundException`
- `OperationTimeoutException`
- `InvalidInputException`

GetColumnStatisticsTaskRuns tindakan (Python: get_column_statistics_task_runs)

Mengambil informasi tentang semua proses yang terkait dengan tabel yang ditentukan.

Permintaan

- `DatabaseName` – Wajib: String UTF-8.

  Nama basis data tempat tabel berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Ukuran maksimum respons.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Response

- `ColumnStatisticsTaskRuns` – Susunan objek ColumnStatisticsTaskRun.

  Daftar tugas statistik kolom berjalan.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika tidak semua tugas yang belum dikembalikan.

Kesalahan

- `OperationTimeoutException`

ListColumnStatisticsTaskRuns tindakan (Python: list_column_statistics_task_runs)

Daftar semua tugas yang dijalankan untuk akun tertentu.

Permintaan

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Ukuran maksimum respons.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Response

- `ColumnStatisticsTaskRunIds` — Susunan string UTF-8, tidak lebih dari 100 string.

  Sebuah daftar kolom statistik tugas menjalankan ID.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika tidak semua ID task run yang belum dikembalikan.

Kesalahan

- `OperationTimeoutException`

StopColumnStatisticsTaskRun aksi (Python: stop_column_statistics_task_run)

Menghentikan tugas yang dijalankan untuk tabel yang ditentukan.

Permintaan

- `DatabaseName` – Wajib: String UTF-8.

  Nama basis data tempat tabel berada.
- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel.

Response

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `ColumnStatisticsTaskNotRunningException`
- `ColumnStatisticsTaskStoppingException`
- `OperationTimeoutException`

API penjadwal perayap

Crawler scheduler API menjelaskan tipe data AWS Glue crawler, bersama dengan API untuk membuat, menghapus, memperbarui, dan mencantumkan crawler.

Jenis Data

- Struktur jadwal

Struktur jadwal

Sebuah objek penjadwalan menggunakan pernyataan `cron` untuk menjadwalkan sebuah peristiwa.

Bidang

- `ScheduleExpression` – String UTF-8.

  Sebuah ekspresi `cron` yang digunakan untuk menentukan jadwal (lihat Jadwal Berbasis Waktu untuk Tugas dan Crawler. Sebagai contoh, untuk menjalankan sesuatu setiap hari pada 12:15 UTC, Anda harus menentukan `cron(15 12 * * ? *)`.
- `State` – String UTF-8 (nilai yang valid: `SCHEDULED` | `NOT_SCHEDULED` | `TRANSITIONING`).

  Status jadwal.

Operasi

- UpdateCrawlerSchedule tindakan (Python: update_crawler_schedule)
- StartCrawlerSchedule tindakan (Python: start_crawler_schedule)
- StopCrawlerSchedule tindakan (Python: stop_crawler_schedule)

UpdateCrawlerSchedule tindakan (Python: update_crawler_schedule)

Memperbarui jadwal sebuah crawler dengan menggunakan ekspresi `cron`.

Permintaan

- `CrawlerName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler yang jadwalnya akan diperbarui.
- `Schedule` – String UTF-8.

  Ekspresi `cron` yang diperbarui yang digunakan untuk menentukan jadwal (lihat Jadwal Berbasis Waktu untuk Tugas dan Crawler. Sebagai contoh, untuk menjalankan sesuatu setiap hari pada 12:15 UTC, Anda harus menentukan `cron(15 12 * * ? *)`.

Response

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `VersionMismatchException`
- `SchedulerTransitioningException`
- `OperationTimeoutException`

StartCrawlerSchedule tindakan (Python: start_crawler_schedule)

Mengubah status jadwal crawler yang ditentukan untuk `SCHEDULED`, kecuali crawler sudah berjalan atau status jadwal sudah `SCHEDULED`.

Permintaan

- `CrawlerName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler yang akan dijadwal.

Response

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `SchedulerRunningException`
- `SchedulerTransitioningException`
- `NoScheduleException`
- `OperationTimeoutException`

StopCrawlerSchedule tindakan (Python: stop_crawler_schedule)

Menetapkan status jadwal dari crawler yang ditentukan untuk `NOT_SCHEDULED`, tapi tidak menghentikan crawler jika sudah berjalan.

Permintaan

- `CrawlerName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler yang status jadwalnya akan ditetapkan.

Response

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `SchedulerNotRunningException`
- `SchedulerTransitioningException`
- `OperationTimeoutException`

Pembuatan otomatis API Skrip ETL

API pembuatan skrip ETL menjelaskan tipe data dan API untuk menghasilkan skrip ETL di. AWS Glue

Jenis Data

- CodeGenNode struktur
- CodeGenNodeArg struktur
- CodeGenEdge struktur
- Struktur lokasi
- CatalogEntry struktur
- MappingEntry struktur

CodeGenNode struktur

Merepresentasikan simpul dalam grafik asiklik terarah (DAG)

Bidang

- `Id` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Identifier string pattern.

  Sebuah simpul pengenal yang unik dalam grafik simpul.

- `NodeType` – Wajib: String UTF-8.

  Jenis simpul yang ini.

- `Args` — Wajib: Susunan objek CodeGenNodeArg, tidak lebih dari 50 struktur.

  Properti simpul, dalam bentuk pasangan nama-nilai.

- `LineNumber` — Nomor (bilangan bulat).

  Nomor baris dari simpul.

CodeGenNodeArg struktur

Argumen atau properti dari sebuah simpul.

Bidang

- `Name` – Wajib: String UTF-8.

  Nama argumen atau properti.

- `Value` – Wajib: String UTF-8.

  Nilai argumen atau properti.

- `Param` – Boolean.

  BETUL jika nilai yang digunakan sebagai parameter.

CodeGenEdge struktur

Merepresentasikan edge direksional dalam sebuah grafik asiklik terarah (DAG).

Bidang

- `Source` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Identifier string pattern.

  ID dari simpul di mana edge dimulai.
- `Target` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Identifier string pattern.

  ID dari simpul di mana edge berakhir.
- `TargetParameter` – String UTF-8.

  Target dari edge.

Struktur lokasi

Lokasi sumber daya.

Bidang

- `Jdbc` — Susunan objek CodeGenNodeArg, tidak lebih dari 50 struktur.

  Lokasi JDBC.
- `S3` — Susunan objek CodeGenNodeArg, tidak lebih dari 50 struktur.

  Lokasi Amazon Simple Storage Service (Amazon S3).
- `DynamoDB` — Susunan objek CodeGenNodeArg, tidak lebih dari 50 struktur.

  Lokasi tabel Amazon DynamoDB.

CatalogEntry struktur

Menentukan definisi tabel dalam AWS Glue Data Catalog.

Bidang

- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Basis data di mana metadata tabel berada.

- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tabel yang dimaksud.

MappingEntry struktur

Mendefinisikan pemetaan.

Bidang

- `SourceTable` – String UTF-8.

  Nama tabel sumber.
- `SourcePath` – String UTF-8.

  Path sumber.
- `SourceType` – String UTF-8.

  Jenis sumber.
- `TargetTable` – String UTF-8.

  Tabel target.
- `TargetPath` – String UTF-8.

  Path target.
- `TargetType` – String UTF-8.

  Jenis target.

Operasi

- CreateScript tindakan (Python: create_script)
- GetDataflowGraph tindakan (Python: get_dataflow_graph)
- GetMapping tindakan (Python: get_mapping)
- GetPlan tindakan (Python: get_plan)

CreateScript tindakan (Python: create_script)

Mengubah grafik asiklik terarah (DAG) menjadi kode.

Permintaan

- `DagNodes` – Susunan objek CodeGenNode.

  Daftar simpul dalam DAG.
- `DagEdges` – Susunan objek CodeGenEdge.

  Daftar edge dalam DAG.
- `Language` – String UTF-8 (nilai yang valid: `PYTHON` | `SCALA`).

  Bahasa pemrograman dari kode yang dihasilkan dari DAG.

Response

- `PythonScript` – String UTF-8.

  Skrip Python yang dihasilkan dari DAG.
- `ScalaCode` – String UTF-8.

  Kode Scala yang dihasilkan dari DAG.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetDataflowGraph tindakan (Python: get_dataflow_graph)

Mengubah skrip Python menjadi sebuah grafik asiklik terarah (DAG).

Permintaan

- `PythonScript` – String UTF-8.

Skrip Python yang akan diubah.

Response

- `DagNodes` – Susunan objek CodeGenNode.

  Daftar simpul dalam DAG yang dihasilkan.
- `DagEdges` – Susunan objek CodeGenEdge.

  Daftar edge dalam DAG yang dihasilkan.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetMapping tindakan (Python: get_mapping)

Membuat pemetaan.

Permintaan

- `Source` — Wajib: Sebuah objek CatalogEntry.

  Menentukan tabel sumber.
- `Sinks` – Susunan objek CatalogEntry.

  Daftar tabel target.
- `Location` — Sebuah objek Lokasi.

  Parameter untuk pemetaan.

Response

- `Mapping` – Wajib: Susunan objek MappingEntry.

Daftar pemetaan untuk target yang ditentukan.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `EntityNotFoundException`

GetPlan tindakan (Python: get_plan)

Dapatkan kode untuk melakukan pemetaan tertentu.

Permintaan

- `Mapping` – Wajib: Susunan objek MappingEntry.

  Daftar pemetaan dari tabel sumber untuk tabel target.
- `Source` — Wajib: Sebuah objek CatalogEntry.

  Tabel sumber.
- `Sinks` – Susunan objek CatalogEntry.

  Tabel target.
- `Location` — Sebuah objek Lokasi.

  Parameter untuk pemetaan.
- `Language` – String UTF-8 (nilai yang valid: `PYTHON` | `SCALA`).

  Bahasa pemrograman kode untuk melakukan pemetaan.
- `AdditionalPlanOptionsMap` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

  Sebuah peta untuk menahan tambahan parameter nilai kunci opsional.

Saat ini, pasangan nilai kunci ini didukung:

- `inferSchema` — Menentukan apakah akan mengatur `inferSchema` ke BETUL atau SALAH untuk skrip default yang dihasilkan oleh tugas AWS Glue. Misalnya, untuk mengatur `inferSchema` ke BETUL, berikan pasangan nilai kunci berikut:

  ```
  --additional-plan-options-map '{"inferSchema":"true"}'
  ```

Response

- `PythonScript` – String UTF-8.

  Sebuah skrip Python untuk melakukan pemetaan.
- `ScalaCode` – String UTF-8.

  Kode Scala untuk melakukan pemetaan.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

API pekerjaan visual

Visual job API memungkinkan Anda membuat pekerjaan integrasi data dengan menggunakan AWS Glue API dari objek JSON yang mewakili konfigurasi visual suatu AWS Glue pekerjaan.

Daftar `CodeGenConfigurationNodes` disediakan untuk membuat atau memperbarui API pekerjaan untuk mendaftarkan DAG di AWS Glue Studio untuk pekerjaan yang dibuat dan menghasilkan kode terkait.

Jenis data

- CodeGenConfigurationNode struktur
- Struktur JDBC ConnectorOptions
- StreamingDataPreviewOptions struktur

- AthenaConnectorSource struktur
- Struktur JDBC ConnectorSource
- SparkConnectorSource struktur
- CatalogSource struktur
- Struktur MySQL CatalogSource
- Struktur CatalogSource PostgreSQL
- Struktur CatalogSource OracleSQL
- Struktur ServerCatalogSource MicrosoftSQL
- CatalogKinesisSource struktur
- DirectKinesisSource struktur
- KinesisStreamingSourceOptions struktur
- CatalogKafkaSource struktur
- DirectKafkaSource struktur
- KafkaStreamingSourceOptions struktur
- RedshiftSource struktur
- AmazonRedshiftSource struktur
- AmazonRedshiftNodeData struktur
- AmazonRedshiftAdvancedOption struktur
- Struktur opsi
- Struktur S3 CatalogSource
- Struktur S3 SourceAdditionalOptions
- Struktur S3 CsvSource
- Struktur DirectJDBCSource
- Struktur S3 DirectSourceAdditionalOptions
- Struktur S3 JsonSource
- Struktur S3 ParquetSource
- Struktur S3 DeltaSource
- Struktur S3 CatalogDeltaSource
- CatalogDeltaSource struktur

- Struktur S3 HudiSource
- Struktur S3 CatalogHudiSource
- CatalogHudiSource struktur
- Struktur DynamoDB CatalogSource
- RelationalCatalogSource struktur
- Struktur JDBC ConnectorTarget
- SparkConnectorTarget struktur
- BasicCatalogTarget struktur
- Struktur MySQL CatalogTarget
- Struktur CatalogTarget PostgreSQL
- Struktur CatalogTarget OracleSQL
- Struktur ServerCatalogTarget MicrosoftSQL
- RedshiftTarget struktur
- AmazonRedshiftTarget struktur
- UpsertRedshiftTargetOptions struktur
- Struktur S3 CatalogTarget
- Struktur S3 GlueParquetTarget
- CatalogSchemaChangePolicy struktur
- Struktur S3 DirectTarget
- Struktur S3 HudiCatalogTarget
- Struktur S3 HudiDirectTarget
- Struktur S3 DeltaCatalogTarget
- Struktur S3 DeltaDirectTarget
- DirectSchemaChangePolicy struktur
- ApplyMapping struktur
- Struktur pemetaan
- SelectFields struktur
- DropFields struktur
- RenameField struktur

- Struktur keran
- Bergabunglah dengan struktur
- JoinColumn struktur
- SplitFields struktur
- SelectFromCollection struktur
- FillMissingValues struktur
- Struktur filter
- FilterExpression struktur
- FilterValue struktur
- CustomCode struktur
- Struktur SparkSQL
- SqlAlias struktur
- DropNullFields struktur
- NullCheckBoxList struktur
- NullValueField struktur
- Struktur tipe data
- Gabungkan struktur
- Struktur serikat
- Struktur Piidetection
- Struktur agregat
- DropDuplicates struktur
- GovernedCatalogTarget struktur
- GovernedCatalogSource struktur
- AggregateOperation struktur
- GlueSchema struktur
- GlueStudioSchemaColumn struktur
- GlueStudioColumn struktur
- DynamicTransform struktur
- TransformConfigParameter struktur
- EvaluateDataQuality struktur

- Struktur DQ ResultsPublishingOptions
- Struktur DQ StopJobOnFailureOptions
- EvaluateDataQualityMultiFrame struktur
- Struktur resep
- RecipeReference struktur
- SnowflakeNodeData struktur
- SnowflakeSource struktur
- SnowflakeTarget struktur
- ConnectorDataSource struktur
- ConnectorDataTarget struktur

CodeGenConfigurationNode struktur

`CodeGenConfigurationNode`menghitung semua jenis Node yang valid. Satu dan hanya satu variabel anggotanya yang dapat diisi.

Bidang

- `AthenaConnectorSource` — Sebuah objek AthenaConnectorSource.

  Menentukan konektor ke sumber data Amazon Athena.
- `JDBCConnectorSource` — Sebuah objek JDBC ConnectorSource.

  Menentukan konektor ke sumber data JDBC.
- `SparkConnectorSource` — Sebuah objek SparkConnectorSource.

  Menentukan konektor ke sumber data Apache Spark.
- `CatalogSource` — Sebuah objek CatalogSource.

  Menentukan penyimpanan data dalam Katalog AWS Glue Data.
- `RedshiftSource` — Sebuah objek RedshiftSource.

  Menentukan penyimpanan data Amazon Redshift.
- `S3CatalogSource` — Sebuah objek S3 CatalogSource.

  Menentukan penyimpanan data Amazon S3 di Katalog Data AWS Glue .

- `S3CsvSource` — Sebuah objek S3 CsvSource.

  Menentukan penyimpanan data nilai yang dipisahkan perintah (CSV) yang disimpan di Amazon S3.
- `S3JsonSource` — Sebuah objek S3 JsonSource.

  Menentukan penyimpanan data JSON yang disimpan di Amazon S3.
- `S3ParquetSource` — Sebuah objek S3 ParquetSource.

  Menentukan penyimpanan data Apache Parquet yang disimpan di Amazon S3.
- `RelationalCatalogSource` — Sebuah objek RelationalCatalogSource.

  Menentukan penyimpanan data katalog relasional dalam Katalog AWS Glue Data.
- `DynamoDBCatalogSource` — Sebuah objek DynamoDB CatalogSource.

  Menentukan penyimpanan data DynamoDBC Catalog di Data Catalog. AWS Glue
- `JDBCConnectorTarget` — Sebuah objek JDBC ConnectorTarget.

  Menentukan target data yang menulis ke Amazon S3 di penyimpanan kolumnar Apache Parquet.
- `SparkConnectorTarget` — Sebuah objek SparkConnectorTarget.

  Menentukan target yang menggunakan konektor Apache Spark.
- `CatalogTarget` — Sebuah objek BasicCatalogTarget.

  Menentukan target yang menggunakan tabel AWS Glue Data Catalog.
- `RedshiftTarget` — Sebuah objek RedshiftTarget.

  Menentukan target yang menggunakan Amazon Redshift.
- `S3CatalogTarget` — Sebuah objek S3 CatalogTarget.

  Menentukan target data yang menulis ke Amazon S3 menggunakan Katalog Data AWS Glue .
- `S3GlueParquetTarget` — Sebuah objek S3 GlueParquetTarget.

  Menentukan target data yang menulis ke Amazon S3 di penyimpanan kolumnar Apache Parquet.
- `S3DirectTarget` — Sebuah objek S3 DirectTarget.

  Menentukan target data yang menulis ke Amazon S3.
- `ApplyMapping` — Sebuah objek ApplyMapping.

Menentukan transformasi yang memetakan kunci properti data dalam sumber data ke kunci properti data dalam target data. Anda dapat mengganti nama kunci, memodifikasi tipe data untuk kunci, dan memilih kunci mana yang akan dibuang dari set data.

- `SelectFields` — Sebuah objek SelectFields.

  Menentukan transformasi yang memilih kunci properti data yang ingin Anda simpan.

- `DropFields` — Sebuah objek DropFields.

  Menentukan transformasi yang memilih kunci properti data yang ingin Anda drop.

- `RenameField` — Sebuah objek RenameField.

  Menentukan transformasi yang mengganti nama kunci properti data tunggal.

- `Spigot` — Sebuah objek Spigot.

  Menentukan transformasi yang menulis sampel data ke bucket Amazon S3.

- `Join` — Sebuah objek Join.

  Menentukan transformasi yang menggabungkan dua dataset menjadi satu dataset menggunakan frase perbandingan pada kunci properti data tertentu. Anda dapat menggunakan join bagian dalam, luar, kiri, kanan, kiri semi, dan lawan kiri.

- `SplitFields` — Sebuah objek SplitFields.

  Menentukan transformasi yang membagi kunci properti data menjadi dua. `DynamicFrames` Outputnya adalah kumpulan`DynamicFrames`: satu dengan kunci properti data yang dipilih, dan satu dengan kunci properti data yang tersisa.

- `SelectFromCollection` — Sebuah objek SelectFromCollection.

  Menentukan transformasi yang memilih salah satu `DynamicFrame` dari koleksi. `DynamicFrames` Outputnya adalah yang dipilih `DynamicFrame`

- `FillMissingValues` — Sebuah objek FillMissingValues.

  Menentukan transformasi yang menempatkan catatan dalam dataset yang memiliki nilai hilang dan menambahkan bidang baru dengan nilai ditentukan oleh imputasi. Kumpulan data input digunakan untuk melatih model pembelajaran mesin yang menentukan nilai yang hilang seharusnya.

- `Filter` — Sebuah objek Filter.

  Menentukan transformasi yang membagi dataset menjadi dua, berdasarkan kondisi filter.

- `CustomCode` — Sebuah objek CustomCode.

  Menentukan transformasi yang menggunakan kode kustom yang Anda berikan untuk melakukan transformasi data. Outputnya adalah kumpulan DynamicFrames.

- `SparkSQL` — Sebuah objek SparkSQL.

  Menentukan transformasi di mana Anda memasukkan query SQL menggunakan sintaks Spark SQL untuk mengubah data. Outputnya adalah satu `DynamicFrame`.

- `DirectKinesisSource` — Sebuah objek DirectKinesisSource.

  Menentukan sumber data Amazon Kinesis langsung.

- `DirectKafkaSource` — Sebuah objek DirectKafkaSource.

  Menentukan toko data Apache Kafka.

- `CatalogKinesisSource` — Sebuah objek CatalogKinesisSource.

  Menentukan sumber data Kinesis dalam Katalog Data AWS Glue .

- `CatalogKafkaSource` — Sebuah objek CatalogKafkaSource.

  Menentukan penyimpanan data Apache Kafka dalam Katalog Data.

- `DropNullFields` — Sebuah objek DropNullFields.

  Menentukan transformasi yang menghapus kolom dari dataset jika semua nilai dalam kolom adalah 'null'. Secara default, AWS Glue Studio akan mengenali objek null, tetapi beberapa nilai seperti string kosong, string yang “null”, -1 integer atau placeholder lain seperti nol, tidak secara otomatis dikenali sebagai nol.

- `Merge` — Sebuah objek Gabungkan.

  Menentukan transformasi yang menggabungkan `DynamicFrame` dengan pementasan `DynamicFrame` berdasarkan kunci utama yang ditentukan untuk mengidentifikasi catatan. Catatan duplikat (catatan dengan kunci primer yang sama) tidak di-deduplikasi.

- `Union` — Sebuah objek Union.

  Menentukan transformasi yang menggabungkan baris dari dua atau lebih dataset menjadi hasil tunggal.

- `PIIDetection` — Sebuah objek Piideteksi.

  Menentukan transformasi yang mengidentifikasi, menghapus atau menutupi data PII.

- `Aggregate` — Sebuah objek Agregat.

  Menentukan transformasi yang mengelompokkan baris dengan bidang yang dipilih dan menghitung nilai agregat dengan fungsi tertentu.
- `DropDuplicates` — Sebuah objek DropDuplicates.

  Menentukan transformasi yang menghapus baris data berulang dari kumpulan data.
- `GovernedCatalogTarget` — Sebuah objek GovernedCatalogTarget.

  Menentukan target data yang menulis ke katalog yang diatur.
- `GovernedCatalogSource` — Sebuah objek GovernedCatalogSource.

  Menentukan sumber data dalam Katalog Data yang diatur.
- `MicrosoftSQLServerCatalogSource` — Sebuah objek MicrosoftSQL ServerCatalogSource.

  Menentukan sumber data server Microsoft SQL dalam Katalog AWS Glue Data.
- `MySQLCatalogSource` — Sebuah objek MySQL CatalogSource.

  Menentukan sumber data MySQL dalam Katalog Data. AWS Glue
- `OracleSQLCatalogSource` — Sebuah objek OracleSQL CatalogSource.

  Menentukan sumber data Oracle dalam Katalog AWS Glue Data.
- `PostgreSQLCatalogSource` — Sebuah objek PostgreSQL CatalogSource.

  Menentukan sumber data PostgressQL dalam Katalog Data. AWS Glue
- `MicrosoftSQLServerCatalogTarget` — Sebuah objek MicrosoftSQL ServerCatalogTarget.

  Menentukan target yang menggunakan Microsoft SQL.
- `MySQLCatalogTarget` — Sebuah objek MySQL CatalogTarget.

  Menentukan target yang menggunakan MySQL.
- `OracleSQLCatalogTarget` — Sebuah objek OracleSQL CatalogTarget.

  Menentukan target yang menggunakan Oracle SQL.
- `PostgreSQLCatalogTarget` — Sebuah objek PostgreSQL CatalogTarget.

  Menentukan target yang menggunakan Postgres SQL.
- `DynamicTransform` — Sebuah objek DynamicTransform.

Menentukan transformasi visual kustom yang dibuat oleh pengguna.

- `EvaluateDataQuality` — Sebuah objek EvaluateDataQuality.

  Menentukan kriteria evaluasi kualitas data Anda.

- `S3CatalogHudiSource` — Sebuah objek S3 CatalogHudiSource.

  Menentukan sumber data Hudi yang terdaftar di Katalog AWS Glue Data. Sumber data harus disimpan di Amazon S3.

- `CatalogHudiSource` — Sebuah objek CatalogHudiSource.

  Menentukan sumber data Hudi yang terdaftar di Katalog AWS Glue Data.

- `S3HudiSource` — Sebuah objek S3 HudiSource.

  Menentukan sumber data Hudi yang disimpan di. Amazon S3

- `S3HudiCatalogTarget` — Sebuah objek S3 HudiCatalogTarget.

  Menentukan target yang menulis ke sumber data Hudi di Katalog AWS Glue Data.

- `S3HudiDirectTarget` — Sebuah objek S3 HudiDirectTarget.

  Menentukan target yang menulis ke sumber data Hudi di. Amazon S3

- `S3CatalogDeltaSource` — Sebuah objek S3 CatalogDeltaSource.

  Menentukan sumber data Delta Lake yang terdaftar di Katalog AWS Glue Data. Sumber data harus disimpan di Amazon S3.

- `CatalogDeltaSource` — Sebuah objek CatalogDeltaSource.

  Menentukan sumber data Delta Lake yang terdaftar di Katalog AWS Glue Data.

- `S3DeltaSource` — Sebuah objek S3 DeltaSource.

  Menentukan sumber data Delta Lake yang disimpan di. Amazon S3

- `S3DeltaCatalogTarget` — Sebuah objek S3 DeltaCatalogTarget.

  Menentukan target yang menulis ke sumber data Delta Lake di Katalog AWS Glue Data.

- `S3DeltaDirectTarget` — Sebuah objek S3 DeltaDirectTarget.

  Menentukan target yang menulis ke sumber data Delta Lake di. Amazon S3

- `AmazonRedshiftSource` — Sebuah objek AmazonRedshiftSource.

Menentukan target yang menulis ke sumber data di Amazon Redshift.

- `AmazonRedshiftTarget` — Sebuah objek AmazonRedshiftTarget.

  Menentukan target yang menulis ke target data di Amazon Redshift.

- `EvaluateDataQualityMultiFrame` — Sebuah objek EvaluateDataQualityMultiFrame.

  Menentukan kriteria evaluasi kualitas data Anda. Memungkinkan beberapa input data dan mengembalikan koleksi Dynamic Frames.

- `Recipe` — Sebuah objek Resep.

  Menentukan simpul AWS Glue DataBrew resep.

- `SnowflakeSource` — Sebuah objek SnowflakeSource.

  Menentukan sumber data Snowflake.

- `SnowflakeTarget` — Sebuah objek SnowflakeTarget.

  Menentukan target yang menulis ke sumber data Snowflake.

- `ConnectorDataSource` — Sebuah objek ConnectorDataSource.

  Menentukan sumber yang dihasilkan dengan pilihan koneksi standar.

- `ConnectorDataTarget` — Sebuah objek ConnectorDataTarget.

  Menentukan target yang dihasilkan dengan pilihan koneksi standar.

Struktur JDBC ConnectorOptions

Opsi koneksi tambahan untuk konektor.

Bidang

- `FilterPredicate` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Klausul kondisi tambahan untuk memfilter data dari sumber. Sebagai contoh:

  ```
  BillingCity='Mountain View'
  ```

  Saat menggunakan kueri alih-alih nama tabel, Anda harus memvalidasi bahwa kueri berfungsi dengan yang ditentukan`filterPredicate`.

- `PartitionColumn` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama kolom integer yang digunakan untuk partisi. Opsi ini bekerja hanya ketika ia disertakan dengan `lowerBound`, `upperBound`, dan `numPartitions`. Pilihan ini bekerja dengan cara yang sama seperti pada pembaca Spark SQL JDBC.

- `LowerBound` — Jumlah (panjang), tidak lebih dari Tidak ada.

  Nilai minimum `partitionColumn` yang digunakan untuk memutuskan langkah partisi.

- `UpperBound` — Jumlah (panjang), tidak lebih dari Tidak ada.

  Nilai maksimum `partitionColumn` yang digunakan untuk memutuskan langkah partisi.

- `NumPartitions` — Jumlah (panjang), tidak lebih dari Tidak ada.

  Jumlah partisi. Nilai ini, bersama dengan `lowerBound` (inklusif) dan `upperBound` (eksklusif), membentuk langkah partisi untuk ekspresi klausul `WHERE` yang dihasilkan yang digunakan untuk membagi `partitionColumn`.

- `JobBookmarkKeys` – Susunan string UTF-8.

  Nama kunci bookmark pekerjaan untuk mengurutkan.

- `JobBookmarkKeysSortOrder` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan urutan urutan naik atau turun.

- `DataTypeMapping` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8 (nilai yang valid: `ARRAY` | | `BIGINT` | `BINARY` | `BIT` | `BLOB` | `BOOLEAN` | `CHAR` | `CLOB` | `DATALINK` | `DATE` | `DECIMAL` | `DISTINCT` | `DOUBLE` | `FLOAT` | `INTEGER` | `JAVA_OBJECT` | `LONGNVARCHAR` | `LONGVARBINARY` | `LONGVARCHAR` | `NCHAR` | `NCLOB` | `NULL` | | `NUMERIC` | `NVARCHAR` | `OTHER` | `REAL` | `REF` | `REF_CURSOR` | `ROWID` | `SMALLINT` | `SQLXML` | `STRUCT` | `TIME` | `TIME_WITH_TIMEZONE` `TIMESTAMP` `TIMESTAMP_WITH_TIMEZONE` | `TINYINT` | `VARBINARY` |`VARCHAR`).

  Setiap nilai adalah string UTF-8 (nilai valid: `DATE` | | `STRING` | `TIMESTAMP` | `INT` | `FLOAT` | `LONG` | `BIGDECIMAL` `BYTE` `SHORT` |`DOUBLE`).

  Pemetaan tipe data kustom yang membangun pemetaan dari tipe data JDBC ke tipe data. AWS Glue Misalnya, opsi `"dataTypeMapping":{"FLOAT":"STRING"}` memetakan bidang data tipe JDBC `FLOAT` ke dalam `String` tipe Java dengan memanggil `ResultSet.getString()` metode driver, dan menggunakannya untuk membangun catatan. AWS Glue Objek `ResultSet`

dilaksanakan oleh masing-masing driver, sehingga perilaku bersifat spesifik untuk driver yang Anda gunakan. Lihat dokumentasi untuk driver JDBC Anda untuk memahami bagaimana driver melakukan konversi.

StreamingDataPreviewOptions struktur

Menentukan pilihan yang terkait dengan pratinjau data untuk melihat sampel data Anda.

Bidang

- `PollingTime`— Jumlah (panjang), setidaknya 10.

  Waktu pemungutan suara dalam milidetik.
- `RecordPollingLimit`— Jumlah (panjang), setidaknya 1.

  Batas jumlah catatan yang disurvei.

AthenaConnectorSource struktur

Menentukan konektor ke sumber data Amazon Athena.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama dari sumber data.
- `ConnectionName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama koneksi yang dikaitkan dengan konektor.
- `ConnectorName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama konektor yang membantu mengakses penyimpanan data di Studio. AWS Glue
- `ConnectionType` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Jenis koneksi, seperti marketplace.athena atau custom.athena, menunjuk koneksi ke toko data Amazon Athena.
- `ConnectionTable` — String UTF-8, yang cocok dengan Custom string pattern #41.

  Nama tabel di sumber data.

- `SchemaName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama grup log Cloudwatch untuk dibaca. Misalnya, `/aws-glue/jobs/output`.
- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber Athena kustom.

Struktur JDBC ConnectorSource

Menentukan konektor ke sumber data JDBC.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama dari sumber data.
- `ConnectionName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama koneksi yang dikaitkan dengan konektor.
- `ConnectorName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama konektor yang membantu mengakses penyimpanan data di Studio. AWS Glue
- `ConnectionType` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Jenis koneksi, seperti marketplace.jdbc atau custom.jdbc, menunjuk koneksi ke penyimpanan data JDBC.
- `AdditionalOptions` — Sebuah objek JDBC ConnectorOptions.

  Opsi koneksi tambahan untuk konektor.
- `ConnectionTable` — String UTF-8, yang cocok dengan Custom string pattern #41.

  Nama tabel di sumber data.
- `Query` — String UTF-8, yang cocok dengan Custom string pattern #42.

  Tabel atau query SQL untuk mendapatkan data dari. Anda dapat menentukan salah satu dari `ConnectionTable` atau `query`, bukan keduanya.
- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber JDBC kustom.

SparkConnectorSource struktur

Menentukan konektor ke sumber data Apache Spark.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama dari sumber data.
- `ConnectionName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama koneksi yang dikaitkan dengan konektor.
- `ConnectorName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama konektor yang membantu mengakses penyimpanan data di Studio. AWS Glue
- `ConnectionType` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Jenis koneksi, seperti marketplace.spark atau custom.spark, menunjuk koneksi ke penyimpanan data Apache Spark.
- `AdditionalOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

  Opsi koneksi tambahan untuk konektor.
- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber percikan kustom.

CatalogSource struktur

Menentukan penyimpanan data dalam Katalog AWS Glue Data.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama penyimpanan data

- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.

- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.

Struktur MySQL CatalogSource

Menentukan sumber data MySQL dalam Katalog Data. AWS Glue

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama dari sumber data.

- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.

- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.

Struktur CatalogSource PostgreSQL

Menentukan sumber data PostgressQL dalam Katalog Data. AWS Glue

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama dari sumber data.

- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.

- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.

Struktur CatalogSource OracleSQL

Menentukan sumber data Oracle dalam Katalog AWS Glue Data.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama dari sumber data.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.

Struktur ServerCatalogSource MicrosoftSQL

Menentukan sumber data server Microsoft SQL dalam Katalog AWS Glue Data.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama dari sumber data.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.

CatalogKinesisSource struktur

Menentukan sumber data Kinesis dalam Katalog Data AWS Glue .

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

Nama dari sumber data.

- `WindowSize` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Jumlah waktu yang dihabiskan untuk memproses setiap batch mikro.
- `DetectSchema` – Boolean.

  Apakah akan secara otomatis menentukan skema dari data yang masuk.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.
- `StreamingOptions` — Sebuah objek KinesisStreamingSourceOptions.

  Opsi tambahan untuk sumber data streaming Kinesis.
- `DataPreviewOptions` — Sebuah objek StreamingDataPreviewOptions.

  Opsi tambahan untuk pratinjau data.

DirectKinesisSource struktur

Menentukan sumber data Amazon Kinesis langsung.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama dari sumber data.
- `WindowSize` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Jumlah waktu yang dihabiskan untuk memproses setiap batch mikro.
- `DetectSchema` – Boolean.

  Apakah akan secara otomatis menentukan skema dari data yang masuk.
- `StreamingOptions` — Sebuah objek KinesisStreamingSourceOptions.

  Opsi tambahan untuk sumber data streaming Kinesis.

- `DataPreviewOptions` — Sebuah objek StreamingDataPreviewOptions.

  Opsi tambahan untuk pratinjau data.

KinesisStreamingSourceOptions struktur

Opsi tambahan untuk sumber data streaming Amazon Kinesis.

Bidang

- `EndpointUrl` — String UTF-8, yang cocok dengan Custom string pattern #40.

  URL dari titik akhir Kinesis.
- `StreamName` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama aliran data Kinesis.
- `Classification` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Klasifikasi opsional.
- `Delimiter` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan karakter pembatas.
- `StartingPosition` – String UTF-8 (nilai yang valid: `latest="LATEST"` | `trim_horizon="TRIM_HORIZON"` | `earliest="EARLIEST"` | `timestamp="TIMESTAMP"`).

  Posisi awal dalam aliran data Kinesis untuk membaca data dari. Nilai yang mungkin adalah`"latest"`,, `"trim_horizon""earliest"`, atau string stempel waktu dalam format UTC dalam pola `yyyy-mm-ddTHH:MM:SSZ` (di mana `Z` mewakili zona waktu UTC offset dengan +/-. Misalnya: “2023-04-04T 08:00:00-04:00 “). Nilai default-nya adalah `"latest"`.

  Catatan: Menggunakan nilai yang merupakan string stempel waktu dalam format UTC untuk “startingPosition” hanya didukung untuk versi 4.0 atau yang lebih baru. AWS Glue
- `MaxFetchTimeInMs`— Jumlah (panjang), tidak lebih dari Tidak ada.

  Waktu maksimum yang dihabiskan untuk pelaksana pekerjaan untuk membaca catatan untuk batch saat ini dari aliran data Kinesis, ditentukan dalam milidetik (ms). Beberapa panggilan `GetRecords` API dapat dilakukan dalam waktu ini. Nilai default-nya adalah `1000`.
- `MaxFetchRecordsPerShard`— Jumlah (panjang), tidak lebih dari Tidak ada.

Jumlah maksimum catatan yang diambil per pecahan dalam aliran data Kinesis per mikrobatch. Catatan: Klien dapat melampaui batas ini jika pekerjaan streaming telah membaca catatan tambahan dari Kinesis (dalam panggilan get-records yang sama). Jika `MaxFetchRecordsPerShard` perlu ketat maka itu harus kelipatan`MaxRecordPerRead`. Nilai default-nya adalah `100000`.

- `MaxRecordPerRead`— Jumlah (panjang), tidak lebih dari Tidak ada.

  Jumlah maksimum record untuk diambil dari Kinesis data stream di setiap operasi GetRecords. Nilai default-nya adalah `10000`.

- `AddIdleTimeBetweenReads` – Boolean.

  Menambahkan penundaan waktu antara dua operasi GetRecords berturut-turut. Nilai default-nya adalah `"False"`. Opsi ini hanya dapat dikonfigurasi untuk Glue versi 2.0 dan di atasnya.

- `IdleTimeBetweenReadsInMs`— Jumlah (panjang), tidak lebih dari Tidak ada.

  Waktu tunda minimum antara dua operasi GetRecords berturut-turut, ditentukan dalam ms. Nilai default-nya adalah `1000`. Opsi ini hanya dapat dikonfigurasi untuk Glue versi 2.0 dan di atasnya.

- `DescribeShardInterval`— Jumlah (panjang), tidak lebih dari Tidak ada.

  Interval waktu minimum antara dua panggilan ListShards API untuk skrip Anda untuk mempertimbangkan resharding. Nilai default-nya adalah `1s`.

- `NumRetries` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Jumlah maksimum percobaan ulang untuk permintaan API Kinesis Data Streams. Nilai default-nya adalah `3`.

- `RetryIntervalMs`— Jumlah (panjang), tidak lebih dari Tidak ada.

  Periode waktu pendinginan (ditentukan dalam ms) sebelum mencoba kembali panggilan API Kinesis Data Streams. Nilai default-nya adalah `1000`.

- `MaxRetryIntervalMs`— Jumlah (panjang), tidak lebih dari Tidak ada.

  Periode waktu pendinginan maksimum (ditentukan dalam ms) antara dua percobaan ulang panggilan API Kinesis Data Streams. Nilai default-nya adalah `10000`.

- `AvoidEmptyBatches` – Boolean.

  Hindari membuat pekerjaan microbatch kosong dengan memeriksa data yang belum dibaca di aliran data Kinesis sebelum batch dimulai. Nilai default-nya adalah `"False"`.

- `StreamArn` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama Sumber Daya Amazon (ARN) dari aliran data Kinesis.

- `RoleArn` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama Sumber Daya Amazon (ARN) dari peran yang akan diambil menggunakan AWS Security Token Service (AWS STS). Peran ini harus memiliki izin untuk mendeskripsikan atau membaca operasi rekaman untuk aliran data Kinesis. Anda harus menggunakan parameter ini saat mengakses aliran data di akun yang berbeda. Digunakan bersama dengan`"awsSTSSessionName"`.

- `RoleSessionName` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Pengidentifikasi untuk sesi dengan asumsi peran menggunakan AWS STS. Anda harus menggunakan parameter ini saat mengakses aliran data di akun yang berbeda. Digunakan bersama dengan`"awsSTSRoleARN"`.

- `AddRecordTimestamp` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Ketika opsi ini diatur ke 'true', output data akan berisi kolom tambahan bernama “__src_timestamp” yang menunjukkan waktu ketika catatan terkait diterima oleh aliran. Nilai defaultnya adalah 'salah'. Opsi ini didukung di AWS Glue versi 4.0 atau yang lebih baru.

- `EmitConsumerLagMetrics` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Ketika opsi ini disetel ke 'true', untuk setiap batch, itu akan memancarkan metrik untuk durasi antara rekaman tertua yang diterima oleh aliran dan waktu tiba. AWS Glue CloudWatch Nama metriknya adalah “glue.driver.streaming. maxConsumerLagInMs”. Nilai defaultnya adalah 'salah'. Opsi ini didukung di AWS Glue versi 4.0 atau yang lebih baru.

- `StartingTimestamp` – String UTF-8.

  Stempel waktu catatan dalam aliran data Kinesis untuk mulai membaca data dari. Nilai yang mungkin adalah string stempel waktu dalam format UTC dari pola `yyyy-mm-ddTHH:MM:SSZ` (di mana Z mewakili offset zona waktu UTC dengan +/-. Misalnya: “2023-04-04T 08:00:00 + 08:00 “).

CatalogKafkaSource struktur

Menentukan penyimpanan data Apache Kafka dalam Katalog Data.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama penyimpanan data
- `WindowSize` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Jumlah waktu yang dihabiskan untuk memproses setiap batch mikro.
- `DetectSchema` – Boolean.

  Apakah akan secara otomatis menentukan skema dari data yang masuk.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.
- `StreamingOptions` — Sebuah objek KafkaStreamingSourceOptions.

  Menentukan opsi streaming.
- `DataPreviewOptions` — Sebuah objek StreamingDataPreviewOptions.

  Menentukan pilihan yang terkait dengan pratinjau data untuk melihat sampel data Anda.

DirectKafkaSource struktur

Menentukan toko data Apache Kafka.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama penyimpanan data
- `StreamingOptions` — Sebuah objek KafkaStreamingSourceOptions.

  Menentukan opsi streaming.
- `WindowSize` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Jumlah waktu yang dihabiskan untuk memproses setiap batch mikro.

- `DetectSchema` – Boolean.

  Apakah akan secara otomatis menentukan skema dari data yang masuk.

- `DataPreviewOptions` — Sebuah objek StreamingDataPreviewOptions.

  Menentukan pilihan yang terkait dengan pratinjau data untuk melihat sampel data Anda.

KafkaStreamingSourceOptions struktur

Opsi tambahan untuk streaming.

Bidang

- `BootstrapServers` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Daftar URL server bootstrap, misalnya, sebagaib`-1.vpc-test-2.o4q88o.c6.kafka.us-east-1.amazonaws.com:9094`. Opsi ini harus ditentukan dalam panggilan API atau didefinisikan dalam metadata tabel dalam Katalog Data.

- `SecurityProtocol` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Protokol yang digunakan untuk berkomunikasi dengan broker. Nilai yang mungkin adalah `"SSL"` atau `"PLAINTEXT"`.

- `ConnectionName` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama koneksi.

- `TopicName` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama topik seperti yang ditentukan dalam Apache Kafka. Anda harus menentukan setidaknya satu `"topicName"`, `"assign"` atau `"subscribePattern"`.

- `Assign` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Spesifik `TopicPartitions` untuk dikonsumsi. Anda harus menentukan setidaknya satu `"topicName"`, `"assign"` atau `"subscribePattern"`.

- `SubscribePattern` — String UTF-8, yang cocok dengan Custom string pattern #40.

  String regex Java yang mengidentifikasi daftar topik untuk berlangganan. Anda harus menentukan setidaknya satu `"topicName"`, `"assign"` atau `"subscribePattern"`.

- `Classification` — String UTF-8, yang cocok dengan Custom string pattern #40.

Klasifikasi opsional.

- `Delimiter` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan karakter pembatas.

- `StartingOffsets` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Posisi awal dalam topik Kafka untuk membaca data dari. Nilai yang mungkin adalah `"earliest"` atau `"latest"`. Nilai default-nya adalah `"latest"`.

- `EndingOffsets` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Titik akhir ketika kueri batch berakhir. Nilai yang mungkin adalah `"latest"` atau string JSON yang menentukan sebuah ending offset untuk setiap `TopicPartition`.

- `PollTimeoutMs`— Jumlah (panjang), tidak lebih dari Tidak ada.

  Batas waktu dalam milidetik untuk polling data dari Kafka di pelaksana pekerjaan Spark. Nilai default-nya adalah `512`.

- `NumRetries` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Berapa kali untuk mencoba lagi sebelum gagal mengambil offset Kafka. Nilai default-nya adalah `3`.

- `RetryIntervalMs`— Jumlah (panjang), tidak lebih dari Tidak ada.

  Waktu dalam milidetik untuk menunggu sebelum mencoba lagi untuk mengambil offset Kafka. Nilai default-nya adalah `10`.

- `MaxOffsetsPerTrigger`— Jumlah (panjang), tidak lebih dari Tidak ada.

  Batas laju pada jumlah maksimum offset yang diproses per interval pemicu. Jumlah total offset yang ditentukan dibagi secara proporsional di seluruh `topicPartitions` dengan volume yang berbeda. Nilai default-nya adalah nol, yang berarti bahwa konsumen membaca semua offset sampai diketahui offset terbaru.

- `MinPartitions` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Jumlah minimum partisi yang diinginkan untuk dibaca dari Kafka. Nilai default-nya adalah nol, yang berarti bahwa jumlah partisi spark sama dengan jumlah partisi Kafka.

- `IncludeHeaders` – Boolean.

  Apakah akan menyertakan header Kafka. Ketika opsi diatur ke “true”, output data akan berisi kolom tambahan bernama “glue_streaming_kafka_headers” dengan tipe. `Array[Struct(key:`

`String, value: String)]` Nilai defaultnya adalah “false”. Opsi ini hanya tersedia dalam AWS Glue versi 3.0 atau yang lebih baru.

- `AddRecordTimestamp` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Ketika opsi ini diatur ke 'true', output data akan berisi kolom tambahan bernama “__src_timestamp” yang menunjukkan waktu ketika catatan terkait diterima oleh topik. Nilai defaultnya adalah 'salah'. Opsi ini didukung di AWS Glue versi 4.0 atau yang lebih baru.

- `EmitConsumerLagMetrics` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Ketika opsi ini disetel ke 'true', untuk setiap batch, itu akan memancarkan metrik untuk durasi antara catatan tertua yang diterima oleh topik dan waktu tiba. AWS Glue CloudWatch Nama metriknya adalah “glue.driver.streaming. maxConsumerLagInMs”. Nilai defaultnya adalah 'salah'. Opsi ini didukung di AWS Glue versi 4.0 atau yang lebih baru.

- `StartingTimestamp` – String UTF-8.

  Stempel waktu catatan dalam topik Kafka untuk mulai membaca data dari. Nilai yang mungkin adalah string stempel waktu dalam format UTC dari pola `yyyy-mm-ddTHH:MM:SSZ` (di mana Z mewakili offset zona waktu UTC dengan +/-. Misalnya: “2023-04-04T 08:00:00 + 08:00 “).

  Hanya satu dari `StartingTimestamp` atau `StartingOffsets` harus ditetapkan.

RedshiftSource struktur

Menentukan penyimpanan data Amazon Redshift.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama penyimpanan data Amazon Redshift.

- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Database untuk dibaca.

- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Tabel database untuk dibaca.

- `RedshiftTmpDir` — String UTF-8, yang cocok dengan Custom string pattern #40.

Jalur Amazon S3 tempat data sementara dapat dipentaskan saat menyalin dari database.

- `TmpDirIAMRole` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Peran IAM dengan izin.

AmazonRedshiftSource struktur

Menentukan sumber Amazon Redshift.

Bidang

- `Name` — String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama sumber Amazon Redshift.
- `Data` — Sebuah objek AmazonRedshiftNodeData.

  Menentukan data node sumber Amazon Resshift.

AmazonRedshiftNodeData struktur

Menentukan node Amazon Redshift.

Bidang

- `AccessType` — String UTF-8, yang cocok dengan Custom string pattern #39.

  Jenis akses untuk koneksi Redshift. Bisa berupa koneksi langsung atau koneksi katalog.
- `SourceType` — String UTF-8, yang cocok dengan Custom string pattern #39.

  Jenis sumber untuk menentukan apakah tabel tertentu adalah sumber atau kueri kustom.
- `Connection` — Sebuah objek Opsi.

  AWS Glue Koneksi ke cluster Redshift.
- `Schema` — Sebuah objek Opsi.

  Nama skema Redshift saat bekerja dengan koneksi langsung.
- `Table` — Sebuah objek Opsi.

Nama tabel Redshift saat bekerja dengan koneksi langsung.

- `CatalogDatabase` — Sebuah objek Opsi.

  Nama database Katalog AWS Glue Data saat bekerja dengan katalog data.

- `CatalogTable` — Sebuah objek Opsi.

  Nama tabel Katalog AWS Glue Data saat bekerja dengan katalog data.

- `CatalogRedshiftSchema` – String UTF-8.

  Nama skema Redshift saat bekerja dengan katalog data.

- `CatalogRedshiftTable` – String UTF-8.

  Tabel database untuk dibaca.

- `TempDir` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Jalur Amazon S3 tempat data sementara dapat dipentaskan saat menyalin dari database.

- `IamRole` — Sebuah objek Opsi.

  Tidak wajib. Nama peran digunakan saat koneksi ke S3. Peran IAM tidak akan default ke peran pada pekerjaan saat dibiarkan kosong.

- `AdvancedOptions` – Susunan objek AmazonRedshiftAdvancedOption.

  Nilai opsional saat menghubungkan ke cluster Redshift.

- `SampleQuery` – String UTF-8.

  SQL digunakan untuk mengambil data dari sumber Redshift saat 'kueri'. SourceType

- `PreAction` – String UTF-8.

  SQL yang digunakan sebelum MERGE atau APPEND dengan upsert dijalankan.

- `PostAction` – String UTF-8.

  SQL yang digunakan sebelum MERGE atau APPEND dengan upsert dijalankan.

- `Action` – String UTF-8.

  Menentukan bagaimana menulis ke cluster Redshift akan terjadi.

- `TablePrefix` — String UTF-8, yang cocok dengan Custom string pattern #39.

Menentukan awalan untuk tabel.

- `Upsert` – Boolean.

  Tindakan yang digunakan pada Redshift tenggelam saat melakukan APPEND.

- `MergeAction` — String UTF-8, yang cocok dengan Custom string pattern #39.

  Tindakan yang digunakan saat menentukan bagaimana MERGE di wastafel Redshift akan ditangani.

- `MergeWhenMatched` — String UTF-8, yang cocok dengan Custom string pattern #39.

  Tindakan yang digunakan saat menentukan bagaimana MERGE di wastafel Redshift akan ditangani ketika rekor yang ada cocok dengan rekor baru.

- `MergeWhenNotMatched` — String UTF-8, yang cocok dengan Custom string pattern #39.

  Tindakan yang digunakan saat menentukan bagaimana MERGE di wastafel Redshift akan ditangani ketika rekaman yang ada tidak cocok dengan rekor baru.

- `MergeClause` – String UTF-8.

  SQL digunakan dalam penggabungan kustom untuk menangani catatan yang cocok.

- `CrawlerConnection` – String UTF-8.

  Menentukan nama koneksi yang terkait dengan tabel katalog yang digunakan.

- `TableSchema` – Susunan objek Opsi.

  Array output skema untuk node tertentu.

- `StagingTable` – String UTF-8.

  Nama tabel pementasan sementara yang digunakan saat melakukan MERGE atau APPEND dengan upsert.

- `SelectedColumns` – Susunan objek Opsi.

  Daftar nama kolom yang digunakan untuk menentukan record yang cocok saat melakukan MERGE atau APPEND dengan upsert.

AmazonRedshiftAdvancedOption struktur

Menentukan nilai opsional saat menghubungkan ke cluster Redshift.

Bidang

- `Key` – String UTF-8.

  Kunci untuk opsi koneksi tambahan.
- `Value` – String UTF-8.

  Nilai untuk opsi koneksi tambahan.

Struktur opsi

Menentukan nilai pilihan.

Bidang

- `Value` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan nilai opsi.
- `Label` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan label pilihan.
- `Description` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan deskripsi opsi.

Struktur S3 CatalogSource

Menentukan penyimpanan data Amazon S3 di Katalog Data AWS Glue .

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama penyimpanan data
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Database untuk dibaca.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Tabel database untuk dibaca.

- `PartitionPredicate` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Partisi yang memenuhi predikat ini dihapus. File dalam periode penyimpanan dalam partisi ini tidak dihapus. Diatur ke `""` — kosong secara default.
- `AdditionalOptions` — Sebuah objek S3 SourceAdditionalOptions.

  Menentukan pilihan koneksi tambahan.

Struktur S3 SourceAdditionalOptions

Menentukan opsi koneksi tambahan untuk penyimpanan data Amazon S3.

Bidang

- `BoundedSize` — Nomor (panjang).

  Menetapkan batas atas untuk ukuran target dataset dalam byte yang akan diproses.
- `BoundedFiles` — Nomor (panjang).

  Menetapkan batas atas untuk jumlah target file yang akan diproses.

Struktur S3 CsvSource

Menentukan penyimpanan data nilai yang dipisahkan perintah (CSV) yang disimpan di Amazon S3.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama penyimpanan data
- `Paths` – Wajib: Susunan string UTF-8.

  Daftar jalur Amazon S3 untuk dibaca.
- `CompressionType` – String UTF-8 (nilai yang valid: `gzip="GZIP"` | `bzip2="BZIP2"`).

  Menentukan bagaimana data dikompresi. Hal ini umumnya tidak diperlukan jika data memiliki sebuah ekstensi file standar. Nilai yang mungkin adalah `"gzip"` dan`"bzip"`).
- `Exclusions` – Susunan string UTF-8.

String yang berisi daftar JSON pola glob gaya Unix untuk dikecualikan. Misalnya, “[\" **.pdf\ "]” mengecualikan semua file PDF.

- `GroupSize` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Ukuran kelompok target dalam byte. Default-nya dihitung berdasarkan ukuran input data dan ukuran klaster Anda. Ketika ada kurang dari 50.000 file input, `"groupFiles"` harus diatur ke `"inPartition"` agar ini berlaku.

- `GroupFiles` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Pengelompokan file diaktifkan secara default ketika input berisi lebih dari 50.000 file. Untuk mengaktifkan pengelompokan dengan kurang dari 50.000 file, atur parameter ini ke “InPartition”. Untuk menonaktifkan pengelompokan dalam grup ketika ada lebih dari 50.000 file, tetapkan parameter ini ke `"none"`.

- `Recurse` – Boolean.

  Jika disetel ke true, secara rekursif membaca file di semua subdirektori di bawah jalur yang ditentukan.

- `MaxBand` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Opsi ini mengontrol durasi dalam milidetik setelah itu daftar s3 cenderung konsisten. File dengan cap waktu modifikasi yang termasuk dalam milidetik MaxBand terakhir dilacak secara khusus saat menggunakan JobBookmarks untuk memperhitungkan konsistensi Amazon S3. Sebagian besar pengguna tidak perlu mengatur opsi ini. Default-nya adalah 900000 milidetik, atau 15 menit.

- `MaxFilesInBand` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Opsi ini menentukan jumlah maksimum file untuk menyimpan dari detik MaxBand terakhir. Jika jumlah ini terlampaui, file tambahan akan dilewati dan hanya diproses dalam eksekusi tugas berikutnya.

- `AdditionalOptions` — Sebuah objek S3 DirectSourceAdditionalOptions.

  Menentukan pilihan koneksi tambahan.

- `Separator` – Wajib: String UTF-8 (nilai yang valid: `comma="COMMA"` | `ctrla="CTRLA"` | `pipe="PIPE"` | `semicolon="SEMICOLON"` | `tab="TAB"`).

  Menentukan karakter pembatas. Defaultnya adalah koma: “,”, tetapi karakter lain dapat ditentukan.

- `Escaper` — String UTF-8, yang cocok dengan Custom string pattern #41.

Menentukan karakter yang akan digunakan untuk melarikan diri. Pilihan ini hanya digunakan saat membaca file CSV saja. Nilai default-nya adalah none. Jika diaktifkan, karakter yang ada langsung setelahnya digunakan apa adanya, kecuali untuk satu set karakter escape yang sudah sangat dikenal (\n, \r, \t, dan \0).

- QuoteChar – Wajib: String UTF-8 (nilai yang valid: quote="QUOTE" | quillemet="QUILLEMET" | single_quote="SINGLE_QUOTE" | disabled="DISABLED").

  Menentukan karakter yang akan digunakan untuk mengutip. Default-nya adalah kutipan ganda: '"'. Atur ini ke -1 untuk menonaktifkan pengutipan seluruhnya.

- Multiline – Boolean.

  Nilai Boolean yang menentukan apakah catatan tunggal dapat menjangkau beberapa baris. Hal ini dapat terjadi ketika bidang berisi karakter baris baru yang dikutip. Anda harus menyetel opsi ini ke True jika ada catatan yang mencakup beberapa baris. Nilai default-nya adalah False, yang memungkinkan untuk pemecahan file yang lebih agresif selama penguraian.

- WithHeader – Boolean.

  Nilai Boolean yang menentukan apakah akan memperlakukan baris pertama sebagai header. Nilai default-nya adalah False.

- WriteHeader – Boolean.

  Sebuah nilai Boolean yang menentukan apakah untuk menulis header untuk output. Nilai default-nya adalah True.

- SkipFirst – Boolean.

  Nilai Boolean yang menentukan apakah akan melewati baris data pertama. Nilai default-nya adalah False.

- OptimizePerformance – Boolean.

  Nilai Boolean yang menentukan apakah akan menggunakan pembaca CSV SIMD canggih bersama dengan format memori kolumnar berbasis Apache Arrow. Hanya tersedia dalam AWS Glue versi 3.0.

- OutputSchemas – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber CSV S3.

Struktur DirectJDBCSource

Menentukan koneksi sumber JDBC langsung.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama koneksi sumber JDBC.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Database koneksi sumber JDBC.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Tabel koneksi sumber JDBC.
- `ConnectionName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama koneksi sumber JDBC.
- `ConnectionType` – Wajib: String UTF-8 (nilai yang valid: `sqlserver` | `mysql` | `oracle` | `postgresql` | `redshift`).

  Jenis koneksi sumber JDBC.
- `RedshiftTmpDir` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Direktori temp dari sumber JDBC Redshift.

Struktur S3 DirectSourceAdditionalOptions

Menentukan opsi koneksi tambahan untuk penyimpanan data Amazon S3.

Bidang

- `BoundedSize` — Nomor (panjang).

  Menetapkan batas atas untuk ukuran target dataset dalam byte yang akan diproses.
- `BoundedFiles` — Nomor (panjang).

  Menetapkan batas atas untuk jumlah target file yang akan diproses.
- `EnableSamplePath` – Boolean.

Menetapkan opsi untuk mengaktifkan jalur sampel.

- `SamplePath` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Jika diaktifkan, menentukan jalur sampel.

Struktur S3 JsonSource

Menentukan penyimpanan data JSON yang disimpan di Amazon S3.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama penyimpanan data

- `Paths` – Wajib: Susunan string UTF-8.

  Daftar jalur Amazon S3 untuk dibaca.

- `CompressionType` – String UTF-8 (nilai yang valid: `gzip="GZIP"` | `bzip2="BZIP2"`).

  Menentukan bagaimana data dikompresi. Hal ini umumnya tidak diperlukan jika data memiliki sebuah ekstensi file standar. Nilai yang mungkin adalah `"gzip"` dan`"bzip"`).

- `Exclusions` – Susunan string UTF-8.

  String yang berisi daftar JSON pola glob gaya Unix untuk dikecualikan. Misalnya, “[\" **.pdf\ "]” mengecualikan semua file PDF.

- `GroupSize` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Ukuran kelompok target dalam byte. Default-nya dihitung berdasarkan ukuran input data dan ukuran klaster Anda. Ketika ada kurang dari 50.000 file input, `"groupFiles"` harus diatur ke `"inPartition"` agar ini berlaku.

- `GroupFiles` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Pengelompokan file diaktifkan secara default ketika input berisi lebih dari 50.000 file. Untuk mengaktifkan pengelompokan dengan kurang dari 50.000 file, atur parameter ini ke “InPartition”. Untuk menonaktifkan pengelompokan dalam grup ketika ada lebih dari 50.000 file, tetapkan parameter ini ke `"none"`.

- `Recurse` – Boolean.

Jika disetel ke true, secara rekursif membaca file di semua subdirektori di bawah jalur yang ditentukan.

- `MaxBand` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Opsi ini mengontrol durasi dalam milidetik setelah itu daftar s3 cenderung konsisten. File dengan cap waktu modifikasi yang termasuk dalam milidetik MaxBand terakhir dilacak secara khusus saat menggunakan JobBookmarks untuk memperhitungkan konsistensi Amazon S3. Sebagian besar pengguna tidak perlu mengatur opsi ini. Default-nya adalah 900000 milidetik, atau 15 menit.

- `MaxFilesInBand` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Opsi ini menentukan jumlah maksimum file untuk menyimpan dari detik MaxBand terakhir. Jika jumlah ini terlampaui, file tambahan akan dilewati dan hanya diproses dalam eksekusi tugas berikutnya.

- `AdditionalOptions` — Sebuah objek S3 DirectSourceAdditionalOptions.

  Menentukan pilihan koneksi tambahan.

- `JsonPath` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Sebuah JsonPath string yang mendefinisikan data JSON.

- `Multiline` – Boolean.

  Nilai Boolean yang menentukan apakah catatan tunggal dapat menjangkau beberapa baris. Hal ini dapat terjadi ketika bidang berisi karakter baris baru yang dikutip. Anda harus menyetel opsi ini ke True jika ada catatan yang mencakup beberapa baris. Nilai default-nya adalah `False`, yang memungkinkan untuk pemecahan file yang lebih agresif selama penguraian.

- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber S3 JSON.

Struktur S3 ParquetSource

Menentukan penyimpanan data Apache Parquet yang disimpan di Amazon S3.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

Nama penyimpanan data

- `Paths` – Wajib: Susunan string UTF-8.

  Daftar jalur Amazon S3 untuk dibaca.

- `CompressionType` – String UTF-8 (nilai yang valid: `snappy="SNAPPY"` | `lzo="LZO"` | `gzip="GZIP"` | `uncompressed="UNCOMPRESSED"` | `none="NONE"`).

  Menentukan bagaimana data dikompresi. Hal ini umumnya tidak diperlukan jika data memiliki sebuah ekstensi file standar. Nilai yang mungkin adalah `"gzip"` dan`"bzip"`).

- `Exclusions` – Susunan string UTF-8.

  String yang berisi daftar JSON pola glob gaya Unix untuk dikecualikan. Misalnya, “[\" **.pdf\ "]” mengecualikan semua file PDF.

- `GroupSize` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Ukuran kelompok target dalam byte. Default-nya dihitung berdasarkan ukuran input data dan ukuran klaster Anda. Ketika ada kurang dari 50.000 file input, `"groupFiles"` harus diatur ke `"inPartition"` agar ini berlaku.

- `GroupFiles` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Pengelompokan file diaktifkan secara default ketika input berisi lebih dari 50.000 file. Untuk mengaktifkan pengelompokan dengan kurang dari 50.000 file, atur parameter ini ke “InPartition”. Untuk menonaktifkan pengelompokan dalam grup ketika ada lebih dari 50.000 file, tetapkan parameter ini ke `"none"`.

- `Recurse` – Boolean.

  Jika disetel ke true, secara rekursif membaca file di semua subdirektori di bawah jalur yang ditentukan.

- `MaxBand` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Opsi ini mengontrol durasi dalam milidetik setelah itu daftar s3 cenderung konsisten. File dengan cap waktu modifikasi yang termasuk dalam milidetik MaxBand terakhir dilacak secara khusus saat menggunakan JobBookmarks untuk memperhitungkan konsistensi Amazon S3. Sebagian besar pengguna tidak perlu mengatur opsi ini. Default-nya adalah 900000 milidetik, atau 15 menit.

- `MaxFilesInBand` — Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

Opsi ini menentukan jumlah maksimum file untuk menyimpan dari detik MaxBand terakhir. Jika jumlah ini terlampaui, file tambahan akan dilewati dan hanya diproses dalam eksekusi tugas berikutnya.

- `AdditionalOptions` — Sebuah objek S3 DirectSourceAdditionalOptions.

  Menentukan pilihan koneksi tambahan.

- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber Parket S3.

Struktur S3 DeltaSource

Menentukan sumber data Delta Lake yang disimpan di. Amazon S3

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama sumber Danau Delta.

- `Paths` – Wajib: Susunan string UTF-8.

  Daftar jalur Amazon S3 untuk dibaca.

- `AdditionalDeltaOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

  Menentukan pilihan koneksi tambahan.

- `AdditionalOptions` — Sebuah objek S3 DirectSourceAdditionalOptions.

  Menentukan pilihan tambahan untuk konektor.

- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber Delta Lake.

Struktur S3 CatalogDeltaSource

Menentukan sumber data Delta Lake yang terdaftar di Katalog AWS Glue Data. Sumber data harus disimpan di Amazon S3.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama sumber data Danau Delta.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.
- `AdditionalDeltaOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

  Menentukan pilihan koneksi tambahan.
- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber Delta Lake.

CatalogDeltaSource struktur

Menentukan sumber data Delta Lake yang terdaftar di Katalog AWS Glue Data.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama sumber data Danau Delta.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.

- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.
- `AdditionalDeltaOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

  Menentukan pilihan koneksi tambahan.
- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber Delta Lake.

Struktur S3 HudiSource

Menentukan sumber data Hudi yang disimpan di. Amazon S3

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama sumber Hudi.
- `Paths` – Wajib: Susunan string UTF-8.

  Daftar jalur Amazon S3 untuk dibaca.
- `AdditionalHudiOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

  Menentukan pilihan koneksi tambahan.
- `AdditionalOptions` — Sebuah objek S3 DirectSourceAdditionalOptions.

  Menentukan pilihan tambahan untuk konektor.
- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber Hudi.

Struktur S3 CatalogHudiSource

Menentukan sumber data Hudi yang terdaftar di Katalog AWS Glue Data. Sumber data Hudi harus disimpan di Amazon S3.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama sumber data Hudi.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.
- `AdditionalHudiOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

  Menentukan pilihan koneksi tambahan.
- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber Hudi.

CatalogHudiSource struktur

Menentukan sumber data Hudi yang terdaftar di Katalog AWS Glue Data.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama sumber data Hudi.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.

- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.
- `AdditionalHudiOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

  Menentukan pilihan koneksi tambahan.
- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber Hudi.

Struktur DynamoDB CatalogSource

Menentukan sumber data DynamoDB dalam Katalog Data. AWS Glue

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama dari sumber data.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.

RelationalCatalogSource struktur

Menentukan sumber data database Relasional dalam Katalog AWS Glue Data.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama dari sumber data.

- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk dibaca.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk dibaca.

Struktur JDBC ConnectorTarget

Menentukan target data yang menulis ke Amazon S3 di penyimpanan kolumnar Apache Parquet.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.
- `ConnectionName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama koneksi yang dikaitkan dengan konektor.
- `ConnectionTable` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #41.

  Nama tabel dalam target data.
- `ConnectorName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama konektor yang akan digunakan.
- `ConnectionType` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Jenis koneksi, seperti marketplace.jdbc atau custom.jdbc, menunjuk koneksi ke target data JDBC.
- `AdditionalOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

  Opsi koneksi tambahan untuk konektor.
- `OutputSchemas` – Susunan objek GlueSchema.

Menentukan skema data untuk target JDBC.

SparkConnectorTarget struktur

Menentukan target yang menggunakan konektor Apache Spark.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.
- `ConnectionName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama koneksi untuk konektor Apache Spark.
- `ConnectorName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama konektor Apache Spark.
- `ConnectionType` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Jenis koneksi, seperti marketplace.spark atau custom.spark, menunjuk koneksi ke penyimpanan data Apache Spark.
- `AdditionalOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

  Opsi koneksi tambahan untuk konektor.
- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk target percikan kustom.

BasicCatalogTarget struktur

Menentukan target yang menggunakan tabel AWS Glue Data Catalog.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data Anda.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Database yang berisi tabel yang ingin Anda gunakan sebagai target. Basis data ini harus sudah ada dalam Katalog Data.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Tabel yang mendefinisikan skema data output Anda. Tabel ini sudah harus ada dalam Katalog Data.

Struktur MySQL CatalogTarget

Menentukan target yang menggunakan MySQL.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk menulis.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk menulis ke.

Struktur CatalogTarget PostgreSQL

Menentukan target yang menggunakan Postgres SQL.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.

- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk menulis.

- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk menulis ke.

Struktur CatalogTarget OracleSQL

Menentukan target yang menggunakan Oracle SQL.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.

- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk menulis.

- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk menulis ke.

Struktur ServerCatalogTarget MicrosoftSQL

Menentukan target yang menggunakan Microsoft SQL.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk menulis.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk menulis ke.

RedshiftTarget struktur

Menentukan target yang menggunakan Amazon Redshift.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk menulis.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk menulis ke.
- `RedshiftTmpDir` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Jalur Amazon S3 tempat data sementara dapat dipentaskan saat menyalin dari database.
- `TmpDirIAMRole` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Peran IAM dengan izin.

- `UpsertRedshiftOptions` — Sebuah objek UpsertRedshiftTargetOptions.

  Kumpulan opsi untuk mengonfigurasi operasi upsert saat menulis ke target Redshift.

AmazonRedshiftTarget struktur

Menentukan target Amazon Redshift.

Bidang

- `Name` — String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target Amazon Redshift.
- `Data` — Sebuah objek AmazonRedshiftNodeData.

  Menentukan data node target Amazon Redshift.
- `Inputs`— Array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.

UpsertRedshiftTargetOptions struktur

Opsi untuk mengonfigurasi operasi upsert saat menulis ke target Redshift.

Bidang

- `TableLocation` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Lokasi fisik tabel Redshift.
- `ConnectionName` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama koneksi yang digunakan untuk menulis ke Redshift.
- `UpsertKeys` – Susunan string UTF-8.

  Kunci yang digunakan untuk menentukan apakah akan melakukan pembaruan atau menyisipkan.

Struktur S3 CatalogTarget

Menentukan target data yang menulis ke Amazon S3 menggunakan Katalog Data AWS Glue .

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.
- `PartitionKeys` – Susunan string UTF-8.

  Menentukan partisi asli menggunakan urutan kunci.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk menulis ke.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk menulis.
- `SchemaChangePolicy` — Sebuah objek CatalogSchemaChangePolicy.

  Kebijakan yang menentukan perilaku update untuk crawler.

Struktur S3 GlueParquetTarget

Menentukan target data yang menulis ke Amazon S3 di penyimpanan kolumnar Apache Parquet.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.
- `PartitionKeys` – Susunan string UTF-8.

  Menentukan partisi asli menggunakan urutan kunci.
- `Path` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Satu jalur Amazon S3 untuk menulis.

- `Compression` – String UTF-8 (nilai yang valid: `snappy="SNAPPY"` | `lzo="LZO"` | `gzip="GZIP"` | `uncompressed="UNCOMPRESSED"` | `none="NONE"`).

  Menentukan bagaimana data dikompresi. Hal ini umumnya tidak diperlukan jika data memiliki sebuah ekstensi file standar. Nilai yang mungkin adalah `"gzip"` dan`"bzip"`).
- `SchemaChangePolicy` — Sebuah objek DirectSchemaChangePolicy.

  Kebijakan yang menentukan perilaku update untuk crawler.

CatalogSchemaChangePolicy struktur

Kebijakan yang menentukan perilaku update untuk crawler.

Bidang

- `EnableUpdateCatalog` – Boolean.

  Apakah akan menggunakan perilaku pembaruan yang ditentukan saat crawler menemukan skema yang diubah.
- `UpdateBehavior` – String UTF-8 (nilai yang valid: `UPDATE_IN_DATABASE` | `LOG`).

  Perilaku pembaruan ketika crawler menemukan skema yang berubah.

Struktur S3 DirectTarget

Menentukan target data yang menulis ke Amazon S3.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.
- `PartitionKeys` – Susunan string UTF-8.

  Menentukan partisi asli menggunakan urutan kunci.
- `Path` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

Satu jalur Amazon S3 untuk menulis.

- `Compression` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan bagaimana data dikompresi. Hal ini umumnya tidak diperlukan jika data memiliki sebuah ekstensi file standar. Nilai yang mungkin adalah `"gzip"` dan`"bzip"`).

- `Format` – Wajib: String UTF-8 (nilai yang valid: `json="JSON"` | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | `delta="DELTA"`).

  Menentukan format output data untuk target.

- `SchemaChangePolicy` — Sebuah objek DirectSchemaChangePolicy.

  Kebijakan yang menentukan perilaku update untuk crawler.

Struktur S3 HudiCatalogTarget

Menentukan target yang menulis ke sumber data Hudi di Katalog AWS Glue Data.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.

- `PartitionKeys` – Susunan string UTF-8.

  Menentukan partisi asli menggunakan urutan kunci.

- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk menulis ke.

- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk menulis.

- `AdditionalOptions` – Wajib: Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

Menentukan pilihan koneksi tambahan untuk konektor.

- `SchemaChangePolicy` — Sebuah objek CatalogSchemaChangePolicy.

  Kebijakan yang menentukan perilaku update untuk crawler.

Struktur S3 HudiDirectTarget

Menentukan target yang menulis ke sumber data Hudi di. Amazon S3

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.

- `Path` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Jalur Amazon S3 dari sumber data Hudi Anda untuk menulis.

- `Compression` – Wajib: String UTF-8 (nilai yang valid: `gzip="GZIP"` | `lzo="LZO"` | `uncompressed="UNCOMPRESSED"` | `snappy="SNAPPY"`).

  Menentukan bagaimana data dikompresi. Hal ini umumnya tidak diperlukan jika data memiliki sebuah ekstensi file standar. Nilai yang mungkin adalah `"gzip"` dan`"bzip"`).

- `PartitionKeys` – Susunan string UTF-8.

  Menentukan partisi asli menggunakan urutan kunci.

- `Format` – Wajib: String UTF-8 (nilai yang valid: `json="JSON"` | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | `delta="DELTA"`).

  Menentukan format output data untuk target.

- `AdditionalOptions` – Wajib: Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

Menentukan pilihan koneksi tambahan untuk konektor.

- `SchemaChangePolicy` — Sebuah objek DirectSchemaChangePolicy.

  Kebijakan yang menentukan perilaku update untuk crawler.

Struktur S3 DeltaCatalogTarget

Menentukan target yang menulis ke sumber data Delta Lake di Katalog AWS Glue Data.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.

- `PartitionKeys` – Susunan string UTF-8.

  Menentukan partisi asli menggunakan urutan kunci.

- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk menulis ke.

- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk menulis.

- `AdditionalOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

  Menentukan pilihan koneksi tambahan untuk konektor.

- `SchemaChangePolicy` — Sebuah objek CatalogSchemaChangePolicy.

  Kebijakan yang menentukan perilaku update untuk crawler.

Struktur S3 DeltaDirectTarget

Menentukan target yang menulis ke sumber data Delta Lake di. Amazon S3

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.

- `PartitionKeys` – Susunan string UTF-8.

  Menentukan partisi asli menggunakan urutan kunci.

- `Path` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Jalur Amazon S3 dari sumber data Delta Lake Anda untuk menulis.

- `Compression` – Wajib: String UTF-8 (nilai yang valid: `uncompressed="UNCOMPRESSED"` | `snappy="SNAPPY"`).

  Menentukan bagaimana data dikompresi. Hal ini umumnya tidak diperlukan jika data memiliki sebuah ekstensi file standar. Nilai yang mungkin adalah `"gzip"` dan`"bzip"`).

- `Format` – Wajib: String UTF-8 (nilai yang valid: `json="JSON"` | `csv="CSV"` | `avro="AVRO"` | `orc="ORC"` | `parquet="PARQUET"` | `hudi="HUDI"` | `delta="DELTA"`).

  Menentukan format output data untuk target.

- `AdditionalOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

  Menentukan pilihan koneksi tambahan untuk konektor.

- `SchemaChangePolicy` — Sebuah objek DirectSchemaChangePolicy.

  Kebijakan yang menentukan perilaku update untuk crawler.

DirectSchemaChangePolicy struktur

Kebijakan yang menentukan perilaku update untuk crawler.

Bidang

- `EnableUpdateCatalog` – Boolean.

  Apakah akan menggunakan perilaku pembaruan yang ditentukan saat crawler menemukan skema yang diubah.

- `UpdateBehavior` – String UTF-8 (nilai yang valid: `UPDATE_IN_DATABASE` | `LOG`).

  Perilaku pembaruan ketika crawler menemukan skema yang berubah.

- `Table` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan tabel dalam database yang berlaku kebijakan perubahan skema.

- `Database` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan database yang berlaku untuk kebijakan perubahan skema.

ApplyMapping struktur

Menentukan transformasi yang memetakan kunci properti data dalam sumber data ke kunci properti data dalam target data. Anda dapat mengganti nama kunci, memodifikasi tipe data untuk kunci, dan memilih kunci mana yang akan dibuang dari set data.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input data diidentifikasi oleh nama node mereka.

- `Mapping` – Wajib: Susunan objek Pemetaan.

  Menentukan pemetaan kunci properti data dalam sumber data untuk kunci properti data dalam target data.

Struktur pemetaan

Menentukan pemetaan kunci properti data.

Bidang

- `ToKey` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Setelah menerapkan pemetaan, apa nama kolom seharusnya. Bisa sama dengan`FromPath`.

- `FromPath` – Susunan string UTF-8.

  Tabel atau kolom yang akan dimodifikasi.

- `FromType` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Jenis data yang akan dimodifikasi.

- `ToType` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Tipe data yang akan dimodifikasi untuk data.

- `Dropped` – Boolean.

  Jika benar, maka kolom dihapus.

- `Children` – Susunan objek Pemetaan.

  Hanya berlaku untuk struktur data bersarang. Jika Anda ingin mengubah struktur induk, tetapi juga salah satu anaknya, Anda dapat mengisi strucutre data ini. Hal ini juga`Mapping`, tetapi `FromPath` akan menjadi induk `FromPath` ditambah `FromPath` dari struktur ini.

  Untuk bagian anak-anak, misalkan Anda memiliki struktur:

  ```
  { "FromPath": "OuterStructure", "ToKey": "OuterStructure", "ToType":
  "Struct", "Dropped": false, "Chidlren": [{ "FromPath": "inner", "ToKey":
  "inner", "ToType": "Double", "Dropped": false, }] }
  ```

  Anda dapat menentukan `Mapping` yang terlihat seperti:

  ```
  { "FromPath": "OuterStructure", "ToKey": "OuterStructure", "ToType":
  "Struct", "Dropped": false, "Chidlren": [{ "FromPath": "inner", "ToKey":
  "inner", "ToType": "Double", "Dropped": false, }] }
  ```

SelectFields struktur

Menentukan transformasi yang memilih kunci properti data yang ingin Anda simpan.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input data diidentifikasi oleh nama node mereka.
- `Paths` – Wajib: Susunan string UTF-8.

  Sebuah jalur JSON ke variabel dalam struktur data.

DropFields struktur

Menentukan transformasi yang memilih kunci properti data yang ingin Anda drop.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input data diidentifikasi oleh nama node mereka.
- `Paths` – Wajib: Susunan string UTF-8.

  Sebuah jalur JSON ke variabel dalam struktur data.

RenameField struktur

Menentukan transformasi yang mengganti nama kunci properti data tunggal.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

Nama node transformasi.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input data diidentifikasi oleh nama node mereka.

- `SourcePath` – Wajib: Susunan string UTF-8.

  Sebuah jalur JSON ke variabel dalam struktur data untuk sumber data.

- `TargetPath` – Wajib: Susunan string UTF-8.

  Sebuah jalur JSON ke variabel dalam struktur data untuk data target.

Struktur keran

Menentukan transformasi yang menulis sampel data ke bucket Amazon S3.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input data diidentifikasi oleh nama node mereka.

- `Path` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Jalur di Amazon S3 tempat transformasi akan menulis subset catatan dari kumpulan data ke file JSON di bucket Amazon S3.

- `Topk`— Angka (bilangan bulat), tidak lebih dari 100.

  Menentukan sejumlah catatan untuk menulis mulai dari awal dataset.

- `Prob`— Jumlah (ganda), tidak lebih dari 1.

  Probabilitas (nilai desimal dengan nilai maksimum 1) untuk memilih catatan yang diberikan. Nilai 1 menunjukkan bahwa setiap baris yang dibaca dari kumpulan data harus dimasukkan dalam output sampel.

Bergabunglah dengan struktur

Menentukan transformasi yang menggabungkan dua dataset menjadi satu dataset menggunakan frase perbandingan pada kunci properti data tertentu. Anda dapat menggunakan join bagian dalam, luar, kiri, kanan, kiri semi, dan lawan kiri.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 2 atau lebih dari 2 string.

  Input data diidentifikasi oleh nama node mereka.

- `JoinType`- Diperlukan: UTF-8 string (nilai valid: `equijoin="EQUIJOIN"` | | `left="LEFT"` | `right="RIGHT"` `outer="OUTER"` `leftsemi="LEFT_SEMI"` |`leftanti="LEFT_ANTI"`).

  Menentukan jenis bergabung yang akan dilakukan pada dataset.

- `Columns`— Diperlukan: Sebuah array JoinColumn objek, tidak kurang dari 2 atau lebih dari 2 struktur.

  Daftar dua kolom yang akan digabungkan.

JoinColumn struktur

Menentukan kolom yang akan bergabung.

Bidang

- `From` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Kolom yang akan bergabung.

- `Keys` – Wajib: Susunan string UTF-8.

  Kunci kolom yang akan digabungkan.

SplitFields struktur

Menentukan transformasi yang membagi kunci properti data menjadi dua. `DynamicFrames` Outputnya adalah kumpulan`DynamicFrames`: satu dengan kunci properti data yang dipilih, dan satu dengan kunci properti data yang tersisa.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input data diidentifikasi oleh nama node mereka.

- `Paths` – Wajib: Susunan string UTF-8.

  Sebuah jalur JSON ke variabel dalam struktur data.

SelectFromCollection struktur

Menentukan transformasi yang memilih salah satu `DynamicFrame` dari koleksi. `DynamicFrames` Outputnya adalah yang dipilih `DynamicFrame`

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input data diidentifikasi oleh nama node mereka.

- `Index` — Wajib: Nomor (bilangan bulat), tidak lebih dari Tidak Ada.

  Indeks DynamicFrame untuk yang akan dipilih.

FillMissingValues struktur

Menentukan transformasi yang menempatkan catatan dalam dataset yang memiliki nilai hilang dan menambahkan bidang baru dengan nilai ditentukan oleh imputasi. Kumpulan data input digunakan untuk melatih model pembelajaran mesin yang menentukan nilai yang hilang seharusnya.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input data diidentifikasi oleh nama node mereka.
- `ImputedPath` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Sebuah jalur JSON ke variabel dalam struktur data untuk dataset yang diperhitungkan.
- `FilledPath` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Sebuah jalur JSON ke variabel dalam struktur data untuk dataset yang diisi.

Struktur filter

Menentukan transformasi yang membagi dataset menjadi dua, berdasarkan kondisi filter.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input data diidentifikasi oleh nama node mereka.
- `LogicalOperator` – Wajib: String UTF-8 (nilai yang valid: `AND` | `OR`).

  Operator digunakan untuk memfilter baris dengan membandingkan nilai kunci dengan nilai tertentu.
- `Filters` – Wajib: Susunan objek FilterExpression.

  Menentukan ekspresi filter.

FilterExpression struktur

Menentukan ekspresi filter.

Bidang

- `Operation` – Wajib: String UTF-8 (nilai yang valid: `EQ` | `LT` | `GT` | `LTE` | `GTE` | `REGEX` | `ISNULL`).

  Jenis operasi yang harus dilakukan dalam ekspresi.
- `Negated` – Boolean.

  Apakah ekspresi itu akan dinegasikan.
- `Values` – Wajib: Susunan objek FilterValue.

  Daftar nilai filter.

FilterValue struktur

Merupakan entri tunggal dalam daftar nilai untuk a`FilterExpression`.

Bidang

- `Type` – Wajib: String UTF-8 (nilai yang valid: `COLUMNEXTRACTED` | `CONSTANT`).

  Jenis nilai filter.
- `Value` – Wajib: Susunan string UTF-8.

  Nilai yang akan dikaitkan.

CustomCode struktur

Menentukan transformasi yang menggunakan kode kustom yang Anda berikan untuk melakukan transformasi data. Outputnya adalah kumpulan DynamicFrames.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.

- `Inputs` — Wajib: Susunan string UTF-8, setidaknya 1 string.

  Input data diidentifikasi oleh nama node mereka.
- `Code` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #35.

  Kode kustom yang digunakan untuk melakukan transformasi data.
- `ClassName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama didefinisikan untuk kelas node kode kustom.
- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk mengubah kode kustom.

Struktur SparkSQL

Menentukan transformasi di mana Anda memasukkan query SQL menggunakan sintaks Spark SQL untuk mengubah data. Outputnya adalah satu `DynamicFrame`.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.
- `Inputs` — Wajib: Susunan string UTF-8, setidaknya 1 string.

  Input data diidentifikasi oleh nama node mereka. Anda dapat mengaitkan nama tabel dengan setiap node input untuk digunakan dalam query SQL. Nama yang Anda pilih harus memenuhi batasan penamaan Spark SQL.
- `SqlQuery` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #42.

  Kueri SQL yang harus menggunakan sintaks Spark SQL dan mengembalikan satu set data.
- `SqlAliases` – Wajib: Susunan objek SqlAlias.

  Daftar alias. Sebuah alias memungkinkan Anda untuk menentukan nama apa yang akan digunakan dalam SQL untuk input yang diberikan. Misalnya, Anda memiliki sumber data bernama "". MyDataSource Jika Anda menentukan `From` sebagai MyDataSource, dan `Alias` as SqlName, maka di SQL Anda dapat melakukan:

  ```
  select * from SqlName
  ```

dan itu mendapat data dari MyDataSource.

- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk transformasi SparkSQL.

SqlAlias struktur

Merupakan entri tunggal dalam daftar nilai untuk`SqlAliases`.

Bidang

- `From` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #39.

  Sebuah tabel, atau kolom dalam tabel.

- `Alias` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #41.

  Nama sementara yang diberikan ke tabel, atau kolom dalam tabel.

DropNullFields struktur

Menentukan transformasi yang menghapus kolom dari dataset jika semua nilai dalam kolom adalah 'null'. Secara default, AWS Glue Studio akan mengenali objek null, tetapi beberapa nilai seperti string kosong, string yang “null”, -1 integer atau placeholder lain seperti nol, tidak secara otomatis dikenali sebagai nol.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input data diidentifikasi oleh nama node mereka.

- `NullCheckBoxList` — Sebuah objek NullCheckBoxList.

  Struktur yang mewakili apakah nilai-nilai tertentu diakui sebagai nilai nol untuk dihapus.

- `NullTextList` — Susunan objek NullValueField, tidak lebih dari 50 struktur.

Struktur yang menentukan daftar NullValueField struktur yang mewakili nilai null kustom seperti nol atau nilai lain yang digunakan sebagai placeholder null yang unik untuk dataset.

`DropNullFields`Transformasi menghapus nilai null khusus hanya jika nilai placeholder null dan tipe data cocok dengan data.

NullCheckBoxList struktur

Merupakan apakah nilai-nilai tertentu diakui sebagai nilai nol untuk dihapus.

Bidang

- `IsEmpty` – Boolean.

  Menentukan bahwa string kosong dianggap sebagai nilai null.

- `IsNullString` – Boolean.

  Menentukan bahwa nilai yang mengeja kata 'null' dianggap sebagai nilai null.

- `IsNegOne` – Boolean.

  Menentukan bahwa nilai integer -1 dianggap sebagai nilai null.

NullValueField struktur

Merupakan nilai null kustom seperti nol atau nilai lain yang digunakan sebagai placeholder null yang unik untuk kumpulan data.

Bidang

- `Value` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nilai placeholder null.

- `Datatype` — Wajib: Sebuah objek JenisData.

  Jenis data dari nilai.

Struktur tipe data

Struktur yang mewakili tipe data dari nilai.

Bidang

- `Id` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #39.

  Jenis data dari nilai.
- `Label` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #39.

  Label yang ditetapkan ke tipe data.

Gabungkan struktur

Menentukan transformasi yang menggabungkan `DynamicFrame` dengan pementasan `DynamicFrame` berdasarkan kunci utama yang ditentukan untuk mengidentifikasi catatan. Catatan duplikat (catatan dengan kunci primer yang sama) tidak di-deduplikasi.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 2 atau lebih dari 2 string.

  Input data diidentifikasi oleh nama node mereka.
- `Source` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #39.

  Sumber `DynamicFrame` yang akan digabung dengan `DynamicFrame` pementasan.
- `PrimaryKeys` – Wajib: Susunan string UTF-8.

  Daftar bidang kunci utama untuk mencocokkan catatan dari sumber dan pementasan frame dinamis.

Struktur serikat

Menentukan transformasi yang menggabungkan baris dari dua atau lebih dataset menjadi hasil tunggal.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 2 atau lebih dari 2 string.

  Input ID node ke transformasi.
- `UnionType` – Wajib: String UTF-8 (nilai yang valid: `ALL` | `DISTINCT`).

  Menunjukkan jenis transformasi Union.

  Tentukan `ALL` untuk menggabungkan semua baris dari sumber data ke hasil DynamicFrame. Serikat yang dihasilkan tidak menghapus baris duplikat.

  Tentukan `DISTINCT` untuk menghapus baris duplikat dalam hasil DynamicFrame.

Struktur Piidetection

Menentukan transformasi yang mengidentifikasi, menghapus atau menutupi data PII.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input ID node ke transformasi.
- `PiiType` – Wajib: String UTF-8 (nilai yang valid: `RowAudit` | `RowMasking` | `ColumnAudit` | `ColumnMasking`).

  Menunjukkan jenis transformasi PIIDetection.
- `EntityTypesToDetect` – Wajib: Susunan string UTF-8.

  Menunjukkan jenis entitas yang akan diidentifikasi oleh transformasi PIIdetection sebagai data PII.

  Entitas tipe PII meliputi: PERSON_NAME, DATE, USA_SNN, EMAIL, USA_ITIN, USA_PASSPORT_NUMBER, PHONE_NUMBER, BANK_ACCOUNT, IP_ADDRESS, MAC_ADDRESS, USA_CPT_CODE, USA_HCPCS_CODE, USA_NATIONAL_DRUG_CODE,

USA_MEDICARE_BENEFICIARY_IDENTIFIER, USA_HEALTH_INSURANCE_CLAIM_NUMBER, CREDIT_CARD, USA_NATIONAL_PROVIDER_IDENTIFIER, USA_DEA_NUMBER, USA_DRIVING_LICENSE

- `OutputColumnName` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Menunjukkan nama kolom keluaran yang akan berisi jenis entitas apa pun yang terdeteksi di baris itu.

- `SampleFraction`— Jumlah (ganda), tidak lebih dari 1.

  Menunjukkan fraksi data yang akan diambil sampel saat memindai entitas PII.

- `ThresholdFraction`— Jumlah (ganda), tidak lebih dari 1.

  Menunjukkan fraksi data yang harus dipenuhi agar kolom diidentifikasi sebagai data PII.

- `MaskValue`— String UTF-8, panjangnya tidak lebih dari 256 byte, cocok dengan file. Custom string pattern #37

  Menunjukkan nilai yang akan menggantikan entitas yang terdeteksi.

Struktur agregat

Menentukan transformasi yang mengelompokkan baris dengan bidang yang dipilih dan menghitung nilai agregat dengan fungsi tertentu.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Menentukan bidang dan baris untuk digunakan sebagai input untuk transformasi agregat.

- `Groups` – Wajib: Susunan string UTF-8.

  Menentukan bidang untuk kelompok oleh.

- `Aggs`— Wajib: Sebuah array AggregateOperation objek, tidak kurang dari 1 atau lebih dari 30 struktur.

  Menentukan fungsi agregat yang akan dilakukan pada bidang tertentu.

DropDuplicates struktur

Menentukan transformasi yang menghapus baris data berulang dari kumpulan data.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node transformasi.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input data diidentifikasi oleh nama node mereka.
- `Columns` – Susunan string UTF-8.

  Nama kolom yang akan digabungkan atau dihapus jika diulang.

GovernedCatalogTarget struktur

Menentukan target data yang menulis ke Amazon S3 menggunakan Katalog Data AWS Glue .

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target data.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.
- `PartitionKeys` – Susunan string UTF-8.

  Menentukan partisi asli menggunakan urutan kunci.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama tabel dalam database untuk menulis ke.
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Nama database untuk menulis.
- `SchemaChangePolicy` — Sebuah objek CatalogSchemaChangePolicy.

  Kebijakan yang menentukan perilaku pembaruan untuk katalog yang diatur.

GovernedCatalogSource struktur

Menentukan penyimpanan data dalam Katalog AWS Glue Data yang diatur.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama penyimpanan data
- `Database` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Database untuk dibaca.
- `Table` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Tabel database untuk dibaca.
- `PartitionPredicate` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Partisi yang memenuhi predikat ini dihapus. File dalam periode penyimpanan dalam partisi ini tidak dihapus. Diatur ke `""` — kosong secara default.
- `AdditionalOptions` — Sebuah objek S3 SourceAdditionalOptions.

  Menentukan pilihan koneksi tambahan.

AggregateOperation struktur

Menentukan set parameter yang diperlukan untuk melakukan agregasi dalam transformasi agregat.

Bidang

- `Column` – Wajib: Susunan string UTF-8.

  Menentukan kolom pada kumpulan data di mana fungsi agregasi akan diterapkan.
- `AggFunc`— Diperlukan: UTF-8 string (nilai valid: `avg` | | `countDistinct` | `count` | `first` | `last` | `kurtosis` | `max` | `min` | `skewness` | `stddev_samp` | | `stddev_pop` | `sum` `sumDistinct` `var_samp` |`var_pop`).

  Menentukan fungsi agregasi untuk menerapkan.

  Fungsi agregasi yang mungkin meliputi: avg countDistinct, count, first, last, kurtosis, max, min, skewness, stddev_samp, stddev_pop, sum, sumDistinct, var_samp, var_pop

GlueSchema struktur

Menentukan skema yang ditetapkan pengguna ketika skema tidak dapat ditentukan oleh. AWS Glue

Bidang

- `Columns` – Susunan objek GlueStudioSchemaColumn.

  Menentukan definisi kolom yang membentuk AWS Glue skema.

GlueStudioSchemaColumn struktur

Menentukan satu kolom dalam definisi AWS Glue skema.

Bidang

- `Name`- Wajib: UTF-8 string, tidak lebih dari 1024 byte panjang, cocok dengan. Single-line string pattern

  Nama kolom dalam skema AWS Glue Studio.
- `Type` — String UTF-8, sepanjang tidak lebih dari 131072, yang cocok dengan Single-line string pattern.

  Jenis sarang untuk kolom ini dalam skema AWS Glue Studio.

GlueStudioColumn struktur

Menentukan satu kolom di AWS Glue Studio.

Bidang

- `Key` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #41.

  Kunci kolom di AWS Glue Studio.
- `FullPath` – Wajib: Susunan string UTF-8.

  TURL lengkap kolom di AWS Glue Studio.
- `Type`— Diperlukan: UTF-8 string (nilai valid: `array="ARRAY"` | `bigint="BIGINT"` | `bigint array="BIGINT_ARRAY"` | `binary="BINARY"` | `binary array="BINARY_ARRAY"` |

```
boolean="BOOLEAN"|boolean array="BOOLEAN_ARRAY"|byte="BYTE"|byte
array="BYTE_ARRAY"|char="CHAR"||char array="CHAR_ARRAY"|choice="CHOICE"
|choice array="CHOICE_ARRAY"|date="DATE"|date array="DATE_ARRAY"
|decimal="DECIMAL"|decimal array="DECIMAL_ARRAY"|double="DOUBLE"
|double array="DOUBLE_ARRAY"|enum="ENUM"|enum array="ENUM_ARRAY"|
float="FLOAT"|float array="FLOAT_ARRAY"|int="INT"||int array="INT_ARRAY"
|interval="INTERVAL"|interval array="INTERVAL_ARRAY"|long="LONG"
|long array="LONG_ARRAY"|object="OBJECT"|short="SHORT"|short
array="SHORT_ARRAY" smallint="SMALLINT" smallint array="SMALLINT_ARRAY"|
string="STRING"|string array="STRING_ARRAY"|timestamp="TIMESTAMP"
|timestamp array="TIMESTAMP_ARRAY"|tinyint="TINYINT" tinyint
array="TINYINT_ARRAY"|varchar="VARCHAR"|varchar array="VARCHAR_ARRAY"|
null="NULL"|unknown="UNKNOWN"|unknown array="UNKNOWN_ARRAY").
```

tJenis kolom di AWS Glue Studio.

- `Children`— Sebuah array dari struktur.

  TAnak-anak dari kolom induk di AWS Glue Studio.

DynamicTransform struktur

Menentukan set parameter yang diperlukan untuk melakukan transformasi dinamis.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan nama transformasi dinamis.
- `TransformName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan nama transformasi dinamis seperti yang muncul di editor visual AWS Glue Studio.
- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Menentukan input untuk transformasi dinamis yang diperlukan.
- `Parameters` – Susunan objek TransformConfigParameter.

  Menentukan parameter transformasi dinamis.
- `FunctionName` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

Menentukan nama fungsi transformasi dinamis.

- `Path` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan jalur sumber transformasi dinamis dan file konfigurasi.

- `Version` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Bidang ini tidak digunakan dan akan usang dalam rilis future.

- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk transformasi dinamis.

TransformConfigParameter struktur

Menentukan parameter dalam file konfigurasi dari transformasi dinamis.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan nama parameter dalam file konfigurasi dari transformasi dinamis.

- `Type` – Wajib: String UTF-8 (nilai yang valid: `str="STR"` | `int="INT"` | `float="FLOAT"` | `complex="COMPLEX"` | `bool="BOOL"` | `list="LIST"` | `null="NULL"`).

  Menentukan jenis parameter dalam file konfigurasi dari transformasi dinamis.

- `ValidationRule` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan aturan validasi dalam file konfigurasi dari transformasi dinamis.

- `ValidationMessage` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Menentukan pesan validasi dalam file konfigurasi dari transformasi dinamis.

- `Value` – Susunan string UTF-8.

  Menentukan nilai parameter dalam file konfigurasi dari transformasi dinamis.

- `ListType` – String UTF-8 (nilai yang valid: `str="STR"` | `int="INT"` | `float="FLOAT"` | `complex="COMPLEX"` | `bool="BOOL"` | `list="LIST"` | `null="NULL"`).

  Menentukan jenis daftar parameter dalam file konfigurasi dari transformasi dinamis.

- `IsOptional` – Boolean.

  Menentukan apakah parameter opsional atau tidak dalam file konfigurasi dari transformasi dinamis.

EvaluateDataQuality struktur

Menentukan kriteria evaluasi kualitas data Anda.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama evaluasi kualitas data.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Input dari evaluasi kualitas data Anda.

- `Ruleset`- Wajib: UTF-8 string, tidak kurang dari 1 atau lebih dari 65536 byte panjang, cocok dengan. Custom string pattern #38

  Aturan untuk evaluasi kualitas data Anda.

- `Output` – String UTF-8 (nilai yang valid: `PrimaryInput` | `EvaluationResults`).

  Output dari evaluasi kualitas data Anda.

- `PublishingOptions` — Sebuah objek DQ ResultsPublishingOptions.

  Opsi untuk mengonfigurasi bagaimana hasil Anda dipublikasikan.

- `StopJobOnFailureOptions` — Sebuah objek DQ StopJobOnFailureOptions.

  Opsi untuk mengonfigurasi bagaimana pekerjaan Anda akan berhenti jika evaluasi kualitas data Anda gagal.

Struktur DQ ResultsPublishingOptions

Opsi untuk mengonfigurasi bagaimana hasil evaluasi kualitas data Anda dipublikasikan.

Bidang

- `EvaluationContext` — String UTF-8, yang cocok dengan Custom string pattern #39.

Konteks evaluasi.

- `ResultsS3Prefix` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Awalan Amazon S3 diawali dengan hasil.
- `CloudWatchMetricsEnabled` – Boolean.

  Aktifkan metrik untuk hasil kualitas data Anda.
- `ResultsPublishingEnabled` – Boolean.

  Aktifkan penerbitan untuk hasil kualitas data Anda.

Struktur DQ StopJobOnFailureOptions

Opsi untuk mengonfigurasi bagaimana pekerjaan Anda akan berhenti jika evaluasi kualitas data Anda gagal.

Bidang

- `StopJobOnFailureTiming` – String UTF-8 (nilai yang valid: `Immediate` | `AfterDataLoad`).

  Kapan harus berhenti bekerja jika evaluasi kualitas data Anda gagal. Pilihannya Segera atau AfterDataLoad.

EvaluateDataQualityMultiFrame struktur

Menentukan kriteria evaluasi kualitas data Anda.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama evaluasi kualitas data.
- `Inputs` — Wajib: Susunan string UTF-8, setidaknya 1 string.

  Input dari evaluasi kualitas data Anda. Input pertama dalam daftar ini adalah sumber data primer.
- `AdditionalDataSources` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #43

Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

Alias semua sumber data kecuali primer.

- `Ruleset`- Wajib: UTF-8 string, tidak kurang dari 1 atau lebih dari 65536 byte panjang, cocok dengan. Custom string pattern #38

  Aturan untuk evaluasi kualitas data Anda.

- `PublishingOptions` — Sebuah objek DQ ResultsPublishingOptions.

  Opsi untuk mengonfigurasi bagaimana hasil Anda dipublikasikan.

- `AdditionalOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8 (nilai valid: `performanceTuning.caching="CacheOption"` |`observations.scope="ObservationsOption"`).

  Setiap nilai adalah string UTF-8.

  Opsi untuk mengonfigurasi perilaku runtime transformasi.

- `StopJobOnFailureOptions` — Sebuah objek DQ StopJobOnFailureOptions.

  Opsi untuk mengonfigurasi bagaimana pekerjaan Anda akan berhenti jika evaluasi kualitas data Anda gagal.

Struktur resep

Node AWS Glue Studio yang menggunakan AWS Glue DataBrew resep dalam AWS Glue pekerjaan.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node AWS Glue Studio.

- `Inputs`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke node resep, diidentifikasi oleh id.

- `RecipeReference` — Wajib: Sebuah objek RecipeReference.

  Referensi ke DataBrew resep yang digunakan oleh node.

RecipeReference struktur

Referensi ke AWS Glue DataBrew resep.

Bidang

- `RecipeArn` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  ARN resepnya. DataBrew
- `RecipeVersion`- Diperlukan: string UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 16 byte.

  DataBrew Resepnya. RecipeVersion

SnowflakeNodeData struktur

Menentukan konfigurasi untuk node Snowflake di Studio. AWS Glue

Bidang

- `SourceType` — String UTF-8, yang cocok dengan Custom string pattern #39.

  Menentukan bagaimana data diambil ditentukan. Nilai-nilai yang valid: `"table"`, `"query"`.
- `Connection` — Sebuah objek Opsi.

  Menentukan Koneksi Katalog AWS Glue Data ke titik akhir Snowflake.
- `Schema` – String UTF-8.

  Menentukan skema database Snowflake untuk node Anda untuk digunakan.
- `Table` – String UTF-8.

  Menentukan tabel Snowflake untuk node Anda untuk digunakan.
- `Database` – String UTF-8.

  Menentukan database Snowflake untuk node Anda untuk digunakan.
- `TempDir` — String UTF-8, yang cocok dengan Custom string pattern #40.

  Saat ini tidak digunakan.
- `IamRole` — Sebuah objek Opsi.

  Saat ini tidak digunakan.

- `AdditionalOptions` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #40

  Setiap nilai adalah string UTF-8, cocok dengan. Custom string pattern #40

  Menentukan opsi tambahan diteruskan ke konektor Snowflake. Jika opsi ditentukan di tempat lain di node ini, ini akan diutamakan.
- `SampleQuery` – String UTF-8.

  String SQL digunakan untuk mengambil data dengan sourcetype. `query`
- `PreAction` – String UTF-8.

  String SQL berjalan sebelum konektor Snowflake melakukan tindakan standarnya.
- `PostAction` – String UTF-8.

  String SQL berjalan setelah konektor Snowflake melakukan tindakan standarnya.
- `Action` – String UTF-8.

  Menentukan tindakan apa yang harus diambil saat menulis ke tabel dengan data yang sudah ada sebelumnya. Nilai yang valid: `append`,`merge`,`truncate`,`drop`.
- `Upsert` – Boolean.

  Digunakan saat Action`append`. Menentukan perilaku resolusi ketika baris sudah ada. Jika benar, baris yang sudah ada sebelumnya akan diperbarui. Jika salah, baris-baris itu akan dimasukkan.
- `MergeAction` — String UTF-8, yang cocok dengan Custom string pattern #39.

  Menentukan tindakan gabungan. Nilai-nilai yang valid: `simple`, `custom`. Jika sederhana, perilaku penggabungan didefinisikan oleh `MergeWhenMatched` dan `MergeWhenNotMatched`. Jika kustom, ditentukan oleh`MergeClause`.
- `MergeWhenMatched` — String UTF-8, yang cocok dengan Custom string pattern #39.

  Menentukan cara menyelesaikan catatan yang cocok dengan data yang sudah ada sebelumnya saat menggabungkan. Nilai-nilai yang valid: `update`, `delete`.
- `MergeWhenNotMatched` — String UTF-8, yang cocok dengan Custom string pattern #39.

  Menentukan cara memproses catatan yang tidak cocok dengan data yang sudah ada sebelumnya saat menggabungkan. Nilai-nilai yang valid: `insert`, `none`.

- `MergeClause` – String UTF-8.

  Pernyataan SQL yang menentukan perilaku gabungan kustom.

- `StagingTable` – String UTF-8.

  Nama tabel pementasan yang digunakan saat melakukan `merge` atau meningkatkan tindakan. `append` Data ditulis ke tabel ini, kemudian dipindahkan ke `table` oleh postaction yang dihasilkan.

- `SelectedColumns` – Susunan objek Opsi.

  Menentukan kolom yang digabungkan untuk mengidentifikasi catatan saat mendeteksi kecocokan untuk penggabungan dan upserts. Daftar struktur dengan`value`, `label` dan `description` kunci. Setiap struktur menggambarkan kolom.

- `AutoPushdown` – Boolean.

  Menentukan apakah permintaan otomatis pushdown diaktifkan. Jika pushdown diaktifkan, maka ketika kueri dijalankan di Spark, jika bagian dari kueri dapat “didorong ke bawah” ke server Snowflake, itu didorong ke bawah. Ini meningkatkan kinerja beberapa kueri.

- `TableSchema` – Susunan objek Opsi.

  Secara manual mendefinisikan skema target untuk node. Daftar struktur dengan`value`, `label` dan `description` kunci. Setiap struktur mendefinisikan kolom.

SnowflakeSource struktur

Menentukan sumber data Snowflake.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama sumber data Snowflake.

- `Data` — Wajib: Sebuah objek SnowflakeNodeData.

  Konfigurasi untuk sumber data Snowflake.

- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema yang ditentukan pengguna untuk data output Anda.

SnowflakeTarget struktur

Menentukan target Snowflake.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama target Snowflake.
- `Data` — Wajib: Sebuah objek SnowflakeNodeData.

  Menentukan data dari node target Snowflake.
- `Inputs`— Array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.

ConnectorDataSource struktur

Menentukan sumber yang dihasilkan dengan pilihan koneksi standar.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama simpul sumber ini.
- `ConnectionType` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Ituconnection`Type`, sebagaimana disediakan untuk AWS Glue perpustakaan yang mendasarinya. Tipe node ini mendukung jenis koneksi berikut:
  - `opensearch`
  - `azuresql`
  - `azurecosmos`
  - `bigquery`
  - `saphana`
  - `teradata`
  - `vertica`
- `Data` – Wajib: Susunan peta pasangan nilai kunci.

Setiap kunci adalah string UTF-8.

Setiap nilai adalah string UTF-8.

Peta yang menentukan opsi koneksi untuk node. Anda dapat menemukan opsi koneksi standar untuk jenis koneksi yang sesuai di bagian Parameter koneksi AWS Glue dokumentasi.

- `OutputSchemas` – Susunan objek GlueSchema.

  Menentukan skema data untuk sumber ini.

ConnectorDataTarget struktur

Menentukan target yang dihasilkan dengan pilihan koneksi standar.

Bidang

- `Name` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #43.

  Nama node target ini.

- `ConnectionType` — Wajib: String UTF-8, yang cocok dengan Custom string pattern #40.

  Ituconnection`Type`, sebagaimana disediakan untuk AWS Glue perpustakaan yang mendasarinya. Tipe node ini mendukung jenis koneksi berikut:

  - `opensearch`
  - `azuresql`
  - `azurecosmos`
  - `bigquery`
  - `saphana`
  - `teradata`
  - `vertica`

- `Data` – Wajib: Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

Peta yang menentukan opsi koneksi untuk node. Anda dapat menemukan opsi koneksi standar untuk jenis koneksi yang sesuai di bagian Parameter koneksi AWS Glue dokumentasi.

- `Inputs`— Array string UTF-8, tidak kurang dari 1 atau lebih dari 1 string.

  Node yang merupakan input ke target data.

API Tugas

Jobs API menjelaskan tipe data lowongan dan berisi API untuk bekerja dengan pekerjaan, menjalankan pekerjaan, dan pemicu diAWS Glue.

Topik

- Tugas
- Tugas berjalan
- Pemicu

Tugas

Jobs API menjelaskan tipe data dan API yang terkait dengan pembuatan, pembaruan, penghapusan, atau tampilan pekerjaan di AWS Glue.

Jenis data

- Struktur Job
- ExecutionProperty struktur
- NotificationProperty struktur
- JobCommand struktur
- ConnectionsList struktur
- JobUpdate struktur
- SourceControlDetails struktur

Struktur Job

Menentukan sebuah definisi tugas.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama yang Anda tetapkan untuk definisi tugas ini.
- `JobMode` – String UTF-8 (nilai yang valid: `SCRIPT=""` | `VISUAL=""` | `NOTEBOOK=""`).

  Mode yang menggambarkan bagaimana pekerjaan dibuat. Nilai yang valid adalah:
  - `SCRIPT`- Pekerjaan dibuat menggunakan editor skrip AWS Glue Studio.
  - `VISUAL`- Pekerjaan dibuat menggunakan editor visual AWS Glue Studio.
  - `NOTEBOOK`- Pekerjaan itu dibuat menggunakan notebook sesi interaktif.

  Ketika `JobMode` bidang hilang atau null, `SCRIPT` ditetapkan sebagai nilai default.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi tugas tersebut.
- `LogUri` – String UTF-8.

  Bidang ini disimpan untuk penggunaan masa depan.
- `Role` – String UTF-8.

  Nama atau Amazon Resource Name (ARN) dari IAM role yang dikaitkan dengan tugas ini.
- `CreatedOn` — Stempel waktu.

  Waktu dan tanggal saat definisi tugas ini dibuat.
- `LastModifiedOn` — Stempel waktu.

  Titik dalam waktu terakhir ketika definisi tugas ini dimodifikasi.
- `ExecutionProperty` — Sebuah objek ExecutionProperty.

  Sebuah `ExecutionProperty` yang menentukan jumlah maksimum eksekusi bersamaan yang diperbolehkan untuk tugas ini.
- `Command` — Sebuah objek JobCommand.

  `JobCommand` yang menjalankan tugas ini.
- `DefaultArguments` – Susunan peta pasangan nilai kunci.

Setiap kunci adalah string UTF-8.

Setiap nilai adalah string UTF-8.

Argumen default untuk setiap menjalankan pekerjaan ini, ditetapkan sebagai pasangan nama-nilai.

Anda dapat menentukan argumen di sini yang digunakan skrip eksekusi pekerjaan Anda sendiri, serta argumen yang AWS Glue dikonsumsi sendiri.

Argumen Job dapat dicatat. Jangan berikan rahasia plaintext sebagai argumen. Ambil rahasia dari AWS Glue Connection, AWS Secrets Manager atau mekanisme manajemen rahasia lainnya jika Anda ingin menyimpannya di dalam Job.

Untuk informasi tentang cara menentukan dan menggunakan argumen Tugas Anda sendiri, lihat topik Memanggil API AWS Glue dalam Python dalam panduan developer.

Untuk informasi tentang argumen yang dapat Anda berikan ke bidang ini saat mengonfigurasi pekerjaan Spark, lihat Parameter Khusus yang Digunakan menurut AWS Glue topik dalam panduan pengembang.

Untuk informasi tentang argumen yang dapat Anda berikan ke bidang ini saat mengonfigurasi pekerjaan Ray, lihat Menggunakan parameter pekerjaan di pekerjaan Ray di panduan pengembang.

- `NonOverridableArguments` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

  Argumen untuk pekerjaan ini yang tidak diganti saat memberikan argumen pekerjaan dalam menjalankan pekerjaan, ditentukan sebagai pasangan nama-nilai.

- `Connections` — Sebuah objek ConnectionsList.

  Koneksi yang digunakan untuk tugas ini.

- `MaxRetries` — Nomor (bilangan bulat).

  Jumlah maksimum kali untuk mencoba kembali pekerjaan ini setelah JobRun gagal.

- `AllocatedCapacity` — Nomor (bilangan bulat).

Bidang ini tidak lagi digunakan. Gunakan `MaxCapacity` sebagai gantinya.

Jumlah unit pemrosesan AWS Glue data (DPU) yang dialokasikan untuk menjalankan pekerjaan ini. Anda dapat mengalokasikan minimal 2 DPU; defaultnya adalah 10. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Waktu habis tugas, dalam satuan menit. Ini adalah waktu maksimum yang bisa digunakan oleh eksekusi tugas untuk menggunakan sumber daya sebelum eksekusi dihentikan dan memasuki status `TIMEOUT`. Standarnya adalah 2.880 menit (48 jam) untuk pekerjaan batch.

  Pekerjaan streaming harus memiliki nilai batas waktu kurang dari 7 hari atau 10080 menit. Ketika nilai dibiarkan kosong, pekerjaan akan dimulai ulang setelah 7 hari berdasarkan jika Anda belum menyiapkan jendela pemeliharaan. Jika Anda memiliki jendela pemeliharaan pengaturan, itu akan dimulai ulang selama jendela pemeliharaan setelah 7 hari.

- `MaxCapacity` — Nomor (ganda).

  Untuk Glue versi 1.0 atau pekerjaan sebelumnya, menggunakan tipe pekerja standar, jumlah unit pemrosesan AWS Glue data (DPU) yang dapat dialokasikan saat pekerjaan ini berjalan. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

  Untuk Glue versi 2.0 atau pekerjaan yang lebih baru, Anda tidak dapat menentukan`Maximum capacity`. Sebaliknya, Anda harus menentukan `Worker type` dan `Number of workers`.

  Jangan mengatur `MaxCapacity` jika Anda menggunakan `WorkerType` dan `NumberOfWorkers`.

  Nilai yang dapat dialokasikan untuk `MaxCapacity` tergantung pada apakah Anda menjalankan tugas shell Python, tugas ETL Apache Spark, atau tugas ETL Apache Spark streaming:

  - Ketika anda menentukan tugas shell Python (`JobCommand.Name`="pythonshell"), Anda dapat mengalokasikan 0,0625 atau 1 DPU. Default-nya adalah 0,0625 DPU.
  - Bila Anda menentukan tugas ETL Apache Spark (`JobCommand.Name`="glueetl") atau tugas ETL Apache Spark streaming (`JobCommand.Name`="gluestreaming"), Anda dapat mengalokasikan dari 2 hingga 100 dPU. Default-nya adalah 10 DPU. Jenis tugas ini tidak dapat memiliki alokasi DPU pecahan.

- `WorkerType` – String UTF-8 (nilai yang valid: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan ketika sebuah tugas dieksekusi. Menerima nilai G.1X, G.2X, G.4X, G.8X atau G.025X untuk pekerjaan Spark. Menerima nilai Z.2X untuk pekerjaan Ray.

  - Untuk tipe `G.1X` pekerja, setiap pekerja memetakan ke 1 DPU (4 vCPU, memori 16 GB) dengan disk 84GB (sekitar 34GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang terukur dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
  - Untuk tipe `G.2X` pekerja, setiap pekerja memetakan ke 2 DPU (8 vCPU, memori 32 GB) dengan disk 128GB (sekitar 77GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang terukur dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
  - Untuk tipe `G.4X` pekerja, setiap pekerja memetakan ke 4 DPU (16 vCPU, memori 64 GB) dengan disk 256GB (sekitar 235GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru di AWS Wilayah berikut: AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (Oregon), Asia Pasifik (Singapura), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Kanada (Tengah), Eropa (Frankfurt), Eropa (Irlandia), dan Eropa (Stockholm).
  - Untuk tipe `G.8X` pekerja, setiap pekerja memetakan ke 8 DPU (32 vCPU, memori 128 GB) dengan disk 512GB (sekitar 487GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru, di AWS Wilayah yang sama seperti yang didukung untuk tipe `G.4X` pekerja.
  - Untuk tipe `G.025X` pekerja, setiap pekerja memetakan ke 0,25 DPU (2 vCPU, memori 4 GB) dengan disk 84GB (sekitar 34GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan streaming volume rendah. Jenis pekerja ini hanya tersedia untuk pekerjaan streaming AWS Glue versi 3.0.

  - Untuk tipe `Z.2X` pekerja, setiap pekerja memetakan ke 2 M-DPU (8vCPU, memori 64 GB) dengan disk 128 GB (sekitar 120GB gratis), dan menyediakan hingga 8 pekerja Ray berdasarkan autoscaler.
- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah pekerja dari `workerType` ditentukan yang dialokasikan ketika sebuah tugas dieksekusi.
- `SecurityConfiguration` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama struktur `SecurityConfiguration` yang akan digunakan dengan tugas ini.
- `NotificationProperty` — Sebuah objek NotificationProperty.

  Menentukan properti konfigurasi dari sebuah notifikasi tugas.
- `Running` – Boolean.

  Bidang ini disimpan untuk penggunaan masa depan.
- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

  Dalam pekerjaan Spark, `GlueVersion` menentukan versi Apache Spark dan Python yang AWS Glue tersedia dalam suatu pekerjaan. Versi Python menunjukkan versi yang didukung untuk tugas tipe Spark.

  Pekerjaan Ray harus diatur `GlueVersion` ke `4.0` atau lebih besar. Namun, versi Ray, Python, dan pustaka tambahan yang tersedia di pekerjaan Ray Anda ditentukan oleh `Runtime` parameter perintah Job.

  Untuk informasi selengkapnya tentang AWS Glue versi yang tersedia dan versi Spark dan Python yang sesuai, lihat Versi Glue di panduan pengembang.

  Tugas yang dibuat tanpa menentukan versi Glue default ke Glue 0.9.
- `CodeGenConfigurationNodes` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #39

  Setiap nilai adalah sebuah objek CodeGenConfigurationNode A.

  Representasi grafik asiklik terarah yang menjadi dasar komponen visual Glue Studio dan pembuatan kode Glue Studio.

- `ExecutionClass`— String UTF-8, panjangnya tidak lebih dari 16 byte (nilai valid: `FLEX`"" |). `STANDARD`""

  Menunjukkan apakah pekerjaan dijalankan dengan kelas eksekusi standar atau fleksibel. Kelas eksekusi standar sangat ideal untuk beban kerja yang sensitif terhadap waktu yang membutuhkan startup pekerjaan cepat dan sumber daya khusus.

  Kelas eksekusi fleksibel cocok untuk pekerjaan yang tidak sensitif terhadap waktu yang waktu mulai dan penyelesaiannya dapat bervariasi.

  Hanya pekerjaan dengan AWS Glue versi 3.0 ke atas dan jenis perintah `glueetl` yang diizinkan untuk disetel `ExecutionClass` ke`FLEX`. Kelas eksekusi fleksibel tersedia untuk pekerjaan Spark.
- `SourceControlDetails` — Sebuah objek SourceControlDetails.

  Detail untuk konfigurasi kontrol sumber untuk pekerjaan, memungkinkan sinkronisasi artefak pekerjaan ke atau dari repositori jarak jauh.
- `MaintenanceWindow` — String UTF-8, yang cocok dengan Custom string pattern #30.

  Bidang ini menentukan hari dalam seminggu dan jam untuk jendela pemeliharaan untuk pekerjaan streaming. AWS Glue secara berkala melakukan kegiatan pemeliharaan. Selama jendela pemeliharaan ini, Anda AWS Glue perlu memulai ulang pekerjaan streaming Anda.

  AWS Glue akan memulai kembali pekerjaan dalam waktu 3 jam dari jendela pemeliharaan yang ditentukan. Misalnya, jika Anda mengatur jendela pemeliharaan untuk hari Senin pukul 10:00 GMT, pekerjaan Anda akan dimulai kembali antara 10:00 GMT hingga 1:00 GMT.
- `ProfileName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil AWS Glue penggunaan yang terkait dengan pekerjaan.

ExecutionProperty struktur

Properti eksekusi dari tugas.

Bidang

- `MaxConcurrentRuns` — Nomor (bilangan bulat).

Jumlah maksimum berjalan bersamaan yang diizinkan untuk tugas. Default-nya adalah 1. Kesalahan dikembalikan ketika ambang batas ini tercapai. Nilai maksimum yang dapat Anda tentukan dikendalikan oleh batas layanan.

NotificationProperty struktur

Menentukan sifat konfigurasi dari notifikasi.

Bidang

- `NotifyDelayAfter` — Nomor (bilangan bulat), minimal 1.

  Setelah tugas dimulai, jumlah menit untuk menunggu sebelum mengirim notifikasi penundaan tugas.

JobCommand struktur

Menentukan kode yang berjalan ketika tugas dijalankan.

Bidang

- `Name` – String UTF-8.

  Nama perintah tugas. Untuk tugas Apache Spark ETL, ini harus berupa `glueetl`. Untuk tugas shell Python, ini harus berupa `pythonshell`. Untuk tugas ETL Apache Spark streaming, ini harus berupa `gluestreaming`. Untuk pekerjaan Ray, ini pastigl`ueray`.
- `ScriptLocation`- String UTF-8, panjangnya tidak lebih dari 400000 byte.

  Menentukan path Amazon Simple Storage Service (Amazon S3) ke skrip yang menjalankan tugas.
- `PythonVersion` — String UTF-8, yang cocok dengan Custom string pattern #21.

  Versi Python yang digunakan untuk menjalankan tugas shell Python. Nilai yang diizinkan adalah 2 atau 3.
- `Runtime`— String UTF-8, panjangnya tidak lebih dari 64 byte, cocok dengan file. Custom string pattern #29

  Dalam pekerjaan Ray, Runtime digunakan untuk menentukan versi Ray, Python, dan pustaka tambahan yang tersedia di lingkungan Anda. Bidang ini tidak digunakan dalam jenis pekerjaan

lain. Untuk nilai lingkungan runtime yang didukung, lihat Lingkungan runtime Ray yang didukung di Panduan AWS Glue Pengembang.

ConnectionsList struktur

Menentukan koneksi yang digunakan oleh tugas.

Bidang

- `Connections` – Susunan string UTF-8.

  Daftar koneksi yang digunakan oleh tugas.

JobUpdate struktur

Menentukan informasi yang digunakan untuk memperbarui definisi tugas yang ada. Definisi tugas sebelumnya benar-benar ditimpa oleh informasi ini.

Bidang

- `JobMode` – String UTF-8 (nilai yang valid: `SCRIPT=""` | `VISUAL=""` | `NOTEBOOK=""`).

  Mode yang menggambarkan bagaimana pekerjaan dibuat. Nilai yang valid adalah:

  - `SCRIPT`- Pekerjaan dibuat menggunakan editor skrip AWS Glue Studio.
  - `VISUAL`- Pekerjaan dibuat menggunakan editor visual AWS Glue Studio.
  - `NOTEBOOK`- Pekerjaan itu dibuat menggunakan notebook sesi interaktif.

  Ketika `JobMode` bidang hilang atau null, `SCRIPT` ditetapkan sebagai nilai default.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi tugas yang sedang didefinisikan.
- `LogUri` – String UTF-8.

  Bidang ini disimpan untuk penggunaan masa depan.
- `Role` – String UTF-8.

  Nama atau Amazon Resource Name (ARN) dari IAM role yang dikaitkan dengan tugas ini (wajib).
- `ExecutionProperty` — Sebuah objek ExecutionProperty.

Sebuah `ExecutionProperty` yang menentukan jumlah maksimum eksekusi bersamaan yang diperbolehkan untuk tugas ini.

- `Command` — Sebuah objek JobCommand.

  `JobCommand` yang menjalankan tugas ini (wajib).

- `DefaultArguments` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

  Argumen default untuk setiap menjalankan pekerjaan ini, ditetapkan sebagai pasangan nama-nilai.

  Anda dapat menentukan argumen di sini yang digunakan skrip eksekusi pekerjaan Anda sendiri, serta argumen yang AWS Glue dikonsumsi sendiri.

  Argumen Job dapat dicatat. Jangan berikan rahasia plaintext sebagai argumen. Ambil rahasia dari AWS Glue Connection, AWS Secrets Manager atau mekanisme manajemen rahasia lainnya jika Anda ingin menyimpannya di dalam Job.

  Untuk informasi tentang cara menentukan dan menggunakan argumen Tugas Anda sendiri, lihat topik Memanggil API AWS Glue dalam Python dalam panduan developer.

  Untuk informasi tentang argumen yang dapat Anda berikan ke bidang ini saat mengonfigurasi pekerjaan Spark, lihat Parameter Khusus yang Digunakan menurut AWS Glue topik dalam panduan pengembang.

  Untuk informasi tentang argumen yang dapat Anda berikan ke bidang ini saat mengonfigurasi pekerjaan Ray, lihat Menggunakan parameter pekerjaan di pekerjaan Ray di panduan pengembang.

- `NonOverridableArguments` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

  Argumen untuk pekerjaan ini yang tidak diganti saat memberikan argumen pekerjaan dalam menjalankan pekerjaan, ditentukan sebagai pasangan nama-nilai.

- `Connections` — Sebuah objek ConnectionsList.

Koneksi yang digunakan untuk tugas ini.

- `MaxRetries` — Nomor (bilangan bulat).

  Jumlah waktu maksimum berapa kali percobaan yang bisa dilakukan untuk tugas ini jika gagal.

- `AllocatedCapacity` — Nomor (bilangan bulat).

  Bidang ini tidak lagi digunakan. Gunakan `MaxCapacity` sebagai gantinya.

  Jumlah unit pemrosesan AWS Glue data (DPU) untuk dialokasikan untuk pekerjaan ini. Anda dapat mengalokasikan minimal 2 DPU; defaultnya adalah 10. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Waktu habis tugas, dalam satuan menit. Ini adalah waktu maksimum yang bisa digunakan oleh eksekusi tugas untuk menggunakan sumber daya sebelum eksekusi dihentikan dan memasuki status `TIMEOUT`. Standarnya adalah 2.880 menit (48 jam) untuk pekerjaan batch.

  Pekerjaan streaming harus memiliki nilai batas waktu kurang dari 7 hari atau 10080 menit. Ketika nilai dibiarkan kosong, pekerjaan akan dimulai ulang setelah 7 hari berdasarkan jika Anda belum menyiapkan jendela pemeliharaan. Jika Anda memiliki jendela pemeliharaan pengaturan, itu akan dimulai ulang selama jendela pemeliharaan setelah 7 hari.

- `MaxCapacity` — Nomor (ganda).

  Untuk Glue versi 1.0 atau pekerjaan sebelumnya, menggunakan tipe pekerja standar, jumlah unit pemrosesan AWS Glue data (DPU) yang dapat dialokasikan saat pekerjaan ini berjalan. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

  Untuk pekerjaan Glue versi 2.0+, Anda tidak dapat menentukan. `Maximum capacity` Sebaliknya, Anda harus menentukan `Worker type` dan `Number of workers`.

  Jangan mengatur `MaxCapacity` jika Anda menggunakan `WorkerType` dan `NumberOfWorkers`.

  Nilai yang dapat dialokasikan untuk `MaxCapacity` tergantung pada apakah Anda menjalankan tugas shell Python, tugas ETL Apache Spark, atau tugas ETL Apache Spark streaming:

  - Ketika anda menentukan tugas shell Python (`JobCommand.Name`="pythonshell"), Anda dapat mengalokasikan 0,0625 atau 1 DPU. Default-nya adalah 0,0625 DPU.

  - Bila Anda menentukan tugas ETL Apache Spark (`JobCommand.Name`="glueetl") atau tugas ETL Apache Spark streaming (`JobCommand.Name`="gluestreaming"), Anda dapat mengalokasikan dari 2 hingga 100 dPU. Default-nya adalah 10 DPU. Jenis tugas ini tidak dapat memiliki alokasi DPU pecahan.
- `WorkerType` – String UTF-8 (nilai yang valid: `Standard`="" | `G.1X`="" | `G.2X`="" | `G.025X`="" | `G.4X`="" | `G.8X`="" | `Z.2X`="").

  Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan ketika sebuah tugas dieksekusi. Menerima nilai G.1X, G.2X, G.4X, G.8X atau G.025X untuk pekerjaan Spark. Menerima nilai Z.2X untuk pekerjaan Ray.

  - Untuk tipe `G.1X` pekerja, setiap pekerja memetakan ke 1 DPU (4 vCPU, memori 16 GB) dengan disk 84GB (sekitar 34GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang terukur dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
  - Untuk tipe `G.2X` pekerja, setiap pekerja memetakan ke 2 DPU (8 vCPU, memori 32 GB) dengan disk 128GB (sekitar 77GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang terukur dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
  - Untuk tipe `G.4X` pekerja, setiap pekerja memetakan ke 4 DPU (16 vCPU, memori 64 GB) dengan disk 256GB (sekitar 235GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru di AWS Wilayah berikut: AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (Oregon), Asia Pasifik (Singapura), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Kanada (Tengah), Eropa (Frankfurt), Eropa (Irlandia), dan Eropa (Stockholm).
  - Untuk tipe `G.8X` pekerja, setiap pekerja memetakan ke 8 DPU (32 vCPU, memori 128 GB) dengan disk 512GB (sekitar 487GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru, di AWS Wilayah yang sama seperti yang didukung untuk tipe `G.4X` pekerja.

  - Untuk tipe `G.025X` pekerja, setiap pekerja memetakan ke 0,25 DPU (2 vCPU, memori 4 GB) dengan disk 84GB (sekitar 34GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan streaming volume rendah. Jenis pekerja ini hanya tersedia untuk pekerjaan streaming AWS Glue versi 3.0.
  - Untuk tipe `Z.2X` pekerja, setiap pekerja memetakan ke 2 M-DPU (8vCPU, memori 64 GB) dengan disk 128 GB (sekitar 120GB gratis), dan menyediakan hingga 8 pekerja Ray berdasarkan autoscaler.
- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah pekerja dari `workerType` ditentukan yang dialokasikan ketika sebuah tugas dieksekusi.
- `SecurityConfiguration` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama struktur `SecurityConfiguration` yang akan digunakan dengan tugas ini.
- `NotificationProperty` — Sebuah objek NotificationProperty.

  Menentukan properti konfigurasi notifikasi tugas.
- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

  Dalam pekerjaan Spark, `GlueVersion` menentukan versi Apache Spark dan Python yang AWS Glue tersedia dalam suatu pekerjaan. Versi Python menunjukkan versi yang didukung untuk tugas tipe Spark.

  Pekerjaan Ray harus diatur `GlueVersion` ke `4.0` atau lebih besar. Namun, versi Ray, Python, dan pustaka tambahan yang tersedia di pekerjaan Ray Anda ditentukan oleh `Runtime` parameter perintah Job.

  Untuk informasi selengkapnya tentang AWS Glue versi yang tersedia dan versi Spark dan Python yang sesuai, lihat Versi Glue di panduan pengembang.

  Tugas yang dibuat tanpa menentukan versi Glue default ke Glue 0.9.
- `CodeGenConfigurationNodes` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #39

  Setiap nilai adalah sebuah objek CodeGenConfigurationNode A.

Representasi grafik asiklik terarah yang menjadi dasar komponen visual Glue Studio dan pembuatan kode Glue Studio.

- `ExecutionClass`— String UTF-8, panjangnya tidak lebih dari 16 byte (nilai valid: `FLEX`"" |). `STANDARD`""

  Menunjukkan apakah pekerjaan dijalankan dengan kelas eksekusi standar atau fleksibel. Kelas eksekusi standar sangat ideal untuk beban kerja yang sensitif terhadap waktu yang membutuhkan startup pekerjaan cepat dan sumber daya khusus.

  Kelas eksekusi fleksibel cocok untuk pekerjaan yang tidak sensitif terhadap waktu yang waktu mulai dan penyelesaiannya dapat bervariasi.

  Hanya pekerjaan dengan AWS Glue versi 3.0 ke atas dan jenis perintah `glueetl` yang diizinkan untuk disetel `ExecutionClass` ke`FLEX`. Kelas eksekusi fleksibel tersedia untuk pekerjaan Spark.

- `SourceControlDetails` — Sebuah objek SourceControlDetails.

  Detail untuk konfigurasi kontrol sumber untuk pekerjaan, memungkinkan sinkronisasi artefak pekerjaan ke atau dari repositori jarak jauh.

- `MaintenanceWindow` — String UTF-8, yang cocok dengan Custom string pattern #30.

  Bidang ini menentukan hari dalam seminggu dan jam untuk jendela pemeliharaan untuk pekerjaan streaming. AWS Glue secara berkala melakukan kegiatan pemeliharaan. Selama jendela pemeliharaan ini, Anda AWS Glue perlu memulai ulang pekerjaan streaming Anda.

  AWS Glue akan memulai kembali pekerjaan dalam waktu 3 jam dari jendela pemeliharaan yang ditentukan. Misalnya, jika Anda mengatur jendela pemeliharaan untuk hari Senin pukul 10:00 GMT, pekerjaan Anda akan dimulai kembali antara 10:00 GMT hingga 1:00 GMT.

- `ProfileName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil AWS Glue penggunaan yang terkait dengan pekerjaan.

SourceControlDetails struktur

Detail untuk konfigurasi kontrol sumber untuk pekerjaan, memungkinkan sinkronisasi artefak pekerjaan ke atau dari repositori jarak jauh.

Bidang

- `Provider` – String UTF-8.

  Penyedia untuk repositori jarak jauh.

- `Repository` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 512 byte.

  Nama repositori jarak jauh yang berisi artefak pekerjaan.

- `Owner` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 512 byte.

  Pemilik repositori jarak jauh yang berisi artefak pekerjaan.

- `Branch` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 512 byte.

  Cabang opsional di repositori jarak jauh.

- `Folder` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 512 byte.

  Folder opsional di repositori jarak jauh.

- `LastCommitId` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 512 byte.

  ID komit terakhir untuk komit di repositori jarak jauh.

- `LastSyncTimestamp` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 512 byte.

  Tanggal dan waktu sinkronisasi pekerjaan terakhir dilakukan.

- `AuthStrategy` – String UTF-8.

  Jenis otentikasi, yang dapat berupa token otentikasi yang disimpan di AWS Secrets Manager, atau token akses pribadi.

- `AuthToken` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 512 byte.

  Nilai token otorisasi.

Operasi

- CreateJob tindakan (Python: create_job)
- UpdateJob tindakan (Python: update_job)
- GetJob tindakan (Python: get_job)

- GetJobs tindakan (Python: get_jobs)
- DeleteJob tindakan (Python: delete_job)
- ListJobs tindakan (Python: list_jobs)
- BatchGetJobs tindakan (Python: batch_get_jobs)

CreateJob tindakan (Python: create_job)

Menciptakan sebuah definisi tugas baru.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama yang Anda tetapkan untuk definisi tugas ini. Harus unik dalam akun Anda.
- `JobMode` – String UTF-8 (nilai yang valid: `SCRIPT=""` | `VISUAL=""` | `NOTEBOOK=""`).

  Mode yang menggambarkan bagaimana pekerjaan dibuat. Nilai yang valid adalah:
  - `SCRIPT`- Pekerjaan dibuat menggunakan editor skrip AWS Glue Studio.
  - `VISUAL`- Pekerjaan dibuat menggunakan editor visual AWS Glue Studio.
  - `NOTEBOOK`- Pekerjaan itu dibuat menggunakan notebook sesi interaktif.

  Ketika `JobMode` bidang hilang atau null, `SCRIPT` ditetapkan sebagai nilai default.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi tugas yang sedang didefinisikan.
- `LogUri` – String UTF-8.

  Bidang ini disimpan untuk penggunaan masa depan.
- `Role` – Wajib: String UTF-8.

  Nama atau Amazon Resource Name (ARN) dari IAM role yang dikaitkan dengan tugas ini.
- `ExecutionProperty` — Sebuah objek ExecutionProperty.

  Sebuah `ExecutionProperty` yang menentukan jumlah maksimum eksekusi bersamaan yang diperbolehkan untuk tugas ini.

- `Command` — Wajib: Sebuah objek JobCommand.

  `JobCommand` yang menjalankan tugas ini.

- `DefaultArguments` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

  Argumen default untuk setiap menjalankan pekerjaan ini, ditetapkan sebagai pasangan nama-nilai.

  Anda dapat menentukan argumen di sini yang digunakan skrip eksekusi pekerjaan Anda sendiri, serta argumen yang AWS Glue dikonsumsi sendiri.

  Argumen Job dapat dicatat. Jangan berikan rahasia plaintext sebagai argumen. Ambil rahasia dari AWS Glue Connection, AWS Secrets Manager atau mekanisme manajemen rahasia lainnya jika Anda ingin menyimpannya di dalam Job.

  Untuk informasi tentang cara menentukan dan menggunakan argumen Tugas Anda sendiri, lihat topik Memanggil API AWS Glue dalam Python dalam panduan developer.

  Untuk informasi tentang argumen yang dapat Anda berikan ke bidang ini saat mengonfigurasi pekerjaan Spark, lihat Parameter Khusus yang Digunakan menurut AWS Glue topik dalam panduan pengembang.

  Untuk informasi tentang argumen yang dapat Anda berikan ke bidang ini saat mengonfigurasi pekerjaan Ray, lihat Menggunakan parameter pekerjaan di pekerjaan Ray di panduan pengembang.

- `NonOverridableArguments` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

  Argumen untuk pekerjaan ini yang tidak diganti saat memberikan argumen pekerjaan dalam menjalankan pekerjaan, ditentukan sebagai pasangan nama-nilai.

- `Connections` — Sebuah objek ConnectionsList.

  Koneksi yang digunakan untuk tugas ini.

- `MaxRetries` — Nomor (bilangan bulat).

Jumlah waktu maksimum berapa kali percobaan yang bisa dilakukan untuk tugas ini jika gagal.

- `AllocatedCapacity` — Nomor (bilangan bulat).

  Parameter ini tidak lagi digunakan. Gunakan `MaxCapacity` sebagai gantinya.

  Jumlah unit pemrosesan AWS Glue data (DPU) yang akan dialokasikan ke Job ini. Anda dapat mengalokasikan minimal 2 DPU; defaultnya adalah 10. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Waktu habis tugas, dalam satuan menit. Ini adalah waktu maksimum yang bisa digunakan oleh eksekusi tugas untuk menggunakan sumber daya sebelum eksekusi dihentikan dan memasuki status `TIMEOUT`. Standarnya adalah 2.880 menit (48 jam) untuk pekerjaan batch.

  Pekerjaan streaming harus memiliki nilai batas waktu kurang dari 7 hari atau 10080 menit. Ketika nilai dibiarkan kosong, pekerjaan akan dimulai ulang setelah 7 hari berdasarkan jika Anda belum menyiapkan jendela pemeliharaan. Jika Anda memiliki jendela pemeliharaan pengaturan, itu akan dimulai ulang selama jendela pemeliharaan setelah 7 hari.

- `MaxCapacity` — Nomor (ganda).

  Untuk Glue versi 1.0 atau pekerjaan sebelumnya, menggunakan tipe pekerja standar, jumlah unit pemrosesan AWS Glue data (DPU) yang dapat dialokasikan saat pekerjaan ini berjalan. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

  Untuk pekerjaan Glue versi 2.0+, Anda tidak dapat menentukan. `Maximum capacity` Sebaliknya, Anda harus menentukan `Worker type` dan `Number of workers`.

  Jangan mengatur `MaxCapacity` jika Anda menggunakan `WorkerType` dan `NumberOfWorkers`.

  Nilai yang dapat dialokasikan untuk `MaxCapacity` tergantung pada apakah Anda menjalankan tugas shell Python, tugas ETL Apache Spark, atau tugas ETL Apache Spark streaming:

  - Ketika anda menentukan tugas shell Python (`JobCommand.Name`="pythonshell"), Anda dapat mengalokasikan 0,0625 atau 1 DPU. Default-nya adalah 0,0625 DPU.
  - Bila Anda menentukan tugas ETL Apache Spark (`JobCommand.Name`="glueetl") atau tugas ETL Apache Spark streaming (`JobCommand.Name`="gluestreaming"), Anda dapat mengalokasikan

dari 2 hingga 100 dPU. Default-nya adalah 10 DPU. Jenis tugas ini tidak dapat memiliki alokasi DPU pecahan.

- `SecurityConfiguration` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama struktur `SecurityConfiguration` yang akan digunakan dengan tugas ini.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Tag yang akan digunakan dengan tugas ini. Anda dapat menggunakan tag untuk membatasi akses ke tugas. Untuk informasi selengkapnya tentang AWS tag AWS Glue, lihat Tag AWS Glue di panduan pengembang.
- `NotificationProperty` — Sebuah objek NotificationProperty.

  Menentukan properti konfigurasi dari sebuah notifikasi tugas.
- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

  Dalam pekerjaan Spark, `GlueVersion` menentukan versi Apache Spark dan Python yang AWS Glue tersedia dalam suatu pekerjaan. Versi Python menunjukkan versi yang didukung untuk tugas tipe Spark.

  Pekerjaan Ray harus diatur `GlueVersion` ke `4.0` atau lebih besar. Namun, versi Ray, Python, dan pustaka tambahan yang tersedia di pekerjaan Ray Anda ditentukan oleh `Runtime` parameter perintah Job.

  Untuk informasi selengkapnya tentang AWS Glue versi yang tersedia dan versi Spark dan Python yang sesuai, lihat Versi Glue di panduan pengembang.

  Tugas yang dibuat tanpa menentukan versi Glue default ke Glue 0.9.
- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah pekerja dari `workerType` ditentukan yang dialokasikan ketika sebuah tugas dieksekusi.
- `WorkerType` – String UTF-8 (nilai yang valid: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan ketika sebuah tugas dieksekusi. Menerima nilai G.1X, G.2X, G.4X, G.8X atau G.025X untuk pekerjaan Spark. Menerima nilai Z.2X untuk pekerjaan Ray.

- Untuk tipe `G.1X` pekerja, setiap pekerja memetakan ke 1 DPU (4 vCPU, memori 16 GB) dengan disk 84GB (sekitar 34GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang terukur dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
- Untuk tipe `G.2X` pekerja, setiap pekerja memetakan ke 2 DPU (8 vCPU, memori 32 GB) dengan disk 128GB (sekitar 77GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang terukur dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
- Untuk tipe `G.4X` pekerja, setiap pekerja memetakan ke 4 DPU (16 vCPU, memori 64 GB) dengan disk 256GB (sekitar 235GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru di AWS Wilayah berikut: AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (Oregon), Asia Pasifik (Singapura), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Kanada (Tengah), Eropa (Frankfurt), Eropa (Irlandia), dan Eropa (Stockholm).
- Untuk tipe `G.8X` pekerja, setiap pekerja memetakan ke 8 DPU (32 vCPU, memori 128 GB) dengan disk 512GB (sekitar 487GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru, di AWS Wilayah yang sama seperti yang didukung untuk tipe `G.4X` pekerja.
- Untuk tipe `G.025X` pekerja, setiap pekerja memetakan ke 0,25 DPU (2 vCPU, memori 4 GB) dengan disk 84GB (sekitar 34GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan streaming volume rendah. Jenis pekerja ini hanya tersedia untuk pekerjaan streaming AWS Glue versi 3.0.
- Untuk tipe `Z.2X` pekerja, setiap pekerja memetakan ke 2 M-DPU (8vCPU, memori 64 GB) dengan disk 128 GB (sekitar 120GB gratis), dan menyediakan hingga 8 pekerja Ray berdasarkan autoscaler.

- `CodeGenConfigurationNodes` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8, cocok dengan. Custom string pattern #39

  Setiap nilai adalah sebuah objek CodeGenConfigurationNode A.

  Representasi grafik asiklik terarah yang menjadi dasar komponen visual Glue Studio dan pembuatan kode Glue Studio.

- `ExecutionClass`— String UTF-8, panjangnya tidak lebih dari 16 byte (nilai valid: `FLEX=""` |). `STANDARD=""`

  Menunjukkan apakah pekerjaan dijalankan dengan kelas eksekusi standar atau fleksibel. Kelas eksekusi standar sangat ideal untuk beban kerja yang sensitif terhadap waktu yang membutuhkan startup pekerjaan cepat dan sumber daya khusus.

  Kelas eksekusi fleksibel cocok untuk pekerjaan yang tidak sensitif terhadap waktu yang waktu mulai dan penyelesaiannya dapat bervariasi.

  Hanya pekerjaan dengan AWS Glue versi 3.0 ke atas dan jenis perintah `glueetl` yang diizinkan untuk disetel `ExecutionClass` ke`FLEX`. Kelas eksekusi fleksibel tersedia untuk pekerjaan Spark.

- `SourceControlDetails` — Sebuah objek SourceControlDetails.

  Detail untuk konfigurasi kontrol sumber untuk pekerjaan, memungkinkan sinkronisasi artefak pekerjaan ke atau dari repositori jarak jauh.

- `MaintenanceWindow` — String UTF-8, yang cocok dengan Custom string pattern #30.

  Bidang ini menentukan hari dalam seminggu dan jam untuk jendela pemeliharaan untuk pekerjaan streaming. AWS Glue secara berkala melakukan kegiatan pemeliharaan. Selama jendela pemeliharaan ini, Anda AWS Glue perlu memulai ulang pekerjaan streaming Anda.

  AWS Glue akan memulai kembali pekerjaan dalam waktu 3 jam dari jendela pemeliharaan yang ditentukan. Misalnya, jika Anda mengatur jendela pemeliharaan untuk hari Senin pukul 10:00 GMT, pekerjaan Anda akan dimulai kembali antara 10:00 GMT hingga 1:00 GMT.

- `ProfileName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil AWS Glue penggunaan yang terkait dengan pekerjaan.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama unik yang disediakan untuk definisi tugas ini.

Kesalahan

- `InvalidInputException`
- `IdempotentParameterMismatchException`
- `AlreadyExistsException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`

UpdateJob tindakan (Python: update_job)

Memperbarui sebuah definisi tugas yang ada. Definisi tugas sebelumnya benar-benar ditimpa oleh informasi ini.

Permintaan

- `JobName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi tugas yang akan diperbarui.
- `JobUpdate` — Wajib: Sebuah objek JobUpdate.

  Menentukan nilai-nilai yang dapat digunakan untuk memperbarui definisi tugas. Konfigurasi yang tidak ditentukan dihapus atau diatur ulang ke nilai default.
- `ProfileName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil AWS Glue penggunaan yang terkait dengan pekerjaan.

Respons

- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Mengembalikan nama definisi tugas diperbarui.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

GetJob tindakan (Python: get_job)

Mengambil definisi tugas yang ada.

Permintaan

- `JobName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi tugas yang akan diambil.

Respons

- `Job` — Sebuah objek Pekerjaan.

  Definisi tugas yang diminta.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`

- `OperationTimeoutException`

GetJobs tindakan (Python: get_jobs)

Mengambil semua definisi tugas saat ini.

Permintaan

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Ukuran maksimum respons.

Respons

- `Jobs` – Susunan objek Pekerjaan.

  Daftar definisi tugas.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika tidak semua definisi tugas yang belum dikembalikan.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

DeleteJob tindakan (Python: delete_job)

Menghapus definisi tugas tertentu. Jika definisi tugas tidak ditemukan, tidak ada pengecualian yang dibuang.

Permintaan

- `JobName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi tugas yang akan dihapus.

Respons

- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi tugas yang telah dihapus.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

ListJobs tindakan (Python: list_jobs)

Mengambil nama semua sumber daya pekerjaan di AWS akun ini, atau sumber daya dengan tag yang ditentukan. Operasi ini memungkinkan Anda melihat sumber daya yang tersedia di akun Anda, dan nama-namanya.

Operasi ini mengambil kolom `Tags` opsional, yang dapat Anda gunakan sebagai filter pada respon sehingga tag sumber daya dapat diambil sebagai sebuah grup. Jika Anda memilih untuk menggunakan pem-filter-an tag, maka hanya sumber daya dengan tag saja yang diambil.

Permintaan

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah permintaan kelanjutan.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Ukuran maksimum daftar yang akan dikembalikan.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Menentukan untuk mengembalikan hanya sumber daya ditandai saja.

Respons

- `JobNames` – Susunan string UTF-8.

  Nama-nama semua tugas dalam akun, atau tugas dengan tag yang ditentukan.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika daftar yang dikembalikan tidak berisi metrik terakhir yang tersedia.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchGetJobs tindakan (Python: batch_get_jobs)

Mengembalikan daftar metadata sumber daya untuk daftar tertentu dari nama tugas. Setelah memanggil operasi `ListJobs`, Anda dapat memanggil operasi ini untuk mengakses data yang Anda telah diberikan izinnya. Operasi ini mendukung semua izin IAM, termasuk syarat izin yang menggunakan tag.

Permintaan

- `JobNames` – Wajib: Susunan string UTF-8.

  Daftar nama tugas, mungkin nama yang dikembalikan oleh operasi `ListJobs`.

Respons

- `Jobs` – Susunan objek Pekerjaan.

  Daftar definisi tugas.

- `JobsNotFound` – Susunan string UTF-8.

  Daftar nama tugas yang tidak ditemukan.

Kesalahan

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

Tugas berjalan

Jobs Runs API menjelaskan tipe data dan API yang terkait dengan memulai, menghentikan, atau melihat pekerjaan berjalan, dan mengatur ulang bookmark pekerjaan, di. AWS Glue Riwayat Job run dapat diakses selama 90 hari untuk alur kerja dan pekerjaan Anda.

Jenis data

- JobRun struktur
- Struktur pendahulu
- JobBookmarkEntry struktur
- BatchStopJobRunSuccessfulSubmission struktur
- BatchStopJobRunError struktur
- NotificationProperty struktur

JobRun struktur

Berisi informasi tentang sebuah eksekusi tugas.

Bidang

- `Id` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari eksekusi tugas ini.

- `Attempt` — Nomor (bilangan bulat).

  Jumlah usaha untuk menjalankan tugas ini.

- `PreviousRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari eksekusi sebelumnya dari tugas ini. Misalnya, `JobRunId` yang ditentukan dalam tindakan `StartJobRun`.

- `TriggerName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pemicu yang memulai eksekusi tugas ini.

- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi tugas yang digunakan dalam eksekusi ini.

- `JobMode` – String UTF-8 (nilai yang valid: `SCRIPT=""` | `VISUAL=""` | `NOTEBOOK=""`).

  Mode yang menggambarkan bagaimana pekerjaan dibuat. Nilai yang valid adalah:

  - `SCRIPT`- Pekerjaan dibuat menggunakan editor skrip AWS Glue Studio.
  - `VISUAL`- Pekerjaan dibuat menggunakan editor visual AWS Glue Studio.
  - `NOTEBOOK`- Pekerjaan itu dibuat menggunakan notebook sesi interaktif.

  Ketika `JobMode` bidang hilang atau null, `SCRIPT` ditetapkan sebagai nilai default.

- `StartedOn` — Stempel waktu.

  Tanggal dan waktu saat eksekusi tugas ini dimulai.

- `LastModifiedOn` — Stempel waktu.

  Terakhir kali saat eksekusi tugas ini dijalankan.

- `CompletedOn` — Stempel waktu.

Tanggal dan waktu saat eksekusi tugas ini selesai.

- `JobRunState`— UTF-8 string (nilai valid: `STARTING` | | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT` `ERROR` `WAITING` |`EXPIRED`).

  Status eksekusi tugas saat ini. Untuk informasi lebih lanjut tentang status tugas yang telah dihentikan secara tidak normal, lihat Status Eksekusi Tugas AWS Glue.

- `Arguments` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

  Argumen tugas yang terkait dengan eksekusi ini. Untuk eksekusi tugas ini, mereka mengganti argumen default yang diatur dalam definisi tugas itu sendiri.

  Anda dapat menentukan argumen di sini yang digunakan skrip eksekusi pekerjaan Anda sendiri, serta argumen yang AWS Glue dikonsumsi sendiri.

  Argumen Job dapat dicatat. Jangan berikan rahasia plaintext sebagai argumen. Ambil rahasia dari AWS Glue Connection, AWS Secrets Manager atau mekanisme manajemen rahasia lainnya jika Anda ingin menyimpannya di dalam Job.

  Untuk informasi tentang cara menentukan dan menggunakan argumen Tugas Anda sendiri, lihat topik Memanggil API AWS Glue dalam Python dalam panduan developer.

  Untuk informasi tentang argumen yang dapat Anda berikan ke bidang ini saat mengonfigurasi pekerjaan Spark, lihat Parameter Khusus yang Digunakan menurut AWS Glue topik dalam panduan pengembang.

  Untuk informasi tentang argumen yang dapat Anda berikan ke bidang ini saat mengonfigurasi pekerjaan Ray, lihat Menggunakan parameter pekerjaan di pekerjaan Ray di panduan pengembang.

- `ErrorMessage` – String UTF-8.

  Pesan kesalahan yang terkait dengan eksekusi tugas ini.

- `PredecessorRuns` – Susunan objek Pendahulu.

  Daftar pendahulu untuk eksekusi tugas ini.

- `AllocatedCapacity` — Nomor (bilangan bulat).

  Bidang ini tidak lagi digunakan. Gunakan `MaxCapacity` sebagai gantinya.

  Jumlah unit pemrosesan AWS Glue data (DPU) yang dialokasikan untuk ini. JobRun Dapat dialokasikan dari 2 hingga 100 DPU; default-nya adalah 10. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

- `ExecutionTime` — Nomor (bilangan bulat).

  Jumlah waktu (dalam satuan detik) di mana eksekusi tugas ini menggunakan sumber daya.

- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Waktu habis `JobRun`, dalam satuan menit. Ini adalah waktu maksimum yang bisa digunakan oleh eksekusi tugas untuk menggunakan sumber daya sebelum eksekusi dihentikan dan memasuki status `TIMEOUT`. Nilai ini mengesampingkan nilai batas waktu yang ditetapkan dalam pekerjaan induk.

  Pekerjaan streaming harus memiliki nilai batas waktu kurang dari 7 hari atau 10080 menit. Ketika nilai dibiarkan kosong, pekerjaan akan dimulai ulang setelah 7 hari berdasarkan jika Anda belum menyiapkan jendela pemeliharaan. Jika Anda memiliki jendela pemeliharaan pengaturan, itu akan dimulai ulang selama jendela pemeliharaan setelah 7 hari.

- `MaxCapacity` — Nomor (ganda).

  Untuk Glue versi 1.0 atau pekerjaan sebelumnya, menggunakan tipe pekerja standar, jumlah unit pemrosesan AWS Glue data (DPU) yang dapat dialokasikan saat pekerjaan ini berjalan. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

  Untuk pekerjaan Glue versi 2.0+, Anda tidak dapat menentukan. `Maximum capacity` Sebaliknya, Anda harus menentukan `Worker type` dan `Number of workers`.

  Jangan mengatur `MaxCapacity` jika Anda menggunakan `WorkerType` dan `NumberOfWorkers`.

  Nilai yang dapat dialokasikan untuk `MaxCapacity` tergantung pada apakah Anda menjalankan tugas shell Python, tugas ETL Apache Spark, atau tugas ETL Apache Spark streaming:

  - Ketika anda menentukan tugas shell Python (`JobCommand.Name`="pythonshell"), Anda dapat mengalokasikan 0,0625 atau 1 DPU. Default-nya adalah 0,0625 DPU.

  - Bila Anda menentukan tugas ETL Apache Spark (`JobCommand.Name`="glueetl") atau tugas ETL Apache Spark streaming (`JobCommand.Name`="gluestreaming"), Anda dapat mengalokasikan dari 2 hingga 100 dPU. Default-nya adalah 10 DPU. Jenis tugas ini tidak dapat memiliki alokasi DPU pecahan.
- `WorkerType` – String UTF-8 (nilai yang valid: `Standard`="" | `G.1X`="" | `G.2X`="" | `G.025X`="" | `G.4X`="" | `G.8X`="" | `Z.2X`="").

  Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan ketika sebuah tugas dieksekusi. Menerima nilai G.1X, G.2X, G.4X, G.8X atau G.025X untuk pekerjaan Spark. Menerima nilai Z.2X untuk pekerjaan Ray.

  - Untuk tipe `G.1X` pekerja, setiap pekerja memetakan ke 1 DPU (4 vCPU, memori 16 GB) dengan disk 84GB (sekitar 34GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang skalabel dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
  - Untuk tipe `G.2X` pekerja, setiap pekerja memetakan ke 2 DPU (8 vCPU, memori 32 GB) dengan disk 128GB (sekitar 77GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang skalabel dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
  - Untuk tipe `G.4X` pekerja, setiap pekerja memetakan ke 4 DPU (16 vCPU, memori 64 GB) dengan disk 256GB (sekitar 235GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru di AWS Wilayah berikut: AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (Oregon), Asia Pasifik (Singapura), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Kanada (Tengah), Eropa (Frankfurt), Eropa (Irlandia), dan Eropa (Stockholm).
  - Untuk tipe `G.8X` pekerja, setiap pekerja memetakan ke 8 DPU (32 vCPU, memori 128 GB) dengan disk 512GB (sekitar 487GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru, di AWS Wilayah yang sama seperti yang didukung untuk tipe `G.4X` pekerja.

  - Untuk tipe `G.025X` pekerja, setiap pekerja memetakan ke 0,25 DPU (2 vCPU, memori 4 GB) dengan disk 84GB (sekitar 34GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan streaming volume rendah. Jenis pekerja ini hanya tersedia untuk pekerjaan streaming AWS Glue versi 3.0.
  - Untuk tipe `Z.2X` pekerja, setiap pekerja memetakan ke 2 M-DPU (8vCPU, memori 64 GB) dengan disk 128 GB (sekitar 120GB gratis), dan menyediakan hingga 8 pekerja Ray berdasarkan autoscaler.
- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah pekerja dari `workerType` ditentukan yang dialokasikan ketika sebuah tugas dieksekusi.
- `SecurityConfiguration` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama struktur `SecurityConfiguration` yang akan digunakan dengan eksekusi tugas ini.
- `LogGroupName` – String UTF-8.

  Nama grup log untuk pencatatan aman yang dapat dienkripsi sisi server di Amazon. CloudWatch AWS KMS Nama ini dapat `/aws-glue/jobs/`, dalam hal ini enkripsi default-nya adalah `NONE`. Jika Anda menambahkan nama peran dan nama `SecurityConfiguration` (dengan kata lain, `/aws-glue/jobs-yourRoleName-yourSecurityConfigurationName/`), maka konfigurasi keamanan tersebut digunakan untuk mengenkripsi grup log.
- `NotificationProperty` — Sebuah objek NotificationProperty.

  Menentukan properti konfigurasi dari sebuah notifikasi eksekusi tugas.
- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

  Dalam pekerjaan Spark, `GlueVersion` menentukan versi Apache Spark dan Python yang AWS Glue tersedia dalam suatu pekerjaan. Versi Python menunjukkan versi yang didukung untuk tugas tipe Spark.

  Pekerjaan Ray harus diatur `GlueVersion` ke `4.0` atau lebih besar. Namun, versi Ray, Python, dan pustaka tambahan yang tersedia di pekerjaan Ray Anda ditentukan oleh `Runtime` parameter perintah Job.

  Untuk informasi selengkapnya tentang AWS Glue versi yang tersedia dan versi Spark dan Python yang sesuai, lihat Versi Glue di panduan pengembang.

Tugas yang dibuat tanpa menentukan versi Glue default ke Glue 0.9.

- `DPUSeconds` — Nomor (ganda).

  Bidang ini dapat diatur untuk menjalankan tugas dengan kelas eksekusi `FLEX` atau saat Auto Scaling diaktifkan, dan mewakili total waktu yang dijalankan setiap pelaksana selama siklus hidup pekerjaan yang dijalankan dalam hitungan detik, dikalikan dengan faktor DPU (1 untuk, 2 untuk`G.1X`, atau 0,25 untuk pekerja). `G.2X` `G.025X` Nilai ini mungkin berbeda dari `executionEngineRuntime` * `MaxCapacity` seperti dalam kasus pekerjaan Auto Scaling, karena jumlah pelaksana yang berjalan pada waktu tertentu mungkin kurang dari. `MaxCapacity` Oleh karena itu, ada kemungkinan bahwa nilai `DPUSeconds` kurang dari `executionEngineRuntime` *`MaxCapacity`.

- `ExecutionClass`— String UTF-8, panjangnya tidak lebih dari 16 byte (nilai valid: `FLEX`"" |). `STANDARD`""

  Menunjukkan apakah pekerjaan dijalankan dengan kelas eksekusi standar atau fleksibel. Kelas eksekusi standar sangat ideal untuk beban kerja yang sensitif terhadap waktu yang membutuhkan startup pekerjaan cepat dan sumber daya khusus.

  Kelas eksekusi fleksibel cocok untuk pekerjaan yang tidak sensitif terhadap waktu yang waktu mulai dan penyelesaiannya dapat bervariasi.

  Hanya pekerjaan dengan AWS Glue versi 3.0 ke atas dan jenis perintah `glueetl` yang diizinkan untuk disetel `ExecutionClass` ke`FLEX`. Kelas eksekusi fleksibel tersedia untuk pekerjaan Spark.

- `MaintenanceWindow` — String UTF-8, yang cocok dengan Custom string pattern #30.

  Bidang ini menentukan hari dalam seminggu dan jam untuk jendela pemeliharaan untuk pekerjaan streaming. AWS Glue secara berkala melakukan kegiatan pemeliharaan. Selama jendela pemeliharaan ini, Anda AWS Glue perlu memulai ulang pekerjaan streaming Anda.

  AWS Glue akan memulai kembali pekerjaan dalam waktu 3 jam dari jendela pemeliharaan yang ditentukan. Misalnya, jika Anda mengatur jendela pemeliharaan untuk hari Senin pukul 10:00 GMT, pekerjaan Anda akan dimulai kembali antara 10:00 GMT hingga 1:00 GMT.

- `ProfileName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil AWS Glue penggunaan yang terkait dengan pekerjaan dijalankan.

Struktur pendahulu

Sebuah eksekusi tugas yang digunakan dalam predikat dari sebuah pemicu bersyarat yang memicu eksekusi tugas ini.

Bidang

- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi tugas yang digunakan oleh eksekusi tugas pendahulunya.
- `RunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID eksekusi tugas dari eksekusi tugas pendahulu.

JobBookmarkEntry struktur

Mendefinisikan titik di mana sebuah tugas dapat melanjutkan pengolahan.

Bidang

- `JobName` – String UTF-8.

  Nama tugas yang dimaksud.
- `Version` — Nomor (bilangan bulat).

  Versi tugas.
- `Run` — Nomor (bilangan bulat).

  Nomor ID eksekusi.
- `Attempt` — Nomor (bilangan bulat).

  Nomor ID percobaan.
- `PreviousRunId` – String UTF-8.

  Pengenal eksekusi unik yang terkait dengan eksekusi tugas sebelumnya.
- `RunId` – String UTF-8.

  Nomor ID eksekusi.

- `JobBookmark` – String UTF-8.

  Bookmark itu sendiri.

BatchStopJobRunSuccessfulSubmission struktur

Mencatat permintaan sukses untuk menghentikan `JobRun` yang ditentukan.

Bidang

- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi tugas yang digunakan dalam eksekusi tugas yang dihentikan.
- `JobRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  `JobRunId` dari eksekusi tugas yang telah dihentikan.

BatchStopJobRunError struktur

Catatan kesalahan yang terjadi ketika mencoba untuk menghentikan eksekusi tugas yang ditentukan.

Bidang

- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi tugas yang digunakan dalam eksekusi tugas yang bersangkutan.
- `JobRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  `JobRunId` dari eksekusi tugas yang bersangkutan.
- `ErrorDetail` — Sebuah objek ErrorDetail.

  Menentukan detail tentang kesalahan yang ditemui.

NotificationProperty struktur

Menentukan sifat konfigurasi dari notifikasi.

Bidang

- `NotifyDelayAfter` — Nomor (bilangan bulat), minimal 1.

  Setelah tugas dimulai, jumlah menit untuk menunggu sebelum mengirim notifikasi penundaan tugas.

Operasi

- StartJobRun tindakan (Python: start_job_run)
- BatchStopJobRun tindakan (Python: batch_stop_job_run)
- GetJobRun tindakan (Python: get_job_run)
- GetJobRuns tindakan (Python: get_job_runs)
- GetJobBookmark tindakan (Python: get_job_bookmark)
- GetJobBookmarks tindakan (Python: get_job_bookmarks)
- ResetJobBookmark tindakan (Python: reset_job_bookmark)

StartJobRun tindakan (Python: start_job_run)

Mulai menjalankan sebuah eksekusi tugas dengan menggunakan definisi tugas.

Permintaan

- `JobName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi tugas yang akan digunakan.
- `JobRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari `JobRun` sebelumnya yang akan diulang.
- `Arguments` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8.

Setiap nilai adalah string UTF-8.

Argumen tugas yang terkait dengan eksekusi ini. Untuk eksekusi tugas ini, mereka mengganti argumen default yang diatur dalam definisi tugas itu sendiri.

Anda dapat menentukan argumen di sini yang digunakan skrip eksekusi pekerjaan Anda sendiri, serta argumen yang AWS Glue dikonsumsi sendiri.

Argumen Job dapat dicatat. Jangan berikan rahasia plaintext sebagai argumen. Ambil rahasia dari AWS Glue Connection, AWS Secrets Manager atau mekanisme manajemen rahasia lainnya jika Anda ingin menyimpannya di dalam Job.

Untuk informasi tentang cara menentukan dan menggunakan argumen Tugas Anda sendiri, lihat topik Memanggil API AWS Glue dalam Python dalam panduan developer.

Untuk informasi tentang argumen yang dapat Anda berikan ke bidang ini saat mengonfigurasi pekerjaan Spark, lihat Parameter Khusus yang Digunakan menurut AWS Glue topik dalam panduan pengembang.

Untuk informasi tentang argumen yang dapat Anda berikan ke bidang ini saat mengonfigurasi pekerjaan Ray, lihat Menggunakan parameter pekerjaan di pekerjaan Ray di panduan pengembang.

- `AllocatedCapacity` — Nomor (bilangan bulat).

  Bidang ini tidak lagi digunakan. Gunakan `MaxCapacity` sebagai gantinya.

  Jumlah unit pemrosesan AWS Glue data (DPU) untuk dialokasikan untuk ini. JobRun Anda dapat mengalokasikan minimal 2 DPU; defaultnya adalah 10. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Waktu habis `JobRun`, dalam satuan menit. Ini adalah waktu maksimum yang bisa digunakan oleh eksekusi tugas untuk menggunakan sumber daya sebelum eksekusi dihentikan dan memasuki status `TIMEOUT`. Nilai ini mengesampingkan nilai batas waktu yang ditetapkan dalam pekerjaan induk.

  Pekerjaan streaming harus memiliki nilai batas waktu kurang dari 7 hari atau 10080 menit. Ketika nilai dibiarkan kosong, pekerjaan akan dimulai ulang setelah 7 hari berdasarkan jika Anda belum

menyiapkan jendela pemeliharaan. Jika Anda memiliki jendela pemeliharaan pengaturan, itu akan dimulai ulang selama jendela pemeliharaan setelah 7 hari.

- `MaxCapacity` — Nomor (ganda).

  Untuk Glue versi 1.0 atau pekerjaan sebelumnya, menggunakan tipe pekerja standar, jumlah unit pemrosesan AWS Glue data (DPU) yang dapat dialokasikan saat pekerjaan ini berjalan. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

  Untuk pekerjaan Glue versi 2.0+, Anda tidak dapat menentukan. `Maximum capacity` Sebaliknya, Anda harus menentukan `Worker type` dan `Number of workers`.

  Jangan mengatur `MaxCapacity` jika Anda menggunakan `WorkerType` dan `NumberOfWorkers`.

  Nilai yang dapat dialokasikan untuk `MaxCapacity` tergantung pada apakah Anda menjalankan tugas shell Python, tugas ETL Apache Spark, atau tugas ETL Apache Spark streaming:

  - Ketika anda menentukan tugas shell Python (`JobCommand.Name="pythonshell"`), Anda dapat mengalokasikan 0,0625 atau 1 DPU. Default-nya adalah 0,0625 DPU.
  - Bila Anda menentukan tugas ETL Apache Spark (`JobCommand.Name="glueetl"`) atau tugas ETL Apache Spark streaming (`JobCommand.Name="gluestreaming"`), Anda dapat mengalokasikan dari 2 hingga 100 dPU. Default-nya adalah 10 DPU. Jenis tugas ini tidak dapat memiliki alokasi DPU pecahan.
- `SecurityConfiguration` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama struktur `SecurityConfiguration` yang akan digunakan dengan eksekusi tugas ini.
- `NotificationProperty` — Sebuah objek NotificationProperty.

  Menentukan properti konfigurasi dari sebuah notifikasi eksekusi tugas.
- `WorkerType` – String UTF-8 (nilai yang valid: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan ketika sebuah tugas dieksekusi. Menerima nilai G.1X, G.2X, G.4X, G.8X atau G.025X untuk pekerjaan Spark. Menerima nilai Z.2X untuk pekerjaan Ray.

  - Untuk tipe `G.1X` pekerja, setiap pekerja memetakan ke 1 DPU (4 vCPU, memori 16 GB) dengan disk 84GB (sekitar 34GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami

merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang skalabel dan hemat biaya untuk menjalankan sebagian besar pekerjaan.

- Untuk tipe `G.2X` pekerja, setiap pekerja memetakan ke 2 DPU (8 vCPU, memori 32 GB) dengan disk 128GB (sekitar 77GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang skalabel dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
- Untuk tipe `G.4X` pekerja, setiap pekerja memetakan ke 4 DPU (16 vCPU, memori 64 GB) dengan disk 256GB (sekitar 235GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru di AWS Wilayah berikut: AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (Oregon), Asia Pasifik (Singapura), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Kanada (Tengah), Eropa (Frankfurt), Eropa (Irlandia), dan Eropa (Stockholm).
- Untuk tipe `G.8X` pekerja, setiap pekerja memetakan ke 8 DPU (32 vCPU, memori 128 GB) dengan disk 512GB (sekitar 487GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru, di AWS Wilayah yang sama seperti yang didukung untuk tipe `G.4X` pekerja.
- Untuk tipe `G.025X` pekerja, setiap pekerja memetakan ke 0,25 DPU (2 vCPU, memori 4 GB) dengan disk 84GB (sekitar 34GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan streaming volume rendah. Jenis pekerja ini hanya tersedia untuk pekerjaan streaming AWS Glue versi 3.0.
- Untuk tipe `Z.2X` pekerja, setiap pekerja memetakan ke 2 M-DPU (8vCPU, memori 64 GB) dengan disk 128 GB (sekitar 120GB gratis), dan menyediakan hingga 8 pekerja Ray berdasarkan autoscaler.

- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah pekerja dari `workerType` ditentukan yang dialokasikan ketika sebuah tugas dieksekusi.

- `ExecutionClass`— String UTF-8, panjangnya tidak lebih dari 16 byte (nilai valid: `FLEX`"" |). `STANDARD`""

Menunjukkan apakah pekerjaan dijalankan dengan kelas eksekusi standar atau fleksibel. Kelas eksekusi standar sangat ideal untuk beban kerja yang sensitif terhadap waktu yang membutuhkan startup pekerjaan cepat dan sumber daya khusus.

Kelas eksekusi fleksibel cocok untuk pekerjaan yang tidak sensitif terhadap waktu yang waktu mulai dan penyelesaiannya dapat bervariasi.

Hanya pekerjaan dengan AWS Glue versi 3.0 ke atas dan jenis perintah `glueetl` yang diizinkan untuk disetel `ExecutionClass` ke`FLEX`. Kelas eksekusi fleksibel tersedia untuk pekerjaan Spark.

- `ProfileName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil AWS Glue penggunaan yang terkait dengan pekerjaan dijalankan.

Respons

- `JobRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID yang ditetapkan untuk eksekusi tugas ini.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentRunsExceededException`

BatchStopJobRun tindakan (Python: batch_stop_job_run)

Menghentikan satu atau beberapa eksekusi tugas untuk definisi tugas tertentu.

Permintaan

- `JobName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi tugas yang akan menghentikan eksekusi tugas.
- `JobRunIds` – Wajib: Susunan string UTF-8, tidak kurang dari 1 atau lebih dari 25 string.

  Daftar `JobRunIds` yang harus dihentikan untuk definisi tugas itu.

Respons

- `SuccessfulSubmissions` – Susunan objek BatchStopJobRunSuccessfulSubmission.

  Daftar JobRuns yang berhasil diajukan untuk dihentikan.
- `Errors` – Susunan objek BatchStopJobRunError.

  Daftar kesalahan yang ditemui dalam mencoba untuk menghentikan `JobRuns`, termasuk `JobRunId` yang ditemui kesalahannya dan detail tentang kesalahannya.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetJobRun tindakan (Python: get_job_run)

Mengambil metadata untuk eksekusi tugas tertentu. Riwayat Job run dapat diakses selama 90 hari untuk alur kerja dan pekerjaan Anda.

Permintaan

- `JobName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi tugas yang sedang dijalankan.

- `RunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari eksekusi tugas.
- `PredecessorsIncluded` – Boolean.

  BETUL jika daftar eksekusi pendahulu harus dikembalikan.

Respons

- `JobRun` — Sebuah objek JobRun.

  Metadata eksekusi tugas yang diminta.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetJobRuns tindakan (Python: get_job_runs)

Mengambil metadata untuk semua eksekusi tugas dari definisi tugas tertentu.

Permintaan

- `JobName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama definisi tugas untuk yang akan diambil semua eksekusi tugas-nya.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.
- `MaxResults`— Angka (bilangan bulat), tidak kurang dari 1 atau lebih dari 200.

  Ukuran maksimum respons.

Respons

- `JobRuns` – Susunan objek JobRun.

  Daftar objek metadata eksekusi tugas.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika bukan semua eksekusi tugas yang diminta yang telah dikembalikan.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetJobBookmark tindakan (Python: get_job_bookmark)

Mengembalikan informasi tentang entri bookmark tugas.

Untuk informasi selengkapnya tentang mengaktifkan dan menggunakan bookmark pekerjaan, lihat:

- Melacak data yang diproses menggunakan bookmark pekerjaan
- Parameter Job yang digunakan oleh AWS Glue
- Struktur Job

Permintaan

- `JobName` – Wajib: String UTF-8.

  Nama tugas yang dimaksud.

- `Version` — Nomor (bilangan bulat).

  Versi tugas.

- `RunId` – String UTF-8.

  Pengenal unik eksekusi yang terkait dengan eksekusi tugas ini.

Respons

- `JobBookmarkEntry` — Sebuah objek JobBookmarkEntry.

  Sebuah struktur yang mendefinisikan titik di mana sebuah tugas dapat melanjutkan pengolahan.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ValidationException`

GetJobBookmarks tindakan (Python: get_job_bookmarks)

Mengembalikan informasi tentang entri bookmark tugas. Daftar ini diurutkan berdasarkan nomor versi secara menurun.

Untuk informasi selengkapnya tentang mengaktifkan dan menggunakan bookmark pekerjaan, lihat:

- Melacak data yang diproses menggunakan bookmark pekerjaan
- Parameter Job yang digunakan oleh AWS Glue
- Struktur Job

Permintaan

- `JobName` – Wajib: String UTF-8.

  Nama tugas yang dimaksud.

- `MaxResults` — Nomor (bilangan bulat).

  Ukuran maksimum respons.

- `NextToken` — Nomor (bilangan bulat).

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Respons

- `JobBookmarkEntries` – Susunan objek JobBookmarkEntry.

  Daftar entri bookmark tugas yang mendefinisikan titik di mana tugas dapat melanjutkan pemrosesan.

- `NextToken` — Nomor (bilangan bulat).

  Sebuah token kelanjutan, yang memiliki nilai 1 jika semua entri dikembalikan, atau > 1 jika tidak semua eksekusi tugas yang diminta telah dikembalikan.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

ResetJobBookmark tindakan (Python: reset_job_bookmark)

Mengatur ulang sebuah entri bookmark.

Untuk informasi selengkapnya tentang mengaktifkan dan menggunakan bookmark pekerjaan, lihat:

- Melacak data yang diproses menggunakan bookmark pekerjaan
- Parameter Job yang digunakan oleh AWS Glue
- Struktur Job

Permintaan

- `JobName` – Wajib: String UTF-8.

  Nama tugas yang dimaksud.

- `RunId` – String UTF-8.

  Pengenal unik eksekusi yang terkait dengan eksekusi tugas ini.

Respons

- `JobBookmarkEntry` — Sebuah objek JobBookmarkEntry.

  Entri bookmark pengaturan ulang.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

Pemicu

Triggers API menjelaskan tipe data dan API yang terkait dengan pembuatan, pembaruan, atau penghapusan, serta memulai dan menghentikan pemicu pekerjaan. AWS Glue

Jenis data

- Struktur pemicu
- TriggerUpdate struktur
- Struktur predikat
- Struktur kondisi
- Struktur aksi
- EventBatchingCondition struktur

Struktur pemicu

Informasi tentang sebuah pemicu tertentu.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pemicu.

- `WorkflowName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang dikaitkan dengan pemicu.

- `Id` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Disimpan untuk digunakan di masa depan.

- `Type` – String UTF-8 (nilai yang valid: `SCHEDULED` | `CONDITIONAL` | `ON_DEMAND` | `EVENT`).

  Jenis pemicu yang ini.

- `State` – String UTF-8 (nilai yang valid: `CREATING` | `CREATED` | `ACTIVATING` | `ACTIVATED` | `DEACTIVATING` | `DEACTIVATED` | `DELETING` | `UPDATING`).

  Status pemicu saat ini.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi pemicu ini.

- `Schedule` – String UTF-8.

  Sebuah ekspresi `cron` yang digunakan untuk menentukan jadwal (lihat Jadwal Berbasis Waktu untuk Tugas dan Crawler. Sebagai contoh, untuk menjalankan sesuatu setiap hari pada 12:15 UTC, Anda harus menentukan `cron(15 12 * * ? *)`.

- `Actions` – Susunan objek Tindakan.

  Tindakan yang dimulai oleh pemicu ini.

- `Predicate` — Sebuah objek Predikat.

  Predikat pemicu ini, yang menentukan kapan akan aktif.

- `EventBatchingCondition` — Sebuah objek EventBatchingKondisi.

  Kondisi batch yang harus dipenuhi (jumlah peristiwa tertentu yang diterima atau jendela waktu batch kedaluwarsa) sebelum pemicu EventBridge peristiwa terjadi kebakaran.

TriggerUpdate struktur

Struktur yang digunakan untuk memberikan informasi yang digunakan untuk memperbarui sebuah pemicu. Objek ini memperbarui definisi pemicu sebelumnya dengan menimpa semuanya.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Disimpan untuk digunakan di masa depan.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi pemicu ini.

- `Schedule` – String UTF-8.

  Sebuah ekspresi `cron` yang digunakan untuk menentukan jadwal (lihat Jadwal Berbasis Waktu untuk Tugas dan Crawler. Sebagai contoh, untuk menjalankan sesuatu setiap hari pada 12:15 UTC, Anda harus menentukan `cron(15 12 * * ? *)`.

- `Actions` – Susunan objek Tindakan.

  Tindakan yang dimulai oleh pemicu ini.

- `Predicate` — Sebuah objek Predikat.

  Predikat pemicu ini, yang menentukan kapan akan aktif.

- `EventBatchingCondition` — Sebuah objek EventBatchingKondisi.

  Kondisi batch yang harus dipenuhi (jumlah peristiwa tertentu yang diterima atau jendela waktu batch kedaluwarsa) sebelum pemicu EventBridge peristiwa terjadi kebakaran.

Struktur predikat

Menentukan predikat pemicu, yang menentukan kapan terjadi inisiasi.

Bidang

- `Logical` – String UTF-8 (nilai yang valid: `AND` | `ANY`).

Bidang opsional hanya jika satu kondisi tercantum. Jika beberapa kondisi tercantum, bidang ini diperlukan.

- `Conditions` – Susunan objek Syarat.

  Daftar syarat yang menentukan kapan pemicu akan aktif.

Struktur kondisi

Menentukan kondisi yang memicu inisiasi.

Bidang

- `LogicalOperator` – String UTF-8 (nilai yang valid: `EQUALS`).

  Operator logika.

- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tugas yang `JobRuns`-nya berlaku syarat ini, dan di mana pemicu ini menunggu.

- `State`— UTF-8 string (nilai valid: `STARTING` | | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT` `ERROR` `WAITING` |`EXPIRED`).

  Status syarat. Saat ini, satu-satunya status tugas yang didengarkan oleh pemicu adalah `SUCCEEDED`, `STOPPED`, `FAILED`, dan `TIMEOUT`. Satu-satunya status crawler yang didengarkan oleh pemicu adalah `SUCCEEDED`, `FAILED`, dan `CANCELLED`.

- `CrawlerName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama crawler yang berlaku syarat ini padanya.

- `CrawlState` – String UTF-8 (nilai valid: `RUNNING` | `CANCELLING` | `CANCELLED` | `SUCCEEDED` | `FAILED` | `ERROR`).

  Status crawler yang berlaku syarat ini padanya.

Struktur aksi

Menentukan tindakan yang dimulai oleh pemicu.

Bidang

- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tugas yang akan dijalankan.
- `Arguments` – Susunan peta pasangan nilai kunci.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

  Argumen tugas yang digunakan saat pemicu ini aktif. Agar tugas ini berjalan, argumen default yang diatur dalam ketentuan tugas itu sendiri diganti.

  Anda dapat menentukan argumen di sini yang digunakan skrip eksekusi pekerjaan Anda sendiri, serta argumen yang AWS Glue dikonsumsi sendiri.

  Untuk informasi tentang cara menentukan dan menggunakan argumen Tugas Anda sendiri, lihat topik Memanggil API AWS Glue dalam Python dalam panduan developer.

  Untuk informasi tentang pasangan kunci-nilai yang AWS Glue digunakan untuk menyiapkan pekerjaan Anda, lihat Parameter Khusus yang Digunakan menurut AWS Glue topik dalam panduan pengembang.
- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Waktu habis `JobRun`, dalam satuan menit. Ini adalah waktu maksimum yang bisa digunakan oleh eksekusi tugas untuk menggunakan sumber daya sebelum eksekusi dihentikan dan memasuki status `TIMEOUT`. Nilai default-nya adalah 2.880 menit (48 jam). Hal ini menimpa nilai habis waktu yang ditetapkan dalam tugas induk.
- `SecurityConfiguration` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama struktur `SecurityConfiguration` yang akan digunakan dengan tindakan ini.
- `NotificationProperty` — Sebuah objek NotificationProperty.

  Menentukan properti konfigurasi dari sebuah notifikasi eksekusi tugas.
- `CrawlerName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Nama crawler yang akan digunakan dengan tindakan ini.

EventBatchingCondition struktur

Kondisi batch yang harus dipenuhi (jumlah peristiwa tertentu yang diterima atau jendela waktu batch kedaluwarsa) sebelum pemicu EventBridge peristiwa terjadi kebakaran.

Bidang

- `BatchSize` — Wajib: Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100.

  Jumlah peristiwa yang harus diterima dari Amazon EventBridge sebelum EventBridge peristiwa memicu kebakaran.
- `BatchWindow` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 900.

  Jendela waktu dalam hitungan detik setelah EventBridge peristiwa memicu kebakaran. Jendela dimulai ketika peristiwa pertama diterima.

Operasi

- CreateTrigger tindakan (Python: create_trigger)
- StartTrigger tindakan (Python: start_trigger)
- GetTrigger tindakan (Python: get_trigger)
- GetTriggers tindakan (Python: get_trigger)
- UpdateTrigger tindakan (Python: update_trigger)
- StopTrigger tindakan (Python: stop_trigger)
- DeleteTrigger tindakan (Python: delete_trigger)
- ListTriggers tindakan (Python: list_trigger)
- BatchGetTriggers tindakan (Python: batch_get_trigger)

CreateTrigger tindakan (Python: create_trigger)

Menciptakan sebuah pemicu baru.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pemicu.

- `WorkflowName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang dikaitkan dengan pemicu.

- `Type` – Wajib: String UTF-8 (nilai yang valid: `SCHEDULED` | `CONDITIONAL` | `ON_DEMAND` | `EVENT`).

  Jenis pemicu baru.

- `Schedule` – String UTF-8.

  Sebuah ekspresi `cron` yang digunakan untuk menentukan jadwal (lihat Jadwal Berbasis Waktu untuk Tugas dan Crawler. Sebagai contoh, untuk menjalankan sesuatu setiap hari pada 12:15 UTC, Anda harus menentukan `cron(15 12 * * ? *)`.

  Bidang ini wajib ketika jenis pemicu-nya adalah TERJADWAL.

- `Predicate` — Sebuah objek Predikat.

  Sebuah predikat untuk menentukan kapan pemicu baru harus aktif.

  Bidang ini wajib ketika jenis pemicu-nya adalah `CONDITIONAL`.

- `Actions` – Wajib: Susunan objek Tindakan.

  Tindakan yang dimulai oleh pemicu ini saat pemicu tersebut aktif.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah deskripsi dari pemicu baru.

- `StartOnCreation` – Boolean.

  Atur ke `true` untuk memulai pemicu `SCHEDULED` dan `CONDITIONAL` ketika dibuat. BETUL tidak didukung untuk pemicu `ON_DEMAND`.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

Tag yang akan digunakan dengan pemicu ini. Anda dapat menggunakan tag untuk membatasi akses ke pemicu tersebut. Untuk informasi selengkapnya tentang AWS tag AWS Glue, lihat Tag AWS Glue di panduan pengembang.

- `EventBatchingCondition` — Sebuah objek EventBatchingKondisi.

  Kondisi batch yang harus dipenuhi (jumlah peristiwa tertentu yang diterima atau jendela waktu batch kedaluwarsa) sebelum pemicu EventBridge peristiwa terjadi kebakaran.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pemicu.

Kesalahan

- `AlreadyExistsException`
- `EntityNotFoundException`
- `InvalidInputException`
- `IdempotentParameterMismatchException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`

StartTrigger tindakan (Python: start_trigger)

Memulai pemicu yang ada. Lihat Memicu Tugas untuk informasi tentang bagaimana berbagai jenis pemicu dimulai.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pemicu yang akan dimulai.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pemicu yang sudah dimulai.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentRunsExceededException`

GetTrigger tindakan (Python: get_trigger)

Mengambil definisi dari sebuah pemicu.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pemicu yang akan diambil.

Respons

- `Trigger` — Sebuah objek Pemicu.

Definisi pemicu yang diminta.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

GetTriggers tindakan (Python: get_trigger)

Mendapatkan semua pemicu yang dikaitkan dengan sebuah tugas.

Permintaan

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

- `DependentJobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tugas yang untuknya pemicu akan diambil. Pemicu yang dapat memulai tugas ini dikembalikan, dan jika tidak ada pemicu seperti itu, maka semua pemicu dikembalikan.

- `MaxResults`— Angka (bilangan bulat), tidak kurang dari 1 atau lebih dari 200.

  Ukuran maksimum respons.

Respons

- `Triggers` – Susunan objek Pemicu.

  Daftar pemicu untuk tugas yang ditentukan.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika bukan semua pemicu yang diminta yang belum dikembalikan.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

UpdateTrigger tindakan (Python: update_trigger)

Memperbarui sebuah definisi pemicu.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pemicu yang akan diperbarui.
- `TriggerUpdate` — Wajib: Sebuah objek TriggerUpdate.

  Nilai-nilai baru yang digunakan untuk memperbarui pemicu.

Respons

- `Trigger` — Sebuah objek Pemicu.

  Definisi pemicu yang dihasilkan.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

StopTrigger tindakan (Python: stop_trigger)

Menghentikan pemicu tertentu.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pemicu yang akan dihentikan.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pemicu yang sudah dihentikan.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

DeleteTrigger tindakan (Python: delete_trigger)

Menghapus pemicu tertentu. Jika pemicu tidak ditemukan, tidak ada pengecualian yang dibuang.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pemicu yang akan dihapus.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pemicu yang sudah dihapus.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

ListTriggers tindakan (Python: list_trigger)

Mengambil nama semua sumber daya pemicu di AWS akun ini, atau sumber daya dengan tag yang ditentukan. Operasi ini memungkinkan Anda melihat sumber daya yang tersedia di akun Anda, dan nama-namanya.

Operasi ini mengambil kolom `Tags` opsional, yang dapat Anda gunakan sebagai filter pada respon sehingga tag sumber daya dapat diambil sebagai sebuah grup. Jika Anda memilih untuk menggunakan pem-filter-an tag, maka hanya sumber daya dengan tag saja yang diambil.

Permintaan

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah permintaan kelanjutan.
- `DependentJobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama tugas yang untuknya pemicu akan diambil. Pemicu yang dapat memulai tugas ini dikembalikan. Jika tidak ada pemicu seperti itu, maka semua pemicu dikembalikan.
- `MaxResults`— Angka (bilangan bulat), tidak kurang dari 1 atau lebih dari 200.

  Ukuran maksimum daftar yang akan dikembalikan.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

Menentukan untuk mengembalikan hanya sumber daya ditandai saja.

Respons

- `TriggerNames` – Susunan string UTF-8.

  Nama dari semua pemicu dalam akun, atau pemicu dengan tag yang ditentukan.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika daftar yang dikembalikan tidak berisi metrik terakhir yang tersedia.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchGetTriggers tindakan (Python: batch_get_trigger)

Mengembalikan daftar metadata sumber daya untuk daftar nama pemicu tertentu. Setelah memanggil operasi `ListTriggers`, Anda dapat memanggil operasi ini untuk mengakses data yang Anda telah diberikan izinnya. Operasi ini mendukung semua izin IAM, termasuk syarat izin yang menggunakan tag.

Permintaan

- `TriggerNames` – Wajib: Susunan string UTF-8.

  Daftar nama pemicu, mungkin nama yang dikembalikan dari operasi `ListTriggers`.

Respons

- `Triggers` – Susunan objek Pemicu.

  Daftar definisi pemicu.
- `TriggersNotFound` – Susunan string UTF-8.

  Daftar nama pemicu yang tidak ditemukan.

Kesalahan

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

API sesi interaktif

API sesi interaktif menjelaskan AWS Glue API yang terkait dengan penggunaan sesi AWS Glue interaktif untuk membangun dan menguji skrip ekstrak, transformasi, dan pemuatan (ETL) untuk integrasi data.

Jenis data

- Struktur sesi
- SessionCommand struktur
- Struktur pernyataan
- StatementOutput struktur
- StatementOutputData struktur
- ConnectionsList struktur

Struktur sesi

Periode di mana lingkungan runtime Spark jarak jauh berjalan.

Bidang

- `Id` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID sesi.

- `CreatedOn` — Stempel waktu.

  Waktu dan tanggal saat sesi dibuat.

- `Status` – String UTF-8 (nilai valid: `PROVISIONING` | `READY` | `FAILED` | `TIMEOUT` | `STOPPING` | `STOPPED`).

  Status sesi.

- `ErrorMessage` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Pesan kesalahan ditampilkan selama sesi.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi sesi.

- `Role`— String UTF-8, panjangnya tidak kurang dari 20 atau lebih dari 2048 byte, cocok dengan. Custom string pattern #26

  Nama atau Nama Sumber Daya Amazon (ARN) dari peran IAM yang terkait dengan Sesi.

- `Command` — Sebuah objek SessionCommand.

  Perintah Object.see. SessionCommand

- `DefaultArguments`— Sebuah array peta pasangan kunci-nilai, tidak lebih dari 75 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Setiap nilai adalah string UTF-8, panjangnya tidak lebih dari 4096 byte, cocok dengan. URI address multi-line string pattern

  Sebuah array peta pasangan kunci-nilai. Max adalah 75 pasang.

- `Connections` — Sebuah objek ConnectionsList.

Jumlah koneksi yang digunakan untuk sesi tersebut.

- `Progress` — Nomor (ganda).

  Progres eksekusi kode sesi.

- `MaxCapacity` — Nomor (ganda).

  Jumlah unit pemrosesan AWS Glue data (DPU) yang dapat dialokasikan saat pekerjaan berjalan. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan memori 16 GB.

- `SecurityConfiguration` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama SecurityConfiguration struktur yang akan digunakan dengan sesi.

- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

  AWS Glue Versi ini menentukan versi Apache Spark dan Python yang mendukung. AWS Glue GlueVersion Harus lebih besar dari 2.0.

- `DataAccessId`— String UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 36 byte.

  ID akses data sesi.

- `PartitionId`— String UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 36 byte.

  ID partisi dari sesion.

- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah pekerja yang ditentukan `WorkerType` untuk digunakan untuk sesi tersebut.

- `WorkerType` – String UTF-8 (nilai yang valid: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Jenis pekerja standar yang dialokasikan saat sesi berjalan. Menerima nilai`G.1X`,, `G.2X``G.4X`, atau `G.8X` untuk sesi Spark. Menerima nilai `Z.2X` untuk sesi Ray.

- `CompletedOn` — Stempel waktu.

  Tanggal dan waktu sesi ini selesai.

- `ExecutionTime` — Nomor (ganda).

Total waktu sesi berjalan.

- `DPUSeconds` — Nomor (ganda).

  DPU yang dikonsumsi oleh sesi (rumus: ExecutionTime * MaxCapacity).

- `IdleTimeout` — Nomor (bilangan bulat).

  Jumlah menit saat idle sebelum sesi habis.

- `ProfileName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil AWS Glue penggunaan yang terkait dengan sesi.

SessionCommand struktur

`SessionCommand`Yang menjalankan pekerjaan.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Menentukan nama. SessionCommand Bisa 'glueetl' atau 'gluestreaming'.

- `PythonVersion` — String UTF-8, yang cocok dengan Custom string pattern #21.

  Menentukan versi Python. Versi Python menunjukkan versi yang didukung untuk tugas tipe Spark.

Struktur pernyataan

Pernyataan atau permintaan untuk tindakan tertentu terjadi dalam suatu sesi.

Bidang

- `Id` — Nomor (bilangan bulat).

  ID pernyataan.

- `Code` – String UTF-8.

  Kode eksekusi pernyataan.

- `State` – String UTF-8 (nilai valid: `WAITING` | `RUNNING` | `AVAILABLE` | `CANCELLING` | `CANCELLED` | `ERROR`).

  Negara saat permintaan ditindaklanjuti.
- `Output` — Sebuah objek StatementOutput.

  Output di JSON.
- `Progress` — Nomor (ganda).

  Progres eksekusi kode.
- `StartedOn` — Nomor (panjang).

  Waktu dan tanggal unix bahwa definisi pekerjaan dimulai.
- `CompletedOn` — Nomor (panjang).

  Waktu dan tanggal unix bahwa definisi pekerjaan selesai.

StatementOutput struktur

Output eksekusi kode dalam format JSON.

Bidang

- `Data` — Sebuah objek StatementOutputData.

  Output eksekusi kode.
- `ExecutionCount` — Nomor (bilangan bulat).

  Hitungan eksekusi output.
- `Status` – String UTF-8 (nilai valid: `WAITING` | `RUNNING` | `AVAILABLE` | `CANCELLING` | `CANCELLED` | `ERROR`).

  Status output eksekusi kode.
- `ErrorName` – String UTF-8.

  Nama kesalahan dalam output.
- `ErrorValue` – String UTF-8.

  Nilai kesalahan output.

- `Traceback` – Susunan string UTF-8.

  Penelusuran balik output.

StatementOutputData struktur

Output eksekusi kode dalam format JSON.

Bidang

- `TextPlain` – String UTF-8.

  Output eksekusi kode dalam format teks.

ConnectionsList struktur

Menentukan koneksi yang digunakan oleh tugas.

Bidang

- `Connections` – Susunan string UTF-8.

  Daftar koneksi yang digunakan oleh tugas.

Operasi

- CreateSession tindakan (Python: create_session)
- StopSession tindakan (Python: stop_session)
- DeleteSession tindakan (Python: delete_session)
- GetSession tindakan (Python: get_session)
- ListSessions tindakan (Python: list_sessions)
- RunStatement tindakan (Python: run_statement)
- CancelStatement tindakan (Python: cancel_statement)
- GetStatement tindakan (Python: get_statement)
- ListStatements tindakan (Python: list_statement)

CreateSession tindakan (Python: create_session)

Membuat sesi baru.

Permintaan

Permintaan untuk membuat sesi baru.

- `Id` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID permintaan sesi.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi sesi.
- `Role`- Wajib: UTF-8 string, tidak kurang dari 20 atau lebih dari 2048 byte panjang, cocok dengan. Custom string pattern #26

  Peran IAM ARN
- `Command` — Wajib: Sebuah objek SessionCommand.

  `SessionCommand`Yang menjalankan pekerjaan.
- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Jumlah menit sebelum waktu sesi habis. Default untuk pekerjaan Spark ETL adalah 48 jam (2880 menit), masa pakai sesi maksimum untuk jenis pekerjaan ini. Konsultasikan dokumentasi untuk jenis pekerjaan lainnya.
- `IdleTimeout` — Nomor (bilangan bulat), minimal 1.

  Jumlah menit saat idle sebelum waktu sesi habis. Default untuk pekerjaan Spark ETL adalah nilai Timeout. Konsultasikan dokumentasi untuk jenis pekerjaan lainnya.
- `DefaultArguments`— Sebuah array peta pasangan kunci-nilai, tidak lebih dari 75 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Setiap nilai adalah string UTF-8, panjangnya tidak lebih dari 4096 byte, cocok dengan. URI address multi-line string pattern

Sebuah array peta pasangan kunci-nilai. Max adalah 75 pasang.

- `Connections` — Sebuah objek ConnectionsList.

  Jumlah koneksi yang akan digunakan untuk sesi.

- `MaxCapacity` — Nomor (ganda).

  Jumlah unit pemrosesan AWS Glue data (DPU) yang dapat dialokasikan saat pekerjaan berjalan. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan memori 16 GB.

- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah pekerja yang ditentukan `WorkerType` untuk digunakan untuk sesi tersebut.

- `WorkerType` – String UTF-8 (nilai yang valid: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan ketika sebuah tugas dieksekusi. Menerima nilai G.1X, G.2X, G.4X, atau G.8X untuk pekerjaan Spark. Menerima nilai Z.2X untuk notebook Ray.

  - Untuk tipe `G.1X` pekerja, setiap pekerja memetakan ke 1 DPU (4 vCPU, memori 16 GB) dengan disk 84GB (sekitar 34GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang terukur dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
  - Untuk tipe `G.2X` pekerja, setiap pekerja memetakan ke 2 DPU (8 vCPU, memori 32 GB) dengan disk 128GB (sekitar 77GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk beban kerja seperti transformasi data, gabungan, dan kueri, untuk menawarkan cara yang terukur dan hemat biaya untuk menjalankan sebagian besar pekerjaan.
  - Untuk tipe `G.4X` pekerja, setiap pekerja memetakan ke 4 DPU (16 vCPU, memori 64 GB) dengan disk 256GB (sekitar 235GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru di AWS Wilayah berikut: AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (Oregon), Asia Pasifik (Singapura), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Kanada (Tengah), Eropa (Frankfurt), Eropa (Irlandia), dan Eropa (Stockholm).

  - Untuk tipe `G.8X` pekerja, setiap pekerja memetakan ke 8 DPU (32 vCPU, memori 128 GB) dengan disk 512GB (sekitar 487GB gratis), dan menyediakan 1 eksekutor per pekerja. Kami merekomendasikan jenis pekerja ini untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Jenis pekerja ini hanya tersedia untuk pekerjaan Spark ETL AWS Glue versi 3.0 atau yang lebih baru, di AWS Wilayah yang sama seperti yang didukung untuk tipe `G.4X` pekerja.
  - Untuk tipe `Z.2X` pekerja, setiap pekerja memetakan ke 2 M-DPU (8vCPU, memori 64 GB) dengan disk 128 GB (sekitar 120GB gratis), dan menyediakan hingga 8 pekerja Ray berdasarkan autoscaler.
- `SecurityConfiguration` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama SecurityConfiguration struktur yang akan digunakan dengan sesi
- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

  AWS Glue Versi ini menentukan versi Apache Spark dan Python yang mendukung. AWS Glue GlueVersion Harus lebih besar dari 2.0.
- `DataAccessId`— String UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 36 byte.

  ID akses data sesi.
- `PartitionId`— String UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 36 byte.

  ID partisi sesi.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Peta pasangan nilai kunci (tag) milik sesi.
- `RequestOrigin` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Asal usul permintaan.

- `ProfileName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil AWS Glue penggunaan yang terkait dengan sesi.

Respons

- `Session` — Sebuah objek Sesi.

  Mengembalikan objek sesi dalam respon.

Kesalahan

- `AccessDeniedException`
- `IdempotentParameterMismatchException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ValidationException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`

StopSession tindakan (Python: stop_session)

Menghentikan sesi.

Permintaan

- `Id` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID sesi yang akan dihentikan.
- `RequestOrigin` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Asal usul permintaan.

Respons

- `Id` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Mengembalikan Id dari sesi berhenti.

Kesalahan

- `AccessDeniedException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`
- `ConcurrentModificationException`

DeleteSession tindakan (Python: delete_session)

Menghapus sesi.

Permintaan

- `Id` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID sesi yang akan dihapus.
- `RequestOrigin` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Nama asal permintaan sesi hapus.

Respons

- `Id` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Mengembalikan ID dari sesi dihapus.

Kesalahan

- `AccessDeniedException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`
- `ConcurrentModificationException`

GetSession tindakan (Python: get_session)

Mengambil sesi.

Permintaan

- `Id` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID sesi.
- `RequestOrigin` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Asal usul permintaan.

Respons

- `Session` — Sebuah objek Sesi.

  Objek sesi dikembalikan dalam respons.

Kesalahan

- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

- `InvalidInputException`

ListSessions tindakan (Python: list_sessions)

Ambil daftar sesi.

Permintaan

- `NextToken`- String UTF-8, panjangnya tidak lebih dari 400000 byte.

  Token untuk set hasil berikutnya, atau null jika tidak ada hasil lagi.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah hasil maksimum.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Tag milik sesi.
- `RequestOrigin` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Asal usul permintaan.

Respons

- `Ids` – Susunan string UTF-8.

  Mengembalikan ID sesi.
- `Sessions` – Susunan objek Sesi.

  Mengembalikan objek sesi.
- `NextToken`- String UTF-8, panjangnya tidak lebih dari 400000 byte.

  Token untuk set hasil berikutnya, atau null jika tidak ada hasil lagi.

Kesalahan

- `AccessDeniedException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

RunStatement tindakan (Python: run_statement)

Mengeksekusi pernyataan.

Permintaan

- `SessionId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Id Sesi dari pernyataan yang akan dijalankan.
- `Code`- Diperlukan: string UTF-8, panjangnya tidak lebih dari 68000 byte.

  Kode pernyataan yang akan dijalankan.
- `RequestOrigin` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Asal usul permintaan.

Respons

- `Id` — Nomor (bilangan bulat).

  Mengembalikan Id dari pernyataan yang dijalankan.

Kesalahan

- `EntityNotFoundException`
- `AccessDeniedException`
- `InternalServiceException`
- `OperationTimeoutException`

- `InvalidInputException`
- `ValidationException`
- `ResourceNumberLimitExceededException`
- `IllegalSessionStateException`

CancelStatement tindakan (Python: cancel_statement)

Membatalkan pernyataan.

Permintaan

- `SessionId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Sesi dari pernyataan yang akan dibatalkan.
- `Id` — Wajib: Nomor (bilangan bulat).

  ID pernyataan yang akan dibatalkan.
- `RequestOrigin` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Asal usul permintaan untuk membatalkan pernyataan.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`

GetStatement tindakan (Python: get_statement)

Mengambil pernyataan.

Permintaan

- `SessionId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Sesi pernyataan.
- `Id` — Wajib: Nomor (bilangan bulat).

  Id dari pernyataan tersebut.
- `RequestOrigin` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Asal usul permintaan.

Respons

- `Statement` — Sebuah objek Pernyataan.

  Mengembalikan pernyataan.

Kesalahan

- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`

ListStatements tindakan (Python: list_statement)

Daftar pernyataan untuk sesi.

Permintaan

- `SessionId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID Sesi dari pernyataan.
- `RequestOrigin` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Asal usul permintaan untuk membuat daftar pernyataan.
- `NextToken`- String UTF-8, panjangnya tidak lebih dari 400000 byte.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Respons

- `Statements` – Susunan objek Pernyataan.

  Mengembalikan daftar pernyataan.
- `NextToken`- String UTF-8, panjangnya tidak lebih dari 400000 byte.

  Token kelanjutan, jika tidak semua pernyataan telah dikembalikan.

Kesalahan

- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `IllegalSessionStateException`

API titik akhir pengembangan

API titik akhir Pengembangan menjelaskan AWS Glue API yang terkait dengan pengujian menggunakan kustom DevEndpoint.

Jenis data

- DevEndpoint struktur
- DevEndpointCustomLibraries struktur

DevEndpoint struktur

Sebuah titik akhir pengembangan di mana developer secara jarak jauh dapat men-debug skrip extract, transform, and load (ETL).

Bidang

- `EndpointName` – String UTF-8.

  Nama `DevEndpoint`.
- `RoleArn` — String UTF-8, yang cocok dengan AWS IAM ARN string pattern.

  Amazon Resource Name (ARN) dari IAM role yang digunakan di `DevEndpoint` ini.
- `SecurityGroupIds` – Susunan string UTF-8.

  Daftar pengidentifikasi grup keamanan yang digunakan dalam `DevEndpoint` ini.
- `SubnetId` – String UTF-8.

  ID subnet untuk `DevEndpoint` ini.
- `YarnEndpointAddress` – String UTF-8.

  Alamat titik akhir YARN yang digunakan oleh `DevEndpoint` ini.
- `PrivateAddress` – String UTF-8.

  Alamat IP privat untuk mengakses `DevEndpoint` dalam VPC jika `DevEndpoint` dibuat dalam satu. Kolom `PrivateAddress` hanya ada ketika Anda membuat `DevEndpoint` dalam VPC Anda.
- `ZeppelinRemoteSparkInterpreterPort` — Nomor (bilangan bulat).

  Port Apache Zeppelin untuk penerjemah Apache Spark jarak jauh.
- `PublicAddress` – String UTF-8.

  Alamat IP publik yang digunakan oleh `DevEndpoint` ini. Kolom `PublicAddress` ini hanya ada ketika Anda membuat `DevEndpoint` non-virtual private cloud (VPC).

- `Status` – String UTF-8.

  Status `DevEndpoint` saat ini.

- `WorkerType` – String UTF-8 (nilai yang valid: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan untuk titik akhir pengembangan. Menerima nilai Standard, G.1X, atau G.2X.

  - Untuk jenis pekerja `Standard`, setiap pekerja menyediakan 4 vCPU, memori 16 GB dan disk 50GB, dan 2 pelaksana per pekerja.
  - Untuk jenis pekerja `G.1X`, setiap pekerja memetakan ke 1 DPU (4 vCPU, memori 16 GB, disk 64 GB), dan menyediakan 1 pelaksana per pekerja. Kami merekomendasikan jenis pekerja ini untuk tugas yang membutuhkan banyak memori.
  - Untuk jenis pekerja `G.2X`, setiap pekerja memetakan ke 2 DPU (8 vCPU, memori 32 GB, disk 128 GB), dan menyediakan 1 pelaksana per pekerja. Kami merekomendasikan jenis pekerja ini untuk tugas yang membutuhkan banyak memori.

  Masalah yang diketahui: ketika titik akhir pengembangan dibuat dengan konfigurasi `G.2X` `WorkerType`, driver Spark untuk titik akhir pengembangan akan berjalan pada 4 vCPU, 16 GB memori, dan 64 GB disk.

- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

  Versi Glue menentukan versi Apache Spark dan Python yang mendukung. AWS Glue Versi Python menunjukkan versi yang didukung untuk skrip ETL berjalan Anda pada titik akhir pengembangan.

  Untuk informasi selengkapnya tentang AWS Glue versi yang tersedia dan versi Spark dan Python yang sesuai, lihat Versi Glue di panduan pengembang.

  Titik akhir pengembangan yang dibuat tanpa menentukan versi default Glue untuk GLue 0.9.

  Anda dapat menentukan versi support Python untuk titik akhir pengembangan dengan menggunakan parameter `Arguments` dalam API `CreateDevEndpoint` atau `UpdateDevEndpoint`. Jika tidak ada argumen yang disediakan, versi default menjadi Python 2.

- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah pekerja dari `workerType` yang ditetapkan yang dialokasikan ke titik akhir pengembangan.

Jumlah maksimum pekerja yang dapat Anda tentukan adalah 299 untuk `G.1X`, dan 149 untuk `G.2X`.

- `NumberOfNodes` — Nomor (bilangan bulat).

  Jumlah Unit Pemrosesan AWS Glue Data (DPU) yang dialokasikan untuk ini. `DevEndpoint`

- `AvailabilityZone` – String UTF-8.

  AWS Availability Zone di mana `DevEndpoint` ini berada.

- `VpcId` – String UTF-8.

  ID Virtual Private Cloud (VPC) yang digunakan oleh `DevEndpoint`.

- `ExtraPythonLibsS3Path` – String UTF-8.

  Path ke satu atau beberapa perpustakaan Python dalam sebuah bucket Amazon S3 yang harus dimuat di `DevEndpoint` Anda. Beberapa nilai harus berupa jalur lengkap yang dipisahkan dengan koma.

  > **Note**
  >
  > Anda hanya dapat menggunakan perpustakaan Python murni dengan `DevEndpoint`. Perpustakaan yang mengandalkan ekstensi C, seperti perpustakaan analisis data Python, yakni pandas, saat ini tidak didukung.

- `ExtraJarsS3Path` – String UTF-8.

  Path satu atau beberapa file `.jar` Java dalam bucket S3 yang seharusnya dimuat di `DevEndpoint` Anda.

  > **Note**
  >
  > Anda hanya dapat menggunakan perpustakaan murni Java/Scala dengan `DevEndpoint`.

- `FailureReason` – String UTF-8.

  Alasan kegagalan saat ini dalam `DevEndpoint` ini.

- `LastUpdateStatus` – String UTF-8.

  Status pembaruan terakhir.

- `CreatedTimestamp` — Stempel waktu.

  Titik waktu di mana ini DevEndpoint dibuat.

- `LastModifiedTimestamp` — Stempel waktu.

  Titik waktu di mana `DevEndpoint` ini terakhir diubah.

- `PublicKey` – String UTF-8.

  Kunci publik yang akan digunakan oleh `DevEndpoint` ini untuk autentikasi. Atribut ini disediakan untuk kompatibilitas karena atribut yang direkomendasikan untuk digunakan adalah kunci publik.

- `PublicKeys` — Susunan string UTF-8, tidak lebih dari 5 string.

  Daftar kunci publik yang akan digunakan oleh `DevEndpoints` untuk autentikasi. Menggunakan atribut ini lebih disukai daripada kunci publik tunggal karena kunci publik ini memungkinkan Anda untuk memiliki kunci privat yang berbeda untuk setiap klien.

  > **Note**
  >
  > Jika Anda sebelumnya membuat titik akhir dengan sebuah kunci publik, Anda harus menghapus kunci tersebut untuk dapat mengatur daftar kunci publik. Panggil operasi API `UpdateDevEndpoint` dengan konten kunci publik di atribut `deletePublicKeys`, dan daftar kunci baru di atribut `addPublicKeys`.

- `SecurityConfiguration` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama struktur `SecurityConfiguration` yang akan digunakan dengan `DevEndpoint` ini.

- `Arguments` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 100 pasang.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

  Sebuah peta argumen yang digunakan untuk mengkonfigurasi `DevEndpoint`.

  Argumen valid adalah:

  - `"--enable-glue-datacatalog": ""`

Anda dapat menentukan versi support Python untuk titik akhir pengembangan dengan menggunakan parameter `Arguments` dalam API `CreateDevEndpoint` atau `UpdateDevEndpoint`. Jika tidak ada argumen yang disediakan, versi default menjadi Python 2.

DevEndpointCustomLibraries struktur

Perpustakaan kustom yang akan dimuat ke titik akhir pengembangan.

Bidang

- `ExtraPythonLibsS3Path` – String UTF-8.

  Path ke satu atau beberapa perpustakaan Python dalam sebuah bucket Amazon Simple Storage Service (Amazon S3) yang harus dimuat di `DevEndpoint` Anda. Beberapa nilai harus path lengkap yang dipisahkan dengan koma.

  > **Note**
  >
  > Anda hanya dapat menggunakan perpustakaan Python murni dengan `DevEndpoint`. Perpustakaan yang mengandalkan ekstensi C, seperti perpustakaan analisis data Python, yakni pandas, saat ini tidak didukung.

- `ExtraJarsS3Path` – String UTF-8.

  Path satu atau beberapa file `.jar` Java dalam bucket S3 yang seharusnya dimuat di `DevEndpoint` Anda.

  > **Note**
  >
  > Anda hanya dapat menggunakan perpustakaan murni Java/Scala dengan `DevEndpoint`.

Operasi

- CreateDevEndpoint tindakan (Python: create_dev_endpoint)
- UpdateDevEndpoint tindakan (Python: update_dev_endpoint)
- DeleteDevEndpoint tindakan (Python: delete_dev_endpoint)

- GetDevEndpoint tindakan (Python: get_dev_endpoint)
- GetDevEndpoints tindakan (Python: get_dev_endpoints)
- BatchGetDevEndpoints tindakan (Python: batch_get_dev_endpoints)
- ListDevEndpoints tindakan (Python: list_dev_endpoints)

CreateDevEndpoint tindakan (Python: create_dev_endpoint)

Menciptakan sebuah titik akhir pengembangan baru.

Permintaan

- `EndpointName` – Wajib: String UTF-8.

  Nama yang akan ditetapkan untuk `DevEndpoint` baru.
- `RoleArn` — Wajib: String UTF-8, yang cocok dengan AWS IAM ARN string pattern.

  IAM role untuk fungsi `DevEndpoint`.
- `SecurityGroupIds` – Susunan string UTF-8.

  ID grup keamanan untuk grup keamanan yang akan digunakan oleh `DevEndpoint`.
- `SubnetId` – String UTF-8.

  ID subnet untuk `DevEndpoint` yang baru yang akan digunakan.
- `PublicKey` – String UTF-8.

  Kunci publik yang akan digunakan oleh `DevEndpoint` ini untuk autentikasi. Atribut ini disediakan untuk kompatibilitas karena atribut yang direkomendasikan untuk digunakan adalah kunci publik.
- `PublicKeys` — Susunan string UTF-8, tidak lebih dari 5 string.

  Daftar kunci publik yang akan digunakan oleh titik akhir pengembangan untuk autentikasi. Penggunaan atribut ini lebih disukai daripada kunci publik tunggal karena kunci publik ini memungkinkan Anda untuk memiliki kunci privat yang berbeda untuk setiap klien.

> **Note**
>
> Jika Anda sebelumnya membuat titik akhir dengan sebuah kunci publik, Anda harus menghapus kunci tersebut untuk dapat mengatur daftar kunci publik. Panggil API

UpdateDevEndpoint dengan konten kunci publik di atribut deletePublicKeys, dan daftar kunci baru di atribut addPublicKeys.

- NumberOfNodes — Nomor (bilangan bulat).

  Jumlah Unit Pemrosesan AWS Glue Data (DPU) yang akan dialokasikan untuk ini. DevEndpoint

- WorkerType – String UTF-8 (nilai yang valid: Standard="" | G.1X="" | G.2X="" | G.025X="" | G.4X="" | G.8X="" | Z.2X="").

  Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan untuk titik akhir pengembangan. Menerima nilai Standard, G.1X, atau G.2X.

  - Untuk jenis pekerja Standard, setiap pekerja menyediakan 4 vCPU, memori 16 GB dan disk 50GB, dan 2 pelaksana per pekerja.
  - Untuk jenis pekerja G.1X, setiap pekerja memetakan ke 1 DPU (4 vCPU, memori 16 GB, disk 64 GB), dan menyediakan 1 pelaksana per pekerja. Kami merekomendasikan jenis pekerja ini untuk tugas yang membutuhkan banyak memori.
  - Untuk jenis pekerja G.2X, setiap pekerja memetakan ke 2 DPU (8 vCPU, memori 32 GB, disk 128 GB), dan menyediakan 1 pelaksana per pekerja. Kami merekomendasikan jenis pekerja ini untuk tugas yang membutuhkan banyak memori.

  Masalah yang diketahui: ketika titik akhir pengembangan dibuat dengan konfigurasi G.2X WorkerType, driver Spark untuk titik akhir pengembangan akan berjalan pada 4 vCPU, 16 GB memori, dan 64 GB disk.

- GlueVersion — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

  Versi Glue menentukan versi Apache Spark dan Python yang mendukung. AWS Glue Versi Python menunjukkan versi yang didukung untuk skrip ETL berjalan Anda pada titik akhir pengembangan.

  Untuk informasi selengkapnya tentang AWS Glue versi yang tersedia dan versi Spark dan Python yang sesuai, lihat Versi Glue di panduan pengembang.

  Titik akhir pengembangan yang dibuat tanpa menentukan versi default Glue untuk GLue 0.9.

  Anda dapat menentukan versi support Python untuk titik akhir pengembangan dengan menggunakan parameter Arguments dalam API CreateDevEndpoint atau UpdateDevEndpoint. Jika tidak ada argumen yang disediakan, versi default menjadi Python 2.

- NumberOfWorkers — Nomor (bilangan bulat).

Jumlah pekerja dari `workerType` yang ditetapkan yang dialokasikan ke titik akhir pengembangan.

Jumlah maksimum pekerja yang dapat Anda tentukan adalah 299 untuk `G.1X`, dan 149 untuk `G.2X`.

- `ExtraPythonLibsS3Path` – String UTF-8.

  Path ke satu atau beberapa perpustakaan Python dalam sebuah bucket Amazon S3 yang harus dimuat di `DevEndpoint` Anda. Beberapa nilai harus berupa jalur lengkap yang dipisahkan dengan koma.

  > **Note**
  >
  > Anda hanya dapat menggunakan perpustakaan Python murni dengan `DevEndpoint`. Perpustakaan yang mengandalkan ekstensi C, seperti perpustakaan analisis data Python, yakni pandas, saat ini tidak didukung.

- `ExtraJarsS3Path` – String UTF-8.

  Path satu atau beberapa file `.jar` Java dalam bucket S3 yang seharusnya dimuat di `DevEndpoint` Anda.

- `SecurityConfiguration` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama struktur `SecurityConfiguration` yang akan digunakan dengan `DevEndpoint` ini.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Tag untuk digunakan dengan ini DevEndpoint. Anda dapat menggunakan tag untuk membatasi akses ke file DevEndpoint. Untuk informasi selengkapnya tentang AWS tag AWS Glue, lihat Tag AWS Glue di panduan pengembang.

- `Arguments` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 100 pasang.

  Setiap kunci adalah string UTF-8.

Setiap nilai adalah string UTF-8.

Sebuah peta argumen yang digunakan untuk mengkonfigurasi `DevEndpoint`.

Respons

- `EndpointName` – String UTF-8.

  Nama yang ditetapkan untuk `DevEndpoint` baru.
- `Status` – String UTF-8.

  Status `DevEndpoint` baru saat ini.
- `SecurityGroupIds` – Susunan string UTF-8.

  Grup keamanan yang ditetapkan ke `DevEndpoint`.
- `SubnetId` – String UTF-8.

  ID subnet yang ditetapkan ke `DevEndpoint`.
- `RoleArn` — String UTF-8, yang cocok dengan AWS IAM ARN string pattern.

  Amazon Resource Name (ARN) dari peran yang ditetapkan ke `DevEndpoint` baru.
- `YarnEndpointAddress` – String UTF-8.

  Alamat dari titik akhir YARN yang digunakan oleh `DevEndpoint` ini.
- `ZeppelinRemoteSparkInterpreterPort` — Nomor (bilangan bulat).

  Port Apache Zeppelin untuk penerjemah Apache Spark jarak jauh.
- `NumberOfNodes` — Nomor (bilangan bulat).

  Jumlah Unit Pemrosesan AWS Glue Data (DPU) yang dialokasikan untuk ini. DevEndpoint
- `WorkerType` – String UTF-8 (nilai yang valid: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan untuk titik akhir pengembangan. Mungkin nilai Standard, G.1X, atau G.2X.
- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

Versi Glue menentukan versi Apache Spark dan Python yang mendukung. AWS Glue Versi Python menunjukkan versi yang didukung untuk skrip ETL berjalan Anda pada titik akhir pengembangan.

Untuk informasi selengkapnya tentang AWS Glue versi yang tersedia dan versi Spark dan Python yang sesuai, lihat Versi Glue di panduan pengembang.

- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah pekerja dari `workerType` yang ditetapkan yang dialokasikan ke titik akhir pengembangan.

- `AvailabilityZone` – String UTF-8.

  AWS Availability Zone di mana `DevEndpoint` ini berada.

- `VpcId` – String UTF-8.

  ID Virtual Private Cloud (VPC) yang digunakan oleh `DevEndpoint`.

- `ExtraPythonLibsS3Path` – String UTF-8.

  Path ke satu atau beberapa perpustakaan Python dalam sebuah bucket S3 yang harus dimuat di `DevEndpoint` Anda.

- `ExtraJarsS3Path` – String UTF-8.

  Path satu atau beberapa file `.jar` Java dalam bucket S3 yang akan dimuat di `DevEndpoint` Anda.

- `FailureReason` – String UTF-8.

  Alasan kegagalan saat ini dalam `DevEndpoint` ini.

- `SecurityConfiguration` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama struktur `SecurityConfiguration` yang sedang digunakan dengan `DevEndpoint` ini.

- `CreatedTimestamp` — Stempel waktu.

  Titik waktu di mana `DevEndpoint` ini diciptakan.

- `Arguments` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 100 pasang.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

Peta argumen yang digunakan untuk mengkonfigurasi `DevEndpoint` ini.

Argumen valid adalah:

- `"--enable-glue-datacatalog": ""`

Anda dapat menentukan versi support Python untuk titik akhir pengembangan dengan menggunakan parameter `Arguments` dalam API `CreateDevEndpoint` atau `UpdateDevEndpoint`. Jika tidak ada argumen yang disediakan, versi default menjadi Python 2.

Kesalahan

- `AccessDeniedException`
- `AlreadyExistsException`
- `IdempotentParameterMismatchException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `ValidationException`
- `ResourceNumberLimitExceededException`

UpdateDevEndpoint tindakan (Python: update_dev_endpoint)

Pembaruan titik akhir pengembangan yang ditentukan.

Permintaan

- `EndpointName` – Wajib: String UTF-8.

  Nama `DevEndpoint` yang diperbarui.

- `PublicKey` – String UTF-8.

  Kunci publik untuk `DevEndpoint` yang akan digunakan.

- `AddPublicKeys` — Susunan string UTF-8, tidak lebih dari 5 string.

  Daftar kunci publik untuk `DevEndpoint` yang akan digunakan.

- `DeletePublicKeys` — Susunan string UTF-8, tidak lebih dari 5 string.

  Daftar kunci publik yang akan dihapus dari `DevEndpoint`.

- `CustomLibraries` — Sebuah objek DevEndpointCustomLibraries.

  Perpustakaan Python atau Java kustom yang akan dimuat di `DevEndpoint`.

- `UpdateEtlLibraries` – Boolean.

  `True` jika daftar perpustakaan kustom yang akan dimuat di titik akhir pengembangan perlu diperbarui, atau `False` jika sebaliknya.

- `DeleteArguments` – Susunan string UTF-8.

  Daftar kunci argumen yang akan dihapus dari peta argumen yang digunakan untuk mengkonfigurasi `DevEndpoint`.

- `AddArguments` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 100 pasang.

  Setiap kunci adalah string UTF-8.

  Setiap nilai adalah string UTF-8.

  Peta argumen untuk menambahkan peta argumen yang digunakan untuk mengkonfigurasi `DevEndpoint`.

  Argumen valid adalah:

  - `"--enable-glue-datacatalog": ""`

  Anda dapat menentukan versi support Python untuk titik akhir pengembangan dengan menggunakan parameter `Arguments` dalam API `CreateDevEndpoint` atau `UpdateDevEndpoint`. Jika tidak ada argumen yang disediakan, versi default menjadi Python 2.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InternalServiceException`

- `OperationTimeoutException`
- `InvalidInputException`
- `ValidationException`

DeleteDevEndpoint tindakan (Python: delete_dev_endpoint)

Menghapus titik akhir pengembangan yang ditentukan.

Permintaan

- `EndpointName` – Wajib: String UTF-8.

  Nama `DevEndpoint`.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

GetDevEndpoint tindakan (Python: get_dev_endpoint)

Mengambil informasi tentang titik akhir pengembangan yang ditentukan.

> **Note**
>
> Saat Anda membuat titik akhir pengembangan di Virtual Private Cloud (VPC), AWS Glue hanya mengembalikan alamat IP privat saja, dan kolom alamat IP publik tidak diisi. Saat Anda membuat endpoint pengembangan non-VPC, hanya AWS Glue mengembalikan alamat IP publik.

Permintaan

- `EndpointName` – Wajib: String UTF-8.

  Nama `DevEndpoint` dimana informasi akan diambil.

Respons

- `DevEndpoint` — Sebuah objek DevEndpoint.

  Sebuah definisi `DevEndpoint`.

Kesalahan

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

GetDevEndpoints tindakan (Python: get_dev_endpoints)

Mengambil semua titik akhir pengembangan di akun ini AWS .

> **Note**
>
> Saat Anda membuat titik akhir pengembangan di Virtual Private Cloud (VPC), AWS Glue hanya mengembalikan alamat IP privat saja, dan kolom alamat IP publik tidak diisi. Saat Anda membuat endpoint pengembangan non-VPC, hanya AWS Glue mengembalikan alamat IP publik.

Permintaan

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Ukuran maksimum informasi yang akan dikembalikan.

- `NextToken` – String UTF-8.

Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Respons

- `DevEndpoints` – Susunan objek DevEndpoint.

  Daftar definisi `DevEndpoint`.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika tidak semua definisi `DevEndpoint` belum dikembalikan.

Kesalahan

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

BatchGetDevEndpoints tindakan (Python: batch_get_dev_endpoints)

Mengembalikan daftar metadata sumber daya untuk daftar yang nama titik akhir pengembangan yang ditentukan. Setelah memanggil operasi `ListDevEndpoints`, Anda dapat memanggil operasi ini untuk mengakses data yang Anda telah diberikan izinnya. Operasi ini mendukung semua izin IAM, termasuk syarat izin yang menggunakan tag.

Permintaan

- `customerAccountId` – String UTF-8.

  ID AWS akun.
- `DevEndpointNames` – Wajib: Susunan string UTF-8, tidak kurang dari 1 atau lebih dari 25 string.

  Daftar nama `DevEndpoint`, mungkin nama yang dikembalikan oleh operasi `ListDevEndpoint`.

Respons

- `DevEndpoints` – Susunan objek DevEndpoint.

Daftar definisi `DevEndpoint`.

- `DevEndpointsNotFound` – Susunan string UTF-8, tidak kurang dari 1 atau lebih dari 25 string.

  Daftar `DevEndpoints` tidak ditemukan.

Kesalahan

- `AccessDeniedException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

ListDevEndpoints tindakan (Python: list_dev_endpoints)

Mengambil nama-nama semua sumber daya `DevEndpoint` dalam akun AWS ini, atau sumber daya dengan tag yang ditentukan. Operasi ini memungkinkan Anda melihat sumber daya yang tersedia di akun Anda, dan nama-namanya.

Operasi ini mengambil kolom `Tags` opsional, yang dapat Anda gunakan sebagai filter pada respon sehingga tag sumber daya dapat diambil sebagai sebuah grup. Jika Anda memilih untuk menggunakan pem-filter-an tag, maka hanya sumber daya dengan tag saja yang diambil.

Permintaan

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah permintaan kelanjutan.

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Ukuran maksimum daftar yang akan dikembalikan.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Menentukan untuk mengembalikan hanya sumber daya ditandai saja.

Respons

- `DevEndpointNames` – Susunan string UTF-8.

  Nama dari semua `DevEndpoints` dalam akun, atau `DevEndpoints` dengan tag yang ditentukan.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika daftar yang dikembalikan tidak berisi metrik terakhir yang tersedia.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

Registri skema

API registri Skema menjelaskan tipe data dan API yang terkait dengan bekerja dengan skema di. AWS Glue

Jenis data

- RegistryId struktur
- RegistryListItem struktur
- MetadataInfo struktur
- OtherMetadataValueListItem struktur
- SchemaListItem struktur
- SchemaVersionListItem struktur
- MetadataKeyValuePair struktur
- SchemaVersionErrorItem struktur
- ErrorDetails struktur
- SchemaVersionNumber struktur
- SchemaId struktur

RegistryId struktur

Pembungkus struktur yang mungkin berisi nama registri dan Amazon Resource Name (ARN).

Bidang

- `RegistryName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama registri. Digunakan hanya untuk pencarian. Salah satu dari `RegistryArn` atau `RegistryName` harus disediakan.
- `RegistryArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Arn dari registri yang akan diperbarui. Salah satu dari `RegistryArn` atau `RegistryName` harus disediakan.

RegistryListItem struktur

Struktur yang berisi detail untuk sebuah registri.

Bidang

- `RegistryName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama registri.
- `RegistryArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) registri.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah deskripsi registri.
- `Status` – String UTF-8 (nilai yang valid: `AVAILABLE` | `DELETING`).

  Status registri.
- `CreatedTime` – String UTF-8.

Registri data telah dibuat.

- `UpdatedTime` – String UTF-8.

  Tanggal saat registri diperbarui.

MetadataInfo struktur

Struktur yang berisi informasi metadata untuk sebuah versi skema.

Bidang

- `MetadataValue` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 256 byte, yang cocok dengan Custom string pattern #33.

  Nilai yang sesuai kunci metadata.

- `CreatedTime` – String UTF-8.

  Waktu saat entri dibuat.

- `OtherMetadataValueList` – Susunan objek OtherMetadataValueListItem.

  Metadata lainnya yang merupakan milik dari kunci metadata yang sama.

OtherMetadataValueListItem struktur

Struktur yang berisi metadata lain untuk sebuah versi skema yang merupakan milik dari kunci metadata yang sama.

Bidang

- `MetadataValue` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 256 byte, yang cocok dengan Custom string pattern #33.

  Nilai yang sesuai kunci metadata untuk metadata lainnya yang merupakan milik dari kunci metadata yang sama.

- `CreatedTime` – String UTF-8.

  Waktu saat entri dibuat.

SchemaListItem struktur

Sebuah objek yang berisi detail minimal untuk sebuah skema.

Bidang

- `RegistryName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama registri tempat skema berada.

- `SchemaName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama skema.

- `SchemaArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) untuk skema.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah deskripsi untuk skema.

- `SchemaStatus` – String UTF-8 (nilai yang valid: `AVAILABLE` | `PENDING` | `DELETING`).

  Status skema.

- `CreatedTime` – String UTF-8.

  Tanggal dan waktu saat sebuah skema dibuat.

- `UpdatedTime` – String UTF-8.

  Tanggal dan waktu saat sebuah skema diperbarui.

SchemaVersionListItem struktur

Sebuah objek yang berisi detail tentang sebuah versi skema.

Bidang

- `SchemaArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari skema.
- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  Pengenal unik dari versi skema.
- `VersionNumber` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Nomor versi dari skema.
- `Status` – String UTF-8 (nilai yang valid: `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Status versi skema.
- `CreatedTime` – String UTF-8.

  Tanggal dan waktu saat versi skema dibuat.

MetadataKeyValuePair struktur

Sebuah Struktur yang berisi pasangan nilai kunci untuk metadata.

Bidang

- `MetadataKey` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #33.

  Sebuah kunci metadata.
- `MetadataValue` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 256 byte, yang cocok dengan Custom string pattern #33.

  Sebuah nilai yang sesuai kunci metadata.

SchemaVersionErrorItem struktur

Sebuah objek yang berisi detail kesalahan untuk sebuah pada sebuah versi skema.

Bidang

- `VersionNumber` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Nomor versi dari skema.
- `ErrorDetails` — Sebuah objek ErrorDetails.

  Detail kesalahan untuk versi skema.

ErrorDetails struktur

Sebuah objek yang berisi detail kesalahan.

Bidang

- `ErrorCode` – String UTF-8.

  Kode kesalahan untuk sebuah kesalahan.
- `ErrorMessage` – String UTF-8.

  Pesan kesalahan untuk sebuah kesalahan.

SchemaVersionNumber struktur

Sebuah struktur yang berisi informasi skema.

Bidang

- `LatestVersion` – Boolean.

  Versi terbaru yang tersedia untuk skema.
- `VersionNumber` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Nomor versi dari skema.

SchemaId struktur

ID unik skema dalam registri AWS Glue skema.

Bidang

- `SchemaArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari skema. Salah satu dari `SchemaArn` atau `SchemaName` harus disediakan.
- `SchemaName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama skema. Salah satu dari `SchemaArn` atau `SchemaName` harus disediakan.
- `RegistryName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama registri skema yang berisi skema.

Operasi

- CreateRegistry tindakan (Python: create_registry)
- CreateSchema tindakan (Python: create_schema)
- GetSchema tindakan (Python: get_schema)
- ListSchemaVersions tindakan (Python: list_schema_versions)
- GetSchemaVersion tindakan (Python: get_schema_version)
- GetSchemaVersionsDiff tindakan (Python: get_schema_versions_diff)
- ListRegistries tindakan (Python: list_registries)
- ListSchemas tindakan (Python: list_schemas)
- RegisterSchemaVersion tindakan (Python: register_schema_version)
- UpdateSchema tindakan (Python: update_schema)
- CheckSchemaVersionValidity tindakan (Python: check_schema_version_validity)
- UpdateRegistry tindakan (Python: update_registry)
- GetSchemaByDefinition tindakan (Python: get_schema_by_definition)
- GetRegistry tindakan (Python: get_registry)
- PutSchemaVersionMetadata tindakan (Python: put_schema_version_metadata)
- QuerySchemaVersionMetadata tindakan (Python: query_schema_version_metadata)

- RemoveSchemaVersionMetadata tindakan (Python: remove_schema_version_metadata)
- DeleteRegistry tindakan (Python: delete_registry)
- DeleteSchema tindakan (Python: delete_schema)
- DeleteSchemaVersions tindakan (Python: delete_schema_versions)

CreateRegistry tindakan (Python: create_registry)

Menciptakan sebuah registri baru yang dapat digunakan untuk menyimpan sekumpulan skema.

Permintaan

- `RegistryName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama registri yang akan dibuat dengan panjang maksimal 255 karakter, dan mungkin hanya berisi huruf, angka, tanda hubung, garis bawah, tanda dolar, atau tanda hash. Tanpa spasi.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah deskripsi registri. Jika deskripsi tidak tersedia, maka tidak akan ada nilai default untuk ini.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  AWS tag yang berisi pasangan nilai kunci dan dapat dicari berdasarkan konsol, baris perintah, atau API.

Respons

- `RegistryArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari registri yang baru saja dibuat.
- `RegistryName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

Nama registri.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah deskripsi registri.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Tag untuk registri.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`
- `InternalServiceException`

CreateSchema tindakan (Python: create_schema)

Menciptakan sebuah set skema baru dan mendaftar definisi skema. Mengembalikan kesalahan jika set skema sudah ada tanpa benar-benar mendaftarkan versi.

Ketika set skema dibuat, pos pemeriksaan versi akan diatur ke versi pertama. Mode kompatibilitas "DINONAKTIFKAN" membatasi versi skema tambahan agar tidak ditambahkan setelah versi skema pertama. Untuk semua mode kompatibilitas lainnya, validasi pengaturan kompatibilitas akan diterapkan hanya dari versi kedua dan seterusnya ketika API `RegisterSchemaVersion` digunakan.

Ketika API ini dipanggil tanpa `RegistryId`, hal ini akan membuat entri untuk sebuah "default-registry" dalam tabel basis data registri, jika tidak sudah ada.

Permintaan

- `RegistryId` — Sebuah objek RegistryId.

  Ini adalah sebuah bentuk pembungkus berisi bidang identitas registri. Jika ini tidak tersedia, maka registri default akan digunakan. Format ARN untuk hal tersebut adalah: `arn:aws:glue:us-east-2:<customer id>:registry/default-registry:random-5-letter-id`.

- `SchemaName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama skema yang akan dibuat dengan panjang maksimal 255 karakter, dan mungkin hanya berisi huruf, angka, tanda hubung, garis bawah, tanda dolar, atau tanda hash. Tanpa spasi.

- `DataFormat` – Wajib: String UTF-8 (nilai yang valid: `AVRO` | `JSON` | `PROTOBUF`).

  Format data dari definisi skema. Saat ini`AVRO`, `JSON` dan `PROTOBUF` didukung.

- `Compatibility` – String UTF-8 (nilai yang valid: `NONE` | `DISABLED` | `BACKWARD` | `BACKWARD_ALL` | `FORWARD` | `FORWARD_ALL` | `FULL` | `FULL_ALL`).

  Mode kompatibilitas skema. Nilai yang mungkin adalah:

  - NONE: Tidak ada mode kompatibilitas yang berlaku. Anda dapat menggunakan pilihan ini dalam skenario pengembangan atau jika Anda tidak tahu mode kompatibilitas yang ingin Anda terapkan untuk skema. Setiap versi baru yang ditambahkan akan diterima tanpa menjalani pemeriksaan kompatibilitas terlebih dahulu.
  - DISABLED: Pilihan kompatibilitas ini mencegah versioning untuk skema tertentu. Anda dapat menggunakan pilihan ini untuk mencegah dilakukannya versioning masa depan pada sebuah skema.
  - MUNDUR: Pilihan kompatibilitas ini dianjurkan karena memungkinkan penerima data untuk membaca versi skema saat ini dan versi skema sebelumnya. Ini berarti bahwa misalnya, versi skema baru tidak dapat membuang bidang data atau mengubah jenis bidang ini, sehingga tidak dapat dibaca oleh pembaca menggunakan versi sebelumnya.
  - MUNDUR_SEMUA: Pilihan kompatibilitas ini memungkinkan penerima data untuk membaca versi skema saat ini dan versi skema sebelumnya. Anda dapat menggunakan pilihan ini ketika Anda harus menghapus bidang atau menambahkan bidang opsional, dan memeriksa kompatibilitas terhadap semua versi skema sebelumnya.
  - MAJU: Pilihan kompatibilitas ini memungkinkan penerima data untuk membaca kedua versi skema saat ini dan versi skema berikutnya, tetapi tidak selalu versi yang lebih baru. Anda dapat

menggunakan pilihan ini ketika Anda harus menambahkan bidang atau menghapus bidang opsional, tetapi hanya memeriksa kompatibilitas terhadap semua versi skema sebelumnya.

- MAJU_SEMUA: Pilihan kompatibilitas ini memungkinkan penerima data untuk membaca skema yang ditulis oleh produsen dari setiap skema terdaftar baru. Anda dapat menggunakan pilihan ini ketika Anda harus menambahkan bidang atau menghapus bidang opsional, dan memeriksa kompatibilitas terhadap semua versi skema sebelumnya.
- PENUH: Pilihan kompatibilitas ini memungkinkan penerima data untuk membaca data yang ditulis oleh produsen menggunakan versi skema sebelumnya atau berikutnya, tetapi tidak harus selalau versi sebelumnya atau versi setelahnya. Anda dapat menggunakan pilihan ini ketika Anda harus menambahkan atau menghapus bidang opsional, tetapi hanya memeriksa kompatibilitas terhadap versi skema sebelumnya.
- PENUH_SEMUA: Pilihan kompatibilitas ini memungkinkan penerima data untuk membaca data yang ditulis oleh produsen menggunakan semua versi skema sebelumnya. Anda dapat menggunakan pilihan ini ketika Anda harus menambahkan atau menghapus bidang opsional, dan memeriksa kompatibilitas terhadap semua versi skema sebelumnya.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah deskripsi opsional dari skema. Jika deskripsi tidak tersedia, maka tidak akan ada nilai default otomatis untuk ini.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  AWS tag yang berisi pasangan nilai kunci dan dapat dicari berdasarkan konsol, baris perintah, atau API. Jika ditentukan, ikuti AWS tags-on-create polanya.

- `SchemaDefinition` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 170000 byte, yang cocok dengan Custom string pattern #32.

  Definisi skema menggunakan pengaturan `DataFormat` untuk `SchemaName`.

Respons

- `RegistryName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama registri.

- `RegistryArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) registri.

- `SchemaName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama skema.

- `SchemaArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari skema.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah deskripsi skema jika ditentukan saat dibuat.

- `DataFormat` – String UTF-8 (nilai yang valid: `AVRO` | `JSON` | `PROTOBUF`).

  Format data dari definisi skema. Saat ini`AVRO`, `JSON` dan `PROTOBUF` didukung.

- `Compatibility` – String UTF-8 (nilai yang valid: `NONE` | `DISABLED` | `BACKWARD` | `BACKWARD_ALL` | `FORWARD` | `FORWARD_ALL` | `FULL` | `FULL_ALL`).

  Mode kompatibilitas skema.

- `SchemaCheckpoint` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Nomor versi pos pemeriksaan (terakhir kali mode kompatibilitas diubah).

- `LatestSchemaVersion` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Versi terbaru dari skema yang dikaitkan dengan definisi skema yang dikembalikan.

- `NextSchemaVersion` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Versi berikutnya dari skema yang dikaitkan dengan definisi skema yang dikembalikan.

- `SchemaStatus` – String UTF-8 (nilai yang valid: `AVAILABLE` | `PENDING` | `DELETING`).

  Status skema.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Tag untuk skema.
- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  Pengenal unik dari versi skema yang pertama.
- `SchemaVersionStatus` – String UTF-8 (nilai yang valid: `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Status versi skema pertama yang dibuat.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `AlreadyExistsException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`
- `InternalServiceException`

GetSchema tindakan (Python: get_schema)

Menjelaskan skema yang ditentukan secara detail.

Permintaan

- `SchemaId` — Wajib: Sebuah objek SchemaId.

Ini adalah sebuah struktur pembungkus berisi bidang identitas skema. Strukturnya berisi:

- SchemaId$SchemaArn: Nama Sumber Daya Amazon (ARN) dari skema. Baik `SchemaArn`, atau `SchemaName` dan `RegistryName` harus disediakan.
- SchemaId$SchemaName: Nama skema. Baik `SchemaArn`, atau `SchemaName` dan `RegistryName` harus disediakan.

Respons

- `RegistryName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama registri.
- `RegistryArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) registri.
- `SchemaName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama skema.
- `SchemaArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari skema.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah deskripsi skema jika ditentukan saat dibuat
- `DataFormat` – String UTF-8 (nilai yang valid: `AVRO` | `JSON` | `PROTOBUF`).

  Format data dari definisi skema. Saat ini`AVRO`, `JSON` dan `PROTOBUF` didukung.
- `Compatibility` – String UTF-8 (nilai yang valid: `NONE` | `DISABLED` | `BACKWARD` | `BACKWARD_ALL` | `FORWARD` | `FORWARD_ALL` | `FULL` | `FULL_ALL`).

  Mode kompatibilitas skema.
- `SchemaCheckpoint` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

Nomor versi pos pemeriksaan (terakhir kali mode kompatibilitas diubah).

- `LatestSchemaVersion` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Versi terbaru dari skema yang dikaitkan dengan definisi skema yang dikembalikan.
- `NextSchemaVersion` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Versi berikutnya dari skema yang dikaitkan dengan definisi skema yang dikembalikan.
- `SchemaStatus` – String UTF-8 (nilai yang valid: `AVAILABLE` | `PENDING` | `DELETING`).

  Status skema.
- `CreatedTime` – String UTF-8.

  Tanggal dan waktu saat sebuah skema dibuat.
- `UpdatedTime` – String UTF-8.

  Tanggal dan waktu saat sebuah skema diperbarui.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`

ListSchemaVersions tindakan (Python: list_schema_versions)

Mengembalikan daftar versi skema yang telah Anda buat, dengan informasi yang minimal. Versi skema dengan status Dihapus tidak akan disertakan dalam hasil. Hasil kosong akan dikembalikan jika tidak ada versi skema yang tersedia.

Permintaan

- `SchemaId` — Wajib: Sebuah objek SchemaId.

  Ini adalah sebuah struktur pembungkus berisi bidang identitas skema. Strukturnya berisi:
  - SchemaId$SchemaArn: Nama Sumber Daya Amazon (ARN) dari skema. Baik `SchemaArn`, atau `SchemaName` dan `RegistryName` harus disediakan.

  - SchemaId$SchemaName: Nama skema. Baik `SchemaArn`, atau `SchemaName` dan `RegistryName` harus disediakan.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100.

  Jumlah maksimum hasil yang diperlukan untuk setiap halaman. Jika nilai tidak diberikan, maka diatur ke nilai default 25 per halaman.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Respons

- `Schemas` – Susunan objek SchemaVersionListItem.

  Susunan objek `SchemaVersionList` yang berisi detail dari setiap versi skema.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan untuk pemberian nomor halaman untuk daftar token yang ditampilkan, dikembalikan jika segmen saat ini dari daftar tersebut bukan yang terakhir.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`

GetSchemaVersion tindakan (Python: get_schema_version)

Mendapatkan skema yang ditentukan oleh ID uniknya yang ditetapkan ketika versi skema dibuat atau didaftarkan. Versi skema dengan status Dihapus tidak akan disertakan dalam hasil.

Permintaan

- `SchemaId` — Sebuah objek SchemaId.

  Ini adalah sebuah struktur pembungkus berisi bidang identitas skema. Strukturnya berisi:

  - SchemaId$SchemaArn: Nama Sumber Daya Amazon (ARN) dari skema. Baik `SchemaArn`, atau `SchemaName` dan `RegistryName` harus disediakan.
  - SchemaId$SchemaName: Nama skema. Baik `SchemaArn`, atau `SchemaName` dan `RegistryName` harus disediakan.
- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  `SchemaVersionId` dari versi skema. Kolom ini diperlukan untuk mengambil berdasarkan ID skema. Baik ini atau pembungkus `SchemaId` harus disediakan.
- `SchemaVersionNumber` — Sebuah objek SchemaVersionNumber.

  Nomor versi dari skema.

Respons

- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  `SchemaVersionId` dari versi skema.
- `SchemaDefinition` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 170000 byte, yang cocok dengan Custom string pattern #32.

  Definisi skema untuk ID skema.
- `DataFormat` – String UTF-8 (nilai yang valid: `AVRO` | `JSON` | `PROTOBUF`).

  Format data dari definisi skema. Saat ini`AVRO`, `JSON` dan `PROTOBUF` didukung.
- `SchemaArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari skema.
- `VersionNumber` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Nomor versi dari skema.
- `Status` – String UTF-8 (nilai yang valid: `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Status versi skema.
- `CreatedTime` – String UTF-8.

Tanggal dan waktu saat versi skema dibuat.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`

GetSchemaVersionsDiff tindakan (Python: get_schema_versions_diff)

Mengambil perbedaan versi skema dalam jenis perbedaan yang ditentukan antara dua versi skema yang disimpan di Registri Skema.

API ini memungkinkan Anda untuk membandingkan dua versi skema antara dua definisi skema berdasarkan pada skema yang sama.

Permintaan

- `SchemaId` — Wajib: Sebuah objek SchemaId.

  Ini adalah sebuah struktur pembungkus berisi bidang identitas skema. Strukturnya berisi:

  - SchemaId$SchemaArn: Nama Sumber Daya Amazon (ARN) dari skema. Salah satu dari `SchemaArn` atau `SchemaName` harus disediakan.
  - SchemaId$SchemaName: Nama skema. Salah satu dari `SchemaArn` atau `SchemaName` harus disediakan.
- `FirstSchemaVersionNumber` — Wajib: Sebuah objek SchemaVersionNumber.

  Versi skema yang pertama dari dua versi skema yang akan dibandingkan.
- `SecondSchemaVersionNumber` — Wajib: Sebuah objek SchemaVersionNumber.

  Versi skema yang kedua dari dua versi skema yang akan dibandingkan.
- `SchemaDiffType` – Wajib: String UTF-8 (nilai yang valid: `SYNTAX_DIFF`).

  Mengacu pada `SYNTAX_DIFF`, yang merupakan tipe perbedaan yang didukung saat ini.

Respons

- `Diff` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 340000 byte, yang cocok dengan Custom string pattern #32.

  Perbedaan antara skema sebagai string dalam JsonPatch format.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `AccessDeniedException`
- `InternalServiceException`

ListRegistries tindakan (Python: list_registries)

Mengembalikan sebuah daftar registri yang telah Anda buat, dengan informasi registri yang minimal. Registri yang sedang dalam status `Deleting` tidak akan dimasukkan dalam hasil. Hasil kosong akan dikembalikan jika tidak ada registri yang tersedia.

Permintaan

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100.

  Jumlah maksimum hasil yang diperlukan untuk setiap halaman. Jika nilai tidak diberikan, maka diatur ke nilai default 25 per halaman.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Respons

- `Registries` – Susunan objek RegistryListItem.

  Susunan objek `RegistryDetailedListItem` yang berisi detail minimal dari setiap registri.
- `NextToken` – String UTF-8.

Sebuah token kelanjutan untuk pemberian nomor halaman untuk daftar token yang ditampilkan, dikembalikan jika segmen saat ini dari daftar tersebut bukan yang terakhir.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `InternalServiceException`

ListSchemas tindakan (Python: list_schemas)

Mengembalikan sebuah daftar skema dengan detail yang minimal. Skema dengan status Menghapus tidak akan disertakan dalam hasil. Hasil kosong akan dikembalikan jika tidak ada skema yang tersedia.

Saat `RegistryId` tidak disediakan, semua skema di seluruh registri akan menjadi bagian dari respons API.

Permintaan

- `RegistryId` — Sebuah objek RegistryId.

  Pembungkus struktur yang mungkin berisi nama registri dan Amazon Resource Name (ARN).
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100.

  Jumlah maksimum hasil yang diperlukan untuk setiap halaman. Jika nilai tidak diberikan, maka diatur ke nilai default 25 per halaman.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Respons

- `Schemas` – Susunan objek SchemaListItem.

  Susunan objek `SchemaListItem` yang berisi detail dari setiap skema.
- `NextToken` – String UTF-8.

Sebuah token kelanjutan untuk pemberian nomor halaman untuk daftar token yang ditampilkan, dikembalikan jika segmen saat ini dari daftar tersebut bukan yang terakhir.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`

RegisterSchemaVersion tindakan (Python: register_schema_version)

Menambahkan sebuah versi baru untuk skema yang ada. Mengembalikan sebuah kesalahan jika skema versi baru tidak memenuhi persyaratan kompatibilitas set skema. API ini tidak akan membuat set skema baru dan akan mengembalikan kesalahan 404 jika set skema belum ada di Registri Skema.

Jika ini adalah definisi skema pertama yang akan terdaftar di Registri Skema, maka API ini akan menyimpan versi skema dan mengembalikan segera. Jika tidak, panggilan ini memiliki potensi untuk berjalan lebih lama daripada operasi lain karena mode kompatibilitas. Anda dapat memanggil API `GetSchemaVersion` dengan `SchemaVersionId` untuk memeriksa mode kompatibilitas.

Jika definisi skema yang sama sudah disimpan dalam Registri Skema sebagai sebuah versi, maka ID skema dari skema yang ada akan dikembalikan ke pemanggil.

Permintaan

- `SchemaId` — Wajib: Sebuah objek SchemaId.

  Ini adalah sebuah struktur pembungkus berisi bidang identitas skema. Strukturnya berisi:

  - SchemaId$SchemaArn: Nama Sumber Daya Amazon (ARN) dari skema. Baik `SchemaArn`, atau `SchemaName` dan `RegistryName` harus disediakan.
  - SchemaId$SchemaName: Nama skema. Baik `SchemaArn`, atau `SchemaName` dan `RegistryName` harus disediakan.
- `SchemaDefinition` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 170000 byte, yang cocok dengan Custom string pattern #32.

Definisi skema menggunakan pengaturan `DataFormat` untuk `SchemaName`.

Respons

- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  ID unik yang mewakili versi skema ini.
- `VersionNumber` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Versi skema ini (untuk aliran sinkronisasi saja, jika ini adalah versi pertama).
- `Status` – String UTF-8 (nilai yang valid: `AVAILABLE` | `PENDING` | `FAILURE` | `DELETING`).

  Status versi skema.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`
- `InternalServiceException`

UpdateSchema tindakan (Python: update_schema)

Memperbarui deskripsi, pengaturan kompatibilitas, atau versi pos pemeriksaan untuk sebuah set skema.

Untuk memperbarui pengaturan kompatibilitas, panggilan tidak akan memvalidasi kompatibilitas untuk seluruh rangkaian versi skema dengan pengaturan kompatibilitas baru. Jika nilai untuk `Compatibility` disediakan, maka `VersionNumber` (pos pemeriksaan) juga diperlukan. API akan memvalidasi konsistensi nomor versi pos pemeriksaan.

Jika nilai untuk `VersionNumber` (pos pemeriksaan) disediakan, maka `Compatibility` bersifat opsional dan ini dapat digunakan untuk mengatur/me-reset pos pemeriksaan untuk skema tersebut.

Pembaruan ini akan terjadi hanya jika skema dalam status TERSEDIA.

Permintaan

- `SchemaId` — Wajib: Sebuah objek SchemaId.

  Ini adalah sebuah struktur pembungkus berisi bidang identitas skema. Strukturnya berisi:

  - SchemaId$SchemaArn: Nama Sumber Daya Amazon (ARN) dari skema. Salah satu dari `SchemaArn` atau `SchemaName` harus disediakan.
  - SchemaId$SchemaName: Nama skema. Salah satu dari `SchemaArn` atau `SchemaName` harus disediakan.
- `SchemaVersionNumber` — Sebuah objek SchemaVersionNumber.

  Nomor versi yang diperlukan untuk melakukan pemeriksaan. Salah satu dari `VersionNumber` atau `Compatibility` harus disediakan.
- `Compatibility` – String UTF-8 (nilai yang valid: `NONE` | `DISABLED` | `BACKWARD` | `BACKWARD_ALL` | `FORWARD` | `FORWARD_ALL` | `FULL` | `FULL_ALL`).

  Pengaturan kompatibilitas baru untuk skema.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi baru untuk skema.

Respons

- `SchemaArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari skema.
- `SchemaName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama skema.
- `RegistryName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama registri yang berisi skema.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `ConcurrentModificationException`
- `InternalServiceException`

CheckSchemaVersionValidity tindakan (Python: check_schema_version_validity)

Memvalidasi skema yang disediakan. Panggilan ini tidak memiliki efek samping, ia hanya memvalidasi dengan menggunakan skema yang disediakan dengan menggunakan `DataFormat` sebagai formatnya. Karena tidak mengambil nama set skema, maka tidak ada pemeriksaan kompatibilitas yang dilakukan.

Permintaan

- `DataFormat` – Wajib: String UTF-8 (nilai yang valid: `AVRO` | `JSON` | `PROTOBUF`).

  Format data dari definisi skema. Saat ini`AVRO`, `JSON` dan `PROTOBUF` didukung.
- `SchemaDefinition` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 170000 byte, yang cocok dengan Custom string pattern #32.

  Definisi skema yang harus divalidasi.

Respons

- `Valid` – Boolean.

  Mengembalikan BETUL, jika skema ini valid, dan mengembalikan SALAH, jika skema tidak valid.
- `Error` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 5000 byte.

  Pesan kesalahan kegagalan validasi.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `InternalServiceException`

UpdateRegistry tindakan (Python: update_registry)

Memperbarui registri yang ada yang digunakan untuk menyimpan koleksi skema. Properti yang diperbarui berhubungan dengan registri, dan tidak mengubah salah satu skema dalam registri tersebut.

Permintaan

- `RegistryId` — Wajib: Sebuah objek RegistryId.

  Ini adalah sebuah struktur pembungkus yang mungkin berisi nama registri dan Amazon Resource Name (ARN).
- `Description` — Wajib: String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah deskripsi registri. Jika deskripsi tidak tersedia, maka bidang ini tidak akan diperbarui.

Respons

- `RegistryName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama registri yang diperbarui.
- `RegistryArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari registri yang diperbarui.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`

- `EntityNotFoundException`
- `ConcurrentModificationException`
- `InternalServiceException`

GetSchemaByDefinition tindakan (Python: get_schema_by_definition)

Mengambil sebuah skema berdasarkan `SchemaDefinition`. Definisi skema dikirim ke Registri Skema, di-kanonikalisasi, dan di-hash. Jika hash cocok dalam lingkup `SchemaName` atau ARN (atau registri default, jika tidak disediakan), maka metadata dari skema tersebut dikembalikan. Jika tidak, 404 atau NotFound kesalahan dikembalikan. Versi skema dengan status `Deleted` tidak akan disertakan dalam hasil.

Permintaan

- `SchemaId` — Wajib: Sebuah objek SchemaId.

  Ini adalah sebuah struktur pembungkus berisi bidang identitas skema. Strukturnya berisi:

  - SchemaId$SchemaArn: Nama Sumber Daya Amazon (ARN) dari skema. Salah satu dari `SchemaArn` atau `SchemaName` harus disediakan.
  - SchemaId$SchemaName: Nama skema. Salah satu dari `SchemaArn` atau `SchemaName` harus disediakan.
- `SchemaDefinition` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 170000 byte, yang cocok dengan Custom string pattern #32.

  Definisi skema yang diwajibkan detail skema untuknya.

Respons

- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  ID skema dari versi skema.
- `SchemaArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari skema.
- `DataFormat` – String UTF-8 (nilai yang valid: `AVRO` | `JSON` | `PROTOBUF`).

Format data dari definisi skema. Saat iniAVRO, JSON dan PROTOBUF didukung.

- Status – String UTF-8 (nilai yang valid: AVAILABLE | PENDING | FAILURE | DELETING).

  Status versi skema.

- CreatedTime – String UTF-8.

  Tanggal dan waktu saat sebuah skema dibuat.

Kesalahan

- InvalidInputException
- AccessDeniedException
- EntityNotFoundException
- InternalServiceException

GetRegistry tindakan (Python: get_registry)

Menjelaskan registri yang ditentukan secara detail.

Permintaan

- RegistryId — Wajib: Sebuah objek RegistryId.

  Ini adalah sebuah struktur pembungkus yang mungkin berisi nama registri dan Amazon Resource Name (ARN).

Respons

- RegistryName — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama registri.

- RegistryArn — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) registri.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah deskripsi registri.

- `Status` – String UTF-8 (nilai yang valid: `AVAILABLE` | `DELETING`).

  Status registri.

- `CreatedTime` – String UTF-8.

  Tanggal dan waktu saat registri dibuat.

- `UpdatedTime` – String UTF-8.

  Tanggal dan waktu saat registri diperbarui.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`
- `InternalServiceException`

PutSchemaVersionMetadata tindakan (Python: put_schema_version_metadata)

Menempatkan pasangan nilai kunci metadata untuk ID versi skema yang ditentukan. Maksimal 10 pasangan nilai kunci akan diizinkan untuk setiap versi skema. Mereka dapat ditambahkan lebih dari satu atau beberapa panggilan.

Permintaan

- `SchemaId` — Sebuah objek SchemaId.

  ID unik untuk skema.

- `SchemaVersionNumber` — Sebuah objek SchemaVersionNumber.

  Nomor versi dari skema.

- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  ID versi unik dari versi skema.
- `MetadataKeyValue` — Wajib: Sebuah objek MetadataKeyValuePair.

  Nilai yang sesuai kunci metadata.

Respons

- `SchemaArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) untuk skema.
- `SchemaName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama untuk skema.
- `RegistryName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama untuk registri.
- `LatestVersion` – Boolean.

  Skema versi terbaru.
- `VersionNumber` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Nomor versi dari skema.
- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  ID versi unik dari versi skema.
- `MetadataKey` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #33.

  Kunci metadata.
- `MetadataValue` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 256 byte, yang cocok dengan Custom string pattern #33.

Nilai dari kunci metadata.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `AlreadyExistsException`
- `EntityNotFoundException`
- `ResourceNumberLimitExceededException`

QuerySchemaVersionMetadata tindakan (Python: query_schema_version_metadata)

Kueri untuk informasi metadata versi skema.

Permintaan

- `SchemaId` — Sebuah objek SchemaId.

  Sebuah struktur pembungkus yang mungkin berisi nama skema dan Amazon Resource Name (ARN).
- `SchemaVersionNumber` — Sebuah objek SchemaVersionNumber.

  Nomor versi dari skema.
- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  ID versi unik dari versi skema.
- `MetadataList` – Susunan objek MetadataKeyValuePair.

  Mencari pasangan nilai kunci untuk metadata, jika mereka tidak disediakan, maka semua informasi metadata akan diambil.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 50.

  Jumlah maksimum hasil yang diperlukan untuk setiap halaman. Jika nilai tidak diberikan, maka diatur ke nilai default 25 per halaman.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah panggilan kelanjutan.

Respons

- `MetadataInfoMap` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #33.

  Setiap nilai adalah sebuah objek MetadataInfo A.

  Sebuah peta kunci metadata dan nilai-nilai yang dikaitkan.
- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  ID versi unik dari versi skema.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan untuk pemberian nomor halaman untuk daftar token yang ditampilkan, dikembalikan jika segmen saat ini dari daftar tersebut bukan yang terakhir.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`

RemoveSchemaVersionMetadata tindakan (Python: remove_schema_version_metadata)

Menghapus sebuah pasangan nilai kunci dari metadata versi skema untuk ID versi skema yang ditentukan.

Permintaan

- `SchemaId` — Sebuah objek SchemaId.

Sebuah struktur pembungkus yang mungkin berisi nama skema dan Amazon Resource Name (ARN).

- `SchemaVersionNumber` — Sebuah objek SchemaVersionNumber.

  Nomor versi dari skema.

- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  ID versi unik dari versi skema.

- `MetadataKeyValue` — Wajib: Sebuah objek MetadataKeyValuePair.

  Nilai dari kunci metadata.

Respons

- `SchemaArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari skema.

- `SchemaName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama skema.

- `RegistryName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama registri.

- `LatestVersion` – Boolean.

  Skema versi terbaru.

- `VersionNumber` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 100000.

  Nomor versi dari skema.

- `SchemaVersionId` — String UTF-8, sepanjang tidak kurang dari 36 atau lebih dari 36 byte, yang cocok dengan Custom string pattern #17.

  ID versi untuk versi skema.

- `MetadataKey` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #33.

  Kunci metadata.

- `MetadataValue` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 256 byte, yang cocok dengan Custom string pattern #33.

  Nilai dari kunci metadata.

Kesalahan

- `InvalidInputException`
- `AccessDeniedException`
- `EntityNotFoundException`

DeleteRegistry tindakan (Python: delete_registry)

Menghapus registri secara keseluruhan termasuk skema dan semua versinya. Untuk mendapatkan status operasi hapus, Anda dapat memanggil API `GetRegistry` setelah panggilan asinkron. Menghapus registri akan menonaktifkan semua operasi online untuk registri seperti API `UpdateRegistry`, `CreateSchema`, `UpdateSchema`, dan `RegisterSchemaVersion`.

Permintaan

- `RegistryId` — Wajib: Sebuah objek RegistryId.

  Ini adalah sebuah struktur pembungkus yang mungkin berisi nama registri dan Amazon Resource Name (ARN).

Respons

- `RegistryName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama registri yang sedang dihapus.

- `RegistryArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

Amazon Resource Name (ARN) dari registri yang sedang dihapus.

- `Status` – String UTF-8 (nilai yang valid: `AVAILABLE` | `DELETING`).

  Status registri. Sebuah operasi yang sukses akan mengembalikan status `Deleting`.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `AccessDeniedException`
- `ConcurrentModificationException`

DeleteSchema tindakan (Python: delete_schema)

Menghapus seluruh set skema, termasuk set skema dan semua versinya. Untuk mendapatkan status operasi hapus, Anda dapat memanggil API `GetSchema` setelah panggilan asinkron. Menghapus sebuah registri akan menonaktifkan semua operasi online untuk skema, seperti API `GetSchemaByDefinition`, dan `RegisterSchemaVersion`.

Permintaan

- `SchemaId` — Wajib: Sebuah objek SchemaId.

  Ini adalah sebuah struktur pembungkus yang mungkin berisi nama skema dan Amazon Resource Name (ARN).

Respons

- `SchemaArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari skema yang sedang dihapus.

- `SchemaName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #18.

  Nama dari skema yang sedang dihapus.

- `Status` – String UTF-8 (nilai yang valid: `AVAILABLE` | `PENDING` | `DELETING`).

  Status skema.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `AccessDeniedException`
- `ConcurrentModificationException`

DeleteSchemaVersions tindakan (Python: delete_schema_versions)

Menghapus versi skema yang ditentukan. Nomor versi atau rentang versi mungkin disediakan. Jika modus kompatibilitas melarang menghapus sebuah versi yang diperlukan, seperti BACKWARDS_FULL, maka kesalahan akan dikembalikan. Memanggil API `GetSchemaVersions` setelah panggilan ini akan mencantumkan status versi yang dihapus.

Ketika rentang nomor versi berisi versi yang diperiksa, maka API akan mengembalikan konflik 409 dan tidak akan melanjutkan dengan penghapusan. Anda harus menghapus pos pemeriksaan terlebih dahulu menggunakan API `DeleteSchemaCheckpoint` sebelum menggunakan API ini.

Anda tidak dapat menggunakan API `DeleteSchemaVersions` untuk menghapus versi skema pertama dalam set skema tersebut. Versi skema pertama hanya dapat dihapus oleh API `DeleteSchema`. Operasi ini juga akan menghapus `SchemaVersionMetadata` yang dilampirkan pada versi skema. Hapus paksa akan diberlakukan pada basis data.

Jika modus kompatibilitas melarang menghapus sebuah versi yang diperlukan, seperti BACKWARDS_FULL, maka kesalahan akan dikembalikan.

Permintaan

- `SchemaId` — Wajib: Sebuah objek SchemaId.

  Ini adalah sebuah struktur pembungkus yang mungkin berisi nama skema dan Amazon Resource Name (ARN).
- `Versions` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 100000 byte, yang cocok dengan Custom string pattern #34.

Rentang versi dapat diberikan, mungkin dalam format:

- nomor versi tunggal, 5
- rentang, 5-8: menghapus versi 5, 6, 7, 8

Respons

- `SchemaVersionErrors` – Susunan objek SchemaVersionErrorItem.

  Sebuah daftar objek `SchemaVersionErrorItem`, masing-masing berisi kesalahan dan versi skema.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `AccessDeniedException`
- `ConcurrentModificationException`

Alur Kerja

API Alur Kerja menjelaskan tipe data dan API yang terkait dengan pembuatan, pembaruan, atau tampilan alur kerja. AWS Glue Riwayat Job run dapat diakses selama 90 hari untuk alur kerja dan pekerjaan Anda.

Jenis data

- JobNodeDetails struktur
- CrawlerNodeDetails struktur
- TriggerNodeDetails struktur
- Struktur merangkak
- Struktur simpul
- Struktur tepi
- Struktur alur kerja
- WorkflowGraph struktur

- WorkflowRun struktur
- WorkflowRunStatistics struktur
- StartingEventBatchCondition struktur
- Struktur cetak biru
- BlueprintDetails struktur
- LastActiveDefinition struktur
- BlueprintRun struktur

JobNodeDetails struktur

Detail dari simpul Tugas yang ada dalam alur kerja.

Bidang

- `JobRuns` – Susunan objek JobRun.

  Informasi untuk eksekusi tugas yang diwakili oleh simpul tugas.

CrawlerNodeDetails struktur

Detail dari simpul Crawler yang ada dalam alur kerja.

Bidang

- `Crawls` – Susunan objek Crawl.

  Daftar perayapan yang diwakili oleh simpul perayapan.

TriggerNodeDetails struktur

Detail dari simpul Pemicu yang ada dalam alur kerja.

Bidang

- `Trigger` — Sebuah objek Pemicu.

  Informasi pemicu yang diwakili oleh simpul pemicu.

Struktur merangkak

Detail dari sebuah perayapan dalam alur kerja.

Bidang

- `State` – String UTF-8 (nilai valid: `RUNNING` | `CANCELLING` | `CANCELLED` | `SUCCEEDED` | `FAILED` | `ERROR`).

  Status crawler.
- `StartedOn` — Stempel waktu.

  Tanggal dan waktu saat perayapan dimulai.
- `CompletedOn` — Stempel waktu.

  Tanggal dan waktu saat perayapan selesai.
- `ErrorMessage` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Pesan kesalahan yang dikaitkan dengan perayapan.
- `LogGroup` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 512 byte, yang cocok dengan Log group string pattern.

  Grup log yang dikaitkan dengan perayapan.
- `LogStream` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 512 byte, yang cocok dengan Log-stream string pattern.

  Pengaliran log yang dikaitkan dengan perayapan.

Struktur simpul

Node mewakili AWS Glue komponen (pemicu, perayap, atau pekerjaan) pada grafik alur kerja.

Bidang

- `Type` – String UTF-8 (nilai yang valid: `CRAWLER` | `JOB` | `TRIGGER`).

  Jenis AWS Glue komponen yang diwakili oleh node.

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama AWS Glue komponen diwakili oleh node.

- `UniqueId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Id unik yang ditugaskan ke simpul dalam alur kerja.

- `TriggerDetails` — Sebuah objek TriggerNodeDetail.

  Detail dari Pemicu ketika simpul mewakili sebuah Pemicu.

- `JobDetails` — Sebuah objek JobNodeDetail.

  Detail dari Tugas ketika simpul mewakili sebuah Tugas.

- `CrawlerDetails` — Sebuah objek CrawlerNodeDetail.

  Detail dari crawler ketika simpul mewakili sebuah crawler.

Struktur tepi

Tepi mewakili koneksi terarah antara dua AWS Glue komponen yang merupakan bagian dari alur kerja yang dimiliki tepi.

Bidang

- `SourceId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Simpul unik dalam alur kerja di mana egde dimulai.

- `DestinationId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Simpul unik dalam alur kerja di mana egde berakhir.

Struktur alur kerja

Alur kerja adalah kumpulan beberapa AWS Glue pekerjaan dependen dan crawler yang dijalankan untuk menyelesaikan tugas ETL yang kompleks. Alur kerja mengelola eksekusi dan pemantauan semua tugas dan crawler.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja.
- `Description` – String UTF-8.

  Deskripsi alur kerja.
- `DefaultRunProperties` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah string UTF-8.

  Sebuah kumpulan properti yang akan digunakan sebagai bagian dari setiap eksekusi alur kerja. Properti eksekusi disediakan untuk setiap tugas di alur kerja. Sebuah tugas dapat memodifikasi properti untuk tugas berikutnya dalam aliran.
- `CreatedOn` — Stempel waktu.

  Tanggal dan waktu ketika alur kerja dibuat.
- `LastModifiedOn` — Stempel waktu.

  Tanggal dan waktu ketika alur kerja terakhir diubah.
- `LastRun` — Sebuah objek WorkflowRun.

  Informasi tentang eksekusi terakhir dari alur kerja.
- `Graph` — Sebuah objek WorkflowGraph.

  Grafik yang mewakili semua AWS Glue komponen yang termasuk dalam alur kerja sebagai node dan koneksi terarah di antara mereka sebagai tepi.
- `CreationStatus` – String UTF-8 (nilai yang valid: `CREATING` | `CREATED` | `CREATION_FAILED`).

Status pembuatan alur kerja.

- `MaxConcurrentRuns` — Nomor (bilangan bulat).

  Anda dapat menggunakan parameter ini untuk mencegah beberapa pembaruan yang tidak diinginkan terhadap data, untuk mengontrol biaya, atau dalam beberapa kasus, untuk mencegah agar tidak melebihi jumlah maksimum eksekusi yang berjalan bersamaan dari salah satu tugas komponen. Jika Anda membiarkan parameter ini kosong, tidak ada batas jumlah eksekusi alur kerja bersamaan.

- `BlueprintDetails` — Sebuah objek BlueprintDetails.

  Struktur ini menunjukkan detail cetak biru yang darinya alur kerja khusus ini dibuat.

WorkflowGraph struktur

Sebuah grafik alur kerja mewakili alur kerja lengkap yang berisi semua komponen AWS Glue yang ada dalam alur kerja dan semua koneksi yang diarahkan antara mereka.

Bidang

- `Nodes` – Susunan objek Simpul.

  Daftar AWS Glue komponen milik alur kerja yang direpresentasikan sebagai node.

- `Edges` – Susunan objek Edge.

  Daftar semua koneksi yang diarahkan antara simpul milik alur kerja.

WorkflowRun struktur

Sebuah eksekusi alur kerja adalah eksekusi dari sebuah alur kerja yang menyediakan semua informasi waktu aktif.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang dijalankan.

- `WorkflowRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari eksekusi alur kerja ini.
- `PreviousRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari eksekusi alur kerja sebelumnya.
- `WorkflowRunProperties` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah string UTF-8.

  Properti ekseksui alur kerja yang ditetapkan selama eksekusi.
- `StartedOn` — Stempel waktu.

  Tanggal dan waktu ketika eksekusi alur kerja dimulai.
- `CompletedOn` — Stempel waktu.

  Tanggal dan waktu ketika eksekusi alur kerja selesai.
- `Status` – String UTF-8 (nilai yang valid: `RUNNING` | `COMPLETED` | `STOPPING` | `STOPPED` | `ERROR`).

  Status eksekusi alur kerja.
- `ErrorMessage` – String UTF-8.

  Pesan kesalahan ini menjelaskan kesalahan yang mungkin terjadi saat memulai eksekusi alur kerja. Saat ini satu-satunya pesan kesalahannya adalah "Eksekusi bersamaan melebihi untuk alur kerja: `foo`."
- `Statistics` — Sebuah objek WorkflowRunStatistik.

  Statistik eksekusi.
- `Graph` — Sebuah objek WorkflowGraph.

  Grafik yang mewakili semua AWS Glue komponen yang termasuk dalam alur kerja sebagai node dan koneksi terarah di antara mereka sebagai tepi.
- `StartingEventBatchCondition` — Sebuah objek StartingEventBatchCondition.

Syarat batch yang memulai eksekusi alur kerja.

WorkflowRunStatistics struktur

Statistik eksekusi alur kerja menyediakan statistik tentang alur kerja yang dijalankan.

Bidang

- `TotalActions` — Nomor (bilangan bulat).

  Jumlah total Tindakan dalam eksekusi alur kerja.
- `TimeoutActions` — Nomor (bilangan bulat).

  Jumlah total Tindakan yang kehabisan waktu.
- `FailedActions` — Nomor (bilangan bulat).

  Jumlah total Tindakan yang telah gagal.
- `StoppedActions` — Nomor (bilangan bulat).

  Jumlah total Tindakan yang telah dihentikan.
- `SucceededActions` — Nomor (bilangan bulat).

  Jumlah total Tindakan yang telah berhasil.
- `RunningActions` — Nomor (bilangan bulat).

  Jumlah total Tindakan yang dalam status berjalan.
- `ErroredActions` — Nomor (bilangan bulat).

  Menunjukkan jumlah pekerjaan yang berjalan dalam status ERROR dalam menjalankan alur kerja.
- `WaitingActions` — Nomor (bilangan bulat).

  Menunjukkan jumlah pekerjaan berjalan dalam status WAITING dalam alur kerja berjalan.

StartingEventBatchCondition struktur

Syarat batch yang memulai eksekusi alur kerja. Entah jumlah peristiwa dalam ukuran batch tiba, dalam hal ini BatchSize anggota tidak nol, atau jendela batch kedaluwarsa, dalam hal ini BatchWindow anggota tidak nol.

Bidang

- `BatchSize` — Nomor (bilangan bulat).

  Jumlah peristiwa dalam batch.
- `BatchWindow` — Nomor (bilangan bulat).

  Durasi jendela batch dalam hitungan detik.

Struktur cetak biru

Detail sebuah cetak biru.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Nama cetak birunya.
- `Description` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 512 byte.

  Deskripsi cetak biru.
- `CreatedOn` — Stempel waktu.

  Tanggal dan waktu ketika cetak biru terdaftar.
- `LastModifiedOn` — Stempel waktu.

  Tanggal dan waktu ketika cetak biru terakhir diubah.
- `ParameterSpec` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 131072 byte.

  Sebuah string JSON yang menunjukkan daftar spesifikasi parameter untuk cetak biru.
- `BlueprintLocation` – String UTF-8.

Menentukan path di Amazon S3 di mana cetak biru diterbitkan.

- `BlueprintServiceLocation` – String UTF-8.

  Menentukan path di Amazon S3 di mana cetak biru disalin ketika Anda memanggil `CreateBlueprint/UpdateBlueprint` untuk mendaftarkan cetak biru di AWS Glue.

- `Status` – String UTF-8 (nilai yang valid: `CREATING` | `ACTIVE` | `UPDATING` | `FAILED`).

  Status pendaftaran cetak biru.

  - Membuat — Pendaftaran cetak biru sedang berlangsung.
  - Aktif — Cetak biru telah berhasil terdaftar.
  - Memperbarui — Pembaruan untuk pendaftaran cetak biru sedang berlangsung.
  - Gagal — Pendaftaran cetak biru gagal.

- `ErrorMessage` – String UTF-8.

  Pesan kesalahan.

- `LastActiveDefinition` — Sebuah objek LastActiveDefinisi.

  Ketika ada beberapa versi cetak biru dan versi terbarunya memiliki beberapa kesalahan, atribut ini menunjukkan definisi cetak biru terakhir yang sukses yang tersedia dengan layanan.

BlueprintDetails struktur

Detail sebuah cetak biru.

Bidang

- `BlueprintName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Nama cetak birunya.

- `RunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID eksekusi untuk cetak biru ini.

LastActiveDefinition struktur

Ketika ada beberapa versi cetak biru dan versi terbarunya memiliki beberapa kesalahan, atribut ini menunjukkan definisi cetak biru terakhir yang sukses yang tersedia dengan layanan.

Bidang

- `Description` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 512 byte.

  Deskripsi cetak biru.

- `LastModifiedOn` — Stempel waktu.

  Tanggal dan waktu ketika cetak biru terakhir diubah.

- `ParameterSpec` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 131072 byte.

  Sebuah string JSON yang menentukan parameter untuk cetak biru.

- `BlueprintLocation` – String UTF-8.

  Menentukan jalur di Amazon S3 tempat cetak biru diterbitkan oleh pengembang. AWS Glue

- `BlueprintServiceLocation` – String UTF-8.

  Menentukan path di Amazon S3 di mana cetak biru disalin ketika Anda membuat atau memperbarui cetak biru.

BlueprintRun struktur

Detail eksekusi cetak biru.

Bidang

- `BlueprintName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Nama cetak birunya.

- `RunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID eksekusi untuk eksekusi cetak biru ini.

- `WorkflowName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang dibuat sebagai hasil dari eksekusi cetak biru yang sukses. Jika eksekusi cetak biru memiliki kesalahan, tidak akan ada alur kerja yang dibuat.

- `State` – String UTF-8 (nilai yang valid: `RUNNING` | `SUCCEEDED` | `FAILED` | `ROLLING_BACK`).

  Status eksekusi cetak biru. Nilai-nilai yang mungkin adalah:

  - Berjalan — Eksekusi cetak biru sedang berlangsung.
  - Berhasil — Eksekusi cetak biru berhasil diselesaikan.
  - Gagal — Eksekusi cetak biru gagal dan rollback selesai.
  - Melakukan rollback — Eksekusi cetak biru gagal dan rollback sedang berlangsung.

- `StartedOn` — Stempel waktu.

  Tanggal dan waktu ketika eksekusi cetak biru dimulai.

- `CompletedOn` — Stempel waktu.

  Tanggal dan waktu ketika eksekusi cetak biru selesai.

- `ErrorMessage` – String UTF-8.

  Menunjukkan kesalahan apa pun yang terlihat saat menjalankan cetak biru.

- `RollbackErrorMessage` – String UTF-8.

  Jika ada kesalahan saat membuat entitas sebuah alur kerja, kami mencoba untuk melakukan rollbak pada entitas yang dibuat sampai titik itu dan menghapusnya. Atribut ini menunjukkan kesalahan yang terlihat ketika mencoba untuk menghapus entitas yang sudah dibuat.

- `Parameters` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 131072 byte.

  Parameter cetak biru sebagai string. Anda harus memberikan nilai untuk setiap kunci yang diperlukan dari spesifikasi parameter yang didefinisikan dalam `Blueprint$ParameterSpec`.

- `RoleArn` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 1024 byte, yang cocok dengan Custom string pattern #26.

  ARN peran. Peran ini akan diasumsikan oleh AWS Glue layanan dan akan digunakan untuk membuat alur kerja dan entitas lain dari alur kerja.

Operasi

- CreateWorkflow tindakan (Python: create_workflow)
- UpdateWorkflow tindakan (Python: update_workflow)
- DeleteWorkflow tindakan (Python: delete_workflow)
- GetWorkflow tindakan (Python: get_workflow)
- ListWorkflows tindakan (Python: list_workflows)
- BatchGetWorkflows tindakan (Python: batch_get_workflows)
- GetWorkflowRun tindakan (Python: get_workflow_run)
- GetWorkflowRuns tindakan (Python: get_workflow_runs)
- GetWorkflowRunProperties tindakan (Python: get_workflow_run_properties)
- PutWorkflowRunProperties tindakan (Python: put_workflow_run_properties)
- CreateBlueprint tindakan (Python: create_blueprint)
- UpdateBlueprint tindakan (Python: update_blueprint)
- DeleteBlueprint tindakan (Python: delete_blueprint)
- ListBlueprints tindakan (Python: list_blueprints)
- BatchGetBlueprints tindakan (Python: batch_get_blueprints)
- StartBlueprintRun tindakan (Python: start_blueprint_run)
- GetBlueprintRun tindakan (Python: get_blueprint_run)
- GetBlueprintRuns tindakan (Python: get_blueprint_runs)
- StartWorkflowRun tindakan (Python: start_workflow_run)
- StopWorkflowRun tindakan (Python: stop_workflow_run)
- ResumeWorkflowRun tindakan (Python: resume_workflow_run)

CreateWorkflow tindakan (Python: create_workflow)

Menciptakan sebuah alur kerja baru.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Nama yang akan ditetapkan untuk alur kerja. Nama ini harus unik dalam akun Anda.

- `Description` – String UTF-8.

  Deskripsi alur kerja.

- `DefaultRunProperties` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah string UTF-8.

  Sebuah kumpulan properti yang akan digunakan sebagai bagian dari setiap eksekusi alur kerja.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Tag yang akan digunakan dengan alur kerja ini.

- `MaxConcurrentRuns` — Nomor (bilangan bulat).

  Anda dapat menggunakan parameter ini untuk mencegah beberapa pembaruan yang tidak diinginkan terhadap data, untuk mengontrol biaya, atau dalam beberapa kasus, untuk mencegah agar tidak melebihi jumlah maksimum eksekusi yang berjalan bersamaan dari salah satu tugas komponen. Jika Anda membiarkan parameter ini kosong, tidak ada batas jumlah eksekusi alur kerja bersamaan.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang disediakan sebagai bagian dari permintaan.

Kesalahan

- `AlreadyExistsException`

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`

UpdateWorkflow tindakan (Python: update_workflow)

Perbarui alur kerja yang ada.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang akan diperbarui.
- `Description` – String UTF-8.

  Deskripsi alur kerja.
- `DefaultRunProperties` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah string UTF-8.

  Sebuah kumpulan properti yang akan digunakan sebagai bagian dari setiap eksekusi alur kerja.
- `MaxConcurrentRuns` — Nomor (bilangan bulat).

  Anda dapat menggunakan parameter ini untuk mencegah beberapa pembaruan yang tidak diinginkan terhadap data, untuk mengontrol biaya, atau dalam beberapa kasus, untuk mencegah agar tidak melebihi jumlah maksimum eksekusi yang berjalan bersamaan dari salah satu tugas komponen. Jika Anda membiarkan parameter ini kosong, tidak ada batas jumlah eksekusi alur kerja bersamaan.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang ditentukan dalam input.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentModificationException`

DeleteWorkflow tindakan (Python: delete_workflow)

Menghapus sebuah alur kerja.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang akan dihapus.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang ditentukan dalam input.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`

- `OperationTimeoutException`
- `ConcurrentModificationException`

GetWorkflow tindakan (Python: get_workflow)

Mengambil metadata sumber daya untuk sebuah alur kerja.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang akan diambil.
- `IncludeGraph` – Boolean.

  Menentukan apakah akan menyertakan grafik ketika mengembalikan metadata sumber daya alur kerja.

Respons

- `Workflow` — Sebuah objek Alur kerja.

  Metadata sumber daya untuk alur kerja.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

ListWorkflows tindakan (Python: list_workflows)

Daftar nama alur kerja yang sudah dibuat di akun.

Permintaan

- `NextToken` – String UTF-8.

Sebuah token kelanjutan, jika ini adalah permintaan kelanjutan.

- `MaxResults`— Angka (bilangan bulat), tidak kurang dari 1 atau lebih dari 25.

  Ukuran maksimum daftar yang akan dikembalikan.

Respons

- `Workflows` – Susunan string UTF-8, tidak kurang dari 1 atau lebih dari 25 string.

  Daftar nama alur kerja yang dalam akun.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika tidak semua nama alur kerja yang sudah dikembalikan.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchGetWorkflows tindakan (Python: batch_get_workflows)

Mengembalikan daftar metadata sumber daya untuk daftar nama alur kerja tertentu. Setelah memanggil operasi `ListWorkflows`, Anda dapat memanggil operasi ini untuk mengakses data yang Anda telah diberikan izinnya. Operasi ini mendukung semua izin IAM, termasuk syarat izin yang menggunakan tag.

Permintaan

- `Names` – Wajib: Susunan string UTF-8, tidak kurang dari 1 atau lebih dari 25 string.

  Daftar nama alur kerja, mungkin nama yang dikembalikan dari operasi `ListWorkflows`.

- `IncludeGraph` – Boolean.

  Menentukan apakah akan menyertakan grafik ketika mengembalikan metadata sumber daya alur kerja.

Respons

- `Workflows` — Susunan objek Alur kerja, tidak kurang dari 1 atau tidak lebih dari 25 struktur.

  Daftar metadata sumber daya alur kerja.
- `MissingWorkflows` – Susunan string UTF-8, tidak kurang dari 1 atau lebih dari 25 string.

  Daftar nama alur kerja tidak ditemukan.

Kesalahan

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

GetWorkflowRun tindakan (Python: get_workflow_run)

Mengambil metadata untuk eksekusi alur kerja yang ditentukan. Riwayat Job run dapat diakses selama 90 hari untuk alur kerja dan pekerjaan Anda.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang sedang dijalankan.
- `RunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID dari eksekusi alur kerja.
- `IncludeGraph` – Boolean.

  Menentukan apakah akan menyertakan grafik alur kerja dalam respons atau tidak.

Respons

- `Run` — Sebuah objek WorkflowRun.

Metadata eksekusi alur kerja yang diminta.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetWorkflowRuns tindakan (Python: get_workflow_runs)

Mengambil metadata untuk semua eksekusi dari alur kerja yang ditentukan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang metadata eksekusinya harus dikembalikan.
- `IncludeGraph` – Boolean.

  Menentukan apakah akan menyertakan grafik alur kerja dalam respons atau tidak.
- `NextToken` – String UTF-8.

  Ukuran maksimum respons.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah maksimum eksekusi alur kerja yang akan disertakan dalam respons.

Respons

- `Runs` — Susunan objek WorkflowRun, tidak kurang dari 1 atau tidak lebih dari 1000 struktur.

  Daftar objek metadata eksekusi alur kerja.
- `NextToken` – String UTF-8.

Sebuah token kelanjutan, jika bukan semua eksekusi alur kerja yang diminta yang telah dikembalikan.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetWorkflowRunProperties tindakan (Python: get_workflow_run_properties)

Mengambil properti ekseksui alur kerja yang ditetapkan selama eksekusi.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang sudah dijalankan.
- `RunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID eksekusi alur kerja yang properti eksekusinya harus dikembalikan.

Respons

- `RunProperties` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah string UTF-8.

  Properti ekseksui alur kerja yang ditetapkan selama eksekusi yang ditentukan.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

PutWorkflowRunProperties tindakan (Python: put_workflow_run_properties)

Menempatkan properti eksekusi alur kerja yang ditentukan untuk eksekusi alur kerja yang diberikan. Jika sebuah properti sudah ada untuk eksekusi yang ditentukan, maka ia akan menimpa nilai atau menambahkan properti untuk properti yang sudah ada.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang sudah dijalankan.
- `RunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID eksekusi alur kerja yang properti eksekusinya harus diperbarui.
- `RunProperties` – Wajib: Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah string UTF-8.

  Properti yang akan ditempatkan untuk eksekusi yang ditentukan.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `AlreadyExistsException`
- `EntityNotFoundException`
- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentModificationException`

CreateBlueprint tindakan (Python: create_blueprint)

Mendaftarkan cetak biru dengan. AWS Glue

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Nama cetak birunya.
- `Description` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 512 byte.

  Deskripsi cetak biru.
- `BlueprintLocation` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 8192 byte, yang cocok dengan Custom string pattern #28.

  Menentukan sebuah path di Amazon S3 di mana cetak biru diterbitkan.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Tag yang akan diterapkan pada cetak biru ini.

Respons

- Name — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Mengembalikan nama cetak biru yang terdaftar.

Kesalahan

- AlreadyExistsException
- InvalidInputException
- OperationTimeoutException
- InternalServiceException
- ResourceNumberLimitExceededException

UpdateBlueprint tindakan (Python: update_blueprint)

Memperbarui cetak biru terdaftar.

Permintaan

- Name — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Nama cetak birunya.
- Description — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 512 byte.

  Deskripsi cetak biru.
- BlueprintLocation — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 8192 byte, yang cocok dengan Custom string pattern #28.

  Menentukan sebuah path di Amazon S3 di mana cetak biru diterbitkan.

Respons

- Name — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Mengembalikan nama cetak biru yang sudah diperbarui.

Kesalahan

- `EntityNotFoundException`
- `ConcurrentModificationException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `IllegalBlueprintStateException`

DeleteBlueprint tindakan (Python: delete_blueprint)

Menghapus sebuah cetak biru yang sudah ada.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama cetak biru yang akan dihapus.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Mengembalikan nama cetak biru yang sudah dihapus.

Kesalahan

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListBlueprints tindakan (Python: list_blueprints)

Mencantumkan semua nama cetak biru dalam sebuah akun.

Permintaan

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah permintaan kelanjutan.

- `MaxResults`— Angka (bilangan bulat), tidak kurang dari 1 atau lebih dari 25.

  Ukuran maksimum daftar yang akan dikembalikan.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Memfilter daftar dengan tag AWS sumber daya.

Respons

- `Blueprints` – Susunan string UTF-8.

  Daftar nama cetak biru dalam akun.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika tidak semua nama cetak biru yang sudah dikembalikan.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

BatchGetBlueprints tindakan (Python: batch_get_blueprints)

Mengambil informasi tentang sebuah daftar cetak biru.

Permintaan

- `Names` – Wajib: Susunan string UTF-8, tidak kurang dari 1 atau lebih dari 25 string.

  Daftar nama cetak biru.
- `IncludeBlueprint` – Boolean.

  Menentukan apakah akan memasukkan cetak biru dalam respon atau tidak.
- `IncludeParameterSpec` – Boolean.

  Menentukan apakah akan memasukkan parameter, sebagai string JSON, untuk cetak biru dalam respon atau tidak.

Respons

- `Blueprints` – Susunan objek Cetak biru.

  Mengembalikan daftar cetak biru sebagai objek `Blueprints`.
- `MissingBlueprints` – Susunan string UTF-8.

  Mengembalikan daftar `BlueprintNames` yang tidak ditemukan.

Kesalahan

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

StartBlueprintRun tindakan (Python: start_blueprint_run)

Memulai eksekusi baru dari cetak biru yang ditentukan.

Permintaan

- `BlueprintName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Nama cetak birunya.

- `Parameters` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 131072 byte.

  Menentukan parameter sebagai objek `BlueprintParameters`.

- `RoleArn` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 1024 byte, yang cocok dengan Custom string pattern #26.

  Menentukan IAM role yang digunakan untuk membuat alur kerja.

Respons

- `RunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID eksekusi untuk eksekusi cetak biru ini.

Kesalahan

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ResourceNumberLimitExceededException`
- `EntityNotFoundException`
- `IllegalBlueprintStateException`

GetBlueprintRun tindakan (Python: get_blueprint_run)

Mengambil detail dari sebuah eksekusi cetak biru.

Permintaan

- `BlueprintName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #27.

  Nama cetak birunya.

- `RunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID eksekusi untuk eksekusi cetak biru yang ingin Anda ambil.

Respons

- `BlueprintRun` — Sebuah objek BlueprintRun.

  Mengembalikan objek `BlueprintRun`.

Kesalahan

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`

GetBlueprintRuns tindakan (Python: get_blueprint_runs)

Mengambil detail eksekusi cetak biru untuk sebuah cetak biru yang ditentukan.

Permintaan

- `BlueprintName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama cetak birunya.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah permintaan kelanjutan.

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

Ukuran maksimum daftar yang akan dikembalikan.

Respons

- `BlueprintRuns` – Susunan objek BlueprintRun.

  Mengembalikan daftar objek `BlueprintRun`.

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika tidak semua eksekusi cetak biru yang sudah dikembalikan.

Kesalahan

- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`

StartWorkflowRun tindakan (Python: start_workflow_run)

Memulai eksekusi baru dari alur kerja yang ditentukan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang akan dimulai.

- `RunProperties` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah string UTF-8.

  Alur kerja menjalankan properti untuk menjalankan alur kerja baru.

Respons

- `RunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Id untuk eksekusi baru.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `ConcurrentRunsExceededException`

StopWorkflowRun tindakan (Python: stop_workflow_run)

Menghentikan eksekusi dari eksekusi alur kerja yang ditentukan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang akan dihentikan.
- `RunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID eksekusi alur kerja yang akan dihentikan.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `IllegalWorkflowStateException`

ResumeWorkflowRun tindakan (Python: resume_workflow_run)

Memulai ulang simpul yang dipilih dari eksekusi alur kerja sebelumnya yang selesai sebagian dan melanjutkan eksekusi alur kerja. Simpul yang dipilih dan semua simpul yang menjadi hilir dari simpul terpilih tersebut dijalankan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama alur kerja yang akan dilanjutkan.
- `RunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID eksekusi alur kerja yang akan dilanjutkan.
- `NodeIds` – Wajib: Susunan string UTF-8.

  Daftar ID simpul untuk simpul yang ingin Anda mulai ulang. Simpul yang akan dimulai ulang harus memiliki upaya eksekusi dalam eksekusi asli.

Respons

- `RunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID baru yang ditetapkan untuk eksekusi alur kerja yang dilanjutkan. Setiap eksekusi alur kerja yang dilanjutkan akan memiliki ID eksekusi baru.
- `NodeIds` – Susunan string UTF-8.

Daftar ID simpul untuk simpul yang benar-benar dimulai ulang.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `InternalServiceException`
- `OperationTimeoutException`
- `ConcurrentRunsExceededException`
- `IllegalWorkflowStateException`

Profil penggunaan

API profil Penggunaan menjelaskan tipe data dan API yang terkait dengan pembuatan, pembaruan, atau tampilan profil penggunaan di AWS Glue.

Jenis data

- ProfileConfiguration struktur
- ConfigurationObject struktur
- UsageProfileDefinition struktur

ProfileConfiguration struktur

Menentukan nilai pekerjaan dan sesi yang admin mengkonfigurasi dalam profil AWS Glue penggunaan.

Bidang

- `SessionConfiguration` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah sebuah objek ConfigurationObject A.

Peta nilai kunci parameter konfigurasi untuk AWS Glue sesi.

- `JobConfiguration` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah sebuah objek ConfigurationObject A.

  Peta nilai kunci parameter konfigurasi untuk AWS Glue pekerjaan.

ConfigurationObject struktur

Menentukan nilai-nilai yang ditetapkan admin untuk setiap pekerjaan atau sesi parameter dikonfigurasi dalam profil AWS Glue penggunaan.

Bidang

- `DefaultValue` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #31.

  Nilai default untuk parameter.

- `AllowedValues` – Susunan string UTF-8.

  Daftar nilai yang diizinkan untuk parameter.

- `MinValue` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #31.

  Nilai minimum yang diizinkan untuk parameter.

- `MaxValue` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 128 byte, yang cocok dengan Custom string pattern #31.

  Nilai maksimum yang diizinkan untuk parameter.

UsageProfileDefinition struktur

Menjelaskan profil AWS Glue penggunaan.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil penggunaan.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi profil penggunaan.
- `CreatedOn` — Stempel waktu.

  Tanggal dan waktu ketika profil penggunaan dibuat.
- `LastModifiedOn` — Stempel waktu.

  Tanggal dan waktu ketika profil penggunaan terakhir diubah.

Operasi

- CreateUsageProfile tindakan (Python: create_usage_profile)
- GetUsageProfile tindakan (Python: get_usage_profile)
- UpdateUsageProfile tindakan (Python: update_usage_profile)
- DeleteUsageProfile tindakan (Python: delete_usage_profile)
- ListUsageProfiles tindakan (Python: list_usage_profiles)

CreateUsageProfile tindakan (Python: create_usage_profile)

Membuat profil AWS Glue penggunaan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil penggunaan.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

Deskripsi profil penggunaan.

- `Configuration` — Wajib: Sebuah objek ProfileConfiguration.

  `ProfileConfiguration`Objek yang menentukan nilai pekerjaan dan sesi untuk profil.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Daftar tag yang diterapkan ke profil penggunaan.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil penggunaan yang dibuat.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `AlreadyExistsException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `OperationNotSupportedException`

GetUsageProfile tindakan (Python: get_usage_profile)

Mengambil informasi tentang profil AWS Glue penggunaan yang ditentukan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Nama profil penggunaan untuk mengambil.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil penggunaan.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi profil penggunaan.

- `Configuration` — Sebuah objek ProfileConfiguration.

  `ProfileConfiguration`Objek yang menentukan nilai pekerjaan dan sesi untuk profil.

- `CreatedOn` — Stempel waktu.

  Tanggal dan waktu ketika profil penggunaan dibuat.

- `LastModifiedOn` — Stempel waktu.

  Tanggal dan waktu ketika profil penggunaan terakhir diubah.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `OperationNotSupportedException`

UpdateUsageProfile tindakan (Python: update_usage_profile)

Perbarui profil AWS Glue penggunaan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil penggunaan.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi profil penggunaan.

- `Configuration` — Wajib: Sebuah objek ProfileConfiguration.

  `ProfileConfiguration`Objek yang menentukan nilai pekerjaan dan sesi untuk profil.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil penggunaan yang diperbarui.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `OperationNotSupportedException`
- `ConcurrentModificationException`

DeleteUsageProfile tindakan (Python: delete_usage_profile)

Menghapus profil penggunaan AWS Glue yang ditentukan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama profil penggunaan yang akan dihapus.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `OperationNotSupportedException`

ListUsageProfiles tindakan (Python: list_usage_profiles)

Buat daftar semua profil AWS Glue penggunaan.

Permintaan

- `NextToken` — String UTF-8, panjangnya tidak lebih dari 400000 byte.

  Sebuah token kelanjutan, disertakan jika ini adalah sebuah panggilan kelanjutan.
- `MaxResults` — Angka (bilangan bulat), tidak kurang dari 1 atau lebih dari 200.

  Jumlah maksimum profil penggunaan untuk kembali dalam satu respons.

Respons

- `Profiles` – Susunan objek UsageProfileDefinition.

  Daftar objek profil penggunaan (`UsageProfileDefinition`).
- `NextToken` — String UTF-8, panjangnya tidak lebih dari 400000 byte.

Sebuah token kelanjutan, ada jika segmen daftar saat ini bukan yang terakhir.

Kesalahan

- `InternalServiceException`
- `OperationTimeoutException`
- `InvalidInputException`
- `OperationNotSupportedException`

API pembelajaran mesin

Machine learning API menjelaskan tipe data machine learning, dan menyertakan API untuk membuat, menghapus, atau memperbarui transformasi, atau memulai tugas pembelajaran mesin.

Jenis data

- TransformParameters struktur
- EvaluationMetrics struktur
- Struktur MLTransform
- FindMatchesParameters struktur
- FindMatchesMetrics struktur
- ConfusionMatrix struktur
- GlueTable struktur
- TaskRun struktur
- TransformFilterCriteria struktur
- TransformSortCriteria struktur
- TaskRunFilterCriteria struktur
- TaskRunSortCriteria struktur
- TaskRunProperties struktur
- FindMatchesTaskRunProperties struktur
- ImportLabelsTaskRunProperties struktur
- ExportLabelsTaskRunProperties struktur

- LabelingSetGenerationTaskRunProperties struktur
- SchemaColumn struktur
- TransformEncryption struktur
- UserDataEncryption Struktur ML
- ColumnImportance struktur

TransformParameters struktur

Parameter spesifik algoritme yang dikaitkan dengan transformasi machine learning.

Bidang

- `TransformType` – Wajib: String UTF-8 (nilai yang valid: `FIND_MATCHES`).

  Jenis transformasi machine learning.

  Untuk informasi tentang jenis transformasi machine learning, lihat Membuat Machine Learning.
- `FindMatchesParameters` — Sebuah objek FindMatchesParameter.

  Parameter untuk menemukan kecocokan algoritme.

EvaluationMetrics struktur

Metrik evaluasi memberikan perkiraan kualitas dari transformasi machine learning Anda.

Bidang

- `TransformType` – Wajib: String UTF-8 (nilai yang valid: `FIND_MATCHES`).

  Jenis transformasi machine learning.
- `FindMatchesMetrics` — Sebuah objek FindMatchesMetrik.

  Metrik evaluasi untuk algoritme temukan kecocokan.

Struktur MLTransform

Struktur untuk sebuah transformasi machine learning.

Bidang

- `TransformId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID transformasi unik yang dihasilkan untuk transformasi machine learning. ID tersebut dijamin unik dan tidak berubah.

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama yang ditetapkan pengguna untuk transformasi machine learning. Nama tersebut tidak dijamin unik dan dapat diubah setiap saat.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah teks deskripsi dalam bentuk panjang yang ditetapkan pengguna untuk transformasi machine learning. Deskripsi tersebut tidak dijamin unik dan dapat diubah kapan saja.

- `Status` – String UTF-8 (nilai yang valid: `NOT_READY` | `READY` | `DELETING`).

  Status transformasi machine learning saat ini.

- `CreatedOn` — Stempel waktu.

  Sebuah stempel waktu. Waktu dan tanggal saat transformasi machine learning ini diciptakan.

- `LastModifiedOn` — Stempel waktu.

  Sebuah stempel waktu. Titik dalam waktu terakhir ketika transformasi machine learning ini dimodifikasi.

- `InputRecordTables` — Susunan objek GlueTable, tidak lebih dari 10 struktur.

  Daftar definisi AWS Glue tabel yang digunakan oleh transformasi.

- `Parameters` — Sebuah objek TransformParameters.

  Sebuah objek `TransformParameters`. Anda dapat menggunakan parameter untuk menyetel (menyesuaikan) perilaku transformasi machine learning dengan menentukan data apa yang dipelajari dan preferensi Anda pada berbagai tradeoff (seperti precious vs recall, atau akurasi vs biaya).

- `EvaluationMetrics` — Sebuah objek EvaluationMetrics.

Sebuah objek `EvaluationMetrics`. Metrik evaluasi memberikan perkiraan kualitas dari transformasi machine learning Anda.

- `LabelCount` — Nomor (bilangan bulat).

  Pengidentifikasi hitungan untuk file pelabelan yang dihasilkan oleh AWS Glue untuk transformasi ini. Karena Anda membuat transformasi yang lebih baik, maka Anda dapat secara berulang mengunduh, label, dan mengunggah file pelabelan.

- `Schema` — Susunan objek SchemaColumn, tidak lebih dari 100 struktur.

  Sebuah peta pasangan nilai-kunci yang mewakili kolom dan tipe data yang dapat dijalankan oleh transformasi ini terhadapnya. Memiliki batas atas 100 kolom.

- `Role` – String UTF-8.

  Nama atau Amazon Resource Name (ARN) dari IAM role dengan izin yang diperlukan. Izin yang diperlukan mencakup izin peran AWS Glue layanan ke AWS Glue sumber daya, dan izin Amazon S3 yang diperlukan oleh transformasi.

  - Peran ini membutuhkan izin peran AWS Glue layanan untuk memungkinkan akses ke sumber daya. AWS Glue Lihat Melampirkan Kebijakan untuk pengguna IAM yang Mengakses AWS Glue.
  - Peran ini memerlukan izin ke sumber Amazon Simple Storage Service (Amazon S3), target, direktori sementara, skrip, dan perpustakaan yang digunakan oleh tugas yang dijalankan untuk transformasi ini.

- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

  Nilai ini menentukan versi transformasi pembelajaran mesin AWS Glue ini yang kompatibel. Glue 1.0 direkomendasikan untuk sebagian besar pelanggan. Jika nilai tidak diatur, maka kompatibilitas Glue secara default diatur ke Glue 0.9. Untuk informasi selengkapnya, lihat Versi AWS Glue dalam panduan developer.

- `MaxCapacity` — Nomor (ganda).

  Jumlah unit pemrosesan AWS Glue data (DPU) yang dialokasikan untuk tugas berjalan untuk transformasi ini. Anda dapat mengalokasikan dari 2 hingga 100 DPU; default-nya adalah 10. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

`MaxCapacity` adalah pilihan yang eksklusif satu sama lain dengan `NumberOfWorkers` dan `WorkerType`.

- Jika `NumberOfWorkers` atau `WorkerType` diatur, maka `MaxCapacity` tidak dapat diatur.
- Jika `MaxCapacity` diatur, maka `NumberOfWorkers` atau `WorkerType` tidak dapat diatur.
- Jika `WorkerType` diatur, maka `NumberOfWorkers` wajib (dan sebaliknya).
- `MaxCapacity` dan `NumberOfWorkers`, keduanya minimal harus 1.

Saat bidang `WorkerType` diatur ke nilai selain `Standard`, maka bidang `MaxCapacity` diatur secara otomatis dan menjadi baca-saja.

- `WorkerType` – String UTF-8 (nilai yang valid: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan ketika sebuah tugas dari transformasi ini dieksekusi. Menerima nilai Standard, G.1X, atau G.2X.

  - Untuk jenis pekerja `Standard`, setiap pekerja menyediakan 4 vCPU, memori 16 GB dan disk 50GB, dan 2 pelaksana per pekerja.
  - Untuk jenis pekerja `G.1X`, setiap pekerja menyediakan 4 vCPU, memori 16 GB dan disk 64GB, dan 1 pelaksana per pekerja.
  - Untuk jenis pekerja `G.2X`, setiap pekerja menyediakan 8 vCPU, memori 32 GB dan disk 128GB, dan 1 pelaksana per pekerja.

  `MaxCapacity` adalah pilihan yang eksklusif satu sama lain dengan `NumberOfWorkers` dan `WorkerType`.

  - Jika `NumberOfWorkers` atau `WorkerType` diatur, maka `MaxCapacity` tidak dapat diatur.
  - Jika `MaxCapacity` diatur, maka `NumberOfWorkers` atau `WorkerType` tidak dapat diatur.
  - Jika `WorkerType` diatur, maka `NumberOfWorkers` wajib (dan sebaliknya).
  - `MaxCapacity` dan `NumberOfWorkers`, keduanya minimal harus 1.
- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah pekerja dari `workerType` yang ditentukan yang dialokasikan ketika sebuah tugas dari transformasi berjalan.

  Jika `WorkerType` diatur, maka `NumberOfWorkers` wajib (dan sebaliknya).
- `Timeout` — Nomor (bilangan bulat), minimal 1.

Waktu habis dalam hitungan menit dari transformasi machine learning.

- `MaxRetries` — Nomor (bilangan bulat).

  Jumlah percobaan maksimum untuk mencoba kembali setelah sebuah `MLTaskRun` dari transformasi machine learning gagal.

- `TransformEncryption` — Sebuah objek TransformEncryption.

  encryption-at-rest Pengaturan transformasi yang berlaku untuk mengakses data pengguna. Transformasi machine learning dapat mengakses data pengguna yang dienkripsi di Amazon S3 menggunakan KMS.

FindMatchesParameters struktur

Parameter untuk mengonfigurasi perubahan kecocokan penemuan.

Bidang

- `PrimaryKeyColumnName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 1024 byte, yang cocok dengan Single-line string pattern.

  Nama kolom yang secara unik mengidentifikasi baris di tabel sumber. Digunakan untuk membantu mengidentifikasi catatan yang cocok.

- `PrecisionRecallTradeoff` — Nomor (ganda), tidak lebih dari 1.0.

  Nilai yang dipilih saat menyetel transformasi Anda untuk keseimbangan antara presisi dan pemanggilan ulang. Nilai 0,5 artinya tidak ada preferensi; nilai 1,0 artinya bias murni untuk presisi, dan nilai 0,0 artinya bias untuk pemanggilan ulang. Karena ini adalah tradeoff, pemilihan nilai yang mendekati 1,0 artinya pemanggilan ulang yang sangat rendah, dan pemilihan nilai yang mendekati 0,0 menghasilkan presisi yang sangat rendah.

  Metrik presisi menunjukkan seberapa sering model Anda benar saat memprediksi kecocokan.

  Metrik pemanggilan ulang menunjukkan bahwa untuk kecocokan aktual, seberapa sering model Anda memprediksi kecocokan.

- `AccuracyCostTradeoff` — Nomor (ganda), tidak lebih dari 1.0.

  Nilai yang dipilih saat menyetel perubahan Anda untuk keseimbangan antara akurasi dan biaya. Nilai 0,5 artinya sistem menyeimbangkan masalah akurasi dan biaya. Nilai 1,0 artinya bias murni

untuk akurasi, yang biasanya menghasilkan biaya yang lebih tinggi, terkadang sangat tinggi. Nilai 0,0 berarti bias murni untuk biaya, yang menghasilkan perubahan `FindMatches` yang kurang akurat, terkadang dengan akurasi yang tidak dapat diterima.

Akurasi mengukur seberapa baik perubahan menemukan positif sejati dan negatif sejati. Peningkatan akurasi memerlukan lebih banyak sumber daya dan biaya mesin. Tetapi, itu juga menghasilkan peningkatan pemanggilan ulang.

Biaya mengukur berapa banyak sumber daya komputasi, sehingga uang, dikonsumsi untuk menjalankan perubahan.

- `EnforceProvidedLabels` – Boolean.

  Nilai yang diaktifkan atau dinonaktifkan untuk memaksa output agar cocok dengan label yang disediakan dari pengguna. Jika nilainya adalah `True`, perubahan `find matches` memaksa output agar cocok dengan label yang disediakan. Hasilnya menimpa hasil penggabungan normal. Jika nilainya adalah `False`, perubahan `find matches` tidak memastikan semua label yang disediakan diperhatikan, dan hasilnya bergantung pada model terlatih.

  Perhatikan bahwa pengaturan nilai ini ke betul dapat meningkatkan waktu eksekusi penggabungan.

FindMatchesMetrics struktur

Metrik evaluasi untuk algoritme temukan kecocokan. Kualitas transformasi machine learning Anda diukur dengan membuat transformasi Anda memprediksi beberapa kecocokan dan membandingkan hasilnya dengan kecocokan yang diketahui dari set data yang sama. Metrik kualitas didasarkan pada subset data Anda, sehingga mereka tidak tepat.

Bidang

- `AreaUnderPRCurve` — Nomor (ganda), tidak lebih dari 1.0.

  Daerah di bawah kurva presisi/recall (AUPRC) adalah sebuah nomor tunggal yang mengukur kualitas keseluruhan transformasi, yang independen dari pilihan yang dibuat untuk presisi vs recall. Nilai yang lebih tinggi menunjukkan bahwa Anda memiliki precision vs. recall tradeoff yang lebih menarik.

  Untuk informasi selengkapnya, lihat Precision dan recall di Wikipedia.
- `Precision` — Nomor (ganda), tidak lebih dari 1.0.

Metrik presisi menunjukkan seberapa sering transformasi Anda benar saat memprediksi kecocokan. Secara khusus, ia mengukur seberapa baik transformasi menemukan kemungkinan positif sejati dari total positif sejati.

Untuk informasi selengkapnya, lihat Precision dan recall di Wikipedia.

- `Recall` — Nomor (ganda), tidak lebih dari 1.0.

  Metrik pemanggilan ulang menunjukkan bahwa untuk kecocokan aktual, seberapa sering transformasi Anda memprediksi kecocokan. Secara khusus, ia mengukur seberapa baik transformasi menemukan benar positif dari total catatan dalam data sumber.

  Untuk informasi selengkapnya, lihat Precision dan recall di Wikipedia.

- `F1` — Nomor (ganda), tidak lebih dari 1.0.

  Metrik F1 maksimal menunjukkan akurasi transformasi antara 0 dan 1, di mana 1 adalah akurasi terbaik.

  Untuk informasi selengkapnya, lihat Skor F1 di Wikipedia.

- `ConfusionMatrix` — Sebuah objek ConfusionMatrix.

  Matriks kebingungan menunjukkan kepada Anda apa yang transformasi Anda prediksi secara akurat dan jenis kesalahan apa yang dibuat.

  Untuk informasi selengkapnya, lihat Matriks kebingungan di Wikipedia.

- `ColumnImportances` — Susunan objek ColumnImportance, tidak lebih dari 100 struktur.

  Daftar struktur `ColumnImportance` yang berisi metrik nilai penting kolom, yang diurutkan dalam urutan nilai penting yang semakin menurun.

ConfusionMatrix struktur

Matriks kebingungan menunjukkan kepada Anda apa yang transformasi Anda prediksi secara akurat dan jenis kesalahan apa yang dibuat.

Untuk informasi selengkapnya, lihat Matriks kebingungan di Wikipedia.

Bidang

- `NumTruePositives` — Nomor (panjang).

Jumlah kecocokan dalam data yang dengan benar ditemukan oleh transformasi, dalam matriks kebingungan untuk transformasi Anda.

- `NumFalsePositives` — Nomor (panjang).

  Jumlah ketidakcocokan dalam data yang keliru diklasifikasikan oleh transformasi sebagai sebuah kecocokan, dalam matriks kebingungan untuk transformasi Anda.

- `NumTrueNegatives` — Nomor (panjang).

  Jumlah ketidakcocokan dalam data yang dengan benar ditolak oleh transformasi, dalam matriks kebingungan untuk transformasi Anda.

- `NumFalseNegatives` — Nomor (panjang).

  Jumlah kecocokan dalam data yang tidak ditemukan oleh transformasi, dalam matriks kebingungan untuk transformasi Anda.

GlueTable struktur

Database dan tabel dalam AWS Glue Data Catalog yang digunakan untuk input atau output data.

Bidang

- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Sebuah nama basis data di AWS Glue Data Catalog.

- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Sebuah nama tabel di AWS Glue Data Catalog.

- `CatalogId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik untuk AWS Glue Data Catalog.

- `ConnectionName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama koneksi ke AWS Glue Data Catalog.

- `AdditionalOptions`— Sebuah array peta pasangan kunci-nilai, tidak kurang dari 1 atau lebih dari 10 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah string Deskripsi, panjangnya tidak lebih dari 2048 byte, cocok dengan. URI address multi-line string pattern

  Opsi tambahan untuk tabel. Saat ini ada dua kunci yang didukung:

  - `pushDownPredicate`: untuk memfilter pada partisi tanpa harus daftar dan membaca semua file dalam dataset Anda.
  - `catalogPartitionPredicate`: untuk menggunakan pemangkasan partisi sisi server menggunakan indeks partisi di. AWS Glue Data Catalog

TaskRun struktur

Parameter pengambilan sampel yang dikaitkan dengan transformasi machine learning.

Bidang

- `TransformId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik untuk transformasi.

- `TaskRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik untuk eksekusi tugas ini.

- `Status` – String UTF-8 (nilai yang valid: `STARTING` | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT`).

  Status eksekusi tugas yang diminta saat ini.

- `LogGroupName` – String UTF-8.

  Nama grup log untuk pencatatan log yang aman, yang dikaitkan dengan eksekusi tugas ini.

- `Properties` — Sebuah objek TaskRunProperti.

Menentukan properti konfigurasi yang dikaitkan dengan eksekusi tugas ini.

- `ErrorString` – String UTF-8.

  Daftar string kesalahan yang dikaitkan dengan eksekusi tugas ini.

- `StartedOn` — Stempel waktu.

  Tanggal dan waktu saat eksekusi tugas ini dimulai.

- `LastModifiedOn` — Stempel waktu.

  Titik dalam waktu terakhir saat eksekusi tugas yang diminta diperbarui.

- `CompletedOn` — Stempel waktu.

  Titik dalam waktu terakhir saat eksekusi tugas yang diminta selesai.

- `ExecutionTime` — Nomor (bilangan bulat).

  Jumlah waktu (dalam satuan detik) di mana eksekusi tugas ini menggunakan sumber daya.

TransformFilterCriteria struktur

Kriteria yang digunakan untuk mem-filter transformasi machine learning.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Sebuah nama transformasi unik yang digunakan untuk mem-filter transformasi machine learning.

- `TransformType` – String UTF-8 (nilai yang valid: `FIND_MATCHES`).

  Jenis transformasi machine learning yang digunakan untuk mem-filter transformasi machine learning.

- `Status` – String UTF-8 (nilai yang valid: `NOT_READY` | `READY` | `DELETING`).

  Memfilter daftar transformasi machine learning berdasarkan status transformasi terakhir yang diketahui (untuk menunjukkan apakah transformasi dapat digunakan atau tidak). Salah satu "NOT_READY", "READY", atau "DELETING".

- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

  Nilai ini menentukan versi transformasi pembelajaran mesin AWS Glue ini yang kompatibel. Glue 1.0 direkomendasikan untuk sebagian besar pelanggan. Jika nilai tidak diatur, maka kompatibilitas Glue secara default diatur ke Glue 0.9. Untuk informasi selengkapnya, lihat Versi AWS Glue dalam panduan developer.
- `CreatedBefore` — Stempel waktu.

  Waktu dan tanggal sebelum transformasi dibuat.
- `CreatedAfter` — Stempel waktu.

  Waktu dan tanggal setelah transformasi dibuat.
- `LastModifiedBefore` — Stempel waktu.

  Filter berdasarkan modifikasi terakhir transformasi sebelum tanggal ini.
- `LastModifiedAfter` — Stempel waktu.

  Filter berdasarkan modifikasi terakhir transformasi setelah tanggal ini.
- `Schema` — Susunan objek SchemaColumn, tidak lebih dari 100 struktur.

  Filter berdasarkan set data dengan skema tertentu. Objek `Map<Column, Type>` adalah sebuah rangkaian dari pasangan nilai kunci yang mewakili skema yang diterima transformasi ini, di mana `Column` adalah nama kolom, dan `Type` adalah jenis data seperti integer atau string. Memiliki batas atas 100 kolom.

TransformSortCriteria struktur

Kriteria pengurutan yang terkait dengan transformasi machine learning.

Bidang

- `Column` – Wajib: String UTF-8 (nilai yang valid: `NAME` | `TRANSFORM_TYPE` | `STATUS` | `CREATED` | `LAST_MODIFIED`).

  Kolom yang akan digunakan dalam kriteria pengurutan yang terkait dengan transformasi machine learning.
- `SortDirection` – Wajib: String UTF-8 (nilai yang valid: `DESCENDING` | `ASCENDING`).

Arah pengurutan yang akan digunakan dalam kriteria pengurutan yang terkait dengan transformasi machine learning.

TaskRunFilterCriteria struktur

Kriteria yang digunakan untuk mem-filter eksekusi tugas untuk transformasi machine learning.

Bidang

- `TaskRunType` – String UTF-8 (nilai yang valid: `EVALUATION` | `LABELING_SET_GENERATION` | `IMPORT_LABELS` | `EXPORT_LABELS` | `FIND_MATCHES`).

  Jenis eksekusi tugas.
- `Status` – String UTF-8 (nilai yang valid: `STARTING` | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT`).

  Status eksekusi tugas saat ini.
- `StartedBefore` — Stempel waktu.

  Filter berdasarkan eksekusi tugas yang dimulai sebelum tanggal ini.
- `StartedAfter` — Stempel waktu.

  Filter berdasarkan eksekusi tugas yang dimulai setelah tanggal ini.

TaskRunSortCriteria struktur

Kriteria pemilahan yang digunakan untuk mengurutkan daftar eksekusi tugas untuk transformasi machine learning.

Bidang

- `Column` – Wajib: String UTF-8 (nilai yang valid: `TASK_RUN_TYPE` | `STATUS` | `STARTED`).

  Kolom yang akan digunakan untuk mengurutkan daftar eksekusi tugas untuk transformasi machine learning.
- `SortDirection` – Wajib: String UTF-8 (nilai yang valid: `DESCENDING` | `ASCENDING`).

Arah pengurutan yang akan digunakan untuk mengurutkan daftar eksekusi tugas untuk transformasi machine learning.

TaskRunProperties struktur

Properti konfigurasi untuk eksekusi tugas.

Bidang

- `TaskType` – String UTF-8 (nilai yang valid: `EVALUATION` | `LABELING_SET_GENERATION` | `IMPORT_LABELS` | `EXPORT_LABELS` | `FIND_MATCHES`).

  Jenis eksekusi tugas.

- `ImportLabelsTaskRunProperties` — Sebuah objek ImportLabelsTaskRunProperti.

  Properti konfigurasi untuk eksekusi tugas label impor.

- `ExportLabelsTaskRunProperties` — Sebuah objek ExportLabelsTaskRunProperti.

  Properti konfigurasi untuk eksekusi tugas label ekspor.

- `LabelingSetGenerationTaskRunProperties` — Sebuah objek LabelingSetGenerationTaskRunProperties.

  Properti konfigurasi untuk eksekusi tugas pembuatan label set.

- `FindMatchesTaskRunProperties` — Sebuah objek FindMatchesTaskRunProperti.

  Properti konfigurasi untuk eksekusi tugas temukan kecocokan.

FindMatchesTaskRunProperties struktur

Menentukan properti konfigurasi untuk eksekusi tugas Temukan Kecocokan.

Bidang

- `JobId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID tugas untuk eksekusi tugas Temukan Kecocokan.

- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama yang ditetapkan untuk tugas untuk eksekusi tugas Temukan Kecocokan.
- `JobRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID eksekusi tugas untuk eksekusi tugas Temukan Kecocokan.

ImportLabelsTaskRunProperties struktur

Menentukan properti konfigurasi untuk eksekusi tugas label impor.

Bidang

- `InputS3Path` – String UTF-8.

  Path Amazon Simple Storage Service (Amazon S3) tempat Anda akan mengimpor label.
- `Replace` – Boolean.

  Menunjukkan apakah akan menimpa label yang ada.

ExportLabelsTaskRunProperties struktur

Menentukan properti konfigurasi untuk eksekusi tugas label ekspor.

Bidang

- `OutputS3Path` – String UTF-8.

  Path Amazon Simple Storage Service (Amazon S3) tempat Anda akan mengekspor label.

LabelingSetGenerationTaskRunProperties struktur

Menentukan properti konfigurasi untuk eksekusi tugas pembuatan label set.

Bidang

- `OutputS3Path` – String UTF-8.

Path Amazon Simple Storage Service (Amazon S3) tempat Anda akan membuat set pelabelan.

SchemaColumn struktur

Sebuah pasangan nilai-kunci yang mewakili kolom dan tipe data yang dapat dijalankan oleh transformasi ini terhadapnya. Parameter `Schema` dari `MLTransform` yang mungkin terdiri hingga 100 struktur ini.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 1024 byte, yang cocok dengan Single-line string pattern.

  Nama kolom.
- `DataType` — String UTF-8, sepanjang tidak lebih dari 131072, yang cocok dengan Single-line string pattern.

  Jenis data dalam kolom.

TransformEncryption struktur

encryption-at-rest Pengaturan transformasi yang berlaku untuk mengakses data pengguna. Transformasi machine learning dapat mengakses data pengguna yang dienkripsi di Amazon S3 menggunakan KMS.

Selain itu, label yang diimpor dan transformasi yang dilatih sekarang dapat dienkripsi menggunakan kunci KMS yang disediakan pelanggan.

Bidang

- `MlUserDataEncryption` — Sebuah objek UserDataEnkripsi ML.

  Sebuah objek `MLUserDataEncryption` yang berisi mode enkripsi dan ID kunci KMS yang disediakan pelanggan.
- `TaskRunSecurityConfigurationName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama konfigurasi keamanan.

UserDataEncryption Struktur ML

encryption-at-rest Pengaturan transformasi yang berlaku untuk mengakses data pengguna.

Bidang

- `MlUserDataEncryptionMode` – Wajib: String UTF-8 (nilai yang valid: `DISABLED` | `SSE-KMS="SSEKMS"`).

  Mode enkripsi yang diterapkan untuk data pengguna. Nilai yang valid adalah:

  - DISABLED: enkripsi dinonaktifkan
  - SSEKMS: penggunaan enkripsi sisi server dengan AWS Key Management Service (SSE-KMS) untuk data pengguna yang disimpan di Amazon S3.
- `KmsKeyId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID untuk kunci KMS yang disediakan oleh pelanggan.

ColumnImportance struktur

Sebuah struktur yang berisi nama kolom dan skor nilai penting kolom untuk sebuah kolom.

Nilai penting kolom membantu Anda memahami bagaimana kolom berkontribusi pada model Anda, dengan mengidentifikasi kolom mana dalam catatan Anda yang lebih penting daripada kolom yang lain.

Bidang

- `ColumnName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama sebuah kolom.
- `Importance` — Nomor (ganda), tidak lebih dari 1.0.

  Skor nilai penting kolom untuk kolom, dalam desimal.

Operasi

- Tindakan CreateMelTransform (Python: create_ml_transform)

- Tindakan updateMLTransform (Python: update_ml_transform)
- Tindakan DeletemlTransform (Python: delete_ml_transform)
- Tindakan getmlTransform (Python: get_ml_transform)
- Tindakan getmlTransforms (Python: get_ml_transforms)
- Tindakan ListmlTransforms (Python: list_ml_transforms)
- EvaluationTaskRun Tindakan startML (Python: start_ml_evaluation_task_run)
- LabelingSetGenerationTaskRun Tindakan startML (Python: start_ml_labeling_set_generation_task_run)
- TaskRun Tindakan GetMl (Python: get_ml_task_run)
- TaskRuns Tindakan GetMl (Python: get_ml_task_runs)
- TaskRun Tindakan CancelMl (Python: cancel_ml_task_run)
- StartExportLabelsTaskRun tindakan (Python: start_export_labels_task_run)
- StartImportLabelsTaskRun tindakan (Python: start_import_labels_task_run)

Tindakan CreateMelTransform (Python: create_ml_transform)

Menciptakan transformasi pembelajaran AWS Glue mesin. Operasi ini menciptakan transformasi dan semua parameter yang diperlukan untuk melatih transformasi tersebut.

Panggil operasi ini sebagai langkah pertama dalam proses menggunakan sebuah transformasi machine learning (seperti transformasi `FindMatches`) untuk deduplikasi data. Anda dapat memberikan opsi `Description`, selain parameter yang ingin Anda gunakan untuk algoritme Anda.

Anda juga harus menentukan parameter tertentu untuk tugas yang AWS Glue berjalan atas nama Anda sebagai bagian dari pembelajaran dari data Anda dan membuat transformasi pembelajaran mesin berkualitas tinggi. Parameter ini mencakup `Role`, dan opsional, `AllocatedCapacity`, `Timeout`, dan `MaxRetries`. Untuk informasi lebih lanjut, lihat Tugas.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama unik yang Anda berikan untuk transformasi pada saat Anda membuatnya.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

Sebuah deskripsi dari transformasi machine learning yang sedang didefinisikan. Default-nya adalah string kosong.

- `InputRecordTables` — Wajib: Susunan objek GlueTable, tidak lebih dari 10 struktur.

  Daftar definisi AWS Glue tabel yang digunakan oleh transformasi.

- `Parameters` — Wajib: Sebuah objek TransformParameters.

  Parameter algoritmik yang spesifik untuk jenis transformasi yang digunakan. Secara kondisional tergantung pada jenis transformasi.

- `Role` – Wajib: String UTF-8.

  Nama atau Amazon Resource Name (ARN) dari IAM role dengan izin yang diperlukan. Izin yang diperlukan mencakup izin peran AWS Glue layanan ke AWS Glue sumber daya, dan izin Amazon S3 yang diperlukan oleh transformasi.

  - Peran ini membutuhkan izin peran AWS Glue layanan untuk memungkinkan akses ke sumber daya. AWS Glue Lihat Melampirkan Kebijakan untuk pengguna IAM yang Mengakses AWS Glue.
  - Peran ini memerlukan izin ke sumber Amazon Simple Storage Service (Amazon S3), target, direktori sementara, skrip, dan perpustakaan yang digunakan oleh tugas yang dijalankan untuk transformasi ini.

- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

  Nilai ini menentukan versi transformasi pembelajaran mesin AWS Glue ini yang kompatibel. Glue 1.0 direkomendasikan untuk sebagian besar pelanggan. Jika nilai tidak diatur, maka kompatibilitas Glue secara default diatur ke Glue 0.9. Untuk informasi selengkapnya, lihat Versi AWS Glue dalam panduan developer.

- `MaxCapacity` — Nomor (ganda).

  Jumlah unit pemrosesan AWS Glue data (DPU) yang dialokasikan untuk tugas berjalan untuk transformasi ini. Anda dapat mengalokasikan dari 2 hingga 100 DPU; default-nya adalah 10. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

  `MaxCapacity` adalah pilihan yang eksklusif satu sama lain dengan `NumberOfWorkers` dan `WorkerType`.

- Jika NumberOfWorkers atau WorkerType diatur, maka MaxCapacity tidak dapat diatur.
- Jika MaxCapacity diatur, maka NumberOfWorkers atau WorkerType tidak dapat diatur.
- Jika WorkerType diatur, maka NumberOfWorkers wajib (dan sebaliknya).
- MaxCapacity dan NumberOfWorkers, keduanya minimal harus 1.

Saat bidang WorkerType diatur ke nilai selain Standard, maka bidang MaxCapacity diatur secara otomatis dan menjadi baca-saja.

Saat bidang WorkerType diatur ke nilai selain Standard, maka bidang MaxCapacity diatur secara otomatis dan menjadi baca-saja.

- WorkerType – String UTF-8 (nilai yang valid: Standard="" | G.1X="" | G.2X="" | G.025X="" | G.4X="" | G.8X="" | Z.2X="").

  Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan ketika tugas ini dieksekusi. Menerima nilai Standard, G.1X, atau G.2X.

  - Untuk jenis pekerja Standard, setiap pekerja menyediakan 4 vCPU, memori 16 GB dan disk 50GB, dan 2 pelaksana per pekerja.
  - Untuk jenis pekerja G.1X, setiap pekerja menyediakan 4 vCPU, memori 16 GB dan disk 64GB, dan 1 pelaksana per pekerja.
  - Untuk jenis pekerja G.2X, setiap pekerja menyediakan 8 vCPU, memori 32 GB dan disk 128GB, dan 1 pelaksana per pekerja.

  MaxCapacity adalah pilihan yang eksklusif satu sama lain dengan NumberOfWorkers dan WorkerType.

  - Jika NumberOfWorkers atau WorkerType diatur, maka MaxCapacity tidak dapat diatur.
  - Jika MaxCapacity diatur, maka NumberOfWorkers atau WorkerType tidak dapat diatur.
  - Jika WorkerType diatur, maka NumberOfWorkers wajib (dan sebaliknya).
  - MaxCapacity dan NumberOfWorkers, keduanya minimal harus 1.
- NumberOfWorkers — Nomor (bilangan bulat).

  Jumlah pekerja dari workerType ditentukan yang dialokasikan ketika tugas ini dieksekusi.

  Jika WorkerType diatur, maka NumberOfWorkers wajib (dan sebaliknya).
- Timeout — Nomor (bilangan bulat), minimal 1.

Waktu habis untuk eksekusi tugas untuk transformasi ini, dalam satuan menit. Ini adalah waktu maksimum yang bisa digunakan oleh eksekusi tugas untuk transformasi ini untuk menggunakan sumber daya sebelum eksekusi dihentikan dan memasuki status `TIMEOUT`. Nilai default-nya adalah 2.880 menit (48 jam).

- `MaxRetries` — Nomor (bilangan bulat).

  Jumlah percobaan maksimum untuk mencoba kembali tugas untuk transformasi ini setelah eksekusi tugas gagal.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Tag yang akan digunakan dengan transformasi machine learning ini. Anda dapat menggunakan tag untuk membatasi akses ke transformasi machine learning tersebut. Untuk informasi selengkapnya tentang AWS tag AWS Glue, lihat Tag AWS Glue di panduan pengembang.

- `TransformEncryption` — Sebuah objek TransformEncryption.

  encryption-at-rest Pengaturan transformasi yang berlaku untuk mengakses data pengguna. Transformasi machine learning dapat mengakses data pengguna yang dienkripsi di Amazon S3 menggunakan KMS.

Respons

- `TransformId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Sebuah pengenal unik yang dihasilkan untuk transformasi.

Kesalahan

- `AlreadyExistsException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

- `AccessDeniedException`
- `ResourceNumberLimitExceededException`
- `IdempotentParameterMismatchException`

Tindakan updateMLTransform (Python: update_ml_transform)

Memperbarui transformasi machine learning yang ada. Panggil operasi ini untuk menyetel parameter algoritme untuk mencapai hasil yang lebih baik.

Setelah memanggil operasi ini, Anda dapat memanggil operasi `StartMLEvaluationTaskRun` untuk menilai seberapa baik parameter baru Anda mencapai tujuan Anda (seperti meningkatkan kualitas transformasi machine learning Anda, atau membuatnya menjadi lebih hemat biaya).

Permintaan

- `TransformId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengenal unik yang dihasilkan saat transformasi dibuat.
- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama unik yang Anda berikan pada transformasi saat Anda membuatnya.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah deskripsi transformasi. Default-nya adalah string kosong.
- `Parameters` — Sebuah objek TransformParameters.

  Parameter konfigurasi yang spesifik untuk jenis transformasi (algoritme) yang digunakan. Secara kondisional tergantung pada jenis transformasi.
- `Role` – String UTF-8.

  Nama atau Amazon Resource Name (ARN) dari IAM role dengan izin yang diperlukan.
- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

Nilai ini menentukan versi transformasi pembelajaran mesin AWS Glue ini yang kompatibel. Glue 1.0 direkomendasikan untuk sebagian besar pelanggan. Jika nilai tidak diatur, maka kompatibilitas Glue secara default diatur ke Glue 0.9. Untuk informasi selengkapnya, lihat Versi AWS Glue dalam panduan developer.

- `MaxCapacity` — Nomor (ganda).

  Jumlah unit pemrosesan AWS Glue data (DPU) yang dialokasikan untuk tugas berjalan untuk transformasi ini. Anda dapat mengalokasikan dari 2 hingga 100 DPU; default-nya adalah 10. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

  Saat bidang `WorkerType` diatur ke nilai selain `Standard`, maka bidang `MaxCapacity` diatur secara otomatis dan menjadi baca-saja.

- `WorkerType` – String UTF-8 (nilai yang valid: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan ketika tugas ini dieksekusi. Menerima nilai Standard, G.1X, atau G.2X.

  - Untuk jenis pekerja `Standard`, setiap pekerja menyediakan 4 vCPU, memori 16 GB dan disk 50GB, dan 2 pelaksana per pekerja.
  - Untuk jenis pekerja `G.1X`, setiap pekerja menyediakan 4 vCPU, memori 16 GB dan disk 64GB, dan 1 pelaksana per pekerja.
  - Untuk jenis pekerja `G.2X`, setiap pekerja menyediakan 8 vCPU, memori 32 GB dan disk 128GB, dan 1 pelaksana per pekerja.

- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah pekerja dari `workerType` ditentukan yang dialokasikan ketika tugas ini dieksekusi.

- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Waktu habis untuk sebuah eksekusi tugas untuk transformasi ini, dalam satuan menit. Ini adalah waktu maksimum yang bisa digunakan oleh eksekusi tugas untuk transformasi ini untuk menggunakan sumber daya sebelum eksekusi dihentikan dan memasuki status `TIMEOUT`. Nilai default-nya adalah 2.880 menit (48 jam).

- `MaxRetries` — Nomor (bilangan bulat).

Jumlah percobaan maksimum untuk mencoba kembali tugas untuk transformasi ini setelah eksekusi tugas gagal.

Respons

- `TransformId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik untuk transformasi yang sudah diperbarui.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `AccessDeniedException`

Tindakan DeletemlTransform (Python: delete_ml_transform)

Menghapus transformasi pembelajaran AWS Glue mesin. Transformasi machine learning adalah jenis transformasi khusus yang menggunakan machine learning untuk mempelajari detail transformasi yang akan dilakukan dengan belajar dari contoh yang diberikan oleh manusia. Transformasi ini kemudian diselamatkan oleh AWS Glue. Jika Anda tidak lagi memerlukan sebuah transformasi, maka Anda dapat menghapusnya dengan memanggil `DeleteMLTransforms`. Namun, AWS Glue pekerjaan apa pun yang masih mereferensikan transformasi yang dihapus tidak akan berhasil lagi.

Permintaan

- `TransformId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik dari transformasi yang akan dihapus.

Respons

- `TransformId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik dari transformasi yang sudah dihapus.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

Tindakan getmlTransform (Python: get_ml_transform)

Mendapat artefak transformasi pembelajaran AWS Glue mesin dan semua metadata yang sesuai. Transformasi machine learning adalah jenis transformasi khusus yang menggunakan machine learning untuk mempelajari detail transformasi yang akan dilakukan dengan belajar dari contoh yang diberikan oleh manusia. Transformasi ini kemudian diselamatkan oleh AWS Glue. Anda dapat mengambil metadata mereka dengan memanggil `GetMLTransform`.

Permintaan

- `TransformId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik transformasi, yang dihasilkan pada saat transformasi dibuat.

Respons

- `TransformId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik transformasi, yang dihasilkan pada saat transformasi dibuat.
- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Nama unik yang diberikan pada transformasi saat transformasi tersebut dibuat.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah deskripsi transformasi.

- `Status` – String UTF-8 (nilai yang valid: `NOT_READY` | `READY` | `DELETING`).

  Status terakhir yang diketahui dari transformasi (untuk menunjukkan apakah transformasi dapat digunakan atau tidak). Salah satu "NOT_READY", "READY", atau "DELETING".

- `CreatedOn` — Stempel waktu.

  Tanggal dan waktu saat transformasi dibuat.

- `LastModifiedOn` — Stempel waktu.

  Tanggal dan waktu saat transformasi terakhir diubah.

- `InputRecordTables` — Susunan objek GlueTable, tidak lebih dari 10 struktur.

  Daftar definisi AWS Glue tabel yang digunakan oleh transformasi.

- `Parameters` — Sebuah objek TransformParameters.

  Parameter konfigurasi khusus untuk algoritme yang digunakan.

- `EvaluationMetrics` — Sebuah objek EvaluationMetrics.

  Metrik evaluasi terbaru.

- `LabelCount` — Nomor (bilangan bulat).

  Jumlah label yang tersedia untuk transformasi ini.

- `Schema` — Susunan objek SchemaColumn, tidak lebih dari 100 struktur.

  Objek `Map<Column, Type>` yang mewakili skema yang diterima transformasi ini. Memiliki batas atas 100 kolom.

- `Role` – String UTF-8.

  Nama atau Amazon Resource Name (ARN) dari IAM role dengan izin yang diperlukan.

- `GlueVersion` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Custom string pattern #20.

Nilai ini menentukan versi transformasi pembelajaran mesin AWS Glue ini yang kompatibel. Glue 1.0 direkomendasikan untuk sebagian besar pelanggan. Jika nilai tidak diatur, maka kompatibilitas Glue secara default diatur ke Glue 0.9. Untuk informasi selengkapnya, lihat Versi AWS Glue dalam panduan developer.

- `MaxCapacity` — Nomor (ganda).

  Jumlah unit pemrosesan AWS Glue data (DPU) yang dialokasikan untuk tugas berjalan untuk transformasi ini. Anda dapat mengalokasikan dari 2 hingga 100 DPU; default-nya adalah 10. DPU adalah ukuran relatif daya pemrosesan yang terdiri dari 4 vCPU kapasitas komputasi dan 16 GB memori. Untuk informasi lebih lanjut, lihat halaman harga AWS Glue.

  Saat bidang `WorkerType` diatur ke nilai selain `Standard`, maka bidang `MaxCapacity` diatur secara otomatis dan menjadi baca-saja.

- `WorkerType` – String UTF-8 (nilai yang valid: `Standard=""` | `G.1X=""` | `G.2X=""` | `G.025X=""` | `G.4X=""` | `G.8X=""` | `Z.2X=""`).

  Jenis pekerja yang telah ditetapkan sebelumnya yang dialokasikan ketika tugas ini dieksekusi. Menerima nilai Standard, G.1X, atau G.2X.

  - Untuk jenis pekerja `Standard`, setiap pekerja menyediakan 4 vCPU, memori 16 GB dan disk 50GB, dan 2 pelaksana per pekerja.
  - Untuk jenis pekerja `G.1X`, setiap pekerja menyediakan 4 vCPU, memori 16 GB dan disk 64GB, dan 1 pelaksana per pekerja.
  - Untuk jenis pekerja `G.2X`, setiap pekerja menyediakan 8 vCPU, memori 32 GB dan disk 128GB, dan 1 pelaksana per pekerja.

- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah pekerja dari `workerType` ditentukan yang dialokasikan ketika tugas ini dieksekusi.

- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Waktu habis untuk sebuah eksekusi tugas untuk transformasi ini, dalam satuan menit. Ini adalah waktu maksimum yang bisa digunakan oleh eksekusi tugas untuk transformasi ini untuk menggunakan sumber daya sebelum eksekusi dihentikan dan memasuki status `TIMEOUT`. Nilai default-nya adalah 2.880 menit (48 jam).

- `MaxRetries` — Nomor (bilangan bulat).

Jumlah percobaan maksimum untuk mencoba kembali tugas untuk transformasi ini setelah eksekusi tugas gagal.

- `TransformEncryption` — Sebuah objek TransformEncryption.

  encryption-at-rest Pengaturan transformasi yang berlaku untuk mengakses data pengguna. Transformasi machine learning dapat mengakses data pengguna yang dienkripsi di Amazon S3 menggunakan KMS.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

Tindakan getmlTransforms (Python: get_ml_transforms)

Mendapat daftar transformasi pembelajaran AWS Glue mesin yang dapat diurutkan dan dapat disaring. Transformasi machine learning adalah jenis transformasi khusus yang menggunakan machine learning untuk mempelajari detail transformasi yang akan dilakukan dengan belajar dari contoh yang diberikan oleh manusia. Transformasi ini kemudian disimpan oleh AWS Glue, dan Anda dapat mengambil metadata mereka dengan menelepon. `GetMLTransforms`

Permintaan

- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman untuk mengimbangi hasil.

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah hasil maksimum yang akan dikembalikan.

- `Filter` — Sebuah objek TransformFilterKriteria.

  Kriteria transformasi filter.

- `Sort` — Sebuah objek TransformSortKriteria.

  Kriteria pengurutan.

Respons

- `Transforms` – Wajib: Susunan objek MLTransform.

  Sebuah daftar transformasi machine learning.

- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman, jika ada lebih banyak hasil yang tersedia.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

Tindakan ListmlTransforms (Python: list_ml_transforms)

Mengambil daftar yang dapat diurutkan dan disaring dari transformasi pembelajaran AWS Glue mesin yang ada di AWS akun ini, atau sumber daya dengan tag yang ditentukan. Operasi ini mengambil kolom `Tags` opsional, yang dapat Anda gunakan sebagai sebuah filter respon sehingga tag sumber daya dapat diambil sebagai sebuah grup. Jika Anda memilih untuk menggunakan pem-filter-an tag, maka hanya sumber daya dengan tag saja yang diambil.

Permintaan

- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika ini adalah permintaan kelanjutan.

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Ukuran maksimum daftar yang akan dikembalikan.

- `Filter` — Sebuah objek TransformFilterKriteria.

  Sebuah `TransformFilterCriteria` yang digunakan untuk mem-filter transformasi machine learning.

- `Sort` — Sebuah objek TransformSortKriteria.

Sebuah `TransformSortCriteria` yang digunakan untuk mengurutkan transformasi machine learning.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Menentukan untuk mengembalikan hanya sumber daya ditandai saja.

Respons

- `TransformIds` – Wajib: Susunan string UTF-8.

  Pengidentifikasi semua transformasi machine learning dalam akun, atau transformasi machine learning dengan tag yang ditentukan.
- `NextToken` – String UTF-8.

  Sebuah token kelanjutan, jika daftar yang dikembalikan tidak berisi metrik terakhir yang tersedia.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

EvaluationTaskRun Tindakan startML (Python: start_ml_evaluation_task_run)

Memulai tugas untuk memperkirakan kualitas transformasi.

Ketika Anda memberikan set label sebagai contoh kebenaran, pembelajaran AWS Glue mesin menggunakan beberapa contoh tersebut untuk belajar darinya. Selebihnya label digunakan sebagai pengujian untuk memperkirakan kualitas.

Mengembalikan sebuah pengidentifikasi unik untuk eksekusi. Anda dapat memanggil `GetMLTaskRun` untuk mendapatkan informasi lebih lanjut tentang statistik `EvaluationTaskRun`.

Permintaan

- `TransformId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik dari transformasi machine learning.

Respons

- `TaskRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik yang terkait dengan eksekusi ini.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ConcurrentRunsExceededException`
- `MLTransformNotReadyException`

LabelingSetGenerationTaskRun Tindakan startML (Python: start_ml_labeling_set_generation_task_run)

Memulai alur kerja pembelajaran aktif untuk transformasi machine learning Anda untuk meningkatkan kualitas transformasi dengan menghasilkan set label dan menambahkan label.

Saat `StartMLLabelingSetGenerationTaskRun` selesai, AWS Glue akan menghasilkan "set pelabelan" atau serangkaian pertanyaan yang harus dijawab oleh manusia.

Dalam kasus transformasi `FindMatches`, pertanyaan-pertanyaan ini berbentuk, "Apa cara yang benar untuk mengelompokkan baris ini bersama-sama ke dalam grup yang seluruhnya terdiri dari catatan yang cocok?"

Setelah proses pelabelan selesai, Anda dapat mengunggah label Anda dengan memanggil `StartImportLabelsTaskRun`. Setelah `StartImportLabelsTaskRun` selesai, semua eksekusi transformasi machine learning di masa depan akan menggunakan label baru dan yang telah ditingkatkan dan melakukan transformasi kualitas yang lebih tinggi.

Permintaan

- `TransformId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik dari transformasi machine learning.

- `OutputS3Path` – Wajib: String UTF-8.

  Path Amazon Simple Storage Service (Amazon S3) tempat Anda membuat set pelabelan.

Respons

- `TaskRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi eksekusi unik yang dikaitkan dengan eksekusi tugas ini.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ConcurrentRunsExceededException`

TaskRun Tindakan GetMl (Python: get_ml_task_run)

Mendapatkan detail untuk eksekusi tugas tertentu pada sebuah transformasi machine learning. Proses tugas pembelajaran mesin adalah tugas asinkron yang AWS Glue berjalan atas nama Anda sebagai bagian dari berbagai alur kerja pembelajaran mesin. Anda dapat memeriksa statistik dari setiap tugas yang dijalankan dengan memanggil `GetMLTaskRun` dengan `TaskRunID` dan `TransformID` dari transformasi induknya.

Permintaan

- `TransformId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik dari transformasi machine learning.
- `TaskRunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik eksekusi tugas.

Respons

- `TransformId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik eksekusi tugas.
- `TaskRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi eksekusi unik yang dikaitkan dengan eksekusi ini.
- `Status` – String UTF-8 (nilai yang valid: `STARTING` | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT`).

  Status untuk eksekusi tugas ini.
- `LogGroupName` – String UTF-8.

  Nama-nama grup log yang dikaitkan dengan eksekusi tugas.
- `Properties` — Sebuah objek TaskRunProperti.

  Daftar properti yang dikaitkan dengan eksekusi tugas.

- `ErrorString` – String UTF-8.

  String kesalahan yang dikaitkan dengan eksekusi tugas.

- `StartedOn` — Stempel waktu.

  Tanggal dan waktu ketika eksekusi tugas ini dimulai.

- `LastModifiedOn` — Stempel waktu.

  Tanggal dan waktu ketika eksekusi tugas ini terakhir diubah.

- `CompletedOn` — Stempel waktu.

  Tanggal dan waktu ketika eksekusi tugas ini selesai.

- `ExecutionTime` — Nomor (bilangan bulat).

  Jumlah waktu (dalam satuan detik) di mana eksekusi tugas ini menggunakan sumber daya.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

TaskRuns Tindakan GetMl (Python: get_ml_task_runs)

Mendapatkan sebuah daftar eksekusi untuk transformasi machine learning. Proses tugas pembelajaran mesin adalah tugas asinkron yang AWS Glue berjalan atas nama Anda sebagai bagian dari berbagai alur kerja pembelajaran mesin. Anda bisa mendapatkan daftar tugas machine learning yang dapat diurutkan dan disaring dengan memanggil `GetMLTaskRuns` dengan `TransformID` dari transformasi induknya dan parameter opsional lainnya seperti yang didokumentasikan di bagian ini.

Operasi ini mengembalikan daftar eksekusi historis dan harus diberi nomor halaman.

Permintaan

- `TransformId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Pengidentifikasi unik dari transformasi machine learning.

- `NextToken` – String UTF-8.

  Sebuah token untuk pemberian nomor halaman hasil. Nilai default adalah (kosong).

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah hasil maksimum yang akan dikembalikan.

- `Filter` — Sebuah objek TaskRunFilterCriteria.

  Kriteria filter, dalam struktur `TaskRunFilterCriteria`, untuk eksekusi tugas.

- `Sort` — Sebuah objek TaskRunSortCriteria.

  Kriteria pengurutan, dalam struktur `TaskRunSortCriteria`, untuk eksekusi tugas.

Respons

- `TaskRuns` – Susunan objek TaskRun.

  Daftar eksekusi tugas yang dikaitkan dengan transformasi.

- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman, jika ada lebih banyak hasil yang tersedia.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

TaskRun Tindakan CancelMl (Python: cancel_ml_task_run)

Membatalkan (menghentikan) sebuah eksekusi tugas. Proses tugas pembelajaran mesin adalah tugas asinkron yang AWS Glue berjalan atas nama Anda sebagai bagian dari berbagai alur kerja pembelajaran mesin. Anda dapat membatalkan sebuah eksekusi tugas machine learning kapan

saja dengan memanggil `CancelMLTaskRun` dengan `TransformID` transformasi induk dari sebuah eksekusi tugas dan `TaskRunId` dari eksekusi tugas.

Permintaan

- `TransformId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik dari transformasi machine learning.
- `TaskRunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Sebuah pengidentifikasi unik untuk eksekusi tugas.

Respons

- `TransformId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik dari transformasi machine learning.
- `TaskRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik untuk eksekusi tugas.
- `Status` – String UTF-8 (nilai yang valid: `STARTING` | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT`).

  Status untuk eksekusi ini.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

StartExportLabelsTaskRun tindakan (Python: start_export_labels_task_run)

Memulai tugas asinkron untuk mengekspor semua data berlabel untuk sebuah transformasi tertentu. Tugas ini adalah satu-satunya panggilan API terkait label yang bukan merupakan bagian dari alur kerja pembelajaran yang tipikal. Anda biasanya menggunakan `StartExportLabelsTaskRun` saat Anda ingin bekerja dengan semua label yang ada pada saat yang sama, seperti saat Anda ingin menghapus atau mengubah label yang sebelumnya dikirimkan sebagai kebenaran. Operasi API ini menerima `TransformId` yang labelnya ingin Anda ekspor dan path Amazon Simple Storage Service (Amazon S3) yang akan digunakan untuk mengekspor label. Operasi tersebut mengembalikan `TaskRunId`. Anda dapat memeriksa status dari eksekusi tugas Anda dengan memanggil API `GetMLTaskRun`.

Permintaan

- `TransformId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik dari transformasi machine learning.
- `OutputS3Path` – Wajib: String UTF-8.

  Path Amazon S3 di mana Anda mengekspor label.

Respons

- `TaskRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik untuk eksekusi tugas.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

StartImportLabelsTaskRun tindakan (Python: start_import_labels_task_run)

Memungkinkan Anda untuk memberikan label tambahan (contoh kebenaran) yang akan digunakan untuk mengajarkan transformasi machine learning dan meningkatkan kualitasnya. Operasi API ini pada umumnya digunakan sebagai bagian dari alur kerja pembelajaran aktif yang dimulai dengan panggilan `StartMLLabelingSetGenerationTaskRun` dan yang akhirnya menghasilkan peningkatan kualitas pada transformasi machine learning Anda.

Setelah `StartMLLabelingSetGenerationTaskRun` selesai, machine learning AWS Glue akan menghasilkan serangkaian pertanyaan yang akan dijawab oleh manusia. (Menjawab pertanyaan-pertanyaan ini sering disebut 'pelabelan' dalam alur kerja machine learning). Dalam kasus transformasi `FindMatches`, pertanyaan-pertanyaan ini berbentuk, "Apa cara yang benar untuk mengelompokkan baris ini bersama-sama ke dalam grup yang seluruhnya terdiri dari catatan yang cocok?" Setelah proses pelabelan selesai, pengguna mengunggah jawaban/label mereka dengan sebuah panggilan ke `StartImportLabelsTaskRun`. Setelah `StartImportLabelsTaskRun` selesai, semua eksekusi transformasi machine learning di masa depan menggunakan label baru dan yang telah ditingkatkan dan melakukan transformasi kualitas yang lebih tinggi.

Secara default, `StartMLLabelingSetGenerationTaskRun` terus belajar dari dan menggabungkan semua label yang Anda unggah kecuali Anda mengatur `Replace` ke BETUL. Jika Anda mengatur `Replace` ke BETUL, `StartImportLabelsTaskRun` akan menghapus dan melupakan semua label yang diunggah sebelumnya dan hanya belajar hanya dari set yang Anda unggah. Mengganti label dapat membantu jika Anda menyadari bahwa Anda sebelumnya mengunggah label yang salah, dan Anda meyakini bahwa label tersebut memiliki efek negatif pada kualitas transformasi Anda.

Anda dapat memeriksa status dari eksekusi tugas Anda dengan memanggil operasi `GetMLTaskRun`.

Permintaan

- `TransformId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik dari transformasi machine learning.

- `InputS3Path` – Wajib: String UTF-8.

  Path Amazon Simple Storage Service (Amazon S3) tempat Anda mengimpor label.

- `ReplaceAllLabels` – Boolean.

Menunjukkan apakah akan menimpa label yang ada.

Respons

- `TaskRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi unik untuk eksekusi tugas.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `ResourceNumberLimitExceededException`
- `InternalServiceException`

API Kualitas Data

API Kualitas Data menjelaskan tipe data kualitas data, dan menyertakan API untuk membuat, menghapus, atau memperbarui kumpulan aturan, proses, dan evaluasi kualitas data.

Jenis data

- DataSource struktur
- DataQualityRulesetListDetails struktur
- DataQualityTargetTable struktur
- DataQualityRulesetEvaluationRunDescription struktur
- DataQualityRulesetEvaluationRunFilter struktur
- DataQualityEvaluationRunAdditionalRunOptions struktur
- DataQualityRuleRecommendationRunDescription struktur
- DataQualityRuleRecommendationRunFilter struktur
- DataQualityResult struktur

- DataQualityAnalyzerResult struktur
- DataQualityObservation struktur
- MetricBasedObservation struktur
- DataQualityMetricValues struktur
- DataQualityRuleResult struktur
- DataQualityResultDescription struktur
- DataQualityResultFilterCriteria struktur
- DataQualityRulesetFilterCriteria struktur

DataSource struktur

Sumber data (AWS Glue tabel) yang Anda inginkan hasil kualitas datanya.

Bidang

- `GlueTable` — Wajib: Sebuah objek GlueTable.

  Sebuah AWS Glue meja.

DataQualityRulesetListDetails struktur

Menjelaskan kumpulan aturan kualitas data yang dikembalikan oleh. `GetDataQualityRuleset`

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama aturan kualitas data.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi aturan kualitas data.
- `CreatedOn` — Stempel waktu.

  Tanggal dan waktu kumpulan aturan kualitas data dibuat.

- `LastModifiedOn` — Stempel waktu.

  Tanggal dan waktu aturan kualitas data terakhir diubah.

- `TargetTable` — Sebuah objek DataQualityTargetTable.

  Objek yang mewakili AWS Glue tabel.

- `RecommendationRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Ketika kumpulan aturan dibuat dari rekomendasi yang dijalankan, ID run ini dihasilkan untuk menghubungkan keduanya bersama-sama.

- `RuleCount` — Nomor (bilangan bulat).

  Jumlah aturan dalam aturan.

DataQualityTargetTable struktur

Objek yang mewakili AWS Glue tabel.

Bidang

- `TableName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama AWS Glue meja.

- `DatabaseName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama database tempat AWS Glue tabel ada.

- `CatalogId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Id katalog tempat AWS Glue tabel ada.

DataQualityRulesetEvaluationRunDescription struktur

Menjelaskan hasil evaluasi set aturan kualitas data.

Bidang

- `RunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi eksekusi unik yang dikaitkan dengan eksekusi ini.

- `Status` – String UTF-8 (nilai yang valid: `STARTING` | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT`).

  Status untuk eksekusi ini.

- `StartedOn` — Stempel waktu.

  Tanggal dan waktu saat lari dimulai.

- `DataSource` — Sebuah objek DataSource.

  Sumber data (AWS Glue tabel) yang terkait dengan proses.

DataQualityRulesetEvaluationRunFilter struktur

Kriteria filter.

Bidang

- `DataSource` — Wajib: Sebuah objek DataSource.

  Filter berdasarkan sumber data (AWS Glue tabel) yang terkait dengan proses.

- `StartedBefore` — Stempel waktu.

  Filter hasil berdasarkan proses yang dimulai sebelum waktu ini.

- `StartedAfter` — Stempel waktu.

  Filter hasil berdasarkan proses yang dimulai setelah waktu ini.

DataQualityEvaluationRunAdditionalRunOptions struktur

Opsi run tambahan yang dapat Anda tentukan untuk menjalankan evaluasi.

Bidang

- `CloudWatchMetricsEnabled` – Boolean.

  Apakah akan mengaktifkan CloudWatch metrik atau tidak.

- `ResultsS3Prefix` – String UTF-8.

  Awalan untuk Amazon S3 untuk menyimpan hasil.

- `CompositeRuleEvaluationMethod` – String UTF-8 (nilai yang valid: `COLUMN` | `ROW`).

  Tetapkan metode evaluasi untuk aturan komposit dalam kumpulan aturan ke ROW/COLUMN

DataQualityRuleRecommendationRunDescription struktur

Menjelaskan hasil dari rekomendasi aturan kualitas data yang dijalankan.

Bidang

- `RunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi eksekusi unik yang dikaitkan dengan eksekusi ini.

- `Status` – String UTF-8 (nilai yang valid: `STARTING` | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT`).

  Status untuk eksekusi ini.

- `StartedOn` — Stempel waktu.

  Tanggal dan waktu kapan proses ini dimulai.

- `DataSource` — Sebuah objek DataSource.

  Sumber data (AWS Glue tabel) yang terkait dengan rekomendasi dijalankan.

DataQualityRuleRecommendationRunFilter struktur

Filter untuk mencantumkan rekomendasi kualitas data berjalan.

Bidang

- `DataSource` — Wajib: Sebuah objek DataSource.

  Filter berdasarkan sumber data tertentu (AWS Glue tabel).
- `StartedBefore` — Stempel waktu.

  Filter berdasarkan waktu untuk hasil dimulai sebelum waktu yang ditentukan.
- `StartedAfter` — Stempel waktu.

  Filter berdasarkan waktu untuk hasil dimulai setelah waktu yang ditentukan.

DataQualityResult struktur

Menjelaskan hasil kualitas data.

Bidang

- `ResultId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID hasil unik untuk hasil kualitas data.
- `Score` — Nomor (ganda), tidak lebih dari 1.0.

  Skor kualitas data agregat. Merupakan rasio aturan yang diteruskan ke jumlah total aturan.
- `DataSource` — Sebuah objek DataSource.

  Tabel yang terkait dengan hasil kualitas data, jika ada.
- `RulesetName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama ruleset terkait dengan hasil kualitas data.
- `EvaluationContext` – String UTF-8.

  Dalam konteks pekerjaan di AWS Glue Studio, setiap node di kanvas biasanya diberi semacam nama dan node kualitas data akan memiliki nama. Dalam kasus beberapa node, `evaluationContext` dapat membedakan node.
- `StartedOn` — Stempel waktu.

Tanggal dan waktu ketika kualitas data ini berjalan dimulai.

- `CompletedOn` — Stempel waktu.

  Tanggal dan waktu ketika kualitas data ini berjalan selesai.

- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pekerjaan yang terkait dengan hasil kualitas data, jika ada.

- `JobRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID job run terkait dengan hasil kualitas data, jika ada.

- `RulesetEvaluationRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID run unik untuk evaluasi set aturan untuk hasil kualitas data ini.

- `RuleResults`— Sebuah array DataQualityRuleResult objek, tidak lebih dari 2000 struktur.

  Daftar `DataQualityRuleResult` objek yang mewakili hasil untuk setiap aturan.

- `AnalyzerResults`— Sebuah array DataQualityAnalyzerResult objek, tidak lebih dari 2000 struktur.

  Daftar `DataQualityAnalyzerResult` objek yang mewakili hasil untuk setiap analyzer.

- `Observations` — Susunan objek DataQualityObservation, tidak lebih dari 50 struktur.

  Daftar `DataQualityObservation` objek yang mewakili pengamatan yang dihasilkan setelah mengevaluasi aturan dan penganalisis.

DataQualityAnalyzerResult struktur

Menjelaskan hasil evaluasi penganalisis kualitas data.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama penganalisis kualitas data.

- `Description` — String UTF-8, sepanjang tidak lebih dari 2048, yang cocok dengan URI address multi-line string pattern.

  Deskripsi penganalisis kualitas data.

- `EvaluationMessage` — String UTF-8, sepanjang tidak lebih dari 2048, yang cocok dengan URI address multi-line string pattern.

  Pesan evaluasi.

- `EvaluatedMetrics` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah Angka (ganda).

  Peta metrik yang terkait dengan evaluasi penganalisis.

DataQualityObservation struktur

Menjelaskan pengamatan yang dihasilkan setelah mengevaluasi aturan dan penganalisis.

Bidang

- `Description` — String UTF-8, sepanjang tidak lebih dari 2048, yang cocok dengan URI address multi-line string pattern.

  Deskripsi pengamatan kualitas data.

- `MetricBasedObservation` — Sebuah objek MetricBasedObservation.

  Objek tipe yang `MetricBasedObservation` mewakili pengamatan yang didasarkan pada metrik kualitas data yang dievaluasi.

MetricBasedObservation struktur

Menjelaskan pengamatan berbasis metrik yang dihasilkan berdasarkan metrik kualitas data yang dievaluasi.

Bidang

- `MetricName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama metrik kualitas data yang digunakan untuk menghasilkan pengamatan.

- `MetricValues` — Sebuah objek DataQualityMetricValues.

  Objek tipe `DataQualityMetricValues` yang mewakili analisis nilai metrik kualitas data.

- `NewRules` – Susunan string UTF-8.

  Daftar aturan kualitas data baru yang dihasilkan sebagai bagian dari pengamatan berdasarkan nilai metrik kualitas data.

DataQualityMetricValues struktur

Menjelaskan nilai metrik kualitas data menurut analisis data historis.

Bidang

- `ActualValue` — Nomor (ganda).

  Nilai aktual dari metrik kualitas data.

- `ExpectedValue` — Nomor (ganda).

  Nilai yang diharapkan dari metrik kualitas data menurut analisis data historis.

- `LowerLimit` — Nomor (ganda).

  Batas bawah nilai metrik kualitas data menurut analisis data historis.

- `UpperLimit` — Nomor (ganda).

  Batas atas nilai metrik kualitas data menurut analisis data historis.

DataQualityRuleResult struktur

Menjelaskan hasil evaluasi aturan kualitas data.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama aturan kualitas data.

- `Description` — String UTF-8, sepanjang tidak lebih dari 2048, yang cocok dengan URI address multi-line string pattern.

  Deskripsi aturan kualitas data.

- `EvaluationMessage` — String UTF-8, sepanjang tidak lebih dari 2048, yang cocok dengan URI address multi-line string pattern.

  Pesan evaluasi.

- `Result` – String UTF-8 (nilai yang valid: `PASS` | `FAIL` | `ERROR`).

  Status lulus atau gagal untuk aturan.

- `EvaluatedMetrics` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah Angka (ganda).

  Peta metrik yang terkait dengan evaluasi aturan.

DataQualityResultDescription struktur

Menjelaskan hasil kualitas data.

Bidang

- `ResultId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID hasil unik untuk hasil kualitas data ini.

- `DataSource` — Sebuah objek DataSource.

  Nama tabel yang terkait dengan hasil kualitas data.

- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pekerjaan yang terkait dengan hasil kualitas data.

- `JobRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID job run terkait dengan hasil kualitas data.

- `StartedOn` — Stempel waktu.

  Waktu proses dimulai untuk hasil kualitas data ini.

DataQualityResultFilterCriteria struktur

Kriteria yang digunakan untuk mengembalikan hasil kualitas data.

Bidang

- `DataSource` — Sebuah objek DataSource.

  Filter hasil berdasarkan sumber data yang ditentukan. Misalnya, mengambil semua hasil untuk sebuah AWS Glue tabel.

- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Filter hasil dengan nama pekerjaan yang ditentukan.

- `JobRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Filter hasil berdasarkan ID job run yang ditentukan.

- `StartedAfter` — Stempel waktu.

  Filter hasil berdasarkan proses yang dimulai setelah waktu ini.

- `StartedBefore` — Stempel waktu.

  Filter hasil berdasarkan proses yang dimulai sebelum waktu ini.

DataQualityRulesetFilterCriteria struktur

Kriteria yang digunakan untuk menyaring kumpulan aturan kualitas data.

Bidang

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama kriteria filter ruleset.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi kriteria filter ruleset.
- `CreatedBefore` — Stempel waktu.

  Filter pada set aturan yang dibuat sebelum tanggal ini.
- `CreatedAfter` — Stempel waktu.

  Filter pada set aturan yang dibuat setelah tanggal ini.
- `LastModifiedBefore` — Stempel waktu.

  Filter pada aturan set terakhir diubah sebelum tanggal ini.
- `LastModifiedAfter` — Stempel waktu.

  Filter pada aturan set terakhir diubah setelah tanggal ini.
- `TargetTable` — Sebuah objek DataQualityTargetTable.

  Nama dan nama database dari tabel target.

Operasi

- StartDataQualityRulesetEvaluationRun tindakan (Python: start_data_quality_ruleset_evaluation_run)
- CancelDataQualityRulesetEvaluationRun tindakan (Python: cancel_data_quality_ruleset_evaluation_run)
- GetDataQualityRulesetEvaluationRun tindakan (Python: get_data_quality_ruleset_evaluation_run)

- ListDataQualityRulesetEvaluationRuns tindakan (Python: list_data_quality_ruleset_evaluation_runs)
- StartDataQualityRuleRecommendationRun tindakan (Python: start_data_quality_rule_recommendation_run)
- CancelDataQualityRuleRecommendationRun tindakan (Python: cancel_data_quality_rule_recommendation_run)
- GetDataQualityRuleRecommendationRun tindakan (Python: get_data_quality_rule_recommendation_run)
- ListDataQualityRuleRecommendationRuns tindakan (Python: list_data_quality_rule_recommendation_runs)
- GetDataQualityResult tindakan (Python: get_data_quality_result)
- BatchGetDataQualityResult tindakan (Python: batch_get_data_quality_result)
- ListDataQualityResults tindakan (Python: list_data_quality_results)
- CreateDataQualityRuleset tindakan (Python: create_data_quality_ruleset)
- DeleteDataQualityRuleset tindakan (Python: delete_data_quality_ruleset)
- GetDataQualityRuleset tindakan (Python: get_data_quality_ruleset)
- ListDataQualityRulesets tindakan (Python: list_data_quality_rulesets)
- UpdateDataQualityRuleset tindakan (Python: update_data_quality_ruleset)

StartDataQualityRulesetEvaluationRun tindakan (Python: start_data_quality_ruleset_evaluation_run)

Setelah Anda memiliki definisi kumpulan aturan (baik yang direkomendasikan atau milik Anda sendiri), Anda memanggil operasi ini untuk mengevaluasi kumpulan aturan terhadap sumber data (tabel).AWS Glue Evaluasi menghitung hasil yang dapat Anda ambil dengan API. `GetDataQualityResult`

Permintaan

- `DataSource` — Wajib: Sebuah objek DataSource.

  Sumber data (AWS Glue tabel) yang terkait dengan proses ini.

- `Role` – Wajib: String UTF-8.

  IAM Peran yang diberikan untuk mengenkripsi hasil proses.

- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah `G.1X` pekerja yang akan digunakan dalam pelarian. Default-nya adalah 5.

- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Batas waktu untuk berlari dalam hitungan menit. Ini adalah waktu maksimum yang dijalankan dapat mengkonsumsi sumber daya sebelum dihentikan dan memasuki `TIMEOUT` status. Default-nya adalah 2.880 menit (48 jam).

- `ClientToken` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Digunakan untuk idempotensi dan direkomendasikan untuk disetel ke ID acak (seperti UUID) untuk menghindari membuat atau memulai beberapa instance dari sumber daya yang sama.

- `AdditionalRunOptions` — Sebuah objek DataQualityEvaluationRunAdditionalRunOptions.

  Opsi run tambahan yang dapat Anda tentukan untuk menjalankan evaluasi.

- `RulesetNames`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 10 string.

  Daftar nama ruleset.

- `AdditionalDataSources` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah sebuah objek DataSource A.

  Peta string referensi ke sumber data tambahan yang dapat Anda tentukan untuk menjalankan evaluasi.

Respons

- `RunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi eksekusi unik yang dikaitkan dengan eksekusi ini.

Kesalahan

- `InvalidInputException`
- `EntityNotFoundException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ConflictException`

CancelDataQualityRulesetEvaluationRun tindakan (Python: cancel_data_quality_ruleset_evaluation_run)

Membatalkan proses di mana kumpulan aturan sedang dievaluasi terhadap sumber data.

Permintaan

- `RunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi eksekusi unik yang dikaitkan dengan eksekusi ini.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityRulesetEvaluationRun tindakan (Python: get_data_quality_ruleset_evaluation_run)

Mengambil run tertentu di mana kumpulan aturan dievaluasi terhadap sumber data.

Permintaan

- `RunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi eksekusi unik yang dikaitkan dengan eksekusi ini.

Respons

- `RunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi eksekusi unik yang dikaitkan dengan eksekusi ini.
- `DataSource` — Sebuah objek DataSource.

  Sumber data (AWS Glue tabel) yang terkait dengan evaluasi ini dijalankan.
- `Role` – String UTF-8.

  IAM Peran yang diberikan untuk mengenkripsi hasil proses.
- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah `G.1X` pekerja yang akan digunakan dalam pelarian. Default-nya adalah 5.
- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Batas waktu untuk berlari dalam hitungan menit. Ini adalah waktu maksimum yang dijalankan dapat mengkonsumsi sumber daya sebelum dihentikan dan memasuki `TIMEOUT` status. Default-nya adalah 2.880 menit (48 jam).
- `AdditionalRunOptions` — Sebuah objek DataQualityEvaluationRunAdditionalRunOptions.

  Opsi run tambahan yang dapat Anda tentukan untuk menjalankan evaluasi.
- `Status` – String UTF-8 (nilai yang valid: `STARTING` | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT`).

  Status untuk eksekusi ini.
- `ErrorString` – String UTF-8.

  String kesalahan yang terkait dengan proses.
- `StartedOn` — Stempel waktu.

Tanggal dan waktu kapan proses ini dimulai.

- `LastModifiedOn` — Stempel waktu.

  Sebuah stempel waktu. Poin terakhir saat rekomendasi aturan kualitas data ini dijalankan telah dimodifikasi.

- `CompletedOn` — Stempel waktu.

  Tanggal dan waktu saat proses ini selesai.

- `ExecutionTime` — Nomor (bilangan bulat).

  Jumlah waktu (dalam detik) bahwa run mengkonsumsi sumber daya.

- `RulesetNames`— Array string UTF-8, tidak kurang dari 1 atau lebih dari 10 string.

  Daftar nama ruleset untuk lari. Saat ini, parameter ini hanya mengambil satu nama Ruleset.

- `ResultIds`— Array string UTF-8, tidak kurang dari 1 atau lebih dari 10 string.

  Daftar ID hasil untuk hasil kualitas data untuk dijalankan.

- `AdditionalDataSources` – Susunan peta pasangan nilai kunci.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Setiap nilai adalah sebuah objek DataSource A.

  Peta string referensi ke sumber data tambahan yang dapat Anda tentukan untuk menjalankan evaluasi.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityRulesetEvaluationRuns tindakan (Python: list_data_quality_ruleset_evaluation_runs)

Daftar semua proses yang memenuhi kriteria filter, di mana kumpulan aturan dievaluasi terhadap sumber data.

Permintaan

- `Filter` — Sebuah objek DataQualityRulesetEvaluationRunFilter.

  Kriteria filter.

- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman untuk mengimbangi hasil.

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah hasil maksimum yang akan dikembalikan.

Respons

- `Runs` – Susunan objek DataQualityRulesetEvaluationRunDescription.

  Sebuah daftar `DataQualityRulesetEvaluationRunDescription` objek yang mewakili kualitas data ruleset berjalan.

- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman, jika ada lebih banyak hasil yang tersedia.

Kesalahan

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

StartDataQualityRuleRecommendationRun tindakan (Python: start_data_quality_rule_recommendation_run)

Memulai proses rekomendasi yang digunakan untuk menghasilkan aturan ketika Anda tidak tahu aturan apa yang harus ditulis. AWS Glue Kualitas Data menganalisis data dan menghasilkan rekomendasi untuk kumpulan aturan potensial. Anda kemudian dapat melakukan triase set aturan dan memodifikasi kumpulan aturan yang dihasilkan sesuai keinginan Anda.

Rekomendasi berjalan secara otomatis dihapus setelah 90 hari.

Permintaan

- `DataSource` — Wajib: Sebuah objek DataSource.

  Sumber data (AWS Glue tabel) yang terkait dengan proses ini.

- `Role` – Wajib: String UTF-8.

  IAM Peran yang diberikan untuk mengenkripsi hasil proses.

- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah `G.1X` pekerja yang akan digunakan dalam pelarian. Default-nya adalah 5.

- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Batas waktu untuk berlari dalam hitungan menit. Ini adalah waktu maksimum yang dijalankan dapat mengkonsumsi sumber daya sebelum dihentikan dan memasuki `TIMEOUT` status. Default-nya adalah 2.880 menit (48 jam).

- `CreatedRulesetName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Sebuah nama untuk ruleset.

- `ClientToken` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Digunakan untuk idempotensi dan direkomendasikan untuk disetel ke ID acak (seperti UUID) untuk menghindari membuat atau memulai beberapa instance dari sumber daya yang sama.

Respons

- `RunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi eksekusi unik yang dikaitkan dengan eksekusi ini.

Kesalahan

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ConflictException`

CancelDataQualityRuleRecommendationRun tindakan (Python: cancel_data_quality_rule_recommendation_run)

Membatalkan proses rekomendasi yang ditentukan yang digunakan untuk menghasilkan aturan.

Permintaan

- `RunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi eksekusi unik yang dikaitkan dengan eksekusi ini.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityRuleRecommendationRun tindakan (Python: get_data_quality_rule_recommendation_run)

Mendapatkan rekomendasi tertentu yang dijalankan yang digunakan untuk menghasilkan aturan.

Permintaan

- `RunId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi eksekusi unik yang dikaitkan dengan eksekusi ini.

Respons

- `RunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Pengidentifikasi eksekusi unik yang dikaitkan dengan eksekusi ini.
- `DataSource` — Sebuah objek DataSource.

  Sumber data (AWS Glue tabel) yang terkait dengan proses ini.
- `Role` – String UTF-8.

  IAM Peran yang diberikan untuk mengenkripsi hasil proses.
- `NumberOfWorkers` — Nomor (bilangan bulat).

  Jumlah `G.1X` pekerja yang akan digunakan dalam pelarian. Default-nya adalah 5.
- `Timeout` — Nomor (bilangan bulat), minimal 1.

  Batas waktu untuk berlari dalam hitungan menit. Ini adalah waktu maksimum yang dijalankan dapat mengkonsumsi sumber daya sebelum dihentikan dan memasuki `TIMEOUT` status. Default-nya adalah 2.880 menit (48 jam).
- `Status` – String UTF-8 (nilai yang valid: `STARTING` | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT`).

  Status untuk eksekusi ini.
- `ErrorString` – String UTF-8.

String kesalahan yang terkait dengan proses.

- `StartedOn` — Stempel waktu.

  Tanggal dan waktu kapan proses ini dimulai.

- `LastModifiedOn` — Stempel waktu.

  Sebuah stempel waktu. Poin terakhir saat rekomendasi aturan kualitas data ini dijalankan telah dimodifikasi.

- `CompletedOn` — Stempel waktu.

  Tanggal dan waktu saat proses ini selesai.

- `ExecutionTime` — Nomor (bilangan bulat).

  Jumlah waktu (dalam detik) bahwa run mengkonsumsi sumber daya.

- `RecommendedRuleset`— String UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 65536 byte.

  Ketika proses rekomendasi aturan mulai selesai, itu membuat kumpulan aturan yang direkomendasikan (seperangkat aturan). Anggota ini memiliki aturan tersebut dalam format Data Quality Definition Language (DQDL).

- `CreatedRulesetName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama ruleset yang dibuat oleh run.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityRuleRecommendationRuns tindakan (Python: list_data_quality_rule_recommendation_runs)

Daftar rekomendasi berjalan memenuhi kriteria filter.

Permintaan

- `Filter` — Sebuah objek DataQualityRuleRecommendationRunFilter.

  Kriteria filter.

- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman untuk mengimbangi hasil.

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah hasil maksimum yang akan dikembalikan.

Respons

- `Runs` – Susunan objek DataQualityRuleRecommendationRunDescription.

  Daftar objek `DataQualityRuleRecommendationRunDescription`.

- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman, jika ada lebih banyak hasil yang tersedia.

Kesalahan

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityResult tindakan (Python: get_data_quality_result)

Mengambil hasil evaluasi aturan kualitas data.

Permintaan

- `ResultId` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID hasil unik untuk hasil kualitas data.

Respons

- `ResultId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID hasil unik untuk hasil kualitas data.
- `Score` — Nomor (ganda), tidak lebih dari 1.0.

  Skor kualitas data agregat. Merupakan rasio aturan yang diteruskan ke jumlah total aturan.
- `DataSource` — Sebuah objek DataSource.

  Tabel yang terkait dengan hasil kualitas data, jika ada.
- `RulesetName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama ruleset terkait dengan hasil kualitas data.
- `EvaluationContext` – String UTF-8.

  Dalam konteks pekerjaan di AWS Glue Studio, setiap node di kanvas biasanya diberi semacam nama dan node kualitas data akan memiliki nama. Dalam kasus beberapa node, `evaluationContext` dapat membedakan node.
- `StartedOn` — Stempel waktu.

  Tanggal dan waktu ketika proses untuk hasil kualitas data ini dimulai.
- `CompletedOn` — Stempel waktu.

  Tanggal dan waktu ketika proses untuk hasil kualitas data ini selesai.
- `JobName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pekerjaan yang terkait dengan hasil kualitas data, jika ada.

- `JobRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID job run terkait dengan hasil kualitas data, jika ada.

- `RulesetEvaluationRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID run unik yang terkait dengan evaluasi ruleset.

- `RuleResults`— Sebuah array DataQualityRuleResult objek, tidak lebih dari 2000 struktur.

  Daftar `DataQualityRuleResult` objek yang mewakili hasil untuk setiap aturan.

- `AnalyzerResults`— Sebuah array DataQualityAnalyzerResult objek, tidak lebih dari 2000 struktur.

  Daftar `DataQualityAnalyzerResult` objek yang mewakili hasil untuk setiap analyzer.

- `Observations` — Susunan objek DataQualityObservation, tidak lebih dari 50 struktur.

  Daftar `DataQualityObservation` objek yang mewakili pengamatan yang dihasilkan setelah mengevaluasi aturan dan penganalisis.

Kesalahan

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `EntityNotFoundException`

BatchGetDataQualityResult tindakan (Python: batch_get_data_quality_result)

Mengambil daftar hasil kualitas data untuk ID hasil yang ditentukan.

Permintaan

- `ResultIds`— Diperlukan: Sebuah array string UTF-8, tidak kurang dari 1 atau lebih dari 100 string.

Daftar ID hasil unik untuk hasil kualitas data.

Respons

- `Results` – Wajib: Susunan objek DataQualityResult.

  Daftar `DataQualityResult` objek yang mewakili hasil kualitas data.
- `ResultsNotFound`— Array string UTF-8, tidak kurang dari 1 atau lebih dari 100 string.

  Daftar ID hasil yang hasilnya tidak ditemukan.

Kesalahan

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityResults tindakan (Python: list_data_quality_results)

Mengembalikan semua hasil eksekusi kualitas data untuk akun Anda.

Permintaan

- `Filter` — Sebuah objek DataQualityResultFilterCriteria.

  Kriteria filter.
- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman untuk mengimbangi hasil.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah hasil maksimum yang akan dikembalikan.

Respons

- `Results` – Wajib: Susunan objek DataQualityResultDescription.

Daftar objek `DataQualityResultDescription`.

- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman, jika ada lebih banyak hasil yang tersedia.

Kesalahan

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

CreateDataQualityRuleset tindakan (Python: create_data_quality_ruleset)

Membuat aturan kualitas data dengan aturan DQDL diterapkan ke tabel tertentu. AWS Glue

Anda membuat kumpulan aturan menggunakan Data Quality Definition Language (DQDL). Untuk informasi selengkapnya, lihat panduan AWS Glue pengembang.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama unik untuk kumpulan aturan kualitas data.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi aturan kualitas data.
- `Ruleset`- Diperlukan: string UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 65536 byte.

  Aturan Bahasa Definisi Kualitas Data (DQDL). Untuk informasi selengkapnya, lihat panduan AWS Glue pengembang.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

Daftar tag yang diterapkan pada kumpulan aturan kualitas data.

- `TargetTable` — Sebuah objek DataQualityTargetTable.

  Tabel target yang terkait dengan kumpulan aturan kualitas data.

- `RecommendationRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID run unik untuk menjalankan rekomendasi.

- `ClientToken` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Digunakan untuk idempotensi dan direkomendasikan untuk disetel ke ID acak (seperti UUID) untuk menghindari membuat atau memulai beberapa instance dari sumber daya yang sama.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama unik untuk kumpulan aturan kualitas data.

Kesalahan

- `InvalidInputException`
- `AlreadyExistsException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ResourceNumberLimitExceededException`

DeleteDataQualityRuleset tindakan (Python: delete_data_quality_ruleset)

Menghapus aturan kualitas data.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama untuk kumpulan aturan kualitas data.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

GetDataQualityRuleset tindakan (Python: get_data_quality_ruleset)

Mengembalikan aturan yang ada dengan identifier atau nama.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama ruleset.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama ruleset.
- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

Deskripsi dari ruleset.

- `Ruleset`— String UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 65536 byte.

  Aturan Bahasa Definisi Kualitas Data (DQDL). Untuk informasi selengkapnya, lihat panduan AWS Glue pengembang.

- `TargetTable` — Sebuah objek DataQualityTargetTable.

  Nama dan nama database dari tabel target.

- `CreatedOn` — Stempel waktu.

  Sebuah stempel waktu. Waktu dan tanggal pembuatan aturan kualitas data ini.

- `LastModifiedOn` — Stempel waktu.

  Sebuah stempel waktu. Poin terakhir dalam waktu ketika aturan kualitas data ini dimodifikasi.

- `RecommendationRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Ketika kumpulan aturan dibuat dari rekomendasi yang dijalankan, ID run ini dihasilkan untuk menghubungkan keduanya bersama-sama.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

ListDataQualityRulesets tindakan (Python: list_data_quality_rulesets)

Mengembalikan daftar paginasi rulesets untuk daftar tabel yang ditentukan. AWS Glue

Permintaan

- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman untuk mengimbangi hasil.

- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

Jumlah hasil maksimum yang akan dikembalikan.

- `Filter` — Sebuah objek DataQualityRulesetFilterCriteria.

  Kriteria filter.

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Daftar tag pasangan kunci-nilai.

Respons

- `Rulesets` – Susunan objek DataQualityRulesetListDetails.

  Daftar aturan paginasi untuk daftar tabel yang ditentukan. AWS Glue

- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman, jika ada lebih banyak hasil yang tersedia.

Kesalahan

- `EntityNotFoundException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

UpdateDataQualityRuleset tindakan (Python: update_data_quality_ruleset)

Memperbarui aturan kualitas data yang ditentukan.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

Nama aturan kualitas data.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi dari ruleset.

- `Ruleset`— String UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 65536 byte.

  Aturan Bahasa Definisi Kualitas Data (DQDL). Untuk informasi selengkapnya, lihat panduan AWS Glue pengembang.

Respons

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama aturan kualitas data.

- `Description` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Deskripsi dari ruleset.

- `Ruleset`— String UTF-8, panjangnya tidak kurang dari 1 atau lebih dari 65536 byte.

  Aturan Bahasa Definisi Kualitas Data (DQDL). Untuk informasi selengkapnya, lihat panduan AWS Glue pengembang.

Kesalahan

- `EntityNotFoundException`
- `AlreadyExistsException`
- `IdempotentParameterMismatchException`
- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`
- `ResourceNumberLimitExceededException`

API deteksi data sensitif

API deteksi data sensitif menjelaskan API yang digunakan untuk mendeteksi data sensitif di seluruh kolom dan baris data terstruktur Anda.

Jenis Data

- CustomEntityType struktur

CustomEntityType struktur

Objek yang mewakili pola kustom untuk mendeteksi data sensitif di seluruh kolom dan baris data terstruktur Anda.

Bidang

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama untuk pola kustom yang memungkinkannya diambil atau dihapus nanti. Nama ini harus unik per AWS akun.
- `RegexString` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  String ekspresi reguler yang digunakan untuk mendeteksi data sensitif dalam pola kustom.
- `ContextWords`Susunan string UTF-8, tidak kurang dari 1 atau lebih dari 20 string.

  Daftar kata-kata konteks. Jika tidak satu pun dari kata-kata konteks ini ditemukan di sekitar ekspresi reguler, data tidak akan terdeteksi sebagai data sensitif.

  Jika tidak ada kata konteks yang diteruskan, hanya ekspresi reguler yang diperiksa.

Operasi

- CreateCustomEntityType tindakan (Python: create_custom_entity_type)
- DeleteCustomEntityType tindakan (Python: delete_custom_entity_type)
- GetCustomEntityType tindakan (Python: get_custom_entity_type)

- BatchGetCustomEntityTypes tindakan (Python: batch_get_custom_entity_types)
- ListCustomEntityTypes tindakan (Python: list_custom_entity_types)

CreateCustomEntityType tindakan (Python: create_custom_entity_type)

Membuat pola kustom yang digunakan untuk mendeteksi data sensitif di seluruh kolom dan baris data terstruktur Anda.

Setiap pola kustom yang Anda buat menentukan ekspresi reguler dan daftar opsional kata-kata konteks. Jika tidak ada kata konteks yang diteruskan, hanya ekspresi reguler yang diperiksa.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama untuk pola kustom yang memungkinkannya diambil atau dihapus nanti. Nama ini harus unik per AWS akun.
- `RegexString` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  String ekspresi reguler yang digunakan untuk mendeteksi data sensitif dalam pola kustom.
- `ContextWords`Susunan string UTF-8, tidak kurang dari 1 atau lebih dari 20 string.

  Daftar kata-kata konteks. Jika tidak satu pun dari kata-kata konteks ini ditemukan di sekitar ekspresi reguler, data tidak akan terdeteksi sebagai data sensitif.

  Jika tidak ada kata konteks yang diteruskan, hanya ekspresi reguler yang diperiksa.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Daftar tanda yang diterapkan ke tipe entitas kustom.

Response

- Name — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pola kustom yang Anda buat.

Kesalahan

- AccessDeniedException
- AlreadyExistsException
- IdempotentParameterMismatchException
- InternalServiceException
- InvalidInputException
- OperationTimeoutException
- ResourceNumberLimitExceededException

DeleteCustomEntityType tindakan (Python: delete_custom_entity_type)

Menghapus pola kustom dengan menentukan namanya.

Permintaan

- Name — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pola kustom yang ingin Anda hapus.

Response

- Name — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pola kustom yang Anda hapus.

Kesalahan

- `EntityNotFoundException`
- `AccessDeniedException`
- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

GetCustomEntityType tindakan (Python: get_custom_entity_type)

Mengambil detail pola kustom dengan menentukan namanya.

Permintaan

- `Name` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pola kustom yang ingin Anda ambil.

Response

- `Name` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama pola kustom yang Anda ambil.
- `RegexString` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  String ekspresi reguler yang digunakan untuk mendeteksi data sensitif dalam pola kustom.
- `ContextWords`Susunan string UTF-8, tidak kurang dari 1 atau lebih dari 20 string.

  Daftar kata konteks jika ditentukan saat Anda membuat pola kustom. Jika tidak satu pun dari kata-kata konteks ini ditemukan di sekitar ekspresi reguler, data tidak akan terdeteksi sebagai data sensitif.

Kesalahan

- `EntityNotFoundException`
- `AccessDeniedException`
- `InternalServiceException`
- `InvalidInputException`
- `OperationTimeoutException`

BatchGetCustomEntityTypes tindakan (Python: batch_get_custom_entity_types)

Mengambil rincian untuk pola kustom yang ditentukan oleh daftar nama.

Permintaan

- `Names` — Diperlukan: Susunan string UTF-8, tidak kurang dari 1 atau lebih dari 50 string.

  Daftar nama pola kustom yang ingin Anda ambil.

Response

- `CustomEntityTypes` – Susunan objek CustomEntityType.

  Daftar `CustomEntityType` objek yang mewakili pola kustom yang telah dibuat.
- `CustomEntityTypesNotFound`Susunan string UTF-8, tidak kurang dari 1 atau lebih dari 50 string.

  Daftar nama pola khusus yang tidak ditemukan.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`

ListCustomEntityTypes tindakan (Python: list_custom_entity_types)

Daftar semua pola kustom yang telah dibuat.

Permintaan

- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman untuk mengimbangi hasil.
- `MaxResults` — Nomor (bilangan bulat), tidak kurang dari 1 atau lebih dari 1000.

  Jumlah hasil maksimum yang akan dikembalikan.
- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Daftar tanda pasangan nilai-kunci.

Response

- `CustomEntityTypes` – Susunan objek CustomEntityType.

  Daftar `CustomEntityType` objek yang mewakili pola kustom.
- `NextToken` – String UTF-8.

  Sebuah token pemberian nomor halaman, jika ada lebih banyak hasil yang tersedia.

Kesalahan

- `InvalidInputException`
- `OperationTimeoutException`
- `InternalServiceException`

Menandai API di AWS Glue

Jenis data

- Struktur tag

Struktur tag

`Tag`Objek mewakili label yang dapat Anda tetapkan ke AWS sumber daya. Setiap tag terdiri dari kunci dan nilai opsional, yang keduanya Anda tentukan.

Untuk informasi selengkapnya tentang tag, dan mengontrol akses ke sumber daya AWS Glue, lihat AWS Tag di AWS Glue dan Menentukan ARN AWS Glue Sumber Daya di panduan pengembang.

Bidang

- `key` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Kunci tanda. Kunci tersebut diperlukan saat Anda membuat tag pada sebuah objek. Kuncinya peka huruf besar/kecil, dan tidak boleh berisi awalan aws.
- `value` — String UTF-8, dengan panjang tidak lebih dari 256 byte.

  Nilai tanda. Nilai adalah opsional saat Anda membuat tag pada sebuah objek. Nilainya peka huruf besar/kecil, dan tidak boleh mengandung awalan aws.

Operasi

- TagResource tindakan (Python: tag_resource)
- UntagResource tindakan (Python: untag_resource)
- GetTags tindakan (Python: get_tags)

TagResource tindakan (Python: tag_resource)

Penambahan beberapa tag ke sebuah sumber daya. Tag adalah label yang dapat Anda tetapkan ke AWS sumber daya. Di AWS Glue, Anda hanya dapat menandai sumber daya tertentu. Untuk informasi lebih lanjut sumber daya yang dapat diberi tag, lihat Tag AWS di AWS Glue.

Selain izin penandaan untuk memanggil API terkait tag, Anda juga memerlukan `glue:GetConnection` izin untuk memanggil API penandaan pada koneksi, dan `glue:GetDatabase` izin untuk memanggil API penandaan pada database.

Permintaan

- `ResourceArn` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  ARN dari AWS Glue sumber daya untuk menambahkan tag. Untuk informasi selengkapnya tentang ARN AWS Glue sumber daya, lihat pola AWS Glue string ARN.
- `TagsToAdd` — Wajib: Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

  Tag untuk ditambahkan ke sumber daya ini.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `EntityNotFoundException`

UntagResource tindakan (Python: untag_resource)

Menghapus tag dari sebuah sumber daya.

Permintaan

- `ResourceArn` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari sumber daya yang akan dihapus tag-nya.
- `TagsToRemove` — Wajib: Susunan string UTF-8, tidak lebih dari 50 string.

  Tag yang akan dihapus dari sumber daya ini.

Respons

- Tidak ada parameter Respons.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `EntityNotFoundException`

GetTags tindakan (Python: get_tags)

Mengambil daftar tag yang dikaitkan dengan sumber daya tertentu.

Permintaan

- `ResourceArn` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 10240 byte, yang cocok dengan Custom string pattern #22.

  Amazon Resource Name (ARN) dari sumber daya yang akan diambil tag-nya.

Respons

- `Tags` — Sebuah rangkaian peta pasangan nilai kunci, tidak lebih dari 50 pasang.

  Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

Masing-masing kunci adalah sebuah string UTF-8, dengan panjang tidak lebih dari 256 byte.

Tag yang diminta.

Kesalahan

- `InvalidInputException`
- `InternalServiceException`
- `OperationTimeoutException`
- `EntityNotFoundException`

Tipe data umum

Tipe data umum menggambarkan berbagai tipe data umum di. AWS Glue

Struktur tag

`Tag`Objek mewakili label yang dapat Anda tetapkan ke AWS sumber daya. Setiap tag terdiri dari kunci dan nilai opsional, yang keduanya Anda tentukan.

Untuk informasi selengkapnya tentang tag, dan mengontrol akses ke sumber daya AWS Glue, lihat AWS Tag di AWS Glue dan Menentukan ARN AWS Glue Sumber Daya di panduan pengembang.

Bidang

- `key` — String UTF-8, dengan panjang tidak kurang dari 1 atau lebih dari 128 byte.

  Kunci tanda. Kunci tersebut diperlukan saat Anda membuat tag pada sebuah objek. Kuncinya peka huruf besar/kecil, dan tidak boleh berisi awalan aws.
- `value` — String UTF-8, dengan panjang tidak lebih dari 256 byte.

  Nilai tanda. Nilai adalah opsional saat Anda membuat tag pada sebuah objek. Nilainya peka huruf besar/kecil, dan tidak boleh mengandung awalan aws.

DecimalNumber struktur

Berisi nilai numerik dalam format desimal.

Bidang

- `UnscaledValue` — Wajib: Blob.

  Nilai numerik tidak diskalakan.

- `Scale` — Wajib: Nomor (bilangan bulat).

  Skala yang menentukan di mana titik desimal termasuk dalam nilai tidak diskalakan.

ErrorDetail struktur

Berisi detail tentang kesalahan.

Bidang

- `ErrorCode` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Kode yang dikaitkan dengan kesalahan ini.

- `ErrorMessage` — String deskripsi, dengan panjang tidak lebih dari 2048 byte, yang cocok dengan URI address multi-line string pattern.

  Sebuah pesan yang menjelaskan kesalahan.

PropertyPredicate struktur

Menentukan predikat properti.

Bidang

- `Key` — String nilai, dengan panjang tidak lebih dari 1024 byte.

  Kunci properti.

- `Value` — String nilai, dengan panjang tidak lebih dari 1024 byte.

  Nilai properti.

- `Comparator` – String UTF-8 (nilai yang valid: `EQUALS` | `GREATER_THAN` | `LESS_THAN` | `GREATER_THAN_EQUALS` | `LESS_THAN_EQUALS`).

Pembanding digunakan untuk membandingkan properti ini dengan properti yang lain.

ResourceUri struktur

URI untuk sumber daya fungsi.

Bidang

- `ResourceType` – String UTF-8 (nilai yang valid: `JAR` | `FILE` | `ARCHIVE`).

  Jenis sumber daya.

- `Uri` — Pengenal sumber daya seragam (uri), dengan panjang tidak kurang dari 1 atau lebih dari 1024 byte, yang cocok dengan URI address multi-line string pattern.

  URI untuk mengakses sumber daya.

ColumnStatistics struktur

Merepresentasikan statistik kolom-tingkat yang dihasilkan untuk tabel atau partisi.

Bidang

- `ColumnName` — Wajib: String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama kolom di mana statistik menjadi bagiannya.

- `ColumnType` — Wajib: Jenis nama, dengan panjang tidak lebih dari 20000 byte, yang cocok dengan Single-line string pattern.

  Jenis data kolom.

- `AnalyzedTime` — Wajib: Stempel waktu.

  Stempel waktu ketika kolom statistik dihasilkan.

- `StatisticsData` — Wajib: Sebuah objek ColumnStatisticsData.

  Sebuah objek `ColumnStatisticData` yang berisi nilai data statistik.

ColumnStatisticsError struktur

Merangkum objek `ColumnStatistics` yang gagal dan alasan kegagalannya.

Bidang

- `ColumnStatistics` — Sebuah objek ColumnStatistics.

  `ColumnStatistics` dari kolom.
- `Error` — Sebuah objek ErrorDetail.

  Pesan kesalahan dengan alasan kegagalan sebuah operasi.

ColumnError struktur

Merangkum nama kolom yang gagal dan alasan kegagalannya.

Bidang

- `ColumnName` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  Nama kolom yang gagal.
- `Error` — Sebuah objek ErrorDetail.

  Pesan kesalahan dengan alasan kegagalan sebuah operasi.

ColumnStatisticsData struktur

Berisi jenis data statistik kolom individu. Hanya satu objek data yang harus ditetapkan dan ditunjukkan oleh atribut `Type`.

Bidang

- `Type` – Wajib: String UTF-8 (nilai yang valid: `BOOLEAN` | `DATE` | `DECIMAL` | `DOUBLE` | `LONG` | `STRING` | `BINARY`).

  Jenis data statistik kolom.
- `BooleanColumnStatisticsData` — Sebuah objek BooleanColumnStatisticsData.

Data statistik kolom Boolean.

- `DateColumnStatisticsData` — Sebuah objek DateColumnStatisticsData.

  Data statistik kolom tanggal.

- `DecimalColumnStatisticsData` — Sebuah objek DecimalColumnStatisticsData.

  Data statistik kolom desimal. UnscaledValues di dalamnya adalah objek biner yang dikodekan Base64 yang menyimpan endian besar, representasi komplemen dua dari nilai desimal yang tidak diskalakan.

- `DoubleColumnStatisticsData` — Sebuah objek DoubleColumnStatisticsData.

  Data statistik kolom ganda.

- `LongColumnStatisticsData` — Sebuah objek LongColumnStatisticsData.

  Data statistik kolom panjang.

- `StringColumnStatisticsData` — Sebuah objek StringColumnStatisticsData.

  Data statistik kolom string.

- `BinaryColumnStatisticsData` — Sebuah objek BinaryColumnStatisticsData.

  Data statistik kolom biner.

BooleanColumnStatisticsData struktur

Menentukan statistik kolom yang didukung untuk kolom data Boolean.

Bidang

- `NumberOfTrues` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai yang BETUL dalam kolom.

- `NumberOfFalses` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai SALAH dalam kolom.

- `NumberOfNulls` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai nol dalam kolom.

DateColumnStatisticsData struktur

Menentukan statistik kolom yang didukung untuk kolom data stempel waktu.

Bidang

- `MinimumValue` — Stempel waktu.

  Nilai terendah dalam kolom.
- `MaximumValue` — Stempel waktu.

  Nilai tertinggi dalam kolom.
- `NumberOfNulls` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai nol dalam kolom.
- `NumberOfDistinctValues` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai yang berbeda dalam kolom.

DecimalColumnStatisticsData struktur

Menentukan statistik kolom yang didukung untuk kolom data nomor fixed-point.

Bidang

- `MinimumValue` — Sebuah objek DecimalNumber.

  Nilai terendah dalam kolom.
- `MaximumValue` — Sebuah objek DecimalNumber.

  Nilai tertinggi dalam kolom.
- `NumberOfNulls` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai nol dalam kolom.
- `NumberOfDistinctValues` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai yang berbeda dalam kolom.

DoubleColumnStatisticsData struktur

Menentukan statistik kolom yang didukung untuk kolom data angka floating-point.

Bidang

- `MinimumValue` — Nomor (ganda).

  Nilai terendah dalam kolom.
- `MaximumValue` — Nomor (ganda).

  Nilai tertinggi dalam kolom.
- `NumberOfNulls` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai nol dalam kolom.
- `NumberOfDistinctValues` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai yang berbeda dalam kolom.

LongColumnStatisticsData struktur

Menentukan statistik kolom yang didukung untuk kolom data integer.

Bidang

- `MinimumValue` — Nomor (panjang).

  Nilai terendah dalam kolom.
- `MaximumValue` — Nomor (panjang).

  Nilai tertinggi dalam kolom.
- `NumberOfNulls` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai nol dalam kolom.
- `NumberOfDistinctValues` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai yang berbeda dalam kolom.

StringColumnStatisticsData struktur

Menentukan statistik kolom yang didukung untuk nilai-nilai data deret karakter.

Bidang

- `MaximumLength` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Ukuran string terpanjang dalam kolom.
- `AverageLength` — Wajib: Nomor (dua kali lipat), tidak lebih dari Kosong.

  Panjang string rata-rata dalam kolom.
- `NumberOfNulls` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai nol dalam kolom.
- `NumberOfDistinctValues` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai yang berbeda dalam kolom.

BinaryColumnStatisticsData struktur

Menentukan statistik kolom yang didukung untuk nilai data deret bit.

Bidang

- `MaximumLength` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Ukuran deret bit terpanjang dalam kolom.
- `AverageLength` — Wajib: Nomor (dua kali lipat), tidak lebih dari Kosong.

  Rata-rata panjang deret bit dalam kolom.
- `NumberOfNulls` — Wajib: Nomor (panjang), tidak lebih dari Kosong.

  Jumlah nilai nol dalam kolom.

Pola string

API menggunakan ekspresi reguler berikut untuk menentukan konten apa yang valid untuk berbagai parameter string dan anggota:

- Pola string satu baris — "[\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF\t]*"
- Pola string multi-baris alamat URI — "[\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF\r\n\t]*"
- Sebuah pola string Logstash Grok — "[\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF\r\t]*"
- Pola string pengenal — "[A-Za-z_][A-Za-z0-9_]*"
- Pola string ARN AWS IAM — "arn:aws:iam::\d{12}:role/.*"
- Pola string versi — "^[a-zA-Z0-9-_]+$"
- Pola string grup log — "[\.\-_/#A-Za-z0-9]+"
- Pola string log-stream — "[^:*]*"
- Pola string kustom #10 — "[^\r\n]"
- Pola string kustom #11 — "^arn:aws(-(cn|us-gov|iso(-[bef])?))?:secretsmanager:.*$"
- Pola string kustom #12 — "^(https?)://[-a-zA-Z0-9+@#/%?=~_|!:,.;]*[-a-zA-Z0-9+@#/%=~_|]"
- Pola string kustom #13 — "\S+"
- Pola string kustom #14 — "^(https?):\/\/[^\s/$.?#].[^\s]*$"
- Pola string kustom #15 — "^subnet-[a-z0-9]+$"
- Pola string kustom #16 — "[\p{L}\p{N}\p{P}]*"
- Pola string kustom #17 — "[a-f0-9]{8}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{4}-[a-f0-9]{12}"
- Pola string kustom #18 — "[a-zA-Z0-9-_$#.]+"
- Pola string kustom #19 — "^\w+\.\w+\.\w+$"
- Pola string kustom #20 — "^\w+\.\w+$"
- Pola string kustom #21 — "^([2-3]|3[.]9)$"
- Pola string kustom #22 — "arn:(aws|aws-us-gov|aws-cn):glue:.*"
- Pola string kustom #23 — "(^arn:aws:iam::\w{12}:root)”
- Pola string kustom #24 — "^arn:aws(-(cn|us-gov|iso(-[bef])?))?:iam::[0-9]{12}:role/.+”

- Pola string kustom #25 — "arn:aws:kms:.*"
- Pola string kustom #26 — "arn:aws[^:]*:iam::[0-9]*:role/.+"
- Pola string kustom #27 — "[\.\-_A-Za-z0-9]+"
- Pola string kustom #28 — "^s3://([^/]+)/([^/]+/)*([^/]+)$"
- Pola string kustom #29 — ".*"
- Pola string kustom #30 — "^(Sun|Mon|Tue|Wed|Thu|Fri|Sat):([01]?[0-9]|2[0-3])$"
- Pola string kustom #31 — "[a-zA-Z0-9_.-]+"
- Pola string kustom #32 — ".*\S.*"
- Pola string kustom #33 — "[a-zA-Z0-9-=._/@]+"
- Pola string kustom #34 — "[1-9][0-9]*|[1-9][0-9]*-[1-9][0-9]*"
- Pola string kustom #35 — "[\s\S]*"
- Pola string kustom #36 — "([\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF \uDFFF]|[^\S\r\n"'= ;])*"
- Pola string kustom #37 — "[*A-Za-z0-9_-]*"
- Pola string kustom #38 — "([\u0020-\u007E\r\s\n])*"
- Pola string kustom #39 — "[A-Za-z0-9_-]*"
- Pola string kustom #40 — "([\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF \uDFFF]|[^\S\r\n"'])*"
- Pola string kustom #41 — "([\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF \uDFFF]|[^\S\r\n])*"
- Pola string kustom #42 — "([\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF \s])*"
- Pola string kustom #43 — "([\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF \uDFFF]|[^\r\n])*"

Pengecualian

Bagian ini menjelaskan AWS Glue pengecualian yang dapat Anda gunakan untuk menemukan sumber masalah dan memperbaikinya. Untuk informasi lebih lanjut tentang kode kesalahan HTTP dan string untuk pengecualian yang terkait dengan machine learning, lihat the section called “AWS Gluepengecualian pembelajaran mesin”.

AccessDeniedException struktur

Akses ke sumber daya ditolak.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

AlreadyExistsException struktur

Sumber daya yang akan dibuat atau ditambahkan sudah ada.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

ConcurrentModificationException struktur

Dua proses mencoba untuk mengubah sumber daya secara bersamaan.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

ConcurrentRunsExceededException struktur

Terlalu banyak pekerjaan yang sedang dijalankan secara bersamaan.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

CrawlerNotRunningException struktur

Crawler yang ditentukan tidak berjalan.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

CrawlerRunningException struktur

Operasi tidak dapat dilaksanakan karena crawler sudah berjalan.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

CrawlerStoppingException struktur

Crawler yang ditentukan sedang berhenti.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

EntityNotFoundException struktur

Entitas yang ditentukan tidak ada

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.
- `FromFederationSource` – Boolean.

Menunjukkan apakah pengecualian terkait dengan sumber federasi atau tidak.

FederationSourceException struktur

Sumber federasi gagal.

Bidang

- `FederationSourceErrorCode`— UTF-8 string (nilai valid: `AccessDeniedException` | | `EntityNotFoundException` | `InvalidCredentialsException` | `InvalidInputException` | `InvalidResponseException` | `OperationTimeoutException` | `OperationNotSupportedException` `InternalServiceException` `PartialFailureException` |`ThrottlingException`).

  Kode kesalahan masalah.
- `Message` – String UTF-8.

  Pesan yang menjelaskan masalah.

FederationSourceRetryableException struktur

Sumber federasi gagal, tetapi operasi dapat dicoba lagi.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

GlueEncryptionException struktur

Operasi enkripsi gagal.

Bidang

- `Message` – String UTF-8.

  Pesan yang menjelaskan masalah.

IdempotentParameterMismatchException struktur

Pengenal unik yang sama dikaitkan dengan dua catatan yang berbeda.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

IllegalWorkflowStateException struktur

Alur kerja dalam status tidak valid untuk melakukan operasi yang diminta.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

InternalServiceException struktur

Terjadi kesalahan layanan internal.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

InvalidExecutionEngineException struktur

Mesin eksekusi yang tidak dikenal atau tidak valid telah ditentukan.

Bidang

- `message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

InvalidInputException struktur

Masukan yang diberikan tidak valid.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.
- `FromFederationSource` – Boolean.

  Menunjukkan apakah pengecualian terkait dengan sumber federasi atau tidak.

InvalidStateException struktur

Kesalahan yang menunjukkan data Anda dalam keadaan tidak valid.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

InvalidTaskStatusTransitionException struktur

Transisi yang tepat dari satu tugas ke tugas berikutnya gagal.

Bidang

- `message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

JobDefinitionErrorException struktur

Penentuan tugas tidak valid.

Bidang

- `message` – String UTF-8.

Sebuah pesan yang menjelaskan masalah.

JobRunInTerminalStateException struktur

Status terminal dari eksekusi tugas memberikan sinyal kegagalan.

Bidang

- `message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

JobRunInvalidStateTransitionException struktur

Eksekusi tugas mengalami transisi yang tidak valid dari status sumber ke status target.

Bidang

- `jobRunId` — String UTF-8, sepanjang tidak kurang dari 1 atau lebih dari 255 byte, yang cocok dengan Single-line string pattern.

  ID eksekusi tugas yang dimaksud.
- `message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.
- `sourceState`— UTF-8 string (nilai valid: `STARTING` | | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT` `ERROR` `WAITING` |`EXPIRED`).

  Status sumber.
- `targetState`— UTF-8 string (nilai valid: `STARTING` | | `RUNNING` | `STOPPING` | `STOPPED` | `SUCCEEDED` | `FAILED` | `TIMEOUT` `ERROR` `WAITING` |`EXPIRED`).

  Status target.

JobRunNotInTerminalStateException struktur

Eksekusi tugas tidak dalam status terminal.

Bidang

- `message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

LateRunnerException struktur

Eksekutor tugas terlambat.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

NoScheduleException struktur

Tidak ada jadwal yang berlaku.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

OperationTimeoutException struktur

Waktu operasi habis.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

ResourceNotReadyException struktur

Sumber daya belum siap untuk transaksi.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

ResourceNumberLimitExceededException struktur

Batas numerik sumber daya terlampaui.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

SchedulerNotRunningException struktur

Penjadwal yang ditentukan tidak berjalan.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

SchedulerRunningException struktur

Penjadwal yang ditentukan sudah berjalan.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

SchedulerTransitioningException struktur

Penjadwal yang ditentukan sedang melakukan transisi.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

UnrecognizedRunnerException struktur

Eksekutor tugas tidak diakui.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

ValidationException struktur

Nilai tidak dapat divalidasi.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

VersionMismatchException struktur

Ada pertentangan versi.

Bidang

- `Message` – String UTF-8.

  Sebuah pesan yang menjelaskan masalah.

AWS Glue Contoh kode API menggunakan AWS SDK

Contoh kode berikut menunjukkan cara menggunakan AWS Glue kit pengembangan AWS perangkat lunak (SDK).

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Meskipun tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks pada skenario terkait dan contoh lintas layanan.

Skenario adalah contoh kode yang menunjukkan cara menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan yang sama.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Memulai

Halo AWS Glue

Contoh kode berikut menunjukkan cara untuk mulai menggunakan AWS Glue.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
namespace GlueActions;

public class HelloGlue
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for AWS Glue.
```

```
        using var host = Host.CreateDefaultBuilder(args)
            .ConfigureLogging(logging =>
                logging.AddFilter("System", LogLevel.Debug)
                    .AddFilter<DebugLoggerProvider>("Microsoft",
 LogLevel.Information)
                    .AddFilter<ConsoleLoggerProvider>("Microsoft",
 LogLevel.Trace))
            .ConfigureServices((_, services) =>
                services.AddAWSService<IAmazonGlue>()
                .AddTransient<GlueWrapper>()
            )
            .Build();

        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<HelloGlue>();
        var glueClient = host.Services.GetRequiredService<IAmazonGlue>();

        var request = new ListJobsRequest();

        var jobNames = new List<string>();

        do
        {
            var response = await glueClient.ListJobsAsync(request);
            jobNames.AddRange(response.JobNames);
            request.NextToken = response.NextToken;
        }
        while (request.NextToken is not null);

        Console.Clear();
        Console.WriteLine("Hello, Glue. Let's list your existing Glue Jobs:");
        if (jobNames.Count == 0)
        {
            Console.WriteLine("You don't have any AWS Glue jobs.");
        }
        else
        {
            jobNames.ForEach(Console.WriteLine);
        }
    }
}
```

- Untuk detail API, lihat ListJobsdi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Kode untuk file CMake MakeLists C.txt.

```
# Set the minimum required version of CMake for this project.
cmake_minimum_required(VERSION 3.13)

# Set the AWS service components used by this project.
set(SERVICE_COMPONENTS glue)

# Set this project's name.
project("hello_glue")

# Set the C++ standard to use to build this target.
# At least C++ 11 is required for the AWS SDK for C++.
set(CMAKE_CXX_STANDARD 11)

# Use the MSVC variable to determine if this is a Windows build.
set(WINDOWS_BUILD ${MSVC})

if (WINDOWS_BUILD) # Set the location where CMake can find the installed
 libraries for the AWS SDK.
    string(REPLACE ";" "/aws-cpp-sdk-all;" SYSTEM_MODULE_PATH
 "${CMAKE_SYSTEM_PREFIX_PATH}/aws-cpp-sdk-all")
    list(APPEND CMAKE_PREFIX_PATH ${SYSTEM_MODULE_PATH})
endif ()

# Find the AWS SDK for C++ package.
find_package(AWSSDK REQUIRED COMPONENTS ${SERVICE_COMPONENTS})

if (WINDOWS_BUILD AND AWSSDK_INSTALL_AS_SHARED_LIBS)
```

```
    # Copy relevant AWS SDK for C++ libraries into the current binary directory
 for running and debugging.

    # set(BIN_SUB_DIR "/Debug") # if you are building from the command line you
 may need to uncomment this
                                # and set the proper subdirectory to the
 executables' location.

    AWSSDK_CPY_DYN_LIBS(SERVICE_COMPONENTS ""
 ${CMAKE_CURRENT_BINARY_DIR}${BIN_SUB_DIR})
endif ()

add_executable(${PROJECT_NAME}
        hello_glue.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Kode untuk file sumber hello_glue.cpp.

```
#include <aws/core/Aws.h>
#include <aws/glue/GlueClient.h>
#include <aws/glue/model/ListJobsRequest.h>
#include <iostream>

/*
 *  A "Hello Glue" starter application which initializes an AWS Glue client and
 lists the
 *  AWS Glue job definitions.
 *
 *  main function
 *
 *  Usage: 'hello_glue'
 *
 */

int main(int argc, char **argv) {
    Aws::SDKOptions options;
    // Optionally change the log level for debugging.
//   options.loggingOptions.logLevel = Utils::Logging::LogLevel::Debug;
    Aws::InitAPI(options); // Should only be called once.
    int result = 0;
```

```
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::Glue::GlueClient glueClient(clientConfig);

        std::vector<Aws::String> jobs;

        Aws::String nextToken;  // Used for pagination.
        do {
            Aws::Glue::Model::ListJobsRequest listJobsRequest;
            if (!nextToken.empty()) {
                listJobsRequest.SetNextToken(nextToken);
            }

            Aws::Glue::Model::ListJobsOutcome listRunsOutcome =
 glueClient.ListJobs(
                    listJobsRequest);

            if (listRunsOutcome.IsSuccess()) {
                const std::vector<Aws::String> &jobNames =
 listRunsOutcome.GetResult().GetJobNames();
                jobs.insert(jobs.end(), jobNames.begin(), jobNames.end());

                nextToken = listRunsOutcome.GetResult().GetNextToken();
            } else {
                std::cerr << "Error listing jobs. "
                          << listRunsOutcome.GetError().GetMessage()
                          << std::endl;
                result = 1;
                break;
            }
        } while (!nextToken.empty());

        std::cout << "Your account has " << jobs.size() << " jobs."
                  << std::endl;
        for (size_t i = 0; i < jobs.size(); ++i) {
            std::cout << "   " << i + 1 << ". " << jobs[i] << std::endl;
        }
    }
    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Untuk detail API, lihat ListJobsdi Referensi AWS SDK for C++ API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
package com.example.glue;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glue.GlueClient;
import software.amazon.awssdk.services.glue.model.ListJobsRequest;
import software.amazon.awssdk.services.glue.model.ListJobsResponse;
import java.util.List;

public class HelloGlue {
    public static void main(String[] args) {
        GlueClient glueClient = GlueClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listJobs(glueClient);
    }

    public static void listJobs(GlueClient glueClient) {
        ListJobsRequest request = ListJobsRequest.builder()
                .maxResults(10)
                .build();
        ListJobsResponse response = glueClient.listJobs(request);
        List<String> jobList = response.jobNames();
        jobList.forEach(job -> {
            System.out.println("Job Name: " + job);
        });
    }
}
```

- Untuk detail API, lihat ListJobsdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import { ListJobsCommand, GlueClient } from "@aws-sdk/client-glue";

const client = new GlueClient({});

export const main = async () => {
  const command = new ListJobsCommand({});

  const { JobNames } = await client.send(command);
  const formattedJobNames = JobNames.join("\n");
  console.log("Job names: ");
  console.log(formattedJobNames);
  return JobNames;
};
```

- Untuk detail API, lihat ListJobsdi Referensi AWS SDK for JavaScript API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        let mut list_jobs = glue.list_jobs().into_paginator().send();
        while let Some(list_jobs_output) = list_jobs.next().await {
            match list_jobs_output {
                Ok(list_jobs) => {
                    let names = list_jobs.job_names();
                    info!(?names, "Found these jobs")
                }
                Err(err) => return Err(GlueMvpError::from_glue_sdk(err)),
            }
        }
```

- Untuk detail API, lihat ListJobsreferensi AWS SDK for Rust API.

Contoh kode

- Tindakan untuk AWS Glue menggunakan AWS SDK
  - Gunakan CreateCrawler dengan AWS SDK atau CLI
  - Gunakan CreateJob dengan AWS SDK atau CLI
  - Gunakan DeleteCrawler dengan AWS SDK atau CLI
  - Gunakan DeleteDatabase dengan AWS SDK atau CLI
  - Gunakan DeleteJob dengan AWS SDK atau CLI
  - Gunakan DeleteTable dengan AWS SDK atau CLI
  - Gunakan GetCrawler dengan AWS SDK atau CLI
  - Gunakan GetDatabase dengan AWS SDK atau CLI
  - Gunakan GetDatabases dengan AWS SDK atau CLI
  - Gunakan GetJob dengan AWS SDK atau CLI
  - Gunakan GetJobRun dengan AWS SDK atau CLI
  - Gunakan GetJobRuns dengan AWS SDK atau CLI
  - Gunakan GetTables dengan AWS SDK atau CLI
  - Gunakan ListJobs dengan AWS SDK atau CLI
  - Gunakan StartCrawler dengan AWS SDK atau CLI
  - Gunakan StartJobRun dengan AWS SDK atau CLI
- Skenario untuk AWS Glue menggunakan AWS SDK
  - Mulai menjalankan AWS Glue crawler dan lowongan kerja menggunakan SDK AWS

Tindakan untuk AWS Glue menggunakan AWS SDK

Contoh kode berikut menunjukkan cara melakukan AWS Glue tindakan individual dengan AWS SDK. Kutipan ini memanggil AWS Glue API dan merupakan kutipan kode dari program yang lebih besar yang harus dijalankan dalam konteks. Setiap contoh menyertakan tautan ke GitHub, di mana Anda dapat menemukan instruksi untuk mengatur dan menjalankan kode.

Contoh berikut hanya mencakup tindakan yang paling umum digunakan. Untuk daftar lengkapnya, lihat Referensi AWS Glue API.

Contoh

- Gunakan CreateCrawler dengan AWS SDK atau CLI
- Gunakan CreateJob dengan AWS SDK atau CLI
- Gunakan DeleteCrawler dengan AWS SDK atau CLI
- Gunakan DeleteDatabase dengan AWS SDK atau CLI
- Gunakan DeleteJob dengan AWS SDK atau CLI
- Gunakan DeleteTable dengan AWS SDK atau CLI
- Gunakan GetCrawler dengan AWS SDK atau CLI
- Gunakan GetDatabase dengan AWS SDK atau CLI
- Gunakan GetDatabases dengan AWS SDK atau CLI
- Gunakan GetJob dengan AWS SDK atau CLI
- Gunakan GetJobRun dengan AWS SDK atau CLI
- Gunakan GetJobRuns dengan AWS SDK atau CLI
- Gunakan GetTables dengan AWS SDK atau CLI
- Gunakan ListJobs dengan AWS SDK atau CLI
- Gunakan StartCrawler dengan AWS SDK atau CLI
- Gunakan StartJobRun dengan AWS SDK atau CLI

Gunakan **`CreateCrawler`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateCrawler`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
   /// <summary>
   /// Create an AWS Glue crawler.
   /// </summary>
   /// <param name="crawlerName">The name for the crawler.</param>
   /// <param name="crawlerDescription">A description of the crawler.</param>
   /// <param name="role">The AWS Identity and Access Management (IAM) role to
   /// be assumed by the crawler.</param>
   /// <param name="schedule">The schedule on which the crawler will be
executed.</param>
   /// <param name="s3Path">The path to the Amazon Simple Storage Service
(Amazon S3)
   /// bucket where the Python script has been stored.</param>
   /// <param name="dbName">The name to use for the database that will be
   /// created by the crawler.</param>
   /// <returns>A Boolean value indicating the success of the action.</returns>
   public async Task<bool> CreateCrawlerAsync(
       string crawlerName,
       string crawlerDescription,
       string role,
       string schedule,
       string s3Path,
       string dbName)
   {
       var s3Target = new S3Target
       {
           Path = s3Path,
       };

       var targetList = new List<S3Target>
       {
```

```
            s3Target,
        };

        var targets = new CrawlerTargets
        {
            S3Targets = targetList,
        };

        var crawlerRequest = new CreateCrawlerRequest
        {
            DatabaseName = dbName,
            Name = crawlerName,
            Description = crawlerDescription,
            Targets = targets,
            Role = role,
            Schedule = schedule,
        };

        var response = await _amazonGlue.CreateCrawlerAsync(crawlerRequest);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat CreateCrawlerdi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);
```

```
        Aws::Glue::Model::S3Target s3Target;
        s3Target.SetPath("s3://crawler-public-us-east-1/flight/2016/csv");
        Aws::Glue::Model::CrawlerTargets crawlerTargets;
        crawlerTargets.AddS3Targets(s3Target);

        Aws::Glue::Model::CreateCrawlerRequest request;
        request.SetTargets(crawlerTargets);
        request.SetName(CRAWLER_NAME);
        request.SetDatabaseName(CRAWLER_DATABASE_NAME);
        request.SetTablePrefix(CRAWLER_DATABASE_PREFIX);
        request.SetRole(roleArn);

        Aws::Glue::Model::CreateCrawlerOutcome outcome =
client.CreateCrawler(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully created the crawler." << std::endl;
        }
        else {
            std::cerr << "Error creating a crawler. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets("", CRAWLER_DATABASE_NAME, "", bucketName,
clientConfig);
            return false;
        }
```

- Untuk detail API, lihat CreateCrawlerdi Referensi AWS SDK for C++ API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
```

```
import software.amazon.awssdk.services.glue.GlueClient;
import software.amazon.awssdk.services.glue.model.CreateCrawlerRequest;
import software.amazon.awssdk.services.glue.model.CrawlerTargets;
import software.amazon.awssdk.services.glue.model.GlueException;
import software.amazon.awssdk.services.glue.model.S3Target;
import java.util.ArrayList;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class CreateCrawler {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <IAM> <s3Path> <cron> <dbName> <crawlerName>

                Where:
                    IAM - The ARN of the IAM role that has AWS Glue and S3
 permissions.\s
                    s3Path - The Amazon Simple Storage Service (Amazon S3) target
 that contains data (for example, CSV data).
                    cron - A cron expression used to specify the schedule  (i.e.,
 cron(15 12 * * ? *).
                    dbName - The database name.\s
                    crawlerName - The name of the crawler.\s
                """;

        if (args.length != 5) {
            System.out.println(usage);
            System.exit(1);
        }

        String iam = args[0];
        String s3Path = args[1];
        String cron = args[2];
        String dbName = args[3];
```

```
        String crawlerName = args[4];
        Region region = Region.US_EAST_1;
        GlueClient glueClient = GlueClient.builder()
                .region(region)
                .build();

        createGlueCrawler(glueClient, iam, s3Path, cron, dbName, crawlerName);
        glueClient.close();
    }

    public static void createGlueCrawler(GlueClient glueClient,
            String iam,
            String s3Path,
            String cron,
            String dbName,
            String crawlerName) {

        try {
            S3Target s3Target = S3Target.builder()
                    .path(s3Path)
                    .build();

            // Add the S3Target to a list.
            List<S3Target> targetList = new ArrayList<>();
            targetList.add(s3Target);

            CrawlerTargets targets = CrawlerTargets.builder()
                    .s3Targets(targetList)
                    .build();

            CreateCrawlerRequest crawlerRequest = CreateCrawlerRequest.builder()
                    .databaseName(dbName)
                    .name(crawlerName)
                    .description("Created by the AWS Glue Java API")
                    .targets(targets)
                    .role(iam)
                    .schedule(cron)
                    .build();

            glueClient.createCrawler(crawlerRequest);
            System.out.println(crawlerName + " was successfully created");

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat CreateCrawlerdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const createCrawler = (name, role, dbName, tablePrefix, s3TargetPath) => {
  const client = new GlueClient({});

  const command = new CreateCrawlerCommand({
    Name: name,
    Role: role,
    DatabaseName: dbName,
    TablePrefix: tablePrefix,
    Targets: {
      S3Targets: [{ Path: s3TargetPath }],
    },
  });

  return client.send(command);
};
```

- Untuk detail API, lihat CreateCrawlerdi Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun createGlueCrawler(
    iam: String?,
    s3Path: String?,
    cron: String?,
    dbName: String?,
    crawlerName: String,
) {
    val s3Target =
        S3Target {
            path = s3Path
        }

    // Add the S3Target to a list.
    val targetList = mutableListOf<S3Target>()
    targetList.add(s3Target)

    val targetOb =
        CrawlerTargets {
            s3Targets = targetList
        }

    val request =
        CreateCrawlerRequest {
            databaseName = dbName
            name = crawlerName
            description = "Created by the AWS Glue Kotlin API"
            targets = targetOb
            role = iam
            schedule = cron
        }

    GlueClient { region = "us-west-2" }.use { glueClient ->
```

```
        glueClient.createCrawler(request)
        println("$crawlerName was successfully created")
    }
}
```

- Untuk detail API, lihat CreateCrawlerdi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        $crawlerName = "example-crawler-test-" . $uniqid;

        $role = $iamService->getRole("AWSGlueServiceRole-DocExample");

        $path = 's3://crawler-public-us-east-1/flight/2016/csv';
        $glueService->createCrawler($crawlerName, $role['Role']['Arn'],
$databaseName, $path);

   public function createCrawler($crawlerName, $role, $databaseName, $path):
Result
   {
        return $this->customWaiter(function () use ($crawlerName, $role,
$databaseName, $path) {
            return $this->glueClient->createCrawler([
                'Name' => $crawlerName,
                'Role' => $role,
                'DatabaseName' => $databaseName,
                'Targets' => [
                    'S3Targets' =>
                        [[
                            'Path' => $path,
                        ]]
                ],
            ]);
```

```
        });
    }
```

- Untuk detail API, lihat CreateCrawlerdi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def create_crawler(self, name, role_arn, db_name, db_prefix, s3_target):
        """
        Creates a crawler that can crawl the specified target and populate a
        database in your AWS Glue Data Catalog with metadata that describes the
 data
        in the target.

        :param name: The name of the crawler.
        :param role_arn: The Amazon Resource Name (ARN) of an AWS Identity and
 Access
                         Management (IAM) role that grants permission to let AWS
 Glue
                         access the resources it needs.
        :param db_name: The name to give the database that is created by the
 crawler.
```

```
        :param db_prefix: The prefix to give any database tables that are created
 by
                          the crawler.
        :param s3_target: The URL to an S3 bucket that contains data that is
                          the target of the crawler.
        """
        try:
            self.glue_client.create_crawler(
                Name=name,
                Role=role_arn,
                DatabaseName=db_name,
                TablePrefix=db_prefix,
                Targets={"S3Targets": [{"Path": s3_target}]},
            )
        except ClientError as err:
            logger.error(
                "Couldn't create crawler. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Untuk detail API, lihat CreateCrawlerdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
```

```
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Creates a new crawler with the specified configuration.
  #
  # @param name [String] The name of the crawler.
  # @param role_arn [String] The ARN of the IAM role to be used by the crawler.
  # @param db_name [String] The name of the database where the crawler stores its
 metadata.
  # @param db_prefix [String] The prefix to be added to the names of tables that
 the crawler creates.
  # @param s3_target [String] The S3 path that the crawler will crawl.
  # @return [void]
  def create_crawler(name, role_arn, db_name, db_prefix, s3_target)
    @glue_client.create_crawler(
      name: name,
      role: role_arn,
      database_name: db_name,
      targets: {
        s3_targets: [
          {
            path: s3_target
          }
        ]
      }
    )
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not create crawler: \n#{e.message}")
    raise
  end
```

- Untuk detail API, lihat CreateCrawlerdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        let create_crawler = glue
            .create_crawler()
            .name(self.crawler())
            .database_name(self.database())
            .role(self.iam_role.expose_secret())
            .targets(
                CrawlerTargets::builder()
                    .s3_targets(S3Target::builder().path(CRAWLER_TARGET).build())
                    .build(),
            )
            .send()
            .await;

        match create_crawler {
            Err(err) => {
                let glue_err: aws_sdk_glue::Error = err.into();
                match glue_err {
                    aws_sdk_glue::Error::AlreadyExistsException(_) => {
                        info!("Using existing crawler");
                        Ok(())
                    }
                    _ => Err(GlueMvpError::GlueSdk(glue_err)),
                }
            }
            Ok(_) => Ok(()),
        }?;
```

- Untuk detail API, lihat CreateCrawlerreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`CreateJob`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`CreateJob`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Create an AWS Glue job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <param name="roleName">The name of the IAM role to be assumed by
    /// the job.</param>
    /// <param name="description">A description of the job.</param>
    /// <param name="scriptUrl">The URL to the script.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> CreateJobAsync(string dbName, string tableName,
string bucketUrl, string jobName, string roleName, string description, string
scriptUrl)
    {
        var command = new JobCommand
        {
            PythonVersion = "3",
            Name = "glueetl",
            ScriptLocation = scriptUrl,
```

```
        };

        var arguments = new Dictionary<string, string>
        {
            { "--input_database", dbName },
            { "--input_table", tableName },
            { "--output_bucket_url", bucketUrl }
        };

        var request = new CreateJobRequest
        {
            Command = command,
            DefaultArguments = arguments,
            Description = description,
            GlueVersion = "3.0",
            Name = jobName,
            NumberOfWorkers = 10,
            Role = roleName,
            WorkerType = "G.1X"
        };

        var response = await _amazonGlue.CreateJobAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat CreateJobdi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
```

```
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::CreateJobRequest request;
        request.SetName(JOB_NAME);
        request.SetRole(roleArn);
        request.SetGlueVersion(GLUE_VERSION);

        Aws::Glue::Model::JobCommand command;
        command.SetName(JOB_COMMAND_NAME);
        command.SetPythonVersion(JOB_PYTHON_VERSION);
        command.SetScriptLocation(
                Aws::String("s3://") + bucketName + "/" + PYTHON_SCRIPT);
        request.SetCommand(command);

        Aws::Glue::Model::CreateJobOutcome outcome = client.CreateJob(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully created the job." << std::endl;
        }
        else {
            std::cerr << "Error creating the job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
```

- Untuk detail API, lihat CreateJobdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Untuk membuat pekerjaan untuk mengubah data

`create-job`Contoh berikut membuat pekerjaan streaming yang menjalankan skrip yang disimpan di S3.

```
aws glue create-job \
```

```
    --name my-testing-job \
    --role AWSGlueServiceRoleDefault \
    --command '{ \
        "Name": "gluestreaming", \
        "ScriptLocation": "s3://DOC-EXAMPLE-BUCKET/folder/" \
    }' \
    --region us-east-1 \
    --output json \
    --default-arguments '{ \
        "--job-language":"scala", \
        "--class":"GlueApp" \
    }' \
    --profile my-profile \
    --endpoint https://glue.us-east-1.amazonaws.com
```

Isi dari `test_script.scala`:

```
import com.amazonaws.services.glue.ChoiceOption
import com.amazonaws.services.glue.GlueContext
import com.amazonaws.services.glue.MappingSpec
import com.amazonaws.services.glue.ResolveSpec
import com.amazonaws.services.glue.errors.CallSite
import com.amazonaws.services.glue.util.GlueArgParser
import com.amazonaws.services.glue.util.Job
import com.amazonaws.services.glue.util.JsonOptions
import org.apache.spark.SparkContext
import scala.collection.JavaConverters._

object GlueApp {
    def main(sysArgs: Array[String]) {
        val spark: SparkContext = new SparkContext()
        val glueContext: GlueContext = new GlueContext(spark)
        // @params: [JOB_NAME]
        val args = GlueArgParser.getResolvedOptions(sysArgs,
 Seq("JOB_NAME").toArray)
        Job.init(args("JOB_NAME"), glueContext, args.asJava)
        // @type: DataSource
        // @args: [database = "tempdb", table_name = "s3-source",
 transformation_ctx = "datasource0"]
        // @return: datasource0
        // @inputs: []
```

```
        val datasource0 = glueContext.getCatalogSource(database = "tempdb",
 tableName = "s3-source", redshiftTmpDir = "", transformationContext =
 "datasource0").getDynamicFrame()
        // @type: ApplyMapping
        // @args: [mapping = [("sensorid", "int", "sensorid", "int"),
 ("currenttemperature", "int", "currenttemperature", "int"), ("status", "string",
 "status", "string")], transformation_ctx = "applymapping1"]
        // @return: applymapping1
        // @inputs: [frame = datasource0]
        val applymapping1 = datasource0.applyMapping(mappings = Seq(("sensorid",
 "int", "sensorid", "int"), ("currenttemperature", "int", "currenttemperature",
 "int"), ("status", "string", "status", "string")), caseSensitive = false,
 transformationContext = "applymapping1")
        // @type: SelectFields
        // @args: [paths = ["sensorid", "currenttemperature", "status"],
 transformation_ctx = "selectfields2"]
        // @return: selectfields2
        // @inputs: [frame = applymapping1]
        val selectfields2 = applymapping1.selectFields(paths = Seq("sensorid",
 "currenttemperature", "status"), transformationContext = "selectfields2")
        // @type: ResolveChoice
        // @args: [choice = "MATCH_CATALOG", database = "tempdb", table_name =
 "my-s3-sink", transformation_ctx = "resolvechoice3"]
        // @return: resolvechoice3
        // @inputs: [frame = selectfields2]
        val resolvechoice3 = selectfields2.resolveChoice(choiceOption =
 Some(ChoiceOption("MATCH_CATALOG")), database = Some("tempdb"), tableName =
 Some("my-s3-sink"), transformationContext = "resolvechoice3")
        // @type: DataSink
        // @args: [database = "tempdb", table_name = "my-s3-sink",
 transformation_ctx = "datasink4"]
        // @return: datasink4
        // @inputs: [frame = resolvechoice3]
        val datasink4 = glueContext.getCatalogSink(database = "tempdb",
 tableName = "my-s3-sink", redshiftTmpDir = "", transformationContext =
 "datasink4").writeDynamicFrame(resolvechoice3)
        Job.commit()
    }
}
```

Output:

```
{
```

```
    "Name": "my-testing-job"
}
```

Untuk informasi selengkapnya, lihat Menulis Pekerjaan di AWS Glue di Panduan Pengembang AWS Glue.

- Untuk detail API, lihat CreateJobdi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const createJob = (name, role, scriptBucketName, scriptKey) => {
  const client = new GlueClient({});

  const command = new CreateJobCommand({
    Name: name,
    Role: role,
    Command: {
      Name: "glueetl",
      PythonVersion: "3",
      ScriptLocation: `s3://${scriptBucketName}/${scriptKey}`,
    },
    GlueVersion: "3.0",
  });

  return client.send(command);
};
```

- Untuk detail API, lihat CreateJobdi Referensi AWS SDK for JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        $role = $iamService->getRole("AWSGlueServiceRole-DocExample");

        $jobName = 'test-job-' . $uniqid;

        $scriptLocation = "s3://$bucketName/run_job.py";
        $job = $glueService->createJob($jobName, $role['Role']['Arn'],
$scriptLocation);

    public function createJob($jobName, $role, $scriptLocation, $pythonVersion =
'3', $glueVersion = '3.0'): Result
    {
        return $this->glueClient->createJob([
            'Name' => $jobName,
            'Role' => $role,
            'Command' => [
                'Name' => 'glueetl',
                'ScriptLocation' => $scriptLocation,
                'PythonVersion' => $pythonVersion,
            ],
            'GlueVersion' => $glueVersion,
        ]);
    }
```

- Untuk detail API, lihat CreateJobdi Referensi AWS SDK for PHP API.

PowerShell

Alat untuk PowerShell

Contoh 1: Contoh ini menciptakan pekerjaan baru di AWS Glue. Nilai nama perintah selalu**`glueetl`**. AWS Glue mendukung menjalankan skrip pekerjaan yang ditulis dengan Python atau Scala. Dalam contoh ini, skrip pekerjaan (MyTestGlueJob.py) ditulis dengan Python. Parameter Python ditentukan dalam **`$DefArgs`** variabel, dan kemudian diteruskan ke PowerShell perintah dalam **`DefaultArguments`** parameter, yang menerima hashtable. Parameter dalam **`$JobParams`** variabel berasal dari CreateJob API, didokumentasikan dalam topik Jobs (https://docs.aws.amazon.com/glue/latest/dg/ aws-glue-api-jobs -job.html) dari referensi AWS Glue API.

```
$Command = New-Object Amazon.Glue.Model.JobCommand
$Command.Name = 'glueetl'
$Command.ScriptLocation = 's3://aws-glue-scripts-000000000000-us-west-2/admin/
MyTestGlueJob.py'
$Command

$Source = "source_test_table"
$Target = "target_test_table"
$Connections = $Source, $Target

$DefArgs = @{
     '--TempDir' = 's3://aws-glue-temporary-000000000000-us-west-2/admin'
     '--job-bookmark-option' = 'job-bookmark-disable'
     '--job-language' = 'python'
     }
$DefArgs

$ExecutionProp = New-Object Amazon.Glue.Model.ExecutionProperty
$ExecutionProp.MaxConcurrentRuns = 1
$ExecutionProp

$JobParams = @{
    "AllocatedCapacity"    = "5"
    "Command"              = $Command
    "Connections_Connection" = $Connections
    "DefaultArguments"  = $DefArgs
    "Description"       = "This is a test"
    "ExecutionProperty" = $ExecutionProp
    "MaxRetries"        = "1"
```

```
    "Name"               = "MyOregonTestGlueJob"
    "Role"               = "Amazon-GlueServiceRoleForSSM"
    "Timeout"            = "20"
     }

New-GlueJob @JobParams
```

- Untuk detail API, lihat CreateJobdi Referensi AWS Tools for PowerShell Cmdlet.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def create_job(self, name, description, role_arn, script_location):
        """
        Creates a job definition for an extract, transform, and load (ETL) job
 that can
        be run by AWS Glue.

        :param name: The name of the job definition.
        :param description: The description of the job definition.
        :param role_arn: The ARN of an IAM role that grants AWS Glue the
 permissions
                         it requires to run the job.
        :param script_location: The Amazon S3 URL of a Python ETL script that is
 run as
```

```
                                part of the job. The script defines how the data
is
                                transformed.
        """
        try:
            self.glue_client.create_job(
                Name=name,
                Description=description,
                Role=role_arn,
                Command={
                    "Name": "glueetl",
                    "ScriptLocation": script_location,
                    "PythonVersion": "3",
                },
                GlueVersion="3.0",
            )
        except ClientError as err:
            logger.error(
                "Couldn't create job %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Untuk detail API, lihat CreateJobdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
```

```
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Creates a new job with the specified configuration.
  #
  # @param name [String] The name of the job.
  # @param description [String] The description of the job.
  # @param role_arn [String] The ARN of the IAM role to be used by the job.
  # @param script_location [String] The location of the ETL script for the job.
  # @return [void]
  def create_job(name, description, role_arn, script_location)
    @glue_client.create_job(
      name: name,
      description: description,
      role: role_arn,
      command: {
        name: "glueetl",
        script_location: script_location,
        python_version: "3"
      },
      glue_version: "3.0"
    )
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not create job #{name}: \n#{e.message}")
    raise
  end
```

- Untuk detail API, lihat CreateJobdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        let create_job = glue
            .create_job()
            .name(self.job())
            .role(self.iam_role.expose_secret())
            .command(
                JobCommand::builder()
                    .name("glueetl")
                    .python_version("3")
                    .script_location(format!("s3://{}/job.py", self.bucket()))
                    .build(),
            )
            .glue_version("3.0")
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let job_name = create_job.name().ok_or_else(|| {
            GlueMvpError::Unknown("Did not get job name after creating
job".into())
        })?;
```

- Untuk detail API, lihat CreateJobreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteCrawler`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteCrawler`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Delete an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteCrawlerAsync(string crawlerName)
    {
        var response = await _amazonGlue.DeleteCrawlerAsync(new
DeleteCrawlerRequest { Name = crawlerName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat DeleteCrawlerdi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
 (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::DeleteCrawlerRequest request;
        request.SetName(crawler);

        Aws::Glue::Model::DeleteCrawlerOutcome outcome =
 client.DeleteCrawler(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the crawler." << std::endl;
        }
        else {
            std::cerr << "Error deleting the crawler. "
                      << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
```

- Untuk detail API, lihat DeleteCrawlerdi Referensi AWS SDK for C++ API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const deleteCrawler = (crawlerName) => {
  const client = new GlueClient({});

  const command = new DeleteCrawlerCommand({
    Name: crawlerName,
  });
```

```
  return client.send(command);
};
```

- Untuk detail API, lihat DeleteCrawlerdi Referensi AWS SDK for JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        echo "Delete the crawler.\n";
        $glueClient->deleteCrawler([
            'Name' => $crawlerName,
        ]);

    public function deleteCrawler($crawlerName)
    {
        return $this->glueClient->deleteCrawler([
            'Name' => $crawlerName,
        ]);
    }
```

- Untuk detail API, lihat DeleteCrawlerdi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def delete_crawler(self, name):
        """
        Deletes a crawler.

        :param name: The name of the crawler to delete.
        """
        try:
            self.glue_client.delete_crawler(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't delete crawler %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Untuk detail API, lihat DeleteCrawlerdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Deletes a crawler with the specified name.
  #
  # @param name [String] The name of the crawler to delete.
  # @return [void]
  def delete_crawler(name)
    @glue_client.delete_crawler(name: name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete crawler #{name}: \n#{e.message}")
    raise
  end
```

- Untuk detail API, lihat DeleteCrawlerdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        glue.delete_crawler()
            .name(self.crawler())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;
```

- Untuk detail API, lihat DeleteCrawlerreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteDatabase`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteDatabase`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Delete the AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteDatabaseAsync(string dbName)
    {
        var response = await _amazonGlue.DeleteDatabaseAsync(new
DeleteDatabaseRequest { Name = dbName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat DeleteDatabasedi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";
```

```
    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::DeleteDatabaseRequest request;
        request.SetName(database);

        Aws::Glue::Model::DeleteDatabaseOutcome outcome = client.DeleteDatabase(
                request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the database." << std::endl;
        }
        else {
            std::cerr << "Error deleting database. " <<
 outcome.GetError().GetMessage()
                      << std::endl;
            result = false;
        }
```

- Untuk detail API, lihat DeleteDatabasedi Referensi AWS SDK for C++ API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const deleteDatabase = (databaseName) => {
  const client = new GlueClient({});

  const command = new DeleteDatabaseCommand({
    Name: databaseName,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat DeleteDatabasedi Referensi AWS SDK for JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        echo "Delete the databases.\n";
        $glueClient->deleteDatabase([
            'Name' => $databaseName,
        ]);

    public function deleteDatabase($databaseName)
    {
        return $this->glueClient->deleteDatabase([
            'Name' => $databaseName,
        ]);
    }
```

- Untuk detail API, lihat DeleteDatabasedi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def delete_database(self, name):
        """
        Deletes a metadata database from your Data Catalog.

        :param name: The name of the database to delete.
        """
        try:
            self.glue_client.delete_database(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't delete database %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Untuk detail API, lihat DeleteDatabasedi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Removes a specified database from a Data Catalog.
  #
  # @param database_name [String] The name of the database to delete.
  # @return [void]
  def delete_database(database_name)
    @glue_client.delete_database(name: database_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete database: \n#{e.message}")
  end
```

- Untuk detail API, lihat DeleteDatabasedi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        glue.delete_database()
            .name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;
```

- Untuk detail API, lihat DeleteDatabasereferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`DeleteJob`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteJob`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Delete an AWS Glue job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteJobAsync(string jobName)
    {
        var response = await _amazonGlue.DeleteJobAsync(new DeleteJobRequest
{ JobName = jobName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat DeleteJobdi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::DeleteJobRequest request;
        request.SetJobName(job);

        Aws::Glue::Model::DeleteJobOutcome outcome = client.DeleteJob(request);


        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the job." << std::endl;
        }
        else {
            std::cerr << "Error deleting the job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            result = false;
        }
```

- Untuk detail API, lihat DeleteJobdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Untuk menghapus pekerjaan

`delete-job`Contoh berikut menghapus pekerjaan yang tidak lagi diperlukan.

```
aws glue delete-job \
    --job-name my-testing-job
```

Output:

```
{
    "JobName": "my-testing-job"
}
```

Untuk informasi selengkapnya, lihat Bekerja dengan Pekerjaan di AWS Glue Console di Panduan Pengembang AWS Glue.

- Untuk detail API, lihat DeleteJobdi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const deleteJob = (jobName) => {
  const client = new GlueClient({});

  const command = new DeleteJobCommand({
    JobName: jobName,
  });

  return client.send(command);
```

```
};
```

- Untuk detail API, lihat DeleteJobdi Referensi AWS SDK for JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        echo "Delete the job.\n";
        $glueClient->deleteJob([
            'JobName' => $job['Name'],
        ]);

    public function deleteJob($jobName)
    {
        return $this->glueClient->deleteJob([
            'JobName' => $jobName,
        ]);
    }
```

- Untuk detail API, lihat DeleteJobdi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def delete_job(self, job_name):
        """
        Deletes a job definition. This also deletes data about all runs that are
        associated with this job definition.

        :param job_name: The name of the job definition to delete.
        """
        try:
            self.glue_client.delete_job(JobName=job_name)
        except ClientError as err:
            logger.error(
                "Couldn't delete job %s. Here's why: %s: %s",
                job_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Untuk detail API, lihat DeleteJobdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Deletes a job with the specified name.
  #
  # @param job_name [String] The name of the job to delete.
  # @return [void]
  def delete_job(job_name)
    @glue_client.delete_job(job_name: job_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end
```

- Untuk detail API, lihat DeleteJobdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        glue.delete_job()
            .job_name(self.job())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;
```

- Untuk detail API, lihat DeleteJobreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan `DeleteTable` dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`DeleteTable`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Delete a table from an AWS Glue database.
    /// </summary>
    /// <param name="tableName">The table to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteTableAsync(string dbName, string tableName)
    {
        var response = await _amazonGlue.DeleteTableAsync(new DeleteTableRequest
{ Name = tableName, DatabaseName = dbName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Untuk detail API, lihat DeleteTabledi Referensi AWS SDK for .NET API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const deleteTable = (databaseName, tableName) => {
  const client = new GlueClient({});

  const command = new DeleteTableCommand({
    DatabaseName: databaseName,
    Name: tableName,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat DeleteTabledi Referensi AWS SDK for JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        echo "Delete the tables.\n";
```

```
        foreach ($tables['TableList'] as $table) {
            $glueService->deleteTable($table['Name'], $databaseName);
        }

    public function deleteTable($tableName, $databaseName)
    {
        return $this->glueClient->deleteTable([
            'DatabaseName' => $databaseName,
            'Name' => $tableName,
        ]);
    }
```

- Untuk detail API, lihat DeleteTabledi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def delete_table(self, db_name, table_name):
        """
        Deletes a table from a metadata database.

        :param db_name: The name of the database that contains the table.
        :param table_name: The name of the table to delete.
        """
```

```
        try:
            self.glue_client.delete_table(DatabaseName=db_name, Name=table_name)
        except ClientError as err:
            logger.error(
                "Couldn't delete table %s. Here's why: %s: %s",
                table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Untuk detail API, lihat DeleteTabledi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Deletes a table with the specified name.
  #
  # @param database_name [String] The name of the catalog database in which the
 table resides.
```

```
  # @param table_name [String] The name of the table to be deleted.
  # @return [void]
  def delete_table(database_name, table_name)
    @glue_client.delete_table(database_name: database_name, name: table_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end
```

- Untuk detail API, lihat DeleteTabledi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        for t in &self.tables {
            glue.delete_table()
                .name(t.name())
                .database_name(self.database())
                .send()
                .await
                .map_err(GlueMvpError::from_glue_sdk)?;
        }
```

- Untuk detail API, lihat DeleteTablereferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetCrawler`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetCrawler`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Get information about an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A Crawler object describing the crawler.</returns>
    public async Task<Crawler?> GetCrawlerAsync(string crawlerName)
    {
        var crawlerRequest = new GetCrawlerRequest
        {
            Name = crawlerName,
        };

        var response = await _amazonGlue.GetCrawlerAsync(crawlerRequest);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            var databaseName = response.Crawler.DatabaseName;
            Console.WriteLine($"{crawlerName} has the database {databaseName}");
            return response.Crawler;
        }

        Console.WriteLine($"No information regarding {crawlerName} could be
found.");
        return null;
    }
```

- Untuk detail API, lihat GetCrawlerdi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
       Aws::Client::ClientConfiguration clientConfig;
       // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
       // clientConfig.region = "us-east-1";

   Aws::Glue::GlueClient client(clientConfig);

       Aws::Glue::Model::GetCrawlerRequest request;
       request.SetName(CRAWLER_NAME);

       Aws::Glue::Model::GetCrawlerOutcome outcome = client.GetCrawler(request);

       if (outcome.IsSuccess()) {
           Aws::Glue::Model::CrawlerState crawlerState =
outcome.GetResult().GetCrawler().GetState();
           std::cout << "Retrieved crawler with state " <<

Aws::Glue::Model::CrawlerStateMapper::GetNameForCrawlerState(
                           crawlerState)
                   << "." << std::endl;
       }
       else {
           std::cerr << "Error retrieving a crawler.  "
                     << outcome.GetError().GetMessage() << std::endl;
           deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                        clientConfig);
           return false;
       }
```

- Untuk detail API, lihat GetCrawlerdi Referensi AWS SDK for C++ API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glue.GlueClient;
import software.amazon.awssdk.services.glue.model.GetCrawlerRequest;
import software.amazon.awssdk.services.glue.model.GetCrawlerResponse;
import software.amazon.awssdk.services.glue.model.GlueException;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class GetCrawler {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <crawlerName>

                Where:
                    crawlerName - The name of the crawler.\s
                """;
```

```
        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String crawlerName = args[0];
        Region region = Region.US_EAST_1;
        GlueClient glueClient = GlueClient.builder()
                .region(region)
                .build();

        getSpecificCrawler(glueClient, crawlerName);
        glueClient.close();
    }

    public static void getSpecificCrawler(GlueClient glueClient, String
 crawlerName) {
        try {
            GetCrawlerRequest crawlerRequest = GetCrawlerRequest.builder()
                    .name(crawlerName)
                    .build();

            GetCrawlerResponse response = glueClient.getCrawler(crawlerRequest);
            Instant createDate = response.crawler().creationTime();

            // Convert the Instant to readable date
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(createDate);
            System.out.println("The create date of the Crawler is " +
 createDate);

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat GetCrawlerdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const getCrawler = (name) => {
  const client = new GlueClient({});

  const command = new GetCrawlerCommand({
    Name: name,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat GetCrawlerdi Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun getSpecificCrawler(crawlerName: String?) {
    val request =
        GetCrawlerRequest {
            name = crawlerName
```

```
        }
    GlueClient { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getCrawler(request)
        val role = response.crawler?.role
        println("The role associated with this crawler is $role")
    }
}
```

- Untuk detail API, lihat GetCrawlerdi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        echo "Waiting for crawler";
        do {
            $crawler = $glueService->getCrawler($crawlerName);
            echo ".";
            sleep(10);
        } while ($crawler['Crawler']['State'] != "READY");
        echo "\n";

    public function getCrawler($crawlerName)
    {
        return $this->customWaiter(function () use ($crawlerName) {
            return $this->glueClient->getCrawler([
                'Name' => $crawlerName,
            ]);
        });
    }
```

- Untuk detail API, lihat GetCrawlerdi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_crawler(self, name):
        """
        Gets information about a crawler.

        :param name: The name of the crawler to look up.
        :return: Data about the crawler.
        """
        crawler = None
        try:
            response = self.glue_client.get_crawler(Name=name)
            crawler = response["Crawler"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityNotFoundException":
                logger.info("Crawler %s doesn't exist.", name)
            else:
                logger.error(
                    "Couldn't get crawler %s. Here's why: %s: %s",
                    name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
```

```
        return crawler
```

- Untuk detail API, lihat GetCrawlerdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves information about a specific crawler.
  #
  # @param name [String] The name of the crawler to retrieve information about.
  # @return [Aws::Glue::Types::Crawler, nil] The crawler object if found, or nil
 if not found.
  def get_crawler(name)
    @glue_client.get_crawler(name: name)
  rescue Aws::Glue::Errors::EntityNotFoundException
    @logger.info("Crawler #{name} doesn't exist.")
    false
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get crawler #{name}: \n#{e.message}")
    raise
```

```
end
```

- Untuk detail API, lihat GetCrawlerdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
            let tmp_crawler = glue
                .get_crawler()
                .name(self.crawler())
                .send()
                .await
                .map_err(GlueMvpError::from_glue_sdk)?;
```

- Untuk detail API, lihat GetCrawlerreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetDatabase`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetDatabase`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Get information about an AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A Database object containing information about the database.</
returns>
    public async Task<Database> GetDatabaseAsync(string dbName)
    {
        var databasesRequest = new GetDatabaseRequest
        {
            Name = dbName,
        };

        var response = await _amazonGlue.GetDatabaseAsync(databasesRequest);
        return response.Database;
    }
```

- Untuk detail API, lihat GetDatabasedi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::GetDatabaseRequest request;
        request.SetName(CRAWLER_DATABASE_NAME);

        Aws::Glue::Model::GetDatabaseOutcome outcome =
client.GetDatabase(request);

        if (outcome.IsSuccess()) {
            const Aws::Glue::Model::Database &database =
outcome.GetResult().GetDatabase();

            std::cout << "Successfully retrieve the database\n" <<
                      database.Jsonize().View().WriteReadable() << "'." <<
std::endl;
        }
        else {
            std::cerr << "Error getting the database.  "
                      << outcome.GetError().GetMessage() << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
```

- Untuk detail API, lihat GetDatabasedi Referensi AWS SDK for C++ API.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glue.GlueClient;
import software.amazon.awssdk.services.glue.model.GetDatabaseRequest;
import software.amazon.awssdk.services.glue.model.GetDatabaseResponse;
import software.amazon.awssdk.services.glue.model.GlueException;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class GetDatabase {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <databaseName>

                Where:
                    databaseName - The name of the database.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String databaseName = args[0];
        Region region = Region.US_EAST_1;
        GlueClient glueClient = GlueClient.builder()
                .region(region)
                .build();

        getSpecificDatabase(glueClient, databaseName);
```

```
        glueClient.close();
    }

    public static void getSpecificDatabase(GlueClient glueClient, String
 databaseName) {
        try {
            GetDatabaseRequest databasesRequest = GetDatabaseRequest.builder()
                    .name(databaseName)
                    .build();

            GetDatabaseResponse response =
 glueClient.getDatabase(databasesRequest);
            Instant createDate = response.database().createTime();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(createDate);
            System.out.println("The create date of the database is " +
 createDate);

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat GetDatabasedi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const getDatabase = (name) => {
  const client = new GlueClient({});

  const command = new GetDatabaseCommand({
    Name: name,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat GetDatabasedi Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun getSpecificDatabase(databaseName: String?) {
    val request =
        GetDatabaseRequest {
            name = databaseName
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getDatabase(request)
        val dbDesc = response.database?.description
        println("The database description is $dbDesc")
    }
}
```

- Untuk detail API, lihat GetDatabasedi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        $databaseName = "doc-example-database-$uniqid";

        $database = $glueService->getDatabase($databaseName);
        echo "Found a database named " . $database['Database']['Name'] . "\n";

    public function getDatabase(string $databaseName): Result
    {
        return $this->customWaiter(function () use ($databaseName) {
            return $this->glueClient->getDatabase([
                'Name' => $databaseName,
            ]);
        });
    }
```

- Untuk detail API, lihat GetDatabasedi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""
```

```
    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_database(self, name):
        """
        Gets information about a database in your Data Catalog.

        :param name: The name of the database to look up.
        :return: Information about the database.
        """
        try:
            response = self.glue_client.get_database(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't get database %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["Database"]
```

- Untuk detail API, lihat GetDatabasedi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves information about a specific database.
  #
  # @param name [String] The name of the database to retrieve information about.
  # @return [Aws::Glue::Types::Database, nil] The database object if found, or
 nil if not found.
  def get_database(name)
    response = @glue_client.get_database(name: name)
    response.database
rescue Aws::Glue::Errors::GlueException => e
  @logger.error("Glue could not get database #{name}: \n#{e.message}")
  raise
  end
```

- Untuk detail API, lihat GetDatabasedi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        let database = glue
            .get_database()
            .name(self.database())
            .send()
```

```
            .await
            .map_err(GlueMvpError::from_glue_sdk)?
            .to_owned();
        let database = database
            .database()
            .ok_or_else(|| GlueMvpError::Unknown("Could not find
database".into()))?;
```

- Untuk detail API, lihat GetDatabasereferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetDatabases`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetDatabases`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

CLI

AWS CLI

Untuk membuat daftar definisi dari beberapa atau semua database dalam Katalog Data AWS Glue

`get-databases`Contoh berikut mengembalikan informasi tentang database dalam Katalog Data.

```
aws glue get-databases
```

Output:

```
{
    "DatabaseList": [
        {
```

```
            "Name": "default",
            "Description": "Default Hive database",
            "LocationUri": "file:/spark-warehouse",
            "CreateTime": 1602084052.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        },
        {
            "Name": "flights-db",
            "CreateTime": 1587072847.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        },
        {
            "Name": "legislators",
            "CreateTime": 1601415625.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
```

```
            "CatalogId": "111122223333"
        },
        {
            "Name": "tempdb",
            "CreateTime": 1601498566.0,
            "CreateTableDefaultPermissions": [
                {
                    "Principal": {
                        "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                    },
                    "Permissions": [
                        "ALL"
                    ]
                }
            ],
            "CatalogId": "111122223333"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Mendefinisikan Database di Katalog Data Anda di Panduan Pengembang AWS Glue.

- Untuk detail API, lihat GetDatabasesdi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const getDatabases = () => {
  const client = new GlueClient({});

  const command = new GetDatabasesCommand({});

  return client.send(command);
};
```

- Untuk detail API, lihat GetDatabasesdi Referensi AWS SDK for JavaScript API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetJob`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetJob`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

CLI

AWS CLI

Untuk mengambil informasi tentang pekerjaan

`get-job`Contoh berikut mengambil informasi tentang pekerjaan.

```
aws glue get-job \
    --job-name my-testing-job
```

Output:

```
{
    "Job": {
        "Name": "my-testing-job",
        "Role": "Glue_DefaultRole",
        "CreatedOn": 1602805698.167,
        "LastModifiedOn": 1602805698.167,
        "ExecutionProperty": {
            "MaxConcurrentRuns": 1
        },
        "Command": {
```

```
            "Name": "gluestreaming",
            "ScriptLocation": "s3://janetst-bucket-01/Scripts/test_script.scala",
            "PythonVersion": "2"
        },
        "DefaultArguments": {
            "--class": "GlueApp",
            "--job-language": "scala"
        },
        "MaxRetries": 0,
        "AllocatedCapacity": 10,
        "MaxCapacity": 10.0,
        "GlueVersion": "1.0"
    }
}
```

Untuk informasi selengkapnya, lihat Pekerjaan di Panduan Pengembang AWS Glue.

- Untuk detail API, lihat GetJobdi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const getJob = (jobName) => {
  const client = new GlueClient({});

  const command = new GetJobCommand({
    JobName: jobName,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat GetJobdi Referensi AWS SDK for JavaScript API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetJobRun`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetJobRun`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Get information about a specific AWS Glue job run.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <param name="jobRunId">The Id of the job run.</param>
    /// <returns>A JobRun object with information about the job run.</returns>
    public async Task<JobRun> GetJobRunAsync(string jobName, string jobRunId)
    {
        var response = await _amazonGlue.GetJobRunAsync(new GetJobRunRequest
{ JobName = jobName, RunId = jobRunId });
        return response.JobRun;
    }
```

- Untuk detail API, lihat GetJobRundi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::GetJobRunRequest jobRunRequest;
        jobRunRequest.SetJobName(jobName);
        jobRunRequest.SetRunId(jobRunID);

        Aws::Glue::Model::GetJobRunOutcome jobRunOutcome = client.GetJobRun(
                jobRunRequest);

        if (jobRunOutcome.IsSuccess()) {
            std::cout << "Displaying the job run JSON description." << std::endl;
            std::cout
                    <<
jobRunOutcome.GetResult().GetJobRun().Jsonize().View().WriteReadable()
                    << std::endl;
        }
        else {
            std::cerr << "Error get a job run. "
                      << jobRunOutcome.GetError().GetMessage()
                      << std::endl;
        }
```

- Untuk detail API, lihat GetJobRundi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Untuk mendapatkan informasi tentang menjalankan pekerjaan

`get-job-run`Contoh berikut mengambil informasi tentang menjalankan pekerjaan.

```
aws glue get-job-run \
    --job-name "Combine legistators data" \
    --run-id jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e
```

Output:

```
{
    "JobRun": {
        "Id":
 "jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e",
        "Attempt": 0,
        "JobName": "Combine legistators data",
        "StartedOn": 1602873931.255,
        "LastModifiedOn": 1602874075.985,
        "CompletedOn": 1602874075.985,
        "JobRunState": "SUCCEEDED",
        "Arguments": {
            "--enable-continuous-cloudwatch-log": "true",
            "--enable-metrics": "",
            "--enable-spark-ui": "true",
            "--job-bookmark-option": "job-bookmark-enable",
            "--spark-event-logs-path": "s3://aws-glue-assets-111122223333-us-
east-1/sparkHistoryLogs/"
        },
        "PredecessorRuns": [],
        "AllocatedCapacity": 10,
        "ExecutionTime": 117,
        "Timeout": 2880,
        "MaxCapacity": 10.0,
        "WorkerType": "G.1X",
        "NumberOfWorkers": 10,
        "LogGroupName": "/aws-glue/jobs",
        "GlueVersion": "2.0"
    }
}
```

Untuk informasi selengkapnya, lihat Job Runs di Panduan Pengembang AWS Glue.

- Untuk detail API, lihat GetJobRundi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const getJobRun = (jobName, jobRunId) => {
  const client = new GlueClient({});
  const command = new GetJobRunCommand({
    JobName: jobName,
    RunId: jobRunId,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat GetJobRundi Referensi AWS SDK for JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        $jobName = 'test-job-' . $uniqid;

        $outputBucketUrl = "s3://$bucketName";
```

```
        $runId = $glueService->startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl)['JobRunId'];

        echo "waiting for job";
        do {
            $jobRun = $glueService->getJobRun($jobName, $runId);
            echo ".";
            sleep(10);
        } while (!array_intersect([$jobRun['JobRun']['JobRunState']],
['SUCCEEDED', 'STOPPED', 'FAILED', 'TIMEOUT']));
        echo "\n";

    public function getJobRun($jobName, $runId, $predecessorsIncluded = false):
Result
    {
        return $this->glueClient->getJobRun([
            'JobName' => $jobName,
            'RunId' => $runId,
            'PredecessorsIncluded' => $predecessorsIncluded,
        ]);
    }
```

- Untuk detail API, lihat GetJobRundi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
```

```
        self.glue_client = glue_client


    def get_job_run(self, name, run_id):
        """
        Gets information about a single job run.

        :param name: The name of the job definition for the run.
        :param run_id: The ID of the run.
        :return: Information about the run.
        """
        try:
            response = self.glue_client.get_job_run(JobName=name, RunId=run_id)
        except ClientError as err:
            logger.error(
                "Couldn't get job run %s/%s. Here's why: %s: %s",
                name,
                run_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRun"]
```

- Untuk detail API, lihat GetJobRundi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
```

```
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves data for a specific job run.
  #
  # @param job_name [String] The name of the job run to retrieve data for.
  # @return [Glue::Types::GetJobRunResponse]
  def get_job_run(job_name, run_id)
    @glue_client.get_job_run(job_name: job_name, run_id: run_id)
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end
```

- Untuk detail API, lihat GetJobRundi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        let get_job_run = || async {
            Ok::<JobRun, GlueMvpError>(
                glue.get_job_run()
                    .job_name(self.job())
                    .run_id(job_run_id.to_string())
                    .send()
                    .await
                    .map_err(GlueMvpError::from_glue_sdk)?
                    .job_run()
```

```
                    .ok_or_else(|| GlueMvpError::Unknown("Failed to get
job_run".into()))?
                    .to_owned(),
            )
        };

        let mut job_run = get_job_run().await?;
        let mut state =
job_run.job_run_state().unwrap_or(&unknown_state).to_owned();

        while matches!(
            state,
            JobRunState::Starting | JobRunState::Stopping | JobRunState::Running
        ) {
            info!(?state, "Waiting for job to finish");
            tokio::time::sleep(self.wait_delay).await;

            job_run = get_job_run().await?;
            state = job_run.job_run_state().unwrap_or(&unknown_state).to_owned();
        }
```

- Untuk detail API, lihat GetJobRunreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetJobRuns`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetJobRuns`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// Get information about all AWS Glue runs of a specific job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A list of JobRun objects.</returns>
    public async Task<List<JobRun>> GetJobRunsAsync(string jobName)
    {
        var jobRuns = new List<JobRun>();

        var request = new GetJobRunsRequest
        {
            JobName = jobName,
        };

        // No need to loop to get all the log groups--the SDK does it for us
behind the scenes
        var paginatorForJobRuns =
            _amazonGlue.Paginators.GetJobRuns(request);

        await foreach (var response in paginatorForJobRuns.Responses)
        {
            response.JobRuns.ForEach(jobRun =>
            {
                jobRuns.Add(jobRun);
            });
        }

        return jobRuns;
    }
```

- Untuk detail API, lihat GetJobRunsdi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
       Aws::Client::ClientConfiguration clientConfig;
       // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
       // clientConfig.region = "us-east-1";

   Aws::Glue::GlueClient client(clientConfig);

       Aws::Glue::Model::GetJobRunsRequest getJobRunsRequest;
       getJobRunsRequest.SetJobName(jobName);

       Aws::String nextToken; // Used for pagination.
       std::vector<Aws::Glue::Model::JobRun> allJobRuns;
       do {
           if (!nextToken.empty()) {
               getJobRunsRequest.SetNextToken(nextToken);
           }
           Aws::Glue::Model::GetJobRunsOutcome jobRunsOutcome =
client.GetJobRuns(
                   getJobRunsRequest);

           if (jobRunsOutcome.IsSuccess()) {
               const std::vector<Aws::Glue::Model::JobRun> &jobRuns =
jobRunsOutcome.GetResult().GetJobRuns();
               allJobRuns.insert(allJobRuns.end(), jobRuns.begin(),
jobRuns.end());

               nextToken = jobRunsOutcome.GetResult().GetNextToken();
           }
           else {
               std::cerr << "Error getting job runs. "
```

```
                        << jobRunsOutcome.GetError().GetMessage()
                        << std::endl;
                break;
            }
        } while (!nextToken.empty());
```

- Untuk detail API, lihat GetJobRunsdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Untuk mendapatkan informasi tentang semua pekerjaan yang dijalankan untuk suatu pekerjaan

`get-job-runs`Contoh berikut mengambil informasi tentang pekerjaan berjalan untuk suatu pekerjaan.

```
aws glue get-job-runs \
    --job-name "my-testing-job"
```

Output:

```
{
    "JobRuns": [
        {
            "Id":
 "jr_012e176506505074d94d761755e5c62538ee1aad6f17d39f527e9140cf0c9a5e",
            "Attempt": 0,
            "JobName": "my-testing-job",
            "StartedOn": 1602873931.255,
            "LastModifiedOn": 1602874075.985,
            "CompletedOn": 1602874075.985,
            "JobRunState": "SUCCEEDED",
            "Arguments": {
                "--enable-continuous-cloudwatch-log": "true",
                "--enable-metrics": "",
                "--enable-spark-ui": "true",
                "--job-bookmark-option": "job-bookmark-enable",
                "--spark-event-logs-path": "s3://aws-glue-assets-111122223333-us-
east-1/sparkHistoryLogs/"
```

```
            },
            "PredecessorRuns": [],
            "AllocatedCapacity": 10,
            "ExecutionTime": 117,
            "Timeout": 2880,
            "MaxCapacity": 10.0,
            "WorkerType": "G.1X",
            "NumberOfWorkers": 10,
            "LogGroupName": "/aws-glue/jobs",
            "GlueVersion": "2.0"
        },
        {
            "Id":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_2",
            "Attempt": 2,
            "PreviousRunId":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_1",
            "JobName": "my-testing-job",
            "StartedOn": 1602811168.496,
            "LastModifiedOn": 1602811282.39,
            "CompletedOn": 1602811282.39,
            "JobRunState": "FAILED",
            "ErrorMessage": "An error occurred while calling
 o122.pyWriteDynamicFrame.
                Access Denied (Service: Amazon S3; Status Code: 403; Error Code:
 AccessDenied;
                Request ID: 021AAB703DB20A2D;
                S3 Extended Request ID: teZk24Y09TkXzBvMPG502L5VJBhe9DJuWA9/
TXtuGOqfByajkfL/Tlqt5JBGdEGpigAqzdMDM/U=)",
            "PredecessorRuns": [],
            "AllocatedCapacity": 10,
            "ExecutionTime": 110,
            "Timeout": 2880,
            "MaxCapacity": 10.0,
            "WorkerType": "G.1X",
            "NumberOfWorkers": 10,
            "LogGroupName": "/aws-glue/jobs",
            "GlueVersion": "2.0"
        },
        {
            "Id":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f_attempt_1",
            "Attempt": 1,
```

```
            "PreviousRunId":
 "jr_03cc19ddab11c4e244d3f735567de74ff93b0b3ef468a713ffe73e53d1aec08f",
            "JobName": "my-testing-job",
            "StartedOn": 1602811020.518,
            "LastModifiedOn": 1602811138.364,
            "CompletedOn": 1602811138.364,
            "JobRunState": "FAILED",
            "ErrorMessage": "An error occurred while calling
 o122.pyWriteDynamicFrame.
                Access Denied (Service: Amazon S3; Status Code: 403; Error Code:
 AccessDenied;
                Request ID: 2671D37856AE7ABB;
                S3 Extended Request ID: RLJCJw20brV
+PpC6GpORahyF2fp9flB5SSb2bTGPnUSPVizLXRl1PN3QZldb+v1o9qRVktNYbW8=)",
            "PredecessorRuns": [],
            "AllocatedCapacity": 10,
            "ExecutionTime": 113,
            "Timeout": 2880,
            "MaxCapacity": 10.0,
            "WorkerType": "G.1X",
            "NumberOfWorkers": 10,
            "LogGroupName": "/aws-glue/jobs",
            "GlueVersion": "2.0"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Job Runs di Panduan Pengembang AWS Glue.

- Untuk detail API, lihat GetJobRunsdi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const getJobRuns = (jobName) => {
```

```
  const client = new GlueClient({});
  const command = new GetJobRunsCommand({
    JobName: jobName,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat GetJobRunsdi Referensi AWS SDK for JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        $jobName = 'test-job-' . $uniqid;

        $jobRuns = $glueService->getJobRuns($jobName);

    public function getJobRuns($jobName, $maxResults = 0, $nextToken = ''):
Result
    {
        $arguments = ['JobName' => $jobName];
        if ($maxResults) {
            $arguments['MaxResults'] = $maxResults;
        }
        if ($nextToken) {
            $arguments['NextToken'] = $nextToken;
        }
        return $this->glueClient->getJobRuns($arguments);
    }
```

- Untuk detail API, lihat GetJobRunsdi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_job_runs(self, job_name):
        """
        Gets information about runs that have been performed for a specific job
        definition.

        :param job_name: The name of the job definition to look up.
        :return: The list of job runs.
        """
        try:
            response = self.glue_client.get_job_runs(JobName=job_name)
        except ClientError as err:
            logger.error(
                "Couldn't get job runs for %s. Here's why: %s: %s",
                job_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRuns"]
```

- Untuk detail API, lihat GetJobRunsdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves a list of job runs for the specified job.
  #
  # @param job_name [String] The name of the job to retrieve job runs for.
  # @return [Array<Aws::Glue::Types::JobRun>]
  def get_job_runs(job_name)
    response = @glue_client.get_job_runs(job_name: job_name)
    response.job_runs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end
```

- Untuk detail API, lihat GetJobRunsdi Referensi AWS SDK for Ruby API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`GetTables`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`GetTables`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
/// <summary>
/// Get a list of tables for an AWS Glue database.
/// </summary>
/// <param name="dbName">The name of the database.</param>
/// <returns>A list of Table objects.</returns>
public async Task<List<Table>> GetTablesAsync(string dbName)
{
    var request = new GetTablesRequest { DatabaseName = dbName };
    var tables = new List<Table>();

    // Get a paginator for listing the tables.
    var tablePaginator = _amazonGlue.Paginators.GetTables(request);

    await foreach (var response in tablePaginator.Responses)
    {
        tables.AddRange(response.TableList);
    }
```

```
        return tables;
    }
```

- Untuk detail API, lihat GetTablesdi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
       Aws::Client::ClientConfiguration clientConfig;
       // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
       // clientConfig.region = "us-east-1";

   Aws::Glue::GlueClient client(clientConfig);

       Aws::Glue::Model::GetTablesRequest request;
       request.SetDatabaseName(CRAWLER_DATABASE_NAME);
       std::vector<Aws::Glue::Model::Table> all_tables;
       Aws::String nextToken; // Used for pagination.
       do {
           Aws::Glue::Model::GetTablesOutcome outcome =
client.GetTables(request);

           if (outcome.IsSuccess()) {
               const std::vector<Aws::Glue::Model::Table> &tables =
outcome.GetResult().GetTableList();
               all_tables.insert(all_tables.end(), tables.begin(),
tables.end());
               nextToken = outcome.GetResult().GetNextToken();
           }
           else {
               std::cerr << "Error getting the tables. "
                         << outcome.GetError().GetMessage()
```

```
                        << std::endl;
                deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                             clientConfig);
                return false;
            }
        } while (!nextToken.empty());

        std::cout << "The database contains " << all_tables.size()
                  << (all_tables.size() == 1 ?
                      " table." : "tables.") << std::endl;
        std::cout << "Here is a list of the tables in the database.";
        for (size_t index = 0; index < all_tables.size(); ++index) {
            std::cout << "    " << index + 1 << ":  " <<
all_tables[index].GetName()
                      << std::endl;
        }

        if (!all_tables.empty()) {
            int tableIndex = askQuestionForIntRange(
                    "Enter an index to display the database detail ",
                    1, static_cast<int>(all_tables.size()));
            std::cout << all_tables[tableIndex -
1].Jsonize().View().WriteReadable()
                      << std::endl;

            tableName = all_tables[tableIndex - 1].GetName();
        }
```

- Untuk detail API, lihat GetTablesdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Untuk daftar definisi dari beberapa atau semua tabel dalam database yang ditentukan

`get-tables`Contoh berikut mengembalikan informasi tentang tabel dalam database tertentu.

```
aws glue get-tables --database-name 'tempdb'
```

Output:

```
{
    "TableList": [
        {
            "Name": "my-s3-sink",
            "DatabaseName": "tempdb",
            "CreateTime": 1602730539.0,
            "UpdateTime": 1602730539.0,
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "sensorid",
                        "Type": "int"
                    },
                    {
                        "Name": "currenttemperature",
                        "Type": "int"
                    },
                    {
                        "Name": "status",
                        "Type": "string"
                    }
                ],
                "Location": "s3://janetst-bucket-01/test-s3-output/",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SerdeInfo": {
                    "SerializationLibrary": "org.openx.data.jsonserde.JsonSerDe"
                },
                "SortColumns": [],
                "StoredAsSubDirectories": false
            },
            "Parameters": {
                "classification": "json"
            },
            "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
            "IsRegisteredWithLakeFormation": false,
            "CatalogId": "007436865787"
        },
        {
            "Name": "s3-source",
            "DatabaseName": "tempdb",
            "CreateTime": 1602730658.0,
```

```
            "UpdateTime": 1602730658.0,
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "sensorid",
                        "Type": "int"
                    },
                    {
                        "Name": "currenttemperature",
                        "Type": "int"
                    },
                    {
                        "Name": "status",
                        "Type": "string"
                    }
                ],
                "Location": "s3://janetst-bucket-01/",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SortColumns": [],
                "StoredAsSubDirectories": false
            },
            "Parameters": {
                "classification": "json"
            },
            "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
            "IsRegisteredWithLakeFormation": false,
            "CatalogId": "007436865787"
        },
        {
            "Name": "test-kinesis-input",
            "DatabaseName": "tempdb",
            "CreateTime": 1601507001.0,
            "UpdateTime": 1601507001.0,
            "Retention": 0,
            "StorageDescriptor": {
                "Columns": [
                    {
                        "Name": "sensorid",
                        "Type": "int"
                    },
                    {
                        "Name": "currenttemperature",
```

```
                        "Type": "int"
                    },
                    {
                        "Name": "status",
                        "Type": "string"
                    }
                ],
                "Location": "my-testing-stream",
                "Compressed": false,
                "NumberOfBuckets": 0,
                "SerdeInfo": {
                    "SerializationLibrary": "org.openx.data.jsonserde.JsonSerDe"
                },
                "SortColumns": [],
                "Parameters": {
                    "kinesisUrl": "https://kinesis.us-east-1.amazonaws.com",
                    "streamName": "my-testing-stream",
                    "typeOfData": "kinesis"
                },
                "StoredAsSubDirectories": false
            },
            "Parameters": {
                "classification": "json"
            },
            "CreatedBy": "arn:aws:iam::007436865787:user/JRSTERN",
            "IsRegisteredWithLakeFormation": false,
            "CatalogId": "007436865787"
        }
    ]
}
```

Untuk informasi selengkapnya, lihat Mendefinisikan Tabel di Katalog Data AWS Glue di Panduan Pengembang AWS Glue.

- Untuk detail API, lihat GetTablesdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glue.GlueClient;
import software.amazon.awssdk.services.glue.model.GetTableRequest;
import software.amazon.awssdk.services.glue.model.GetTableResponse;
import software.amazon.awssdk.services.glue.model.GlueException;
import java.time.Instant;
import java.time.ZoneId;
import java.time.format.DateTimeFormatter;
import java.time.format.FormatStyle;
import java.util.Locale;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class GetTable {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <dbName> <tableName>

                Where:
                    dbName - The database name.\s
                    tableName - The name of the table.\s
                """;
```

```
        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String dbName = args[0];
        String tableName = args[1];
        Region region = Region.US_EAST_1;
        GlueClient glueClient = GlueClient.builder()
                .region(region)
                .build();

        getGlueTable(glueClient, dbName, tableName);
        glueClient.close();
    }

    public static void getGlueTable(GlueClient glueClient, String dbName, String
 tableName) {
        try {
            GetTableRequest tableRequest = GetTableRequest.builder()
                    .databaseName(dbName)
                    .name(tableName)
                    .build();

            GetTableResponse tableResponse = glueClient.getTable(tableRequest);
            Instant createDate = tableResponse.table().createTime();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
 DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(createDate);
            System.out.println("The create date of the table is " + createDate);

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat GetTablesdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const getTables = (databaseName) => {
  const client = new GlueClient({});

  const command = new GetTablesCommand({
    DatabaseName: databaseName,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat GetTablesdi Referensi AWS SDK for JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        $databaseName = "doc-example-database-$uniqid";

        $tables = $glueService->getTables($databaseName);
```

```
    public function getTables($databaseName): Result
    {
        return $this->glueClient->getTables([
            'DatabaseName' => $databaseName,
        ]);
    }
```

- Untuk detail API, lihat GetTablesdi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def get_tables(self, db_name):
        """
        Gets a list of tables in a Data Catalog database.

        :param db_name: The name of the database to query.
        :return: The list of tables in the database.
        """
        try:
            response = self.glue_client.get_tables(DatabaseName=db_name)
        except ClientError as err:
            logger.error(
                "Couldn't get tables %s. Here's why: %s: %s",
```

```
                db_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["TableList"]
```

- Untuk detail API, lihat GetTablesdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves a list of tables in the specified database.
  #
  # @param db_name [String] The name of the database to retrieve tables from.
  # @return [Array<Aws::Glue::Types::Table>]
  def get_tables(db_name)
    response = @glue_client.get_tables(database_name: db_name)
    response.table_list
```

```
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get tables #{db_name}: \n#{e.message}")
    raise
  end
```

- Untuk detail API, lihat GetTablesdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        let tables = glue
            .get_tables()
            .database_name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let tables = tables.table_list();
```

- Untuk detail API, lihat GetTablesreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`ListJobs`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`ListJobs`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    /// <summary>
    /// List AWS Glue jobs using a paginator.
    /// </summary>
    /// <returns>A list of AWS Glue job names.</returns>
    public async Task<List<string>> ListJobsAsync()
    {
        var jobNames = new List<string>();

        var listJobsPaginator = _amazonGlue.Paginators.ListJobs(new
ListJobsRequest { MaxResults = 10 });
        await foreach (var response in listJobsPaginator.Responses)
        {
            jobNames.AddRange(response.JobNames);
        }

        return jobNames;
    }
```

- Untuk detail API, lihat ListJobsdi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::ListJobsRequest listJobsRequest;

        Aws::String nextToken;
        std::vector<Aws::String> allJobNames;

        do {
            if (!nextToken.empty()) {
                listJobsRequest.SetNextToken(nextToken);
            }
            Aws::Glue::Model::ListJobsOutcome listRunsOutcome = client.ListJobs(
                    listJobsRequest);

            if (listRunsOutcome.IsSuccess()) {
                const std::vector<Aws::String> &jobNames =
listRunsOutcome.GetResult().GetJobNames();
                allJobNames.insert(allJobNames.end(), jobNames.begin(),
jobNames.end());
                nextToken = listRunsOutcome.GetResult().GetNextToken();
            }
            else {
                std::cerr << "Error listing jobs. "
                          << listRunsOutcome.GetError().GetMessage()
                          << std::endl;
            }
        } while (!nextToken.empty());
```

- Untuk detail API, lihat ListJobsdi Referensi AWS SDK for C++ API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const listJobs = () => {
  const client = new GlueClient({});

  const command = new ListJobsCommand({});

  return client.send(command);
};
```

- Untuk detail API, lihat ListJobsdi Referensi AWS SDK for JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        $jobs = $glueService->listJobs();
        echo "Current jobs:\n";
        foreach ($jobs['JobNames'] as $jobsName) {
```

```
            echo "{$jobsName}\n";
        }

    public function listJobs($maxResults = null, $nextToken = null, $tags = []):
 Result
    {
        $arguments = [];
        if ($maxResults) {
            $arguments['MaxResults'] = $maxResults;
        }
        if ($nextToken) {
            $arguments['NextToken'] = $nextToken;
        }
        if (!empty($tags)) {
            $arguments['Tags'] = $tags;
        }
        return $this->glueClient->listJobs($arguments);
    }
```

- Untuk detail API, lihat ListJobsdi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client
```

```
    def list_jobs(self):
        """
        Lists the names of job definitions in your account.

        :return: The list of job definition names.
        """
        try:
            response = self.glue_client.list_jobs()
        except ClientError as err:
            logger.error(
                "Couldn't list jobs. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobNames"]
```

- Untuk detail API, lihat ListJobsdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
```

```
  @logger = logger
end

# Retrieves a list of jobs in AWS Glue.
#
# @return [Aws::Glue::Types::ListJobsResponse]
def list_jobs
  @glue_client.list_jobs
rescue Aws::Glue::Errors::GlueException => e
  @logger.error("Glue could not list jobs: \n#{e.message}")
  raise
end
```

- Untuk detail API, lihat ListJobsdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
    let mut list_jobs = glue.list_jobs().into_paginator().send();
    while let Some(list_jobs_output) = list_jobs.next().await {
        match list_jobs_output {
            Ok(list_jobs) => {
                let names = list_jobs.job_names();
                info!(?names, "Found these jobs")
            }
            Err(err) => return Err(GlueMvpError::from_glue_sdk(err)),
        }
    }
```

- Untuk detail API, lihat ListJobsreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan `StartCrawler` dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`StartCrawler`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
/// <summary>
/// Start an AWS Glue crawler.
/// </summary>
/// <param name="crawlerName">The name of the crawler.</param>
/// <returns>A Boolean value indicating the success of the action.</returns>
public async Task<bool> StartCrawlerAsync(string crawlerName)
{
    var crawlerRequest = new StartCrawlerRequest
    {
        Name = crawlerName,
    };

    var response = await _amazonGlue.StartCrawlerAsync(crawlerRequest);

    return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
}
```

- Untuk detail API, lihat StartCrawlerdi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
       Aws::Client::ClientConfiguration clientConfig;
       // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
       // clientConfig.region = "us-east-1";

   Aws::Glue::GlueClient client(clientConfig);

       Aws::Glue::Model::StartCrawlerRequest request;
       request.SetName(CRAWLER_NAME);

       Aws::Glue::Model::StartCrawlerOutcome outcome =
client.StartCrawler(request);


       if (outcome.IsSuccess() || (Aws::Glue::GlueErrors::CRAWLER_RUNNING ==
                                   outcome.GetError().GetErrorType())) {
           if (!outcome.IsSuccess()) {
               std::cout << "Crawler was already started." << std::endl;
           }
           else {
               std::cout << "Successfully started crawler." << std::endl;
           }

           std::cout << "This may take a while to run." << std::endl;

           Aws::Glue::Model::CrawlerState crawlerState =
Aws::Glue::Model::CrawlerState::NOT_SET;
           int iterations = 0;
           while (Aws::Glue::Model::CrawlerState::READY != crawlerState) {
               std::this_thread::sleep_for(std::chrono::seconds(1));
```

```
                ++iterations;
                if ((iterations % 10) == 0) { // Log status every 10 seconds.
                    std::cout << "Crawler status " <<

Aws::Glue::Model::CrawlerStateMapper::GetNameForCrawlerState(
                                    crawlerState)
                              << ". After " << iterations
                              << " seconds elapsed."
                              << std::endl;
                }
                Aws::Glue::Model::GetCrawlerRequest getCrawlerRequest;
                getCrawlerRequest.SetName(CRAWLER_NAME);

                Aws::Glue::Model::GetCrawlerOutcome getCrawlerOutcome =
client.GetCrawler(
                        getCrawlerRequest);

                if (getCrawlerOutcome.IsSuccess()) {
                    crawlerState =
getCrawlerOutcome.GetResult().GetCrawler().GetState();
                }
                else {
                    std::cerr << "Error getting crawler.  "
                              << getCrawlerOutcome.GetError().GetMessage() <<
std::endl;
                    break;
                }
            }

            if (Aws::Glue::Model::CrawlerState::READY == crawlerState) {
                std::cout << "Crawler finished running after " << iterations
                          << " seconds."
                          << std::endl;
            }
        }
        else {
            std::cerr << "Error starting a crawler.  "
                      << outcome.GetError().GetMessage()
                      << std::endl;

            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
```

```
```

- Untuk detail API, lihat StartCrawlerdi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Untuk memulai crawler

`start-crawler`Contoh berikut memulai crawler.

```
aws glue start-crawler --name my-crawler
```

Output:

```
None
```

Untuk informasi selengkapnya, lihat Mendefinisikan Crawler di Panduan Pengembang AWS Glue.

- Untuk detail API, lihat StartCrawlerdi Referensi AWS CLI Perintah.

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.glue.GlueClient;
import software.amazon.awssdk.services.glue.model.GlueException;
import software.amazon.awssdk.services.glue.model.StartCrawlerRequest;

/**
 * Before running this Java V2 code example, set up your development
```

```
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class StartCrawler {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <crawlerName>

                Where:
                    crawlerName - The name of the crawler.\s
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String crawlerName = args[0];
        Region region = Region.US_EAST_1;
        GlueClient glueClient = GlueClient.builder()
                .region(region)
                .build();

        startSpecificCrawler(glueClient, crawlerName);
        glueClient.close();
    }

    public static void startSpecificCrawler(GlueClient glueClient, String
 crawlerName) {
        try {
            StartCrawlerRequest crawlerRequest = StartCrawlerRequest.builder()
                    .name(crawlerName)
                    .build();

            glueClient.startCrawler(crawlerRequest);

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat StartCrawlerdi Referensi AWS SDK for Java 2.x API.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const startCrawler = (name) => {
  const client = new GlueClient({});

  const command = new StartCrawlerCommand({
    Name: name,
  });

  return client.send(command);
};
```

- Untuk detail API, lihat StartCrawlerdi Referensi AWS SDK for JavaScript API.

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun startSpecificCrawler(crawlerName: String?) {
    val request =
        StartCrawlerRequest {
            name = crawlerName
        }

    GlueClient { region = "us-west-2" }.use { glueClient ->
        glueClient.startCrawler(request)
        println("$crawlerName was successfully started.")
    }
}
```

- Untuk detail API, lihat StartCrawlerdi AWS SDK untuk referensi API Kotlin.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        $crawlerName = "example-crawler-test-" . $uniqid;

        $databaseName = "doc-example-database-$uniqid";

        $glueService->startCrawler($crawlerName);

    public function startCrawler($crawlerName): Result
    {
        return $this->glueClient->startCrawler([
            'Name' => $crawlerName,
        ]);
    }
```

- Untuk detail API, lihat StartCrawlerdi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def start_crawler(self, name):
        """
        Starts a crawler. The crawler crawls its configured target and creates
        metadata that describes the data it finds in the target data source.

        :param name: The name of the crawler to start.
        """
        try:
            self.glue_client.start_crawler(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't start crawler %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

- Untuk detail API, lihat StartCrawlerdi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Starts a crawler with the specified name.
  #
  # @param name [String] The name of the crawler to start.
  # @return [void]
  def start_crawler(name)
    @glue_client.start_crawler(name: name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not start crawler #{name}: \n#{e.message}")
    raise
  end
```

- Untuk detail API, lihat StartCrawlerdi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        let start_crawler =
glue.start_crawler().name(self.crawler()).send().await;

        match start_crawler {
            Ok(_) => Ok(()),
            Err(err) => {
                let glue_err: aws_sdk_glue::Error = err.into();
                match glue_err {
                    aws_sdk_glue::Error::CrawlerRunningException(_) => Ok(()),
                    _ => Err(GlueMvpError::GlueSdk(glue_err)),
                }
            }
        }?;
```

- Untuk detail API, lihat StartCrawlerreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Gunakan **`StartJobRun`** dengan AWS SDK atau CLI

Contoh kode berikut menunjukkan cara menggunakan`StartJobRun`.

Contoh tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Anda dapat melihat tindakan ini dalam konteks dalam contoh kode berikut:

- Memulai crawler dan lowongan kerja

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
/// <summary>
/// Start an AWS Glue job run.
/// </summary>
/// <param name="jobName">The name of the job.</param>
/// <returns>A string representing the job run Id.</returns>
public async Task<string> StartJobRunAsync(
    string jobName,
    string inputDatabase,
    string inputTable,
    string bucketName)
{
    var request = new StartJobRunRequest
    {
        JobName = jobName,
        Arguments = new Dictionary<string, string>
        {
            {"--input_database", inputDatabase},
            {"--input_table", inputTable},
            {"--output_bucket_url", $"s3://{bucketName}/"}
        }
    };

    var response = await _amazonGlue.StartJobRunAsync(request);
    return response.JobRunId;
}
```

- Untuk detail API, lihat StartJobRundi Referensi AWS SDK for .NET API.

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region in which the bucket was created
(overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::Glue::GlueClient client(clientConfig);

        Aws::Glue::Model::StartJobRunRequest request;
        request.SetJobName(JOB_NAME);

        Aws::Map<Aws::String, Aws::String> arguments;
        arguments["--input_database"] = CRAWLER_DATABASE_NAME;
        arguments["--input_table"] = tableName;
        arguments["--output_bucket_url"] = Aws::String("s3://") + bucketName +
"/";
        request.SetArguments(arguments);

        Aws::Glue::Model::StartJobRunOutcome outcome =
client.StartJobRun(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully started the job." << std::endl;

            Aws::String jobRunId = outcome.GetResult().GetJobRunId();

            int iterator = 0;
            bool done = false;
            while (!done) {
                ++iterator;
                std::this_thread::sleep_for(std::chrono::seconds(1));
                Aws::Glue::Model::GetJobRunRequest jobRunRequest;
                jobRunRequest.SetJobName(JOB_NAME);
```

```
                jobRunRequest.SetRunId(jobRunId);

                Aws::Glue::Model::GetJobRunOutcome jobRunOutcome =
client.GetJobRun(
                        jobRunRequest);

                if (jobRunOutcome.IsSuccess()) {
                    const Aws::Glue::Model::JobRun &jobRun =
jobRunOutcome.GetResult().GetJobRun();
                    Aws::Glue::Model::JobRunState jobRunState =
jobRun.GetJobRunState();

                    if ((jobRunState == Aws::Glue::Model::JobRunState::STOPPED)
||
                        (jobRunState == Aws::Glue::Model::JobRunState::FAILED) ||
                        (jobRunState == Aws::Glue::Model::JobRunState::TIMEOUT))
{
                        std::cerr << "Error running job. "
                                  << jobRun.GetErrorMessage()
                                  << std::endl;
                        deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME,
JOB_NAME,
                                     bucketName,
                                     clientConfig);
                        return false;
                    }
                    else if (jobRunState ==
                             Aws::Glue::Model::JobRunState::SUCCEEDED) {
                        std::cout << "Job run succeeded after  " << iterator <<
                                  " seconds elapsed." << std::endl;
                        done = true;
                    }
                    else if ((iterator % 10) == 0) { // Log status every 10
seconds.
                        std::cout << "Job run status " <<

Aws::Glue::Model::JobRunStateMapper::GetNameForJobRunState(
                                         jobRunState) <<
                                  ". " << iterator <<
                                  " seconds elapsed." << std::endl;
                    }
                }
                else {
                    std::cerr << "Error retrieving job run state. "
```

```
                            << jobRunOutcome.GetError().GetMessage()
                            << std::endl;
                    deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
                                 bucketName, clientConfig);
                    return false;
                }
            }
        }
        else {
            std::cerr << "Error starting a job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
bucketName,
                         clientConfig);
            return false;
        }
```

- Untuk detail API, lihat StartJobRundi Referensi AWS SDK for C++ API.

CLI

AWS CLI

Untuk mulai menjalankan pekerjaan

`start-job-run`Contoh berikut memulai pekerjaan.

```
aws glue start-job-run \
    --job-name my-job
```

Output:

```
{
    "JobRunId":
 "jr_22208b1f44eb5376a60569d4b21dd20fcb8621e1a366b4e7b2494af764b82ded"
}
```

Untuk informasi selengkapnya, lihat Menulis Pekerjaan di Panduan Pengembang AWS Glue.

- Untuk detail API, lihat StartJobRundi Referensi AWS CLI Perintah.

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
const startJobRun = (jobName, dbName, tableName, bucketName) => {
  const client = new GlueClient({});

  const command = new StartJobRunCommand({
    JobName: jobName,
    Arguments: {
      "--input_database": dbName,
      "--input_table": tableName,
      "--output_bucket_url": `s3://${bucketName}/`,
    },
  });

  return client.send(command);
};
```

- Untuk detail API, lihat StartJobRundi Referensi AWS SDK for JavaScript API.

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        $jobName = 'test-job-' . $uniqid;
```

```
        $databaseName = "doc-example-database-$uniqid";

        $tables = $glueService->getTables($databaseName);

        $outputBucketUrl = "s3://$bucketName";
        $runId = $glueService->startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl)['JobRunId'];

   public function startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl): Result
   {
       return $this->glueClient->startJobRun([
           'JobName' => $jobName,
           'Arguments' => [
               'input_database' => $databaseName,
               'input_table' => $tables['TableList'][0]['Name'],
               'output_bucket_url' => $outputBucketUrl,
               '--input_database' => $databaseName,
               '--input_table' => $tables['TableList'][0]['Name'],
               '--output_bucket_url' => $outputBucketUrl,
           ],
       ]);
   }
```

- Untuk detail API, lihat StartJobRundi Referensi AWS SDK for PHP API.

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
```

```
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client


    def start_job_run(self, name, input_database, input_table,
output_bucket_name):
        """
        Starts a job run. A job run extracts data from the source, transforms it,
        and loads it to the output bucket.

        :param name: The name of the job definition.
        :param input_database: The name of the metadata database that contains
tables
                               that describe the source data. This is typically
created
                               by a crawler.
        :param input_table: The name of the table in the metadata database that
                            describes the source data.
        :param output_bucket_name: The S3 bucket where the output is written.
        :return: The ID of the job run.
        """
        try:
            # The custom Arguments that are passed to this function are used by
the
            # Python ETL script to determine the location of input and output
data.
            response = self.glue_client.start_job_run(
                JobName=name,
                Arguments={
                    "--input_database": input_database,
                    "--input_table": input_table,
                    "--output_bucket_url": f"s3://{output_bucket_name}/",
                },
            )
        except ClientError as err:
            logger.error(
                "Couldn't start job run %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
```

```
            return response["JobRunId"]
```

- Untuk detail API, lihat StartJobRundi AWS SDK for Python (Boto3) Referensi API.

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Starts a job run for the specified job.
  #
  # @param name [String] The name of the job to start the run for.
  # @param input_database [String] The name of the input database for the job.
  # @param input_table [String] The name of the input table for the job.
  # @param output_bucket_name [String] The name of the output S3 bucket for the
 job.
  # @return [String] The ID of the started job run.
  def start_job_run(name, input_database, input_table, output_bucket_name)
    response = @glue_client.start_job_run(
      job_name: name,
      arguments: {
        '--input_database': input_database,
```

```
        '--input_table': input_table,
        '--output_bucket_url': "s3://#{output_bucket_name}/"
      }
    )
    response.job_run_id
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not start job run #{name}: \n#{e.message}")
    raise
  end
```

- Untuk detail API, lihat StartJobRundi Referensi AWS SDK for Ruby API.

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
        let job_run_output = glue
            .start_job_run()
            .job_name(self.job())
            .arguments("--input_database", self.database())
            .arguments(
                "--input_table",
                self.tables
                    .first()
                    .ok_or_else(|| GlueMvpError::Unknown("Missing crawler
table".into()))?
                    .name(),
            )
            .arguments("--output_bucket_url", self.bucket())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let job = job_run_output
            .job_run_id()
```

```
            .ok_or_else(|| GlueMvpError::Unknown("Missing run id from just
started job".into()))?
            .to_string();
```

- Untuk detail API, lihat StartJobRunreferensi AWS SDK for Rust API.

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Skenario untuk AWS Glue menggunakan AWS SDK

Contoh kode berikut menunjukkan cara menerapkan skenario umum AWS Glue dengan AWS SDK. Skenario ini menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi di dalamnya AWS Glue. Setiap skenario menyertakan tautan ke GitHub, di mana Anda dapat menemukan petunjuk tentang cara mengatur dan menjalankan kode.

Contoh

- Mulai menjalankan AWS Glue crawler dan lowongan kerja menggunakan SDK AWS

Mulai menjalankan AWS Glue crawler dan lowongan kerja menggunakan SDK AWS

Contoh kode berikut ini menunjukkan cara:

- Buat crawler yang merayapi bucket Amazon S3 publik dan membuat database metadata berformat CSV.
- Daftar informasi tentang database dan tabel di situs Anda AWS Glue Data Catalog.
- Buat pekerjaan untuk mengekstrak data CSV dari bucket S3, mengubah data, dan memuat output berformat JSON ke bucket S3 lain.
- Buat daftar informasi tentang menjalankan pekerjaan, melihat data yang diubah, dan membersihkan sumber daya.

Untuk informasi selengkapnya, lihat Tutorial: Memulai AWS Glue Studio.

.NET

AWS SDK for .NET

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat kelas yang membungkus AWS Glue fungsi yang digunakan dalam skenario.

```
using System.Net;

namespace GlueActions;

public class GlueWrapper
{
    private readonly IAmazonGlue _amazonGlue;

    /// <summary>
    /// Constructor for the AWS Glue actions wrapper.
    /// </summary>
    /// <param name="amazonGlue"></param>
    public GlueWrapper(IAmazonGlue amazonGlue)
    {
        _amazonGlue = amazonGlue;
    }

    /// <summary>
    /// Create an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name for the crawler.</param>
    /// <param name="crawlerDescription">A description of the crawler.</param>
    /// <param name="role">The AWS Identity and Access Management (IAM) role to
    /// be assumed by the crawler.</param>
    /// <param name="schedule">The schedule on which the crawler will be
 executed.</param>
    /// <param name="s3Path">The path to the Amazon Simple Storage Service
 (Amazon S3)
    /// bucket where the Python script has been stored.</param>
    /// <param name="dbName">The name to use for the database that will be
```

```
    /// created by the crawler.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> CreateCrawlerAsync(
        string crawlerName,
        string crawlerDescription,
        string role,
        string schedule,
        string s3Path,
        string dbName)
    {
        var s3Target = new S3Target
        {
            Path = s3Path,
        };

        var targetList = new List<S3Target>
        {
            s3Target,
        };

        var targets = new CrawlerTargets
        {
            S3Targets = targetList,
        };

        var crawlerRequest = new CreateCrawlerRequest
        {
            DatabaseName = dbName,
            Name = crawlerName,
            Description = crawlerDescription,
            Targets = targets,
            Role = role,
            Schedule = schedule,
        };

        var response = await _amazonGlue.CreateCrawlerAsync(crawlerRequest);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Create an AWS Glue job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
```

```
    /// <param name="roleName">The name of the IAM role to be assumed by
    /// the job.</param>
    /// <param name="description">A description of the job.</param>
    /// <param name="scriptUrl">The URL to the script.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> CreateJobAsync(string dbName, string tableName,
string bucketUrl, string jobName, string roleName, string description, string
scriptUrl)
    {
        var command = new JobCommand
        {
            PythonVersion = "3",
            Name = "glueetl",
            ScriptLocation = scriptUrl,
        };

        var arguments = new Dictionary<string, string>
        {
            { "--input_database", dbName },
            { "--input_table", tableName },
            { "--output_bucket_url", bucketUrl }
        };

        var request = new CreateJobRequest
        {
            Command = command,
            DefaultArguments = arguments,
            Description = description,
            GlueVersion = "3.0",
            Name = jobName,
            NumberOfWorkers = 10,
            Role = roleName,
            WorkerType = "G.1X"
        };

        var response = await _amazonGlue.CreateJobAsync(request);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteCrawlerAsync(string crawlerName)
    {
        var response = await _amazonGlue.DeleteCrawlerAsync(new
DeleteCrawlerRequest { Name = crawlerName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete the AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteDatabaseAsync(string dbName)
    {
        var response = await _amazonGlue.DeleteDatabaseAsync(new
DeleteDatabaseRequest { Name = dbName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an AWS Glue job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteJobAsync(string jobName)
    {
        var response = await _amazonGlue.DeleteJobAsync(new DeleteJobRequest
{ JobName = jobName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete a table from an AWS Glue database.
    /// </summary>
    /// <param name="tableName">The table to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteTableAsync(string dbName, string tableName)
    {
        var response = await _amazonGlue.DeleteTableAsync(new DeleteTableRequest
{ Name = tableName, DatabaseName = dbName });
```

```
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Get information about an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A Crawler object describing the crawler.</returns>
    public async Task<Crawler?> GetCrawlerAsync(string crawlerName)
    {
        var crawlerRequest = new GetCrawlerRequest
        {
            Name = crawlerName,
        };

        var response = await _amazonGlue.GetCrawlerAsync(crawlerRequest);
        if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
        {
            var databaseName = response.Crawler.DatabaseName;
            Console.WriteLine($"{crawlerName} has the database {databaseName}");
            return response.Crawler;
        }

        Console.WriteLine($"No information regarding {crawlerName} could be
found.");
        return null;
    }


    /// <summary>
    /// Get information about the state of an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A value describing the state of the crawler.</returns>
    public async Task<CrawlerState> GetCrawlerStateAsync(string crawlerName)
    {
        var response = await _amazonGlue.GetCrawlerAsync(
            new GetCrawlerRequest { Name = crawlerName });
        return response.Crawler.State;
    }


    /// <summary>
```

```
    /// Get information about an AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A Database object containing information about the database.</
returns>
    public async Task<Database> GetDatabaseAsync(string dbName)
    {
        var databasesRequest = new GetDatabaseRequest
        {
            Name = dbName,
        };

        var response = await _amazonGlue.GetDatabaseAsync(databasesRequest);
        return response.Database;
    }


    /// <summary>
    /// Get information about a specific AWS Glue job run.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <param name="jobRunId">The Id of the job run.</param>
    /// <returns>A JobRun object with information about the job run.</returns>
    public async Task<JobRun> GetJobRunAsync(string jobName, string jobRunId)
    {
        var response = await _amazonGlue.GetJobRunAsync(new GetJobRunRequest
 { JobName = jobName, RunId = jobRunId });
        return response.JobRun;
    }


    /// <summary>
    /// Get information about all AWS Glue runs of a specific job.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A list of JobRun objects.</returns>
    public async Task<List<JobRun>> GetJobRunsAsync(string jobName)
    {
        var jobRuns = new List<JobRun>();

        var request = new GetJobRunsRequest
        {
            JobName = jobName,
        };
```

```
        // No need to loop to get all the log groups--the SDK does it for us
behind the scenes
        var paginatorForJobRuns =
            _amazonGlue.Paginators.GetJobRuns(request);

        await foreach (var response in paginatorForJobRuns.Responses)
        {
            response.JobRuns.ForEach(jobRun =>
            {
                jobRuns.Add(jobRun);
            });
        }

        return jobRuns;
    }


    /// <summary>
    /// Get a list of tables for an AWS Glue database.
    /// </summary>
    /// <param name="dbName">The name of the database.</param>
    /// <returns>A list of Table objects.</returns>
    public async Task<List<Table>> GetTablesAsync(string dbName)
    {
        var request = new GetTablesRequest { DatabaseName = dbName };
        var tables = new List<Table>();

        // Get a paginator for listing the tables.
        var tablePaginator = _amazonGlue.Paginators.GetTables(request);

        await foreach (var response in tablePaginator.Responses)
        {
            tables.AddRange(response.TableList);
        }

        return tables;
    }


    /// <summary>
    /// List AWS Glue jobs using a paginator.
    /// </summary>
    /// <returns>A list of AWS Glue job names.</returns>
```

```
    public async Task<List<string>> ListJobsAsync()
    {
        var jobNames = new List<string>();

        var listJobsPaginator = _amazonGlue.Paginators.ListJobs(new
ListJobsRequest { MaxResults = 10 });
        await foreach (var response in listJobsPaginator.Responses)
        {
            jobNames.AddRange(response.JobNames);
        }

        return jobNames;
    }


    /// <summary>
    /// Start an AWS Glue crawler.
    /// </summary>
    /// <param name="crawlerName">The name of the crawler.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> StartCrawlerAsync(string crawlerName)
    {
        var crawlerRequest = new StartCrawlerRequest
        {
            Name = crawlerName,
        };

        var response = await _amazonGlue.StartCrawlerAsync(crawlerRequest);

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <summary>
    /// Start an AWS Glue job run.
    /// </summary>
    /// <param name="jobName">The name of the job.</param>
    /// <returns>A string representing the job run Id.</returns>
    public async Task<string> StartJobRunAsync(
        string jobName,
        string inputDatabase,
        string inputTable,
        string bucketName)
    {
```

```
        var request = new StartJobRunRequest
        {
            JobName = jobName,
            Arguments = new Dictionary<string, string>
            {
                {"--input_database", inputDatabase},
                {"--input_table", inputTable},
                {"--output_bucket_url", $"s3://{bucketName}/"}
            }
        };

        var response = await _amazonGlue.StartJobRunAsync(request);
        return response.JobRunId;
    }

}
```

Buat kelas yang menjalankan skenario.

```
global using Amazon.Glue;
global using GlueActions;
global using Microsoft.Extensions.Configuration;
global using Microsoft.Extensions.DependencyInjection;
global using Microsoft.Extensions.Hosting;
global using Microsoft.Extensions.Logging;
global using Microsoft.Extensions.Logging.Console;
global using Microsoft.Extensions.Logging.Debug;



using Amazon.Glue.Model;
using Amazon.S3;
using Amazon.S3.Model;

namespace GlueBasics;

public class GlueBasics
{
    private static ILogger logger = null!;
    private static IConfiguration _configuration = null!;
```

```
   static async Task Main(string[] args)
   {
       // Set up dependency injection for AWS Glue.
       using var host = Host.CreateDefaultBuilder(args)
           .ConfigureLogging(logging =>
               logging.AddFilter("System", LogLevel.Debug)
                   .AddFilter<DebugLoggerProvider>("Microsoft",
LogLevel.Information)
                   .AddFilter<ConsoleLoggerProvider>("Microsoft",
LogLevel.Trace))
           .ConfigureServices((_, services) =>
           services.AddAWSService<IAmazonGlue>()
           .AddTransient<GlueWrapper>()
           .AddTransient<UiWrapper>()
           )
           .Build();

       logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
       .CreateLogger<GlueBasics>();

       _configuration = new ConfigurationBuilder()
           .SetBasePath(Directory.GetCurrentDirectory())
           .AddJsonFile("settings.json") // Load settings from .json file.
           .AddJsonFile("settings.local.json",
               true) // Optionally load local settings.
           .Build();

       // These values are stored in settings.json
       // Once you have run the CDK script to deploy the resources,
       // edit the file to set "BucketName", "RoleName", and "ScriptURL"
       // to the appropriate values. Also set "CrawlerName" to the name
       // you want to give the crawler when it is created.
       string bucketName = _configuration["BucketName"]!;
       string bucketUrl = _configuration["BucketUrl"]!;
       string crawlerName = _configuration["CrawlerName"]!;
       string roleName = _configuration["RoleName"]!;
       string sourceData = _configuration["SourceData"]!;
       string dbName = _configuration["DbName"]!;
       string cron = _configuration["Cron"]!;
       string scriptUrl = _configuration["ScriptURL"]!;
       string jobName = _configuration["JobName"]!;

       var wrapper = host.Services.GetRequiredService<GlueWrapper>();
       var uiWrapper = host.Services.GetRequiredService<UiWrapper>();
```

```
        uiWrapper.DisplayOverview();
        uiWrapper.PressEnter();

        // Create the crawler and wait for it to be ready.
        uiWrapper.DisplayTitle("Create AWS Glue crawler");
        Console.WriteLine("Let's begin by creating the AWS Glue crawler.");

        var crawlerDescription = "Crawler created for the AWS Glue Basics
scenario.";
        var crawlerCreated = await wrapper.CreateCrawlerAsync(crawlerName,
crawlerDescription, roleName, cron, sourceData, dbName);
        if (crawlerCreated)
        {
            Console.WriteLine($"The crawler: {crawlerName} has been created. Now
let's wait until it's ready.");
            CrawlerState crawlerState;
            do
            {
                crawlerState = await wrapper.GetCrawlerStateAsync(crawlerName);
            }
            while (crawlerState != "READY");
            Console.WriteLine($"The crawler {crawlerName} is now ready for
use.");
        }
        else
        {
            Console.WriteLine($"Couldn't create crawler {crawlerName}.");
            return; // Exit the application.
        }

        uiWrapper.DisplayTitle("Start AWS Glue crawler");
        Console.WriteLine("Now let's wait until the crawler has successfully
started.");
        var crawlerStarted = await wrapper.StartCrawlerAsync(crawlerName);
        if (crawlerStarted)
        {
            CrawlerState crawlerState;
            do
            {
                crawlerState = await wrapper.GetCrawlerStateAsync(crawlerName);
            }
            while (crawlerState != "READY");
```

```
            Console.WriteLine($"The crawler {crawlerName} is now ready for
use.");
        }
        else
        {
            Console.WriteLine($"Couldn't start the crawler {crawlerName}.");
            return; // Exit the application.
        }

        uiWrapper.PressEnter();

        Console.WriteLine($"\nLet's take a look at the database: {dbName}");
        var database = await wrapper.GetDatabaseAsync(dbName);

        if (database != null)
        {
            uiWrapper.DisplayTitle($"{database.Name} Details");
            Console.WriteLine($"{database.Name} created on
{database.CreateTime}");
            Console.WriteLine(database.Description);
        }

        uiWrapper.PressEnter();

        var tables = await wrapper.GetTablesAsync(dbName);
        if (tables.Count > 0)
        {
            tables.ForEach(table =>
            {
                Console.WriteLine($"{table.Name}\tCreated:
{table.CreateTime}\tUpdated: {table.UpdateTime}");
            });
        }

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Create AWS Glue job");
        Console.WriteLine("Creating a new AWS Glue job.");
        var description = "An AWS Glue job created using the AWS SDK for .NET";
        await wrapper.CreateJobAsync(dbName, tables[0].Name, bucketUrl, jobName,
roleName, description, scriptUrl);

        uiWrapper.PressEnter();
```

```
        uiWrapper.DisplayTitle("Starting AWS Glue job");
        Console.WriteLine("Starting the new AWS Glue job...");
        var jobRunId = await wrapper.StartJobRunAsync(jobName, dbName,
tables[0].Name, bucketName);
        var jobRunComplete = false;
        var jobRun = new JobRun();
        do
        {
            jobRun = await wrapper.GetJobRunAsync(jobName, jobRunId);
            if (jobRun.JobRunState == "SUCCEEDED" || jobRun.JobRunState ==
"STOPPED" ||
                jobRun.JobRunState == "FAILED" || jobRun.JobRunState ==
"TIMEOUT")
            {
                jobRunComplete = true;
            }
        } while (!jobRunComplete);

        uiWrapper.DisplayTitle($"Data in {bucketName}");

        // Get the list of data stored in the S3 bucket.
        var s3Client = new AmazonS3Client();

        var response = await s3Client.ListObjectsAsync(new ListObjectsRequest
{ BucketName = bucketName });
        response.S3Objects.ForEach(s3Object =>
        {
            Console.WriteLine(s3Object.Key);
        });

        uiWrapper.DisplayTitle("AWS Glue jobs");
        var jobNames = await wrapper.ListJobsAsync();
        jobNames.ForEach(jobName =>
        {
            Console.WriteLine(jobName);
        });

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Get AWS Glue job run information");
        Console.WriteLine("Getting information about the AWS Glue job.");
        var jobRuns = await wrapper.GetJobRunsAsync(jobName);

        jobRuns.ForEach(jobRun =>
```

```
        {

 Console.WriteLine($"{jobRun.JobName}\t{jobRun.JobRunState}\t{jobRun.CompletedOn}");
        });

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Deleting resources");
        Console.WriteLine("Deleting the AWS Glue job used by the example.");
        await wrapper.DeleteJobAsync(jobName);

        Console.WriteLine("Deleting the tables from the database.");
        tables.ForEach(async table =>
        {
            await wrapper.DeleteTableAsync(dbName, table.Name);
        });

        Console.WriteLine("Deleting the database.");
        await wrapper.DeleteDatabaseAsync(dbName);

        Console.WriteLine("Deleting the AWS Glue crawler.");
        await wrapper.DeleteCrawlerAsync(crawlerName);

        Console.WriteLine("The AWS Glue scenario has completed.");
        uiWrapper.PressEnter();
    }
}


namespace GlueBasics;

public class UiWrapper
{
    public readonly string SepBar = new string('-', Console.WindowWidth);

    /// <summary>
    /// Show information about the scenario.
    /// </summary>
    public void DisplayOverview()
    {
        Console.Clear();
        DisplayTitle("Amazon Glue: get started with crawlers and jobs");

        Console.WriteLine("This example application does the following:");
```

```
        Console.WriteLine("\t 1. Create a crawler, pass it the IAM role and the
URL to the public S3 bucket that contains the source data");
        Console.WriteLine("\t 2. Start the crawler.");
        Console.WriteLine("\t 3. Get the database created by the crawler and the
tables in the database.");
        Console.WriteLine("\t 4. Create a job.");
        Console.WriteLine("\t 5. Start a job run.");
        Console.WriteLine("\t 6. Wait for the job run to complete.");
        Console.WriteLine("\t 7. Show the data stored in the bucket.");
        Console.WriteLine("\t 8. List jobs for the account.");
        Console.WriteLine("\t 9. Get job run details for the job that was run.");
        Console.WriteLine("\t10. Delete the demo job.");
        Console.WriteLine("\t11. Delete the database and tables created for the
demo.");
        Console.WriteLine("\t12. Delete the crawler.");
    }

    /// <summary>
    /// Display a message and wait until the user presses enter.
    /// </summary>
    public void PressEnter()
    {
        Console.Write("\nPlease press <Enter> to continue. ");
        _ = Console.ReadLine();
    }

    /// <summary>
    /// Pad a string with spaces to center it on the console display.
    /// </summary>
    /// <param name="strToCenter">The string to center on the screen.</param>
    /// <returns>The string padded to make it center on the screen.</returns>
    public string CenterString(string strToCenter)
    {
        var padAmount = (Console.WindowWidth - strToCenter.Length) / 2;
        var leftPad = new string(' ', padAmount);
        return $"{leftPad}{strToCenter}";
    }

    /// <summary>
    /// Display a line of hyphens, the centered text of the title and another
    /// line of hyphens.
    /// </summary>
    /// <param name="strTitle">The string to be displayed.</param>
    public void DisplayTitle(string strTitle)
```

```
    {
        Console.WriteLine(SepBar);
        Console.WriteLine(CenterString(strTitle));
        Console.WriteLine(SepBar);
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for .NET .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler
  - StartJobRun

C++

SDK untuk C++

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
//! Scenario which demonstrates using AWS Glue to add a crawler and run a job.
/*!
 \\sa runGettingStartedWithGlueScenario()
 \param bucketName: An S3 bucket created in the setup.
 \param roleName: An AWS Identity and Access Management (IAM) role created in the
 setup.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */

bool AwsDoc::Glue::runGettingStartedWithGlueScenario(const Aws::String
 &bucketName,
                                                      const Aws::String &roleName,
                                                      const
 Aws::Client::ClientConfiguration &clientConfig) {
    Aws::Glue::GlueClient client(clientConfig);

    Aws::String roleArn;
    if (!getRoleArn(roleName, roleArn, clientConfig)) {
        std::cerr << "Error getting role ARN for role." << std::endl;
        return false;
    }

    // 1. Upload the job script to the S3 bucket.
    {
        std::cout << "Uploading the job script '"
                  << AwsDoc::Glue::PYTHON_SCRIPT
                  << "'." << std::endl;

        if (!AwsDoc::Glue::uploadFile(bucketName,
                                      AwsDoc::Glue::PYTHON_SCRIPT_PATH,
                                      AwsDoc::Glue::PYTHON_SCRIPT,
```

```
                                        clientConfig)) {
            std::cerr << "Error uploading the job file." << std::endl;
            return false;
        }
    }

    // 2. Create a crawler.
    {
        Aws::Glue::Model::S3Target s3Target;
        s3Target.SetPath("s3://crawler-public-us-east-1/flight/2016/csv");
        Aws::Glue::Model::CrawlerTargets crawlerTargets;
        crawlerTargets.AddS3Targets(s3Target);

        Aws::Glue::Model::CreateCrawlerRequest request;
        request.SetTargets(crawlerTargets);
        request.SetName(CRAWLER_NAME);
        request.SetDatabaseName(CRAWLER_DATABASE_NAME);
        request.SetTablePrefix(CRAWLER_DATABASE_PREFIX);
        request.SetRole(roleArn);

        Aws::Glue::Model::CreateCrawlerOutcome outcome =
client.CreateCrawler(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully created the crawler." << std::endl;
        }
        else {
            std::cerr << "Error creating a crawler. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets("", CRAWLER_DATABASE_NAME, "", bucketName,
clientConfig);
            return false;
        }
    }

    // 3. Get a crawler.
    {
        Aws::Glue::Model::GetCrawlerRequest request;
        request.SetName(CRAWLER_NAME);

        Aws::Glue::Model::GetCrawlerOutcome outcome = client.GetCrawler(request);

        if (outcome.IsSuccess()) {
```

```
            Aws::Glue::Model::CrawlerState crawlerState =
outcome.GetResult().GetCrawler().GetState();
            std::cout << "Retrieved crawler with state " <<

Aws::Glue::Model::CrawlerStateMapper::GetNameForCrawlerState(
                              crawlerState)
                      << "." << std::endl;
        }
        else {
            std::cerr << "Error retrieving a crawler.  "
                      << outcome.GetError().GetMessage() << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
    }

    // 4. Start a crawler.
    {
        Aws::Glue::Model::StartCrawlerRequest request;
        request.SetName(CRAWLER_NAME);

        Aws::Glue::Model::StartCrawlerOutcome outcome =
client.StartCrawler(request);


        if (outcome.IsSuccess() || (Aws::Glue::GlueErrors::CRAWLER_RUNNING ==
                                    outcome.GetError().GetErrorType())) {
            if (!outcome.IsSuccess()) {
                std::cout << "Crawler was already started." << std::endl;
            }
            else {
                std::cout << "Successfully started crawler." << std::endl;
            }

            std::cout << "This may take a while to run." << std::endl;

            Aws::Glue::Model::CrawlerState crawlerState =
Aws::Glue::Model::CrawlerState::NOT_SET;
            int iterations = 0;
            while (Aws::Glue::Model::CrawlerState::READY != crawlerState) {
                std::this_thread::sleep_for(std::chrono::seconds(1));
                ++iterations;
                if ((iterations % 10) == 0) { // Log status every 10 seconds.
```

```
                    std::cout << "Crawler status " <<

Aws::Glue::Model::CrawlerStateMapper::GetNameForCrawlerState(
                                      crawlerState)
                              << ". After " << iterations
                              << " seconds elapsed."
                              << std::endl;
                }
                Aws::Glue::Model::GetCrawlerRequest getCrawlerRequest;
                getCrawlerRequest.SetName(CRAWLER_NAME);

                Aws::Glue::Model::GetCrawlerOutcome getCrawlerOutcome =
client.GetCrawler(
                        getCrawlerRequest);

                if (getCrawlerOutcome.IsSuccess()) {
                    crawlerState =
getCrawlerOutcome.GetResult().GetCrawler().GetState();
                }
                else {
                    std::cerr << "Error getting crawler.  "
                              << getCrawlerOutcome.GetError().GetMessage() <<
std::endl;
                    break;
                }
            }

            if (Aws::Glue::Model::CrawlerState::READY == crawlerState) {
                std::cout << "Crawler finished running after " << iterations
                          << " seconds."
                          << std::endl;
            }
        }
        else {
            std::cerr << "Error starting a crawler.  "
                      << outcome.GetError().GetMessage()
                      << std::endl;

            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
    }
```

```
   // 5. Get a database.
   {
       Aws::Glue::Model::GetDatabaseRequest request;
       request.SetName(CRAWLER_DATABASE_NAME);

       Aws::Glue::Model::GetDatabaseOutcome outcome =
client.GetDatabase(request);

       if (outcome.IsSuccess()) {
           const Aws::Glue::Model::Database &database =
outcome.GetResult().GetDatabase();

           std::cout << "Successfully retrieve the database\n" <<
                     database.Jsonize().View().WriteReadable() << "'." <<
std::endl;
       }
       else {
           std::cerr << "Error getting the database.  "
                     << outcome.GetError().GetMessage() << std::endl;
           deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                        clientConfig);
           return false;
       }
   }

   // 6. Get tables.
   Aws::String tableName;
   {
       Aws::Glue::Model::GetTablesRequest request;
       request.SetDatabaseName(CRAWLER_DATABASE_NAME);
       std::vector<Aws::Glue::Model::Table> all_tables;
       Aws::String nextToken; // Used for pagination.
       do {
           Aws::Glue::Model::GetTablesOutcome outcome =
client.GetTables(request);

           if (outcome.IsSuccess()) {
               const std::vector<Aws::Glue::Model::Table> &tables =
outcome.GetResult().GetTableList();
               all_tables.insert(all_tables.end(), tables.begin(),
tables.end());
               nextToken = outcome.GetResult().GetNextToken();
           }
           else {
```

```
                std::cerr << "Error getting the tables. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                             clientConfig);
                return false;
            }
        } while (!nextToken.empty());

        std::cout << "The database contains " << all_tables.size()
                  << (all_tables.size() == 1 ?
                      " table." : "tables.") << std::endl;
        std::cout << "Here is a list of the tables in the database.";
        for (size_t index = 0; index < all_tables.size(); ++index) {
            std::cout << "    " << index + 1 << ":  " <<
all_tables[index].GetName()
                      << std::endl;
        }

        if (!all_tables.empty()) {
            int tableIndex = askQuestionForIntRange(
                    "Enter an index to display the database detail ",
                    1, static_cast<int>(all_tables.size()));
            std::cout << all_tables[tableIndex -
1].Jsonize().View().WriteReadable()
                      << std::endl;

            tableName = all_tables[tableIndex - 1].GetName();
        }
    }

    // 7. Create a job.
    {
        Aws::Glue::Model::CreateJobRequest request;
        request.SetName(JOB_NAME);
        request.SetRole(roleArn);
        request.SetGlueVersion(GLUE_VERSION);

        Aws::Glue::Model::JobCommand command;
        command.SetName(JOB_COMMAND_NAME);
        command.SetPythonVersion(JOB_PYTHON_VERSION);
        command.SetScriptLocation(
                Aws::String("s3://") + bucketName + "/" + PYTHON_SCRIPT);
        request.SetCommand(command);
```

```
        Aws::Glue::Model::CreateJobOutcome outcome = client.CreateJob(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully created the job." << std::endl;
        }
        else {
            std::cerr << "Error creating the job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, "", bucketName,
                         clientConfig);
            return false;
        }
    }

    // 8. Start a job run.
    {
        Aws::Glue::Model::StartJobRunRequest request;
        request.SetJobName(JOB_NAME);

        Aws::Map<Aws::String, Aws::String> arguments;
        arguments["--input_database"] = CRAWLER_DATABASE_NAME;
        arguments["--input_table"] = tableName;
        arguments["--output_bucket_url"] = Aws::String("s3://") + bucketName +
"/";
        request.SetArguments(arguments);

        Aws::Glue::Model::StartJobRunOutcome outcome =
client.StartJobRun(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully started the job." << std::endl;

            Aws::String jobRunId = outcome.GetResult().GetJobRunId();

            int iterator = 0;
            bool done = false;
            while (!done) {
                ++iterator;
                std::this_thread::sleep_for(std::chrono::seconds(1));
                Aws::Glue::Model::GetJobRunRequest jobRunRequest;
                jobRunRequest.SetJobName(JOB_NAME);
                jobRunRequest.SetRunId(jobRunId);
```

```
                Aws::Glue::Model::GetJobRunOutcome jobRunOutcome =
client.GetJobRun(
                        jobRunRequest);

                if (jobRunOutcome.IsSuccess()) {
                    const Aws::Glue::Model::JobRun &jobRun =
jobRunOutcome.GetResult().GetJobRun();
                    Aws::Glue::Model::JobRunState jobRunState =
jobRun.GetJobRunState();

                    if ((jobRunState == Aws::Glue::Model::JobRunState::STOPPED)
||
                        (jobRunState == Aws::Glue::Model::JobRunState::FAILED) ||
                        (jobRunState == Aws::Glue::Model::JobRunState::TIMEOUT))
{
                        std::cerr << "Error running job. "
                                  << jobRun.GetErrorMessage()
                                  << std::endl;
                        deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME,
JOB_NAME,
                                     bucketName,
                                     clientConfig);
                        return false;
                    }
                    else if (jobRunState ==
                             Aws::Glue::Model::JobRunState::SUCCEEDED) {
                        std::cout << "Job run succeeded after  " << iterator <<
                                  " seconds elapsed." << std::endl;
                        done = true;
                    }
                    else if ((iterator % 10) == 0) { // Log status every 10
seconds.
                        std::cout << "Job run status " <<

Aws::Glue::Model::JobRunStateMapper::GetNameForJobRunState(
                                         jobRunState) <<
                                  ". " << iterator <<
                                  " seconds elapsed." << std::endl;
                    }
                }
                else {
                    std::cerr << "Error retrieving job run state. "
                              << jobRunOutcome.GetError().GetMessage()
```

```
                            << std::endl;
                    deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
                                 bucketName, clientConfig);
                    return false;
                }
            }
        }
        else {
            std::cerr << "Error starting a job. " <<
outcome.GetError().GetMessage()
                      << std::endl;
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
bucketName,
                         clientConfig);
            return false;
        }
    }

    // 9. List the output data stored in the S3 bucket.
    {
        Aws::S3::S3Client s3Client;
        Aws::S3::Model::ListObjectsV2Request request;
        request.SetBucket(bucketName);
        request.SetPrefix(OUTPUT_FILE_PREFIX);

        Aws::String continuationToken; // Used for pagination.
        std::vector<Aws::S3::Model::Object> allObjects;
        do {
            if (!continuationToken.empty()) {
                request.SetContinuationToken(continuationToken);
            }
            Aws::S3::Model::ListObjectsV2Outcome outcome =
s3Client.ListObjectsV2(
                    request);

            if (outcome.IsSuccess()) {
                const std::vector<Aws::S3::Model::Object> &objects =
                        outcome.GetResult().GetContents();
                allObjects.insert(allObjects.end(), objects.begin(),
objects.end());
                continuationToken =
outcome.GetResult().GetNextContinuationToken();
            }
            else {
```

```
                std::cerr << "Error listing objects. "
                          << outcome.GetError().GetMessage()
                          << std::endl;
                break;
            }
        } while (!continuationToken.empty());

        std::cout << "Data from your job is in " << allObjects.size() <<
                  " files in the S3 bucket, " << bucketName << "." << std::endl;

        for (size_t i = 0; i < allObjects.size(); ++i) {
            std::cout << "    " << i + 1 << ". " << allObjects[i].GetKey()
                      << std::endl;
        }

        int objectIndex = askQuestionForIntRange(
                std::string(
                        "Enter the number of a block to download it and see the
 first ") +
                std::to_string(LINES_OF_RUN_FILE_TO_DISPLAY) +
                " lines of JSON output in the block: ", 1,
                static_cast<int>(allObjects.size()));

        Aws::String objectKey = allObjects[objectIndex - 1].GetKey();

        std::stringstream stringStream;
        if (getObjectFromBucket(bucketName, objectKey, stringStream,
                                clientConfig)) {
            for (int i = 0; i < LINES_OF_RUN_FILE_TO_DISPLAY && stringStream; +
+i) {
                std::string line;
                std::getline(stringStream, line);
                std::cout << "    " << line << std::endl;
            }
        }
        else {
            deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
 bucketName,
                         clientConfig);
            return false;
        }
    }

    // 10. List all the jobs.
```

```
    Aws::String jobName;
    {
        Aws::Glue::Model::ListJobsRequest listJobsRequest;

        Aws::String nextToken;
        std::vector<Aws::String> allJobNames;

        do {
            if (!nextToken.empty()) {
                listJobsRequest.SetNextToken(nextToken);
            }
            Aws::Glue::Model::ListJobsOutcome listRunsOutcome = client.ListJobs(
                    listJobsRequest);

            if (listRunsOutcome.IsSuccess()) {
                const std::vector<Aws::String> &jobNames =
listRunsOutcome.GetResult().GetJobNames();
                allJobNames.insert(allJobNames.end(), jobNames.begin(),
jobNames.end());
                nextToken = listRunsOutcome.GetResult().GetNextToken();
            }
            else {
                std::cerr << "Error listing jobs. "
                          << listRunsOutcome.GetError().GetMessage()
                          << std::endl;
            }
        } while (!nextToken.empty());
        std::cout << "Your account has " << allJobNames.size() << " jobs."
                  << std::endl;
        for (size_t i = 0; i < allJobNames.size(); ++i) {
            std::cout << "   " << i + 1 << ". " << allJobNames[i] << std::endl;
        }
        int jobIndex = askQuestionForIntRange(
                Aws::String("Enter a number between 1 and ") +
                std::to_string(allJobNames.size()) +
                " to see the list of runs for a job: ",
                1, static_cast<int>(allJobNames.size()));

        jobName = allJobNames[jobIndex - 1];
    }

    // 11. Get the job runs for a job.
    Aws::String jobRunID;
    if (!jobName.empty()) {
```

```
        Aws::Glue::Model::GetJobRunsRequest getJobRunsRequest;
        getJobRunsRequest.SetJobName(jobName);

        Aws::String nextToken; // Used for pagination.
        std::vector<Aws::Glue::Model::JobRun> allJobRuns;
        do {
            if (!nextToken.empty()) {
                getJobRunsRequest.SetNextToken(nextToken);
            }
            Aws::Glue::Model::GetJobRunsOutcome jobRunsOutcome =
client.GetJobRuns(
                    getJobRunsRequest);

            if (jobRunsOutcome.IsSuccess()) {
                const std::vector<Aws::Glue::Model::JobRun> &jobRuns =
jobRunsOutcome.GetResult().GetJobRuns();
                allJobRuns.insert(allJobRuns.end(), jobRuns.begin(),
jobRuns.end());

                nextToken = jobRunsOutcome.GetResult().GetNextToken();
            }
            else {
                std::cerr << "Error getting job runs. "
                          << jobRunsOutcome.GetError().GetMessage()
                          << std::endl;
                break;
            }
        } while (!nextToken.empty());

        std::cout << "There are " << allJobRuns.size() << " runs in the job '"
                  <<
                  jobName << "'." << std::endl;

        for (size_t i = 0; i < allJobRuns.size(); ++i) {
            std::cout << "   " << i + 1 << ". " << allJobRuns[i].GetJobName()
                      << std::endl;
        }

        int runIndex = askQuestionForIntRange(
                Aws::String("Enter a number between 1 and ") +
                std::to_string(allJobRuns.size()) +
                " to see details for a run: ",
                1, static_cast<int>(allJobRuns.size()));
        jobRunID = allJobRuns[runIndex - 1].GetId();
```

```
    }

    // 12. Get a single job run.
    if (!jobRunID.empty()) {
        Aws::Glue::Model::GetJobRunRequest jobRunRequest;
        jobRunRequest.SetJobName(jobName);
        jobRunRequest.SetRunId(jobRunID);

        Aws::Glue::Model::GetJobRunOutcome jobRunOutcome = client.GetJobRun(
                jobRunRequest);

        if (jobRunOutcome.IsSuccess()) {
            std::cout << "Displaying the job run JSON description." << std::endl;
            std::cout
                    <<
 jobRunOutcome.GetResult().GetJobRun().Jsonize().View().WriteReadable()
                    << std::endl;
        }
        else {
            std::cerr << "Error get a job run. "
                      << jobRunOutcome.GetError().GetMessage()
                      << std::endl;
        }
    }

    return deleteAssets(CRAWLER_NAME, CRAWLER_DATABASE_NAME, JOB_NAME,
 bucketName,
                        clientConfig);
}

//! Cleanup routine to delete created assets.
/*!
 \\sa deleteAssets()
 \param crawler: Name of an AWS Glue crawler.
 \param database: The name of an AWS Glue database.
 \param job: The name of an AWS Glue job.
 \param bucketName: The name of an S3 bucket.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */
bool AwsDoc::Glue::deleteAssets(const Aws::String &crawler, const Aws::String
 &database,
                                const Aws::String &job, const Aws::String
 &bucketName,
```

```
                                const Aws::Client::ClientConfiguration
&clientConfig) {
   const Aws::Glue::GlueClient client(clientConfig);
   bool result = true;

   // 13. Delete a job.
   if (!job.empty()) {
       Aws::Glue::Model::DeleteJobRequest request;
       request.SetJobName(job);

       Aws::Glue::Model::DeleteJobOutcome outcome = client.DeleteJob(request);


       if (outcome.IsSuccess()) {
           std::cout << "Successfully deleted the job." << std::endl;
       }
       else {
           std::cerr << "Error deleting the job. " <<
outcome.GetError().GetMessage()
                     << std::endl;
           result = false;
       }
   }

   // 14. Delete a database.
   if (!database.empty()) {
       Aws::Glue::Model::DeleteDatabaseRequest request;
       request.SetName(database);

       Aws::Glue::Model::DeleteDatabaseOutcome outcome = client.DeleteDatabase(
               request);

       if (outcome.IsSuccess()) {
           std::cout << "Successfully deleted the database." << std::endl;
       }
       else {
           std::cerr << "Error deleting database. " <<
outcome.GetError().GetMessage()
                     << std::endl;
           result = false;
       }
   }

   // 15. Delete a crawler.
```

```
    if (!crawler.empty()) {
        Aws::Glue::Model::DeleteCrawlerRequest request;
        request.SetName(crawler);

        Aws::Glue::Model::DeleteCrawlerOutcome outcome =
 client.DeleteCrawler(request);

        if (outcome.IsSuccess()) {
            std::cout << "Successfully deleted the crawler." << std::endl;
        }
        else {
            std::cerr << "Error deleting the crawler. "
                      << outcome.GetError().GetMessage() << std::endl;
            result = false;
        }
    }

    // 16. Delete the job script and run data from the S3 bucket.
    result &= AwsDoc::Glue::deleteAllObjectsInS3Bucket(bucketName,
                                                       clientConfig);
    return result;
}

//! Routine which uploads a file to an S3 bucket.
/*!
 \\sa uploadFile()
 \param bucketName: An S3 bucket created in the setup.
 \param filePath: The path of the file to upload.
 \param fileName The name for the uploaded file.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */
bool
AwsDoc::Glue::uploadFile(const Aws::String &bucketName,
                         const Aws::String &filePath,
                         const Aws::String &fileName,
                         const Aws::Client::ClientConfiguration &clientConfig) {
    Aws::S3::S3Client s3_client(clientConfig);

    Aws::S3::Model::PutObjectRequest request;
    request.SetBucket(bucketName);
    request.SetKey(fileName);

    std::shared_ptr<Aws::IOStream> inputData =
```

```
            Aws::MakeShared<Aws::FStream>("SampleAllocationTag",
                                          filePath.c_str(),
                                          std::ios_base::in |
 std::ios_base::binary);

    if (!*inputData) {
        std::cerr << "Error unable to read file " << filePath << std::endl;
        return false;
    }

    request.SetBody(inputData);

    Aws::S3::Model::PutObjectOutcome outcome =
            s3_client.PutObject(request);

    if (!outcome.IsSuccess()) {
        std::cerr << "Error: PutObject: " <<
                  outcome.GetError().GetMessage() << std::endl;
    }
    else {
        std::cout << "Added object '" << filePath << "' to bucket '"
                  << bucketName << "'." << std::endl;
    }

    return outcome.IsSuccess();
}

//! Routine which deletes all objects in an S3 bucket.
/*!
 \\sa deleteAllObjectsInS3Bucket()
 \param bucketName: The S3 bucket name.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */
bool AwsDoc::Glue::deleteAllObjectsInS3Bucket(const Aws::String &bucketName,
                                              const
 Aws::Client::ClientConfiguration &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::ListObjectsV2Request listObjectsRequest;
    listObjectsRequest.SetBucket(bucketName);

    Aws::String continuationToken; // Used for pagination.
    bool result = true;
    do {
```

```
        if (!continuationToken.empty()) {
            listObjectsRequest.SetContinuationToken(continuationToken);
        }

        Aws::S3::Model::ListObjectsV2Outcome listObjectsOutcome =
client.ListObjectsV2(
                listObjectsRequest);

        if (listObjectsOutcome.IsSuccess()) {
            const std::vector<Aws::S3::Model::Object> &objects =
listObjectsOutcome.GetResult().GetContents();
            if (!objects.empty()) {
                Aws::S3::Model::DeleteObjectsRequest deleteObjectsRequest;
                deleteObjectsRequest.SetBucket(bucketName);

                std::vector<Aws::S3::Model::ObjectIdentifier> objectIdentifiers;
                for (const Aws::S3::Model::Object &object: objects) {
                    objectIdentifiers.push_back(
                            Aws::S3::Model::ObjectIdentifier().WithKey(
                                    object.GetKey()));
                }
                Aws::S3::Model::Delete objectsDelete;
                objectsDelete.SetObjects(objectIdentifiers);
                objectsDelete.SetQuiet(true);
                deleteObjectsRequest.SetDelete(objectsDelete);

                Aws::S3::Model::DeleteObjectsOutcome deleteObjectsOutcome =
                        client.DeleteObjects(deleteObjectsRequest);

                if (!deleteObjectsOutcome.IsSuccess()) {
                    std::cerr << "Error deleting objects. " <<
                              deleteObjectsOutcome.GetError().GetMessage() <<
std::endl;
                    result = false;
                    break;
                }
                else {
                    std::cout << "Successfully deleted the objects." <<
std::endl;

                }
            }
            else {
                std::cout << "No objects to delete in '" << bucketName << "'."
```

```
                          << std::endl;
            }

            continuationToken =
 listObjectsOutcome.GetResult().GetNextContinuationToken();
        }
        else {
            std::cerr << "Error listing objects. "
                      << listObjectsOutcome.GetError().GetMessage() << std::endl;
            result = false;
            break;
        }
    } while (!continuationToken.empty());

    return result;
}

//! Routine which retrieves an object from an S3 bucket.
/*!
 \\sa getObjectFromBucket()
 \param bucketName: The S3 bucket name.
 \param objectKey: The object's name.
 \param objectStream: A stream to receive the retrieved data.
 \param clientConfig: AWS client configuration.
 \return bool: Successful completion.
 */
bool AwsDoc::Glue::getObjectFromBucket(const Aws::String &bucketName,
                                       const Aws::String &objectKey,
                                       std::ostream &objectStream,
                                       const Aws::Client::ClientConfiguration
 &clientConfig) {
    Aws::S3::S3Client client(clientConfig);
    Aws::S3::Model::GetObjectRequest request;
    request.SetBucket(bucketName);
    request.SetKey(objectKey);

    Aws::S3::Model::GetObjectOutcome outcome = client.GetObject(request);


    if (outcome.IsSuccess()) {
        std::cout << "Successfully retrieved '" << objectKey << "'." <<
 std::endl;
        auto &body = outcome.GetResult().GetBody();
        objectStream << body.rdbuf();
```

```
    }
    else {
        std::cerr << "Error retrieving object. " <<
 outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for C++ .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler
  - StartJobRun

Java

SDK untuk Java 2.x

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
/**
 *
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * To set up the resources, see this documentation topic:
 *
 * https://docs.aws.amazon.com/glue/latest/ug/tutorial-add-crawler.html
 *
 * This example performs the following tasks:
 *
 * 1. Create a database.
 * 2. Create a crawler.
 * 3. Get a crawler.
 * 4. Start a crawler.
 * 5. Get a database.
 * 6. Get tables.
 * 7. Create a job.
 * 8. Start a job run.
 * 9. List all jobs.
 * 10. Get job runs.
 * 11. Delete a job.
 * 12. Delete a database.
 * 13. Delete a crawler.
 */

public class GlueScenario {
```

```
   public static final String DASHES = new String(new char[80]).replace("\0",
"-");

   public static void main(String[] args) throws InterruptedException {
       final String usage = """

               Usage:
                   <iam> <s3Path> <cron> <dbName> <crawlerName> <jobName>\s

               Where:
                   iam - The ARN of the IAM role that has AWS Glue and S3
permissions.\s
                   s3Path - The Amazon Simple Storage Service (Amazon S3) target
that contains data (for example, CSV data).
                   cron - A cron expression used to specify the schedule  (i.e.,
cron(15 12 * * ? *).
                   dbName - The database name.\s
                   crawlerName - The name of the crawler.\s
                   jobName - The name you assign to this job definition.
                   scriptLocation - The Amazon S3 path to a script that runs a
job.
                   locationUri - The location of the database
                   bucketNameSc - The Amazon S3 bucket name used when creating a
job
                   """;

       if (args.length != 9) {
           System.out.println(usage);
           System.exit(1);
       }

       String iam = args[0];
       String s3Path = args[1];
       String cron = args[2];
       String dbName = args[3];
       String crawlerName = args[4];
       String jobName = args[5];
       String scriptLocation = args[6];
       String locationUri = args[7];
       String bucketNameSc = args[8];

       Region region = Region.US_EAST_1;
       GlueClient glueClient = GlueClient.builder()
               .region(region)
```

```
                .build();
        System.out.println(DASHES);
        System.out.println("Welcome to the AWS Glue scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("1. Create a database.");
        createDatabase(glueClient, dbName, locationUri);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("2. Create a crawler.");
        createGlueCrawler(glueClient, iam, s3Path, cron, dbName, crawlerName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("3. Get a crawler.");
        getSpecificCrawler(glueClient, crawlerName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Start a crawler.");
        startSpecificCrawler(glueClient, crawlerName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("5. Get a database.");
        getSpecificDatabase(glueClient, dbName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Wait 5 min for the tables to become available");
        TimeUnit.MINUTES.sleep(5);
        System.out.println("6. Get tables.");
        String myTableName = getGlueTables(glueClient, dbName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("7. Create a job.");
        createJob(glueClient, jobName, iam, scriptLocation);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Start a Job run.");
```

```
        startJob(glueClient, jobName, dbName, myTableName, bucketNameSc);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("9. List all jobs.");
        getAllJobs(glueClient);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("10. Get job runs.");
        getJobRuns(glueClient, jobName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("11. Delete a job.");
        deleteJob(glueClient, jobName);
        System.out.println("*** Wait 5 MIN for the " + crawlerName + " to stop");
        TimeUnit.MINUTES.sleep(5);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("12. Delete a database.");
        deleteDatabase(glueClient, dbName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Delete a crawler.");
        deleteSpecificCrawler(glueClient, crawlerName);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("Successfully completed the AWS Glue Scenario");
        System.out.println(DASHES);
    }

    public static void createDatabase(GlueClient glueClient, String dbName,
String locationUri) {
        try {
            DatabaseInput input = DatabaseInput.builder()
                    .description("Built with the AWS SDK for Java V2")
                    .name(dbName)
                    .locationUri(locationUri)
                    .build();
```

```
            CreateDatabaseRequest request = CreateDatabaseRequest.builder()
                    .databaseInput(input)
                    .build();

            glueClient.createDatabase(request);
            System.out.println(dbName + " was successfully created");

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void createGlueCrawler(GlueClient glueClient,
            String iam,
            String s3Path,
            String cron,
            String dbName,
            String crawlerName) {

        try {
            S3Target s3Target = S3Target.builder()
                    .path(s3Path)
                    .build();

            List<S3Target> targetList = new ArrayList<>();
            targetList.add(s3Target);
            CrawlerTargets targets = CrawlerTargets.builder()
                    .s3Targets(targetList)
                    .build();

            CreateCrawlerRequest crawlerRequest = CreateCrawlerRequest.builder()
                    .databaseName(dbName)
                    .name(crawlerName)
                    .description("Created by the AWS Glue Java API")
                    .targets(targets)
                    .role(iam)
                    .schedule(cron)
                    .build();

            glueClient.createCrawler(crawlerRequest);
            System.out.println(crawlerName + " was successfully created");

        } catch (GlueException e) {
```

```
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void getSpecificCrawler(GlueClient glueClient, String
crawlerName) {
        try {
            GetCrawlerRequest crawlerRequest = GetCrawlerRequest.builder()
                    .name(crawlerName)
                    .build();

            boolean ready = false;
            while (!ready) {
                GetCrawlerResponse response =
glueClient.getCrawler(crawlerRequest);
                String status = response.crawler().stateAsString();
                if (status.compareTo("READY") == 0) {
                    ready = true;
                }
                Thread.sleep(3000);
            }

            System.out.println("The crawler is now ready");

        } catch (GlueException | InterruptedException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static void startSpecificCrawler(GlueClient glueClient, String
crawlerName) {
        try {
            StartCrawlerRequest crawlerRequest = StartCrawlerRequest.builder()
                    .name(crawlerName)
                    .build();

            glueClient.startCrawler(crawlerRequest);
            System.out.println(crawlerName + " was successfully started!");

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
```

```
        }
    }

    public static void getSpecificDatabase(GlueClient glueClient, String
databaseName) {
        try {
            GetDatabaseRequest databasesRequest = GetDatabaseRequest.builder()
                    .name(databaseName)
                    .build();

            GetDatabaseResponse response =
glueClient.getDatabase(databasesRequest);
            Instant createDate = response.database().createTime();

            // Convert the Instant to readable date.
            DateTimeFormatter formatter =
DateTimeFormatter.ofLocalizedDateTime(FormatStyle.SHORT)
                    .withLocale(Locale.US)
                    .withZone(ZoneId.systemDefault());

            formatter.format(createDate);
            System.out.println("The create date of the database is " +
createDate);

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static String getGlueTables(GlueClient glueClient, String dbName) {
        String myTableName = "";
        try {
            GetTablesRequest tableRequest = GetTablesRequest.builder()
                    .databaseName(dbName)
                    .build();

            GetTablesResponse response = glueClient.getTables(tableRequest);
            List<Table> tables = response.tableList();
            if (tables.isEmpty()) {
                System.out.println("No tables were returned");
            } else {
                for (Table table : tables) {
                    myTableName = table.name();
```

```
                    System.out.println("Table name is: " + myTableName);
                }
            }

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return myTableName;
    }

    public static void startJob(GlueClient glueClient, String jobName, String
inputDatabase, String inputTable,
            String outBucket) {
        try {
            Map<String, String> myMap = new HashMap<>();
            myMap.put("--input_database", inputDatabase);
            myMap.put("--input_table", inputTable);
            myMap.put("--output_bucket_url", outBucket);

            StartJobRunRequest runRequest = StartJobRunRequest.builder()
                    .workerType(WorkerType.G_1_X)
                    .numberOfWorkers(10)
                    .arguments(myMap)
                    .jobName(jobName)
                    .build();

            StartJobRunResponse response = glueClient.startJobRun(runRequest);
            System.out.println("The request Id of the job is " +
response.responseMetadata().requestId());

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void createJob(GlueClient glueClient, String jobName, String
iam, String scriptLocation) {
        try {
            JobCommand command = JobCommand.builder()
                    .pythonVersion("3")
                    .name("glueetl")
                    .scriptLocation(scriptLocation)
```

```
                    .build();

            CreateJobRequest jobRequest = CreateJobRequest.builder()
                    .description("A Job created by using the AWS SDK for Java
V2")
                    .glueVersion("2.0")
                    .workerType(WorkerType.G_1_X)
                    .numberOfWorkers(10)
                    .name(jobName)
                    .role(iam)
                    .command(command)
                    .build();

            glueClient.createJob(jobRequest);
            System.out.println(jobName + " was successfully created.");

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void getAllJobs(GlueClient glueClient) {
        try {
            GetJobsRequest jobsRequest = GetJobsRequest.builder()
                    .maxResults(10)
                    .build();

            GetJobsResponse jobsResponse = glueClient.getJobs(jobsRequest);
            List<Job> jobs = jobsResponse.jobs();
            for (Job job : jobs) {
                System.out.println("Job name is : " + job.name());
                System.out.println("The job worker type is : " +
job.workerType().name());
            }

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void getJobRuns(GlueClient glueClient, String jobName) {
        try {
```

```
            GetJobRunsRequest runsRequest = GetJobRunsRequest.builder()
                    .jobName(jobName)
                    .maxResults(20)
                    .build();

            boolean jobDone = false;
            while (!jobDone) {
                GetJobRunsResponse response = glueClient.getJobRuns(runsRequest);
                List<JobRun> jobRuns = response.jobRuns();
                for (JobRun jobRun : jobRuns) {
                    String jobState = jobRun.jobRunState().name();
                    if (jobState.compareTo("SUCCEEDED") == 0) {
                        System.out.println(jobName + " has succeeded");
                        jobDone = true;

                    } else if (jobState.compareTo("STOPPED") == 0) {
                        System.out.println("Job run has stopped");
                        jobDone = true;

                    } else if (jobState.compareTo("FAILED") == 0) {
                        System.out.println("Job run has failed");
                        jobDone = true;

                    } else if (jobState.compareTo("TIMEOUT") == 0) {
                        System.out.println("Job run has timed out");
                        jobDone = true;

                    } else {
                        System.out.println("*** Job run state is " +
jobRun.jobRunState().name());
                        System.out.println("Job run Id is " + jobRun.id());
                        System.out.println("The Glue version is " +
jobRun.glueVersion());
                    }
                    TimeUnit.SECONDS.sleep(5);
                }
            }

        } catch (GlueException | InterruptedException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }
```

```
    public static void deleteJob(GlueClient glueClient, String jobName) {
        try {
            DeleteJobRequest jobRequest = DeleteJobRequest.builder()
                    .jobName(jobName)
                    .build();

            glueClient.deleteJob(jobRequest);
            System.out.println(jobName + " was successfully deleted");

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteDatabase(GlueClient glueClient, String databaseName)
{
        try {
            DeleteDatabaseRequest request = DeleteDatabaseRequest.builder()
                    .name(databaseName)
                    .build();

            glueClient.deleteDatabase(request);
            System.out.println(databaseName + " was successfully deleted");

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void deleteSpecificCrawler(GlueClient glueClient, String
crawlerName) {
        try {
            DeleteCrawlerRequest deleteCrawlerRequest =
DeleteCrawlerRequest.builder()
                    .name(crawlerName)
                    .build();

            glueClient.deleteCrawler(deleteCrawlerRequest);
            System.out.println(crawlerName + " was deleted");

        } catch (GlueException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
```

```
            System.exit(1);
        }
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Java 2.x .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler
  - StartJobRun

JavaScript

SDK untuk JavaScript (v3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Membuat dan menjalankan crawler yang merayapi bucket Amazon Simple Storage Service (Amazon S3) publik dan menghasilkan database metadata yang menjelaskan data berformat CSV yang ditemukannya.

```
const createCrawler = (name, role, dbName, tablePrefix, s3TargetPath) => {
  const client = new GlueClient({});

  const command = new CreateCrawlerCommand({
    Name: name,
    Role: role,
    DatabaseName: dbName,
    TablePrefix: tablePrefix,
    Targets: {
      S3Targets: [{ Path: s3TargetPath }],
    },
  });

  return client.send(command);
};

const getCrawler = (name) => {
  const client = new GlueClient({});

  const command = new GetCrawlerCommand({
    Name: name,
  });

  return client.send(command);
};

const startCrawler = (name) => {
  const client = new GlueClient({});

  const command = new StartCrawlerCommand({
    Name: name,
  });

  return client.send(command);
};

const crawlerExists = async ({ getCrawler }, crawlerName) => {
  try {
    await getCrawler(crawlerName);
```

```
    return true;
  } catch {
    return false;
  }
};

/**
 * @param {{ createCrawler: import('../../../actions/create-
crawler.js').createCrawler}} actions
 */
const makeCreateCrawlerStep = (actions) => async (context) => {
  if (await crawlerExists(actions, process.env.CRAWLER_NAME)) {
    log("Crawler already exists. Skipping creation.");
  } else {
    await actions.createCrawler(
      process.env.CRAWLER_NAME,
      process.env.ROLE_NAME,
      process.env.DATABASE_NAME,
      process.env.TABLE_PREFIX,
      process.env.S3_TARGET_PATH,
    );

    log("Crawler created successfully.", { type: "success" });
  }

  return { ...context };
};

/**
 * @param {(name: string) => Promise<import('@aws-sdk/client-
glue').GetCrawlerCommandOutput>} getCrawler
 * @param {string} crawlerName
 */
const waitForCrawler = async (getCrawler, crawlerName) => {
  const waitTimeInSeconds = 30;
  const { Crawler } = await getCrawler(crawlerName);

  if (!Crawler) {
    throw new Error(`Crawler with name ${crawlerName} not found.`);
  }

  if (Crawler.State === "READY") {
    return;
  }
```

```
  log(`Crawler is ${Crawler.State}. Waiting ${waitTimeInSeconds} seconds...`);
  await wait(waitTimeInSeconds);
  return waitForCrawler(getCrawler, crawlerName);
};

const makeStartCrawlerStep =
  ({ startCrawler, getCrawler }) =>
  async (context) => {
    log("Starting crawler.");
    await startCrawler(process.env.CRAWLER_NAME);
    log("Crawler started.", { type: "success" });

    log("Waiting for crawler to finish running. This can take a while.");
    await waitForCrawler(getCrawler, process.env.CRAWLER_NAME);
    log("Crawler ready.", { type: "success" });

    return { ...context };
  };
```

Daftar informasi tentang database dan tabel di situs Anda AWS Glue Data Catalog.

```
const getDatabase = (name) => {
  const client = new GlueClient({});

  const command = new GetDatabaseCommand({
    Name: name,
  });

  return client.send(command);
};

const getTables = (databaseName) => {
  const client = new GlueClient({});

  const command = new GetTablesCommand({
    DatabaseName: databaseName,
  });

  return client.send(command);
};
```

```
const makeGetDatabaseStep =
  ({ getDatabase }) =>
  async (context) => {
    const {
      Database: { Name },
    } = await getDatabase(process.env.DATABASE_NAME);
    log(`Database: ${Name}`);
    return { ...context };
  };

/**
 * @param {{ getTables: () => Promise<import('@aws-sdk/client-
glue').GetTablesCommandOutput}} config
 */
const makeGetTablesStep =
  ({ getTables }) =>
  async (context) => {
    const { TableList } = await getTables(process.env.DATABASE_NAME);
    log("Tables:");
    log(TableList.map((table) => `  • ${table.Name}\n`));
    return { ...context };
  };
```

Buat dan jalankan job yang mengekstrak data CSV dari bucket Amazon S3 sumber, mengubahnya dengan menghapus dan mengganti nama bidang, dan memuat output berformat JSON ke bucket Amazon S3 lainnya.

```
const createJob = (name, role, scriptBucketName, scriptKey) => {
  const client = new GlueClient({});

  const command = new CreateJobCommand({
    Name: name,
    Role: role,
    Command: {
      Name: "glueetl",
      PythonVersion: "3",
      ScriptLocation: `s3://${scriptBucketName}/${scriptKey}`,
    },
    GlueVersion: "3.0",
  });

  return client.send(command);
```

```
};

const startJobRun = (jobName, dbName, tableName, bucketName) => {
  const client = new GlueClient({});

  const command = new StartJobRunCommand({
    JobName: jobName,
    Arguments: {
      "--input_database": dbName,
      "--input_table": tableName,
      "--output_bucket_url": `s3://${bucketName}/`,
    },
  });

  return client.send(command);
};

const makeCreateJobStep =
  ({ createJob }) =>
  async (context) => {
    log("Creating Job.");
    await createJob(
      process.env.JOB_NAME,
      process.env.ROLE_NAME,
      process.env.BUCKET_NAME,
      process.env.PYTHON_SCRIPT_KEY,
    );
    log("Job created.", { type: "success" });

    return { ...context };
  };

/**
 * @param {(name: string, runId: string) => Promise<import('@aws-sdk/client-
glue').GetJobRunCommandOutput> }  getJobRun
 * @param {string} jobName
 * @param {string} jobRunId
 */
const waitForJobRun = async (getJobRun, jobName, jobRunId) => {
  const waitTimeInSeconds = 30;
  const { JobRun } = await getJobRun(jobName, jobRunId);

  if (!JobRun) {
    throw new Error(`Job run with id ${jobRunId} not found.`);
```

```
  }

  switch (JobRun.JobRunState) {
    case "FAILED":
    case "TIMEOUT":
    case "STOPPED":
      throw new Error(
        `Job ${JobRun.JobRunState}. Error: ${JobRun.ErrorMessage}`,
      );
    case "RUNNING":
      break;
    case "SUCCEEDED":
      return;
    default:
      throw new Error(`Unknown job run state: ${JobRun.JobRunState}`);
  }

  log(
    `Job ${JobRun.JobRunState}. Waiting ${waitTimeInSeconds} more seconds...`,
  );
  await wait(waitTimeInSeconds);
  return waitForJobRun(getJobRun, jobName, jobRunId);
};

/**
 * @param {{ prompter: { prompt: () => Promise<{ shouldOpen: boolean }>} }}
 context
 */
const promptToOpen = async (context) => {
  const { shouldOpen } = await context.prompter.prompt({
    name: "shouldOpen",
    type: "confirm",
    message: "Open the output bucket in your browser?",
  });

  if (shouldOpen) {
    return open(
      `https://s3.console.aws.amazon.com/s3/buckets/${process.env.BUCKET_NAME} to
 view the output.`,
    );
  }
};

const makeStartJobRunStep =
```

```
  ({ startJobRun, getJobRun }) =>
  async (context) => {
    log("Starting job.");
    const { JobRunId } = await startJobRun(
      process.env.JOB_NAME,
      process.env.DATABASE_NAME,
      process.env.TABLE_NAME,
      process.env.BUCKET_NAME,
    );
    log("Job started.", { type: "success" });

    log("Waiting for job to finish running. This can take a while.");
    await waitForJobRun(getJobRun, process.env.JOB_NAME, JobRunId);
    log("Job run succeeded.", { type: "success" });

    await promptToOpen(context);

    return { ...context };
  };
```

Buat daftar informasi tentang pekerjaan berjalan dan lihat beberapa data yang diubah.

```
const getJobRuns = (jobName) => {
  const client = new GlueClient({});
  const command = new GetJobRunsCommand({
    JobName: jobName,
  });

  return client.send(command);
};

const getJobRun = (jobName, jobRunId) => {
  const client = new GlueClient({});
  const command = new GetJobRunCommand({
    JobName: jobName,
    RunId: jobRunId,
  });

  return client.send(command);
};

/**
```

```
 * @typedef {{ prompter: { prompt: () => Promise<{jobName: string}> } }} Context
 */

/**
 * @typedef {() => Promise<import('@aws-sdk/client-
glue').GetJobRunCommandOutput>} getJobRun
 */

/**
 * @typedef {() => Promise<import('@aws-sdk/client-
glue').GetJobRunsCommandOutput} getJobRuns
 */

/**
 *
 * @param {getJobRun} getJobRun
 * @param {string} jobName
 * @param {string} jobRunId
 */
const logJobRunDetails = async (getJobRun, jobName, jobRunId) => {
  const { JobRun } = await getJobRun(jobName, jobRunId);
  log(JobRun, { type: "object" });
};

/**
 *
 * @param {{getJobRuns: getJobRuns, getJobRun: getJobRun }} funcs
 */
const makePickJobRunStep =
  ({ getJobRuns, getJobRun }) =>
  async (/** @type { Context } */ context) => {
    if (context.selectedJobName) {
      const { JobRuns } = await getJobRuns(context.selectedJobName);

      const { jobRunId } = await context.prompter.prompt({
        name: "jobRunId",
        type: "list",
        message: "Select a job run to see details.",
        choices: JobRuns.map((run) => run.Id),
      });

      logJobRunDetails(getJobRun, context.selectedJobName, jobRunId);
    }
```

```
    return { ...context };
  };
```

Hapus semua sumber daya yang dibuat oleh demo.

```
const deleteJob = (jobName) => {
  const client = new GlueClient({});

  const command = new DeleteJobCommand({
    JobName: jobName,
  });

  return client.send(command);
};

const deleteTable = (databaseName, tableName) => {
  const client = new GlueClient({});

  const command = new DeleteTableCommand({
    DatabaseName: databaseName,
    Name: tableName,
  });

  return client.send(command);
};

const deleteDatabase = (databaseName) => {
  const client = new GlueClient({});

  const command = new DeleteDatabaseCommand({
    Name: databaseName,
  });

  return client.send(command);
};

const deleteCrawler = (crawlerName) => {
  const client = new GlueClient({});

  const command = new DeleteCrawlerCommand({
    Name: crawlerName,
  });
```

```
  return client.send(command);
};

/**
 *
 * @param {import('../../../actions/delete-job.js').deleteJob} deleteJobFn
 * @param {string[]} jobNames
 * @param {{ prompter: { prompt: () => Promise<any> }}} context
 */
const handleDeleteJobs = async (deleteJobFn, jobNames, context) => {
  /**
   * @type {{ selectedJobNames: string[] }}
   */
  const { selectedJobNames } = await context.prompter.prompt({
    name: "selectedJobNames",
    type: "checkbox",
    message: "Let's clean up jobs. Select jobs to delete.",
    choices: jobNames,
  });

  if (selectedJobNames.length === 0) {
    log("No jobs selected.");
  } else {
    log("Deleting jobs.");
    await Promise.all(
      selectedJobNames.map((n) => deleteJobFn(n).catch(console.error)),
    );
    log("Jobs deleted.", { type: "success" });
  }
};

/**
 * @param {{
 *   listJobs: import('../../../actions/list-jobs.js').listJobs,
 *   deleteJob: import('../../../actions/delete-job.js').deleteJob
 * }} config
 */
const makeCleanUpJobsStep =
  ({ listJobs, deleteJob }) =>
  async (context) => {
    const { JobNames } = await listJobs();
    if (JobNames.length > 0) {
      await handleDeleteJobs(deleteJob, JobNames, context);
```

```
    }

    return { ...context };
  };

/**
 * @param {import('../../../actions/delete-table.js').deleteTable} deleteTable
 * @param {string} databaseName
 * @param {string[]} tableNames
 */
const deleteTables = (deleteTable, databaseName, tableNames) =>
  Promise.all(
    tableNames.map((tableName) =>
      deleteTable(databaseName, tableName).catch(console.error),
    ),
  );

/**
 * @param {{
 *  getTables: import('../../../actions/get-tables.js').getTables,
 *  deleteTable: import('../../../actions/delete-table.js').deleteTable
 * }} config
 */
const makeCleanUpTablesStep =
  ({ getTables, deleteTable }) =>
  /**
   * @param {{ prompter: { prompt: () => Promise<any>}}} context
   */
  async (context) => {
    const { TableList } = await getTables(process.env.DATABASE_NAME).catch(
      () => ({ TableList: null }),
    );

    if (TableList && TableList.length > 0) {
      /**
       * @type {{ tableNames: string[] }}
       */
      const { tableNames } = await context.prompter.prompt({
        name: "tableNames",
        type: "checkbox",
        message: "Let's clean up tables. Select tables to delete.",
        choices: TableList.map((t) => t.Name),
      });
```

```
      if (tableNames.length === 0) {
        log("No tables selected.");
      } else {
        log("Deleting tables.");
        await deleteTables(deleteTable, process.env.DATABASE_NAME, tableNames);
        log("Tables deleted.", { type: "success" });
      }
    }

    return { ...context };
  };

/**
 * @param {import('../../../actions/delete-database.js').deleteDatabase}
 deleteDatabase
 * @param {string[]} databaseNames
 */
const deleteDatabases = (deleteDatabase, databaseNames) =>
  Promise.all(
    databaseNames.map((dbName) => deleteDatabase(dbName).catch(console.error)),
  );

/**
 * @param {{
 *   getDatabases: import('../../../actions/get-databases.js').getDatabases
 *   deleteDatabase: import('../../../actions/delete-database.js').deleteDatabase
 * }} config
 */
const makeCleanUpDatabasesStep =
  ({ getDatabases, deleteDatabase }) =>
  /**
   * @param {{ prompter: { prompt: () => Promise<any>}} context
   */
  async (context) => {
    const { DatabaseList } = await getDatabases();

    if (DatabaseList.length > 0) {
      /** @type {{ dbNames: string[] }} */
      const { dbNames } = await context.prompter.prompt({
        name: "dbNames",
        type: "checkbox",
        message: "Let's clean up databases. Select databases to delete.",
        choices: DatabaseList.map((db) => db.Name),
      });
```

```
      if (dbNames.length === 0) {
        log("No databases selected.");
      } else {
        log("Deleting databases.");
        await deleteDatabases(deleteDatabase, dbNames);
        log("Databases deleted.", { type: "success" });
      }
    }

    return { ...context };
  };

const cleanUpCrawlerStep = async (context) => {
  log(`Deleting crawler.`);

  try {
    await deleteCrawler(process.env.CRAWLER_NAME);
    log("Crawler deleted.", { type: "success" });
  } catch (err) {
    if (err.name === "EntityNotFoundException") {
      log(`Crawler is already deleted.`);
    } else {
      throw err;
    }
  }

  return { ...context };
};
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for JavaScript .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase

- GetDatabases
- GetJob
- GetJobRun
- GetJobRuns
- GetTables
- ListJobs
- StartCrawler
- StartJobRun

Kotlin

SDK untuk Kotlin

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
suspend fun main(args: Array<String>) {
    val usage = """
        Usage:
            <iam> <s3Path> <cron> <dbName> <crawlerName> <jobName>
 <scriptLocation> <locationUri>

        Where:
            iam - The Amazon Resource Name (ARN) of the AWS Identity and Access
 Management (IAM) role that has AWS Glue and Amazon Simple Storage Service
 (Amazon S3) permissions.
            s3Path - The Amazon Simple Storage Service (Amazon S3) target that
 contains data (for example, CSV data).
            cron - A cron expression used to specify the schedule (for example,
 cron(15 12 * * ? *).
            dbName - The database name.
            crawlerName - The name of the crawler.
            jobName - The name you assign to this job definition.
            scriptLocation - Specifies the Amazon S3 path to a script that runs a
 job.
```

```
            locationUri - Specifies the location of the database
        """

    if (args.size != 8) {
        println(usage)
        exitProcess(1)
    }

    val iam = args[0]
    val s3Path = args[1]
    val cron = args[2]
    val dbName = args[3]
    val crawlerName = args[4]
    val jobName = args[5]
    val scriptLocation = args[6]
    val locationUri = args[7]

    println("About to start the AWS Glue Scenario")
    createDatabase(dbName, locationUri)
    createCrawler(iam, s3Path, cron, dbName, crawlerName)
    getCrawler(crawlerName)
    startCrawler(crawlerName)
    getDatabase(dbName)
    getGlueTables(dbName)
    createJob(jobName, iam, scriptLocation)
    startJob(jobName)
    getJobs()
    getJobRuns(jobName)
    deleteJob(jobName)
    println("*** Wait for 5 MIN so the $crawlerName is ready to be deleted")
    TimeUnit.MINUTES.sleep(5)
    deleteMyDatabase(dbName)
    deleteCrawler(crawlerName)
}

suspend fun createDatabase(
    dbName: String?,
    locationUriVal: String?,
) {
    val input =
        DatabaseInput {
            description = "Built with the AWS SDK for Kotlin"
            name = dbName
            locationUri = locationUriVal
```

```
        }

    val request =
        CreateDatabaseRequest {
            databaseInput = input
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
        glueClient.createDatabase(request)
        println("The database was successfully created")
    }
}

suspend fun createCrawler(
    iam: String?,
    s3Path: String?,
    cron: String?,
    dbName: String?,
    crawlerName: String,
) {
    val s3Target =
        S3Target {
            path = s3Path
        }

    val targetList = ArrayList<S3Target>()
    targetList.add(s3Target)

    val targetOb =
        CrawlerTargets {
            s3Targets = targetList
        }

    val crawlerRequest =
        CreateCrawlerRequest {
            databaseName = dbName
            name = crawlerName
            description = "Created by the AWS Glue Java API"
            targets = targetOb
            role = iam
            schedule = cron
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
```

```
        glueClient.createCrawler(crawlerRequest)
        println("$crawlerName was successfully created")
    }
}

suspend fun getCrawler(crawlerName: String?) {
    val request =
        GetCrawlerRequest {
            name = crawlerName
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getCrawler(request)
        val role = response.crawler?.role
        println("The role associated with this crawler is $role")
    }
}

suspend fun startCrawler(crawlerName: String) {
    val crawlerRequest =
        StartCrawlerRequest {
            name = crawlerName
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
        glueClient.startCrawler(crawlerRequest)
        println("$crawlerName was successfully started.")
    }
}

suspend fun getDatabase(databaseName: String?) {
    val request =
        GetDatabaseRequest {
            name = databaseName
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getDatabase(request)
        val dbDesc = response.database?.description
        println("The database description is $dbDesc")
    }
}

suspend fun getGlueTables(dbName: String?) {
```

```
    val tableRequest =
        GetTablesRequest {
            databaseName = dbName
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getTables(tableRequest)
        response.tableList?.forEach { tableName ->
            println("Table name is ${tableName.name}")
        }
    }
}

suspend fun startJob(jobNameVal: String?) {
    val runRequest =
        StartJobRunRequest {
            workerType = WorkerType.G1X
            numberOfWorkers = 10
            jobName = jobNameVal
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.startJobRun(runRequest)
        println("The job run Id is ${response.jobRunId}")
    }
}

suspend fun createJob(
    jobName: String,
    iam: String?,
    scriptLocationVal: String?,
) {
    val commandOb =
        JobCommand {
            pythonVersion = "3"
            name = "MyJob1"
            scriptLocation = scriptLocationVal
        }

    val jobRequest =
        CreateJobRequest {
            description = "A Job created by using the AWS SDK for Java V2"
            glueVersion = "2.0"
            workerType = WorkerType.G1X
```

```
            numberOfWorkers = 10
            name = jobName
            role = iam
            command = commandOb
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
        glueClient.createJob(jobRequest)
        println("$jobName was successfully created.")
    }
}

suspend fun getJobs() {
    val request =
        GetJobsRequest {
            maxResults = 10
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getJobs(request)
        response.jobs?.forEach { job ->
            println("Job name is ${job.name}")
        }
    }
}

suspend fun getJobRuns(jobNameVal: String?) {
    val request =
        GetJobRunsRequest {
            jobName = jobNameVal
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
        val response = glueClient.getJobRuns(request)
        response.jobRuns?.forEach { job ->
            println("Job name is ${job.jobName}")
        }
    }
}

suspend fun deleteJob(jobNameVal: String) {
    val jobRequest =
        DeleteJobRequest {
            jobName = jobNameVal
```

```
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
        glueClient.deleteJob(jobRequest)
        println("$jobNameVal was successfully deleted")
    }
}

suspend fun deleteMyDatabase(databaseName: String) {
    val request =
        DeleteDatabaseRequest {
            name = databaseName
        }

    GlueClient { region = "us-east-1" }.use { glueClient ->
        glueClient.deleteDatabase(request)
        println("$databaseName was successfully deleted")
    }
}

suspend fun deleteCrawler(crawlerName: String) {
    val request =
        DeleteCrawlerRequest {
            name = crawlerName
        }
    GlueClient { region = "us-east-1" }.use { glueClient ->
        glueClient.deleteCrawler(request)
        println("$crawlerName was deleted")
    }
}
```

- Untuk detail API, lihat topik berikut di referensi API SDK untuk Kotlin AWS .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler

- GetDatabase
- GetDatabases
- GetJob
- GetJobRun
- GetJobRuns
- GetTables
- ListJobs
- StartCrawler
- StartJobRun

PHP

SDK untuk PHP

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

```
namespace Glue;

use Aws\Glue\GlueClient;
use Aws\S3\S3Client;
use AwsUtilities\AWSServiceClass;
use GuzzleHttp\Psr7\Stream;
use Iam\IAMService;

class GettingStartedWithGlue
{
    public function run()
    {
        echo("\n");
        echo("--------------------------------------\n");
        print("Welcome to the AWS Glue getting started demo using PHP!\n");
        echo("--------------------------------------\n");

        $clientArgs = [
```

```
            'region' => 'us-west-2',
            'version' => 'latest',
            'profile' => 'default',
        ];
        $uniqid = uniqid();

        $glueClient = new GlueClient($clientArgs);
        $glueService = new GlueService($glueClient);
        $iamService = new IAMService();
        $crawlerName = "example-crawler-test-" . $uniqid;

        AWSServiceClass::$waitTime = 5;
        AWSServiceClass::$maxWaitAttempts = 20;

        $role = $iamService->getRole("AWSGlueServiceRole-DocExample");

        $databaseName = "doc-example-database-$uniqid";
        $path = 's3://crawler-public-us-east-1/flight/2016/csv';
        $glueService->createCrawler($crawlerName, $role['Role']['Arn'],
$databaseName, $path);
        $glueService->startCrawler($crawlerName);

        echo "Waiting for crawler";
        do {
            $crawler = $glueService->getCrawler($crawlerName);
            echo ".";
            sleep(10);
        } while ($crawler['Crawler']['State'] != "READY");
        echo "\n";

        $database = $glueService->getDatabase($databaseName);
        echo "Found a database named " . $database['Database']['Name'] . "\n";

        //Upload job script
        $s3client = new S3Client($clientArgs);
        $bucketName = "test-glue-bucket-" . $uniqid;
        $s3client->createBucket([
            'Bucket' => $bucketName,
            'CreateBucketConfiguration' => ['LocationConstraint' => 'us-west-2'],
        ]);

        $s3client->putObject([
            'Bucket' => $bucketName,
            'Key' => 'run_job.py',
```

```
            'SourceFile' => __DIR__ . '/flight_etl_job_script.py'
        ]);
        $s3client->putObject([
            'Bucket' => $bucketName,
            'Key' => 'setup_scenario_getting_started.yaml',
            'SourceFile' => __DIR__ . '/setup_scenario_getting_started.yaml'
        ]);

        $tables = $glueService->getTables($databaseName);

        $jobName = 'test-job-' . $uniqid;
        $scriptLocation = "s3://$bucketName/run_job.py";
        $job = $glueService->createJob($jobName, $role['Role']['Arn'],
$scriptLocation);

        $outputBucketUrl = "s3://$bucketName";
        $runId = $glueService->startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl)['JobRunId'];

        echo "waiting for job";
        do {
            $jobRun = $glueService->getJobRun($jobName, $runId);
            echo ".";
            sleep(10);
        } while (!array_intersect([$jobRun['JobRun']['JobRunState']],
['SUCCEEDED', 'STOPPED', 'FAILED', 'TIMEOUT']));
        echo "\n";

        $jobRuns = $glueService->getJobRuns($jobName);

        $objects = $s3client->listObjects([
            'Bucket' => $bucketName,
        ])['Contents'];

        foreach ($objects as $object) {
            echo $object['Key'] . "\n";
        }

        echo "Downloading " . $objects[1]['Key'] . "\n";
        /** @var Stream $downloadObject */
        $downloadObject = $s3client->getObject([
            'Bucket' => $bucketName,
            'Key' => $objects[1]['Key'],
        ])['Body']->getContents();
```

```
        echo "Here is the first 1000 characters in the object.";
        echo substr($downloadObject, 0, 1000);

        $jobs = $glueService->listJobs();
        echo "Current jobs:\n";
        foreach ($jobs['JobNames'] as $jobsName) {
            echo "{$jobsName}\n";
        }

        echo "Delete the job.\n";
        $glueClient->deleteJob([
            'JobName' => $job['Name'],
        ]);

        echo "Delete the tables.\n";
        foreach ($tables['TableList'] as $table) {
            $glueService->deleteTable($table['Name'], $databaseName);
        }

        echo "Delete the databases.\n";
        $glueClient->deleteDatabase([
            'Name' => $databaseName,
        ]);

        echo "Delete the crawler.\n";
        $glueClient->deleteCrawler([
            'Name' => $crawlerName,
        ]);

        $deleteObjects = $s3client->listObjectsV2([
            'Bucket' => $bucketName,
        ]);
        echo "Delete all objects in the bucket.\n";
        $deleteObjects = $s3client->deleteObjects([
            'Bucket' => $bucketName,
            'Delete' => [
                'Objects' => $deleteObjects['Contents'],
            ]
        ]);
        echo "Delete the bucket.\n";
        $s3client->deleteBucket(['Bucket' => $bucketName]);

        echo "This job was brought to you by the number $uniqid\n";
    }
```

```
}

namespace Glue;

use Aws\Glue\GlueClient;
use Aws\Result;

use function PHPUnit\Framework\isEmpty;

class GlueService extends \AwsUtilities\AWSServiceClass
{
    protected GlueClient $glueClient;

    public function __construct($glueClient)
    {
        $this->glueClient = $glueClient;
    }

    public function getCrawler($crawlerName)
    {
        return $this->customWaiter(function () use ($crawlerName) {
            return $this->glueClient->getCrawler([
                'Name' => $crawlerName,
            ]);
        });
    }

    public function createCrawler($crawlerName, $role, $databaseName, $path):
 Result
    {
        return $this->customWaiter(function () use ($crawlerName, $role,
 $databaseName, $path) {
            return $this->glueClient->createCrawler([
                'Name' => $crawlerName,
                'Role' => $role,
                'DatabaseName' => $databaseName,
                'Targets' => [
                    'S3Targets' =>
                        [[
                            'Path' => $path,
                        ]]
                ],
            ]);
        });
```

```
    }

    public function startCrawler($crawlerName): Result
    {
        return $this->glueClient->startCrawler([
            'Name' => $crawlerName,
        ]);
    }

    public function getDatabase(string $databaseName): Result
    {
        return $this->customWaiter(function () use ($databaseName) {
            return $this->glueClient->getDatabase([
                'Name' => $databaseName,
            ]);
        });
    }

    public function getTables($databaseName): Result
    {
        return $this->glueClient->getTables([
            'DatabaseName' => $databaseName,
        ]);
    }

    public function createJob($jobName, $role, $scriptLocation, $pythonVersion =
'3', $glueVersion = '3.0'): Result
    {
        return $this->glueClient->createJob([
            'Name' => $jobName,
            'Role' => $role,
            'Command' => [
                'Name' => 'glueetl',
                'ScriptLocation' => $scriptLocation,
                'PythonVersion' => $pythonVersion,
            ],
            'GlueVersion' => $glueVersion,
        ]);
    }

    public function startJobRun($jobName, $databaseName, $tables,
$outputBucketUrl): Result
    {
        return $this->glueClient->startJobRun([
```

```
            'JobName' => $jobName,
            'Arguments' => [
                'input_database' => $databaseName,
                'input_table' => $tables['TableList'][0]['Name'],
                'output_bucket_url' => $outputBucketUrl,
                '--input_database' => $databaseName,
                '--input_table' => $tables['TableList'][0]['Name'],
                '--output_bucket_url' => $outputBucketUrl,
            ],
        ]);
    }

    public function listJobs($maxResults = null, $nextToken = null, $tags = []):
Result
    {
        $arguments = [];
        if ($maxResults) {
            $arguments['MaxResults'] = $maxResults;
        }
        if ($nextToken) {
            $arguments['NextToken'] = $nextToken;
        }
        if (!empty($tags)) {
            $arguments['Tags'] = $tags;
        }
        return $this->glueClient->listJobs($arguments);
    }

    public function getJobRuns($jobName, $maxResults = 0, $nextToken = ''):
Result
    {
        $arguments = ['JobName' => $jobName];
        if ($maxResults) {
            $arguments['MaxResults'] = $maxResults;
        }
        if ($nextToken) {
            $arguments['NextToken'] = $nextToken;
        }
        return $this->glueClient->getJobRuns($arguments);
    }

    public function getJobRun($jobName, $runId, $predecessorsIncluded = false):
Result
    {
```

```
        return $this->glueClient->getJobRun([
            'JobName' => $jobName,
            'RunId' => $runId,
            'PredecessorsIncluded' => $predecessorsIncluded,
        ]);
    }

    public function deleteJob($jobName)
    {
        return $this->glueClient->deleteJob([
            'JobName' => $jobName,
        ]);
    }

    public function deleteTable($tableName, $databaseName)
    {
        return $this->glueClient->deleteTable([
            'DatabaseName' => $databaseName,
            'Name' => $tableName,
        ]);
    }

    public function deleteDatabase($databaseName)
    {
        return $this->glueClient->deleteDatabase([
            'Name' => $databaseName,
        ]);
    }

    public function deleteCrawler($crawlerName)
    {
        return $this->glueClient->deleteCrawler([
            'Name' => $crawlerName,
        ]);
    }
}
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for PHP .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler

- DeleteDatabase
- DeleteJob
- DeleteTable
- GetCrawler
- GetDatabase
- GetDatabases
- GetJob
- GetJobRun
- GetJobRuns
- GetTables
- ListJobs
- StartCrawler
- StartJobRun

Python

SDK untuk Python (Boto3)

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat kelas yang membungkus AWS Glue fungsi yang digunakan dalam skenario.

```
class GlueWrapper:
    """Encapsulates AWS Glue actions."""

    def __init__(self, glue_client):
        """
        :param glue_client: A Boto3 Glue client.
        """
        self.glue_client = glue_client
```

```
    def get_crawler(self, name):
        """
        Gets information about a crawler.

        :param name: The name of the crawler to look up.
        :return: Data about the crawler.
        """
        crawler = None
        try:
            response = self.glue_client.get_crawler(Name=name)
            crawler = response["Crawler"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityNotFoundException":
                logger.info("Crawler %s doesn't exist.", name)
            else:
                logger.error(
                    "Couldn't get crawler %s. Here's why: %s: %s",
                    name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return crawler


    def create_crawler(self, name, role_arn, db_name, db_prefix, s3_target):
        """
        Creates a crawler that can crawl the specified target and populate a
        database in your AWS Glue Data Catalog with metadata that describes the
data
        in the target.

        :param name: The name of the crawler.
        :param role_arn: The Amazon Resource Name (ARN) of an AWS Identity and
Access
                         Management (IAM) role that grants permission to let AWS
Glue
                         access the resources it needs.
        :param db_name: The name to give the database that is created by the
crawler.
        :param db_prefix: The prefix to give any database tables that are created
by
                          the crawler.
        :param s3_target: The URL to an S3 bucket that contains data that is
```

```
                        the target of the crawler.
        """
        try:
            self.glue_client.create_crawler(
                Name=name,
                Role=role_arn,
                DatabaseName=db_name,
                TablePrefix=db_prefix,
                Targets={"S3Targets": [{"Path": s3_target}]},
            )
        except ClientError as err:
            logger.error(
                "Couldn't create crawler. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def start_crawler(self, name):
        """
        Starts a crawler. The crawler crawls its configured target and creates
        metadata that describes the data it finds in the target data source.

        :param name: The name of the crawler to start.
        """
        try:
            self.glue_client.start_crawler(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't start crawler %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def get_database(self, name):
        """
        Gets information about a database in your Data Catalog.

        :param name: The name of the database to look up.
        :return: Information about the database.
```

```
        """
        try:
            response = self.glue_client.get_database(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't get database %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["Database"]


    def get_tables(self, db_name):
        """
        Gets a list of tables in a Data Catalog database.

        :param db_name: The name of the database to query.
        :return: The list of tables in the database.
        """
        try:
            response = self.glue_client.get_tables(DatabaseName=db_name)
        except ClientError as err:
            logger.error(
                "Couldn't get tables %s. Here's why: %s: %s",
                db_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["TableList"]


    def create_job(self, name, description, role_arn, script_location):
        """
        Creates a job definition for an extract, transform, and load (ETL) job
that can
        be run by AWS Glue.

        :param name: The name of the job definition.
        :param description: The description of the job definition.
```

```
        :param role_arn: The ARN of an IAM role that grants AWS Glue the
permissions
                         it requires to run the job.
        :param script_location: The Amazon S3 URL of a Python ETL script that is
run as
                                part of the job. The script defines how the data
is
                                transformed.
        """
        try:
            self.glue_client.create_job(
                Name=name,
                Description=description,
                Role=role_arn,
                Command={
                    "Name": "glueetl",
                    "ScriptLocation": script_location,
                    "PythonVersion": "3",
                },
                GlueVersion="3.0",
            )
        except ClientError as err:
            logger.error(
                "Couldn't create job %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def start_job_run(self, name, input_database, input_table,
output_bucket_name):
        """
        Starts a job run. A job run extracts data from the source, transforms it,
        and loads it to the output bucket.

        :param name: The name of the job definition.
        :param input_database: The name of the metadata database that contains
tables
                               that describe the source data. This is typically
created
                               by a crawler.
        :param input_table: The name of the table in the metadata database that
```

```
                            describes the source data.
        :param output_bucket_name: The S3 bucket where the output is written.
        :return: The ID of the job run.
        """
        try:
            # The custom Arguments that are passed to this function are used by
the
            # Python ETL script to determine the location of input and output
data.
            response = self.glue_client.start_job_run(
                JobName=name,
                Arguments={
                    "--input_database": input_database,
                    "--input_table": input_table,
                    "--output_bucket_url": f"s3://{output_bucket_name}/",
                },
            )
        except ClientError as err:
            logger.error(
                "Couldn't start job run %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRunId"]


    def list_jobs(self):
        """
        Lists the names of job definitions in your account.

        :return: The list of job definition names.
        """
        try:
            response = self.glue_client.list_jobs()
        except ClientError as err:
            logger.error(
                "Couldn't list jobs. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

```
        else:
            return response["JobNames"]


    def get_job_runs(self, job_name):
        """
        Gets information about runs that have been performed for a specific job
        definition.

        :param job_name: The name of the job definition to look up.
        :return: The list of job runs.
        """
        try:
            response = self.glue_client.get_job_runs(JobName=job_name)
        except ClientError as err:
            logger.error(
                "Couldn't get job runs for %s. Here's why: %s: %s",
                job_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return response["JobRuns"]


    def get_job_run(self, name, run_id):
        """
        Gets information about a single job run.

        :param name: The name of the job definition for the run.
        :param run_id: The ID of the run.
        :return: Information about the run.
        """
        try:
            response = self.glue_client.get_job_run(JobName=name, RunId=run_id)
        except ClientError as err:
            logger.error(
                "Couldn't get job run %s/%s. Here's why: %s: %s",
                name,
                run_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
```

```
            raise
        else:
            return response["JobRun"]


    def delete_job(self, job_name):
        """
        Deletes a job definition. This also deletes data about all runs that are
        associated with this job definition.

        :param job_name: The name of the job definition to delete.
        """
        try:
            self.glue_client.delete_job(JobName=job_name)
        except ClientError as err:
            logger.error(
                "Couldn't delete job %s. Here's why: %s: %s",
                job_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete_table(self, db_name, table_name):
        """
        Deletes a table from a metadata database.

        :param db_name: The name of the database that contains the table.
        :param table_name: The name of the table to delete.
        """
        try:
            self.glue_client.delete_table(DatabaseName=db_name, Name=table_name)
        except ClientError as err:
            logger.error(
                "Couldn't delete table %s. Here's why: %s: %s",
                table_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete_database(self, name):
```

```
        """
        Deletes a metadata database from your Data Catalog.

        :param name: The name of the database to delete.
        """
        try:
            self.glue_client.delete_database(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't delete database %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise


    def delete_crawler(self, name):
        """
        Deletes a crawler.

        :param name: The name of the crawler to delete.
        """
        try:
            self.glue_client.delete_crawler(Name=name)
        except ClientError as err:
            logger.error(
                "Couldn't delete crawler %s. Here's why: %s: %s",
                name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

Buat kelas yang menjalankan skenario.

```
class GlueCrawlerJobScenario:
    """
```

```
    Encapsulates a scenario that shows how to create an AWS Glue crawler and job
and use
    them to transform data from CSV to JSON format.
    """

    def __init__(self, glue_client, glue_service_role, glue_bucket):
        """
        :param glue_client: A Boto3 AWS Glue client.
        :param glue_service_role: An AWS Identity and Access Management (IAM)
role
                                  that AWS Glue can assume to gain access to the
                                  resources it requires.
        :param glue_bucket: An S3 bucket that can hold a job script and output
data
                            from AWS Glue job runs.
        """
        self.glue_client = glue_client
        self.glue_service_role = glue_service_role
        self.glue_bucket = glue_bucket

    @staticmethod
    def wait(seconds, tick=12):
        """
        Waits for a specified number of seconds, while also displaying an
animated
        spinner.

        :param seconds: The number of seconds to wait.
        :param tick: The number of frames per second used to animate the spinner.
        """
        progress = "|/-\\"
        waited = 0
        while waited < seconds:
            for frame in range(tick):
                sys.stdout.write(f"\r{progress[frame % len(progress)]}")
                sys.stdout.flush()
                time.sleep(1 / tick)
            waited += 1

    def upload_job_script(self, job_script):
        """
        Uploads a Python ETL script to an S3 bucket. The script is used by the
AWS Glue
        job to transform data.
```

```
        :param job_script: The relative path to the job script.
        """
        try:
            self.glue_bucket.upload_file(Filename=job_script, Key=job_script)
            print(f"Uploaded job script '{job_script}' to the example bucket.")
        except S3UploadFailedError as err:
            logger.error("Couldn't upload job script. Here's why: %s", err)
            raise

    def run(self, crawler_name, db_name, db_prefix, data_source, job_script,
job_name):
        """
        Runs the scenario. This is an interactive experience that runs at a
command
        prompt and asks you for input throughout.

        :param crawler_name: The name of the crawler used in the scenario. If the
                             crawler does not exist, it is created.
        :param db_name: The name to give the metadata database created by the
crawler.
        :param db_prefix: The prefix to give tables added to the database by the
                          crawler.
        :param data_source: The location of the data source that is targeted by
the
                            crawler and extracted during job runs.
        :param job_script: The job script that is used to transform data during
job
                           runs.
        :param job_name: The name to give the job definition that is created
during the
                         scenario.
        """
        wrapper = GlueWrapper(self.glue_client)
        print(f"Checking for crawler {crawler_name}.")
        crawler = wrapper.get_crawler(crawler_name)
        if crawler is None:
            print(f"Creating crawler {crawler_name}.")
            wrapper.create_crawler(
                crawler_name,
                self.glue_service_role.arn,
                db_name,
                db_prefix,
                data_source,
```

```
            )
            print(f"Created crawler {crawler_name}.")
            crawler = wrapper.get_crawler(crawler_name)
        pprint(crawler)
        print("-" * 88)

        print(
            f"When you run the crawler, it crawls data stored in {data_source}
and "
            f"creates a metadata database in the AWS Glue Data Catalog that
describes "
            f"the data in the data source."
        )
        print("In this example, the source data is in CSV format.")
        ready = False
        while not ready:
            ready = Question.ask_question(
                "Ready to start the crawler? (y/n) ", Question.is_yesno
            )
        wrapper.start_crawler(crawler_name)
        print("Let's wait for the crawler to run. This typically takes a few
minutes.")
        crawler_state = None
        while crawler_state != "READY":
            self.wait(10)
            crawler = wrapper.get_crawler(crawler_name)
            crawler_state = crawler["State"]
            print(f"Crawler is {crawler['State']}.")
        print("-" * 88)

        database = wrapper.get_database(db_name)
        print(f"The crawler created database {db_name}:")
        pprint(database)
        print(f"The database contains these tables:")
        tables = wrapper.get_tables(db_name)
        for index, table in enumerate(tables):
            print(f"\t{index + 1}. {table['Name']}")
        table_index = Question.ask_question(
            f"Enter the number of a table to see more detail: ",
            Question.is_int,
            Question.in_range(1, len(tables)),
        )
        pprint(tables[table_index - 1])
        print("-" * 88)
```

```
        print(f"Creating job definition {job_name}.")
        wrapper.create_job(
            job_name,
            "Getting started example job.",
            self.glue_service_role.arn,
            f"s3://{self.glue_bucket.name}/{job_script}",
        )
        print("Created job definition.")
        print(
            f"When you run the job, it extracts data from {data_source},
transforms it "
            f"by using the {job_script} script, and loads the output into "
            f"S3 bucket {self.glue_bucket.name}."
        )
        print(
            "In this example, the data is transformed from CSV to JSON, and only
a few "
            "fields are included in the output."
        )
        job_run_status = None
        if Question.ask_question(f"Ready to run? (y/n) ", Question.is_yesno):
            job_run_id = wrapper.start_job_run(
                job_name, db_name, tables[0]["Name"], self.glue_bucket.name
            )
            print(f"Job {job_name} started. Let's wait for it to run.")
            while job_run_status not in ["SUCCEEDED", "STOPPED", "FAILED",
"TIMEOUT"]:
                self.wait(10)
                job_run = wrapper.get_job_run(job_name, job_run_id)
                job_run_status = job_run["JobRunState"]
                print(f"Job {job_name}/{job_run_id} is {job_run_status}.")
        print("-" * 88)

        if job_run_status == "SUCCEEDED":
            print(
                f"Data from your job run is stored in your S3 bucket
'{self.glue_bucket.name}':"
            )
            try:
                keys = [
                    obj.key for obj in
self.glue_bucket.objects.filter(Prefix="run-")
                ]
```

```
                for index, key in enumerate(keys):
                    print(f"\t{index + 1}: {key}")
                lines = 4
                key_index = Question.ask_question(
                    f"Enter the number of a block to download it and see the
first {lines} "
                    f"lines of JSON output in the block: ",
                    Question.is_int,
                    Question.in_range(1, len(keys)),
                )
                job_data = io.BytesIO()
                self.glue_bucket.download_fileobj(keys[key_index - 1], job_data)
                job_data.seek(0)
                for _ in range(lines):
                    print(job_data.readline().decode("utf-8"))
            except ClientError as err:
                logger.error(
                    "Couldn't get job run data. Here's why: %s: %s",
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
            print("-" * 88)

        job_names = wrapper.list_jobs()
        if job_names:
            print(f"Your account has {len(job_names)} jobs defined:")
            for index, job_name in enumerate(job_names):
                print(f"\t{index + 1}. {job_name}")
            job_index = Question.ask_question(
                f"Enter a number between 1 and {len(job_names)} to see the list
of runs for "
                f"a job: ",
                Question.is_int,
                Question.in_range(1, len(job_names)),
            )
            job_runs = wrapper.get_job_runs(job_names[job_index - 1])
            if job_runs:
                print(f"Found {len(job_runs)} runs for job {job_names[job_index -
1]}:")
                for index, job_run in enumerate(job_runs):
                    print(
                        f"\t{index + 1}. {job_run['JobRunState']} on "
                        f"{job_run['CompletedOn']:%Y-%m-%d %H:%M:%S}"
```

```
                    )
                run_index = Question.ask_question(
                    f"Enter a number between 1 and {len(job_runs)} to see details
for a run: ",
                    Question.is_int,
                    Question.in_range(1, len(job_runs)),
                )
                pprint(job_runs[run_index - 1])
            else:
                print(f"No runs found for job {job_names[job_index - 1]}")
        else:
            print("Your account doesn't have any jobs defined.")
        print("-" * 88)

        print(
            f"Let's clean up. During this example we created job definition
'{job_name}'."
        )
        if Question.ask_question(
            "Do you want to delete the definition and all runs? (y/n) ",
            Question.is_yesno,
        ):
            wrapper.delete_job(job_name)
            print(f"Job definition '{job_name}' deleted.")
        tables = wrapper.get_tables(db_name)
        print(f"We also created database '{db_name}' that contains these
tables:")
        for table in tables:
            print(f"\t{table['Name']}")
        if Question.ask_question(
            "Do you want to delete the tables and the database? (y/n) ",
            Question.is_yesno,
        ):
            for table in tables:
                wrapper.delete_table(db_name, table["Name"])
                print(f"Deleted table {table['Name']}.")
            wrapper.delete_database(db_name)
            print(f"Deleted database {db_name}.")
        print(f"We also created crawler '{crawler_name}'.")
        if Question.ask_question(
            "Do you want to delete the crawler? (y/n) ", Question.is_yesno
        ):
            wrapper.delete_crawler(crawler_name)
            print(f"Deleted crawler {crawler_name}.")
```

```
        print("-" * 88)


def parse_args(args):
    """
    Parse command line arguments.

    :param args: The command line arguments.
    :return: The parsed arguments.
    """
    parser = argparse.ArgumentParser(
        description="Runs the AWS Glue getting started with crawlers and jobs
 scenario. "
        "Before you run this scenario, set up scaffold resources by running "
        "'python scaffold.py deploy'."
    )
    parser.add_argument(
        "role_name",
        help="The name of an IAM role that AWS Glue can assume. This role must
 grant access "
        "to Amazon S3 and to the permissions granted by the AWSGlueServiceRole "
        "managed policy.",
    )
    parser.add_argument(
        "bucket_name",
        help="The name of an S3 bucket that AWS Glue can access to get the job
 script and "
        "put job results.",
    )
    parser.add_argument(
        "--job_script",
        default="flight_etl_job_script.py",
        help="The name of the job script file that is used in the scenario.",
    )
    return parser.parse_args(args)


def main():
    args = parse_args(sys.argv[1:])
    try:
        print("-" * 88)
        print(
            "Welcome to the AWS Glue getting started with crawlers and jobs
 scenario."
```

```
        )
        print("-" * 88)
        scenario = GlueCrawlerJobScenario(
            boto3.client("glue"),
            boto3.resource("iam").Role(args.role_name),
            boto3.resource("s3").Bucket(args.bucket_name),
        )
        scenario.upload_job_script(args.job_script)
        scenario.run(
            "doc-example-crawler",
            "doc-example-database",
            "doc-example-",
            "s3://crawler-public-us-east-1/flight/2016/csv",
            args.job_script,
            "doc-example-job",
        )
        print("-" * 88)
        print(
            "To destroy scaffold resources, including the IAM role and S3 bucket
 "
            "used in this scenario, run 'python scaffold.py destroy'."
        )
        print("\nThanks for watching!")
        print("-" * 88)
    except Exception:
        logging.exception("Something went wrong with the example.")
```

Buat skrip ETL yang digunakan oleh AWS Glue untuk mengekstrak, mengubah, dan memuat data selama pekerjaan berjalan.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

"""
These custom arguments must be passed as Arguments to the StartJobRun request.
```

```
    --input_database    The name of a metadata database that is contained in
 your
                        AWS Glue Data Catalog and that contains tables that
 describe
                        the data to be processed.
    --input_table       The name of a table in the database that describes the
 data to
                        be processed.
    --output_bucket_url An S3 bucket that receives the transformed output data.
"""
args = getResolvedOptions(
    sys.argv, ["JOB_NAME", "input_database", "input_table", "output_bucket_url"]
)
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args["JOB_NAME"], args)

# Script generated for node S3 Flight Data.
S3FlightData_node1 = glueContext.create_dynamic_frame.from_catalog(
    database=args["input_database"],
    table_name=args["input_table"],
    transformation_ctx="S3FlightData_node1",
)

# This mapping performs two main functions:
# 1. It simplifies the output by removing most of the fields from the data.
# 2. It renames some fields. For example, `fl_date` is renamed to `flight_date`.
ApplyMapping_node2 = ApplyMapping.apply(
    frame=S3FlightData_node1,
    mappings=[
        ("year", "long", "year", "long"),
        ("month", "long", "month", "tinyint"),
        ("day_of_month", "long", "day", "tinyint"),
        ("fl_date", "string", "flight_date", "string"),
        ("carrier", "string", "carrier", "string"),
        ("fl_num", "long", "flight_num", "long"),
        ("origin_city_name", "string", "origin_city_name", "string"),
        ("origin_state_abr", "string", "origin_state_abr", "string"),
        ("dest_city_name", "string", "dest_city_name", "string"),
        ("dest_state_abr", "string", "dest_state_abr", "string"),
        ("dep_time", "long", "departure_time", "long"),
        ("wheels_off", "long", "wheels_off", "long"),
```

```
        ("wheels_on", "long", "wheels_on", "long"),
        ("arr_time", "long", "arrival_time", "long"),
        ("mon", "string", "mon", "string"),
    ],
    transformation_ctx="ApplyMapping_node2",
)

# Script generated for node Revised Flight Data.
RevisedFlightData_node3 = glueContext.write_dynamic_frame.from_options(
    frame=ApplyMapping_node2,
    connection_type="s3",
    format="json",
    connection_options={"path": args["output_bucket_url"], "partitionKeys": []},
    transformation_ctx="RevisedFlightData_node3",
)

job.commit()
```

- Untuk detail API, lihat topik berikut ini adalah Referensi API SDK untuk Python (Boto3)AWS .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler

- StartJobRun

Ruby

SDK untuk Ruby

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Buat kelas yang membungkus AWS Glue fungsi yang digunakan dalam skenario.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing
 a simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods
 for interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
 calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves information about a specific crawler.
  #
  # @param name [String] The name of the crawler to retrieve information about.
  # @return [Aws::Glue::Types::Crawler, nil] The crawler object if found, or nil
 if not found.
  def get_crawler(name)
    @glue_client.get_crawler(name: name)
  rescue Aws::Glue::Errors::EntityNotFoundException
    @logger.info("Crawler #{name} doesn't exist.")
    false
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get crawler #{name}: \n#{e.message}")
    raise
  end
```

```
  # Creates a new crawler with the specified configuration.
  #
  # @param name [String] The name of the crawler.
  # @param role_arn [String] The ARN of the IAM role to be used by the crawler.
  # @param db_name [String] The name of the database where the crawler stores its
metadata.
  # @param db_prefix [String] The prefix to be added to the names of tables that
the crawler creates.
  # @param s3_target [String] The S3 path that the crawler will crawl.
  # @return [void]
  def create_crawler(name, role_arn, db_name, db_prefix, s3_target)
    @glue_client.create_crawler(
      name: name,
      role: role_arn,
      database_name: db_name,
      targets: {
        s3_targets: [
          {
            path: s3_target
          }
        ]
      }
    )
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not create crawler: \n#{e.message}")
    raise
  end

  # Starts a crawler with the specified name.
  #
  # @param name [String] The name of the crawler to start.
  # @return [void]
  def start_crawler(name)
    @glue_client.start_crawler(name: name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not start crawler #{name}: \n#{e.message}")
    raise
  end

  # Deletes a crawler with the specified name.
  #
  # @param name [String] The name of the crawler to delete.
  # @return [void]
  def delete_crawler(name)
```

```
    @glue_client.delete_crawler(name: name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete crawler #{name}: \n#{e.message}")
    raise
  end

  # Retrieves information about a specific database.
  #
  # @param name [String] The name of the database to retrieve information about.
  # @return [Aws::Glue::Types::Database, nil] The database object if found, or
 nil if not found.
  def get_database(name)
    response = @glue_client.get_database(name: name)
    response.database
rescue Aws::Glue::Errors::GlueException => e
  @logger.error("Glue could not get database #{name}: \n#{e.message}")
  raise
  end

  # Retrieves a list of tables in the specified database.
  #
  # @param db_name [String] The name of the database to retrieve tables from.
  # @return [Array<Aws::Glue::Types::Table>]
  def get_tables(db_name)
    response = @glue_client.get_tables(database_name: db_name)
    response.table_list
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get tables #{db_name}: \n#{e.message}")
    raise
  end

  # Creates a new job with the specified configuration.
  #
  # @param name [String] The name of the job.
  # @param description [String] The description of the job.
  # @param role_arn [String] The ARN of the IAM role to be used by the job.
  # @param script_location [String] The location of the ETL script for the job.
  # @return [void]
  def create_job(name, description, role_arn, script_location)
    @glue_client.create_job(
      name: name,
      description: description,
      role: role_arn,
      command: {
```

```
        name: "glueetl",
        script_location: script_location,
        python_version: "3"
      },
      glue_version: "3.0"
    )
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not create job #{name}: \n#{e.message}")
    raise
  end

  # Starts a job run for the specified job.
  #
  # @param name [String] The name of the job to start the run for.
  # @param input_database [String] The name of the input database for the job.
  # @param input_table [String] The name of the input table for the job.
  # @param output_bucket_name [String] The name of the output S3 bucket for the
job.
  # @return [String] The ID of the started job run.
  def start_job_run(name, input_database, input_table, output_bucket_name)
    response = @glue_client.start_job_run(
      job_name: name,
      arguments: {
        '--input_database': input_database,
        '--input_table': input_table,
        '--output_bucket_url': "s3://#{output_bucket_name}/"
      }
    )
    response.job_run_id
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not start job run #{name}: \n#{e.message}")
    raise
  end

  # Retrieves a list of jobs in AWS Glue.
  #
  # @return [Aws::Glue::Types::ListJobsResponse]
  def list_jobs
    @glue_client.list_jobs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not list jobs: \n#{e.message}")
    raise
  end
```

```
  # Retrieves a list of job runs for the specified job.
  #
  # @param job_name [String] The name of the job to retrieve job runs for.
  # @return [Array<Aws::Glue::Types::JobRun>]
  def get_job_runs(job_name)
    response = @glue_client.get_job_runs(job_name: job_name)
    response.job_runs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end

  # Retrieves data for a specific job run.
  #
  # @param job_name [String] The name of the job run to retrieve data for.
  # @return [Glue::Types::GetJobRunResponse]
  def get_job_run(job_name, run_id)
    @glue_client.get_job_run(job_name: job_name, run_id: run_id)
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end

  # Deletes a job with the specified name.
  #
  # @param job_name [String] The name of the job to delete.
  # @return [void]
  def delete_job(job_name)
    @glue_client.delete_job(job_name: job_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end

  # Deletes a table with the specified name.
  #
  # @param database_name [String] The name of the catalog database in which the
table resides.
  # @param table_name [String] The name of the table to be deleted.
  # @return [void]
  def delete_table(database_name, table_name)
    @glue_client.delete_table(database_name: database_name, name: table_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end

  # Removes a specified database from a Data Catalog.
```

```
  #
  # @param database_name [String] The name of the database to delete.
  # @return [void]
  def delete_database(database_name)
    @glue_client.delete_database(name: database_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete database: \n#{e.message}")
  end

  # Uploads a job script file to an S3 bucket.
  #
  # @param file_path [String] The local path of the job script file.
  # @param bucket_resource [Aws::S3::Bucket] The S3 bucket resource to upload the
 file to.
  # @return [void]
  def upload_job_script(file_path, bucket_resource)
    File.open(file_path) do |file|
      bucket_resource.client.put_object({
       body: file,
       bucket: bucket_resource.name,
       key: file_path
     })
    end
  rescue Aws::S3::Errors::S3UploadFailedError => e
    @logger.error("S3 could not upload job script: \n#{e.message}")
    raise
  end

end
```

Buat kelas yang menjalankan skenario.

```
class GlueCrawlerJobScenario
  def initialize(glue_client, glue_service_role, glue_bucket, logger)
    @glue_client = glue_client
    @glue_service_role = glue_service_role
    @glue_bucket = glue_bucket
    @logger = logger
  end

  def run(crawler_name, db_name, db_prefix, data_source, job_script, job_name)
    wrapper = GlueWrapper.new(@glue_client, @logger)
```

```
   new_step(1, "Create a crawler")
   puts "Checking for crawler #{crawler_name}."
   crawler = wrapper.get_crawler(crawler_name)
   if crawler == false
     puts "Creating crawler #{crawler_name}."
     wrapper.create_crawler(crawler_name, @glue_service_role.arn, db_name,
db_prefix, data_source)
     puts "Successfully created #{crawler_name}:"
     crawler = wrapper.get_crawler(crawler_name)
     puts JSON.pretty_generate(crawler).yellow
   end
   print "\nDone!\n".green

   new_step(2, "Run a crawler to output a database.")
   puts "Location of input data analyzed by crawler: #{data_source}"
   puts "Outputs: a Data Catalog database in CSV format containing metadata on
input."
   wrapper.start_crawler(crawler_name)
   puts "Starting crawler... (this typically takes a few minutes)"
   crawler_state = nil
   while crawler_state != "READY"
     custom_wait(15)
     crawler = wrapper.get_crawler(crawler_name)
     crawler_state = crawler[0]["state"]
     print "Status check: #{crawler_state}.".yellow
   end
   print "\nDone!\n".green

   new_step(3, "Query the database.")
   database = wrapper.get_database(db_name)
   puts "The crawler created database #{db_name}:"
   print "#{database}".yellow
   puts "\nThe database contains these tables:"
   tables = wrapper.get_tables(db_name)
   tables.each_with_index do |table, index|
     print "\t#{index + 1}. #{table['name']}".yellow
   end
   print "\nDone!\n".green

   new_step(4, "Create a job definition that runs an ETL script.")
   puts "Uploading Python ETL script to S3..."
   wrapper.upload_job_script(job_script, @glue_bucket)
   puts "Creating job definition #{job_name}:\n"
```

```
    response = wrapper.create_job(job_name, "Getting started example job.",
@glue_service_role.arn, "s3://#{@glue_bucket.name}/#{job_script}")
    puts JSON.pretty_generate(response).yellow
    print "\nDone!\n".green

    new_step(5, "Start a new job")
    job_run_status = nil
    job_run_id = wrapper.start_job_run(
      job_name,
      db_name,
      tables[0]["name"],
      @glue_bucket.name
    )
    puts "Job #{job_name} started. Let's wait for it to run."
    until ["SUCCEEDED", "STOPPED", "FAILED", "TIMEOUT"].include?(job_run_status)
      custom_wait(10)
      job_run = wrapper.get_job_runs(job_name)
      job_run_status = job_run[0]["job_run_state"]
      print "Status check: #{job_name}/#{job_run_id} - #{job_run_status}.".yellow
    end
    print "\nDone!\n".green

    new_step(6, "View results from a successful job run.")
    if job_run_status == "SUCCEEDED"
      puts "Data from your job run is stored in your S3 bucket
'#{@glue_bucket.name}'. Files include:"
      begin

        # Print the key name of each object in the bucket.
        @glue_bucket.objects.each do |object_summary|
          if object_summary.key.include?("run-")
            print "#{object_summary.key}".yellow
          end
        end

        # Print the first 256 bytes of a run file
        desired_sample_objects = 1
        @glue_bucket.objects.each do |object_summary|
          if object_summary.key.include?("run-")
            if desired_sample_objects > 0
              sample_object = @glue_bucket.object(object_summary.key)
              sample = sample_object.get(range: "bytes=0-255").body.read
              puts "\nSample run file contents:"
              print "#{sample}".yellow
```

```
              desired_sample_objects -= 1
            end
          end
        end
      rescue Aws::S3::Errors::ServiceError => e
        logger.error(
          "Couldn't get job run data. Here's why: %s: %s",
          e.response.error.code, e.response.error.message
        )
        raise
      end
    end
    print "\nDone!\n".green

    new_step(7, "Delete job definition and crawler.")
    wrapper.delete_job(job_name)
    puts "Job deleted: #{job_name}."
    wrapper.delete_crawler(crawler_name)
    puts "Crawler deleted: #{crawler_name}."
    wrapper.delete_table(db_name, tables[0]["name"])
    puts "Table deleted: #{tables[0]["name"]} in #{db_name}."
    wrapper.delete_database(db_name)
    puts "Database deleted: #{db_name}."
    print "\nDone!\n".green
  end
end

def main

  banner("../../helpers/banner.txt")
  puts
 "#########################################################################################
  puts "#
                            #".yellow
  puts "#                                          EXAMPLE CODE DEMO:
                            #".yellow
  puts "#                                              AWS Glue
                            #".yellow
  puts "#
                            #".yellow
  puts
 "#########################################################################################
  puts ""
```

```
 puts "You have launched a demo of AWS Glue using the AWS for Ruby v3 SDK. Over
the next 60 seconds, it will"
 puts "do the following:"
 puts "    1. Create a crawler."
 puts "    2. Run a crawler to output a database."
 puts "    3. Query the database."
 puts "    4. Create a job definition that runs an ETL script."
 puts "    5. Start a new job."
 puts "    6. View results from a successful job run."
 puts "    7. Delete job definition and crawler."
 puts ""

 confirm_begin
 billing
 security
 puts "\e[H\e[2J"

 # Set input file names
 job_script_filepath = "job_script.py"
 resource_names = YAML.load_file("resource_names.yaml")

 # Instantiate existing IAM role.
 iam = Aws::IAM::Resource.new(region: "us-east-1")
 iam_role_name = resource_names["glue_service_role"]
 iam_role = iam.role(iam_role_name)

 # Instantiate existing S3 bucket.
 s3 = Aws::S3::Resource.new(region: "us-east-1")
 s3_bucket_name = resource_names["glue_bucket"]
 s3_bucket = s3.bucket(s3_bucket_name)

 scenario = GlueCrawlerJobScenario.new(
   Aws::Glue::Client.new(region: "us-east-1"),
   iam_role,
   s3_bucket,
   @logger
 )

 random_int = rand(10 ** 4)
 scenario.run(
   "doc-example-crawler-#{random_int}",
   "doc-example-database-#{random_int}",
   "doc-example-#{random_int}-",
   "s3://crawler-public-us-east-1/flight/2016/csv",
```

```
    job_script_filepath,
    "doc-example-job-#{random_int}"
  )

  puts "-" * 88
  puts "You have reached the end of this tour of AWS Glue."
  puts "To destroy CDK-created resources, run:\n       cdk destroy"
  puts "-" * 88

end
```

Buat skrip ETL yang digunakan oleh AWS Glue untuk mengekstrak, mengubah, dan memuat data selama pekerjaan berjalan.

```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

"""
These custom arguments must be passed as Arguments to the StartJobRun request.
    --input_database    The name of a metadata database that is contained in
 your
                        AWS Glue Data Catalog and that contains tables that
 describe
                        the data to be processed.
    --input_table       The name of a table in the database that describes the
 data to
                        be processed.
    --output_bucket_url An S3 bucket that receives the transformed output data.
"""
args = getResolvedOptions(
    sys.argv, ["JOB_NAME", "input_database", "input_table", "output_bucket_url"]
)
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args["JOB_NAME"], args)
```

```
# Script generated for node S3 Flight Data.
S3FlightData_node1 = glueContext.create_dynamic_frame.from_catalog(
    database=args["input_database"],
    table_name=args["input_table"],
    transformation_ctx="S3FlightData_node1",
)

# This mapping performs two main functions:
# 1. It simplifies the output by removing most of the fields from the data.
# 2. It renames some fields. For example, `fl_date` is renamed to `flight_date`.
ApplyMapping_node2 = ApplyMapping.apply(
    frame=S3FlightData_node1,
    mappings=[
        ("year", "long", "year", "long"),
        ("month", "long", "month", "tinyint"),
        ("day_of_month", "long", "day", "tinyint"),
        ("fl_date", "string", "flight_date", "string"),
        ("carrier", "string", "carrier", "string"),
        ("fl_num", "long", "flight_num", "long"),
        ("origin_city_name", "string", "origin_city_name", "string"),
        ("origin_state_abr", "string", "origin_state_abr", "string"),
        ("dest_city_name", "string", "dest_city_name", "string"),
        ("dest_state_abr", "string", "dest_state_abr", "string"),
        ("dep_time", "long", "departure_time", "long"),
        ("wheels_off", "long", "wheels_off", "long"),
        ("wheels_on", "long", "wheels_on", "long"),
        ("arr_time", "long", "arrival_time", "long"),
        ("mon", "string", "mon", "string"),
    ],
    transformation_ctx="ApplyMapping_node2",
)

# Script generated for node Revised Flight Data.
RevisedFlightData_node3 = glueContext.write_dynamic_frame.from_options(
    frame=ApplyMapping_node2,
    connection_type="s3",
    format="json",
    connection_options={"path": args["output_bucket_url"], "partitionKeys": []},
    transformation_ctx="RevisedFlightData_node3",
)

job.commit()
```

- Untuk detail API, lihat topik berikut di Referensi API AWS SDK for Ruby .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler
  - StartJobRun

Rust

SDK untuk Rust

> **Note**
>
> Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di AWS Repositori Contoh Kode.

Membuat dan menjalankan crawler yang merayapi bucket Amazon Simple Storage Service (Amazon S3) publik dan menghasilkan database metadata yang menjelaskan data berformat CSV yang ditemukannya.

```
        let create_crawler = glue
            .create_crawler()
```

```
            .name(self.crawler())
            .database_name(self.database())
            .role(self.iam_role.expose_secret())
            .targets(
                CrawlerTargets::builder()
                    .s3_targets(S3Target::builder().path(CRAWLER_TARGET).build())
                    .build(),
            )
            .send()
            .await;

        match create_crawler {
            Err(err) => {
                let glue_err: aws_sdk_glue::Error = err.into();
                match glue_err {
                    aws_sdk_glue::Error::AlreadyExistsException(_) => {
                        info!("Using existing crawler");
                        Ok(())
                    }
                    _ => Err(GlueMvpError::GlueSdk(glue_err)),
                }
            }
            Ok(_) => Ok(()),
        }?;

        let start_crawler =
glue.start_crawler().name(self.crawler()).send().await;

        match start_crawler {
            Ok(_) => Ok(()),
            Err(err) => {
                let glue_err: aws_sdk_glue::Error = err.into();
                match glue_err {
                    aws_sdk_glue::Error::CrawlerRunningException(_) => Ok(()),
                    _ => Err(GlueMvpError::GlueSdk(glue_err)),
                }
            }
        }?;
```

Daftar informasi tentang database dan tabel di situs Anda AWS Glue Data Catalog.

```
        let database = glue
```

```
            .get_database()
            .name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?
            .to_owned();
        let database = database
            .database()
            .ok_or_else(|| GlueMvpError::Unknown("Could not find
database".into()))?;

        let tables = glue
            .get_tables()
            .database_name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let tables = tables.table_list();
```

Buat dan jalankan job yang mengekstrak data CSV dari bucket Amazon S3 sumber, mengubahnya dengan menghapus dan mengganti nama bidang, dan memuat output berformat JSON ke bucket Amazon S3 lainnya.

```
        let create_job = glue
            .create_job()
            .name(self.job())
            .role(self.iam_role.expose_secret())
            .command(
                JobCommand::builder()
                    .name("glueetl")
                    .python_version("3")
                    .script_location(format!("s3://{}/job.py", self.bucket()))
                    .build(),
            )
            .glue_version("3.0")
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let job_name = create_job.name().ok_or_else(|| {
```

```
            GlueMvpError::Unknown("Did not get job name after creating
job".into())
        })?;

        let job_run_output = glue
            .start_job_run()
            .job_name(self.job())
            .arguments("--input_database", self.database())
            .arguments(
                "--input_table",
                self.tables
                    .first()
                    .ok_or_else(|| GlueMvpError::Unknown("Missing crawler
table".into()))?
                    .name(),
            )
            .arguments("--output_bucket_url", self.bucket())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        let job = job_run_output
            .job_run_id()
            .ok_or_else(|| GlueMvpError::Unknown("Missing run id from just
started job".into()))?
            .to_string();
```

Hapus semua sumber daya yang dibuat oleh demo.

```
        glue.delete_job()
            .job_name(self.job())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        for t in &self.tables {
            glue.delete_table()
                .name(t.name())
                .database_name(self.database())
                .send()
                .await
                .map_err(GlueMvpError::from_glue_sdk)?;
```

```
        }

        glue.delete_database()
            .name(self.database())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;

        glue.delete_crawler()
            .name(self.crawler())
            .send()
            .await
            .map_err(GlueMvpError::from_glue_sdk)?;
```

- Untuk detail API, lihat topik berikut dalam referensi API SDK untuk Rust AWS .
  - CreateCrawler
  - CreateJob
  - DeleteCrawler
  - DeleteDatabase
  - DeleteJob
  - DeleteTable
  - GetCrawler
  - GetDatabase
  - GetDatabases
  - GetJob
  - GetJobRun
  - GetJobRuns
  - GetTables
  - ListJobs
  - StartCrawler
  - StartJobRun

Untuk daftar lengkap panduan pengembang AWS SDK dan contoh kode, lihatMenggunakan layanan ini dengan AWS SDK. Topik ini juga mencakup informasi tentang memulai dan detail tentang versi SDK sebelumnya.

Keamanan di AWS Glue

Keamanan cloud di AWS merupakan prioritas tertinggi. Sebagai pelanggan AWS, Anda akan mendapatkan manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara AWS dan Anda. Model tanggung jawab bersama menggambarkan ini sebagai keamanan dari cloud dan keamanan di dalam cloud:

- Keamanan cloud – AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan layanan AWS di dalam AWS Cloud. AWS juga memberi layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari program kepatuhan AWS. Untuk mempelajari program kepatuhan yang berlaku di AWS Glue, lihat Cakupan Layanan Menurut Program Kepatuhan AWS.
- Keamanan di cloud – Tanggung jawab Anda ditentukan menurut layanan AWS yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain termasuk sensitivitas data Anda, persyaratan perusahaan Anda, serta hukum dan peraturan yang berlaku.

Dokumentasi ini akan membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan AWS Glue. Topik berikut akan menunjukkan kepada Anda cara membuat konfigurasi AWS Glue untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelajari cara menggunakan layanan AWS lain yang membantu Anda memantau dan mengamankan sumber daya AWS Glue Anda.

Topik

Perlindungan data di AWS Glue

AWS Glue menawarkan beberapa fitur yang dirancang untuk membantu melindungi data Anda.

Topik

Enkripsi diam

AWS Glue mendukung enkripsi data saat data tidak berpindah untuk Membangun pekerjaan ETL visual dengan AWS Glue Studio dan Mengembangkan skrip menggunakan titik akhir pengembangan. Anda dapat mengkonfigurasi tugas extract, transform, and load (ETL) dan titik akhir pengembangan untuk menggunakan kunci AWS Key Management Service (AWS KMS) untuk menulis data at rest terenkripsi. Anda juga dapat mengenkripsi metadata yang disimpan di AWS Glue Data Catalog dengan menggunakan kunci yang Anda kelola dengan AWS KMS. Selain itu, Anda dapat menggunakan kunci AWS KMS untuk mengenkripsi bookmark tugas dan log yang dihasilkan oleh crawler dan tugas ETL.

Anda dapat mengenkripsi objek metadata AWS Glue Data Catalog di samping data yang ditulis ke Amazon Simple Storage Service (Amazon S3) dan CloudWatch Amazon Logs berdasarkan pekerjaan, crawler, dan titik akhir pengembangan. Saat Anda membuat tugas, crawler, dan titik akhir pengembangan di AWS Glue, Anda dapat memberikan pengaturan enkripsi dengan melampirkan sebuah konfigurasi keamanan. Konfigurasi keamanan berisi kunci enkripsi sisi server yang terkelola Amazon S3 (SSE-S3) atau kunci utama pelanggan (CMC) yang disimpan di AWS KMS (SSE-KM). Anda dapat membuat konfigurasi keamanan menggunakan konsol AWS Glue.

Anda juga dapat mengaktifkan enkripsi seluruh Katalog Data di akun Anda. Anda dapat melakukannya dengan menentukan CMK disimpan di AWS KMS.

> **Important**
>
> AWS Gluehanya mendukung kunci yang dikelola pelanggan simetris. Untuk informasi selengkapnya, lihat Customer Managed Keys (CMK) di Panduan AWS Key Management Service Pengembang.

Dengan enkripsi yang diaktifkan, ketika Anda menambahkan objek Katalog Data, menjalankan crawler, menjalankan tugas, atau memulai titik akhir pengembangan, kunci SSE-S3 atau SSE-KMS digunakan untuk menulis data at rest. Selain itu, Anda dapat mengonfigurasi AWS Glue untuk hanya mengakses penyimpanan data Java Database Connectivity (JDBC) melalui protokol Transport Layer Security (TLS) tepercaya.

Di AWS Glue, Anda mengendalikan pengaturan enkripsi di tempat-tempat berikut:

- Pengaturan Katalog Data Anda.
- Konfigurasi keamanan yang Anda buat.
- Pengaturan enkripsi sisi server (SSE-S3 atau SSE-KMS) yang diberikan sebagai parameter untuk tugas ETL (extract, transform, and load) AWS Glue Anda.

Untuk informasi selengkapnya tentang cara menyiapkan enkripsi, lihat Menyiapkan enkripsi di AWS Glue.

Topik

- Mengenkripsi Katalog Data Anda
- Mengenkripsi kata sandi koneksi
- Mengenkripsi data yang ditulis oleh AWS Glue

Mengenkripsi Katalog Data Anda

AWS Glue Data Catalogenkripsi memberikan keamanan yang ditingkatkan untuk data sensitif Anda. AWS Glueterintegrasi dengan AWS Key Management Service (AWS KMS) untuk mengenkripsi metadata yang disimpan dalam Katalog Data. Anda dapat mengaktifkan atau menonaktifkan pengaturan enkripsi untuk sumber daya di Katalog Data menggunakan AWS Glue konsol atauAWS CLI.

Saat Anda mengaktifkan enkripsi untuk Katalog Data Anda, semua objek baru yang Anda buat akan dienkripsi. Saat Anda menonaktifkan enkripsi, objek baru yang Anda buat tidak akan dienkripsi, tetapi objek terenkripsi yang ada akan tetap dienkripsi.

Anda dapat mengenkripsi seluruh Katalog Data menggunakan kunci enkripsi AWS terkelola atau kunci enkripsi yang dikelola pelanggan. Untuk informasi selengkapnya tentang tipe dan status utama, lihat AWS Key Management Servicekonsep di Panduan AWS Key Management Service Pengembang.

kunci terkelola AWS

AWSkunci terkelola adalah kunci KMS di akun Anda yang dibuat, dikelola, dan digunakan atas nama Anda oleh AWS layanan yang terintegrasi dengannyaAWS KMS. Anda dapat melihat kunci AWS terkelola di akun Anda, melihat kebijakan utamanya, dan mengaudit penggunaannya di AWS CloudTrail log. Namun, Anda tidak dapat mengelola kunci ini atau mengubah izinnya.

Enkripsi saat istirahat secara otomatis terintegrasi dengan AWS KMS untuk mengelola kunci AWS terkelola AWS Glue yang digunakan untuk mengenkripsi metadata Anda. Jika kunci AWS terkelola tidak ada saat Anda mengaktifkan enkripsi metadata, AWS KMS secara otomatis membuat kunci baru untuk Anda.

Untuk informasi selengkapnya, lihat kunci AWS terkelola.

Kunci yang dikelola pelanggan

Kunci yang dikelola pelanggan adalah kunci KMS Akun AWS yang Anda buat, miliki, dan kelola. Anda memiliki kontrol penuh atas kunci KMS ini. Anda dapat:

- Menetapkan dan memelihara kebijakan utama mereka, kebijakan IAM, dan hibah
- Aktifkan dan nonaktifkan
- Putar materi kriptografi mereka
- Tambahkan tag
- Buat alias yang merujuknya
- Jadwalkan mereka untuk dihapus

Untuk informasi selengkapnya tentang mengelola izin kunci terkelola pelanggan, lihat Kunci terkelola pelanggan.

> **Important**
>
> AWS Gluehanya mendukung kunci yang dikelola pelanggan simetris. Daftar kunci KMS hanya menampilkan tombol simetris. Namun, jika Anda memilih Pilih ARN kunci KMS, konsol memungkinkan Anda memasukkan ARN untuk semua jenis kunci. Pastikan bahwa Anda hanya memasukkan ARN untuk kunci simetris.
> Untuk membuat kunci terkelola pelanggan simetris, ikuti langkah-langkah untuk membuat kunci terkelola pelanggan simetris di Panduan AWS Key Management Service Pengembang.

Saat Anda mengaktifkan enkripsi Katalog Data saat istirahat, jenis sumber daya berikut dienkripsi menggunakan kunci KMS:

- Basis data
- Tabel
- Partisi
- Versi tabel
- Statistik kolom
- Fungsi yang ditetapkan pengguna
- Tampilan Katalog Data

AWS Gluekonteks enkripsi

Konteks enkripsi adalah kumpulan opsional pasangan kunci-nilai yang berisi informasi kontekstual tambahan tentang data. AWS KMSmenggunakan konteks enkripsi sebagai data otentikasi tambahan untuk mendukung enkripsi yang diautentikasi. Bila Anda menyertakan konteks enkripsi dalam permintaan untuk mengenkripsi data, AWS KMS mengikat konteks enkripsi ke data terenkripsi. Untuk mendekripsi data, Anda menyertakan konteks enkripsi yang sama dalam permintaan. AWS Gluemenggunakan konteks enkripsi yang sama di semua operasi AWS KMS kriptografi, di mana kuncinya `glue_catalog_id` dan nilainya adalah. `catalogId`

```
"encryptionContext": {
    "glue_catalog_id": "111122223333"
}
```

Bila Anda menggunakan kunci AWS terkelola atau kunci terkelola pelanggan simetris untuk mengenkripsi Katalog Data Anda, Anda juga dapat menggunakan konteks enkripsi dalam catatan audit dan log untuk mengidentifikasi bagaimana kunci tersebut digunakan. Konteks enkripsi juga muncul di log yang dihasilkan oleh AWS CloudTrail atau Amazon CloudWatch log.

Mengaktifkan enkripsi

Anda dapat mengaktifkan enkripsi untuk AWS Glue Data Catalog objek Anda di pengaturan Katalog Data di AWS Glue konsol atau dengan menggunakanAWS CLI.

Console

Untuk mengaktifkan enkripsi menggunakan konsol

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Pilih Katalog Data di panel navigasi.
3. Pada halaman pengaturan Katalog Data, pilih kotak centang Enkripsi metadata, dan pilih kunci. AWS KMS

   Bila Anda mengaktifkan enkripsi, jika Anda tidak menentukan kunci terkelola pelanggan, pengaturan enkripsi menggunakan kunci KMS AWS terkelola.
4. (Opsional) Saat Anda menggunakan kunci yang dikelola pelanggan untuk mengenkripsi Katalog Data Anda, Katalog Data menyediakan opsi untuk mendaftarkan peran IAM untuk mengenkripsi dan mendekripsi sumber daya. Anda harus memberikan izin peran IAM yang AWS Glue dapat diambil atas nama Anda. Ini termasuk AWS KMS izin untuk mengenkripsi dan mendekripsi data.

   Saat Anda membuat sumber daya baru di Katalog Data, AWS Glue asumsikan peran IAM yang disediakan untuk mengenkripsi data. Demikian pula, ketika konsumen mengakses sumber daya, AWS Glue mengasumsikan peran IAM untuk mendekripsi data. Jika Anda mendaftarkan peran IAM dengan izin yang diperlukan, prinsipal panggilan tidak lagi memerlukan izin untuk mengakses kunci dan mendekripsi data.

   > **Important**
   >
   > Anda dapat mendelegasikan operasi KMS ke peran IAM hanya jika Anda menggunakan kunci yang dikelola pelanggan untuk mengenkripsi sumber daya Katalog Data. Fitur delegasi peran KMS tidak mendukung penggunaan kunci AWS terkelola untuk mengenkripsi sumber daya Katalog Data saat ini.

   > **Warning**
   >
   > Bila Anda mengaktifkan peran IAM untuk mendelegasikan operasi KMS, Anda tidak dapat lagi mengakses sumber daya Katalog Data yang sebelumnya dienkripsi dengan kunci terkelola. AWS

a. Untuk mengaktifkan peran IAM yang AWS Glue dapat diasumsikan untuk mengenkripsi dan mendekripsi data atas nama Anda, pilih opsi Delegasikan operasi KMS ke peran IAM.

b. Selanjutnya, pilih peran IAM.

Untuk membuat peran IAM, lihat Membuat peran IAM untuk. AWS Glue

Peran IAM yang AWS Glue mengasumsikan untuk mengakses Katalog Data harus memiliki izin untuk mengenkripsi dan mendekripsi metadata dalam Katalog Data. Anda dapat membuat peran IAM, dan melampirkan kebijakan inline berikut:

- Tambahkan kebijakan berikut untuk menyertakan AWS KMS izin untuk mengenkripsi dan mendekripsi Katalog Data.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:Encrypt",
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:<region>:<account-id>:key/<key-id>"
        }
    ]
}
```

- Selanjutnya, tambahkan kebijakan kepercayaan berikut ke peran AWS Glue layanan untuk mengambil peran IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "glue.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

- Selanjutnya, tambahkan `iam:PassRole` izin ke peran IAM.

```
                    {
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::<account-id>:role/<encryption-role-name>"
            ]
        }
    ]
}
```

Saat Anda mengaktifkan enkripsi, jika Anda belum menentukan peran IAM AWS Glue untuk diasumsikan, prinsipal yang mengakses Katalog Data harus memiliki izin untuk melakukan operasi API berikut:

- `kms:Decrypt`
- `kms:Encrypt`
- `kms:GenerateDataKey`

AWS CLI

Untuk mengaktifkan enkripsi menggunakan SDK atau AWS CLI

- Gunakan Operasi API `PutDataCatalogEncryptionSettings`. Jika tidak ada kunci yang ditentukan, AWS Glue gunakan kunci enkripsi AWS terkelola untuk akun pelanggan untuk mengenkripsi Katalog Data.

```
aws glue put-data-catalog-encryption-settings \
  --data-catalog-encryption-settings '{
      "EncryptionAtRest": {
      "CatalogEncryptionMode": "SSE-KMS-WITH-SERVICE-ROLE",
      "SseAwsKmsKeyId": "arn:aws:kms:<region>:<account-id>:key/<key-id>",
      "CatalogEncryptionServiceRole":"arn:aws:iam::<account-
id>:role/<encryption-role-name>"
    }

  }'
```

Saat Anda mengaktifkan enkripsi, semua objek yang Anda buat di objek Katalog Data dienkripsi. Jika Anda menghapus pengaturan ini, objek yang Anda buat di Katalog Data tidak lagi dienkripsi. Anda dapat terus mengakses objek terenkripsi yang ada di Katalog Data dengan izin KMS yang diperlukan.

> **Important**
>
> Kunci AWS KMS harus tetap tersedia di penyimpanan kunci AWS KMS untuk setiap objek yang dienkripsi dengannya dalam Katalog Data. Jika Anda menghapus kunci tersebut, maka objek tidak dapat lagi didekripsi. Anda mungkin ingin melakukan hal ini dalam beberapa skenario untuk mencegah akses ke metadata Katalog Data.

Memantau kunci KMS Anda untuk AWS Glue

Saat Anda menggunakan kunci KMS dengan sumber daya Katalog Data Anda, Anda dapat menggunakan AWS CloudTrail atau Amazon CloudWatch Log untuk melacak permintaan yang AWS Glue dikirim. AWS KMS AWS CloudTrailmemantau dan merekam operasi KMS yang AWS Glue memanggil untuk mengakses data yang dienkripsi oleh kunci KMS Anda.

Contoh berikut adalah AWS CloudTrail peristiwa untuk `Decrypt` dan `GenerateDataKey` operasi.

Decrypt


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAXPHTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAXPHTESTANDEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-01-10T14:33:56Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "glue.amazonaws.com"
    },
    "eventTime": "2024-01-10T15:18:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "eu-west-2",
    "sourceIPAddress": "glue.amazonaws.com",
    "userAgent": "glue.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "glue_catalog_id": "111122223333"
        },
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "43b019aa-34b8-4798-9b98-ee968b2d63df",
    "eventID": "d7614763-d3fe-4f84-a1e1-3ca4d2a5bbd5",
    "readOnly": true,
```

```
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:<region>:111122223333:key/<key-id>"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "sessionCredentialFromConsole": "true"
}
```

GenerateDataKey

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId":
 "AROAXPHTESTANDEXAMPLE:V_00_GLUE_KMS_GENERATE_DATA_KEY_111122223333",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/
V_00_GLUE_KMS_GENERATE_DATA_KEY_111122223333",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAXPHTESTANDEXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "AKIAIOSFODNN7EXAMPLE",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2024-01-05T21:15:47Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "glue.amazonaws.com"
    },
```

```
    "eventTime": "2024-01-05T21:15:47Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "eu-west-2",
    "sourceIPAddress": "glue.amazonaws.com",
    "userAgent": "glue.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:eu-west-2:AKIAIOSFODNN7EXAMPLE:key/
AKIAIOSFODNN7EXAMPLE",
        "encryptionContext": {
            "glue_catalog_id": "111122223333"
        },
        "keySpec": "AES_256"
    },
    "responseElements": null,
    "requestID": "64d1783a-4b62-44ba-b0ab-388b50188070",
    "eventID": "1c73689b-2ef2-443b-aed7-8c126585ca5e",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:eu-west-2:111122223333:key/AKIAIOSFODNN7EXAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Mengenkripsi kata sandi koneksi

Anda dapat mengambil kata sandi koneksi di AWS Glue Data Catalog dengan menggunakan operasi API `GetConnection` dan `GetConnections`. Kata sandi ini disimpan dalam koneksi Katalog Data

dan digunakan saat AWS Glue menghubungkan ke penyimpanan data Java Database Connectivity (JDBC). Ketika koneksi dibuat atau diperbarui, opsi dalam pengaturan Katalog Data menentukan apakah kata sandi dienkripsi, dan jika demikian, kunci AWS Key Management Service (AWS KMS) apa yang ditentukan.

Pada konsol AWS Glue, Anda dapat mengaktifkan opsi ini di halaman Pengaturan katalog data.

Untuk mengenkripsi kata sandi koneksi

1. Masuk ke AWS Management Console, lalu buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Pilih Pengaturan di panel navigasi.
3. Pada halaman Pengaturan katalog data, pilih Enkripsi kata sandi koneksi, dan pilih kunci AWS KMS.

> **Important**
>
> AWS Glue hanya mendukung kunci master utama pelanggan simetris (CMC). Kunci AWS KMS hanya menampilkan kunci simetris saja. Namun, jika Anda memilih Pilih ARN kunci AWS KMS, konsol memungkinkan Anda memasukkan ARN untuk setiap jenis kunci. Pastikan bahwa Anda hanya memasukkan ARN untuk kunci simetris.

Untuk informasi selengkapnya, lihat Pengaturan Katalog Data .

Mengenkripsi data yang ditulis oleh AWS Glue

Sebuah konfigurasi keamanan adalah seperangkat properti keamanan yang dapat digunakan olehAWS Glue. Anda dapat menggunakan sebuah konfigurasi keamanan untuk mengenkripsi data at rest. Skenario berikut menunjukkan beberapa cara yang dapat Anda pakai untuk menggunakan konfigurasi keamanan.

- Lampirkan konfigurasi keamanan ke AWS Glue crawler untuk menulis Log Amazon CloudWatch terenkripsi. Untuk informasi selengkapnya tentang melampirkan konfigurasi keamanan ke crawler, lihat. the section called “Langkah 3: Konfigurasikan pengaturan keamanan”
- Lampirkan konfigurasi keamanan ke tugas ekstrak, transformasi, dan muat (ETL) untuk menulis target Amazon Simple Storage Service (Amazon S3) terenkripsi dan Log terenkripsi. CloudWatch

- Melampirkan sebuah konfigurasi keamanan untuk tugas ETL untuk menulis bookmark tugas sebagai data Amazon S3 yang dienkripsi.
- Melampirkan sebuah konfigurasi keamanan untuk titik akhir pengembangan untuk menulis target Amazon S3 yang terenkripsi.

> **Important**
>
> Saat ini, sebuah konfigurasi keamanan menimpa pengaturan enkripsi sisi server (SSE-S3) yang diberikan sebagai parameter tugas ETL. Jadi, jika sebuah konfigurasi keamanan dan parameter SSE-S3 dikaitkan dengan sebuah tugas, maka parameter SSE-S3 akan diabaikan.

Untuk informasi selengkapnya tentang konfigurasi keamanan, lihat Bekerja dengan konfigurasi keamanan di konsol AWS Glue.

Topik

- Menyiapkan AWS Glue untuk menggunakan konfigurasi keamanan
- Membuat rute AWS KMS untuk pekerjaan dan crawler VPC
- Bekerja dengan konfigurasi keamanan di konsol AWS Glue

Menyiapkan AWS Glue untuk menggunakan konfigurasi keamanan

Ikuti langkah-langkah berikut untuk menyiapkan lingkungan AWS Glue Anda untuk menggunakan konfigurasi keamanan.

1. Buat atau perbarui kunci AWS Key Management Service (AWS KMS) Anda untuk memberikan AWS KMS izin ke peran IAM yang diteruskan ke AWS Glue crawler dan pekerjaan untuk mengenkripsi Log. CloudWatch Untuk informasi selengkapnya, lihat Mengenkripsi Data Log di CloudWatch Log Menggunakan AWS KMS di Panduan Pengguna Amazon CloudWatch Logs.

   Pada contoh berikut, *"peran1"*, *"peran2"*, dan *"peran3"* adalah IAM role yang diberikan ke crawler dan tugas.

   ```
   {
          "Effect": "Allow",
          "Principal": { "Service": "logs.region.amazonaws.com",
          "AWS": [
   ```

```
                "role1",
                "role2",
                "role3"
            ] },
                "Action": [
                        "kms:Encrypt*",
                        "kms:Decrypt*",
                        "kms:ReEncrypt*",
                        "kms:GenerateDataKey*",
                        "kms:Describe*"
                ],
                "Resource": "*"
}
```

`Service`Pernyataan, ditampilkan sebagai`"Service": "logs.region.amazonaws.com"`, diperlukan jika Anda menggunakan kunci untuk mengenkripsi CloudWatch Log.

2. Pastikan bahwa kunci AWS KMS adalah `ENABLED` sebelum digunakan.

> **Note**
>
> Jika Anda menggunakan Iceberg sebagai kerangka data lake Anda, tabel Iceberg memiliki mekanisme sendiri untuk mengaktifkan enkripsi sisi server. Anda harus mengaktifkan konfigurasi ini selain AWS Glue konfigurasi keamanan. Untuk mengaktifkan enkripsi sisi server pada tabel Iceberg, tinjau panduan dari dokumentasi Iceberg.

Membuat rute AWS KMS untuk pekerjaan dan crawler VPC

Anda dapat connect langsung ke AWS KMS melalui titik akhir privat di virtual private cloud (VPC) Anda alih-alih terhubung melalui internet. Bila Anda menggunakan VPC endpoint, komunikasi antara VPC dan AWS KMS dilakukan sepenuhnya dalam jaringan AWS.

Anda dapat membuat VPC endpoint AWS KMS dalam sebuah VPC. Tanpa langkah ini, tugas atau crawler Anda mungkin gagal dengan `kms timeout` pada tugas atau `internal service exception` pada crawler. Untuk instruksi terperinci, lihat Menghubungkan ke AWS KMS Melalui VPC Endpoint dalam Panduan Developer AWS Key Management Service.

Saat Anda mengikuti petunjuk ini, pada Konsol VPC, Anda harus melakukan hal berikut:

- Pilih Mengaktifkan Nama DNS Privat.

- Pilih Grup keamanan (dengan aturan self-referencing) yang Anda gunakan untuk tugas Anda atau crawler yang mengakses Java Database Connectivity (JDBC). Untuk informasi selengkapnya tentang koneksi AWS Glue, lihat Menghubungkan ke data.

Bila Anda menambahkan sebuah konfigurasi keamanan ke crawler atau tugas yang mengakses penyimpanan data JDBC, maka AWS Glue harus memiliki rute ke titik akhir AWS KMS. Anda dapat menyediakan rute dengan gateway penerjemahan alamat jaringan (NAT) atau dengan VPC endpoint AWS KMS. Untuk membuat gateway NAT, lihat Gateway NAT di Panduan Pengguna Amazon VPC.

Bekerja dengan konfigurasi keamanan di konsol AWS Glue

> ⚠ Warning
>
> AWS Gluekonfigurasi keamanan saat ini tidak didukung dalam pekerjaan Ray.

Sebuha Konfigurasi keamanan di AWS Glue berisi properti yang diperlukan saat Anda menulis data terenkripsi. Anda membuat konfigurasi keamanan pada konsol AWS Glue untuk menyediakan properti enkripsi yang digunakan oleh crawler, tugas, dan pengembangan titik akhir.

Untuk melihat daftar semua konfigurasi keamanan yang telah Anda buat, buka konsol AWS Glue di https://console.aws.amazon.com/glue/ dan pilih Konfigurasi keamanan di panel navigasi.

Daftar Konfigurasi keamanan menampilkan properti berikut tentang masing-masing konfigurasi:

Nama

Nama unik yang Anda berikan saat Anda membuat konfigurasi tersebut. Nama dapat berisi huruf (A-Z), angka (0-9), hypens (-), atau garis bawah (_), dan panjangnya hingga 255 karakter.

Aktifkan enkripsi Amazon S3

Jika diaktifkan, mode enkripsi Amazon Simple Storage Service (Amazon S3) Simple Storage Service (Amazon S3) `SSE-KMS` seperti `SSE-S3` atau diaktifkan untuk penyimpanan metadata dalam katalog data.

Aktifkan enkripsi CloudWatch log Amazon

Jika dihidupkan, mode enkripsi Amazon S3 seperti `SSE-KMS` diaktifkan saat menulis log ke Amazon. CloudWatch

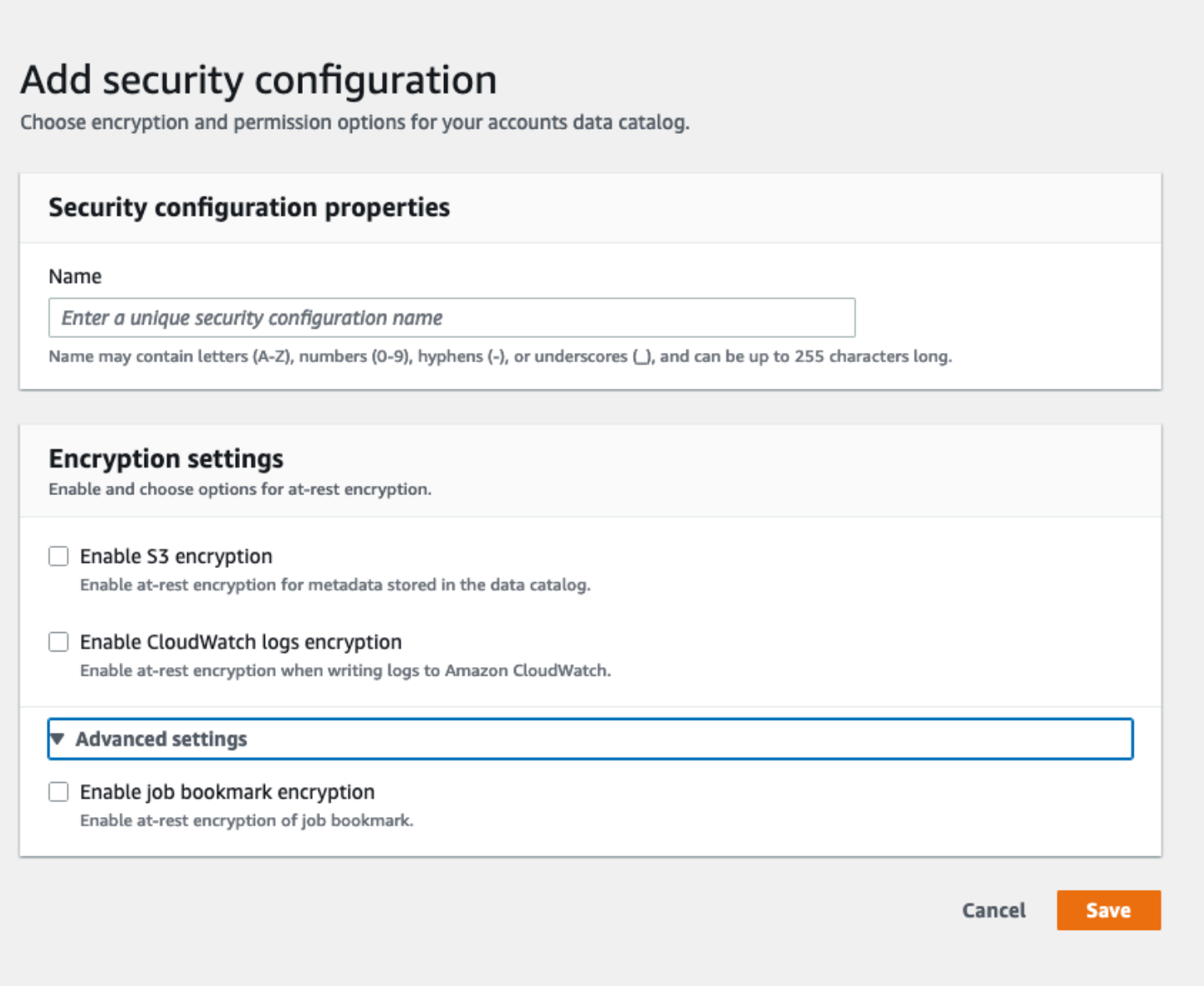
Pengaturan lanjutan: Aktifkan enkripsi bookmark pekerjaan

> Jika diaktifkan, mode enkripsi Amazon S3 seperti `CSE-KMS` diaktifkan saat pekerjaan di-bookmark.

Anda dapat menambahkan atau menghapus konfigurasi di bagian Konfigurasi keamanan pada konsol tersebut. Untuk melihat detail lebih lanjut untuk sebuah konfigurasi, pilih nama konfigurasi dalam daftar tersebut. Detail itu mencakup informasi yang Anda tetapkan saat Anda membuat konfigurasi.

Menambahkan konfigurasi keamanan

Untuk menambahkan sebuah konfigurasi keamanan dengan menggunakan konsol AWS Glue, pada halaman Konfigurasi keamanan, pilih Tambahkan konfigurasi keamanan.



Properti konfigurasi keamanan

Masukkan nama konfigurasi keamanan yang unik. Nama dapat berisi huruf (A-Z), angka (0-9), tanda hubung (-), atau garis bawah (_), dan dapat mencapai 255 karakter.

Pengaturan enkripsi

Anda dapat mengaktifkan enkripsi saat istirahat untuk metadata yang disimpan di Katalog Data di Amazon S3 dan log di Amazon. CloudWatch Untuk mengatur enkripsi data dan metadata dengan kunci AWS Key Management Service (AWS KMS) pada konsol AWS Glue, tambahkan sebuah kebijakan untuk pengguna konsol. Kebijakan ini harus menentukan sumber daya yang diizinkan

sebagai kunci Amazon Resource Name (ARN) yang digunakan untuk mengenkripsi penyimpanan data Amazon S3, seperti dalam contoh berikut.

```
{
"Version": "2012-10-17",
  "Statement": {
  "Effect": "Allow",
  "Action": [
    "kms:GenerateDataKey",
    "kms:Decrypt",
    "kms:Encrypt"],
  "Resource": "arn:aws:kms:region:account-id:key/key-id"}
}
```

> **Important**
>
> Ketika sebuah konfigurasi keamanan dilampirkan ke sebuah crawler atau tugas, IAM role yang diberikan harus memiliki izin AWS KMS. Untuk informasi selengkapnya, lihat Mengenkripsi data yang ditulis oleh AWS Glue.

Ketika Anda menentukan sebuah konfigurasi, Anda dapat memberikan nilai untuk properti berikut:

Aktifkan enkripsi S3

Ketika Anda menulis data Amazon S3, Anda menggunakan enkripsi sisi server dengan kunci terkelola Amazon S3 (SSE-S3) atau enkripsi sisi server dengan kunci terkelola AWS KMS (SSE-KMS). Bidang ini bersifat opsional. Untuk mengizinkan akses ke Amazon S3, pilih AWS KMS, atau pilih Masukkan ARN kunci dan berikan ARN untuk kunci tersebut. Masukkan ARN dalam bentuk `arn:aws:kms:region:account-id:key/key-id`. Anda juga dapat memberikan ARN sebagai alias kunci, seperti `arn:aws:kms:region:account-id:alias/alias-name`.

Jika Anda mengaktifkan Spark UI untuk pekerjaan Anda, file log UI Spark yang diunggah ke Amazon S3 akan diterapkan dengan enkripsi yang sama.

> **Important**
>
> AWS Glue hanya mendukung kunci master utama pelanggan simetris (CMC). Kunci AWS KMS hanya menampilkan kunci simetris saja. Namun, jika Anda memilih Pilih ARN kunci

AWS KMS, konsol memungkinkan Anda memasukkan ARN untuk setiap jenis kunci. Pastikan bahwa Anda hanya memasukkan ARN untuk kunci simetris.

Aktifkan enkripsi CloudWatch Log

Enkripsi sisi server (SSE-KMS) digunakan untuk mengenkripsi Log. CloudWatch Bidang ini bersifat opsional. Untuk mengaktifkannya, pilih kunci AWS KMS, atau pilih Masukkan ARN kunci dan berikan ARN untuk kunci tersebut. Masukkan ARN dalam bentuk `arn:aws:kms:`*`region`*`:`*`account-id`*`:key/`*`key-id`*. Anda juga dapat memberikan ARN sebagai alias kunci, seperti `arn:aws:kms:`*`region`*`:`*`account-id`*`:alias/`*`alias-name`*.

Pengaturan lanjutan: Enkripsi bookmark Job

Enkripsi sisi klien (CSE-KMS) digunakan untuk mengenkripsi bookmark tugas. Bidang ini bersifat opsional. Data bookmark dienkripsi sebelum dikirim ke Amazon S3 untuk penyimpanan. Untuk mengaktifkannya, pilih kunci AWS KMS, atau pilih Masukkan ARN kunci dan berikan ARN untuk kunci tersebut. Masukkan ARN dalam bentuk `arn:aws:kms:`*`region`*`:`*`account-id`*`:key/`*`key-id`*. Anda juga dapat memberikan ARN sebagai alias kunci, seperti `arn:aws:kms:`*`region`*`:`*`account-id`*`:alias/`*`alias-name`*.

Untuk informasi selengkapnya, lihat topik berikut di Panduan Pengguna Amazon Simple Storage Service:

- Untuk informasi selengkapnya tentang `SSE-S3`, lihat Melindungi Data Menggunakan Enkripsi Sisi Server dengan Kunci Enkripsi Terkelola Amazon S3 (SSE-S3).
- Untuk selengkapnya`SSE-KMS`, lihat Melindungi Data Menggunakan Enkripsi Sisi Server dengan. AWS KMS keys
- Untuk selengkapnya`CSE-KMS`, lihat Menggunakan kunci KMS yang disimpan di AWS KMS.

Enkripsi dalam transit

AWSmenyediakan enkripsi Transport Layer Security (TLS) untuk data yang bergerak. Anda dapat mengkonfigurasi pengaturan enkripsi untuk crawler, tugas ETL, dan titik akhir pengembangan menggunakan Konfigurasi keamanan di AWS Glue. Anda dapat mengaktifkan enkripsi AWS Glue Data Catalog melalui pengaturan untuk Katalog Data.

Sejak tanggal 4 September 2018, AWS KMS (bawa kunci Anda sendiri dan enkripsi sisi server) untuk ETL AWS Glue dan AWS Glue Data Catalog didukung.

Kepatuhan FIPS

Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-2 ketika mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Untuk informasi lebih lanjut tentang titik akhir FIPS yang tersedia, lihat Standar Pemrosesan Informasi Federal (FIPS) 140-2.

Manajemen kunci

Anda dapat menggunakan AWS Identity and Access Management (IAM) dengan AWS Glue untuk mendefinisikan pengguna, sumber daya AWS, grup, peran, dan kebijakan terperinci terkait dengan akses, penolakan, dan lainnya.

Anda dapat menentukan akses ke metadata dengan menggunakan kebijakan berbasis sumber daya dan berbasis identitas, sesuai dengan kebutuhan organisasi Anda. Kebijakan berbasis sumber daya mencantumkan prinsipal utama yang diizinkan atau ditolak aksesnya ke sumber daya Anda, sehingga Anda dapat menyiapkan kebijakan seperti kebijakan akses lintas akun. Kebijakan identitas secara khusus dilampirkan pada pengguna, grup, dan peran dalam IAM.

step-by-step Misalnya, lihat Membatasi akses ke Anda AWS Glue Data Catalog dengan izin IAM tingkat sumber daya dan kebijakan berbasis sumber daya di Blog Big Data. AWS

Bagian akses terperinci dari kebijakan tersebut didefinisikan dalam klausul `Resource`. Bagian ini mendefinisikan pada objek AWS Glue Data Catalog mana tindakan dapat dilakukan, dan apa yang mengakibatkan objek dikembalikan oleh operasi itu.

Sebuah titik akhir pengembangan adalah sebuah lingkungan yang dapat Anda gunakan untuk mengembangkan dan menguji skrip AWS Glue. Anda dapat menambahkan, menghapus, atau memutar kunci SSH dari sebuah titik akhir pengembangan.

Sejak tanggal 4 September 2018, AWS KMS (bawa kunci Anda sendiri dan enkripsi sisi server) untuk ETL AWS Glue dan AWS Glue Data Catalog didukung.

AWS Glueketergantungan pada layanan lain AWS

Agar pengguna dapat bekerja dengan konsol AWS Glue, pengguna tersebut harus memiliki seperangkat izin minimum yang memungkinkannya untuk menggunakan sumber daya AWS Glue untuk akun AWS mereka. Selain izin AWS Glue ini, konsol memerlukan izin dari layanan berikut:

- Izin CloudWatch Log Amazon untuk menampilkan log.
- Izin AWS Identity and Access Management (IAM) untuk mencatumkan dan memberikan peran.
- Izin AWS CloudFormation untuk menggunakan tumpukan.
- Izin Amazon Elastic Compute Cloud (Amazon EC2) untuk mencantumkan virtual private cloud (VPC), subnet, grup keamanan, instans, dan objek lainnya (untuk menyiapkan item Amazon EC2 seperti VPC saat menjalankan tugas, crawler, dan menciptakan titik akhir pengembangan).
- Izin Amazon Simple Storage Service (Amazon S3) untuk mencantumkan bucket dan objek, dan untuk mengambil dan menyimpan skrip.
- Izin Amazon Redshift untuk bekerja dengan klaster.
- Izin Amazon Relational Database Service (Amazon RDS) untuk mencantumkan instans.

Titik akhir pengembangan

Sebuah titik akhir pengembangan adalah sebuah lingkungan yang dapat Anda gunakan untuk mengembangkan dan menguji skrip AWS Glue. Anda dapat menggunakan AWS Glue untuk membuat, mengedit, dan menghapus titik akhir pengembangan. Anda dapat membuat daftar semua titik akhir pengembangan yang dibuat. Anda dapat menambahkan, menghapus, atau memutar kunci SSH dari sebuah titik akhir pengembangan. Anda juga dapat membuat notebook yang menggunakan titik akhir pengembangan tersebut.

Anda memberikan nilai konfigurasi untuk penyediaan lingkungan pengembangan. Nilai-nilai ini memberitahu cara AWS Glue mengatur jaringan sehingga Anda dapat mengakses titik akhir pengembangan dengan aman, dan sehingga titik akhir Anda dapat mengakses penyimpanan data Anda. Kemudian, Anda dapat membuat sebuah notebook yang terhubung ke titik akhir pengembangan. Anda menggunakan notebook Anda untuk menulis dan menguji skrip ETL Anda.

Menggunakan AWS Identity and Access Management (IAM) role dengan izin yang serupa dengan IAM role yang Anda gunakan untuk menjalankan tugas ETL AWS Glue. Gunakan virtual private cloud (VPC), subnet, dan grup keamanan untuk membuat titik akhir pengembangan yang dapat connect ke sumber daya data Anda dengan aman. Anda menghasilkan pasangan kunci SSH untuk connect ke lingkungan pengembangan dengan menggunakan SSH.

Anda dapat membuat titik akhir pengembangan untuk data Amazon S3 dan dalam sebuah VPC yang dapat Anda gunakan untuk mengakses set data menggunakan JDBC.

Anda dapat menginstal klien notebook Jupyter di mesin lokal Anda dan menggunakannya untuk men-debug dan menguji skrip ETL pada titik akhir pengembangan. Atau, Anda dapat menggunakan buku

catatan Sagemaker untuk membuat skrip ETL di aktif. JupyterLab AWS Lihat Menggunakan buku SageMaker catatan dengan titik akhir pengembangan Anda.

Tag AWS Glue instans Amazon EC2 dengan nama yang menggunakan prefiks `aws-glue-dev-endpoint`.

Anda dapat mengatur server notebook pada titik akhir pengembangan untuk dijalankan PySpark dengan AWS Glue ekstensi.

Manajemen identitas dan akses untuk AWS Glue

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan sumber daya Glue. AWS IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

> **Note**
>
> Anda dapat memberikan akses ke data Anda di Katalog Data AWS Glue menggunakan AWS Glue metode atau AWS Lake Formation hibah. Anda dapat menggunakan kebijakan AWS Identity and Access Management (IAM) untuk menyetel kontrol akses berbutir halus dengan metode. AWS Glue Lake Formation menggunakan model `GRANT/REVOKE` izin yang lebih sederhana yang mirip dengan `GRANT/REVOKE` perintah dalam sistem database relasional. Bagian ini mencakup informasi tentang cara menggunakan AWS Glue metode. Untuk informasi tentang penggunaan hibah Lake Formation, lihat Memberikan izin Lake Formation di Panduan Pengembang.AWS Lake Formation

Topik

- Audiens
- Mengautentikasi dengan identitas
- Mengelola akses menggunakan kebijakan
- Bagaimana AWS Glue bekerja dengan IAM
- Mengkonfigurasi izin IAM untuk AWS Glue
- AWSContoh kebijakan kontrol akses Glue
- AWS kebijakan terkelola untuk AWS Glue

- Menentukan ARN AWS Glue sumber daya
- Memberikan akses lintas akun
- Pemecahan masalah identitas dan akses AWS Glue

Audiens

Cara Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan di AWS Glue.

Pengguna layanan - Jika Anda menggunakan layanan AWS Glue untuk melakukan pekerjaan Anda, administrator Anda memberi Anda kredensyal dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak fitur AWS Glue untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur di AWS Glue, lihatPemecahan masalah identitas dan akses AWS Glue.

Administrator layanan - Jika Anda bertanggung jawab atas sumber daya AWS Glue di perusahaan Anda, Anda mungkin memiliki akses penuh ke AWS Glue. Tugas Anda adalah menentukan fitur dan sumber daya AWS Glue mana yang harus diakses pengguna layanan Anda. Kemudian, Anda harus mengirimkan permintaan kepada administrator IAM untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep Basic IAM. Untuk mempelajari lebih lanjut tentang bagaimana perusahaan Anda dapat menggunakan IAM dengan AWS Glue, lihatBagaimana AWS Glue bekerja dengan IAM.

Administrator IAM - Jika Anda seorang administrator IAM, Anda mungkin ingin mempelajari detail tentang cara menulis kebijakan untuk mengelola akses ke Glue AWS . Untuk melihat contoh kebijakan berbasis identitas AWS Glue yang dapat Anda gunakan di IAM, lihat. Contoh kebijakan berbasis identitas untuk Glue AWS

Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensyal identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai pengguna IAM, atau dengan mengasumsikan peran IAM.

Anda dapat masuk AWS sebagai identitas federasi dengan menggunakan kredensyal yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (IAM Identity Center), autentikasi masuk tunggal perusahaan Anda, dan kredensyal Google atau Facebook Anda adalah

contoh identitas federasi. Saat Anda masuk sebagai identitas terfederasi, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal AWS akses. Untuk informasi selengkapnya tentang masuk AWS, lihat Cara masuk ke Panduan AWS Sign-In Pengguna Anda Akun AWS.

Jika Anda mengakses AWS secara terprogram, AWS sediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensil Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Untuk informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, lihat Menandatangani permintaan AWS API di Panduan Pengguna IAM.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat Autentikasi multi-faktor dalam Panduan Pengguna AWS IAM Identity Center dan Menggunakan autentikasi multi-faktor (MFA) dalam AWS dalam Panduan Pengguna IAM.

Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihat Tugas yang memerlukan kredensial pengguna root dalam Panduan Pengguna IAM.

Identitas gabungan

Sebagai praktik terbaik, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensyal sementara.

Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, direktori Pusat Identitas AWS Directory Service, atau pengguna mana pun yang mengakses

Layanan AWS dengan menggunakan kredensil yang disediakan melalui sumber identitas. Ketika identitas federasi mengakses Akun AWS, mereka mengambil peran, dan peran memberikan kredensyal sementara.

Untuk manajemen akses terpusat, kami sarankan Anda menggunakan AWS IAM Identity Center. Anda dapat membuat pengguna dan grup di Pusat Identitas IAM, atau Anda dapat menghubungkan dan menyinkronkan ke sekumpulan pengguna dan grup di sumber identitas Anda sendiri untuk digunakan di semua aplikasi Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat Identitas IAM, lihat Apakah itu Pusat Identitas IAM? dalam Panduan Pengguna AWS IAM Identity Center .

Pengguna dan grup IAM

Pengguna IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, kami merekomendasikan untuk mengandalkan kredensial sementara, bukan membuat pengguna IAM yang memiliki kredensial jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan tertentu yang memerlukan kredensial jangka panjang dengan pengguna IAM, kami merekomendasikan Anda merotasi kunci akses. Untuk informasi selengkapnya, lihat Merotasi kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensial jangka panjang dalam Panduan Pengguna IAM.

Grup IAM adalah identitas yang menentukan sekumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin bagi beberapa pengguna sekaligus. Grup mempermudah manajemen izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat memiliki grup yang bernama IAMAdmins dan memberikan izin ke grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari selengkapnya, lihat Kapan harus membuat pengguna IAM (bukan peran) dalam Panduan Pengguna IAM.

Peran IAM

Peran IAM adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Peran ini mirip dengan pengguna IAM, tetapi tidak terkait dengan orang tertentu. Anda dapat mengambil peran IAM untuk sementara AWS Management Console dengan beralih peran. Anda dapat mengambil peran dengan memanggil operasi AWS CLI atau AWS API atau dengan menggunakan URL kustom. Untuk

informasi selengkapnya tentang cara menggunakan peran, lihat Menggunakan peran IAM dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Ketika identitas terfederasi mengautentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat Membuat peran untuk Penyedia Identitas pihak ketiga dalam Panduan Pengguna IAM. Jika menggunakan Pusat Identitas IAM, Anda harus mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM akan mengorelasikan set izin ke peran dalam IAM. Untuk informasi tentang set izin, lihat Set izin dalam Panduan Pengguna AWS IAM Identity Center .
- Izin pengguna IAM sementara – Pengguna atau peran IAM dapat mengambil peran IAM guna mendapatkan berbagai izin secara sementara untuk tugas tertentu.
- Akses lintas akun – Anda dapat menggunakan peran IAM untuk mengizinkan seseorang (prinsipal tepercaya) di akun lain untuk mengakses sumber daya di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan secara langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM.
- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Sebagai contoh, ketika Anda memanggil suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.
  - Sesi akses teruskan (FAS) — Saat Anda menggunakan pengguna IAM atau peran untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.

- Peran layanan – Peran layanan adalah peran IAM yang dijalankan oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Membuat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.
- Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke peran layanan. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan peran IAM untuk mengelola kredensi sementara untuk aplikasi yang berjalan pada instans EC2 dan membuat atau permintaan API. AWS CLI AWS Cara ini lebih dianjurkan daripada menyimpan kunci akses dalam instans EC2. Untuk menetapkan AWS peran ke instans EC2 dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instance yang dilampirkan ke instance. Profil instans berisi peran dan memungkinkan program yang berjalan di instans EC2 mendapatkan kredensial sementara. Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan dalam instans Amazon EC2 dalam Panduan Pengguna IAM.

Untuk mempelajari apakah kita harus menggunakan peran IAM atau pengguna IAM, lihat Kapan harus membuat peran IAM (bukan pengguna) dalam Panduan Pengguna IAM.

Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal (pengguna, pengguna root, atau sesi peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan isi dokumen kebijakan JSON, lihat Gambaran umum kebijakan JSON dalam Panduan Pengguna IAM.

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat

kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasinya. Misalnya, anggaplah Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut bisa mendapatkan informasi peran dari AWS Management Console, API AWS CLI, atau AWS API.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan yang dikelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan terkelola adalah kebijakan mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran dalam. Akun AWS Kebijakan AWS terkelola mencakup kebijakan terkelola dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan yang dikelola atau kebijakan inline, lihat Memilih antara kebijakan yang dikelola dan kebijakan inline dalam Panduan Pengguna IAM.

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACL)

Daftar kontrol akses (ACL) mengendalikan prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, lihat Gambaran umum daftar kontrol akses (ACL) dalam Panduan Developer Amazon Simple Storage Service.

Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Jenis-jenis kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh jenis kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan tempat Anda mengatur izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas ke entitas IAM (pengguna IAM atau peran IAM). Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh batasan izin. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya tentang batasan izin, lihat Batasan izin untuk entitas IAM dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCP) — SCP adalah kebijakan JSON yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU) di. AWS Organizations AWS Organizations adalah layanan untuk mengelompokkan dan mengelola secara terpusat beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur di organisasi, Anda dapat menerapkan kebijakan kontrol layanan (SCP) ke salah satu atau semua akun Anda. SCP membatasi izin untuk entitas di akun anggota, termasuk masing-masing. Pengguna root akun AWS Untuk informasi selengkapnya tentang Organisasi dan SCP, lihat Cara kerja SCP dalam Panduan Pengguna AWS Organizations .
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara programatis untuk peran atau pengguna terfederasi. Izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya, lihat Kebijakan sesi dalam Panduan Pengguna IAM.

Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat Logika evaluasi kebijakan di Panduan Pengguna IAM.

Bagaimana AWS Glue bekerja dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke AWS Glue, pelajari fitur IAM apa yang tersedia untuk digunakan dengan Glue AWS .

Fitur IAM yang dapat Anda gunakan dengan AWS Glue

Fitur IAM	AWS Dukungan Glue
Kebijakan berbasis identitas	Ya
Kebijakan berbasis sumber daya	Parsial
Tindakan kebijakan	Ya
Sumber daya kebijakan	Ya
kunci-kunci persyaratan kebijakan (spesifik layanan)	Ya
ACL	Tidak
ABAC (tanda dalam kebijakan)	Parsial
Kredensial sementara	Ya
Izin prinsipal	Tidak
Peran layanan	Ya
Peran terkait layanan	Tidak

Untuk mendapatkan tampilan tingkat tinggi tentang cara kerja AWS Glue dan AWS layanan lainnya dengan sebagian besar fitur IAM, lihat AWS layanan yang bekerja dengan IAM di Panduan Pengguna IAM.

Kebijakan berbasis identitas untuk Glue AWS

Mendukung kebijakan berbasis identitas	Ya

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta kondisi yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Anda tidak dapat menentukan secara spesifik prinsipal dalam sebuah kebijakan berbasis identitas karena prinsipal berlaku bagi pengguna atau peran yang melekat kepadanya. Untuk mempelajari semua elemen yang dapat Anda gunakan dalam kebijakan JSON, lihat Referensi elemen kebijakan JSON IAM dalam Panduan Pengguna IAM.

AWS Gluemendukung kebijakan berbasis identitas (kebijakan IAM) untuk semua operasi. AWS Glue Dengan melampirkan kebijakan, Anda dapat memberikan izin untuk membuat, mengakses, atau memodifikasi AWS Glue sumber daya, seperti tabel di. AWS Glue Data Catalog

Contoh kebijakan berbasis identitas untuk Glue AWS

Untuk melihat contoh kebijakan berbasis identitas AWS Glue, lihat. Contoh kebijakan berbasis identitas untuk Glue AWS

Kebijakan berbasis sumber daya dalam Glue AWS

Mendukung kebijakan berbasis sumber daya	Parsial

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan

kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan secara spesifik seluruh akun atau entitas IAM di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya. Menambahkan prinsipal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsipal dan sumber daya berbeda Akun AWS, administrator IAM di akun tepercaya juga harus memberikan izin entitas utama (pengguna atau peran) untuk mengakses sumber daya. Mereka memberikan izin dengan melampirkan kebijakan berbasis identitas kepada entitas. Namun, jika kebijakan berbasis sumber daya memberikan akses ke prinsipal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk informasi selengkapnya, lihat Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM.

> **Note**
>
> Anda hanya dapat menggunakan kebijakan AWS Glue sumber daya untuk mengelola izin sumber daya Katalog Data. Anda tidak dapat melampirkannya ke AWS Glue sumber daya lain seperti pekerjaan, pemicu, titik akhir pengembangan, perayap, atau pengklasifikasi. Hanya satu kebijakan sumber daya yang diizinkan per katalog, dan ukurannya dibatasi hingga 10 KB.

Di AWS Glue, kebijakan sumber daya dilampirkan ke katalog, yang merupakan wadah virtual untuk semua jenis sumber daya Katalog Data yang disebutkan sebelumnya. Setiap AWS akun memiliki satu katalog di AWS Wilayah yang ID katalognya sama dengan ID AWS akun. Anda tidak dapat menghapus atau memodifikasi katalog.

Sebuah kebijakan sumber daya dievaluasi untuk semua panggilan API ke katalog di mana prinsipal utama pemanggil disertakan dalam blok `"Principal"` dokumen kebijakan.

Untuk melihat contoh kebijakan berbasis sumber daya AWS Glue, lihat. Contoh kebijakan berbasis sumber daya untuk Glue AWS

Tindakan kebijakan untuk AWS Glue

Mendukung tindakan kebijakan	Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan operasi AWS API terkait. Ada beberapa pengecualian, misalnya tindakan hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Menyertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Untuk melihat daftar tindakan AWS Glue, lihat Tindakan yang ditentukan oleh AWS Glue di Referensi Otorisasi Layanan.

Tindakan kebijakan di AWS Glue menggunakan awalan berikut sebelum tindakan:

```
glue
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan tersebut dengan koma.

```
"Action": [
      "glue:action1",
      "glue:action2"
         ]
```

Anda juga dapat menentukan beberapa tindakan menggunakan wildcard (*). Sebagai contoh, untuk menentukan semua tindakan yang dimulai dengan kata `Get`, sertakan tindakan berikut:

```
"Action": "glue:Get*"
```

Untuk melihat contoh kebijakan, lihat AWSContoh kebijakan kontrol akses Glue.

Sumber daya kebijakan untuk AWS Glue

Mendukung sumber daya kebijakan	Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek yang menjadi target penerapan tindakan. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`. Praktik terbaiknya, tentukan sumber daya menggunakan Amazon Resource Name (ARN). Anda dapat melakukan ini untuk tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": "*"
```

Untuk informasi selengkapnya tentang cara mengontrol akses ke sumber daya AWS Glue menggunakan ARN, lihatMenentukan ARN AWS Glue sumber daya.

Untuk melihat daftar jenis sumber daya AWS Glue dan ARNnya, lihat Sumber Daya yang ditentukan oleh AWS Glue dalam Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda gunakan untuk menentukan ARN dari setiap sumber daya, lihat Tindakan yang ditentukan oleh Glue AWS.

Kunci kondisi kebijakan untuk AWS Glue

Mendukung kunci kondisi kebijakan khusus layanan	Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Artinya, prinsipal manakah yang dapat melakukan tindakan pada sumber daya apa, dan dengan kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan operator kondisi, misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam sebuah pernyataan, atau beberapa kunci dalam elemen `Condition` tunggal, maka AWS akan mengevaluasinya menggunakan operasi `AND` logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan `OR` operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Sebagai contoh, Anda dapat memberikan izin kepada pengguna IAM untuk mengakses sumber daya hanya jika izin tersebut mempunyai tag yang sesuai dengan nama pengguna IAM mereka. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM: variabel dan tag dalam Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi AWS global, lihat kunci konteks kondisi AWS global di Panduan Pengguna IAM.

Untuk melihat daftar kunci kondisi AWS Glue, lihat tombol Kondisi untuk AWS Glue di Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat digunakan untuk menggunakan kunci kondisi, lihat Tindakan yang ditentukan oleh AWS Glue.

Untuk melihat contoh kebijakan, lihat Pengaturan kontrol menggunakan tombol kondisi atau tombol konteks.

ACL dalam AWS Glue

Mendukung ACL	Tidak

Daftar kontrol akses (ACL) mengendalikan pengguna utama mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL serupa dengan kebijakan berbasis sumber daya, meskipun kebijakan tersebut tidak menggunakan format dokumen kebijakan JSON.

ABAC dengan AWS Glue

Mendukung ABAC (tanda dalam kebijakan)	Parsial

Kontrol akses berbasis atribut (ABAC) adalah strategi otorisasi yang menentukan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke entitas IAM (pengguna atau peran) dan ke banyak AWS sumber daya. Penandaan ke entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian rancanglah kebijakan ABAC untuk mengizinkan operasi ketika tag milik prinsipal cocok dengan tag yang ada di sumber daya yang ingin diakses.

ABAC sangat berguna di lingkungan yang berkembang dengan cepat dan berguna di situasi saat manajemen kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tag, berikan informasi tentang tag di elemen kondisi dari kebijakan menggunakan kunci kondisi `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, atau `aws:TagKeys`.

Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya tentang ABAC, lihat Apa itu ABAC? dalam Panduan Pengguna IAM. Untuk melihat tutorial yang menguraikan langkah-langkah pengaturan ABAC, lihat Menggunakan kontrol akses berbasis atribut (ABAC) dalam Panduan Pengguna IAM.

> **Important**
>
> Kunci konteks kondisi hanya berlaku untuk tindakan AWS Glue API pada crawler, pekerjaan, pemicu, dan titik akhir pengembangan. Untuk informasi selengkapnya tentang operasi API mana yang terpengaruh, lihat Kunci kondisi untuk AWS Glue.
> Operasi API Katalog AWS Glue Data saat ini tidak mendukung kunci konteks kondisi `aws:UserAgent` global `aws:referer` dan global.

Untuk melihat contoh kebijakan berbasis identitas untuk membatasi akses ke sumber daya berdasarkan tag pada sumber daya tersebut, lihat Berikan akses menggunakan tag.

Menggunakan kredensyal sementara dengan Glue AWS

Mendukung penggunaan kredensial sementara	Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensyal sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensyal sementara, lihat Layanan AWS yang bekerja dengan IAM di Panduan Pengguna IAM.

Anda menggunakan kredensyal sementara jika Anda masuk AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS menggunakan tautan masuk tunggal (SSO) perusahaan Anda, proses tersebut secara otomatis membuat kredensyal sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna lalu beralih peran. Untuk informasi selengkapnya tentang peralihan peran, lihat Peralihan peran (konsol) dalam Panduan Pengguna IAM.

Anda dapat membuat kredensyal sementara secara manual menggunakan API AWS CLI atau AWS . Anda kemudian dapat menggunakan kredensyal sementara tersebut untuk mengakses. AWS AWS merekomendasikan agar Anda menghasilkan kredensyal sementara secara dinamis alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat Kredensial keamanan sementara di IAM.

Izin utama lintas layanan untuk Glue AWS

Mendukung sesi akses maju (FAS)	Tidak

Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.

Peran layanan untuk AWS Glue

Mendukung peran layanan	Ya

Peran layanan adalah peran IAM yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Membuat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

> **Warning**
>
> Mengubah izin untuk peran layanan dapat merusak fungsionalitas AWS Glue. Edit peran layanan hanya jika AWS Glue memberikan panduan untuk melakukannya.

Untuk petunjuk rinci tentang membuat peran layanan untuk AWS Glue, lihat Langkah 1: Buat kebijakan IAM untuk layanan AWS Glue danLangkah 2: Buat peran IAM untuk AWS Glue.

Peran terkait layanan untuk Glue AWS

Mendukung peran terkait layanan	Tidak

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Untuk detail tentang pembuatan atau manajemen peran terkait layanan, lihat Layanan AWS yang berfungsi dengan IAM. Cari layanan dalam tabel yang memiliki `Yes` di kolom Peran terkait layanan. Pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Mengkonfigurasi izin IAM untuk AWS Glue

Anda menggunakan AWS Identity and Access Management (IAM) untuk menentukan kebijakan dan peran yang AWS Glue digunakan untuk mengakses sumber daya. Langkah-langkah berikut mengarahkan Anda melalui berbagai opsi untuk mengatur izin untukAWS Glue. Bergantung pada kebutuhan bisnis Anda, Anda mungkin harus menambahkan atau mengurangi akses ke sumber daya Anda.

> **Note**
>
> Untuk memulai dengan izin IAM dasar, AWS Glue lihat. Menyiapkan izin IAM untuk AWS Glue

1. Buat kebijakan IAM untuk AWS Glue layanan: Buat kebijakan layanan yang memungkinkan akses ke AWS Glue sumber daya.
2. Buat peran IAM untukAWS Glue: Buat peran IAM, dan lampirkan kebijakan AWS Glue layanan dan kebijakan untuk sumber daya Amazon Simple Storage Service (Amazon S3) yang digunakan oleh. AWS Glue
3. Lampirkan kebijakan ke pengguna atau grup yang mengakses AWS Glue: Lampirkan kebijakan ke pengguna atau grup mana pun yang masuk ke AWS Glue konsol.
4. Membuat kebijakan IAM untuk buku catatan: Buat kebijakan server notebook untuk digunakan dalam pembuatan server notebook pada titik akhir pengembangan.
5. Buat peran IAM untuk buku catatan: Buat peran IAM dan lampirkan kebijakan server notebook.
6. Buat kebijakan IAM untuk SageMaker notebook Amazon: Buat kebijakan IAM untuk digunakan saat membuat SageMaker notebook Amazon pada titik akhir pengembangan.
7. Buat peran IAM untuk SageMaker notebook Amazon: Buat peran IAM dan lampirkan kebijakan untuk memberikan izin saat membuat buku SageMaker catatan Amazon di titik akhir pengembangan.

Langkah 1: Buat kebijakan IAM untuk layanan AWS Glue

Untuk setiap operasi yang mengakses data pada sumber daya AWS yang lain, seperti mengakses objek Anda di Amazon S3, AWS Glue memerlukan izin untuk mengakses sumber daya atas nama Anda. Anda memberikan izin tersebut dengan menggunakan AWS Identity and Access Management (IAM).

> **Note**
>
> Anda dapat melewati langkah ini jika menggunakan kebijakan AWS terkelola**`AWSGlueServiceRole`**.

Pada langkah ini, Anda membuat sebuah kebijakan yang mirip dengan `AWSGlueServiceRole`. Anda dapat menemukan versi terbaru `AWSGlueServiceRole` pada konsol IAM.

Untuk membuat sebuah kebijakan IAM untuk AWS Glue

Kebijakan ini memberikan izin untuk beberapa tindakan Amazon S3 untuk mengelola sumber daya di akun Anda yang diperlukan oleh AWS Glue ketika mengasumsikan peran dengan menggunakan kebijakan ini. Beberapa sumber daya yang ditentukan dalam kebijakan ini merujuk ke nama default yang digunakan oleh AWS Glue bucket Amazon S3, skrip ETL Amazon S3, Log, CloudWatch dan sumber daya Amazon EC2. Sederhananya, AWS Glue menulis beberapa objek Amazon S3 ke dalam bucket di akun Anda yang mempunyai prefiks dengan `aws-glue-*` secara default.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di sebelah kiri, pilih Kebijakan.
3. Pilih Buat Kebijakan.
4. Pada layar Buat kebijakan, navigasikan ke tab untuk mengedit JSON. Buat sebuah dokumen kebijakan dengan pernyataan JSON berikut, dan kemudian pilih Tinjau kebijakan.

> **Note**
>
> Menambahkan izin yang diperlukan untuk sumber daya Amazon S3. Anda mungkin ingin bagian sumber daya atas kebijakan akses Anda hanya mencakup sumber daya yang diperlukan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:*",
                "s3:GetBucketLocation",
                "s3:ListBucket",
                "s3:ListAllMyBuckets",
                "s3:GetBucketAcl",
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeRouteTables",
```

```
                "ec2:CreateNetworkInterface",
                "ec2:DeleteNetworkInterface",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcAttribute",
                "iam:ListRolePolicies",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "cloudwatch:PutMetricData"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:CreateBucket",
                "s3:PutBucketPublicAccessBlock"
            ],
            "Resource": [
                "arn:aws:s3:::aws-glue-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::aws-glue-*/*",
                "arn:aws:s3:::*/*aws-glue-*/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::crawler-public*",
```

```
                "arn:aws:s3:::aws-glue-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:AssociateKmsKey"
            ],
            "Resource": [
                "arn:aws:logs:*:*:log-group:/aws-glue/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags",
                "ec2:DeleteTags"
            ],
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "aws-glue-service-resource"
                    ]
                }
            },
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:instance/*"
            ]
        }
    ]
}
```

Tabel berikut menjelaskan izin yang diberikan oleh kebijakan ini.

Aksi	Sumber	Deskripsi
`"glue:*"`	`"*"`	Pemberian izin untuk menjalankan semua operasi API AWS Glue.
`"s3:GetBucketLocation", "s3:ListBucket", "s3:ListAllMyBuckets", "s3:GetBucketAcl",`	`"*"`	Memungkinkan pencantuman bucket Amazon S3 dari crawler, tugas, titik akhir pengembangan, dan server notebook.
`"ec2:DescribeVpcEndpoints", "ec2:DescribeRouteTables", "ec2:CreateNetworkInterface", "ec2:DeleteNetworkInterface", "ec2:DescribeNetworkInterfaces", "ec2:DescribeSecurityGroups", "ec2:DescribeSubnets", "ec2:DescribeVpcAttribute",`	`"*"`	Memungkinkan setup item jaringan Amazon EC2, seperti virtual private cloud (VPC) ketika menjalankan tugas, crawler, dan titik akhir pengembangan.
`"iam:ListRolePolicies", "iam:GetRole", "iam:GetRolePolicy"`	`"*"`	Memungkinkan daftar IAM role dari crawler, tugas, titik akhir pengembangan, dan server notebook.
`"cloudwatch:PutMetricData"`	`"*"`	Memungkinkan menulis CloudWatch metrik untuk pekerjaan.

Aksi	Sumber	Deskripsi
`"s3:CreateBucket", "s3:PutBucketPublicAccessBlock"`	`"arn:aws:s3:::aws-glue-*"`	Memungkinkan pembuatan bucket Amazon S3 di akun Anda dari tugas dan server notebook. Konvensi penamaan: Menggunakan folder Amazon S3 bernama aws-glue-. Memungkinkan AWS Glue untuk membuat bucket yang memblokir akses publik.
`"s3:GetObject", "s3:PutObject", "s3:DeleteObject"`	`"arn:aws:s3:::aws-glue-*/*", "arn:aws:s3:::*/*aws-glue-*/*"`	Memungkinkan untuk mendapatkan, menempatkan, dan menghapus objek Amazon S3 ke akun Anda ketika menyimpan objek seperti skrip ETL dan lokasi server notebook. Konvensi penamaan: Pemberian izin untuk bucket Amazon S3 atau folder yang namanya menggunakan prefiks aws-glue-.

Aksi	Sumber	Deskripsi
`"s3:GetObject"`	`"arn:aws:s3:::crawler-public*", "arn:aws:s3:::aws-glue-*"`	Memungkinkan untuk mendapatkan objek Amazon S3 yang digunakan oleh contoh dan tutorial dari crawler dan tugas. Konvensi penamaan: Nama bucket Amazon S3 yang dimulai dengan crawler-public dan aws-glue-.
`"logs:CreateLogGroup", "logs:CreateLogStream", "logs:PutLogEvents"`	`"arn:aws:logs:*:*:log-group:/aws-glue/*"`	Memungkinkan menulis log ke CloudWatch Log. Konvensi penamaan: AWS Glue menulis log ke grup log yang namanya dimulai dengan aws-glue.
`"ec2:CreateTags", "ec2:DeleteTags"`	`"arn:aws:ec2:*:*:network-interface/*", "arn:aws:ec2:*:*:security-group/*", "arn:aws:ec2:*:*:instance/*"`	Memungkinkan penandaan sumber daya Amazon EC2 yang dibuat untuk titik akhir pengembangan. Konvensi penamaan: AWS Glue menandai antarmuka jaringan Amazon EC2, grup keamanan, dan instans dengan. aws-glue-service-resource

5. Pada layar Tinjau Kebijakan, masukkan Nama kebijakanAnda, misalnya GlueServiceRolePolicy. Masukkan deskripsi opsional, dan bila Anda puas dengan kebijakan ini, pilih Buat kebijakan.

Langkah 2: Buat peran IAM untuk AWS Glue

Anda harus memberikan izin IAM role yang dapat diambil oleh AWS Glue ketika memanggil layanan lain atas nama Anda. Izin ini termasuk izin akses ke Amazon S3 untuk sumber, target, skrip, dan direktori sementara yang Anda gunakan dengan AWS Glue. Izin diperlukan oleh crawler, tugas, dan titik akhir pengembangan.

Anda memberikan izin tersebut dengan menggunakan AWS Identity and Access Management (IAM). Menambahkan kebijakan ke IAM role yang Anda berikan ke AWS Glue.

Untuk membuat peran IAM untukAWS Glue

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Pilih Buat peran.
4. Untuk jenis peran, pilih Layanan AWS, temukan dan pilih Glue, dan pilih Berikutnya: Izin.
5. Pada halaman kebijakan Lampirkan izin, pilih kebijakan yang berisi izin yang diperlukan; misalnya, kebijakan terkelola untuk AWS Glue izin umum dan kebijakan AWS terkelola AmazonS3 **`AWSGlueServiceRole`** untuk akses FullAccess ke sumber daya Amazon S3. AWS Lalu pilih Selanjutnya: Tinjauan.

   > **Note**
   >
   > Pastikan bahwa salah satu kebijakan dalam peran ini memberikan izin ke sumber Amazon S3 dan target Anda. Anda mungkin ingin memberikan kebijakan Anda sendiri untuk akses ke sumber daya Amazon S3 tertentu. Sumber data memerlukan izin `s3:ListBucket` dan `s3:GetObject`. Target data memerlukan izin `s3:ListBucket`, `s3:PutObject`, dan `s3:DeleteObject`. Untuk informasi selengkapnya tentang membuat kebijakan Amazon S3 untuk sumber daya Anda, lihat Menentukan Sumber Daya dalam sebuah Kebijakan. Untuk contoh kebijakan Amazon S3, lihat Menulis Kebijakan IAM: Cara Memberikan Akses ke Bucket Amazon S3.
   > Jika Anda berencana untuk mengakses sumber dan target Amazon S3 yang dienkripsi dengan SSE-KMS, lampirkan sebuah kebijakan yang memungkinkan crawler, tugas, dan titik akhir pengembangan AWS Glue untuk mendekripsi data. Untuk informasi selengkapnya, lihat Melindungi Data Menggunakan Enkripsi Sisi Server dengan Kunci yang Dikelola AWS KMS (SSE-KMS).

Berikut adalah contohnya.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "kms:Decrypt"
         ],
         "Resource":[
            "arn:aws:kms:*:account-id-without-hyphens:key/key-id"
         ]
      }
   ]
}
```

6. Untuk nama Peran, masukkan nama untuk peran Anda; misalnya,`AWSGlueServiceRoleDefault`. Buat peran dengan nama yang diawali dengan string `AWSGlueServiceRole` untuk memungkinkan peran diteruskan dari pengguna konsol ke layanan. AWS Gluekebijakan yang diberikan mengharapkan peran layanan IAM dimulai`AWSGlueServiceRole`. Jika tidak, Anda harus menambahkan sebuah kebijakan untuk memungkinkan pengguna dengan izin `iam:PassRole` untuk IAM role agar sesuai dengan konvensi penamaan Anda. Pilih Buat Peran.

> **Note**
>
> Saat Anda membuat buku catatan dengan peran, peran tersebut kemudian diteruskan ke sesi interaktif sehingga peran yang sama dapat digunakan di kedua tempat. Dengan demikian, `iam:PassRole` izin harus menjadi bagian dari kebijakan peran.
> Buat kebijakan baru untuk peran Anda menggunakan contoh berikut. Ganti nomor akun dengan nomor Anda sendiri dan nama peran.
>
> ```
> {
>     "Version": "2012-10-17",
>     "Statement": [
>         {
>             "Effect": "Allow",
>             "Action": "iam:PassRole",
>             "Resource": "arn:aws:iam::090000000210:role/<role_name>"
> ```

```
        }
    ]
}
```

Langkah 3: Lampirkan kebijakan ke pengguna atau grup yang mengakses AWS Glue

Administrator harus menetapkan izin untuk setiap pengguna, grup, atau peran menggunakan AWS Glue konsol atau AWS Command Line Interface ()AWS CLI. Anda memberikan izin tersebut dengan menggunakan AWS Identity and Access Management (IAM), melalui kebijakan. Langkah ini menjelaskan penetapan izin kepada pengguna atau grup.

Ketika Anda menyelesaikan langkah ini, pengguna atau grup Anda memiliki kebijakan berikut yang dilampirkan:

- Kebijakan AWS terkelola `AWSGlueConsoleFullAccess` atau kebijakan kustom GlueConsoleAccessPolicy
- **`AWSGlueConsoleSageMakerNotebookFullAccess`**
- **`CloudWatchLogsReadOnlyAccess`**
- **`AWSCloudFormationReadOnlyAccess`**
- **`AmazonAthenaFullAccess`**

Untuk melampirkan kebijakan inline dan menyematkannya di pengguna atau grup

Anda dapat melampirkan kebijakan AWS terkelola atau kebijakan sebaris ke pengguna atau grup untuk mengakses AWS Glue konsol. Beberapa sumber daya yang ditentukan dalam kebijakan ini merujuk ke nama default yang digunakan oleh AWS Glue bucket Amazon S3, skrip ETL Amazon S3, Log CloudWatch , dan sumber daya Amazon EC2. AWS CloudFormation Sederhananya, AWS Glue menulis beberapa objek Amazon S3 ke dalam bucket di akun Anda yang mempunyai prefiks dengan `aws-glue-*` secara default.

> **Note**
>
> Anda dapat melewati langkah ini jika menggunakan kebijakan AWS terkelola **`AWSGlueConsoleFullAccess`**.

> **Important**
>
> AWS Glue memerlukan izin untuk mengambil peran yang digunakan untuk melakukan tugas atas nama Anda. Untuk mencapai hal ini, Anda menambahkan **`iam:PassRole`** izin ke AWS Glue pengguna atau grup Anda. Kebijakan ini memberikan izin untuk peran yang dimulai dengan `AWSGlueServiceRole` untuk peran layanan AWS Glue, dan `AWSGlueServiceNotebookRole` untuk peran yang diperlukan saat Anda membuat server notebook. Anda juga dapat membuat kebijakan Anda sendiri untuk izin `iam:PassRole` yang mengikuti konvensi penamaan Anda.
>
> Sesuai praktik keamanan terbaik, disarankan untuk membatasi akses dengan memperketat kebijakan untuk lebih membatasi akses ke bucket Amazon S3 dan grup log Amazon CloudWatch. Untuk contoh kebijakan Amazon S3, lihat Menulis Kebijakan IAM: Cara Memberikan Akses ke Bucket Amazon S3.

Pada langkah ini, Anda membuat sebuah kebijakan yang mirip dengan `AWSGlueConsoleFullAccess`. Anda dapat menemukan versi terbaru `AWSGlueConsoleFullAccess` pada konsol IAM.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Pengguna atau Grup pengguna.
3. Dalam daftar, pilih nama pengguna atau grup untuk menyematkan kebijakan.
4. Pilih tab Izin dan, jika diperlukan, perluas bagian Kebijakan izin.
5. Pilih tautan Tambahkan kebijakan inline.
6. Pada layar Buat kebijakan, navigasikan ke tab untuk mengedit JSON. Buat sebuah dokumen kebijakan dengan pernyataan JSON berikut, dan kemudian pilih Tinjau kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:*",
                "redshift:DescribeClusters",
                "redshift:DescribeClusterSubnetGroups",
                "iam:ListRoles",
```

```
                "iam:ListUsers",
                "iam:ListGroups",
                "iam:ListRolePolicies",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:ListAttachedRolePolicies",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeRouteTables",
                "ec2:DescribeVpcAttribute",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeInstances",
                "rds:DescribeDBInstances",
                "rds:DescribeDBClusters",
                "rds:DescribeDBSubnetGroups",
                "s3:ListAllMyBuckets",
                "s3:ListBucket",
                "s3:GetBucketAcl",
                "s3:GetBucketLocation",
                "cloudformation:DescribeStacks",
                "cloudformation:GetTemplateSummary",
                "dynamodb:ListTables",
                "kms:ListAliases",
                "kms:DescribeKey",
                "cloudwatch:GetMetricData",
                "cloudwatch:ListDashboards"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::*/*aws-glue-*/*",
                "arn:aws:s3:::aws-glue-*"
            ]
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "tag:GetResources"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:CreateBucket",
                "s3:PutBucketPublicAccessBlock"            ],
            "Resource": [
                "arn:aws:s3:::aws-glue-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:GetLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:*:*:/aws-glue/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack"
            ],
            "Resource": "arn:aws:cloudformation:*:*:stack/aws-glue*/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:image/*",
```

```
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:volume/*"
            ]
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:role/AWSGlueServiceRole*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "glue.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:role/AWSGlueServiceNotebookRole*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:iam::*:role/service-role/AWSGlueServiceRole*"
            ],
            "Condition": {
                "StringLike": {
```

```
                    "iam:PassedToService": [
                        "glue.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

Tabel berikut menjelaskan izin yang diberikan oleh kebijakan ini.

Aksi	Sumber	Deskripsi
`"glue:*"`	`"*"`	Pemberian izin untuk menjalankan semua operasi API AWS Glue. Jika sebelumnya Anda telah membuat kebijakan Anda tanpa tindakan `"glue:*"`, maka Anda harus menambahkan izin individu berikut ke kebijakan Anda: • “lem:ListCrawlers” • “lem:BatchGetCrawlers” • “lem:ListTriggers” • “lem:BatchGetTriggers” • “lem:ListDevEndpoints” • “lem:BatchGetDevEndpoints” • “lem:ListJobs” • “lem:BatchGetJobs”
`"redshift:DescribeClusters", "redshift:DescribeClusterSubnetGroups"`	`"*"`	Memungkinkan pembuatan koneksi ke Amazon Redshift.

Aksi	Sumber	Deskripsi
`"iam:ListRoles", "iam:ListRolePolicies", "iam:GetRole", "iam:GetRolePolicy", "iam:ListAttachedRolePolicies"`	`"*"`	Memungkinkan pencantuman IAM role saat bekerja dengan crawler, tugas, titik akhir pengembangan, dan server notebook.
`"ec2:DescribeSecurityGroups", "ec2:DescribeSubnets", "ec2:DescribeVpcs", "ec2:DescribeVpcEndpoints", "ec2:DescribeRouteTables", "ec2:DescribeVpcAttribute", "ec2:DescribeKeyPairs", "ec2:DescribeInstances"`	`"*"`	Memungkinkan setup item jaringan Amazon EC2, seperti VPC, saat menjalankan tugas, crawler, dan titik akhir pengembangan.
`"rds:DescribeDBInstances"`	`"*"`	Memungkinkan pembuatan koneksi ke Amazon RDS.
`"s3:ListAllMyBuckets", "s3:ListBucket", "s3:GetBucketAcl", "s3:GetBucketLocation"`	`"*"`	Memungkinkan pencantuman bucket Amazon S3 saat bekerja dengan crawler, tugas, titik akhir pengembangan, dan server notebook.
`"dynamodb:ListTables"`	`"*"`	Memungkinkan pencantuman tabel DynamoDB.
`"kms:ListAliases", "kms:DescribeKey"`	`"*"`	Memungkinkan bekerja dengan kunci KMS.

Aksi	Sumber	Deskripsi
`"cloudwatch:GetMetricData", "cloudwatch:ListDashboards"`	`"*"`	Memungkinkan bekerja dengan CloudWatch metrik.
`"s3:GetObject", "s3:PutObject"`	`"arn:aws:s3::: aws-glue-*/*", "arn:aws:s3::: */*aws-glue-*/*", "arn:aws:s3::: aws-glue-*"`	Memungkinkan untuk mendapatkan dan menempatkan objek Amazon S3 ke akun Anda ketika menyimpan objek seperti skrip ETL dan lokasi server notebook. Konvensi penamaan: Pemberian izin untuk bucket Amazon S3 atau folder yang namanya menggunakan prefiks aws-glue-.
`"tag:GetResources"`	`"*"`	Memungkinkan pengambilan tag AWS.

Aksi	Sumber	Deskripsi
`"s3:CreateBucket", "s3:PutBucketPublicAccessBlock"`	`"arn:aws:s3::: aws-glue-*"`	Memungkinkan pembuatan sebuah bucket Amazon S3 ke akun Anda saat menyimpan objek seperti skrip ETL dan lokasi server notebook. Konvensi penamaan: Pemberian izin untuk bucket Amazon S3 yang namanya menggunakan prefiks aws-glue-. Memungkinkan AWS Glue untuk membuat bucket yang memblokir akses publik.
`"logs:GetLogEvents"`	`"arn:aws:logs:*:*: /aws-glue/*"`	Memungkinkan pengambilan CloudWatch Log. Konvensi penamaan: AWS Glue menulis log untuk grup log yang namanya dimulai dengan aws-glue-.

Aksi	Sumber	Deskripsi
`"cloudformation:CreateStack", "cloudformation:DeleteStack"`	`"arn:aws:cloudformation:*:*:stack/aws-glue*/*"`	Memungkinkan pengelolaan tumpukan AWS CloudFormation saat bekerja dengan server notebook. Konvensi penamaan: AWS Glue menciptakan tumpukan yang namanya dimulai dengan aws-glue.
`"ec2:RunInstances"`	`"arn:aws:ec2:*:*:instance/*", "arn:aws:ec2:*:*:key-pair/*", "arn:aws:ec2:*:*:image/*", "arn:aws:ec2:*:*:security-group/*", "arn:aws:ec2:*:*:network-interface/*", "arn:aws:ec2:*:*:subnet/*", "arn:aws:ec2:*:*:volume/*"`	Memungkinkan untuk menjalankan titik akhir pengembangan dan server notebook.
`"iam:PassRole"`	`"arn:aws:iam::*:role/ AWSGlueServiceRole*"`	Memungkinkan AWS Glue untuk mengambil `PassRole` izin untuk peran yang dimulai dengan`AWSGlueServiceRole` .

Aksi	Sumber	Deskripsi
`"iam:PassRole"`	`"arn:aws:iam::*:role/ AWSGlueServiceNotebookRole*"`	Memungkinkan Amazon EC2 untuk mengambil `PassRole` izin untuk peran yang dimulai dengan. `AWSGlueServiceNotebookRole`
`"iam:PassRole"`	`"arn:aws:iam::*:role/service-role/AWSGlueServiceRole*"`	Memungkinkan AWS Glue untuk mengambil `PassRole` izin untuk peran yang dimulai dengan`service-role/AWSGlueServiceRole` .

7. Pada layar Kebijakan tinjauan, masukkan nama untuk kebijakan tersebut, misalnya GlueConsoleAccessPolicy. Jika Anda puas dengan kebijakan ini, pilih Buat kebijakan. Pastikan bahwa tidak ada kesalahan yang muncul di kotak merah yang ada di bagian atas layar. Perbaiki apapun yang dilaporkan.

> **Note**
>
> Jika Gunakan pemformatan otomatis dipilih, maka kebijakan akan diformat ulang setiap kali Anda membuka kebijakan atau memilih Validasi kebijakan.

Untuk melampirkan kebijakan AWSGlueConsoleFullAccess terkelola

Anda dapat melampirkan `AWSGlueConsoleFullAccess` kebijakan untuk memberikan izin yang diperlukan oleh pengguna AWS Glue konsol.

> **Note**
>
> Anda dapat melewati langkah ini jika Anda sudah membuat kebijakan Anda sendiri untuk akses konsol AWS Glue.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih kotak centang di sebelah AWSGlueConsoleFullAccess. Anda bisa memakai menu Filter dan kotak pencarian untuk mem-filter daftar kebijakan.
4. Pilih Tindakan kebijakan, lalu pilih Lampirkan.
5. Pilih pengguna untuk dilampiri kebijakan ini. Anda bisa menggunakan menu Filter dan kotak pencarian untuk mem-filter daftar entitas utama. Setelah memilih pengguna yang akan dilampiri kebijakan, pilih Lampirkan kebijakan.

Untuk melampirkan kebijakan **`AWSGlueConsoleSageMakerNotebookFullAccess`** terkelola

Anda dapat melampirkan `AWSGlueConsoleSageMakerNotebookFullAccess` kebijakan ke pengguna untuk mengelola SageMaker buku catatan yang dibuat di AWS Glue konsol. Selain izin AWS Glue konsol lain yang diperlukan, kebijakan ini memberikan akses ke sumber daya yang diperlukan untuk mengelola SageMaker buku catatan.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih kotak centang di sebelah AWSGlueConsoleSageMakerNotebookFullAccess. Anda bisa memakai menu Filter dan kotak pencarian untuk mem-filter daftar kebijakan.
4. Pilih Tindakan kebijakan, lalu pilih Lampirkan.
5. Pilih pengguna untuk dilampiri kebijakan ini. Anda bisa menggunakan menu Filter dan kotak pencarian untuk mem-filter daftar entitas utama. Setelah memilih pengguna yang akan dilampiri kebijakan, pilih Lampirkan kebijakan.

Untuk melampirkan kebijakan CloudWatchLogsReadOnlyAccess terkelola

Anda dapat melampirkan CloudWatchLogsReadOnlyAccesskebijakan ke pengguna untuk melihat log yang dibuat oleh AWS Glue pada konsol CloudWatch Log.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.

2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih kotak centang di sebelah CloudWatchLogsReadOnlyAccess. Anda bisa memakai menu Filter dan kotak pencarian untuk mem-filter daftar kebijakan.
4. Pilih Tindakan kebijakan, lalu pilih Lampirkan.
5. Pilih pengguna untuk dilampiri kebijakan ini. Anda bisa menggunakan menu Filter dan kotak pencarian untuk mem-filter daftar entitas utama. Setelah memilih pengguna yang akan dilampiri kebijakan, pilih Lampirkan kebijakan.

Untuk melampirkan kebijakan AWSCloudFormationReadOnlyAccess terkelola

Anda dapat melampirkan AWSCloudFormationReadOnlyAccesskebijakan ke pengguna untuk melihat AWS CloudFormation tumpukan yang digunakan AWS Glue di AWS CloudFormation konsol.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih kotak centang di sebelah AWSCloudFormationReadOnlyAccess. Anda bisa memakai menu Filter dan kotak pencarian untuk mem-filter daftar kebijakan.
4. Pilih Tindakan kebijakan, lalu pilih Lampirkan.
5. Pilih pengguna untuk dilampiri kebijakan ini. Anda bisa menggunakan menu Filter dan kotak pencarian untuk mem-filter daftar entitas utama. Setelah memilih pengguna yang akan dilampiri kebijakan, pilih Lampirkan kebijakan.

Untuk melampirkan kebijakan AmazonAthenaFullAccess terkelola

Anda dapat melampirkan AmazonAthenaFullAccesskebijakan ke pengguna untuk melihat data Amazon S3 di konsol Athena.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Kebijakan.
3. Dalam daftar kebijakan, pilih kotak centang di sebelah AmazonAthenaFullAccess. Anda bisa memakai menu Filter dan kotak pencarian untuk mem-filter daftar kebijakan.
4. Pilih Tindakan kebijakan, lalu pilih Lampirkan.

5. Pilih pengguna untuk dilampiri kebijakan ini. Anda bisa menggunakan menu Filter dan kotak pencarian untuk mem-filter daftar entitas utama. Setelah memilih pengguna yang akan dilampiri kebijakan, pilih Lampirkan kebijakan.

Langkah 4: Buat kebijakan IAM untuk server notebook

Jika Anda berencana untuk menggunakan notebook dengan titik akhir pengembangan, maka Anda harus menentukan izin ketika Anda membuat notebook server. Anda memberikan izin tersebut dengan menggunakan AWS Identity and Access Management (IAM).

Kebijakan ini memberikan izin untuk beberapa tindakan Amazon S3 untuk mengelola sumber daya di akun Anda yang diperlukan oleh AWS Glue ketika mengasumsikan peran dengan menggunakan kebijakan ini. Beberapa sumber daya yang ditentukan dalam kebijakan ini merujuk ke nama default yang digunakan oleh AWS Glue untuk bucket Amazon S3, skrip ETL Amazon S3, dan sumber daya Amazon EC2. Sederhananya, AWS Glue secara default menulis beberapa objek Amazon S3 ke dalam bucket di akun Anda yang mempunyai prefiks dengan `aws-glue-*`.

> **Note**
>
> Anda dapat melewati langkah ini jika menggunakan kebijakan AWS terkelola **`AWSGlueServiceNotebookRole`**.

Pada langkah ini, Anda membuat sebuah kebijakan yang mirip dengan `AWSGlueServiceNotebookRole`. Anda dapat menemukan versi terbaru `AWSGlueServiceNotebookRole` pada konsol IAM.

Untuk membuat sebuah kebijakan IAM untuk notebook

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di sebelah kiri, pilih Kebijakan.
3. Pilih Buat Kebijakan.
4. Pada layar Buat kebijakan, navigasikan ke tab untuk mengedit JSON. Buat sebuah dokumen kebijakan dengan pernyataan JSON berikut, dan kemudian pilih Tinjau kebijakan.

```
{
   "Version":"2012-10-17",
```

```
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "glue:CreateDatabase",
            "glue:CreatePartition",
            "glue:CreateTable",
            "glue:DeleteDatabase",
            "glue:DeletePartition",
            "glue:DeleteTable",
            "glue:GetDatabase",
            "glue:GetDatabases",
            "glue:GetPartition",
            "glue:GetPartitions",
            "glue:GetTable",
            "glue:GetTableVersions",
            "glue:GetTables",
            "glue:UpdateDatabase",
            "glue:UpdatePartition",
            "glue:UpdateTable",
            "glue:GetJobBookmark",
            "glue:ResetJobBookmark",
            "glue:CreateConnection",
            "glue:CreateJob",
            "glue:DeleteConnection",
            "glue:DeleteJob",
            "glue:GetConnection",
            "glue:GetConnections",
            "glue:GetDevEndpoint",
            "glue:GetDevEndpoints",
            "glue:GetJob",
            "glue:GetJobs",
            "glue:UpdateJob",
            "glue:BatchDeleteConnection",
            "glue:UpdateConnection",
            "glue:GetUserDefinedFunction",
            "glue:UpdateUserDefinedFunction",
            "glue:GetUserDefinedFunctions",
            "glue:DeleteUserDefinedFunction",
            "glue:CreateUserDefinedFunction",
            "glue:BatchGetPartition",
            "glue:BatchDeletePartition",
            "glue:BatchCreatePartition",
            "glue:BatchDeleteTable",
```

```
            "glue:UpdateDevEndpoint",
            "s3:GetBucketLocation",
            "s3:ListBucket",
            "s3:ListAllMyBuckets",
            "s3:GetBucketAcl"
         ],
         "Resource":[
            "*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
         ],
         "Resource":[
            "arn:aws:s3:::crawler-public*",
            "arn:aws:s3:::aws-glue*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:PutObject",
            "s3:DeleteObject"
         ],
         "Resource":[
            "arn:aws:s3:::aws-glue*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "ec2:CreateTags",
            "ec2:DeleteTags"
         ],
         "Condition":{
            "ForAllValues:StringEquals":{
               "aws:TagKeys":[
                  "aws-glue-service-resource"
               ]
            }
         },
         "Resource":[
```

```
            "arn:aws:ec2:*:*:network-interface/*",
            "arn:aws:ec2:*:*:security-group/*",
            "arn:aws:ec2:*:*:instance/*"
        ]
    }
  ]
}
```

Tabel berikut menjelaskan izin yang diberikan oleh kebijakan ini.

Aksi	Sumber Daya	Deskripsi
`"glue:*"`	`"*"`	Pemberian izin untuk menjalankan semua operasi API AWS Glue.
`"s3:GetBucketLocation", "s3:ListBucket", "s3:ListAllMyBuckets", "s3:GetBucketAcl"`	`"*"`	Memungkinkan pencantuman bucket Amazon S3 dari server notebook.
`"s3:GetObject"`	`"arn:aws:s3:::crawler-public*", "arn:aws:s3:::aws-glue-*"`	Memungkinkan untuk mendapatkan objek Amazon S3 yang digunakan oleh contoh dan tutorial dari notebook. Konvensi penamaan: Nama bucket Amazon S3 yang dimulai dengan crawler-public dan aws-glue-.

Aksi	Sumber Daya	Deskripsi
`"s3:PutObject", "s3:DeleteObject"`	`"arn:aws:s3:::aws-glue*"`	Memungkinkan untuk menempatkan dan menghapus objek Amazon S3 ke akun Anda dari notebook. Konvensi penamaan: Menggunakan folder Amazon S3 bernama aws-glue.
`"ec2:CreateTags", "ec2:DeleteTags"`	`"arn:aws:ec2:*:*:network-interface/*", "arn:aws:ec2:*:*:security-group/*", "arn:aws:ec2:*:*:instance/*"`	Memungkinkan penandaan sumber daya Amazon EC2 yang dibuat untuk server notebook. Konvensi penamaan: AWS Glue menandai instans Amazon EC2 dengan. aws-glue-service-resource

5. Pada layar Tinjau Kebijakan, masukkan Nama kebijakanAnda, misalnya GlueServiceNotebookPolicyDefault. Masukkan deskripsi opsional, dan bila Anda puas dengan kebijakan ini, pilih Buat kebijakan.

Langkah 5: Buat peran IAM untuk server notebook

Jika Anda berencana untuk menggunakan notebook dengan titik akhir pengembangan, Anda perlu memberikan izin IAM role. Anda memberikan izin tersebut dengan menggunakan AWS Identity and Access Management IAM, melalui IAM role.

> **Note**
>
> Saat Anda menciptakan sebuah IAM role menggunakan konsol IAM, konsol akan menciptakan profil instans secara otomatis dan memberikan nama yang sama sesuai dengan perannya.

Untuk membuat IAM role untuk notebook

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Pilih Buat peran.
4. Untuk jenis peran, pilih Layanan AWS, temukan dan pilih EC2, dan pilih kasus penggunaan EC2, lalu pilih Berikutnya: Izin.
5. Pada halaman Kebijakan izin melampirkan, pilih kebijakan yang berisi izin yang diperlukan; misalnya, untuk AWS Glue izin umum dan kebijakan AWS terkelola AmazonS3 AWSGlueServiceNotebookRoleuntuk akses FullAccess ke sumber daya Amazon S3. Lalu pilih Selanjutnya: Tinjauan.

> **Note**
>
> Pastikan bahwa salah satu kebijakan dalam peran ini memberikan izin ke sumber Amazon S3 dan target Anda. Selain itu, konfirmasi bahwa kebijakan Anda memungkinkan akses penuh ke lokasi di mana Anda menyimpan notebook Anda ketika Anda membuat sebuah server notebook. Anda mungkin ingin memberikan kebijakan Anda sendiri untuk akses ke sumber daya Amazon S3 tertentu. Untuk informasi selengkapnya tentang membuat kebijakan Amazon S3 untuk sumber daya Anda, lihat Menentukan Sumber Daya dalam sebuah Kebijakan.
> Jika Anda berencana untuk mengakses sumber dan target Amazon S3 yang dienkripsi dengan SSE-KMS, lampirkan sebuah kebijakan yang memungkinkan notebook untuk mendekripsi data. Untuk informasi selengkapnya, lihat Melindungi Data Menggunakan Enkripsi Sisi Server dengan Kunci yang Dikelola AWS KMS (SSE-KMS).
> Berikut adalah contohnya.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "kms:Decrypt"
         ],
         "Resource":[
            "arn:aws:kms:*:account-id-without-hyphens:key/key-id"
```

```
            ]
        }
    ]
}
```

6. Untuk Nama peran, masukkan nama untuk peran Anda. Buat peran dengan nama yang diawali dengan string `AWSGlueServiceNotebookRole` untuk memungkinkan peran diteruskan dari pengguna konsol ke server notebook. AWS Gluekebijakan yang diberikan mengharapkan peran layanan IAM dimulai`AWSGlueServiceNotebookRole`. Jika tidak, Anda harus menambahkan sebuah kebijakan untuk memungkinkan pengguna dengan izin `iam:PassRole` untuk IAM role agar sesuai dengan konvensi penamaan Anda. Misalnya, masukkan `AWSGlueServiceNotebookRoleDefault`. Lalu pilih Buat peran.

Langkah 6: Buat kebijakan IAM untuk notebook SageMaker

Jika Anda berencana untuk menggunakan SageMaker buku catatan dengan titik akhir pengembangan, Anda harus menentukan izin saat membuat buku catatan. Anda memberikan izin tersebut dengan menggunakan AWS Identity and Access Management (IAM).

Untuk membuat kebijakan IAM untuk buku catatan SageMaker

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di sebelah kiri, pilih Kebijakan.
3. Pilih Buat Kebijakan.
4. Pada halaman Buat kebijakan, navigasikan ke tab untuk mengedit JSON. Buat sebuah dokumen kebijakan dengan pernyataan JSON berikut. Edit *`bucket-name`*, *`region-code`*, dan *`account-id`* untuk lingkungan Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:ListBucket"
            ],
            "Effect": "Allow",
            "Resource": [
```

```
                "arn:aws:s3:::bucket-name"
            ]
        },
        {
            "Action": [
                "s3:GetObject"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::bucket-name*"
            ]
        },
        {
            "Action": [
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents",
                "logs:CreateLogGroup"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:logs:region-code:account-id:log-group:/aws/sagemaker/*",
                "arn:aws:logs:region-code:account-id:log-group:/aws/sagemaker/
*:log-stream:aws-glue-*"
            ]
        },
        {
            "Action": [
                "glue:UpdateDevEndpoint",
                "glue:GetDevEndpoint",
                "glue:GetDevEndpoints"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:glue:region-code:account-id:devEndpoint/*"
            ]
        },
        {
            "Action": [
                "sagemaker:ListTags"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:sagemaker:region-code:account-id:notebook-instance/*"
```

```
            ]
        }
    ]
}
```

Lalu pilih Peninjauan kebijakan.

Tabel berikut menjelaskan izin yang diberikan oleh kebijakan ini.

Aksi	Sumber Daya	Deskripsi
"s3:ListBucket*"	"arn:aws:s3::: *bucket-name* "	Berikan izin untuk membuat daftar bucket Amazon S3.
"s3:GetObject"	"arn:aws:s3::: *bucket-name* *"	Memberikan izin untuk mendapatkan objek Amazon S3 yang digunakan SageMaker oleh notebook.
"logs:CreateLogStream", "logs:DescribeLogStreams", "logs:PutLogEvents", "logs:CreateLogGroup"	"arn:aws:logs: *region-code* :*account-id* :log-group:/aws/sagemaker/*", "arn:aws:logs: *region-code* :*account-id* :log-group:/aws/sagemaker/*:log-stream:aws-glue-*"	Memberikan izin untuk menulis log ke Amazon CloudWatch Logs dari notebook. Konvensi penamaan: Menulis untuk grup log yang namanya dimulai dengan aws-glue.
"glue:UpdateDevEndpoint", "glue:GetDevEndpoint", "glue:GetDevEndpoints"	"arn:aws:glue: *region-code* :*account-id* :devEndpoint/*"	Memberikan izin untuk menggunakan titik akhir pengembangan dari SageMaker notebook.

Aksi	Sumber Daya	Deskripsi
`"sagemaker:ListTags"`	`"arn:aws:sagemaker : region-co de :account-i d :notebook-instance /*"`	Memberikan izin untuk mengembalikan tag untuk SageMaker sumber daya. `aws-glue-dev-endpo int` Tag diperlukan pada SageMaker notebook untuk menghubungkan notebook ke titik akhir pengembangan.

5. Pada layar Tinjau Kebijakan, masukkan Nama kebijakanAnda, misalnya `AWSGlueSageMakerNotebook`. Masukkan deskripsi opsional, dan bila Anda puas dengan kebijakan ini, pilih Buat kebijakan.

Langkah 7: Buat peran IAM untuk notebook SageMaker

Jika Anda berencana untuk menggunakan SageMaker buku catatan dengan titik akhir pengembangan, Anda harus memberikan izin peran IAM. Anda memberikan izin tersebut dengan menggunakan AWS Identity and Access Management (IAM), melalui IAM role.

Untuk membuat peran IAM untuk notebook SageMaker

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Pilih Buat peran.
4. Untuk tipe peran, pilih AWSLayanan, temukan dan pilih SageMaker, lalu pilih SageMaker - Kasus penggunaan eksekusi. Kemudian pilih Selanjutnya: Izin.
5. Pada halaman Kebijakan izin melampirkan, pilih kebijakan yang berisi izin yang diperlukan; misalnya,. AmazonSageMakerFullAccess Pilih Selanjutnya: Tinjauan.

   Jika Anda berencana untuk mengakses sumber dan target Amazon S3 yang dienkripsi dengan SSE-KMS, lampirkan sebuah kebijakan yang memungkinkan notebook untuk mendekripsi data, seperti yang ditunjukkan dalam contoh berikut. Untuk informasi selengkapnya, lihat Melindungi Data Menggunakan Enkripsi Sisi Server dengan Kunci yang Dikelola AWS KMS (SSE-KMS).

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "kms:Decrypt"
         ],
         "Resource":[
            "arn:aws:kms:*:account-id-without-hyphens:key/key-id"
         ]
      }
   ]
}
```

6. Untuk Nama peran, masukkan nama untuk peran Anda. Untuk mengizinkan peran diteruskan dari pengguna konsol ke SageMaker, gunakan nama yang diawali dengan string`AWSGlueServiceSageMakerNotebookRole`. AWS Glueasalkan kebijakan mengharapkan peran IAM dimulai`AWSGlueServiceSageMakerNotebookRole`. Jika tidak, Anda harus menambahkan sebuah kebijakan untuk memungkinkan pengguna dengan izin `iam:PassRole` untuk IAM role agar sesuai dengan konvensi penamaan Anda.

   Misalnya, masukkan`AWSGlueServiceSageMakerNotebookRole-Default`, lalu pilih Buat peran.

7. Setelah Anda membuat peran, lampirkan kebijakan yang memungkinkan izin tambahan yang diperlukan untuk membuat SageMaker buku catatan. AWS Glue

   Buka peran yang baru saja Anda buat`AWSGlueServiceSageMakerNotebookRole-Default`, dan pilih Lampirkan kebijakan. Lampirkan kebijakan yang Anda buat namanya `AWSGlueSageMakerNotebook` ke peran.

AWSContoh kebijakan kontrol akses Glue

Bagian ini berisi contoh kebijakan kontrol akses berbasis identitas (IAM) dan kebijakan sumber daya. AWS Glue

Daftar Isi

Contoh kebijakan berbasis identitas untuk Glue AWS

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau memodifikasi sumber daya AWS Glue. Pengguan dan peran tersebut juga tidak dapat melakukan tugas dengan menggunakan API AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS. Untuk mengabulkan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian akan dapat menambahkan kebijakan IAM ke peran, dan para pengguna dapat mengambil peran.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM menggunakan contoh dokumen kebijakan JSON ini, silakan lihat Membuat kebijakan IAM di Panduan Pengguna IAM.

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh AWS Glue, termasuk format ARN untuk setiap jenis sumber daya, lihat Kunci tindakan, sumber daya, dan kondisi untuk AWS Glue di Referensi Otorisasi Layanan.

> **Note**
>
> Contoh yang diberikan di bagian ini semuanya menggunakan `us-west-2` Wilayah. Anda dapat menggantinya dengan AWS Wilayah yang ingin Anda gunakan.

Topik

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya AWS Glue di akun Anda. Tindakan ini mengenakan biaya kepada Anda Akun AWS. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan terkelola AWS dan beralih ke izin dengan hak akses paling rendah – Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan terkelola AWS yang memberikan izin untuk banyak kasus penggunaan umum. Kebijakan tedapat di Akun AWS Anda. Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan AWS yang khusus untuk kasus penggunaan Anda. Untuk

informasi selengkapnya, silakan lihat kebijakan-kebijakan terkelola AWS atau kebijakan-kebijakan terkelola AWS untuk fungsi tugas di Panduan Pengguna IAM.

- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukan ini dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan pengguna IAM untuk mengajukan izin, silakan lihat Kebijakan dan izin di IAM di Panduan Pengguna IAM.
- Gunakan syarat dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu syarat ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis syarat kebijakan untuk menentukan bahwa semua pengajuan harus dikirim menggunakan SSL. Anda juga dapat menggunakan syarat untuk memberi akses ke tindakan layanan jika digunakan melalui Layanan AWS yang spesifk, sepertiAWS CloudFormation. Untuk informasi selengkapnya, silakan lihat Elemen kebijakan JSON IAM: Syarat di Panduan Pengguna IAM.
- Gunakan Analizer Akses IAM untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – Analizer Akses IAM memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. Analizer Akses IAM menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, silakan lihat validasi kebijakan Analizer Akses IAM di Panduan Pengguna IAM.
- Memerlukan autentikasi multi-faktor (MFA) – Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Akun AWS Anda, aktifkan MFA untuk keamanan tambahan. Untuk meminta MFA ketika operasi API dipanggil, tambahkan syarat MFA pada kebijakan Anda. Untuk informasi selengkapnya, silakan lihat Mengonfigurasi akses API yang diproteksi MFA di Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, silakan lihat Praktik terbaik keamanan di IAM di Panduan Pengguna IAM.

Izin tingkat sumber daya hanya berlaku untuk objek tertentu AWS Glue

Anda hanya dapat menentukan kontrol berbutir halus untuk objek tertentu di. AWS Glue Oleh karena itu Anda harus menulis kebijakan IAM dari klien Anda sehingga operasi API yang mengizinkan

Amazon Resource Names (ARN) untuk pernyataan `Resource` tidak dicampur dengan operasi API yang tidak mengizinkan ARN.

Misalnya, kebijakan IAM berikut memungkinkan operasi API untuk `GetClassifier` dan `GetJobRun`. Ini mendefinisikan `Resource` sebagai * karena AWS Glue tidak mengizinkan ARN untuk pengklasifikasi dan menjalankan pekerjaan. Karena ARN diperbolehkan untuk operasi API tertentu seperti `GetDatabase` dan `GetTable`, maka ARN dapat ditentukan pada setengah kedua kebijakan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetClassifier*",
                "glue:GetJobRun*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:Get*"
            ],
            "Resource": [
                "arn:aws:glue:us-east-1:123456789012:catalog",
                "arn:aws:glue:us-east-1:123456789012:database/default",
                "arn:aws:glue:us-east-1:123456789012:table/default/e*1*",
                "arn:aws:glue:us-east-1:123456789012:connection/connection2"
            ]
        }
    ]
}
```

Untuk daftar AWS Glue objek yang memungkinkan ARN, lihatARN Sumber Daya.

Menggunakan konsol AWS Glue

Untuk mengakses konsol AWS Glue, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang sumber daya AWS Glue di AndaAkun AWS. Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada

izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna atau peran) dengan kebijakan tersebut.

Anda tidak perlu meloloskan izin konsol minimum bagi pengguna yang hanya melakukan panggilan ke API AWS CLI atau AWS. Jika tidak, akses hanya diizinkan ke tindakan-tindakan yang sesuai dengan operasi API yang sedang mereka coba lakukan.

Untuk memastikan bahwa pengguna dan peran masih dapat menggunakan konsol AWS Glue, lampirkan juga AWS Glue `ConsoleAccess` atau kebijakan `ReadOnly` AWS terkelola ke entitas. Untuk informasi selengkapnya, silakan lihat Menambah izin untuk pengguna di Panduan Pengguna IAM.

Agar pengguna dapat bekerja dengan konsol AWS Glue, pengguna tersebut harus memiliki seperangkat izin minimum yang memungkinkannya untuk menggunakan sumber daya AWS Glue untuk akun AWS mereka. Selain izin AWS Glue ini, konsol memerlukan izin dari layanan berikut:

- Izin CloudWatch Log Amazon untuk menampilkan log.
- Izin AWS Identity and Access Management (IAM) untuk mencatumkan dan memberikan peran.
- Izin AWS CloudFormation untuk menggunakan tumpukan.
- Izin Amazon Elastic Compute Cloud (Amazon EC2) untuk mencantumkan VPC, subnet, grup keamanan, instans, dan objek lainnya.
- Izin Amazon Simple Storage Service (Amazon S3) untuk mencantumkan bucket dan objek, dan untuk mengambil dan menyimpan skrip.
- Izin Amazon Redshift untuk bekerja dengan klaster.
- Izin Amazon Relational Database Service (Amazon RDS) untuk mencantumkan instans.

Untuk informasi selengkapnya tentang izin yang diperlukan pengguna untuk melihat dan bekerja dengan AWS Glue konsol, lihatLangkah 3: Lampirkan kebijakan ke pengguna atau grup yang mengakses AWS Glue.

Jika Anda membuat kebijakan IAM yang lebih ketat dari izin minimum yang diperlukan, konsol tersebut tidak akan berfungsi sebagaimana dimaksudkan untuk pengguna dengan kebijakan IAM tersebut. Untuk memastikan bahwa pengguna tersebut masih dapat menggunakan AWS Glue konsol, lampirkan juga kebijakan `AWSGlueConsoleFullAccess` terkelola seperti yang dijelaskan dalamAWS kebijakan terkelola (standar) untuk AWS Glue.

Izinkan para pengguna untuk melihat izin mereka sendiri

Contoh ini menunjukkan cara Anda dapat membuat kebijakan yang mengizinkan para pengguna IAM untuk melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan pada konsol atau secara terprogram menggunakan API AWS CLI atau AWS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Berikan izin hanya-baca ke tabel

Kebijakan ini memberikan izin baca-saja ke sebuah tabel `books` dalam basis data `db1`. Untuk informasi selengkapnya tentang sumber daya Amazon Resource Name (ARN), lihat ARN Katalog Data.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "GetTablesActionOnBooks",
      "Effect": "Allow",
      "Action": [
        "glue:GetTables",
        "glue:GetTable"
      ],
      "Resource": [
        "arn:aws:glue:us-west-2:123456789012:catalog",
        "arn:aws:glue:us-west-2:123456789012:database/db1",
        "arn:aws:glue:us-west-2:123456789012:table/db1/books"
      ]
    }
  ]
}
```

Kebijakan ini memberikan izin baca-saja ke tabel bernama `books` dalam basis data bernama `db1`. Untuk memberikan `Get` izin ke tabel, izin ke katalog dan sumber daya database juga diperlukan.

Kebijakan berikut memberikan izin minimum yang diperlukan untuk membuat tabel `tbl` dalam basis data `db1`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:CreateTable"
      ],
      "Resource": [
        "arn:aws:glue:us-west-2:123456789012:table/db1/tbl1",
        "arn:aws:glue:us-west-2:123456789012:database/db1",
        "arn:aws:glue:us-west-2:123456789012:catalog"
```

```
      ]
    }
  ]
}
```

Filter tabel dengan GetTables izin

Asumsikan bahwa ada tiga tabel—yaitu, `customers`, `stores`, dan `store_sales`—dalam basis data db1. Kebijakan berikut memberikan izin `GetTables` untuk `stores` dan `store_sales`, tetapi tidak untuk `customers`. Ketika Anda memanggil `GetTables` dengan kebijakan ini, hasilnya hanya berisi dua tabel yang mendapat otorisasi (tabel `customers` tidak dikembalikan).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GetTablesExample",
            "Effect": "Allow",
            "Action": [
                "glue:GetTables"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:database/db1",
                "arn:aws:glue:us-west-2:123456789012:table/db1/store_sales",
                "arn:aws:glue:us-west-2:123456789012:table/db1/stores"
            ]
        }
    ]
}
```

Anda dapat menyederhanakan kebijakan sebelumnya dengan menggunakan `store*` untuk mencocokkan dengan nama tabel yang dimulai dengan `store`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GetTablesExample2",
            "Effect": "Allow",
            "Action": [
                "glue:GetTables"
```

```
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:database/db1",
                "arn:aws:glue:us-west-2:123456789012:table/db1/store*"
            ]
        }
    ]
}
```

Demikian pula, menggunakan `/db1/*` untuk mencocokkan dengan semua tabel di `db1`, kebijakan berikut memberikan akses `GetTables` ke semua tabel di `db1`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GetTablesReturnAll",
            "Effect": "Allow",
            "Action": [
                "glue:GetTables"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:database/db1",
                "arn:aws:glue:us-west-2:123456789012:table/db1/*"
            ]
        }
    ]
}
```

Jika tidak ada ARN tabel yang disediakan, panggilan untuk `GetTables` berhasil, tetapi ia hanya mengembalikan daftar kosong.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GetTablesEmptyResults",
            "Effect": "Allow",
            "Action": [
                "glue:GetTables"
```

```
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:database/db1"
            ]
        }
    ]
}
```

Jika basis data ARN tidak ada dalam kebijakan, maka panggilan ke `GetTables` akan gagal dengan sebuah `AccessDeniedException`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "GetTablesAccessDeny",
            "Effect": "Allow",
            "Action": [
                "glue:GetTables"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:table/db1/*"
            ]
        }
    ]
}
```

Berikan akses penuh ke tabel dan semua partisi

Kebijakan ini memberikan semua izin pada sebuah tabel bernama `books` dalam basis data `db1`. Ini termasuk izin baca dan tulis pada tabel itu sendiri, pada versi yang diarsipkannya, dan pada semua partisinya.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessOnTable",
            "Effect": "Allow",
            "Action": [
```

```
                "glue:CreateTable",
                "glue:GetTable",
                "glue:GetTables",
                "glue:UpdateTable",
                "glue:DeleteTable",
                "glue:BatchDeleteTable",
                "glue:GetTableVersion",
                "glue:GetTableVersions",
                "glue:DeleteTableVersion",
                "glue:BatchDeleteTableVersion",
                "glue:CreatePartition",
                "glue:BatchCreatePartition",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:BatchGetPartition",
                "glue:UpdatePartition",
                "glue:DeletePartition",
                "glue:BatchDeletePartition"
            ],
            "Resource": [
                        "arn:aws:glue:us-west-2:123456789012:catalog",
                        "arn:aws:glue:us-west-2:123456789012:database/db1",
                        "arn:aws:glue:us-west-2:123456789012:table/db1/books"
                        ]
        }
    ]
}
```

Kebijakan sebelumnya dapat disederhanakan dalam prakteknya.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "FullAccessOnTable",
            "Effect": "Allow",
            "Action": [
                "glue:*Table*",
                "glue:*Partition*"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:database/db1",
```

```
                "arn:aws:glue:us-west-2:123456789012:table/db1/books"
            ]
        }
    ]
}
```

Perhatikan bahwa kedetailan minimum dari kontrol akses detail adalah pada tingkat tabel. Ini berarti bahwa Anda tidak dapat memberikan akses kepada pengguna ke beberapa partisi dalam sebuah tabel, tetapi tidak dalam tabel lainnya, atau beberapa kolom tabel, tetapi tidak lainnya. Seorang pengguna memiliki akses ke semua tabel, atau tidak sama sekali.

Kontrol akses dengan awalan nama dan penolakan eksplisit

Dalam contoh ini, misalkan basis data dan tabel di Katalog Data Glue AWS Anda diatur menggunakan prefiks nama. Basis data dalam tahap pengembangan memiliki nama prefiks `dev-`, dan yang dalam produksi memiliki prefiks nama `prod-`. Anda dapat menggunakan kebijakan berikut untuk memberikan developer akses penuh ke semua basis data, tabel, UDF, dan sebagainya, yang memiliki prefiks `dev-`. Namun Anda memberikan akses baca-saja ke semua yang memiliki prefiks `prod-`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DevAndProdFullAccess",
            "Effect": "Allow",
            "Action": [
                "glue:*Database*",
                "glue:*Table*",
                "glue:*Partition*",
                "glue:*UserDefinedFunction*",
                "glue:*Connection*"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:catalog",
                "arn:aws:glue:us-west-2:123456789012:database/dev-*",
                "arn:aws:glue:us-west-2:123456789012:database/prod-*",
                "arn:aws:glue:us-west-2:123456789012:table/dev-*/*",
                "arn:aws:glue:us-west-2:123456789012:table/*/dev-*",
                "arn:aws:glue:us-west-2:123456789012:table/prod-*/*",
                "arn:aws:glue:us-west-2:123456789012:table/*/prod-*",
                "arn:aws:glue:us-west-2:123456789012:userDefinedFunction/dev-*/*",
```

```
                "arn:aws:glue:us-west-2:123456789012:userDefinedFunction/*/dev-*",
                "arn:aws:glue:us-west-2:123456789012:userDefinedFunction/prod-*/*",
                "arn:aws:glue:us-west-2:123456789012:userDefinedFunction/*/prod-*",
                "arn:aws:glue:us-west-2:123456789012:connection/dev-*",
                "arn:aws:glue:us-west-2:123456789012:connection/prod-*"
            ]
        },
        {
            "Sid": "ProdWriteDeny",
            "Effect": "Deny",
            "Action": [
                "glue:*Create*",
                "glue:*Update*",
                "glue:*Delete*"
            ],
            "Resource": [
                "arn:aws:glue:us-west-2:123456789012:database/prod-*",
                "arn:aws:glue:us-west-2:123456789012:table/prod-*/*",
                "arn:aws:glue:us-west-2:123456789012:table/*/prod-*",
                "arn:aws:glue:us-west-2:123456789012:userDefinedFunction/prod-*/*",
                "arn:aws:glue:us-west-2:123456789012:userDefinedFunction/*/prod-*",
                "arn:aws:glue:us-west-2:123456789012:connection/prod-*"
            ]
        }
    ]
}
```

Pernyataan kedua dalam kebijakan sebelumnya menggunakan `deny` eksplisit. Anda dapat menggunakan `deny` eksplisit untuk menimpa izin `allow` yang diberikan kepada prinsipal utama. Hal ini memungkinkan Anda mengunci akses ke sumber daya penting dan mencegah kebijakan lain memberikan akses kepada mereka secara tidak sengaja.

Dalam contoh sebelumnya, meskipun pernyataan pertama memberikan akses penuh ke sumber daya `prod-`, pernyataan kedua secara eksplisit mencabut akses tulis ke mereka, yang hanya menyisakan akses baca ke sumber daya `prod-`.

Berikan akses menggunakan tag

Misalnya, anggaplah Anda ingin membatasi akses ke sebuah pemicu `t2` ke pengguna tertentu bernama `Tom` di akun Anda. Semua pengguna lain, termasuk `Sam`, memiliki akses ke pemicu `t1`. Pemicu `t1` dan `t2` memiliki properti berikut.

```
aws glue get-triggers
{
    "Triggers": [
        {
            "State": "CREATED",
            "Type": "SCHEDULED",
            "Name": "t1",
            "Actions": [
                {
                    "JobName": "j1"
                }
            ],
            "Schedule": "cron(0 0/1 * * ? *)"
        },
        {
            "State": "CREATED",
            "Type": "SCHEDULED",
            "Name": "t2",
            "Actions": [
                {
                    "JobName": "j1"
                }
            ],
            "Schedule": "cron(0 0/1 * * ? *)"
        }
    ]
}
```

AWS GlueAdministrator melampirkan nilai tag `Tom` (`aws:ResourceTag/Name": "Tom"`) untuk memicu`t2`. AWS GlueAdministrator juga memberi Tom kebijakan IAM dengan pernyataan kondisi berdasarkan tag. Akibatnya, Tom hanya dapat menggunakan AWS Glue operasi yang bekerja pada sumber daya dengan nilai tag`Tom`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
         "Effect": "Allow",
         "Action": "glue:*",
         "Resource": "*",
         "Condition": {
             "StringEquals": {
```

```
                "aws:ResourceTag/Name": "Tom"
              }
           }
          }
    ]
}
```

Ketika Tom mencoba mengakses pemicu `t1`, ia menerima pesan akses ditolak. Sementara itu, ia dapat berhasil mengambil pemicu `t2`.

```
aws glue get-trigger --name t1

An error occurred (AccessDeniedException) when calling the GetTrigger operation:
 User: Tom is not authorized to perform: glue:GetTrigger on resource: arn:aws:glue:us-
east-1:123456789012:trigger/t1

aws glue get-trigger --name t2
{
    "Trigger": {
        "State": "CREATED",
        "Type": "SCHEDULED",
        "Name": "t2",
        "Actions": [
            {
                "JobName": "j1"
            }
        ],
        "Schedule": "cron(0 0/1 * * ? *)"
    }
}
```

Tom tidak dapat menggunakan operasi `GetTriggers` API jamak untuk membuat daftar pemicu karena operasi ini tidak mendukung pemfilteran pada tag.

Untuk memberi Tom aksesGetTriggers, AWS Glue administrator membuat kebijakan yang membagi izin menjadi dua bagian. Satu bagian memungkinkan Tom mengakses semua pemicu dengan operasi API `GetTriggers`. Bagian yang lain memungkinkan Tom mengakses ke operasi API yang ditandai dengan nilai `Tom`. Dengan kebijakan ini, Tom diperbolehkan menggunakan akses `GetTriggers` dan `GetTrigger` ke pemicu `t2`.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "glue:GetTriggers",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "glue:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Name": "Tom"
                }
            }
        }
    ]
}
```

Tolak akses menggunakan tag

Pendekatan kebijakan sumber daya lainnya adalah dengan secara eksplisit menolak akses ke sumber daya.

> **Important**
>
> Kebijakan penolakan eksplisit tidak berfungsi untuk operasi API jamak seperti. `GetTriggers`

Dalam contoh kebijakan berikut, semua operasi AWS Glue pekerjaan diperbolehkan. Namun, `Effect` pernyataan kedua secara eksplisit menolak akses ke pekerjaan yang ditandai dengan kunci dan nilai. `Team` `Special`

Ketika administrator melampirkan kebijakan berikut ke identitas, identitas dapat mengakses semua pekerjaan kecuali yang ditandai dengan `Team` kunci dan `Special` nilai.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": "glue:*",
      "Resource": "arn:aws:glue:us-east-1:123456789012:job/*"
    },
    {
      "Effect": "Deny",
      "Action": "glue:*",
      "Resource": "arn:aws:glue:us-east-1:123456789012:job/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Team": "Special"
        }
      }
    }
  ]
}
```

Gunakan tag dengan operasi API daftar dan batch

Pendekatan ketiga untuk penulisan kebijakan sumber daya adalah dengan memungkinkan akses ke sumber daya menggunakan operasi API `List` untuk mencantumkan sumber daya untuk nilai tag. Kemudian, gunakan operasi API `Batch` yang sesuai untuk memungkinkan akses ke detail sumber daya tertentu. Dengan pendekatan ini, administrator tidak perlu mengizinkan akses ke operasi API `GetCrawlers`, `GetDevEndpoints`, `GetJobs`, atau `GetTriggers` majemuk. Sebaliknya, Anda dapat memungkinkan kemampuan untuk mencantumkan sumber daya dengan operasi API berikut:

- `ListCrawlers`
- `ListDevEndpoints`
- `ListJobs`
- `ListTriggers`

Dan, Anda dapat memungkinkan kemampuan untuk mendapatkan detail tentang masing-masing sumber daya dengan operasi API berikut:

- `BatchGetCrawlers`
- `BatchGetDevEndpoints`
- `BatchGetJobs`
- `BatchGetTriggers`

Sebagai administrator, untuk menggunakan pendekatan ini, Anda dapat melakukan hal berikut ini:

1. Tambahkan tag ke crawler, titik akhir pengembangan, tugas, dan pemicu Anda.
2. Tolak akses pengguna ke operasi API `Get` seperti `GetCrawlers`, `GetDevEndponts`, `GetJobs`, dan `GetTriggers`.
3. Untuk memungkinkan pengguna untuk mengetahui sumber daya yang ditandai yang mana yang mereka miliki aksesnya, memungkinkan akses pengguna ke operasi API `List` seperti `ListCrawlers`, `ListDevEndponts`, `ListJobs`, dan `ListTriggers`.
4. Tolak akses pengguna ke API AWS Glue penandaan, seperti `TagResource` dan`UntagResource`.
5. Izinkan akses pengguna ke detail sumber daya dengan operasi API `BatchGet` seperti `BatchGetCrawlers`, `BatchGetDevEndponts`, `BatchGetJobs`, dan `BatchGetTriggers`.

Misalnya, saat memanggil operasi `ListCrawlers`, berikan nilai tag yang cocok dengan nama pengguna. Maka hasilnya adalah sebuah daftar crawler yang cocok dengan nilai tag yang disediakan. Berikan daftar nama `BatchGetCrawlers` untuk mendapatkan detail tentang setiap crawler dengan tag yang diberikan.

Misalnya, jika Tom hanya dapat mengambil detail pemicu yang ditandai, administrator dapat menambahkan tag ke pemicu`Tom`, menolak akses ke operasi `GetTriggers` API ke semua pengguna`Tom`, dan mengizinkan akses ke semua pengguna ke dan. `ListTriggers` `BatchGetTriggers`

Berikut ini adalah kebijakan sumber daya yang AWS Glue diberikan administrator kepada Tom. Di bagian pertama kebijakan, operasi AWS Glue API ditolak untuk`GetTriggers`. Pada bagian kedua dari kebijakan tersebut, `ListTriggers` diperbolehkan untuk semua sumber daya. Namun demikian, di bagian ketiga, sumber daya yang ditandai dengan `Tom` tersebut diperbolehkan mengakses dengan akses `BatchGetTriggers`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "glue:GetTriggers",
            "Resource": "*"
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "glue:ListTriggers"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:BatchGetTriggers"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Name": "Tom"
                }
            }
        }
    ]
}
```

Dengan menggunakan pemicu yang sama seperti contoh sebelumnya, Tom dapat mengakses pemicu `t2`, tapi tidak pemicu `t1`. Contoh berikut menunjukkan hasil ketika Tom mencoba untuk mengakses `t1` dan `t2` dengan `BatchGetTriggers`.

```
aws glue batch-get-triggers --trigger-names t2
{
    "Triggers": {
        "State": "CREATED",
        "Type": "SCHEDULED",
        "Name": "t2",
        "Actions": [
            {
                "JobName": "j2"
            }
        ],
        "Schedule": "cron(0 0/1 * * ? *)"
    }
}
```

```
aws glue batch-get-triggers  --trigger-names t1

An error occurred (AccessDeniedException) when calling the BatchGetTriggers operation:
 No access to any requested resource.
```

Contoh berikut menunjukkan hasil ketika Tom mencoba untuk mengakses pemicu `t2` dan pemicu `t3` (yang tidak ada) dalam panggilan `BatchGetTriggers` yang sama. Perhatikan bahwa karena Tom memiliki akses ke pemicu `t2` dan itu ada, hanya `t2` yang dikembalikan. Meskipun Tom diperbolehkan untuk mengakses pemicu `t3`, namun pemicu `t3` tidak ada, jadi `t3` dikembalikan dalam respon dalam sebuah daftar `"TriggersNotFound": []`.

```
aws glue batch-get-triggers --trigger-names t2 t3
{
    "Triggers": {
        "State": "CREATED",
        "Type": "SCHEDULED",
        "Name": "t2",
        "Actions": [
            {
                "JobName": "j2"
            }
        ],
        "TriggersNotFound": ["t3"],
        "Schedule": "cron(0 0/1 * * ? *)"
    }
}
```

Pengaturan kontrol menggunakan tombol kondisi atau tombol konteks

Anda dapat menggunakan tombol kondisi atau kunci konteks saat memberikan izin untuk membuat dan memperbarui pekerjaan. Bagian-bagian ini membahas kunci:

- Kebijakan kontrol yang mengontrol pengaturan menggunakan tombol kondisi
- Kebijakan kontrol yang mengontrol setelan menggunakan tombol konteks

Kebijakan kontrol yang mengontrol pengaturan menggunakan tombol kondisi

AWSGlue menyediakan tiga kunci kondisi IAM`glue:VpcIds`,`glue:SubnetIds`, dan`glue:SecurityGroupIds`. Anda dapat menggunakan kunci kondisi dalam kebijakan IAM saat

memberikan izin untuk membuat dan memperbarui pekerjaan. Anda dapat menggunakan pengaturan ini untuk memastikan bahwa pekerjaan atau sesi tidak dibuat (atau diperbarui ke) untuk berjalan di luar lingkungan VPC yang diinginkan. Informasi pengaturan VPC bukanlah input langsung dari `CreateJob` permintaan, tetapi disimpulkan dari bidang “koneksi” pekerjaan yang menunjuk ke koneksi. AWS Glue

Contoh penggunaan

Buat koneksi tipe AWS Glue jaringan bernama "traffic-monitored-connection" dengan “ VpcId vpc-id1234" yang diinginkan,, dan. SubnetIds SecurityGroupIds

Tentukan kondisi kunci kondisi untuk `CreateJob` dan `UpdateJob` tindakan dalam kebijakan IAM.

```
{
  "Effect": "Allow",
  "Action": [
    "glue:CreateJob",
    "glue:UpdateJob"
  ],
  "Resource": [
    "*"
  ],
  "Condition": {
    "ForAnyValue:StringLike": {
      "glue:VpcIds": [
        "vpc-id1234"
      ]
    }
  }
}
```

Anda dapat membuat kebijakan IAM serupa untuk melarang membuat AWS Glue pekerjaan tanpa menentukan informasi koneksi.

Membatasi sesi pada VPC

<123>Untuk menegakkan sesi yang dibuat agar berjalan dalam VPC tertentu, Anda membatasi izin peran dengan menambahkan `Deny` efek pada `glue:CreateSession` tindakan dengan syarat lem: vpc-id tidak sama dengan vpc-. Misalnya:

```
"Effect": "Deny",
```

```
"Action": [
    "glue:CreateSession"
 ],
"Condition": {
    "StringNotEquals" : {"glue:VpcIds" : ["vpc-123"]}
}
```

Anda juga dapat menerapkan sesi yang dibuat untuk dijalankan dalam VPC dengan menambahkan `Deny` efek pada `glue:CreateSession` tindakan dengan syarat bahwa `glue:vpc-id` adalah null. Misalnya:

```
{
    "Effect": "Deny",
    "Action": [
        "glue:CreateSession"
    ],
      "Condition": {
        "Null": {"glue:VpcIds": true}
    }
},
{
    "Effect": "Allow",
    "Action": [
        "glue:CreateSession"
    ],
    "Resource": ["*"]
}
```

Kebijakan kontrol yang mengontrol setelan menggunakan tombol konteks

AWSGlue menyediakan kunci konteks (`glue:CredentialIssuingService=glue.amazonaws.com`) untuk setiap sesi peran AWS Glue yang tersedia untuk pekerjaan dan titik akhir pengembang. Ini memungkinkan Anda untuk menerapkan kontrol keamanan untuk tindakan yang diambil oleh AWS Glue skrip. AWS Gluemenyediakan kunci konteks lain (`glue:RoleAssumedBy=glue.amazonaws.com`) untuk setiap sesi peran di mana AWS Glue melakukan panggilan ke AWS layanan lain atas nama pelanggan (bukan oleh titik akhir job/dev, tetapi langsung oleh layanan). AWS Glue

Contoh penggunaan

Tentukan izin bersyarat dalam kebijakan IAM dan lampirkan ke peran yang akan digunakan oleh pekerjaan. AWS Glue Ini memastikan tindakan tertentu diizinkan/ditolak berdasarkan apakah sesi peran digunakan untuk lingkungan runtime AWS Glue pekerjaan.

```
{
    "Effect": "Allow",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::confidential-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:CredentialIssuingService": "glue.amazonaws.com"
        }
    }
}
```

Menolak identitas kemampuan untuk membuat sesi pratinjau data

Bagian ini berisi contoh kebijakan IAM yang digunakan untuk menolak identitas kemampuan untuk membuat sesi pratinjau data. Lampirkan kebijakan ini ke identitas, yang terpisah dari peran yang digunakan oleh sesi pratinjau data selama dijalankan.

```
{
    "Sid": "DatapreviewDeny",
    "Effect": "Deny",
     "Action": [
           "glue:CreateSession"
     ],
     "Resource": [
          "arn:aws:glue:*:*:session/glue-studio-datapreview*"
      ]
 }
```

Contoh kebijakan berbasis sumber daya untuk Glue AWS

Bagian ini berisi contoh kebijakan berbasis sumber daya, termasuk kebijakan yang memberikan akses lintas akun.

Contoh menggunakan AWS Command Line Interface (AWS CLI) untuk berinteraksi dengan operasi API AWS Glue layanan. Anda dapat melakukan operasi yang sama di AWS Glue konsol atau menggunakan salah satu AWS SDK.

> **Important**
>
> Dengan mengubah kebijakan AWS Glue sumber daya, Anda mungkin secara tidak sengaja mencabut izin untuk AWS Glue pengguna yang ada di akun Anda dan menyebabkan gangguan yang tidak terduga. Coba contoh-contoh ini hanya di akun pengembangan atau pengujian, dan pastikan bahwa mereka tidak merusak alur kerja yang ada sebelum Anda melakukan perubahan.

Topik

- Pertimbangan untuk menggunakan kebijakan berbasis sumber daya dengan Glue AWS
- Menggunakan kebijakan sumber daya untuk mengontrol akses di akun yang sama

Pertimbangan untuk menggunakan kebijakan berbasis sumber daya dengan Glue AWS

> **Note**
>
> Kebijakan IAM dan kebijakan AWS Glue sumber daya membutuhkan beberapa detik untuk disebarkan. Setelah Anda melampirkan sebuah kebijakan baru, Anda mungkin memperhatikan bahwa kebijakan lama masih berlaku sampai kebijakan baru telah disebarkan melalui sistem.

Anda menggunakan sebuah dokumen kebijakan yang ditulis dalam format JSON untuk membuat atau memodifikasi sebuah kebijakan sumber daya. Sintaks kebijakan sama dengan kebijakan IAM berbasis identitas (lihat referensi kebijakan IAM JSON), dengan pengecualian berikut:

- Sebuah `"Principal"` atau blok `"NotPrincipal"` untuk setiap pernyataan kebijakan.
- `"Principal"`Atau `"NotPrincipal"` harus mengidentifikasi prinsip-prinsip yang ada yang valid. Pola wildcard (seperti `arn:aws:iam::`*`account-id`*`:user/*`) tidak diperbolehkan.
- `"Resource"`Blok dalam kebijakan mengharuskan semua ARN resource untuk mencocokkan sintaks ekspresi reguler berikut (di mana yang pertama `%s` adalah *`region`*, dan yang kedua `%s` adalah *`account-id`*):

```
*arn:aws:glue:%s:%s:(\*|[a-zA-Z\*]+\/?.*)
```

Misalnya, `arn:aws:glue:us-west-2:`*`account-id`*`:*` dan `arn:aws:glue:us-west-2:`*`account-id`*`:database/default` keduanya diperbolehkan, namun `*` tidak diperbolehkan.

- Tidak seperti kebijakan berbasis identitas, kebijakan AWS Glue sumber daya hanya boleh berisi Nama Sumber Daya Amazon (ARN) sumber daya yang termasuk dalam katalog tempat kebijakan dilampirkan. ARN tersebut selalu dimulai dengan `arn:aws:glue:`.
- Kebijakan tidak dapat menyebabkan identitas yang membuatnya terkunci dari pembuatan atau modifikasi kebijakan lebih lanjut.
- Dokumen JSON sumber daya kebijakan tidak dapat melebihi ukuran 10 KB.

Menggunakan kebijakan sumber daya untuk mengontrol akses di akun yang sama

Dalam contoh ini, pengguna admin di akun A menciptakan sebuah kebijakan sumber daya yang memberikan pengguna IAM `Alice` di Akun A akses penuh ke katalog. Alice tidak memiliki kebijakan IAM terlampir.

Untuk melakukan ini, pengguna admin menjalankan AWS CLI perintah berikut.

```
# Run as admin of Account A
$ aws glue put-resource-policy --profile administrator-name --region us-west-2 --
policy-in-json '{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": [
          "arn:aws:iam::account-A-id:user/Alice"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:*"
      ],
      "Resource": [
        "arn:aws:glue:us-west-2:account-A-id:*"
      ]
    }
  ]
}'
```

Alih-alih memasukkan dokumen kebijakan JSON sebagai bagian dari perintah AWS CLI, Anda dapat menyimpan dokumen kebijakan dalam file dan kemudian me-referensi path file dalam perintah AWS CLI tersebut, yang menggunakan prefiks `file://`. Berikut ini adalah contoh bagaimana Anda dapat melakukan hal itu.

```
$ echo '{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": {
        "AWS": [
          "arn:aws:iam::account-A-id:user/Alice"
        ]
      },
      "Effect": "Allow",
      "Action": [
        "glue:*"
      ],
      "Resource": [
        "arn:aws:glue:us-west-2:account-A-id:*"
      ]
    }
  ]
}' > /temp/policy.json

$ aws glue put-resource-policy --profile admin1 \
    --region us-west-2 --policy-in-json file:///temp/policy.json
```

Setelah kebijakan sumber daya ini disebarkan, Alice dapat mengakses semua AWS Glue sumber daya di Akun A, sebagai berikut.

```
# Run as user Alice
$ aws glue create-database --profile alice --region us-west-2 --database-input '{
    "Name": "new_database",
    "Description": "A new database created by Alice",
    "LocationUri": "s3://my-bucket"
}'

$ aws glue get-table --profile alice --region us-west-2 --database-name "default" --
table-name "tbl1"}
```

Menanggapi `get-table` panggilan Alice, AWS Glue layanan mengembalikan yang berikut ini.

```
{
  "Table": {
    "Name": "tbl1",
    "PartitionKeys": [],
    "StorageDescriptor": {
        ......
    },
    ......
  }
}
```

AWS kebijakan terkelola untuk AWS Glue

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS Kebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pembaruan akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan AWS terkelola saat baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat kebijakan yang dikelola AWS dalam Panduan Pengguna IAM.

AWS kebijakan terkelola (standar) untuk AWS Glue

AWS mengatasi banyak kasus penggunaan umum dengan menyediakan kebijakan IAM mandiri yang dibuat dan dikelola oleh. AWS Kebijakan AWS terkelola ini memberikan izin yang diperlukan untuk kasus penggunaan umum sehingga Anda dapat menghindari keharusan menyelidiki izin apa yang diperlukan. Untuk informasi selengkapnya, lihat kebijakan yang dikelola AWS dalam Panduan Pengguna IAM.

Kebijakan AWS terkelola berikut, yang dapat Anda lampirkan ke identitas di akun Anda, khusus untuk AWS Glue dan dikelompokkan berdasarkan skenario kasus penggunaan:

- AWSGlueConsoleFullAccessMemberikan akses penuh ke AWS Glue sumber daya ketika identitas yang dilampirkan kebijakan menggunakan. AWS Management Console Jika Anda mengikuti konvensi penamaan untuk sumber daya yang ditentukan dalam kebijakan ini, maka pengguna memiliki kemampuan konsol penuh. Kebijakan ini biasanya dilampirkan ke pengguna AWS Glue konsol.
- AWSGlueServiceRole— Memberikan akses ke sumber daya yang diperlukan berbagai AWS Glue proses untuk dijalankan atas nama Anda. Sumber daya ini termasukAWS Glue, Amazon S3, IAM, CloudWatch Log, dan Amazon EC2. Jika Anda mengikuti konvensi penamaan untuk sumber daya yang ditentukan dalam kebijakan ini, AWS Glue proses memiliki izin yang diperlukan. Kebijakan ini biasanya dilampirkan pada peran yang ditentukan saat menentukan crawler, tugas, dan titik akhir pengembangan.
- AwsGlueSessionUserRestrictedServiceRole— Menyediakan akses penuh ke semua AWS Glue sumber daya kecuali untuk sesi. Hal ini memungkinkan pengguna untuk membuat dan menggunakan hanya sesi interaktif yang terkait dengan pengguna. Kebijakan ini mencakup izin lain yang diperlukan AWS Glue untuk mengelola AWS Glue sumber daya di AWS layanan lain. Kebijakan ini juga memungkinkan penambahan tag ke AWS Glue sumber daya di AWS layanan lain.

  > **Note**
  >
  > Untuk mencapai manfaat keamanan penuh, jangan berikan kebijakan ini kepada pengguna yang ditugaskan `AWSGlueServiceRoleAWSGlueConsoleFullAccess`, atau `AWSGlueConsoleSageMakerNotebookFullAccess` kebijakan.

- AwsGlueSessionUserRestrictedPolicy— Menyediakan akses untuk membuat sesi AWS Glue interaktif menggunakan operasi `CreateSession` API hanya jika kunci tag “pemilik” dan nilai yang cocok dengan ID AWS pengguna penerima tugas disediakan. Kebijakan identitas ini dilampirkan ke pengguna IAM yang memanggil operasi `CreateSession` API. Kebijakan ini juga mengizinkan penerima tugas untuk berinteraksi dengan sumber daya sesi AWS Glue interaktif yang dibuat dengan tag “pemilik” dan nilai yang cocok dengan ID pengguna mereka. AWS Kebijakan ini menolak izin untuk mengubah atau menghapus tag “pemilik” dari sumber daya AWS Glue sesi setelah sesi dibuat.

> **Note**
>
> Untuk mencapai manfaat keamanan penuh, jangan berikan kebijakan ini kepada pengguna yang ditugaskan `AWSGlueServiceRoleAWSGlueConsoleFullAccess`, atau `AWSGlueConsoleSageMakerNotebookFullAccess` kebijakan.

- AwsGlueSessionUserRestrictedNotebookServiceRole— Menyediakan akses yang cukup ke sesi AWS Glue Studio notebook untuk berinteraksi dengan sumber daya sesi AWS Glue interaktif tertentu. Ini adalah sumber daya yang dibuat dengan nilai tag “pemilik” yang cocok dengan ID AWS pengguna prinsipal (pengguna atau peran IAM) yang membuat buku catatan. Untuk informasi selengkapnya tentang tag ini, lihat bagan Nilai kunci utama di Panduan Pengguna IAM.

  Kebijakan peran layanan ini dilampirkan ke peran yang ditentukan dengan perintah ajaib di dalam buku catatan atau diteruskan sebagai peran ke operasi `CreateSession` API. Kebijakan ini juga mengizinkan prinsipal untuk membuat sesi AWS Glue interaktif dari antarmuka AWS Glue Studio notebook hanya jika kunci tag “pemilik” dan nilai cocok dengan ID AWS pengguna prinsipal. Kebijakan ini menolak izin untuk mengubah atau menghapus tag “pemilik” dari sumber daya AWS Glue sesi setelah sesi dibuat. Kebijakan ini juga mencakup izin untuk menulis dan membaca dari bucket Amazon S3, CloudWatch menulis log, serta membuat serta menghapus tag untuk sumber daya Amazon EC2 yang digunakan oleh. AWS Glue

  > **Note**
  >
  > Untuk mencapai manfaat keamanan penuh, jangan berikan kebijakan ini untuk peran yang ditugaskan`AWSGlueServiceRole`,`AWSGlueConsoleFullAccess`, atau `AWSGlueConsoleSageMakerNotebookFullAccess` kebijakan.

- AwsGlueSessionUserRestrictedNotebookPolicy— Menyediakan akses untuk membuat sesi AWS Glue interaktif dari antarmuka AWS Glue Studio notebook hanya jika ada kunci tag “pemilik” dan nilai yang cocok dengan ID AWS penggunaDari prinsipal (pengguna atau peran IAM) yang membuat buku catatan. Untuk informasi selengkapnya tentang tag ini, lihat bagan Nilai kunci utama di Panduan Pengguna IAM.

  Kebijakan ini dilampirkan pada prinsipal (pengguna atau peran IAM) yang membuat sesi dari antarmuka AWS Glue Studio notebook. Kebijakan ini juga memungkinkan akses yang cukup ke AWS Glue Studio buku catatan untuk berinteraksi dengan sumber daya sesi AWS Glue interaktif tertentu. Ini adalah sumber daya yang dibuat dengan nilai tag “pemilik” yang cocok dengan ID AWS

pengguna prinsipal. Kebijakan ini menolak izin untuk mengubah atau menghapus tag “pemilik” dari sumber daya AWS Glue sesi setelah sesi dibuat.

- AWSGlueServiceNotebookRole— Memberikan akses ke AWS Glue sesi yang dimulai di AWS Glue Studio buku catatan. Kebijakan ini memungkinkan daftar dan mendapatkan informasi sesi untuk semua sesi, tetapi hanya mengizinkan pengguna untuk membuat dan menggunakan sesi yang ditandai dengan ID AWS pengguna mereka. Kebijakan ini menolak izin untuk mengubah atau menghapus tag “pemilik” dari sumber daya AWS Glue sesi yang ditandai dengan ID mereka AWS .

  Tetapkan kebijakan ini kepada AWS pengguna yang membuat lowongan menggunakan antarmuka notebook diAWS Glue Studio.
- AWSGlueConsoleSageMakerNotebookFullAccess— Memberikan akses penuh ke AWS Glue dan SageMaker sumber daya ketika identitas yang dilampirkan kebijakan menggunakan. AWS Management Console Jika Anda mengikuti konvensi penamaan untuk sumber daya yang ditentukan dalam kebijakan ini, maka pengguna memiliki kemampuan konsol penuh. Kebijakan ini biasanya dilampirkan ke pengguna AWS Glue konsol yang mengelola SageMaker buku catatan.
- AWSGlueSchemaRegistryFullAccessMemberikan akses penuh ke sumber AWS Glue Schema Registry ketika identitas yang dilampirkan kebijakan menggunakan atau. AWS Management Console AWS CLI Jika Anda mengikuti konvensi penamaan untuk sumber daya yang ditentukan dalam kebijakan ini, maka pengguna memiliki kemampuan konsol penuh. Kebijakan ini biasanya dilampirkan ke pengguna AWS Glue konsol atau AWS CLI yang mengelola Registri AWS Glue Skema.
- AWSGlueSchemaRegistryReadonlyAccess— Memberikan akses hanya-baca ke sumber daya Registri AWS Glue Skema ketika identitas yang dilampirkan kebijakan menggunakan atau. AWS Management Console AWS CLI Jika Anda mengikuti konvensi penamaan untuk sumber daya yang ditentukan dalam kebijakan ini, maka pengguna memiliki kemampuan konsol penuh. Kebijakan ini biasanya dilampirkan ke pengguna AWS Glue konsol atau AWS CLI yang menggunakan Registri AWS Glue Skema.

> **Note**
>
> Anda dapat meninjau kebijakan-kebijakan izin ini dengan masuk ke konsol IAM dan mencari kebijakan-kebijakan tertentu di sana.

Anda juga dapat membuat kebijakan IAM khusus Anda sendiri untuk memberikan izin untuk tindakan dan sumber daya Glue AWS . Anda dapat melampirkan kebijakan-kebijakan kustom ini ke pengguna IAM atau grup yang memerlukan izin-izin tersebut.

AWS Glue update ke kebijakan AWS terkelola

Lihat detail tentang pembaruan kebijakan AWS AWS terkelola untuk Glue sejak layanan ini mulai melacak perubahan ini. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan umpan RSS di halaman riwayat AWS Glue Document.

Perubahan	Deskripsi	Tanggal
AwsGlueSessionUserRestrictedPolicy — Pembaruan kecil untuk kebijakan yang ada.	Tambahkan `glue:StartCompletion` dan `glue:GetCompletion` ke kebijakan. Diperlukan untuk integrasi data Amazon Q di AWS Glue.	April, 30, 2024
AwsGlueSessionUserRestrictedNotebookServiceRole — Pembaruan kecil untuk kebijakan yang ada.	Tambahkan `glue:StartCompletion` dan `glue:GetCompletion` ke kebijakan. Diperlukan untuk integrasi data Amazon Q di AWS Glue.	April, 30, 2024
AwsGlueSessionUserRestrictedServiceRole — Pembaruan kecil untuk kebijakan yang ada.	Tambahkan `glue:StartCompletion` dan `glue:GetCompletion` ke kebijakan. Diperlukan untuk integrasi data Amazon Q di AWS Glue.	April, 30, 2024
AWSGlueServiceNotebookRole — Pembaruan kecil untuk kebijakan yang ada.	Tambahkan `glue:StartCompletion` dan `glue:GetCompletion` ke kebijakan. Diperlukan untuk	Jan 30, 2024

Perubahan	Deskripsi	Tanggal
	integrasi data Amazon Q di AWS Glue.	
AwsGlueSessionUserRestrictedNotebookPolicy — Pembaruan kecil untuk kebijakan yang ada.	Tambahkan `glue:StartCompletion` dan `glue:GetCompletion` ke kebijakan. Diperlukan untuk integrasi data Amazon Q di AWS Glue.	Nov 29, 2023
AWSGlueServiceNotebookRole — Pembaruan kecil untuk kebijakan yang ada.	Tambahkan `codewhisperer:GenerateRecommendations` ke kebijakan. Diperlukan untuk fitur baru di mana AWS Glue menghasilkan CodeWhisperer rekomendasi.	Okt 9, 2023
AWSGlueServiceRole — Pembaruan kecil untuk kebijakan yang ada.	Kencangkan ruang lingkup CloudWatch izin untuk lebih mencerminkan logging AWS Glue.	Agustus 4, 2023
AWSGlueConsoleFullAccess — Pembaruan kecil untuk kebijakan yang ada.	Tambahkan Daftar `databrew` resep dan Jelaskan izin ke kebijakan. Diperlukan untuk menyediakan akses administratif penuh untuk fitur-fitur baru di mana AWS Glue dapat mengakses resep.	9 Mei 2023

Perubahan	Deskripsi	Tanggal
AWSGlueConsoleFullAccess — Pembaruan kecil untuk kebijakan yang ada.	Tambahkan `cloudformation:ListStacks` ke kebijakan. Mempertahankan kemampuan yang ada setelah perubahan persyaratan AWS CloudFormation otorisasi.	Maret 28, 2023
Kebijakan terkelola baru ditambahkan untuk fitur sesi interaktif: • AwsGlueSessionUserRestrictedServiceRole • AwsGlueSessionUserRestrictedPolicy • AwsGlueSessionUserRestrictedNotebookServiceRole • AwsGlueSessionUserRestrictedNotebookPolicy	Kebijakan ini dirancang untuk memberikan keamanan tambahan untuk sesi interaktif dan notebook diAWS Glue Studio. Kebijakan membatasi akses ke operasi `CreateSession` API sehingga hanya pemilik yang memiliki akses.	30 November 2021

Perubahan	Deskripsi	Tanggal
AWSGlueConsoleSageMakerNotebookFullAccess — Perbarui ke kebijakan yang ada.	Menghapus sumber daya ARN (`arn:aws:s3:::aws-glue-*/*`) redundan untuk tindakan yang memberikan izin baca/tulis di bucket Amazon S3 yang digunakan untuk menyimpan skrip dan file sementaraAWS Glue. Perbaikan masalah sintaksis dengan mengubah `"StringEquals"` menjadi `"ForAnyValue:StringLike"` , dan memindahkan baris `"Effect": "Allow"` sehingga berada di depan baris `"Action":` saat urutan mereka tidak sebagaimana mestinya.	15 Juli 2021
AWSGlueConsoleFullAccess — Perbarui ke kebijakan yang ada.	Menghapus sumber daya ARN (`arn:aws:s3:::aws-glue-*/*`) redundan untuk tindakan yang memberikan izin baca/tulis di bucket Amazon S3 yang digunakan untuk menyimpan skrip dan file sementaraAWS Glue.	15 Juli 2021
AWS Gluemulai melacak perubahan.	AWS Gluemulai melacak perubahan untuk kebijakan yang AWS dikelola.	10 Juni 2021

Menentukan ARN AWS Glue sumber daya

DiAWS Glue, Anda dapat mengontrol akses ke sumber daya menggunakan kebijakan AWS Identity and Access Management (IAM). Dalam kebijakan, Anda menggunakan Amazon Resource Name (ARN) untuk mengidentifikasi sumber daya yang mengikuti kebijakan tersebut. Tidak semua sumber daya dalam AWS Glue mendukung ARN.

Topik

ARN Katalog Data

Sumber daya Katalog Data memiliki struktur hirarkis, dengan `catalog` sebagai akar.

```
arn:aws:glue:region:account-id:catalog
```

Setiap akun AWS memiliki satu Katalog Data tunggal di Wilayah AWS dengan ID akun 12 digit sebagai ID katalog. Sumber daya ini memiliki ARN unik yang dikaitkan dengan mereka, seperti yang ditunjukkan di tabel berikut.

Jenis sumber daya	Format ARN
Katalog	`arn:aws:glue:` *`region`*`:`*`account-id`* `:catalog` Misalnya: `arn:aws:glue:us-east-1:123456789012:catalog`
Basis Data	`arn:aws:glue:` *`region`*`:`*`account-id`* `:database/` *`database name`* Misalnya: `arn:aws:glue:us-east-1:123456789012:database/db1`

Jenis sumber daya	Format ARN
Tabel	`arn:aws:glue:` *region*`:`*account-id* `:table/`*database name*`/`*table name* Misalnya: `arn:aws:glue:us-east-1:123456789012:table/db1/tbl1`
Fungsi yang ditetapkan pengguna	`arn:aws:glue:` *region*`:`*account-id* `:userDefinedFunction/` *database name*`/`*user-defined function name* Misalnya: `arn:aws:glue:us-east-1:123456789012:userDefinedFunction/db1/func1`
Koneksi	`arn:aws:glue:` *region*`:`*account-id* `:connection/` *connection name* Misalnya: `arn:aws:glue:us-east-1:123456789012:connection/connection1`
Sesi Interaktif	`arn:aws:glue:` *region*`:`*account-id* `:session/` *interactive session id* Misalnya: `arn:aws:glue:us-east-1:123456789012:session/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111`

Untuk mengaktifkan kontrol akses detail, Anda dapat menggunakan ARN ini dalam kebijakan IAM dan kebijakan sumber daya untuk memberikan dan menolak akses ke sumber daya tertentu. Wildcard diperbolehkan dalam kebijakan tersebut. Misalnya, ARN berikut cocok dengan semua tabel dalam basis data `default`.

```
arn:aws:glue:us-east-1:123456789012:table/default/*
```

> **Important**
>
> Semua operasi yang dilakukan pada sumber daya Katalog Data memerlukan izin pada sumber daya dan semua pendahulu dari sumber daya tersebut. Misalnya, untuk membuat sebuah partisi untuk tabel memerlukan izin pada tabel, basis data, dan katalog di mana tabel

berada. Contoh berikut menunjukkan izin yang diperlukan untuk membuat partisi pada tabel `PrivateTable` di basis data `PrivateDatabase` dalam Katalog Data.

```
{
   "Sid": "GrantCreatePartitions",
   "Effect": "Allow",
   "Action": [
       "glue:BatchCreatePartitions"
   ],
   "Resource": [
       "arn:aws:glue:us-east-1:123456789012:table/PrivateDatabase/PrivateTable",
       "arn:aws:glue:us-east-1:123456789012:database/PrivateDatabase",
       "arn:aws:glue:us-east-1:123456789012:catalog"
   ]
}
```

Selain izin pada sumber daya dan semua pendahulunya, semua operasi penghapusan memerlukan izin pada semua anak dari sumber daya tersebut. Misalnya, menghapus sebuah basis data memerlukan izin pada semua tabel dan fungsi yang ditetapkan pengguna dalam basis data tersebut, selain basis data dan katalog di mana basis data berada. Contoh berikut menunjukkan izin yang diperlukan untuk menghapus basis data `PrivateDatabase` dalam Katalog Data.

```
{
   "Sid": "GrantDeleteDatabase",
   "Effect": "Allow",
   "Action": [
       "glue:DeleteDatabase"
   ],
   "Resource": [
       "arn:aws:glue:us-east-1:123456789012:table/PrivateDatabase/*",
       "arn:aws:glue:us-east-1:123456789012:userDefinedFunction/PrivateDatabase/
*",
       "arn:aws:glue:us-east-1:123456789012:database/PrivateDatabase",
       "arn:aws:glue:us-east-1:123456789012:catalog"
   ]
}
```

Singkatnya, tindakan pada sumber daya Katalog Data mengikuti aturan izin berikut ini:

- Tindakan pada katalog hanya memerlukan izin pada katalog saja.

- Tindakan pada sebuah basis data memerlukan izin pada basis data dan katalog.
- Menghapus tindakan pada sebuah basis data memerlukan izin pada basis data dan katalog ditambah pada semua tabel dan fungsi yang ditetapkan pengguna dalam basis data tersebut.
- Tindakan pada sebuah tabel, partisi, atau versi tabel memerlukan izin pada tabel, basis data, dan katalog.
- Tindakan pada fungsi yang ditetapkan pengguna memerlukan izin pada fungsi yang ditetapkan pengguna, basis data, dan katalog.
- Tindakan pada sebuah koneksi memerlukan izin pada koneksi dan katalog.

ARN untuk objek non-katalog di AWS Glue

Beberapa AWS Glue sumber daya memungkinkan izin tingkat sumber daya untuk mengontrol akses menggunakan ARN. Anda dapat menggunakan ARN ini dalam kebijakan IAM Anda untuk mengaktifkan kontrol akses yang detail. Tabel berikut mencantumkan sumber daya yang dapat berisi ARN sumber daya.

Jenis sumber daya	Format ARN
Crawler	`arn:aws:glue:` *`region`*`:`*`account-id`* `:crawler/` *`crawler-name`* Misalnya: `arn:aws:glue:us-east-1:123456789012:crawler/mycrawler`
Tugas	`arn:aws:glue:` *`region`*`:`*`account-id`* `:job/`*`job-name`* Misalnya: `arn:aws:glue:us-east-1:123456789012:job/testjob`
Pemicu	`arn:aws:glue:` *`region`*`:`*`account-id`* `:trigger/` *`trigger-name`* Misalnya: `arn:aws:glue:us-east-1:123456789012:trigger/sampletrigger`

Jenis sumber daya	Format ARN
Titik akhir pengembangan	`arn:aws:glue:` *`region`*`:`*`account-id`* `:devEndpoint/` *`development-endpoint-name`* Misalnya: `arn:aws:glue:us-east-1:123456789012:devEndpoint/temporarydevendpoint`
Transformasi Machine Learning	`arn:aws:glue:` *`region`*`:`*`account-id`* `:mlTransform/` *`transform-id`* Misalnya: `arn:aws:glue:us-east-1:123456789012:mlTransform/tfm-1234567890`

Kontrol akses untuk AWS Glue operasi API tunggal non-katalog

AWS GlueOperasi API tunggal non-katalog bertindak pada satu item (titik akhir pengembangan). Contohnya adalah `GetDevEndpoint`, `CreateUpdateDevEndpoint`, dan `UpdateDevEndpoint`. Untuk operasi ini, sebuah kebijakan harus menempatkan nama API di blok `"action"` dan ARN sumber daya di blok `"resource"`.

Misalkan Anda ingin mengizinkan pengguna untuk memanggil operasi `GetDevEndpoint`. Kebijakan berikut memberikan izin minimum yang diperlukan untuk titik akhir yang bernama `myDevEndpoint-1`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "MinimumPermissions",
            "Effect": "Allow",
            "Action": "glue:GetDevEndpoint",
            "Resource": "arn:aws:glue:us-east-1:123456789012:devEndpoint/
myDevEndpoint-1"
        }
    ]
}
```

Kebijakan berikut memungkinkan akses `UpdateDevEndpoint` ke sumber daya yang cocok dengan `myDevEndpoint-` yang memiliki sebuah wildcard (*).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PermissionWithWildcard",
            "Effect": "Allow",
            "Action": "glue:UpdateDevEndpoint",
            "Resource": "arn:aws:glue:us-east-1:123456789012:devEndpoint/myDevEndpoint-
*"
        }
    ]
}
```

Anda dapat menggabungkan dua kebijakan seperti dalam contoh berikut ini. Anda mungkin melihat `EntityNotFoundException` untuk setiap titik akhir pengembangan yang namanya dimulai dengan A. Namun, kesalahan akses ditolak akan dikembalikan ketika Anda mencoba untuk mengakses titik akhir pengembangan lainnya.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CombinedPermissions",
            "Effect": "Allow",
            "Action": [
                "glue:UpdateDevEndpoint",
                "glue:GetDevEndpoint"
            ],
            "Resource": "arn:aws:glue:us-east-1:123456789012:devEndpoint/A*"
        }
    ]
}
```

Kontrol akses untuk operasi API AWS Glue non-katalog yang mengambil beberapa item

Beberapa operasi AWS Glue API mengambil beberapa item (seperti beberapa titik akhir pengembangan); misalnya,. `GetDevEndpoints` Untuk operasi ini, Anda dapat menentukan hanya wildcard (*) sumber daya, dan bukan ARN tertentu.

Sebagai contoh, untuk memasukkan `GetDevEndpoints` dalam kebijakan, sumber daya harus terlingkup ke wildcard (*). Operasi tunggal (`GetDevEndpoint`, `CreateDevEndpoint`, dan `DeleteDevendpoint`) juga terlingkup untuk semua (*) sumber daya dalam contoh.

```
{
            "Sid": "PluralAPIIncluded",
            "Effect": "Allow",
            "Action": [
                "glue:GetDevEndpoints",
                "glue:GetDevEndpoint",
                "glue:CreateDevEndpoint",
                "glue:UpdateDevEndpoint"
            ],
            "Resource": [
                "*"
            ]
}
```

Kontrol akses untuk operasi BatchGet API AWS Glue non-katalog

Beberapa operasi AWS Glue API mengambil beberapa item (seperti beberapa titik akhir pengembangan); misalnya,. `BatchGetDevEndpoints` Untuk operasi ini, Anda dapat menentukan sebuah ARN untuk membatasi lingkup sumber daya yang dapat diakses.

Misalnya, untuk mengizinkan akses ke titik akhir pengembangan tertentu, sertakan `BatchGetDevEndpoints` dalam kebijakan dengan ARN sumber dayanya.

```
{
            "Sid": "BatchGetAPIIncluded",
            "Effect": "Allow",
            "Action": [
                "glue:BatchGetDevEndpoints"
            ],
            "Resource": [
```

```
                "arn:aws:glue:us-east-1:123456789012:devEndpoint/de1"
            ]
}
```

Dengan kebijakan ini, Anda dapat berhasil mengakses titik akhir pengembangan bernama `de1`. Namun, jika Anda mencoba untuk mengakses titik akhir pengembangan bernama `de2`, maka kesalahan akan dikembalikan.

```
An error occurred (AccessDeniedException) when calling the BatchGetDevEndpoints
 operation: No access to any requested resource.
```

> **Important**
>
> Untuk pendekatan alternatif untuk menyiapkan kebijakan IAM, seperti menggunakan operasi API `List` dan `BatchGet`, lihat Contoh kebijakan berbasis identitas untuk Glue AWS.

Memberikan akses lintas akun

Memberikan akses ke sumber daya Katalog Data di seluruh akun memungkinkan tugas extract, transform, and load (ETL) untuk meng-kueri dan menggabungkan data dari akun yang berbeda.

Topik

- Metode untuk memberikan akses lintas akun di AWS Glue
- Menambahkan atau memperbarui kebijakan sumber daya Katalog Data
- Melakukan panggilan API lintas akun
- Melakukan panggilan ETL lintas akun
- Pencatatan lintas akun CloudTrail
- Kepemilikan dan penagihan sumber daya lintas akun
- Batasan akses lintas akun

Metode untuk memberikan akses lintas akun di AWS Glue

Anda dapat memberikan akses ke data Anda ke AWS akun eksternal dengan menggunakan AWS Glue metode atau dengan menggunakan hibah AWS Lake Formation lintas akun. AWS GlueMetode

menggunakan kebijakan AWS Identity and Access Management (IAM) untuk mencapai kontrol akses berbutir halus. Lake Formation menggunakan model izin `GRANT/REVOKE` yang lebih sederhana yang mirip dengan perintah `GRANT/REVOKE` dalam sistem basis data relasional.

Bagian ini menjelaskan penggunaan AWS Glue metode. Untuk informasi tentang menggunakan pemberian Lake Formation lintas kaun, lihat Memberikan Izin Lake Formation di Panduan Developer AWS Lake Formation .

Ada dua AWS Glue metode untuk memberikan akses lintas akun ke sumber daya:

- Menggunakan kebijakan sumber daya Katalog Data
- Menggunakan sebuah IAM role

Memberikan akses lintas akun menggunakan kebijakan sumber daya

Berikut ini adalah langkah-langkah umum untuk memberikan akses lintas akun menggunakan sebuah kebijakan sumber daya Katalog Data:

1. Administrator (atau identitas yang diotorisasi lainnya) di akun A melampirkan sebuah kebijakan sumber daya untuk Katalog Data di Akun A. Kebijakan ini memberikan kepada Akun B izin lintas akun tertentu untuk melakukan operasi pada sumber daya di katalog Akun A.
2. Administrator di Akun B melampirkan kebijakan IAM ke identitas IAM di Akun B yang mendelegasikan izin yang diterima dari Akun A.

   Identitas di Akun B sekarang memiliki akses ke sumber daya yang ditentukan di Akun A.

   Identitas memerlukan izin dari pemilik sumber daya (Akun A) dan akun induknya (Akun B) untuk dapat mengakses sumber daya.

Memberikan akses lintas akun menggunakan sebuah IAM role

Berikut ini adalah langkah-langkah umum untuk memberikan akses lintas akun menggunakan sebuah IAM role:

1. Administrator (atau identitas yang mendapat otorisasi lainnya) di akun yang memiliki sumber daya (Akun A) menciptakan sebuah IAM role.
2. Administrator di Akun A melampirkan sebuah kebijakan pada peran yang memberikan izin lintas akun untuk mengakses ke sumber daya yang dimaksud.

3. Administrator di Akun A melampirkan sebuah kebijakan kepercayaan pada peran yang mengidentifikasi identitas IAM di akun yang berbeda (Akun B) sebagai prinsipal utama yang dapat mengambil peran tersebut.

   Prinsipal dalam kebijakan kepercayaan juga dapat menjadi kepala AWS layanan jika Anda ingin memberikan izin AWS layanan untuk mengambil peran tersebut.
4. Administrator di Akun B sekarang mendelegasikan izin untuk satu atau beberapa identitas IAM di Akun B sehingga mereka dapat mengambil peran itu. Dengan demikian, itu akan memberikan kepada identitas-identitas yang ada di Akun B akses ke sumber daya yang ada di Akun A.

Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mendelegasikan izin, lihat Manajemen Akses dalam Panduan Pengguna IAM. Untuk informasi lebih lanjut tentang pengguna, kelompok, peran, dan izin, lihat Identitas (Pengguna, Grup, dan Peran) dalam Panduan Pengguna IAM.

Untuk perbandingan kedua pendekatan ini, lihat Bagaimana peran IAM berbeda dari kebijakan berbasis sumber daya di Panduan Pengguna IAM. AWS Gluemendukung kedua opsi, dengan batasan bahwa kebijakan sumber daya hanya dapat memberikan akses ke sumber daya Katalog Data.

Misalnya, untuk memberikan `Dev` peran di Akun B akses ke database `db1` di Akun A, lampirkan kebijakan sumber daya berikut ke katalog di Akun A.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:GetDatabase"
      ],
      "Principal": {"AWS": [
        "arn:aws:iam::account-B-id:role/Dev"
      ]},
      "Resource": [
        "arn:aws:glue:us-east-1:account-A-id:catalog",
        "arn:aws:glue:us-east-1:account-A-id:database/db1"
      ]
    }
  ]
```

```
}
```

Selain itu, Akun B harus melampirkan kebijakan IAM berikut ke `Dev` peran sebelum benar-benar mendapatkan akses ke db1 Akun A.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "glue:GetDatabase"
      ],
      "Resource": [
        "arn:aws:glue:us-east-1:account-A-id:catalog",
        "arn:aws:glue:us-east-1:account-A-id:database/db1"
      ]
    }
  ]
}
```

Menambahkan atau memperbarui kebijakan sumber daya Katalog Data

Anda dapat menambahkan atau memperbarui kebijakan sumber daya Katalog AWS Glue Data menggunakan konsol, API, atau AWS Command Line Interface (AWS CLI).

> **Important**
>
> Jika Anda telah melakukan pemberian izin lintas akun dari akun Anda dengan AWS Lake Formation, untuk menambahkan atau memperbarui kebijakan sumber daya Katalog Data akan memerlukan sebuah langkah tambahan. Untuk informasi selengkapnya, lihat Mengelola izin lintas akun menggunakan keduanya AWS Glue dan Lake Formation di Panduan AWS Lake Formation Pengembang.
> Untuk menentukan apakah hibah lintas akun Lake Formation ada, gunakan operasi `glue:GetResourcePolicies` API atau. AWS CLI Jika `glue:GetResourcePolicies` mengembalikan kebijakan apa pun selain kebijakan Katalog Data yang sudah ada, maka hibah Lake Formation ada. Untuk informasi selengkapnya, lihat Melihat semua hibah lintas akun menggunakan operasi GetResourcePolicies API di Panduan AWS Lake Formation Pengembang.

Untuk menambah atau memperbarui kebijakan sumber daya Katalog Data (konsol)

1. Buka konsol AWS Glue di https://console.aws.amazon.com/glue/.

   Masuk sebagai pengguna administratif AWS Identity and Access Management (IAM) yang memiliki `glue:PutResourcePolicy` izin.
2. Pada panel navigasi, silakan pilih Pengaturan.
3. Pada halaman Pengaturan katalog data, pada Izin, tempel kebijakan sumber daya ke area teks. Lalu, pilih Simpan.

   Jika konsol tersebut menampilkan pemberitahuan yang menyatakan bahwa izin dalam kebijakan tersebut merupakan tambahan pada izin yang diberikan menggunakan Lake Formation, pilih Lanjutkan.

Untuk menambah atau memperbarui kebijakan sumber daya Katalog Data (AWS CLI)

- Mengirimkan sebuah perintah `aws glue put-resource-policy`. Jika pemberian Lake Formation sudah ada, pastikan Anda menyertakan pilihan `--enable-hybrid` dengan nilai `'TRUE'`.

  Untuk contoh menggunakan perintah ini, lihat Contoh kebijakan berbasis sumber daya untuk Glue AWS.

Melakukan panggilan API lintas akun

Semua AWS Glue Data Catalog operasi memiliki `CatalogId` bidang. Jika izin yang diperlukan telah diberikan untuk mengaktifkan akses lintas akun, pemanggil dapat membuat panggilan API Katalog Data di seluruh akun. Pemanggil tersebut melakukan hal ini dengan memberikan ID akun AWS target di `CatalogId` sehingga dapat mengakses sumber daya di akun target tersebut.

Jika tidak ada `CatalogId` nilai yang diberikan, AWS Glue gunakan ID akun pemanggil sendiri secara default, dan panggilan tidak lintas akun.

Melakukan panggilan ETL lintas akun

Beberapa AWS Glue PySpark dan Scala API memiliki kolom ID katalog. Jika semua izin yang diperlukan telah diberikan untuk mengaktifkan akses lintas akun, tugas ETL dapat membuat PySpark dan Scala panggilan ke operasi API di seluruh akun dengan meneruskan ID AWS akun target di bidang ID katalog untuk mengakses sumber daya Katalog Data di akun target.

Jika tidak ada nilai ID katalog yang diberikan, AWS Glue gunakan ID akun pemanggil sendiri secara default, dan panggilan tidak lintas akun.

Untuk PySpark API yang mendukung`catalog_id`, lihatGlueContext kelas. Untuk API Scala yang mendukung `catalogId`, lihat AWS GlueAPI Scala GlueContext .

Contoh berikut menunjukkan izin yang diperlukan oleh penerima untuk menjalankan sebuah tugas ETL. Dalam contoh ini, *`grantee-account-id`*adalah klien yang menjalankan pekerjaan dan *`grantor-account-id`*merupakan pemilik sumber daya. `catalog-id` Contoh ini memberikan izin untuk semua sumber daya katalog di akun pemberi. Untuk membatasi lingkup sumber daya yang diberikan, Anda dapat memberikan ARN spesifik untuk katalog, basis data, tabel, dan koneksi.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "glue:GetConnection",
                "glue:GetDatabase",
                "glue:GetTable",
                "glue:GetPartition"
            ],
            "Principal": {"AWS": ["arn:aws:iam::grantee-account-id:root"]},
            "Resource": [
                "arn:aws:glue:us-east-1:grantor-account-id:*"
            ]
        }
    ]
}
```

> **Note**
>
> Jika sebuah tabel di akun pemberi mengarahkan ke lokasi Amazon S3 yang juga ada di akun pemberi, maka IAM role yang digunakan untuk menjalankan tugas ETL di akun penerima harus memiliki izin untuk mencantumkan dan mengambil objek dari akun pemberi.

Mengingat bahwa klien di Akun A sudah memiliki izin untuk membuat dan menjalankan tugas ETL, berikut ini adalah langkah-langkah dasar untuk mengatur tugas ETL untuk akses lintas akun:

1. Izinkan akses data lintas akun (lewati langkah ini jika akses lintas akun Amazon S3 sudah ditetapkan).
   a. Memperbarui kebijakan bucket Amazon S3 di Akun B untuk memungkinkan akses lintas-akun dari Akun A.
   b. Memperbarui kebijakan IAM di akun A untuk memungkinkan akses ke bucket yang ada di Akun B.
2. Izinkan akses Katalog Data lintas akun.
   a. Membuat atau memperbarui kebijakan sumber daya yang dilampirkan pada Katalog Data di Akun B untuk memungkinkan akses dari Akun A.
   b. Memperbarui kebijakan IAM di akun A untuk memungkinkan akses ke Katalog Data yang ada di Akun B.

Pencatatan lintas akun CloudTrail

Saat pekerjaan AWS Glue ekstrak, transformasi, dan muat (ETL) mengakses data dasar tabel Katalog Data yang dibagikan melalui hibah AWS Lake Formation lintas akun, ada perilaku pencatatan tambahan. AWS CloudTrail

Untuk tujuan diskusi ini, AWS akun yang membagikan tabel adalah akun pemilik, dan akun tempat tabel dibagikan adalah akun penerima. Ketika pekerjaan ETL di akun penerima mengakses data dalam tabel di akun pemilik, CloudTrail peristiwa akses data yang ditambahkan ke log untuk akun penerima akan disalin ke log akun pemilik. CloudTrail Hal ini agar akun pemilik tersebut dapat melacak akses data oleh berbagai akun penerima. Secara default, CloudTrail peristiwa tidak menyertakan pengidentifikasi utama yang dapat dibaca manusia (ARN utama). Administrator yang ada di akun penerima dapat memilih untuk menyertakan ARN utama dalam log.

Untuk informasi selengkapnya, lihat CloudTrailLogging lintas akun di Panduan AWS Lake Formation Pengembang.

Lihat Juga

- the section called “Pencatatan dan pemantauan”

Kepemilikan dan penagihan sumber daya lintas akun

Ketika pengguna dalam satu AWS akun (Akun A) membuat sumber daya baru seperti database di akun yang berbeda (Akun B), sumber daya tersebut kemudian dimiliki oleh Akun B, akun tempat pembuatannya. Administrator di Akun B secara otomatis mendapatkan izin penuh untuk mengakses sumber daya baru tersebut, termasuk untuk membaca, menulis, dan memberikan izin akses ke akun ketiga. Pengguna di Akun A dapat mengakses sumber daya yang baru mereka buat tersebut jika mereka memiliki izin yang sesuai yang diberikan oleh Akun B.

Biaya penyimpanan dan biaya lain yang secara langsung dikaitkan dengan sumber daya baru akan ditagih ke Akun B, pemilik sumber daya. Biaya permintaan dari pengguna yang membuat sumber daya ditagih ke akun peminta, yakni Akun A.

Untuk informasi selengkapnya tentang AWS Glue penagihan dan harga, lihat Cara Kerja AWS Harga.

Batasan akses lintas akun

AWS Glueakses lintas akun memiliki batasan sebagai berikut:

- Akses lintas akun ke tidak AWS Glue diizinkan jika Anda membuat database dan tabel menggunakan Amazon Athena atau Amazon Redshift Spectrum sebelum AWS Glue dukungan wilayah untuk dan akun pemilik sumber daya belum memigrasikan katalog data Amazon Athena ke. AWS Glue Anda dapat menemukan status migrasi saat ini menggunakan GetCatalogImportStatus (get_catalog_import_status). Untuk detail selengkapnya tentang cara memigrasikan katalog AWS Glue Athena, lihat Memutakhirkan ke Panduan Pengguna AWS Glue Data Catalog step-by-step Amazon Athena.
- Akses lintas akun didukung hanya untuk sumber daya Katalog Data, termasuk basis data, tabel, fungsi yang ditetapkan pengguna, dan koneksi.
- Akses lintas akun ke Katalog Data dari Athena mengharuskan Anda mendaftarkan katalog sebagai sumber daya Athena`DataCatalog`. Untuk petunjuk, lihat Mendaftarkan AWS Glue Data Catalog dari akun lain di Panduan Pengguna Amazon Athena.

Pemecahan masalah identitas dan akses AWS Glue

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan AWS Glue dan IAM.

Topik

- Saya tidak berwenang untuk melakukan tindakan di AWS Glue
- Saya tidak berwenang untuk melakukan iam: PassRole
- Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses sumber daya AWS Glue saya

Saya tidak berwenang untuk melakukan tindakan di AWS Glue

Jika Anda menerima pesan kesalahan bahwa Anda tidak memiliki otorisasi untuk melakukan tindakan, kebijakan Anda harus diperbarui agar Anda dapat melakukan tindakan tersebut.

Contoh kesalahan berikut terjadi ketika pengguna IAM `mateojackson` mencoba menggunakan konsol untuk melihat detail tentang suatu sumber daya *`my-example-widget`* rekaan, tetapi tidak memiliki izin `glue:`*`GetWidget`* rekaan.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 glue:GetWidget on resource: my-example-widget
```

Dalam hal ini, kebijakan untuk pengguna `mateojackson` harus diperbarui untuk mengizinkan akses ke sumber daya *`my-example-widget`* dengan menggunakan tindakan `glue:`*`GetWidget`*.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya tidak berwenang untuk melakukan iam: PassRole

Jika Anda menerima kesalahan bahwa Anda tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran ke AWS Glue.

Beberapa Layanan AWS memungkinkan Anda untuk meneruskan peran yang ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait layanan. Untuk melakukannya, Anda harus memiliki izin untuk meneruskan peran ke layanan.

Contoh kesalahan berikut terjadi ketika pengguna IAM bernama `marymajor` mencoba menggunakan konsol untuk melakukan tindakan di AWS Glue. Namun, tindakan tersebut memerlukan layanan untuk mendapatkan izin yang diberikan oleh peran layanan. Mary tidak memiliki izin untuk meneruskan peran tersebut pada layanan.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dalam kasus ini, kebijakan Mary harus diperbarui agar dia mendapatkan izin untuk melakukan tindakan `iam:PassRole` tersebut.

Jika Anda memerlukan bantuan, hubungi AWS administrator Anda. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses sumber daya AWS Glue saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis sumber daya atau daftar kontrol akses (ACL), Anda dapat menggunakan kebijakan tersebut untuk memberi orang akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa referensi berikut:

- Untuk mengetahui apakah AWS Glue mendukung fitur-fitur ini, lihatBagaimana AWS Glue bekerja dengan IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda di seluruh sumber daya Akun AWS yang Anda miliki, lihat Menyediakan akses ke pengguna IAM di pengguna lain Akun AWS yang Anda miliki di Panduan Pengguna IAM.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda kepada pihak ketiga Akun AWS, lihat Menyediakan akses yang Akun AWS dimiliki oleh pihak ketiga dalam Panduan Pengguna IAM.
- Untuk mempelajari cara memberikan akses melalui federasi identitas, lihat Menyediakan akses ke pengguna terautentikasi eksternal (federasi identitas) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara menggunakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Akses sumber daya lintas akun di IAM di Panduan Pengguna IAM.

Pencatatan dan pemantauan di AWS Glue

Anda dapat mengotomatisasi eksekusi dari tugas ETL (ekstrak, mengubah, dan memuat) Anda. AWS Glue menyediakan metrik untuk crawler dan tugas yang dapat Anda pantau. Setelah menyiapkan AWS Glue Data Catalog dengan metadata yang dibutuhkan, AWS Glue menyediakan statistik kondisi health lingkungan Anda. Anda dapat mengotomatisasi pengaktifan crawler dan tugas dengan jadwal

berbasis waktu berdasarkan cron. Anda juga dapat memicu tugas ketika pemicu berbasis peristiwa aktif.

AWS Glue terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau layanan AWS di AWS Glue. Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara terus menerus ke bucket Amazon Simple Storage Service (Amazon S3), Amazon Logs, dan CloudWatch Amazon Events. CloudWatch Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut.

Gunakan Amazon CloudWatch Events untuk mengotomatiskan AWS layanan Anda dan merespons secara otomatis peristiwa sistem seperti masalah ketersediaan aplikasi atau perubahan sumber daya. Acara dari AWS layanan dikirimkan ke CloudWatch Acara dalam waktu nyaris nyata. Anda dapat menulis aturan sederhana untuk menunjukkan kejadian mana yang sesuai kepentingan, dan tindakan otomatis apa yang diambil ketika suatu kejadian sesuai dengan suatu aturan.

Lihat juga

- Mengotomatisasi AWS Glue dengan Acara CloudWatch
- Pencatatan lintas akun CloudTrail

Aspek penting keamanan di cloud adalah logging. Anda harus mengonfigurasi logging dengan cara yang tidak menangkap rahasia dan materi rahasia sambil menangkap informasi yang diperlukan untuk men-debug dan mengamankan infastruktur cloud Anda. Pastikan untuk membiasakan diri dengan apa yang sedang dicatat.

Validasi kepatuhan untuk AWS Glue

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat Program AWS Kepatuhan Program AWS .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Panduan Memulai Cepat Keamanan dan Kepatuhan — Panduan penerapan ini membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar AWS yang berfokus pada keamanan dan kepatuhan.
- Arsitektur untuk Keamanan dan Kepatuhan HIPAA di Amazon Web Services — Whitepaper ini menjelaskan bagaimana perusahaan dapat menggunakan AWS untuk membuat aplikasi yang memenuhi syarat HIPAA.

> **Note**
>
> Tidak semua memenuhi Layanan AWS syarat HIPAA. Untuk informasi selengkapnya, lihat Referensi Layanan yang Memenuhi Syarat HIPAA.

- AWS Sumber Daya AWS — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- AWS Panduan Kepatuhan Pelanggan - Memahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan untuk kontrol keamanan di berbagai kerangka kerja (termasuk Institut Standar dan Teknologi Nasional (NIST), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi (ISO)).
- Mengevaluasi Sumber Daya dengan Aturan dalam Panduan AWS Config Pengembang — AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- AWS Security Hub— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS. Security Hub menggunakan kontrol keamanan untuk sumber daya AWS Anda serta untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat Referensi kontrol Security Hub.
- Amazon GuardDuty — Ini Layanan AWS mendeteksi potensi ancaman terhadap beban kerja Akun AWS, kontainer, dan data Anda dengan memantau lingkungan Anda untuk aktivitas yang mencurigakan dan berbahaya. GuardDuty dapat membantu Anda mengatasi berbagai persyaratan kepatuhan, seperti PCI DSS, dengan memenuhi persyaratan deteksi intrusi yang diamanatkan oleh kerangka kerja kepatuhan tertentu.
- AWS Audit ManagerIni Layanan AWS membantu Anda terus mengaudit AWS penggunaan Anda untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap peraturan dan standar industri.

Ketahanan di AWS Glue

Infrastruktur AWS global dibangun di sekitar AWS Wilayah dan Zona Ketersediaan. AWS Wilayah menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan. Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan failover di antara Zona Ketersediaan tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur biasa yang terdiri dari satu atau beberapa pusat data.

Untuk informasi selengkapnya tentang AWS Wilayah dan Availability Zone, lihat Infrastruktur AWS Global.

Untuk informasi selengkapnya tentang ketahanan AWS Glue pekerjaan, lihat Kesalahan: Perilaku failover antara VPC di. AWS Glue

Keamanan infrastruktur dalam AWS Glue

Sebagai suatu layanan terkelola, AWS Glue dilindungi oleh prosedur keamanan jaringan global AWS yang dijelaskan dalam laporan resmi Amazon Web Services: Gambaran Umum Proses Keamanan.

Anda menggunakan panggilan API yang dipublikasikan AWS untuk mengakses AWS Glue melalui jaringan. Klien harus mendukung Keamanan Lapisan Pengangkutan (TLS) 1.0 atau versi yang lebih baru. Kami merekomendasikan TLS 1.2 atau versi yang lebih baru. Klien juga harus mendukung suite cipher dengan perfect forward secrecy (PFS) seperti Ephemeral Diffie-Hellman (DHE) atau Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Sebagian besar sistem modern seperti Java 7 dan sistem yang lebih baru mendukung mode ini.

Selain itu, permintaan harus ditandatangani menggunakan access key ID dan secret access key yang terkait dengan principal IAM. Atau Anda bisa menggunakan AWS Security Token Service (AWS STS) untuk membuat kredensial keamanan sementara guna menandatangani permintaan.

Topik

- AWS Glue dan titik akhir VPC antarmuka (AWS PrivateLink)
- Amazon VPC Bersama

AWS Glue dan titik akhir VPC antarmuka (AWS PrivateLink)

Anda dapat membangun hubungan privat antara VPC Anda dan AWS Glue dengan membuat VPC endpoint antarmuka. Endpoint antarmuka didukung oleh AWS PrivateLink, teknologi yang memungkinkan Anda mengakses AWS Glue API secara pribadi tanpa gateway internet, perangkat NAT, koneksi VPN, atau koneksi Direct AWS Connect. Instans dalam VPC Anda tidak memerlukan alamat IP publik untuk berkomunikasi dengan API AWS Glue. Lalu lintas antara VPC Anda dan AWS Glue tidak meninggalkan jaringan Amazon.

Setiap titik akhir antarmuka diwakili oleh satu atau lebih Antarmuka Jaringan Elastis dalam subnet Anda.

Untuk informasi selengkapnya, lihat Antarmuka VPC endpoint (AWS PrivateLink) dalam Panduan Pengguna Amazon VPC.

Pertimbangan untuk VPC endpoint AWS Glue

Sebelum Anda menyiapkan VPC endpoint antarmuka untuk AWS Glue, pastikan bahwa Anda meninjau Properti titik akhir antarmuka dan pembatasan dalam Panduan Pengguna Amazon VPC.

AWS Glue mendukung panggilan ke semua tindakan API dari VPC Anda.

Buat VPC endpoint antarmuka untuk AWS Glue

Anda dapat membuat VPC endpoint untuk layanan AWS Glue menggunakan konsol Amazon VPC atau AWS Command Line Interface (AWS CLI). Untuk informasi lebih lanjut, lihat Membuat titik akhir antarmuka dalam Panduan Pengguna Amazon VPC.

Buat titik akhir VPC untuk AWS Glue menggunakan nama layanan berikut:

- com.amazonaws. *wilayah.lem*

Jika Anda mengaktifkan DNS pribadi untuk titik akhir, Anda dapat membuat permintaan API untuk AWS Glue menggunakan nama DNS default untuk Wilayah, misalnya,. `glue.us-east-1.amazonaws.com`

Untuk informasi lebih lanjut, lihat Mengakses layanan melalui titik akhir antarmuka dalam Panduan Pengguna Amazon VPC.

Membuat kebijakan VPC endpoint untuk AWS Glue

Anda dapat melampirkan kebijakan titik akhir ke VPC endpoint yang mengendalikan akses ke AWS Glue. Kebijakan menentukan informasi berikut ini:

- Prinsip-prinsip yang dapat melakukan tindakan.
- Tindakan yang dapat dilakukan.
- Sumber daya yang dapat digunakan untuk mengambil tindakan.

Untuk informasi lebih lanjut, lihat Mengendalikan akses ke layanan dengan VPC endpoint di Panduan Pengguna Amazon VPC.

Contoh: Kebijakan titik akhir VPC AWS Glue untuk mengizinkan pembuatan dan pembaruan lapangan kerja

Berikut adalah contoh kebijakan titik akhir untuk AWS Glue. Jika dilampirkan ke sebuah titik akhir, kebijakan ini memberikan akses ke tindakan AWS Glue yang terdaftar untuk semua yang utama di semua sumber daya.

```
{
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
             "glue:CreateJob",
             "glue:UpdateJob",
             "iam:PassRole"
         ],
         "Resource":"*"
      }
   ]
}
```

Contoh: Kebijakan titik akhir VPC untuk mengizinkan akses Katalog Data hanya-baca

Berikut adalah contoh kebijakan titik akhir untuk AWS Glue. Jika dilampirkan ke sebuah titik akhir, kebijakan ini memberikan akses ke tindakan AWS Glue yang terdaftar untuk semua yang utama di semua sumber daya.

```
{
    "Statement": [
        {
            "Principal": "*",
            "Effect": "Allow",
            "Action": [
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:GetTable",
                "glue:GetTables",
                "glue:GetTableVersion",
                "glue:GetTableVersions",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:BatchGetPartition",
                "glue:SearchTables"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon VPC Bersama

AWS Glue mendukung virtual private cloud (VPC) bersama di Amazon Virtual Private Cloud. Amazon VPC bersama memungkinkan beberapa akun AWS untuk menciptakan sumber daya aplikasi mereka, seperti instans Amazon EC2 dan basis data Amazon Relational Database Service (Amazon RDS), ke dalam VPC Amazon bersama yang dikelola secara terpusat. Dalam model ini, akun yang memiliki VPC (pemilik) membagikan satu atau beberapa subnet dengan akun lain (peserta) yang termasuk dalam organisasi yang sama dari AWS Organizations. Setelah subnet dibagikan, peserta dapat melihat, membuat, mengubah, dan menghapus sumber daya aplikasi mereka di subnet yang dibagikan dengan mereka.

Di AWS Glue, untuk membuat koneksi dengan subnet bersama, Anda harus membuat grup keamanan dalam akun Anda dan melampirkan grup keamanan ke subnet bersama tersebut.

Untuk informasi lebih lanjut, lihat topik-topik ini:

- Bekerja dengan VPC Bersama di Panduan Pengguna Amazon VPC
- Apa itu AWS Organizations? di Panduan Pengguna AWS Organizations

Pemecahan Masalah AWS Glue

Topik

Mengumpulkan informasi AWS Glue pemecahan masalah

Jika Anda mengalami kesalahan atau perilaku tak terduga di AWS Glue dan perlu meng-kontak AWS Support, maka Anda harus terlebih dahulu mengumpulkan informasi tentang nama, ID, dan log yang terkait dengan tindakan yang gagal. Dengan menyediakan informasi ini akan memungkinkan AWS Support untuk membantu Anda mengatasi masalah yang Anda alami.

Bersama dengan ID akun, kumpulkan informasi berikut untuk masing-masing jenis kegagalan:

Ketika crawler gagal, kumpulkan informasi berikut:

- Nama crawler

  Log dari crawler run terletak di CloudWatch Log di bawah`/aws-glue/crawlers`.

Ketika koneksi uji gagal, kumpulkan informasi berikut:

- Nama koneksi
- ID Koneksi
- String koneksi JDBC dalam bentuk `jdbc:protocol://host:port/database-name`.

  Log dari koneksi pengujian terletak di CloudWatch Log di bawah `/aws-glue/testconnection`.

Ketika tugas gagal, kumpulkan informasi berikut:

- Nama tugas
- ID eksekusi tugas dalam bentuk `jr_xxxxx`.

  Log dari pekerjaan berjalan terletak di CloudWatch Log di bawah `/aws-glue/jobs`.

Memecahkan masalah kesalahan untuk Spark AWS Glue

Jika Anda mengalami kesalahan AWS Glue, gunakan solusi berikut untuk membantu Anda menemukan sumber masalah dan memperbaikinya.

> **Note**
>
> AWS Glue GitHub Repositori berisi panduan pemecahan masalah tambahan di AWS Glue Pertanyaan yang Sering Diajukan.

Topik

Kesalahan: Sumber daya tidak tersedia

Jika AWS Glue menampilkan pesan sumber daya yang tidak tersedia, Anda dapat melihat pesan kesalahan atau log untuk membantu Anda mempelajari lebih lanjut tentang masalah tersebut. Tugas berikut menjelaskan metode umum untuk menyelesaikan masalah.

- Untuk koneksi dan titik akhir pengembangan yang Anda gunakan, periksa apakah klaster Anda tidak kehabisan antarmuka jaringan elastis.

Kesalahan: Tidak dapat menemukan titik akhir S3 atau gateway NAT untuk SubnetID di VPC

Periksa ID subnet dan ID VPC dalam pesan tersebut untuk membantu Anda mendiagnosis masalahnya.

- Periksa apakah Anda telah menyiapkan VPC endpoint Amazon S3, yang diperlukan dengan AWS Glue. Selain itu, periksa gateway NAT Anda apakah sudah menjadi bagian dari konfigurasi Anda. Untuk informasi selengkapnya, lihat Amazon VPC endpoint untuk Amazon S3.

Kesalahan: Aturan masuk dalam grup keamanan diperlukan

Setidaknya harus ada satu grup keamanan yang membuka semua port ingress-nya. Untuk membatasi lalu lintas, grup keamanan sumber dalam aturan inbound Anda dapat dibatasi untuk grup keamanan yang sama.

- Untuk setiap koneksi yang Anda gunakan, periksa grup keamanan Anda apakah ada aturan inbound yang self-referencing. Untuk informasi selengkapnya, lihat Menyiapkan akses jaringan ke penyimpanan data.
- Ketika Anda menggunakan titik akhir pengembangan, periksa grup keamanan Anda apakah ada aturan inbound yang self-referencing. Untuk informasi selengkapnya, lihat Menyiapkan akses jaringan ke penyimpanan data.

Kesalahan: Aturan keluar dalam grup keamanan diperlukan

Setidaknya harus ada satu grup keamanan yang membuka semua port egress-nya. Untuk membatasi lalu lintas, grup keamanan sumber dalam aturan outbound Anda dapat dibatasi untuk grup keamanan yang sama.

- Untuk setiap koneksi yang Anda gunakan, periksa grup keamanan Anda apakah ada aturan outbound yang self-referencing. Untuk informasi selengkapnya, lihat Menyiapkan akses jaringan ke penyimpanan data.
- Ketika Anda menggunakan titik akhir pengembangan, periksa grup keamanan Anda apakah ada aturan outbound yang self-referencing. Untuk informasi selengkapnya, lihat Menyiapkan akses jaringan ke penyimpanan data.

Kesalahan: Job run gagal karena peran yang diteruskan harus diberikan izin peran untuk layanan AWS Glue

Pengguna yang mendefinisikan tugas harus memiliki izin untuk `iam:PassRole` untuk AWS Glue.

- Saat pengguna membuat AWS Glue pekerjaan, konfirmasikan bahwa peran pengguna berisi kebijakan yang berisi `iam:PassRole` untuk AWS Glue. Untuk informasi selengkapnya, lihat Langkah 3: Lampirkan kebijakan ke pengguna atau grup yang mengakses AWS Glue.

Kesalahan: DescribeVpcEndpoints tindakan tidak sah. tidak dapat memvalidasi VPC ID vpc-id

- Periksa kebijakan yang diteruskan AWS Glue untuk `ec2:DescribeVpcEndpoints` mendapatkan izin.

Kesalahan: DescribeRouteTables tindakan tidak sah. tidak dapat memvalidasi subnet id: Subnet-ID di VPC id: vpc-id

- Periksa kebijakan yang diteruskan AWS Glue untuk `ec2:DescribeRouteTables` mendapatkan izin.

Kesalahan: Gagal memanggil ec2: DescribeSubnets

- Periksa kebijakan yang diteruskan AWS Glue untuk `ec2:DescribeSubnets` mendapatkan izin.

Kesalahan: Gagal memanggil ec2: DescribeSecurityGroups

- Periksa kebijakan yang diteruskan AWS Glue untuk `ec2:DescribeSecurityGroups` mendapatkan izin.

Kesalahan: Tidak dapat menemukan subnet untuk AZ

- Availability Zone mungkin tidak tersedia untuk AWS Glue. Membuat dan menggunakan subnet baru di Availability Zone yang berbeda dari yang ditentukan dalam pesan.

Kesalahan: Pengecualian Job run saat menulis ke target JDBC

Ketika Anda menjalankan sebuah tugas yang menulis ke target JDBC, tugas mungkin mengalami kesalahan dalam skenario berikut:

- Jika tugas Anda menulis ke sebuah tabel Microsoft SQL Server, dan tabel tersebut memiliki kolom yang didefinisikan sebagai jenis `Boolean`, maka tabel harus ditentukan sebelumnya dalam basis data SQL Server. Saat Anda menentukan pekerjaan di AWS Glue konsol menggunakan target SQL Server dengan opsi Buat tabel di target data Anda, jangan memetakan kolom sumber apa pun ke kolom target dengan tipe `Boolean` data. Anda mungkin mengalami kesalahan saat tugas berjalan.

  Anda dapat menghindari kesalahan tersebut dengan melakukan hal berikut:

  - Pilih tabel yang ada yang mempunyai kolom Boolean.
  - Edit transformasi `ApplyMapping` dan petakan kolom Boolean di sumber ke nomor atau string dalam target.

  - Edit transformasi `ApplyMapping` untuk menghapus kolom Boolean dari sumber.
- Jika tugas Anda menulis ke tabel Oracle, maka Anda mungkin perlu menyesuaikan panjang nama objek Oracle. Dalam beberapa versi Oracle, panjang pengenal maksimum terbatas pada 30 byte atau 128 byte. Batas ini mempengaruhi nama tabel dan nama kolom dari penyimpanan data target Oracle.

  Anda dapat menghindari kesalahan tersebut dengan melakukan hal berikut:

  - Berikan nama pada tabel target Oracle dengan panjang yang masih dalam batas untuk versi Anda.
  - Nama kolom default dihasilkan dari nama bidang dalam data. Untuk menangani kasus ketika nama kolom lebih panjang dari batas yang berlaku, gunakan transformasi `ApplyMapping` atau `RenameField` untuk mengubah nama kolom sehingga masih dalam batas.

Kesalahan: Amazon S3: Operasi tidak valid untuk kelas penyimpanan objek

Jika AWS Glue mengembalikan kesalahan ini, AWS Glue pekerjaan Anda mungkin membaca data dari tabel yang memiliki partisi di seluruh tingkatan kelas penyimpanan Amazon S3.

- Dengan menggunakan pengecualian kelas penyimpanan, Anda dapat memastikan bahwa AWS Glue pekerjaan Anda akan bekerja pada tabel yang memiliki partisi di seluruh tingkatan kelas penyimpanan ini. Tanpa pengecualian, pekerjaan yang membaca data dari tingkatan ini gagal dengan kesalahan berikut:. `AmazonS3Exception: The operation is not valid for the object's storage class`

  Untuk informasi selengkapnya, lihat Tidak termasuk kelas penyimpanan Amazon S3.

Kesalahan: Batas waktu Amazon S3

Jika AWS Glue mengembalikan kesalahan kehabisan waktu koneksi, itu mungkin karena mencoba mengakses bucket Amazon S3 di AWS Wilayah lain.

- Titik akhir VPC Amazon S3 hanya dapat merutekan lalu lintas ke bucket dalam suatu Wilayah. AWS Jika Anda perlu connect ke bucket di Wilayah lain, solusi yang mungkin adalah dengan menggunakan gateway NAT. Untuk informasi selengkapnya, lihat Gateway NAT.

Kesalahan: Akses Amazon S3 ditolak

Jika AWS Glue mengembalikan kesalahan akses ditolak ke bucket atau objek Amazon S3, itu mungkin karena peran IAM yang diberikan tidak memiliki kebijakan dengan izin ke penyimpanan data Anda.

- Tugas ETL harus memiliki akses ke penyimpanan data Amazon S3 yang digunakan sebagai sumber atau target. Sebuah crawler harus memiliki akses ke penyimpanan data Amazon S3 yang di-crawling-nya. Untuk informasi selengkapnya, lihat Langkah 2: Buat peran IAM untuk AWS Glue.

Kesalahan: ID kunci akses Amazon S3 tidak ada

Jika AWS Glue mengembalikan ID kunci akses tidak ada kesalahan saat menjalankan pekerjaan, itu mungkin karena salah satu alasan berikut:

- Tugas ETL menggunakan sebuah IAM role untuk mengakses penyimpanan data, konfirmasi bahwa IAM role untuk tugas Anda tidak dihapus sebelum tugas dimulai.
- IAM role berisi izin untuk mengakses penyimpanan data Anda, konfirmasi bahwa kebijakan Amazon S3 yang dilampirkan berisi `s3:ListBucket` yang benar.

Kesalahan: Job run gagal saat mengakses Amazon S3 dengan URI `s3a://`

Jika eksekusi tugas mengembalikan kesalahan seperti Gagal mengurai dokumen XML dengan kelas handler, itu mungkin karena kegagalan mencoba mencantumkan ratusan file menggunakan URI `s3a://`. Mengakses penyimpanan data Anda menggunakan URI `s3://` sebagai gantinya. Jejak pengecualian berikut menyoroti kesalahan untuk mencari:

```
1. com.amazonaws.SdkClientException: Failed to parse XML document with handler class
 com.amazonaws.services.s3.model.transform.XmlResponsesSaxParser$ListBucketHandler
2. at
 com.amazonaws.services.s3.model.transform.XmlResponsesSaxParser.parseXmlInputStream(XmlRespons
3. at
 com.amazonaws.services.s3.model.transform.XmlResponsesSaxParser.parseListBucketObjectsResponse
4. at com.amazonaws.services.s3.model.transform.Unmarshallers
$ListObjectsUnmarshaller.unmarshall(Unmarshallers.java:70)
```

```
5. at com.amazonaws.services.s3.model.transform.Unmarshallers
$ListObjectsUnmarshaller.unmarshall(Unmarshallers.java:59)
6. at
 com.amazonaws.services.s3.internal.S3XmlResponseHandler.handle(S3XmlResponseHandler.java:62)
7. at
 com.amazonaws.services.s3.internal.S3XmlResponseHandler.handle(S3XmlResponseHandler.java:31)
8. at
 com.amazonaws.http.response.AwsResponseHandlerAdapter.handle(AwsResponseHandlerAdapter.java:70
9. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.handleResponse(AmazonHttpClient.java:1554)
10. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.executeOneRequest(AmazonHttpClient.java:1272)
11. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.executeHelper(AmazonHttpClient.java:1056)
12. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.doExecute(AmazonHttpClient.java:743)
13. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.executeWithTimer(AmazonHttpClient.java:717)
14. at com.amazonaws.http.AmazonHttpClient
$RequestExecutor.execute(AmazonHttpClient.java:699)
15. at com.amazonaws.http.AmazonHttpClient$RequestExecutor.access
$500(AmazonHttpClient.java:667)
16. at com.amazonaws.http.AmazonHttpClient
$RequestExecutionBuilderImpl.execute(AmazonHttpClient.java:649)
17. at com.amazonaws.http.AmazonHttpClient.execute(AmazonHttpClient.java:513)
18. at com.amazonaws.services.s3.AmazonS3Client.invoke(AmazonS3Client.java:4325)
19. at com.amazonaws.services.s3.AmazonS3Client.invoke(AmazonS3Client.java:4272)
20. at com.amazonaws.services.s3.AmazonS3Client.invoke(AmazonS3Client.java:4266)
21. at com.amazonaws.services.s3.AmazonS3Client.listObjects(AmazonS3Client.java:834)
22. at org.apache.hadoop.fs.s3a.S3AFileSystem.getFileStatus(S3AFileSystem.java:971)
23. at
 org.apache.hadoop.fs.s3a.S3AFileSystem.deleteUnnecessaryFakeDirectories(S3AFileSystem.java:115
24. at org.apache.hadoop.fs.s3a.S3AFileSystem.finishedWrite(S3AFileSystem.java:1144)
25. at org.apache.hadoop.fs.s3a.S3AOutputStream.close(S3AOutputStream.java:142)
26. at org.apache.hadoop.fs.FSDataOutputStream
$PositionCache.close(FSDataOutputStream.java:74)
27. at org.apache.hadoop.fs.FSDataOutputStream.close(FSDataOutputStream.java:108)
28. at org.apache.parquet.hadoop.ParquetFileWriter.end(ParquetFileWriter.java:467)
29. at
 org.apache.parquet.hadoop.InternalParquetRecordWriter.close(InternalParquetRecordWriter.java:1
30. at
 org.apache.parquet.hadoop.ParquetRecordWriter.close(ParquetRecordWriter.java:112)
31. at
 org.apache.spark.sql.execution.datasources.parquet.ParquetOutputWriter.close(ParquetOutputWrit
```

```
32. at org.apache.spark.sql.execution.datasources.FileFormatWriter
$SingleDirectoryWriteTask.releaseResources(FileFormatWriter.scala:252)
33. at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun
$org$apache$spark$sql$execution$datasources$FileFormatWriter$$executeTask
$3.apply(FileFormatWriter.scala:191)
34. at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun
$org$apache$spark$sql$execution$datasources$FileFormatWriter$$executeTask
$3.apply(FileFormatWriter.scala:188)
35. at org.apache.spark.util.Utils
$.tryWithSafeFinallyAndFailureCallbacks(Utils.scala:1341)
36. at org.apache.spark.sql.execution.datasources.FileFormatWriter$.org$apache$spark
$sql$execution$datasources$FileFormatWriter$$executeTask(FileFormatWriter.scala:193)
37. at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun$write$1$
$anonfun$3.apply(FileFormatWriter.scala:129)
38. at org.apache.spark.sql.execution.datasources.FileFormatWriter$$anonfun$write$1$
$anonfun$3.apply(FileFormatWriter.scala:128)
39. at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:87)
40. at org.apache.spark.scheduler.Task.run(Task.scala:99)
41. at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:282)
42. at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1149)
43. at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:624)
44. at java.lang.Thread.run(Thread.java:748)
```

Kesalahan: Token layanan Amazon S3 kedaluwarsa

Saat memindahkan data ke dan dari Amazon Redshift dengan menggunakan kredensial Amazon S3 sementara yang kedaluwarsa setelah 1 jam. Jika Anda memiliki tugas yang berjalan lama, ia mungkin gagal. Untuk informasi tentang cara menyiapkan tugas yang berjalan lama Anda untuk memindahkan data ke dan dari Amazon Redshift, lihat aws-glue-programming-etl-connect-redshift-home.

Kesalahan: Tidak ada DNS pribadi untuk antarmuka jaringan yang ditemukan

Jika tugas gagal atau titik akhir pengembangan gagal untuk penyediaan, hal itu mungkin karena masalah dalam penyiapan jaringan.

- Jika Anda menggunakan DNS yang disediakan Amazon, nilai `enableDnsHostnames` harus diatur ke BETUL. Untuk informasi lebih lanjut, lihat DNS.

Kesalahan: Penyediaan titik akhir pengembangan gagal

Jika AWS Glue gagal menyediakan titik akhir pengembangan dengan sukses, itu mungkin karena masalah dalam pengaturan jaringan.

- Ketika Anda menentukan titik akhir pengembangan, VPC, subnet, dan grup keamanan divalidasi untuk mengonfirmasi apakah mereka memenuhi persyaratan yang ditentukan.
- Jika Anda memberikan kunci publik SSH opsional, periksa apakah kunci tersebut adalah kunci publik SSH yang benar.
- Periksa di konsol VPC bahwa VPC Anda menggunakan Set opsi DHCP. Untuk informasi selengkapnya, lihat Set opsi DHCP.
- Jika klaster tetap dalam status PROVISSIONING, kontak AWS Support.

Kesalahan: Server notebook CREATE_FAILED

Jika AWS Glue gagal membuat server notebook untuk titik akhir pengembangan, itu mungkin karena salah satu masalah berikut:

- AWS Glue meneruskan peran IAM ke Amazon EC2 saat menyiapkan server notebook. IAM role harus memiliki hubungan kepercayaan dengan Amazon EC2.
- IAM role harus memiliki profil instans dengan nama yang sama. Ketika Anda membuat peran untuk Amazon EC2 dengan konsol IAM, profil instans dengan nama yang sama secara otomatis dibuat. Periksa kesalahan dalam log terkait dengan nama profil instans `iamInstanceProfile.name` yang tidak valid. Untuk informasi selengkapnya, lihat Menggunakan Profil Instance.
- Periksa apakah peran Anda memiliki izin untuk mengakses bucket `aws-glue*` dalam kebijakan yang Anda berikan untuk membuat notebook server.

Kesalahan: Notebook lokal gagal memulai

Jika notebook lokal Anda gagal untuk memulai dan melaporkan kesalahan bahwa direktori atau folder tidak dapat ditemukan, hal itu mungkin karena terjadinya salah satu masalah berikut:

- Jika Anda menjalankan pada Microsoft Windows, pastikan bahwa variabel lingkungan `JAVA_HOME` mengarahkan ke direktori Java yang benar. Dimungkinkan untuk memperbarui Java tanpa memperbarui variabel ini, dan jika menunjuk ke folder yang tidak ada lagi, notebook Jupyter gagal untuk memulai.

Kesalahan: Menjalankan crawler gagal

Jika AWS Glue gagal menjalankan crawler dengan sukses untuk membuat katalog data Anda, itu mungkin karena salah satu alasan berikut. Pertama periksa apakah ada kesalahan yang tercantum dalam daftar crawler konsol AWS Glue . Periksa apakah ada ikon tanda seru di samping nama crawler dan arahkan kursor ke ikon tersebut untuk melihat pesan terkait.

- Periksa log untuk crawler yang dijalankan di CloudWatch Log di bawah`/aws-glue/crawlers`.

Kesalahan: Partisi tidak diperbarui

Jika partisi Anda tidak diperbarui di Katalog Data saat Anda menjalankan pekerjaan ETL, pernyataan log ini dari `DataSink` kelas di CloudWatch log dapat membantu:

- `"Attempting to fast-forward updates to the Catalog - nameSpace:"` — Menunjukkan basis data mana, tabel, dan catalogId mana yang berusaha untuk dimodifikasi oleh tugas ini. Jika pernyataan ini tidak ada, periksa apakah `enableUpdateCatalog` diatur ke BETUL dan diberikan dengan semestinya sebagai parameter `getSink()` atau di `additional_options`.
- `"Schema change policy behavior:"` — Menunjukkan nilai skema `updateBehavior` yang Anda berikan.
- `"Schemas qualify (schema compare):"` — Bisa BETUL atau SALAH.
- `"Schemas qualify (case-insensitive compare):"` — Bisa BETUL atau SALAH.
- Jika keduanya salah dan Anda tidak `updateBehavior` disetel ke `UPDATE_IN_DATABASE`, maka DynamicFrame skema Anda harus identik atau berisi subset kolom yang terlihat dalam skema tabel Katalog Data.

Untuk informasi selengkapnya tentang memperbarui partisi, lihatMemperbarui skema, dan menambahkan partisi baru di Katalog Data menggunakan AWS Glue pekerjaan ETL.

Kesalahan: Pembaruan bookmark Job gagal karena ketidakcocokan versi

Anda mungkin mencoba membuat parameter AWS Glue pekerjaan untuk menerapkan transformasi/logika yang sama pada kumpulan data yang berbeda di Amazon S3. Anda ingin melacak file yang diproses di lokasi yang disediakan. Saat Anda menjalankan pekerjaan yang sama pada bucket

sumber yang sama dan menulis ke tujuan yang sama/berbeda secara bersamaan (konkurensi> 1) pekerjaan gagal dengan kesalahan ini:

```
py4j.protocol.Py4JJavaError: An error occurred while
 callingz:com.amazonaws.services.glue.util.Job.commit.:com.amazonaws.services.gluejobexecutor.m
 Continuation update failed due to version mismatch. Expected version 2 but found
 version 3
```

Solusi: atur konkurensi ke 1 atau jangan menjalankan pekerjaan secara bersamaan.

Saat ini AWS Glue bookmark tidak mendukung proses pekerjaan bersamaan dan komit akan gagal.

Kesalahan: Pekerjaan memproses ulang data saat bookmark pekerjaan diaktifkan

Mungkin ada kasus ketika Anda telah mengaktifkan bookmark AWS Glue pekerjaan, tetapi tugas ETL Anda memproses ulang data yang sudah diproses dalam proses sebelumnya. Periksa penyebab-penyebab umum kesalahan ini:

Konkurensi Maksimum

Menyetel jumlah maksimum proses bersamaan untuk pekerjaan yang lebih besar dari nilai default 1 dapat mengganggu bookmark pekerjaan. Hal ini dapat terjadi ketika bookmark pekerjaan memeriksa waktu modifikasi terakhir dari objek untuk memverifikasi objek mana yang perlu diproses ulang. Untuk informasi selengkapnya, lihat pembahasan konkurensi maksimum di Mengkonfigurasi properti pekerjaan untuk pekerjaan Spark di AWS Glue.

Objek Tugas Tidak Ada

Pastikan bahwa skrip eksekusi tugas Anda berakhir dengan melakukan commit berikut:

```
job.commit()
```

Saat Anda menyertakan objek ini, AWS Glue catat stempel waktu dan jalur pekerjaan yang dijalankan. Jika Anda menjalankan pekerjaan lagi dengan jalur yang sama, hanya AWS Glue memproses file baru. Jika Anda tidak menyertakan objek ini dan bookmark tugas diaktifkan, maka tugas akan memproses ulang file yang sudah diproses bersama dengan file baru dan membuat redundansi di penyimpanan data target tugas.

Parameter Konteks Transformasi Tidak Ada

Konteks transformasi adalah parameter opsional dalam `GlueContext`, namun bookmark tugas tidak akan berfungsi jika Anda tidak memasukkannya. Untuk mengatasi kesalahan ini, tambahkan parameter konteks transformasi saat Anda membuat DynamicFrame, seperti yang ditunjukkan berikut:

```
sample_dynF=create_dynamic_frame_from_catalog(database,
 table_name,transformation_ctx="sample_dynF")
```

Sumber Input

Jika Anda menggunakan basis data relasional (koneksi JDBC) untuk sumber input, maka bookmark tugas hanya akan berfungsi jika kunci primer tabel berada dalam urutan yang berurutan. Bookmark tugas bekerja untuk baris baru, tetapi tidak untuk baris yang diperbarui. Hal itu karena bookmark tugas mencari kunci primer, yang sudah ada. Hal ini tidak berlaku jika sumber input Anda adalah Amazon Simple Storage Service (Amazon S3).

Waktu Terakhir Dimodifikasi

Untuk sumber input Amazon S3, bookmark tugas memeriksa waktu modifikasi terakhir atas objek, bukan nama file, untuk memverifikasi objek mana yang perlu diproses ulang. Jika data sumber masukan Anda telah dimodifkasi sejak eksekusi tugas terakhir Anda, maka file akan diproses kembali ketika Anda menjalankan tugas itu lagi.

Kesalahan: Perilaku failover antara VPC di AWS Glue

Proses berikut digunakan untuk failover untuk pekerjaan di AWS Glue 4.0 dan versi sebelumnya.

Ringkasan: AWS Glue koneksi dipilih pada saat pekerjaan dijalankan. Jika job run mengalami beberapa masalah, (kurangnya alamat IP, konektivitas ke sumber, masalah routing), job run akan gagal. Jika percobaan ulang dikonfigurasi, AWS Glue akan mencoba lagi dengan koneksi yang sama.

1. Untuk setiap upaya run, AWS Glue akan memeriksa kesehatan koneksi dalam urutan yang tercantum dalam konfigurasi pekerjaan, diberikan sampai menemukan satu yang dapat digunakan. Dalam kasus kegagalan Availability Zone (AZ), koneksi dari AZ tersebut akan gagal dalam pemeriksaan dan akan dilewati.
2. AWS Glue memvalidasi koneksi dengan yang berikut:
   - memeriksa id dan subnet Amazon VPC yang valid.
   - memeriksa apakah gateway NAT atau titik akhir VPC Amazon ada.

- memeriksa bahwa subnet memiliki lebih dari 0 alamat IP yang dialokasikan.
- memeriksa apakah AZ sehat.

AWS Glue tidak dapat memverifikasi konektivitas pada saat pengajuan pekerjaan dijalankan.

3. Untuk pekerjaan yang menggunakan Amazon VPC, semua driver dan pelaksana akan dibuat di AZ yang sama dengan koneksi yang dipilih pada saat pengiriman pekerjaan dijalankan.
4. Jika percobaan ulang dikonfigurasi, AWS Glue akan mencoba lagi dengan koneksi yang sama. Ini karena kami tidak dapat menjamin masalah dengan koneksi ini berjalan lama. Jika AZ gagal, pekerjaan yang ada berjalan (tergantung pada tahap pekerjaan yang dijalankan) di AZ tersebut dapat gagal. Coba lagi harus mendeteksi kegagalan AZ dan memilih AZ lain untuk proses baru.

Memecahkan masalah kesalahan crawler saat crawler menggunakan kredenal Lake Formation

Gunakan informasi di bawah ini untuk mendiagnosis dan memperbaiki berbagai masalah saat mengonfigurasi crawler menggunakan kredenal Lake Formation.

Kesalahan: Lokasi S3: s3://examplepath tidak terdaftar

Agar crawler dapat berjalan menggunakan kredenal Lake Formation, Anda harus terlebih dahulu menyiapkan izin Lake Formation. Untuk mengatasi kesalahan ini, harap daftarkan lokasi target Amazon S3 dengan Lake Formation. Untuk informasi selengkapnya, lihat Mendaftarkan lokasi Amazon S3.

Kesalahan: Pengguna/Peran tidak diizinkan untuk melakukan: lakeformation: pada sumber daya GetDataAccess

Harap tambahkan `lakeformation:GetDataAccess` izin ke peran crawler menggunakan konsol IAM atau. AWS CLI Dengan izin ini, Lake Formation memberikan permintaan kredensil sementara untuk mengakses data. Lihat kebijakan di bawah ini:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "lakeformation:GetDataAccess"
```

```
    ],
    "Resource": "*"
  }
}
```

Kesalahan: Izin Lake Formation tidak mencukupi pada (Nama database: ExampleDatabase, Nama Tabel: ExampleTable)

Di konsol Lake Formation (https://console.aws.amazon.com/lakeformation/), berikan izin akses peran crawler (`Create`,`Describe`,`Alter`) pada database, yang ditetapkan sebagai database keluaran. Anda juga dapat memberikan izin di atas meja. Untuk informasi selengkapnya, lihat Memberikan izin database menggunakan metode sumber daya bernama.

Kesalahan: Izin Lake Formation tidak mencukupi di s3://examplepath

1. Perayapan lintas akun
   a. Masuk ke konsol Lake Formation (https://console.aws.amazon.com/lakeformation/) menggunakan akun tempat bucket Amazon S3 terdaftar (akun B). Berikan izin lokasi data ke akun tempat crawler akan dijalankan. Ini akan memungkinkan crawler membaca data dari lokasi Amazon S3 target.
   b. Di akun tempat crawler dibuat (akun A), berikan izin lokasi data pada lokasi Amazon S3 target ke peran IAM yang digunakan untuk menjalankan crawler sehingga crawler dapat membaca data dari tujuan di Lake Formation. Untuk informasi selengkapnya, lihat Memberikan izin lokasi data (akun eksternal).
2. Perayapan dalam akun (crawler dan lokasi Amazon S3 terdaftar berada di akun yang sama) - Berikan izin lokasi data ke peran IAM yang digunakan untuk crawler yang dijalankan di lokasi Amazon S3 sehingga crawler dapat membaca data dari target di Lake Formation. Untuk informasi selengkapnya, lihat Memberikan izin lokasi data (akun yang sama).

Pertanyaan yang sering diajukan tentang konfigurasi crawler menggunakan kredensil Lake Formation

1. Bagaimana cara mengonfigurasi crawler untuk dijalankan menggunakan kredenal Lake Formation menggunakan konsol? AWS

   Di AWS Glue konsol (https://console.aws.amazon.com/glue/), saat mengonfigurasi crawler, pilih opsi Gunakan kredenal Lake Formation untuk merayapi sumber data Amazon S3. Untuk crawling

lintas akun, tentukan Akun AWS ID tempat lokasi Amazon S3 target terdaftar di Lake Formation. Untuk perayapan dalam akun, bidang accountID bersifat opsional.

2. Bagaimana cara mengonfigurasi crawler untuk dijalankan menggunakan kredensil Lake Formation? AWS CLI

   Selama panggilan `CreateCrawler` API, tambahkan`LakeFormationConfiguration`:

   ```
   "LakeFormationConfiguration": {
       "UseLakeFormationCredentials": true,
       "AccountId": "111111111111" (AWS account ID where the target Amazon S3 location
    is registered with Lake Formation)
     }
   ```

3. Apa target yang didukung untuk crawler yang menggunakan kredenal Lake Formation?

   Crawler yang menggunakan kredensi Lake Formation hanya didukung untuk Amazon S3 (perayapan dalam akun dan lintas akun), target Katalog Data dalam akun (di mana lokasi dasarnya adalah Amazon S3), dan target Apache Iceberg.

4. Dapatkah saya merayapi beberapa bucket Amazon S3 sebagai bagian dari crawler tunggal menggunakan kredenal Lake Formation?

   Tidak, untuk merayapi target menggunakan penjual kredenal Lake Formation, lokasi Amazon S3 yang mendasarinya harus milik bucket yang sama. Misalnya, pelanggan dapat menggunakan beberapa lokasi target (`s3://bucket1/folder1, s3://bucket1/folder2`) jika mereka berada di bawah ember yang sama (bucket1). Menentukan bucket yang berbeda (s3://bucket1/folder1, s3://bucket2/folder2) tidak didukung.

Pemecahan masalah AWS Glue untuk kesalahan Ray dari log

AWS Gluemenyediakan akses ke log yang dipancarkan oleh proses Ray selama pekerjaan dijalankan. Jika Anda menemukan kesalahan atau perilaku tak terduga dalam pekerjaan Ray, pertama-tama kumpulkan informasi dari log untuk menentukan penyebab kegagalan. Kami juga menyediakan log serupa untuk sesi interaktif. Log sesi disediakan dengan `/aws-glue/ray/sessions` awalan.

Garis log dikirim ke CloudWatch dalam waktu nyata, karena pekerjaan Anda dijalankan. Pernyataan cetak ditambahkan ke CloudWatch log setelah proses selesai. Log disimpan selama dua minggu setelah pekerjaan dijalankan.

Memeriksa log pekerjaan Ray

Ketika pekerjaan gagal, kumpulkan nama pekerjaan dan ID pekerjaan Anda. Anda dapat menemukannya di AWS Glue konsol. Arahkan ke halaman lowongan, lalu arahkan ke tab Runs. Log pekerjaan ray disimpan dalam grup CloudWatch log khusus berikut.

- `/aws-glue/ray/jobs/script-log/`— Menyimpan log yang dipancarkan oleh skrip Ray utama Anda.
- `/aws-glue/ray/jobs/ray-monitor-log/`— Menyimpan log yang dipancarkan oleh proses autoscaler Ray. Log ini dihasilkan untuk node kepala dan bukan untuk node pekerja lainnya.
- `/aws-glue/ray/jobs/ray-gcs-logs/`— Menyimpan log yang dipancarkan oleh proses GCS (toko kontrol global). Log ini dihasilkan untuk node kepala dan bukan untuk node pekerja lainnya.
- `/aws-glue/ray/jobs/ray-process-logs/`— Menyimpan log yang dipancarkan oleh proses Ray lainnya (terutama agen dasbor) yang berjalan di simpul kepala. Log ini dihasilkan untuk node kepala dan bukan untuk node pekerja lainnya.
- `/aws-glue/ray/jobs/ray-raylet-logs/`— Menyimpan log yang dipancarkan oleh setiap proses raylet. Log ini dikumpulkan dalam satu aliran untuk setiap node pekerja, termasuk node kepala.
- `/aws-glue/ray/jobs/ray-worker-out-logs/`— Menyimpan `stdout` log untuk setiap pekerja di cluster. Log ini dihasilkan untuk setiap node pekerja, termasuk node kepala.
- `/aws-glue/ray/jobs/ray-worker-err-logs/`— Menyimpan `stderr` log untuk setiap pekerja di cluster. Log ini dihasilkan untuk setiap node pekerja, termasuk node kepala.
- `/aws-glue/ray/jobs/ray-runtime-env-log/`— Menyimpan log tentang proses penyiapan Ray. Log ini dihasilkan untuk setiap node pekerja, termasuk node kepala.

Memecahkan masalah kesalahan pekerjaan Ray

Untuk memahami organisasi grup log Ray, dan untuk menemukan grup log yang akan membantu Anda memecahkan masalah kesalahan Anda, ada baiknya memiliki informasi latar belakang tentang arsitektur Ray.

Dalam AWS Glue ETL, seorang pekerja berhubungan dengan sebuah instance. Saat Anda mengonfigurasi pekerja untuk suatu AWS Glue pekerjaan, Anda mengatur jenis dan jumlah instance yang didedikasikan untuk pekerjaan tersebut. Ray menggunakan istilah pekerja dengan cara yang berbeda.

Ray menggunakan node kepala dan node pekerja untuk membedakan tanggung jawab sebuah instance dalam cluster Ray. Node pekerja Ray dapat meng-host beberapa proses aktor yang melakukan perhitungan untuk mencapai hasil komputasi terdistribusi Anda. Aktor yang menjalankan replika fungsi disebut replika. Aktor replika juga bisa disebut proses pekerja. Replika juga dapat berjalan pada node kepala, yang dikenal sebagai head karena menjalankan proses tambahan untuk mengoordinasikan cluster.

Setiap aktor yang berkontribusi pada perhitungan Anda menghasilkan aliran lognya sendiri. Ini memberi kami beberapa wawasan:

- Jumlah proses yang memancarkan log mungkin lebih besar dari jumlah pekerja yang dialokasikan untuk pekerjaan itu. Seringkali, setiap inti pada setiap instance memiliki aktor.
- Node kepala Ray memancarkan manajemen cluster dan log startup. Sebaliknya, node pekerja Ray hanya memancarkan log untuk pekerjaan yang dilakukan pada mereka.

Untuk informasi selengkapnya tentang arsitektur Ray, lihat Whitepaper Arsitektur dalam dokumentasi Ray.

Area masalah: Akses Amazon S3

Periksa pesan kegagalan dari pekerjaan yang dijalankan. Jika itu tidak memberikan informasi yang cukup, periksa`/aws-glue/ray/jobs/script-log/`.

Area masalah: Manajemen ketergantungan PIP

Memeriksa.`/aws-glue/ray/jobs/ray-runtime-env-log/`

Area masalah: Memeriksa nilai antara dalam proses utama

Tulis ke `stderr` atau `stdout` dari skrip utama Anda, dan ambil log dari`/aws-glue/ray/jobs/script-log/`.

Area masalah: Memeriksa nilai antara dalam proses anak

Menulis ke `stderr` atau `stdout` dari `remote` fungsi Anda. Kemudian, ambil log dari `/aws-glue/ray/jobs/ray-worker-out-logs/` atau`/aws-glue/ray/jobs/ray-worker-err-logs/`. Fungsi Anda mungkin telah berjalan pada replika apa pun, jadi Anda mungkin harus memeriksa beberapa log untuk menemukan output yang Anda inginkan.

Area masalah: Menafsirkan alamat IP dalam pesan kesalahan

Dalam situasi kesalahan tertentu, pekerjaan Anda mungkin memancarkan pesan kesalahan yang berisi alamat IP. Alamat IP ini bersifat sementara, dan digunakan oleh cluster untuk mengidentifikasi dan berkomunikasi antar node. Log untuk node akan dipublikasikan ke aliran log dengan akhiran unik berdasarkan alamat IP.

Di CloudWatch, Anda dapat memfilter log Anda untuk memeriksa yang spesifik untuk alamat IP ini dengan mengidentifikasi akhiran ini. Misalnya, diberikan *FAILED_IP dan JOB_RUN_ID*, Anda dapat mengidentifikasi akhiran dengan:

```
filter @logStream like /JOB_RUN_ID/
| filter @message like /IP-/
| parse @message "IP-[*]" as ip
| filter ip like /FAILED_IP/
| fields replace(ip, ":", "_") as uIP
| stats count_distinct by uIP as logStreamSuffix
| display logStreamSuffix
```

AWS Gluepengecualian pembelajaran mesin

Topik ini menjelaskan kode kesalahan HTTP dan string untuk pengecualian AWS Glue yang berkaitan dengan machine learning. Kode kesalahan dan string kesalahan disediakan untuk setiap aktivitas machine learning yang mungkin terjadi ketika Anda melakukan sebuah operasi. Selain itu, Anda juga dapat melihat apakah mungkin untuk mencoba lagi operasi yang mengakibatkan kesalahan tersebut.

CancelMl TaskRunActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”
  - “Tidak ada Tugas ML Run yang ditemukan untuk [taskRunId]: di akun [accountID] untuk transformasi [transformName].”

  OK untuk mencoba lagi: Tidak.

CreateMl TaskRunActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- InvalidInputException (400)
  - “Kegagalan layanan internal yang diakibatkan input tak terduga.”
  - “Sebuah sumber masukan tabel AWS Glue harus ditentukan dalam transformasi.”
  - “Kolom sumber masukan [columnName] memiliki tipe data yang tidak valid yang ditentukan dalam katalog.”
  - “Tepat satu tabel catatan masukan harus disediakan.”
  - “Harus menentukan nama basis data.”
  - “Harus menentukan nama tabel.”
  - “Skema tidak didefinisikan pada transformasi.”
  - “Skema harus berisi kunci primer yang diberikan: [primaryKey].”
  - “Masalah saat mengambil skema katalog data: [pesan].”
  - “Tidak dapat mengatur Kapasitas Maksimal dan Nomor/Tipe Pekerja secara bersamaan.”
  - “Keduanya WorkerType dan NumberOfWorkers harus diatur.”
  - “MaxCapacity seharusnya >= [MaxCapacity].”
  - “NumberOfWorkers seharusnya >= [MaxCapacity].”
  - “Coba lagi maksimal tidak boleh negatif.”
  - “Parameter Temukan Kecocokan belum ditetapkan.”
  - “Kunci primer harus ditentukan dalam parameter Temukan Kecocokan.”

  OK untuk mencoba lagi: Tidak.
- AlreadyExistsException (400)
  - “Transformasi dengan nama [transformName] sudah ada.”

  OK untuk mencoba lagi: Tidak.
- IdempotentParameterMismatchException (400)
  - “Idempoten membuat permintaan untuk transformasi [transformName] memiliki parameter ketidakcocokan.”

  OK untuk mencoba lagi: Tidak.
- InternalServiceException (500)

- “Kegagalan dependensi.”

OK untuk mencoba lagi: Ya.

- ResourceNumberLimitExceededException (400)
  - “Jumlah Transformasi ML ([count]) telah melampaui batas [limit] transformasi.”

  OK untuk mencoba lagi: Ya, setelah Anda menghapus transformasi untuk memberi ruang bagi yang baru ini.

DeleteMl TransformActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName]”

  OK untuk mencoba lagi: Tidak.

GetMl TaskRunActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”
  - “Tidak ada Tugas ML Run yang ditemukan untuk [taskRunId]: di akun [accountID] untuk transformasi [transformName].”

  OK untuk mencoba lagi: Tidak.

GetMl TaskRunsActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”
  - “Tidak ada Tugas ML Run yang ditemukan untuk [taskRunId]: di akun [accountID] untuk transformasi [transformName].”

OK untuk mencoba lagi: Tidak.

GetMl TransformActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”

  OK untuk mencoba lagi: Tidak.

GetMl TransformsActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”

  OK untuk mencoba lagi: Tidak.
- InvalidInputException (400)
  - “ID Akun tidak boleh kosong.”
  - “Penyortiran tidak didukung untuk kolom [column].”
  - “[column] tidak boleh kosong.”
  - “Kegagalan layanan internal yang diakibatkan input tak terduga.”

  OK untuk mencoba lagi: Tidak.

GetSaveLocationForTransformArtifactActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”

  OK untuk mencoba lagi: Tidak.
- InvalidInputException (400)

- “Tipe artefak yang tidak didukung [artifactType].”
- “Kegagalan layanan internal yang diakibatkan input tak terduga.”

OK untuk mencoba lagi: Tidak.

GetTaskRunArtifactActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”
  - “Tidak ada Tugas ML Run yang ditemukan untuk [taskRunId]: di akun [accountID] untuk transformasi [transformName].”

  OK untuk mencoba lagi: Tidak.
- InvalidInputException (400)
  - “Nama file ‘[fileName]’ tidak valid untuk dipublikasikan.”
  - “Tidak dapat mengambil artefak untuk jenis tugas [taskType].”
  - “Tidak dapat mengambil artefak untuk [artifactType].”
  - “Kegagalan layanan internal yang diakibatkan input tak terduga.”

  OK untuk mencoba lagi: Tidak.

PublishMl TransformModelActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”
  - “Model yang sudah ada dengan versi - [version] tidak dapat ditemukan untuk akun id - [accountId] - dan id transformasi - [transformId].”

  OK untuk mencoba lagi: Tidak.
- InvalidInputException (400)
  - “Nama file ‘[fileName]’ tidak valid untuk dipublikasikan.”

- “Tanda minus di depan ilegal pada string tak diberi tanda [string].”
- “Angka buruk di akhir [string].”
- “Nilai string [string] melebihi rentang panjang tak diberi tanda.”
- “Kegagalan layanan internal yang diakibatkan input tak terduga.”

OK untuk mencoba lagi: Tidak.

PullLatestML TransformModelActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”

  OK untuk mencoba lagi: Tidak.
- InvalidInputException (400)
  - “Kegagalan layanan internal yang diakibatkan input tak terduga.”

  OK untuk mencoba lagi: Tidak.
- ConcurrentModificationException (400)
  - “Tidak dapat membuat versi model untuk melatih sisipan yang mengakses bersamaan memiliki parameter ketidakcocokan.”
  - “Model transformasi ML untuk id transformasi [transformId] sudah kedaluwarsa atau sedang diperbarui oleh proses lain; Silakan coba lagi.”

  OK untuk mencoba lagi: Ya.

PutJobMetadataForML TransformActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”
  - “Tidak ada Tugas ML Run yang ditemukan untuk [taskRunId]: di akun [accountID] untuk transformasi [transformName].”

OK untuk mencoba lagi: Tidak.

- InvalidInputException (400)
  - “Kegagalan layanan internal yang diakibatkan input tak terduga.”
  - “Jenis metadata pekerjaan tidak dikenal [jobType].”
  - “Harus memberikan ID eksekusi tugas untuk memperbarui.”

  OK untuk mencoba lagi: Tidak.

StartExportLabelsTaskRunActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”
  - “Tidak ada set label ada untuk Id transformasi [transformId] di id akun [accountId].”

  OK untuk mencoba lagi: Tidak.
- InvalidInputException (400)
  - “[message].”
  - “Path S3 yang disediakan tidak berada di wilayah yang sama dengan transformasi. Mengharapkan wilayah - [region], tapi mendapatkan - [region].”

  OK untuk mencoba lagi: Tidak.

StartImportLabelsTaskRunActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”

  OK untuk mencoba lagi: Tidak.
- InvalidInputException (400)
  - “[message].”
  - “Path file label tidak valid.”

- “Tidak dapat mengakses file label pada [labelPath]. [message].”
- “Tidak dapat menggunakan IAM role yang disediakan dalam transformasi. Peran: [role].”
- “File label tidak valid dengan ukuran 0.”
- “Path S3 yang disediakan tidak berada di wilayah yang sama dengan transformasi. Mengharapkan wilayah - [region], tapi mendapatkan - [region].”

OK untuk mencoba lagi: Tidak.

- ResourceNumberLimitExceededException (400)
  - “Label file telah melampaui batas [limit] MB.”

  OK untuk mencoba lagi: Tidak. Pertimbangkan untuk memecahkan file label Anda menjadi beberapa file yang lebih kecil.

StartML EvaluationTaskRunActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”

  OK untuk mencoba lagi: Tidak.
- InvalidInputException (400)
  - “Tepat satu tabel catatan masukan harus disediakan.”
  - “Harus menentukan nama basis data.”
  - “Harus menentukan nama tabel.”
  - “Parameter Temukan Kecocokan belum ditetapkan.”
  - “Kunci primer harus ditentukan dalam parameter Temukan Kecocokan.”

  OK untuk mencoba lagi: Tidak.
- ML TransformNotReadyException (400)
  - “Operasi ini hanya dapat diterapkan untuk transformasi yang sedang dalam status READY.”

  OK untuk mencoba lagi: Tidak.
- InternalServiceException (500)
  - “Kegagalan dependensi.”

OK untuk mencoba lagi: Ya.

- ConcurrentRunsExceededException (400)
  - “Jumlah Eksekusi Tugas ML [count] telah melampaui batas transformasi [limit] eksekusi tugas.”
  - “Jumlah Eksekusi Tugas ML [count] telah melampaui batas [limit] eksekusi tugas.”

  OK untuk mencoba lagi: Ya, setelah menunggu eksekusi tugas selesai.

StartML LabelingSetGenerationTaskRunActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”

  OK untuk mencoba lagi: Tidak.
- InvalidInputException (400)
  - “Tepat satu tabel catatan masukan harus disediakan.”
  - “Harus menentukan nama basis data.”
  - “Harus menentukan nama tabel.”
  - “Parameter Temukan Kecocokan belum ditetapkan.”
  - “Kunci primer harus ditentukan dalam parameter Temukan Kecocokan.”

  OK untuk mencoba lagi: Tidak.
- InternalServiceException (500)
  - “Kegagalan dependensi.”

  OK untuk mencoba lagi: Ya.
- ConcurrentRunsExceededException (400)
  - “Jumlah Eksekusi Tugas ML [count] telah melampaui batas transformasi [limit] eksekusi tugas.”

  OK untuk mencoba lagi: Ya, setelah eksekusi tugas selesai.

UpdateML TransformActivity

Kegiatan ini memiliki pengecualian sebagai berikut:

- EntityNotFoundException (400)
  - “Tidak dapat menemukan MLTransform di akun [accountId] dengan handel [transformName].”

  OK untuk mencoba lagi: Tidak.
- InvalidInputException (400)
  - “Transformasi lain dengan nama [transformName] sudah ada.”
  - “[message].”
  - “Nama transformasi tidak boleh kosong.”
  - “Tidak dapat mengatur Kapasitas Maksimal dan Nomor/Tipe Pekerja secara bersamaan.”
  - “Keduanya WorkerType dan NumberOfWorkers harus diatur.”
  - “MaxCapacity seharusnya >= [minMaxCapacity].”
  - “NumberOfWorkers seharusnya >= [minNumWorkers].”
  - “Coba lagi maksimal tidak boleh negatif.”
  - “Kegagalan layanan internal yang diakibatkan input tak terduga.”
  - “Parameter Temukan Kecocokan belum ditetapkan.”
  - “Kunci primer harus ditentukan dalam parameter Temukan Kecocokan.”

  OK untuk mencoba lagi: Tidak.
- AlreadyExistsException (400)
  - “Transformasi dengan nama [transformName] sudah ada.”

  OK untuk mencoba lagi: Tidak.
- IdempotentParameterMismatchException (400)
  - “Idempoten membuat permintaan untuk transformasi [transformName] memiliki parameter ketidakcocokan.”

  OK untuk mencoba lagi: Tidak.

Kuota AWS Glue

Anda dapat menghubungi AWS Support untuk meminta kenaikan kuota untuk kuota layanan yang tercantum dalam. Referensi Umum AWS Kecuali dinyatakan lain, setiap kuota bersifat khusus per Wilayah. Untuk informasi lebih lanjut, lihat Kuota dan Titik Akhir AWS Glue.

Meningkatkan AWS Glue kinerja

Strategi dasar untuk penyetelan kinerja

Untuk meningkatkan AWS Glue kinerja, Anda dapat mempertimbangkan untuk memperbarui AWS Glue parameter terkait kinerja tertentu. Saat mempersiapkan untuk menyetel parameter, gunakan praktik terbaik berikut:

- Tentukan tujuan kinerja Anda sebelum mulai mengidentifikasi masalah.
- Gunakan metrik untuk mengidentifikasi masalah sebelum mencoba mengubah parameter penyetelan.

Untuk hasil yang paling konsisten saat menyetel pekerjaan, kembangkan strategi dasar untuk pekerjaan penyetelan Anda.

Umumnya, penyetelan kinerja dilakukan dalam alur kerja berikut:

1. Tentukan tujuan kinerja.
2. Ukur metrik.
3. Identifikasi kemacetan.
4. Kurangi dampak kemacetan.
5. Ulangi langkah 2-4 sampai Anda mencapai target yang diinginkan.

Strategi penyetelan untuk jenis pekerjaan Anda

Pekerjaan Spark —ikuti panduan dalam Praktik terbaik untuk penyetelan AWS Glue kinerja untuk pekerjaan Apache Spark di Panduan Preskriptif. AWS

Pekerjaan lain —Anda dapat menyetel AWS Glue pekerjaan shell Ray dan AWS Glue Python dengan mengadaptasi strategi yang tersedia di lingkungan runtime lainnya.

Meningkatkan kinerja AWS Glue untuk pekerjaan Apache Spark

AWS GlueUntuk meningkatkan kinerja Spark, Anda dapat mempertimbangkan untuk memperbarui parameter terkait kinerja AWS Glue dan Spark tertentu.

Untuk informasi selengkapnya tentang strategi spesifik untuk mengidentifikasi kemacetan melalui metrik dan mengurangi dampaknya, lihat Praktik terbaik untuk penyetelan kinerja untuk pekerjaan Apache Spark di Panduan AWS Glue Preskriptif. AWS Panduan ini memperkenalkan Anda pada topik utama yang berlaku untuk Apache Spark di semua lingkungan runtime, seperti arsitektur Spark dan Kumpulan Data Terdistribusi Tangguh. Dengan menggunakan topik-topik tersebut, panduan ini memandu Anda untuk menerapkan strategi penyetelan kinerja tertentu, seperti mengoptimalkan shuffle dan memparalelkan tugas.

Anda dapat mengidentifikasi kemacetan dengan mengonfigurasi AWS Glue untuk menampilkan UI Spark. Untuk informasi selengkapnya, lihat the section called “Pemantauan dengan Spark UI”.

Selain itu, AWS Glue menyediakan fitur kinerja yang mungkin berlaku untuk jenis penyimpanan data tertentu yang terhubung dengan pekerjaan Anda. Informasi referensi tentang parameter kinerja untuk penyimpanan data dapat ditemukan dithe section called “Parameter koneksi”.

Mengoptimalkan pembacaan dengan pushdown di Glue ETL AWS

Pushdown adalah teknik optimasi yang mendorong logika tentang mengambil data lebih dekat ke sumber data Anda. Sumbernya bisa berupa database atau sistem file seperti Amazon S3. Saat menjalankan operasi tertentu secara langsung pada sumbernya, Anda dapat menghemat waktu dan daya pemrosesan dengan tidak membawa semua data melalui jaringan ke mesin Spark yang dikelola oleh GlueAWS.

Cara lain untuk mengatakan ini adalah bahwa pushdown mengurangi pemindaian data. Untuk informasi lebih lanjut tentang proses mengidentifikasi kapan teknik ini tepat, lihat Mengurangi jumlah pemindaian data dalam Panduan Pekerjaan Glue for Apache Spark Praktik Terbaik untuk Penyetelan kinerja AWS Glue untuk Apache Spark pada AWS Panduan Preskriptif.

Predikat pushdown pada file yang disimpan di Amazon S3

Saat bekerja dengan file di Amazon S3 yang telah diatur berdasarkan awalan, Anda dapat memfilter jalur Amazon S3 target Anda dengan menentukan predikat pushdown. Daripada membaca kumpulan data lengkap dan menerapkan filter dalam a`DynamicFrame`, Anda dapat langsung menerapkan filter ke metadata partisi yang disimpan di Katalog Data AWS Glue. Pendekatan ini memungkinkan Anda untuk secara selektif membuat daftar dan hanya membaca data yang diperlukan. Untuk informasi selengkapnya tentang proses ini, termasuk menulis ke ember berdasarkan partisi, lihatthe section called “Mengelola partisi”.

Anda mencapai predikat pushdown di Amazon S3 dengan menggunakan parameter. `push_down_predicate` Pertimbangkan ember di Amazon S3 yang telah Anda partisi berdasarkan tahun, bulan, dan hari. Jika Anda ingin mengambil data pelanggan untuk Juni 2022, Anda dapat menginstruksikan AWS Glue untuk hanya membaca jalur Amazon S3 yang relevan. `push_down_predicate`Dalam hal ini adalah`year='2022' and month='06'`. Menyatukan semuanya, operasi baca dapat dicapai seperti di bawah ini:

Python

```
customer_records = glueContext.create_dynamic_frame.from_catalog(
    database = "customer_db",
    table_name = "customer_tbl",
    push_down_predicate = "year='2022' and month='06'"
)
```

Scala

```
val customer_records = glueContext.getCatalogSource(
database="customer_db",
tableName="customer_tbl",
pushDownPredicate="year='2022' and month='06'"
).getDynamicFrame()
```

Dalam skenario sebelumnya, `push_down_predicate` mengambil daftar semua partisi dari Katalog Data AWS Glue dan menyaringnya sebelum membaca file Amazon S3 yang mendasarinya. Meskipun ini membantu dalam banyak kasus, ketika bekerja dengan kumpulan data yang memiliki jutaan partisi, proses daftar partisi dapat memakan waktu. Untuk mengatasi masalah ini, pemangkasan partisi sisi server dapat digunakan untuk meningkatkan kinerja. Ini dilakukan dengan membangun indeks Partisi untuk data Anda di Katalog Data AWS Glue. Untuk informasi selengkapnya tentang indeks partisi, lihatthe section called “Bekerja dengan indeks partisi”. Anda kemudian dapat menggunakan `catalogPartitionPredicate` opsi untuk mereferensikan indeks. Untuk contoh mengambil partisi dengan`catalogPartitionPredicate`, lihat. the section called “Katalog predikat partisi”

Pushdown saat bekerja dengan sumber JDBC

Pembaca AWS Glue JDBC yang digunakan dalam `GlueContext` mendukung pushdown pada database yang didukung dengan menyediakan kueri SQL khusus yang dapat berjalan langsung pada

sumbernya. Ini dapat dicapai dengan mengatur `sampleQuery` parameter. Kueri sampel Anda dapat menentukan kolom mana yang akan dipilih serta menyediakan predikat pushdown untuk membatasi data yang ditransfer ke mesin Spark.

Secara default, kueri sampel beroperasi pada satu node, yang dapat mengakibatkan kegagalan pekerjaan saat berhadapan dengan volume data yang besar. Untuk menggunakan fitur ini untuk menanyakan data dalam skala besar, Anda harus mengonfigurasi partisi kueri dengan menyetel `enablePartitioningForSampleQuery` ke true, yang akan mendistribusikan kueri ke beberapa node di seluruh kunci pilihan Anda. Partisi kueri juga memerlukan beberapa parameter konfigurasi lain yang diperlukan. Untuk informasi selengkapnya tentang partisi kueri, lihat. the section called “Membaca dari JDBC secara paralel”

Saat mengatur`enablePartitioningForSampleQuery`, AWS Glue akan menggabungkan predikat pushdown Anda dengan predikat partisi saat menanyakan database Anda. Anda `sampleQuery` harus diakhiri dengan `AND` for AWS Glue untuk menambahkan kondisi partisi. (Jika Anda tidak memberikan predikat pushdown, `sampleQuery` harus diakhiri dengan a). `WHERE` Lihat contoh di bawah ini, di mana kita menekan predikat untuk hanya mengambil baris yang `id` lebih besar dari 1000. Ini hanya `sampleQuery` akan mengembalikan kolom nama dan lokasi untuk baris yang `id` lebih besar dari nilai yang ditentukan:

Python

```
sample_query = "select name, location from customer_tbl WHERE id>=1000 AND"
customer_records = glueContext.create_dynamic_frame.from_catalog(
    database="customer_db",
    table_name="customer_tbl",
    sample_query = "select name, location from customer_tbl WHERE id>=1000 AND",

    additional_options = {
                           "hashpartitions": 36 ,
                           "hashfield":"id",
                           "enablePartitioningForSampleQuery":True,
                           "sampleQuery":sample_query
                          }
)
```

Scala

```
val additionalOptions = Map(
        "hashpartitions" -> "36",
```

```
        "hashfield" -> "id",
        "enablePartitioningForSampleQuery" -> "true",
        "sampleQuery" -> "select name, location from customer_tbl WHERE id >= 1000
 AND"
        )

    val customer_records = glueContext.getCatalogSource(
        database="customer_db",
        tableName="customer_tbl").getDynamicFrame()
```

> **Note**
>
> Jika `customer_tbl` memiliki nama yang berbeda dalam Katalog Data dan datastore yang mendasarinya, Anda harus memberikan nama tabel yang mendasarinya di sample_query, karena kueri diteruskan ke datastore yang mendasarinya.

Anda juga dapat melakukan query terhadap tabel JDBC tanpa mengintegrasikan dengan AWS Glue Data Catalog. Alih-alih memberikan nama pengguna dan kata sandi sebagai parameter ke metode, Anda dapat menggunakan kembali kredensil dari koneksi yang sudah ada sebelumnya dengan menyediakan dan. `useConnectionProperties` `connectionName` Dalam contoh ini, kita mengambil kredensil dari koneksi yang disebut. `my_postgre_connection`

Python

```
connection_options_dict = {
    "useConnectionProperties": True,
    "connectionName": "my_postgre_connection",
    "dbtable":"customer_tbl",
    "sampleQuery":"select name, location from customer_tbl WHERE id>=1000 AND",
    "enablePartitioningForSampleQuery":True,
    "hashfield":"id",
    "hashpartitions":36
    }

customer_records = glueContext.create_dynamic_frame.from_options(
    connection_type="postgresql",
    connection_options=connection_options_dict
    )
```

Scala

```
val connectionOptionsJson = """
      {
        "useConnectionProperties": true,
        "connectionName": "my_postgre_connection",
        "dbtable": "customer_tbl",
        "sampleQuery": "select name, location from customer_tbl WHERE id>=1000 AND",
        "enablePartitioningForSampleQuery" : true,
        "hashfield" : "id",
        "hashpartitions" : 36
      }
    """

    val connectionOptions = new JsonOptions(connectionOptionsJson)

    val dyf = glueContext.getSource("postgresql",
 connectionOptions).getDynamicFrame()
```

Catatan dan batasan untuk pushdown di Glue AWS

Pushdown, sebagai sebuah konsep, berlaku saat membaca dari sumber non-streaming. AWS Glue mendukung berbagai sumber - kemampuan untuk menekan tergantung pada sumber dan konektor.

- Saat menghubungkan ke Snowflake, Anda dapat menggunakan opsi iniquery. Fungsionalitas serupa ada di konektor Redshift di AWS Glue 4.0 dan versi yang lebih baru. Untuk informasi lebih lanjut tentang membaca dari Snowflake denganquery, lihat. the section called “Baca dari Snowflake”
- Pembaca DynamoDB ETL tidak mendukung filter atau predikat pushdown. MongoDB dan DocumentDB juga tidak mendukung fungsi semacam ini.
- Saat membaca dari data yang disimpan di Amazon S3 dalam format tabel terbuka, metode partisi untuk file di Amazon S3 tidak lagi cukup. Untuk membaca dan menulis dari partisi menggunakan format tabel terbuka, lihat dokumentasi untuk formatnya.
- DynamicFrame metode tidak melakukan pushdown proyeksi Amazon S3. Semua kolom akan dibaca dari file yang melewati filter predikat.
- Saat bekerja dengan custom.jdbc konektor di AWS Glue, kemampuan untuk menekan tergantung pada sumber dan konektor. Harap tinjau dokumentasi konektor yang sesuai untuk mengonfirmasi apakah dan bagaimana mendukung pushdown di GlueAWS.

Menggunakan penskalaan otomatis untuk AWS Glue

Auto Scaling tersedia untuk AWS Glue ETL dan pekerjaan streaming Anda dengan AWS Glue versi 3.0 atau yang lebih baru.

Dengan Auto Scaling diaktifkan, Anda akan mendapatkan manfaat berikut:

- AWS Gluesecara otomatis menambahkan dan menghapus pekerja dari cluster tergantung pada paralelisme pada setiap tahap atau microbatch dari pekerjaan yang dijalankan.
- Ini menghilangkan kebutuhan bagi Anda untuk bereksperimen dan memutuskan jumlah pekerja yang akan ditugaskan untuk pekerjaan AWS Glue ETL Anda.
- Jika Anda memilih jumlah maksimum pekerja, AWS Glue akan memilih sumber daya ukuran yang tepat untuk beban kerja.
- Anda dapat melihat bagaimana ukuran klaster berubah selama pekerjaan dijalankan dengan melihat CloudWatch metrik pada halaman rincian pekerjaan yang dijalankan di AWS Glue Studio.

Auto Scaling untuk AWS Glue ETL dan pekerjaan streaming memungkinkan penskalaan sesuai permintaan dan pengurangan sumber daya komputasi pekerjaan Anda. AWS Glue Peningkatan skala sesuai permintaan membantu Anda untuk hanya mengalokasikan sumber daya komputasi yang diperlukan pada awalnya pada startup yang dijalankan pekerjaan, dan juga untuk menyediakan sumber daya yang diperlukan sesuai permintaan selama pekerjaan.

Auto Scaling juga mendukung penskalaan dinamis dari sumber daya AWS Glue pekerjaan selama pekerjaan. Selama menjalankan pekerjaan, ketika lebih banyak pelaksana diminta oleh aplikasi Spark Anda, lebih banyak pekerja akan ditambahkan ke cluster. Ketika eksekutor telah menganggur tanpa tugas komputasi aktif, pelaksana dan pekerja terkait akan dihapus.

Skenario umum di mana Auto Scaling membantu biaya dan pemanfaatan untuk aplikasi Spark Anda termasuk driver Spark yang mencantumkan sejumlah besar file di Amazon S3 atau melakukan pemuatan saat pelaksana tidak aktif, tahapan Spark berjalan dengan hanya beberapa pelaksana karena penyediaan berlebihan, dan kemiringan data atau permintaan komputasi yang tidak merata di seluruh tahapan Spark.

Persyaratan

Auto Scaling hanya tersedia untuk AWS Glue versi 3.0 atau yang lebih baru. Untuk menggunakan Auto Scaling, Anda dapat mengikuti panduan migrasi untuk memigrasi pekerjaan yang ada ke AWS

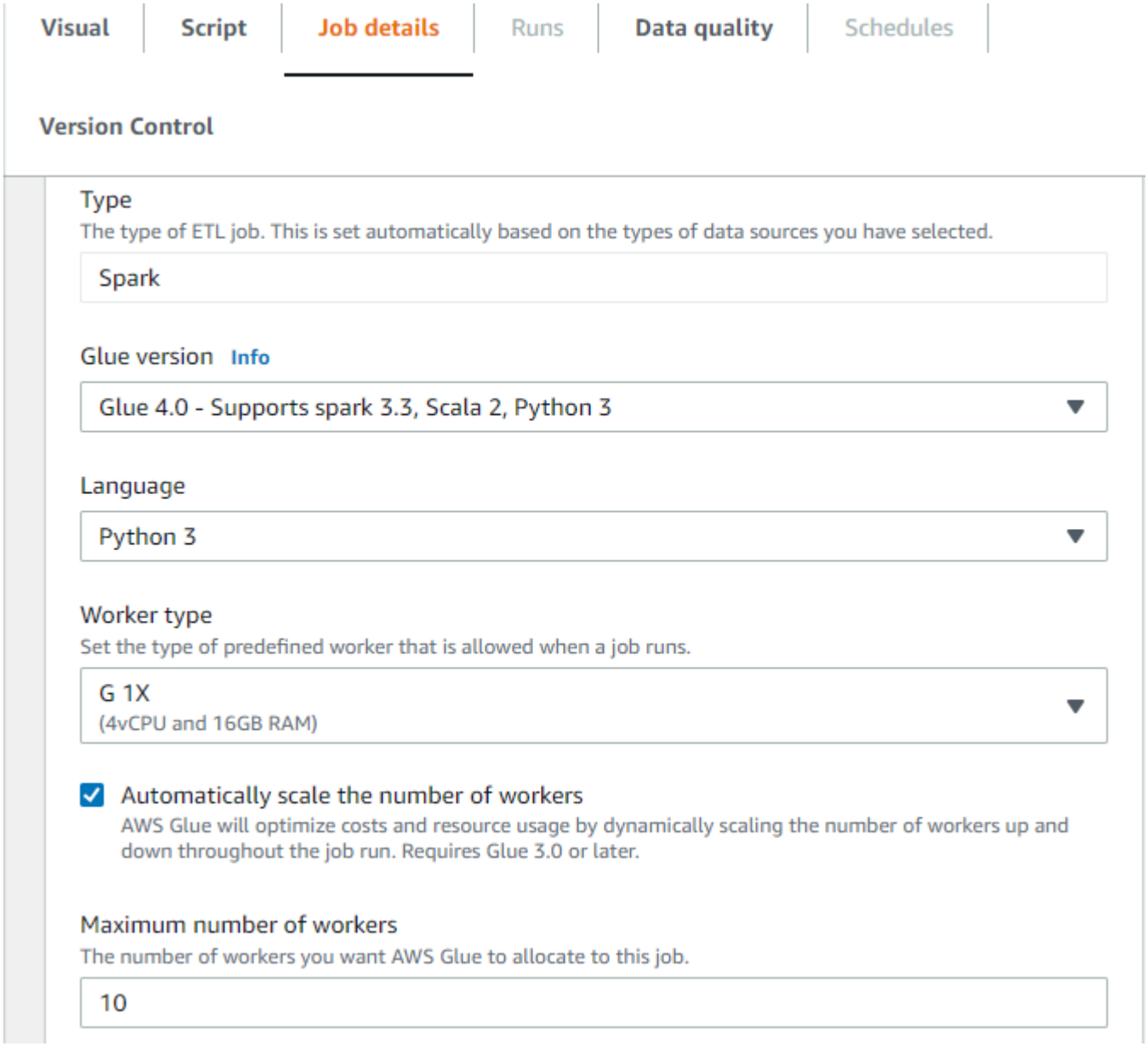
Glue versi 3.0 atau yang lebih baru atau membuat pekerjaan baru dengan AWS Glue versi 3.0 atau yang lebih baru.

Auto Scaling tersedia untuk AWS Glue pekerjaan dengan tipe pekerja`G.1X`,,`G.2X`, `G.4X``G.8X`, atau `G.025X` (hanya untuk pekerjaan Streaming). DPU standar tidak didukung.

Mengaktifkan Auto Scaling di AWS Glue Studio

Pada tab Job details di AWS Glue Studio, pilih jenis sebagai Spark atau Spark Streaming, dan versi Glue sebagai **`Glue 3.0`** atau. **`Glue 4.0`** Kemudian kotak centang akan muncul di bawah Jenis Pekerja.

- Pilih opsi Skala jumlah pekerja secara otomatis.
- Tetapkan jumlah maksimum pekerja untuk menentukan jumlah maksimum pekerja yang dapat dijual untuk menjalankan pekerjaan.



Mengaktifkan Auto Scaling dengan AWS CLI atau SDK

Untuk mengaktifkan Auto Scaling Dari AWS CLI untuk menjalankan pekerjaan Anda, jalankan `start-job-run` dengan konfigurasi berikut:

```
{
    "JobName": "<your job name>",
    "Arguments": {
        "--enable-auto-scaling": "true"
    },
    "WorkerType": "G.2X", // G.1X and G.2X are allowed for Auto Scaling Jobs
```

```
    "NumberOfWorkers": 20, // represents Maximum number of workers
    ...other job run configurations...
}
```

Setelah menjalankan tugas ETL selesai, Anda juga dapat menelepon `get-job-run` untuk memeriksa penggunaan sumber daya aktual dari pekerjaan yang dijalankan dalam DPU-detik. Catatan: bidang baru DPUSeconds hanya akan muncul untuk pekerjaan batch Anda di AWS Glue 3.0 atau yang lebih baru diaktifkan dengan Auto Scaling. Bidang ini tidak didukung untuk pekerjaan streaming.

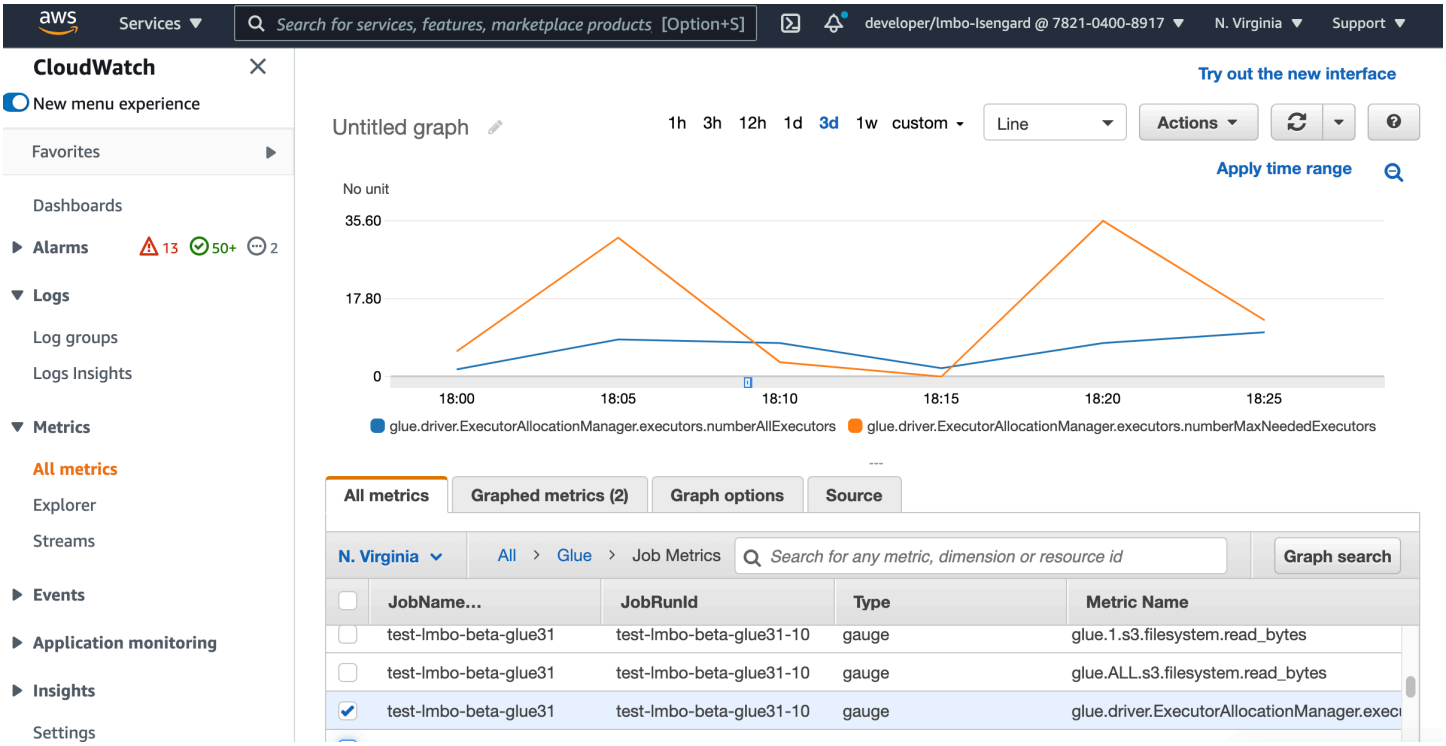
```
$ aws glue get-job-run --job-name your-job-name --run-id jr_xx --endpoint https://
glue.us-east-1.amazonaws.com --region us-east-1
{
    "JobRun": {
        ...
        "GlueVersion": "3.0",
        "DPUSeconds": 386.0
    }
}
```

Anda juga dapat mengonfigurasi job run dengan Auto Scaling menggunakan AWS GlueSDK dengan konfigurasi yang sama.

Memantau Auto Scaling dengan metrik Amazon CloudWatch

Metrik CloudWatch pelaksana tersedia untuk pekerjaan AWS Glue 3.0 atau yang lebih baru jika Anda mengaktifkan Auto Scaling. Metrik dapat digunakan untuk memantau permintaan dan penggunaan pelaksana yang dioptimalkan dalam aplikasi Spark mereka yang diaktifkan dengan Auto Scaling. Untuk informasi selengkapnya, lihat Pemantauan AWS Glue menggunakan CloudWatch metrik Amazon.

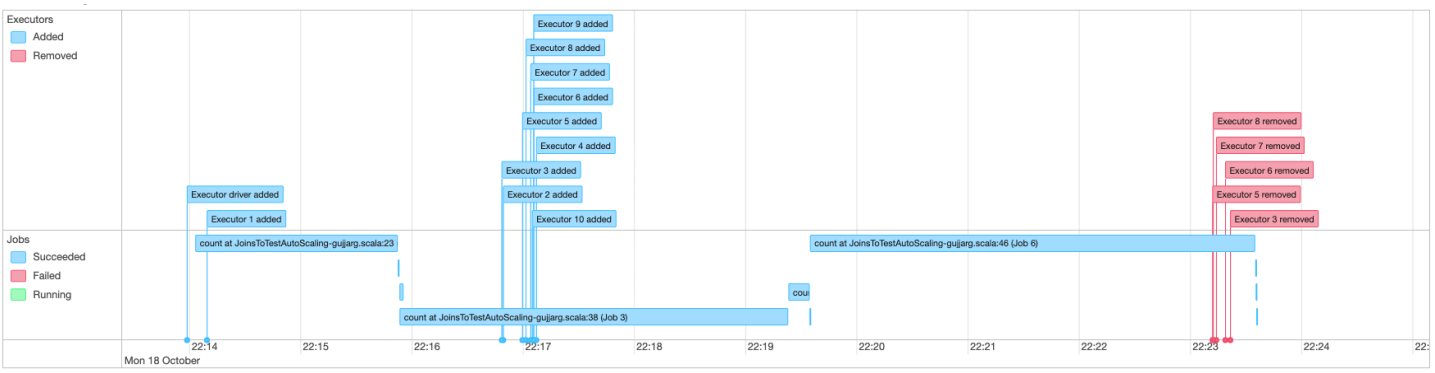
- lem.driver. ExecutorAllocationManager.pelaksana. numberAllExecutors
- lem.driver. ExecutorAllocationManager.pelaksana. numberMaxNeededPelaksana



Untuk detail selengkapnya tentang metrik ini, lihatPemantauan perencanaan kapasitas DPU.

Memantau Auto Scaling dengan Spark UI

Dengan Auto Scaling diaktifkan, Anda juga dapat memantau pelaksana yang ditambahkan dan dihapus dengan penskalaan dinamis dan penurunan skala berdasarkan permintaan dalam pekerjaan AWS Glue Anda menggunakan UI Glue Spark. Untuk informasi selengkapnya, lihat Mengaktifkan UI web Apache Spark untuk pekerjaan AWS Glue.



Pemantauan pekerjaan Auto Scaling menjalankan penggunaan DPU

Anda dapat menggunakan tampilan AWS Glue Studio Job run untuk memeriksa penggunaan DPU dari pekerjaan Auto Scaling Anda.

1. Pilih Monitoring dari panel AWS Glue Studio navigasi. Halaman Monitoring muncul.
2. Gulir ke bawah ke bagan Job running.
3. Arahkan ke pekerjaan yang Anda minati dan gulir ke kolom jam DPU untuk memeriksa penggunaan untuk menjalankan pekerjaan tertentu.

Batasan

AWS Gluestreaming Auto Scaling saat ini tidak mendukung DataFrame gabungan streaming dengan statis yang DataFrame dibuat di luar. `ForEachBatch` Statis yang DataFrame dibuat di dalam `ForEachBatch` akan berfungsi seperti yang diharapkan.

Partisi beban kerja dengan eksekusi terbatas

Kesalahan dalam aplikasi Spark biasanya timbul dari skrip Spark yang tidak efisien, yang didistribusikan dalam memori eksekusi transformasi skala besar, dan kelainan set data. Ada banyak alasan yang dapat menyebabkan masalah kekurangan memori pada driver atau eksekutor, misalnya data skew, mencantumkan terlalu banyak objek, atau data shuffle besar. Masalah ini sering muncul bila Anda memproses sejumlah besar data backlog dengan Spark.

AWS Glue memungkinkan Anda untuk memecahkan masalah OOM dan membuat pemrosesan ETL Anda menjadi lebih mudah dengan pemartisian beban kerja. Dengan mengaktifkan pemartisian beban kerja, setiap eksekusi tugas ETL hanya mengambil data yang belum diproses, dengan batas atas pada ukuran set data atau jumlah file yang akan diproses dengan eksekusi tugas ini. Eksekusi tugas di masa depan akan memproses data yang tersisa. Misalnya, jika ada 1000 file yang harus diproses, Anda dapat mengatur jumlah file menjadi 500 dan memisahkan mereka menjadi dua eksekusi tugas.

Pemartisian beban kerja hanya didukung untuk sumber data Amazon S3.

Mengaktifkan partisi beban kerja

Anda dapat mengaktifkan eksekusi terbatas dengan secara manual menetapkan opsi dalam skrip Anda atau dengan menambahkan properti tabel katalog.

Untuk mengaktifkan pemartisian beban kerja dengan eksekusi terbatas dalam skrip Anda:

1. Untuk menghindari pemrosesan ulang data, aktifkan bookmark tugas di tugas baru atau tugas yang sudah ada. Untuk informasi selengkapnya, lihat Melacak Data yang Diproses Menggunakan Bookmark Tugas.
2. Ubah skrip Anda dan tetapkan batas yang dibatasi dalam pilihan tambahan di API AWS Glue `getSource`. Anda juga harus mengatur konteks transformasi untuk bookmark tugas untuk menyimpan elemen `state`. Sebagai contoh:

   Python

   ```
   glueContext.create_dynamic_frame.from_catalog(
       database = "database",
       table_name = "table_name",
       redshift_tmp_dir = "",
       transformation_ctx = "datasource0",
       additional_options = {
           "boundedFiles" : "500", # need to be string
         # "boundedSize" : "1000000000" unit is byte
       }
   )
   ```

   Skala

   ```
   val datasource0 = glueContext.getCatalogSource(
       database = "database", tableName = "table_name", redshiftTmpDir = "",
       transformationContext = "datasource0",
       additionalOptions = JsonOptions(
           Map("boundedFiles" -> "500") // need to be string
         //"boundedSize" -> "1000000000" unit is byte
       )
   ).getDynamicFrame()
   ```

   ```
   val connectionOptions = JsonOptions(
       Map("paths" -> List(baseLocation), "boundedFiles" -> "30")
   )
   val source = glueContext.getSource("s3", connectionOptions, "datasource0", "")
   ```

Untuk mengaktifkan pemartisian beban kerja dengan eksekusi terbatas dalam tabel Katalog Data Anda:

1. Atur pasangan nilai kunci dalam kolom `parameters` struktur tabel Anda dalam Katalog Data. Untuk informasi selengkapnya, lihat Melihat dan Mengedit Detail Tabel.
2. Atur batas atas untuk ukuran set data atau jumlah file yang diproses:

   - Atur `boundedSize` dengan ukuran target set data dalam byte. Eksekusi tugas akan berhenti setelah mencapai ukuran target dari tabel.
   - Atut `boundedFiles` dengan jumlah file target. Eksekusi tugas akan berhenti setelah memproses jumlah file target.

> **Note**
>
> Anda hanya harus menetapkan salah satunya, `boundedSize` atau `boundedFiles`, karena hanya satu batas yang didukung.

Menyiapkan AWS Glue pemicu untuk menjalankan pekerjaan secara otomatis

Setelah Anda mengaktifkan eksekusi terbatas, Anda dapat menyiapkan pemicu AWS Glue untuk secara otomatis menjalankan tugas dan secara bertahap memuat data dengan eksekusi berurutan. Buka konsol AWS Glue dan buat pemicu, atur waktu jadwal, dan lampirkan ke tugas Anda. Maka ia akan secara otomatis memicu eksekusi tugas berikutnya dan memproses batch data yang baru.

Anda juga dapat menggunakan alur kerja AWS Glue untuk mengatur beberapa tugas agar memproses data dari partisi yang berbeda secara paralel. Untuk informasi selengkapnya, lihat Pemicu AWS Glue dan Alur Kerja AWS Glue.

Untuk informasi lebih lanjut tentang kasus penggunaan dan opsi, silakan lihat blog Mengoptimalkan aplikasi Spark dengan pemartisian beban kerja di AWS Glue.

Masalah yang diketahui untuk AWS Glue

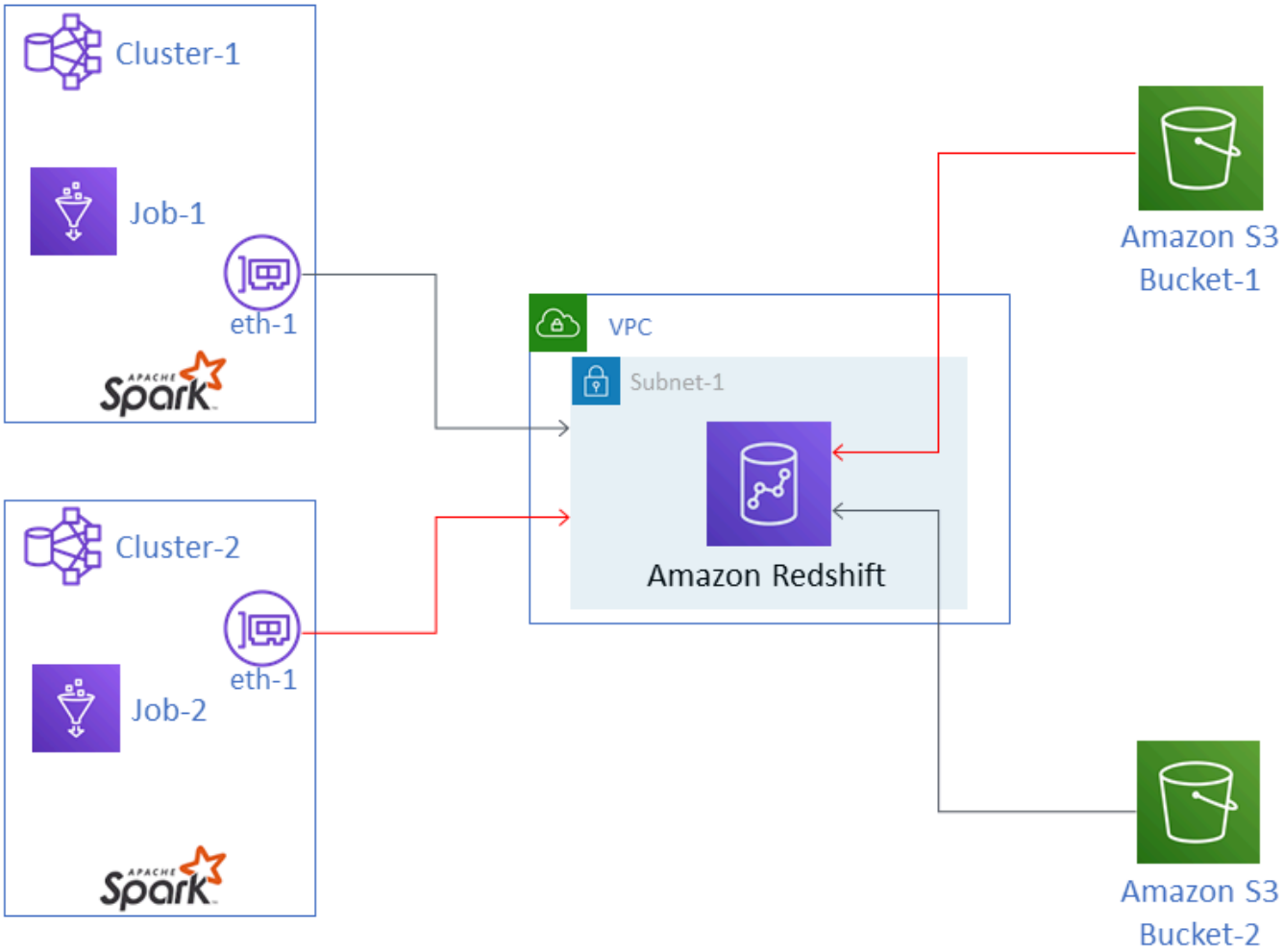
Perhatikan masalah yang diketahui berikut untuk AWS Glue.

Topik

Mencegah akses data lintas pekerjaan

Pertimbangkan situasi di mana Anda memiliki dua tugas Spark AWS Glue dalam satu akun AWS, masing-masing berjalan di klaster Spark AWS Glue secara terpisah. Tugas menggunakan koneksi AWS Glue untuk mengakses sumber daya di virtual private cloud (VPC) yang sama. Dalam situasi ini, sebuah tugas yang berjalan di satu klaster mungkin dapat mengakses data dari tugas yang berjalan di klaster lain.

Diagram berikut menggambarkan contoh dari situasi ini.



Dalam diagram tersebut, `Job-1` AWS Glue sedang berjalan di `Cluster-1`, dan Job-2 berjalan di `Cluster-2`. Kedua tugas bekerja dengan instans yang sama dari Amazon Redshift, yang berada di `Subnet-1` dari sebuah VPC. `Subnet-1` bisa berupa subnet publik atau privat.

`Job-1` mengubah data dari `Bucket-1` Amazon Simple Storage Service (Amazon S3) dan menulis data ke Amazon Redshift. `Job-2` melakukan hal yang sama pada data di `Bucket-2`. `Job-1` menggunakan AWS Identity and Access Management (IAM) role `Role-1` (tidak ditampilkan), yang memberikan akses ke `Bucket-1`. `Job-2` menggunakan `Role-2` (tidak ditampilkan), yang memberikan akses ke `Bucket-2`.

Tugas ini memiliki path jaringan yang memungkinkan mereka untuk berkomunikasi dengan klaster masing-masing dan dengan demikian mengakses data masing-masing. Misalnya, `Job-2` dapat

mengakses data di `Bucket-1`. Dalam diagram tersebut, hal ini ditunjukkan sebagai path warna merah.

Untuk mencegah situasi ini, kami sarankan Anda melampirkan konfigurasi keamanan yang berbeda untuk `Job-1` dan `Job-2`. Dengan melampirkan konfigurasi keamanan, akses lintas-tugas ke data diblokir berdasarkan sertifikat yang dibuat oleh AWS Glue. Konfigurasi keamanan dapat berupa konfigurasi dummy. Artinya, Anda dapat membuat konfigurasi keamanan tanpa mengaktifkan enkripsi data Amazon S3, data CloudWatch Amazon, atau bookmark pekerjaan. Ketiga opsi enkripsi tersebut dapat dinonaktifkan.

Untuk informasi tentang konfigurasi keamanan, lihat the section called “Mengenkripsi data yang ditulis oleh AWS Glue”.

Melampirkan konfigurasi keamanan pada sebuah tugas

1. Buka konsol AWS Glue di https://console.aws.amazon.com/glue/.
2. Pada halaman Mengkonfigurasi properti tugas untuk tugas tersebut, perluas bagian Konfigurasi keamanan, perpustakaan skrip, dan parameter tugas.
3. Pilih sebuah konfigurasi keamanan dalam daftar.

Riwayat dokumentasi untuk AWS Glue

Perubahan	Deskripsi	Tanggal
Support untuk profil AWS Glue penggunaan	Admin dapat membuat profil AWS Glue penggunaan untuk berbagai kelas pengguna dalam akun, seperti pengembang, penguji, dan tim produk. Fleksibilitas ini memungkinkan administrator untuk menerapkan penggunaan dan kontrol biaya yang berbeda untuk setiap kelas pengguna. Untuk informasi selengkapnya, lihat Menyiapkan profil AWS Glue penggunaan.	Juni 18, 2024
Support untuk konektor Salesforce untuk AWS Glue Spark	Menambahkan informasi tentang AWS Glue konektor baru untuk Salesforce. Fitur ini memungkinkan Anda menggunakan Spark AWS Glue untuk membaca dan menulis ke Salesforce di AWS Glue versi 4.0 dan yang lebih baru. Untuk informasi selengkapnya, lihat Menghubungkan ke Salesforce.	22 Mei 2024
Integrasi data Amazon Q di AWS Glue (GA)	Integrasi data Amazon Q AWS Glue adalah kemampuan AI generatif baru AWS Glue yang memungkinkan insinyur	April 30, 2024

data dan pengembang ETL untuk membangun pekerjaan integrasi data menggunakan bahasa alami. Insinyur dan pengembang dapat meminta Q ke pekerjaan penulis, memecahkan masalah dan menjawab pertanyaa n tentang AWS Glue dan integrasi data. Untuk informasi selengkapnya, lihat Integrasi data Amazon Q di AWS Glue. Fitur ini mencakup pembaruan ke`AwsGlueSe ssionUserRestricte dPolicy` ,`AwsGlueSe ssionUserRestricte dNotebookServiceRo le` , dan kebijakan `AwsGlueSessionUser RestrictedServiceR ole` AWS terkelola. Untuk informasi selengkapnya, lihat AWS Glue pembaruan kebijakan AWS terkelola.

Integrasi data Amazon Q di AWS Glue (pratinjau)	Integrasi data Amazon Q AWS Glue adalah kemampuan AI generatif baru AWS Glue yang memungkinkan insinyur data dan pengembang ETL untuk membangun pekerjaan integrasi data menggunakan bahasa alami. Insinyur dan pengembang dapat meminta Q ke pekerjaan penulis, memecahkan masalah dan menjawab pertanyaan tentang AWS Glue dan integrasi data. Untuk informasi selengkap nya, lihat Integrasi data Amazon Q di AWS Glue. Fitur ini mencakup pembaruan pada kebijakan `AwsGlueSessionUserRestrictedNotebookPolicy` AWS terkelola. Untuk informasi selengkapnya, lihat AWS Glue pembaruan kebijakan AWS terkelola.	Januari 30, 2024
Perbarui ke dokumentasi untuk AWS Glue Streaming	Menambahkan babak baru dengan konten baru dan direorganisasi untuk AWS Glue Streaming. Konten ini menjelaskan cara kerja streaming AWS Glue, karakteristik pemrosesan data waktu nyata, dan cara memantau pekerjaan streaming Anda. Untuk informasi selengkapnya, lihat AWS Glue Streaming.	27 Desember 2023

Support untuk menggunakan deteksi data sensitif berbutir halus	Transformasi Detect Sensitive Data menyediakan kemampuan untuk mendeteksi, menutupi, atau menghapus entitas yang Anda tentukan, atau yang telah ditentukan sebelumnya olehAWS Glue. Tindakan berbutir halus lebih lanjut memungkinkan Anda menerapkan tindakan spesifik per entitas. Untuk informasi selengkapnya, lihat Menggunakan deteksi data sensitif berbutir halus.	26 November 2023
Support untuk memantau pekerjaan dengan metrik AWS Glue Observability	Gunakan metrik AWS Glue Observability untuk menghasilkan wawasan tentang apa yang terjadi di dalam pekerjaan Apache Spark Anda AWS Glue untuk meningkatkan triaging dan analisis masalah. Untuk informasi selengkapnya, lihat Metrik Monitoring with AWS Glue Observability.	26 November 2023

Support untuk deteksi anomali dalam Kualitas Data AWS Glue	AWS GlueDeteksi anomali Kualitas Data menerapkan algoritma pembelajaran mesin (ML) pada statistik data dari waktu ke waktu untuk mendeteksi pola abnormal dan masalah kualitas data tersembunyi yang sulit dideteksi melalui aturan. Untuk informasi lebih lanjut, lihat Deteksi anomali dalam Kualitas AWS Glue Data.	26 November 2023
Perbarui ke perilaku logging UI Spark default	Pekerjaan percikan yang menghasilkan log UI Spark sekarang akan menulis dengan pola nama file yang berbeda untuk mendukung Spark UI di konsol. AWS Glue Ini tidak mengubah perilaku CloudWatch log. Anda dapat kembali ke perilaku lama dengan memperbarui konfigurasi pekerjaan Anda. Untuk informasi selengkapnya, lihat Memantau pekerjaan menggunakan UI web Apache Spark.	17 November 2023

Support untuk sumber data baru di AWS Glue untuk Spark	Koneksi ke Amazon OpenSearch Service, Azure SQL, Azure Cosmos untuk NoSQL, SAP HANA Teradata Vantage dan Vertica sekarang didukung secara native di dalamnya. AWS Glue Selain itu, koneksi ke sumber data ini, bersama dengan MongoDB, sekarang tersedia untuk digunakan di AWS Glue editor visual Studio. Untuk informasi selengkapnya, lihat Jenis dan opsi koneksi untuk ETL di AWS Glue untuk Spark untuk informasi tentang AWS Glue dukungan Spark dan Menambahkan AWS Glue koneksi untuk informasi tentang penggunaan di editor visual AWS Glue Studio.	17 November 2023
Support untuk menghasilkan statistik kolom	Anda dapat menghitung statistik tingkat kolom untuk AWS Glue Data Catalog tabel dalam format data seperti Parket, ORC, JSON, ION, CSV, dan XMLtanpa menyiapkan pipeline data tambahan. Untuk informasi selengkapnya, lihat Bekerja dengan statistik kolom.	16 November 2023

Support untuk pemadatan data untuk tabel Iceberg	Untuk kinerja pembacaan yang lebih baik oleh layanan AWS analitik seperti Amazon Athena dan Amazon EMR, dan pekerjaan AWS Glue ETL, Katalog Data menyediakan pemadatan terkelola (proses yang memadatkan objek Amazon S3 kecil menjadi objek yang lebih besar) untuk tabel Iceberg di Katalog Data. Untuk informasi selengkapnya, lihat Mengoptimalkan tabel Gunung Es.	13 November 2023
Perbarui ke perilaku tunggu jalankan pekerjaan	Pekerjaan shell Spark dan Python standar sekarang akan beralih `WAITING` ke situasi tertentu, daripada segera beralih ke. `FAILED` Untuk informasi selengkapnya, lihat status menjalankan AWS Glue pekerjaan.	8 November 2023
AWS Glue Studiopanduan pengguna dikonsolidasikan ke dalam panduan AWS Glue pengembang	Panduan AWS Glue Studio pengguna telah dipindahk an ke panduan pengembang untuk membuat satu panduan pengguna terpadu untukAWS Glue Studio, AWS Glue konsol, dan akses AWS Glue Studio terprogram.	25 Oktober 2023

Memperbarui ke kebijakan AWSGlueServiceNotebookRole AWS terkelola	Menambahkan informasi tentang pembaruan kecil pada kebijakan AWSGlueServiceNotebookRole AWS terkelola. Untuk informasi selengkapnya, lihat AWS GluePembaruan Kebijakan AWS Terkelola.	9 Oktober 2023
AWS Glue Studiomendukung lima transformasi bawaan baru	AWS Glue Studiomendukung lima transformasi bawaan baru berikut: Rekam pencocokan, Hapus baris nol, Parse kolom JSON, Ekstrak jalur JSON, dan ekstraktor Regex. Untuk informasi selengkapnya, lihat Mengedit node transformasi data AWS Glue terkelola.	11 Agustus 2023
Memperbarui ke kebijakan AWSGlueServiceRole AWS terkelola	Menambahkan informasi tentang pembaruan kecil pada kebijakan AWSGlueServiceRole AWS terkelola. Untuk informasi selengkapnya, lihat AWS GluePembaruan Kebijakan AWS Terkelola.	4 Agustus 2023
Support untuk merangkak tabel Apache Hudi	Menambahkan informasi tentang penggunaan AWS Glue untuk merayapi tabel Hudi di bucket Amazon S3 dan mendaftarkan tabel Hudi ke. AWS Glue Data Catalog Untuk informasi selengkapnya, lihat Penyimpanan data mana yang dapat saya jelajahi? , dan properti Crawler.	21 Juli 2023

Memperbarui ke kebijakan AWSGlueConsoleFullAccess AWS terkelola	Menambahkan informasi tentang pembaruan kecil pada kebijakan AWSGlueConsoleFullAccess AWS terkelola. Untuk informasi selengkapnya, lihat AWS GluePembaruan Kebijakan AWS Terkelola.	14 Juli 2023
Support untuk merangkak tabel Apache Iceberg	Menambahkan informasi tentang penggunaan AWS Glue untuk merayapi tabel Iceberg di bucket Amazon S3 dan mendaftarkan tabel Iceberg ke. AWS Glue Data Catalog Untuk informasi selengkapnya, lihat Penyimpanan data mana yang dapat saya jelajahi? , dan properti Crawler.	7 Juli 2023
Support untuk AWS Glue dengan Ray	Menambahkan informasi tentang AWS Glue Ray, mesin baru yang dapat mendukung AWS Glue pekerjaan. Reorganisasi yang ada AWS Glue dengan konten Spark untuk disambiguasi.	30 Mei 2023

Support untuk Kualitas AWS Glue Data (GA)	AWS Glue Kualitas data sekarang tersedia secara umum. AWS Glue Kualitas Data membantu Anda mengevaluasi dan memantau kualitas data Anda. Untuk informasi tentang cara menggunakan Kualitas AWS Glue Data dengan Katalog Data, lihat Kualitas AWS Glue Data. Untuk mempelajari tentang Kualitas AWS Glue Data AWS Glue Studio, lihat Mengevaluasi kualitas data dengan AWS Glue Studio.	24 Mei 2023
Support untuk tipe pekerja yang lebih besar untuk pekerjaan Apache Spark	Support sekarang tersedia untuk penggunaan tipe `G.4X` dan `G.8X` pekerja untuk pekerjaan Apache Spark. Jenis pekerja ini sesuai untuk pekerjaan yang beban kerjanya berisi transformasi, agregasi, gabungan, dan kueri Anda yang paling menuntut. Untuk informasi selengkapnya, lihat Menambahkan lowongan di AWS Glue.	8 Mei 2023

Support untuk membuat indeks partisi saat merayapi tabel	Menambahkan informasi tentang bagaimana crawler mendukung pembuatan indeks partisi untuk tabel yang terdeteksi crawler. Untuk informasi selengkapnya, lihat Mengatur opsi konfigurasi crawler indeks partisi.	24 April 2023
Support untuk metrik penggunaan sumber daya	Menambahkan informasi tentang melihat penggunaan sumber daya layanan dan mengonfigurasi alarm di Amazon. CloudWatch Untuk informasi selengkapnya, lihat pemantauan AWS Glue sumber daya.	April 7, 2023
Memperbarui ke kebijakan AWSGlueConsoleFullAccess AWS terkelola	Menambahkan informasi tentang pembaruan kecil pada kebijakan AWSGlueConsoleFullAccess AWS terkelola. Untuk informasi selengkapnya, lihat AWS GluePembaruan Kebijakan AWS Terkelola.	Maret 28, 2023

Menambahkan panduan untuk menggunakan AWS Glue dengan AWS SDK dengan contoh	Panduan AWS Glue Pengembang memiliki dua bagian baru yang menyediakan informasi untuk membantu Anda menggunakan AWS Glue AWS SDK. Untuk informasi selengkapnya, lihat contoh Menggunakan AWS Glue dengan AWS SDK dan Kode untuk AWS Glue menggunakan AWS SDK.	23 Februari 2023
Perbarui ke dokumentasi untuk IAM dengan AWS Glue	Menata ulang dan menambahkan informasi tentang penggunaan IAM dengan. AWS Glue Untuk informasi selengkapnya, lihat Identitas dan manajemen akses untuk AWS Glue.	15 Februari 2023
Support untuk menjalankan pekerjaan streaming ETL di AWS Glue versi 4.0	Menambahkan informasi tentang dukungan untuk menjalankan pekerjaan ETL streaming di Glue versi 4.0, dan opsi baru untuk menghubungkan ke cluster Kafka atau cluster Amazon Managed Streaming for Apache Kafka, dan Amazon Kinesis Data Streams. Untuk informasi selengkapnya, lihat Menambahkan Pekerjaan ETL Streaming di AWS Glue dan Jenis dan opsi Koneksi untuk ETL di. AWS Glue	Februari 8, 2023

Support untuk merayapi sumber data MongoDB Atlas	Menambahkan informasi tentang penggunaan AWS Glue untuk merayapi sumber data MongoDB Atlas. Untuk informasi selengkapnya, lihat Penyimpanan data mana yang dapat saya jelajahi? , properti koneksi MongoDB dan MongoDB Atlas, dan Menggunakan koneksi MongoDB atau MongoDB Atlas.	6 Februari 2023

Support untuk merangkak tabel Delta Lake menggunakan konektor Delta Lake asli	Menambahkan informasi tentang penggunaan AWS Glue untuk merayapi tabel Delta Lake menggunakan konektor Delta Lake asli. Fitur ini memungkinkan Anda untuk menggunakan mesin AWS kueri untuk menanyakan log transaksi Delta secara langsung dan menggunakan fitur seperti perjalanan waktu dan jaminan ACID, dan untuk menyinkronkan metadata Delta Lake Anda dari file transaksi Amazon S3 ke dalam Katalog Data untuk mengaktifkan izin kolom pada kueri Anda di Lake Formation. Untuk informasi selengkapnya, lihat Cara menentukan opsi konfigurasi untuk penyimpanan data Delta Lake, dan Menanyakan tabel Danau Delta.	Desember 15, 2022

Support untuk Kualitas AWS Glue Data (pratinjau)	Support sekarang tersedia untuk Kualitas AWS Glue Data (pratinjau). AWS Glue Kualitas Data membantu Anda mengevaluasi dan memantau kualitas data Anda saat menggunakan AWS Glue 3.0. Untuk informasi tentang cara menggunakan Kualitas AWS Glue Data dengan Katalog Data, lihat Kualitas AWS Glue Data (pratinjau). Untuk mempelajari tentang Kualitas AWS Glue Data AWS Glue Studio, lihat Mengevaluasi kualitas data dengan AWS Glue Studio.	30 November 2022
Support untuk konektor Amazon Redshift Spark baru dengan fitur baru dan peningkatan kinerja	Support sekarang tersedia untuk konektor Amazon Redshift Spark baru dengan driver JDBC baru untuk digunakan dengan pekerjaan AWS Glue ETL untuk membangun aplikasi Apache Spark yang membaca dari dan menulis ke data di Amazon Redshift sebagai bagian dari pipa konsumsi dan transformasi data Anda. Untuk informasi selengkapnya, lihat Memindahkan data ke dan dari Amazon Redshift.	29 November 2022

Support untuk AWS Glue versi 4.0.	Menambahkan informasi tentang dukungan untuk AWS Glue versi 4.0. Fitur termasuk dukungan asli untuk kerangka kerja danau data terbuka dengan Apache Hudi, Delta Lake, dan Apache Iceberg, dan dukungan asli untuk Plugin Penyimpan an Cloud Shuffle berbasis Amazon S3 (plugin Apache Spark) untuk menggunakan Amazon S3 untuk mengocokk an dan kapasitas penyimpan an elastis. Untuk informasi selengkapnya, lihat Catatan AWS Glue Rilis dan Memigrasi AWS Glue pekerjaan ke AWS Glue versi 4.0.	28 November 2022
AWS Glue Studiosekarang menawarkan transformasi visual khusus	Transformasi visual khusus memungkinkan pelanggan mendefinisikan, menggunak an kembali, dan berbagi logika ETL khusus bisnis di antara tim mereka. Untuk informasi selengkapnya, lihat Transform asi visual khusus.	28 November 2022

Support untuk menggunakan AWS Glue crawler untuk mempublikasikan metadata untuk penyimpanan data JDBC	Support sekarang tersedia untuk menggunakan AWS Glue crawler untuk mempublikasikan metadata seperti komentar dan rawtypes ke Katalog Data untuk penyimpanan data JDBC. Untuk informasi selengkapnya, lihat Parameter yang disetel pada tabel Katalog Data menurut crawler, properti Crawler, dan JdbcTarget struktur.	18 November 2022
Support untuk merayapi penyimpanan data Snowflake	Support sekarang tersedia untuk digunakan AWS Glue untuk merayapi tabel dan tampilan Snowflake, dan untuk mempublikasikan metadata ke Katalog Data sebagai entri tabel. Untuk tabel eksternal Snowflake di Amazon S3, crawler juga merayapi lokasi Amazon S3 dan jenis format file tabel eksternal dan mengisi sebagai parameter Tabel. Untuk informasi selengkapnya, lihat Penyimpanan data mana yang dapat saya jelajahi? , properti AWS Glue koneksi, dan Parameter diatur pada tabel Katalog Data oleh crawler.	18 November 2022

Support untuk manajemen shuffle yang lebih baik dari aplikasi Spark Anda	Support sekarang tersedia untuk Plugin Cloud Shuffle Storage baru untuk Apache Spark. Untuk informasi selengkapnya, lihat Plugin AWS Glue Spark shuffle dengan Amazon S3 dan Cloud Shuffle Storage Plugin untuk Apache Spark.	15 November 2022
Menambahkan dukungan untuk target Katalog Data saat mempercepat pemberitahuan peristiwa crawl Amazon S3	Selain dukungan yang ada untuk target Amazon S3, dukungan sekarang tersedia untuk mempercepat crawl untuk target Katalog Data menggunakan pemberita huan peristiwa. Amazon S3 Untuk informasi selengkap nya, lihat Mempercepat Crawl Menggunakan Pemberitahuan Amazon S3 Acara.	13 Oktober 2022
Support untuk menentukan jumlah maksimum tabel yang dapat dibuat oleh crawler	Support sekarang tersedia untuk menentukan jumlah maksimum tabel yang diizinkan untuk dibuat oleh crawler. Untuk informasi selengkapnya, lihat Cara menentukan jumlah maksimum tabel yang diizinkan untuk dibuat oleh crawler.	September 6, 2022

Dukungan untuk Python 3.9 dalam pekerjaan shell Python di AWS Glue	Support sekarang tersedia untuk menjalankan skrip yang kompatibel dengan Python 3.9 dalam AWS Glue pekerjaan shell Python di, dan untuk memilih untuk menggunakan kumpulan pustaka pra-paket. Untuk informasi selengkapnya, lihat pekerjaan shell Python di. AWS Glue	Agustus 11, 2022
Support untuk menjalankan AWS Glue pekerjaan yang tidak mendesak atau tidak sensitif terhadap waktu dengan kapasitas cadangan	Support sekarang tersedia untuk konfigurasi pekerjaan fleksibel untuk pekerjaan yang tidak mendesak seperti pekerjaan pra-produksi, pengujian, dan pemuatan data satu kali. Untuk informasi selengkapnya, lihat Menambahkan lowongan di AWS Glue.	Agustus 9, 2022
Support untuk tipe pekerja baru untuk pekerjaan streaming	Support sekarang tersedia untuk penggunaan tipe `G.025X` pekerja untuk pekerjaan streaming volume rendah. Untuk informasi selengkapnya, lihat Menambahkan lowongan di AWS Glue.	14 Juli 2022

Support untuk penggunaan Kafka SASL dalam koneksi AWS Glue	Support sekarang tersedia untuk penggunaan Kafka SASL dalam AWS Glue koneksi. Untuk informasi selengkapnya, lihat Properti koneksi AWS Glue Kafka untuk otentikasi klien.	Juli 5, 2022
Support untuk konektor Apache kafka untuk skema protobuf	Support sekarang tersedia untuk Apache Kafka Connector untuk skema Protobuf. Untuk informasi selengkapnya, lihat Skema Registri AWS Glue.	9 Juni 2022
Support untuk Auto Scaling untuk AWS Glue pekerjaan (GA)	Menambahkan informasi tentang penggunaan Auto Scaling untuk pekerjaan di AWS Glue versi 3.0 untuk menskalakan sumber daya komputasi secara dinamis. Untuk informasi selengkap nya, lihat Menggunakan Auto Scaling untuk. AWS Glue	April 14, 2022
Perbarui dokumentasi untuk AWS Glue mengembangkan dan menguji skrip AWS Glue pekerjaan	Menata ulang dan menambahkan informasi tentang metode pengemban gan dan pengujian yang tersedia untukAWS Glue, termasuk instruksi untuk mengembangkan dengan Docker. Untuk informasi selengkapnya, lihat Mengembangkan dan menguji skrip AWS Glue pekerjaan.	Maret 14, 2022

Penambahan buffer protokol (protobuf) sebagai format data yang didukung untuk registri skema AWS Glue	Menambahkan informasi tentang Protobuf sebagai format data yang didukung (selain AVRO dan JSON). Untuk informasi selengkapnya, lihat Skema Registri AWS Glue.	25 Februari 2022
Support untuk merangkak tabel Delta Lake	Menambahkan informasi tentang penggunaan AWS Glue untuk merayapi tabel Delta Lake. Untuk informasi selengkapnya, lihat Cara menentukan opsi konfigurasi untuk penyimpanan data Delta Lake.	Februari 24, 2022
Support untuk wawasan AWS Glue pekerjaan	Menambahkan informasi tentang penggunaan wawasan AWS Glue pekerjaan untuk menyederhanakan debugging dan pengoptimalan pekerjaan untuk pekerjaan Anda. AWS Glue Untuk informasi selengkapnya, lihat Memantau dengan wawasan AWS Glue pekerjaan.	8 Februari 2022

Support untuk merayapi tabel Katalog Data yang didukung Amazon S3 menggunakan titik akhir VPC	Selain penyimpanan data Amazon S3, Anda dapat mengonfigurasi tabel Katalog Data yang didukung Amazon S3 agar hanya diakses oleh lingkungan Amazon Virtual Private Cloud (Amazon VPC), untuk tujuan keamanan, audit, atau kontrol. Untuk informasi selengkapnya, lihat Merayapi tabel Katalog Data yang didukung Amazon S3 atau Amazon S3 menggunakan Titik Akhir VPC.	3 Februari, 2022
Dukungan untuk tabel yang diatur Lake Formation	Menambahkan informasi tentang AWS Glue dukungan untuk tabel yang diatur Lake Formation, yang mendukung transaksi ACID, pemadatan data otomatis, dan kueri perjalanan waktu. Untuk informasi selengkapnya, lihat AWS GlueAPI dan panduan AWS Lake Formation pengembang.	30 November 2021
Kebijakan AWS terkelola baru ditambahkan untuk sesi interaktif dan buku catatan	Kebijakan terkelola baru untuk IAM memberikan keamanan yang ditingkatkan untuk digunakan AWS Glue dengan sesi interaktif dan notebook. Untuk informasi selengkapnya, lihat Kebijakan AWS Terkelola untuk AWS Glue.	30 November 2021

Registri skema Glue sekarang didukung dengan pekerjaan streaming	Anda dapat membuat pekerjaan streaming yang mengakses tabel yang merupakan bagian dari Glue Schema Registry. Untuk informasi lebih lanjut, lihat Registri AWS Glue Skema dan Menambahkan Pekerjaan ETL Streaming di. AWS Glue	15 November 2021
Support untuk fitur machine learning baru	Menambahkan informasi tentang fitur baru untuk transformasi pembelajaran mesin Find match, termasuk pencocokan inkremental dan skor kecocokan. Untuk informasi selengkapnya, lihat Menemukan Kecocokan Tambahan dan Memperkirakan Kualitas Pertandingan menggunakan Skor Keyakinan Pertandingan.	Oktober 31, 2021
(Pratinjau pribadi) Dukungan untuk AWS Glue pekerjaan fleksibel	Menambahkan informasi tentang mengonfigurasi pekerjaan AWS Glue Spark dengan kelas eksekusi yang fleksibel, sesuai untuk pekerjaan yang tidak sensitif terhadap waktu yang waktu mulai dan penyelesaiannya dapat bervariasi. Untuk informasi selengkapnya, lihat Menambahkan Tugas di AWS Glue.	29 Oktober 2021

Support untuk mempercepat crawl menggunakan pemberitahuan acara Amazon S3	Menambahkan informasi tentang mempercepat crawl menggunakan pemberitahuan Amazon S3 acara. Untuk informasi selengkapnya, lihat Mempercepat Crawl Menggunakan Pemberitahuan Amazon S3 Acara.	Oktober 15, 2021
Opsi konfigurasi keamanan tambahan yang terkait dengan kontrol akses dan VPC	Menambahkan informasi tentang bagaimana Anda dapat mengonfigurasi izin kontrol akses baru AWS Glue dan konfigurasi VPC. Untuk informasi selengkapnya, lihat AWSTag di AWS Glue, Kebijakan Berbasis Identitas (Kebijakan IAM) yang Mengontrol Pengaturan Menggunakan Kunci Kondisi atau Kunci Konteks, dan Mengonfigurasi semua AWS panggilan untuk melalui VPC Anda.	13 Oktober 2021
Dukungan untuk kebijakan titik akhir VPC	Menambahkan informasi tentang dukungan untuk kebijakan titik akhir Virtual Private Cloud (VPC) di. AWS Glue Untuk informasi selengkapnya, lihat AWS Gluedan antarmuka titik akhir VPC ().AWS PrivateLink	11 Oktober 2021
Glue Studio sekarang tersedia di China	AWS Glue Studiosekarang tersedia di wilayah China Beijing dan Ningxia.	11 Oktober 2021

AWS Glue Studiomenawarkan penulisan buku catatan, untuk pengeditan pekerjaan interaktif	Notebook membantu Anda menulis dan mengeksekusi kode, memvisualisasikan hasilnya, dan berbagi wawasan. Biasanya, ilmuwan data menggunakan notebook untuk eksperimen dan tugas eksplorasi data. Untuk informasi selengkapnya, lihat Menggunakan Notebook.	1 Oktober 2021
Akses langsung ke sumber streaming sekarang tersedia	Saat menambahkan sumber data ke pekerjaan ETL Anda di editor visual, Anda dapat memberikan informasi untuk mengakses aliran data alih-alih harus menggunakan database dan tabel Katalog Data.	30 September 2021
Mendokumentasikan kebijakan dukungan AWS Glue versi	Menambahkan informasi tentang kebijakan dukungan AWS Glue versi dan fase akhir masa pakai untuk AWS Glue versi tertentu. Untuk informasi selengkapnya, lihat kebijakan dukungan AWS Glue versi.	24 September 2021
Konektor kustom sekarang dapat digunakan dengan pratinjau data	Saat mengedit simpul sumber data menggunakan konektor khusus, Anda dapat melihat pratinjau kumpulan data dengan memilih tab pratinjau Dat. Untuk informasi selengkapnya, lihat Konektor Kustom.	24 September 2021

Support untuk sesi AWS Glue interaktif (pratinjau pribadi)	(Pratinjau pribadi) Menambahkan informasi tentang penggunaan sesi AWS Glue interaktif untuk menjalankan beban kerja Spark di cloud dari Notebook Jupyter apa pun. Sesi interaktif adalah metode yang disukai untuk mengembangkan kode AWS Glue ekstrak, transformasi, dan muat (ETL) Anda saat Anda menggunakan AWS Glue 2.0 atau yang lebih baru. Untuk informasi selengkapnya, lihat Menyiapkan dan Menjalankan sesi AWS Glue interaktif untuk Jupyter Notebook.	Agustus 24, 2021
Support untuk membuat alur kerja dari blueprints (GA)	Menambahkan informasi tentang pengkodean kasus penggunaan ekstrak, transformasi, dan beban (ETL) umum dalam cetak biru dan kemudian membuat alur kerja dari cetak biru. Memungkinkan analis data untuk dengan mudah membuat dan menjalankan proses-proses ETL yang kompleks. Untuk informasi selengkapnya, lihat Melakukan Aktivitas ETL Kompleks Menggunakan cetak biru dan Alur Kerja di. AWS Glue	23 Agustus 2021

Support untuk AWS Glue versi 3.0.	Menambahkan informasi tentang dukungan untuk AWS Glue versi 3.0 yang mendukung peningkatan mesin Apache Spark 3.0 untuk menjalankan pekerjaan Apache Spark ETL, dan pengoptimalan dan peningkatan lainnya. Untuk informasi selengkapnya, lihat Catatan AWS Glue Rilis dan Memigrasi AWS Glue pekerjaan ke AWS Glue versi 3.0. Fitur lain dalam rilis ini termasuk manajer AWS Glue acak, pembaca CSV vektor SIMD, dan predikat partisi katalog. Untuk informasi selengkapnya, lihat AWS GlueSpark shuffle manager dengan Amazon S3, Opsi Format untuk Input dan Output ETL, dan Pemfilteran sisi server menggunakan predikat AWS Glue partisi katalog.	18 Agustus 2021
AWS GovCloud (US) Region	AWS Glue Studiosekarang tersedia di AWS GovCloud (US) Region	18 Agustus 2021

Penulisan shell Python tersedia di AWS Glue Studio	Saat membuat pekerjaan baru, Anda sekarang dapat memilih untuk membuat pekerjaan shell Python. Untuk informasi selengkapnya, lihat Memulai proses pembuatan pekerjaan dan Mengedit pekerjaan shell Python di. AWS Glue Studio	13 Agustus 2021
Support untuk memulai alur kerja dengan acara Amazon EventBridge	Menambahkan informasi tentang cara AWS Glue dapat menjadi konsumen peristiwa dalam arsitektur didorong-peristiwa. Untuk informasi selengkapnya, lihat Memulai AWS Glue Alur Kerja dengan EventBridge Acara Amazon dan Melihat EventBridge Peristiwa yang Memulai Alur Kerja.	14 Juli 2021
Penambahan JSON sebagai format data yang didukung untuk registri AWS Glue skema	Menambahkan informasi tentang JSON sebagai sebuah format data yang didukung (selain AVRO). Untuk informasi selengkap nya, lihat Skema Registri AWS Glue.	30 Juni 2021

Buat pekerjaan AWS Glue streaming tanpa tabel Katalog Data	Fungsi Python `create_data_frame_from_options` atau `getSource` untuk skrip Scala mendukung pembuatan tugas ETL streaming yang me-referensi aliran data secara langsung alih-alih mengharuskan tabel Katalog Data.	15 Juni 2021
AWS Gluetransformasi pembelajaran mesin sekarang mendukung kunci AWS Key Management Service	Anda dapat menentukan konfigurasi atau AWS KMS kunci keamanan saat mengonfigurasi transformasi AWS Glue Machine Learning dengan konsol, CLI, atau API. AWS Glue Untuk informasi selengkapnya, lihat Menggunakan Enkripsi Data dengan Transformasi Machine Learning dan API Machine Learning AWS Glue.	15 Juni 2021
Memperbarui ke kebijakan AWSGlueConsoleFullAccess AWS terkelola	Menambahkan informasi tentang pembaruan kecil pada kebijakan AWSGlueConsoleFullAccess AWS terkelola. Untuk informasi selengkapnya, lihat AWS GluePembaruan Kebijakan AWS Terkelola.	10 Juni 2021

Melihat kumpulan data pekerjaan Anda saat membuat dan mengedit pekerjaan	Anda dapat menggunakan tab Pratinjau data untuk simpul dalam diagram tugas Anda untuk melihat contoh data yang diproses oleh simpul tersebut. Untuk informasi selengkapnya, lihat Menggunakan pratinjau data dalam editor tugas visual.	7 Juni 2021
Support untuk menentukan nilai yang menunjukkan lokasi tabel untuk output crawler.	Penambahan informasi tentang penentuan sebuah nilai yang menunjukkan lokasi tabel saat mengkonfigurasi output crawler. Untuk informasi selengkapnya, lihat Cara menentukan lokasi tabel.	4 Juni 2021
Support untuk merayapi sampel file dalam kumpulan data saat merayapi penyimpanan data Amazon S3	Menambahkan informasi tentang cara melakukan perayapan pada contoh file saat melakukan perayapan pada Amazon S3. Untuk informasi selengkapnya, lihat Properti Crawler.	10 Mei 2021

Support untuk penulis parket yang AWS Glue dioptimalkan	Menambahkan informasi tentang menggunakan penulis parket yang AWS Glue dioptimalkan DynamicFr ames untuk membuat atau memperbarui tabel dengan `parquet` klasifikasi. Untuk informasi selengkapnya, lihat Membuat Tabel, Memperbar ui Skema, dan Menambahk an Partisi Baru dalam Katalog Data dari Tugas ETL AWS Glue dan Pilihan Format untuk Input dan Output ETL di AWS Glue.	4 Mei 2021
Support untuk password otentikasi klien kafka	Penambahan informasi tentang bagaimana tugas ETL streaming dalam AWS Glue mendukung autentika si sertifikat klien SSL dengan produsen pengaliran Apache Kafka. Anda sekarang dapat memberikan sebuah sertifika t kustom sementara yang menentukan koneksi AWS Glue ke klaster Apache Kafka, yang akan digunakan oleh AWS Glue melakukan ketika autentikasi dengannya. Untuk informasi selengkapnya, lihat Properti Koneksi AWS Glue dan API Koneksi.	28 April 2021

Support untuk mengkonsumsi data dari Amazon Kinesis Data Streams di akun lain dalam streaming pekerjaan ETL	Penambahan informasi tentang cara membuat tugas ETL streaming untuk mengkonsumsi data dari Amazon Kinesis Data Streams di akun lain. Untuk informasi selengkapnya, lihat Menambahkan Tugas ETL Streaming di AWS Glue.	30 Maret 2021
Transformasi SQL tersedia	Anda dapat menggunakan simpul transformasi SQL untuk menulis transform asi Anda sendiri dalam bentuk kueri SQL. Untuk informasi selengkapnya, lihat Menggunakan kueri SQL untuk men-transfromasi data.	23 Maret 2021
Support untuk membuat alur kerja dari cetak biru (pratinjau publik)	(Pratinjau publik) Penambaha n informasi tentang pengkodea n umum kasus penggunaa n extract, transform, and load (ETL) dalam cetak biru dan kemudian membuat alur kerja dari cetak biru. Memungkinkan analis data untuk dengan mudah membuat dan menjalank an proses-proses ETL yang kompleks. Untuk informasi selengkapnya, lihat Melakukan Aktivitas ETL Kompleks Menggunakan cetak biru dan Alur Kerja di. AWS Glue	22 Maret 2021

Konektor dapat digunakan untuk target data	Menggunakan kustom atau AWS Marketplace konektor untuk target data Anda sekarang didukung. Untuk informasi selengkapnya, lihat Menulis tugas dengan konektor kustom.	15 Maret 2021
Support untuk metrik kepentingan kolom untuk transformasi pembelajaran AWS Glue mesin	Menambahkan tentang cara melihat metrik nilai penting kolom saat bekerja dengan transformasi machine learning AWS Glue. Untuk informasi selengkapnya, lihat Bekerja dengan Transformasi Machine Learning pada Konsol AWS Glue	5 Februari 2021
Penjadwalan Job sekarang tersedia di AWS Glue Studio	Anda dapat menentukan jadwal berbasis waktu untuk pekerjaan Anda berjalan. AWS Glue Studio Anda dapat menggunakan konsol tersebut untuk membuat jadwal dasar, atau menentukan jadwal yang lebih kompleks menggunak an sintaksis cron mirip-Unix. Untuk informasi selengkapnya, lihat Menjadwalkan eksekusi tugas.	21 Desember 2020

AWS GlueKonektor Kustom dirilis	Konektor Kustom AWS Glue memungkinkan Anda menemukan dan berlangga nan konektor di AWS Marketplace. Kami juga merilis antarmuka waktu aktif Spark AWS Glue untuk mencolokkan konektor yang dibangun untuk Sumber Data Apache Spark, kueri gabungan Athena, dan API JDBC. Untuk informasi selengkapnya, lihat Menggunakan Konektor dan koneksi dengan AWS Glue Studio.	21 Desember 2020
Support untuk menjalankan pekerjaan streaming ETL di AWS Glue versi 2.0	Penambahan informasi tentang support untuk menjalankan streaming tugas ETL di Glue versi 2.0. Untuk informasi selengkapnya, lihat Menambahkan Tugas ETL Streaming di AWS Glue.	18 Desember 2020
Support untuk partisi beban kerja dengan eksekusi terbatas	Penambahan informasi tentang memungkinkan beban kerja pemartisian untuk mengkonfigurasi batas atas pada ukuran set data, atau jumlah file yang diproses pada eksekusi tugas ETL. Untuk informasi selengkapnya, lihat Pemartisian Beban Kerja dengan Eksekusi Terbatas.	23 November 2020

Support untuk manajemen partisi yang disempurnakan	Penambahan informasi tentang cara menggunakan API baru untuk menambah atau menghapus indeks partisi ke/dari tabel yang ada. Untuk informasi selengkapnya, lihat Bekerja dengan Indeks Partisi.	23 November 2020
Support untuk registri AWS Glue skema	Penambahan informasi tentang menggunakan Skema Registri AWS Glue untuk menemukan, mengontrol, dan mengembangkan skema secara terpusat. Untuk informasi selengkapnya, lihat Skema Registri AWS Glue.	19 November 2020
Support untuk format masukan grok dalam streaming pekerjaan ETL	Penambahan informasi tentang menerapkan pola Grok ke sumber streaming seperti file berkas log. Untuk informasi selengkapnya, lihat Menerapkan Pola Grok untuk Sumber Streaming.	17 November 2020
Support untuk menambahkan tag ke alur kerja di konsol AWS Glue	Penambahan informasi tentang cara menambahkan tag saat membuat sebuah alur kerja dengan menggunakan konsol AWS Glue. Untuk informasi selengkapnya, lihat Membuat dan Membangun Alur Kerja Menggunakan Konsol AWS Glue.	27 Oktober 2020

Support untuk proses crawler inkremental	Penambahan informasi tentang support untuk eksekusi crawler tambahan, yang hanya melakukan perayapan pada folder Amazon S3 yang ditambahkan sejak eksekusi terakhir. Untuk informasi selengkapnya, lihat Perayapan Tambahan.	21 Oktober 2020
Dukungan untuk deteksi skema untuk streaming sumber data ETL. dukungan untuk sumber data ETL streaming Avro dan kafka yang dikelola sendiri	Tugas extract, transform, and load (ETL) streaming di AWS Glue sekarang dapat secara otomatis mendeteksi skema catatan masuk dan menangani perubahan skema untuk masing-masing catatan. Sekarang mendukung sumber data Kafka dikelola sendiri. Tugas ETL streaming sekarang mendukung format Avro dalam sumber data. Untuk informasi selengkapnya, lihat ETL Streaming di AWS Glue, Mendefinisikan Properti Tugas untuk Tugas ETL Streaming, dan Catatan dan Pembatasan untuk Sumber Streaming Avro.	7 Oktober 2020

Support untuk merayapi sumber data MongoDB dan DocumentDB	Penambahan informasi tentang support untuk melakukan perayapan pada sumber data MongoDB dan Amazon DocumentDB (dengan kompatibilitas MongoDB). Untuk informasi selengkapnya, lihat Mendefinisikan Crawler.	5 Oktober 2020
Support untuk kepatuhan FIPS	Penambahan informasi tentang titik akhir FIPS bagi pelanggan yang memerlukan modul kriptografi yang divalidasi FIPS 140-2 ketika mengakses data menggunakan AWS Glue. Untuk informasi lebih lanjut, lihat Kepatuhan FIPS.	23 September 2020
AWS Glue Studiomenyediakan antarmuka visual yang mudah digunakan untuk membuat dan memantau pekerjaan	Anda sekarang dapat menggunakan antarmuka berbasis grafik sederhana untuk menyusun tugas yang memindahkan dan men-transformasi data dan menjalankannya di AWS Glue. Anda kemudian dapat menggunakan dasbor job run AWS Glue Studio untuk memantau eksekusi ETL dan memastikan bahwa pekerjaan Anda beroperasi sebagaimana dimaksud. Untuk informasi selengkapnya, lihat Panduan Pengguna AWS Glue Studio.	23 September 2020

Support untuk membuat indeks tabel untuk meningkatkan kinerja kueri	Penambahan informasi tentang membuat indeks tabel untuk memungkinkan Anda untuk mengambil subset dari partisi dari sebuah tabel. Untuk informasi selengkapnya, lihat Bekerja dengan Indeks Partisi.	9 September 2020
Support untuk mengurangi waktu startup saat menjalankan pekerjaan Apache Spark ETL di AWS Glue versi 2.0.	Penambahan informasi tentang support untuk AWS Glue versi 2.0 yang menyediakan infrastruktur yang ditingkatkan untuk menjalankan tugas ETL Apache Spark dengan pengurangan waktu pemulaian, perubahan pada pencatatan log, dan dukungan untuk menentukan modul Python tambahan di tingkat tugas. Untuk informasi selengkapnya, lihat Catatan Rilis AWS Glue dan Menjalankan Tugas ETL Spark dengan Waktu Pemulaian yang Dikurangi.	10 Agustus 2020
Support untuk membatasi jumlah alur kerja yang berjalan bersamaan.	Penambahan informasi tentang cara membatasi jumlah eksekusi alur kerja bersamaan untuk alur kerja tertentu. Untuk informasi selengkapnya, lihat Membuat dan Membangun Alur Kerja Menggunakan Konsol AWS Glue.	10 Agustus 2020

Support untuk merayapi penyimpanan data Amazon S3 menggunakan titik akhir VPC	Penambahan informasi tentang cara mengkonfigurasi penyimpanan data Amazon S3 Anda sehingga diakses hanya oleh lingkungan Amazon Virtual Private Cloud (Amazon VPC), untuk tujuan keamanan, audit, atau kontrol. Untuk informasi selengkapnya, lihat Melakukan Perayapan Penyimpanan Data Amazon S3 dengan Menggunakan VPC Endpoint.	7 Agustus 2020
Support untuk melanjutkan alur kerja berjalan	Penambahan informasi tentang cara melanjutkan eksekusi alur kerja yang hanya sebagian selesai karena satu atau beberapa simpul (tugas atau crawler) tidak berhasil diselesaikan. Untuk informasi selengkapnya, lihat Memperbaiki dan Melanjutkan Eksekusi Alur Kerja.	27 Juli 2020
Support untuk mengaktifkan sertifikat CA pribadi dalam koneksi kafka di. AWS Glue	Penambahan informasi tentang pilihan koneksi baru yang mendukung dimungkin kannya sertifikat CA privat untuk koneksi Kafka di AWS Glue. Untuk informasi selengkapnya, lihat Jenis dan Pilihan Koneksi untuk ETL dalam AWS Glue dan Parameter Khusus yang Digunakan oleh AWS Glue.	20 Juli 2020

Support untuk membaca data DynamoDB di akun lain	Penambahan informasi tentang dukungan AWS Glue untuk membaca data dari tabel DynamoDB akun AWS yang lain. Untuk informasi selengkapnya, lihat Membaca dari Data DynamoDB di Akun Lain.	17 Juli 2020
Support untuk koneksi penulis DynamoDB AWS Glue di versi 1.0 atau yang lebih baru	Penambahan informasi tentang dukungan untuk penulis DynamoDB, dan opsi koneksi baru atau diperbarui untuk DynamoDB untuk baca atau tulis. Untuk informasi selengkapnya, lihat Jenis dan Pilihan Koneksi untuk ETL dalam AWS Glue.	17 Juli 2020
Dukungan untuk tautan sumber daya dan untuk kontrol akses lintas akun menggunakan keduanya AWS Glue dan Lake Formation	Penambahan konten tentang objek Katalog Data baru yang disebut tautan sumber daya, dan tentang cara mengelola pembagian sumber daya Katalog Data di seluruh akun dengan AWS Glue dan AWS Lake Formation. Untuk informasi selengkapnya, lihat Memberikan Akses Lintas Akun dan Tautan Sumber Daya Tabel.	7 Juli 2020

Support untuk rekaman sampling saat merayapi penyimpanan data DynamoDB	Penambahan informasi tentang properti baru yang dapat Anda konfigurasikan saat melakukan perayapan pada sebuah penyimpan an data DynamoDB. Untuk informasi selengkapnya, lihat Properti Crawler.	12 Juni 2020
Support untuk menghentikan proses alur kerja.	Penambahan informasi tentang cara menghentikan eksekusi alur kerja untuk alur kerja tertentu. Untuk informasi selengkapnya, lihat Menghenti kan Eksekusi Alur Kerja.	14 Mei 2020
Support untuk pekerjaan Spark streaming ETL	Penambahan informasi tentang membuat tugas extract, transform, and load (ETL) dengan sumber data streaming. Untuk informasi selengkapnya, lihat Menambahkan Tugas ETL Streaming di AWS Glue.	27 April 2020

Support untuk membuat tabel, memperbarui skema, dan menambahkan partisi baru di Katalog Data setelah menjalankan pekerjaan ETL	Penambahan informasi tentang bagaimana Anda dapat mengaktifkan membuat tabel, memperbarui skema, dan menambahkan partisi baru untuk melihat hasil tugas ETL Anda dalam Katalog Data. Untuk informasi selengkapnya, lihat Membuat Tabel, Memperbarui Skema, dan Menambahkan Partisi Baru dalam Katalog Data Tugas ETL AWS Glue.	2 April 2020
Support untuk menentukan versi untuk format data Apache Avro sebagai input dan output ETL di AWS Glue	Penambahan informasi tentang menentukan versi untuk format data Apache Avro sebagai input dan output ETL di AWS Glue. Versi default 1.7. Anda dapat menggunakan pilihan format `version` untuk menentukan Avro versi 1.8 untuk mengaktifkan baca/tulis logis. Untuk informasi selengkapnya, lihat Format Pilihan untuk Input dan Output ETL di AWS Glue.	31 Maret 2020

Support untuk committer EMRFS S3 yang dioptimalkan untuk menulis data Parket ke Amazon S3	Penambahan informasi tentang cara mengatur bendera baru untuk mengaktif kan pelaksana dioptimalkan-S3 EMRFR untuk menulis data Parquet ke Amazon S3 saat membuat atau memperbar ui tugas AWS Glue. Untuk informasi selengkapnya, lihat Parameter Khusus yang Digunakan oleh AWS Glue.	30 Maret 2020
Support untuk pembelaja ran mesin berubah sebagai sumber daya yang dikelola oleh tag AWS sumber daya	Menambahkan informasi tentang penggunaan tag AWS sumber daya untuk mengelola dan mengontrol akses ke pembelajaran mesin Anda berubah. AWS Glue Anda dapat menetapkan tag AWS sumber daya untuk pekerjaan , pemicu, titik akhir, crawler, dan transformasi pembelaja ran mesin. AWS Glue Untuk informasi selengkapnya, lihat Tag AWS di AWS Glue.	2 Maret 2020
Support untuk argumen pekerjaan yang tidak dapat di-overrideable	Penambahan informasi tentang dukungan untuk parameter tugas khusus yang tidak dapat diganti dalam pemicu atau ketika Anda menjalankan tugas. Untuk informasi selengkapnya, lihat Menambahkan Tugas di AWS Glue.	12 Februari 2020

Support untuk transformasi baru untuk bekerja dengan dataset di Amazon S3	Penambahan informasi tentang transformasi baru (Merge, Purge, dan Transitio n) dan pengecualian kelas penyimpanan Amazon S3 untuk aplikasi Apache Spark untuk bekerja dengan set data di Amazon S3. Untuk informasi selengkapnya tentang dukungan untuk transformasi ini untuk Python, mergeDyna micFramelihat dan Bekerja dengan Kumpulan Data di Amazon S3. Untuk Scala, lihat mergeDynamicFrames dan AWS GlueScala API GlueContext .	16 Januari 2020
Support untuk memperbarui Katalog Data dengan informasi partisi baru dari pekerjaan ETL	Menambahkan informasi tentang cara mengkode skrip ekstrak, mengubah, dan memuat (ETL) untuk memperbarui AWS Glue Data Catalog dengan informasi partisi baru. Dengan kemampuan ini, Anda tidak lagi harus menjalankan ulang crawler setelah tugas selesai untuk melihat partisi baru. Untuk informasi selengkapnya, lihat Memperbarui Katalog Data dengan Partisi Baru.	15 Januari 2020

Tutorial baru: Menggunakan SageMaker notebook	Menambahkan tutorial yang menunjukkan cara menggunakan SageMaker notebook Amazon untuk membantu mengembangkan skrip ETL dan machine learning Anda. Lihat Tutorial: Gunakan SageMaker Notebook Amazon dengan Titik Akhir Pengembangan Anda.	3 Januari 2020
Dukungan untuk membaca dari MongoDB dan Amazon DocumentDB (dengan kompatibilitas MongoDB)	Penambahan informasi tentang jenis koneksi baru dan pilihan koneksi untuk membaca dari dan menulis ke MongoDB dan Amazon DocumentDB (dengan Kompatibilitas MongoDB). Untuk informasi selengkapnya, lihat Jenis dan Pilihan Koneksi untuk ETL dalam AWS Glue.	17 Desember 2019

Berbagai koreksi dan klarifikasi	Penambahan koreksi dan klarifikasi secara keseluruhan. Penghapusan entri dari bab Masalah yang Dikenal. Penambahan peringatan bahwa AWS Glue hanya mendukung kunci utama pelanggan simetris (CMK) ketika menentukan pengaturan enkripsi Katalog Data dan membuat konfigurasi keamanan. Penambahan catatan bahwa AWS Glue tidak mendukung penulisan ke Amazon DynamoDB.	9 Desember 2019
Support untuk driver JDBC kustom	Penambahan informasi tentang menghubungkan ke sumber data dan target dengan driver JDBC yang tidak didukung secara asli oleh AWS Glue, seperti MySQL versi 8 dan Oracle Database versi 18. Untuk informasi selengkapnya, lihat Nilai connectionType JDBC.	25 November 2019

Support untuk menghubungkan SageMaker notebook ke titik akhir pengembangan yang berbeda	Menambahkan informasi tentang bagaimana Anda dapat menghubungkan SageMaker notebook ke titik akhir pengembangan yang berbeda. Pembaruan untuk menjelaskan tindakan konsol baru untuk beralih ke titik akhir pengembangan baru, dan kebijakan SageMaker IAM baru. Untuk informasi selengkapnya, lihat Bekerja dengan Notebook di AWS Glue Konsol dan Membuat Kebijakan IAM untuk Notebook Amazon SageMaker .	21 November 2019
Support untuk AWS Glue versi dalam transformasi pembelajaran mesin	Penambahan informasi tentang cara mendefinisikan versi AWS Glue dalam transformasi machine learning untuk menunjukkan versi AWS Glue mana yang kompatibel dengan sebuah transformasi machine learning. Untuk informasi selengkapnya, lihat Bekerja dengan Transformasi Machine Learning pada Konsol AWS Glue.	21 November 2019

Support untuk memundurkan bookmark pekerjaan Anda	Penambahan informasi tentang cara memutar kembali bookmark tugas Anda ke setiap tugas yang dijalanka n sebelumnya, sehingga tugas berikutnya menjalankan pemrosesan ulang data hanya dari eksekusi tugas yang ditandai. Dijelaskan dua sub-pilihan baru untuk pilihan `job-bookmark-pause` yang memungkinkan Anda untuk menjalankan sebuah tugas antara dua bookmark. Untuk informasi selengkapnya, lihat Melacak Data yang Diproses Menggunakan Bookmark Tugas dan Parameter Khusus yang Digunakan oleh AWS Glue.	22 Oktober 2019
Support untuk sertifikat JDBC kustom untuk menghubungkan ke penyimpanan data	Penambahan informasi tentang dukungan AWS Glue sertifikat JDBC kustom untuk koneksi SSL ke sumber data atau target data AWS Glue. Untuk informasi selengkapnya, lihat Bekerja dengan Koneksi pada Konsol AWS Glue.	10 Oktober 2019

Dukungan untuk roda Python	Penambahan informasi tentang dukungan AWS Glue dari file wheel (bersama dengan file egg) sebagai dependensi untuk tugas shell Python. Untuk informasi selengkapnya, lihat Memberikan Perpustakaan Python Anda Sendiri.	26 September 2019
Support untuk pembuatan versi titik akhir pengembangan di AWS Glue	Penambahan informasi tentang cara mendefinisikan `Glue version` dalam titik akhir pengembangan. `Glue version` menentukan versi Apache Spark dan Python yang didukung oleh AWS Glue. Untuk informasi selengkapnya, lihat Menambahkan Titik Akhir Pengembangan.	19 September 2019
Support untuk monitoring AWS Glue menggunakan Spark UI	Penambahan informasi tentang cara menggunakan Apache Spark UI untuk memantau dan melakukan debug pada tugas ETL AWS Glue yang berjalan pada sistem tugas AWS Glue, dan aplikasi Spark pada AWS Glue titik akhir pengembangan. Untuk informasi selengkapnya, lihat Pemantauan AWS Glue dengan Menggunakan Spark UI.	19 September 2019

Peningkatan dukungan untuk pengembangan skrip ETL lokal menggunakan pustaka ETL publik AWS Glue	Memperbarui konten perpustakaan ETL AWS Glue untuk mencerminkan bahwa AWS Glue versi 1.0 sekarang sudah didukung. Untuk informasi selengkap nya, lihat Mengembangkan dan Menguji Skrip ETL Secara Lokal dengan Menggunakan Perpustakaan ETL AWS Glue.	18 September 2019
Support untuk mengecual ikan kelas penyimpanan Amazon S3 saat menjalankan pekerjaan	Penambahan informasi tentang pengecualian kelas penyimpanan Amazon S3 ketika menjalankan tugas ETL AWS Glue yang membaca file atau partisi dari Amazon S3. Untuk informasi selengkapnya, lihat Mengecualikan Kelas Penyimpanan Amazon S3.	29 Agustus 2019
Support untuk pengembangan skrip ETL lokal menggunakan pustaka AWS Glue ETL publik	Penambahan informasi tentang bagaimana mengembangkan dan menguji skrip ETL Python dan Scala secara lokal tanpa perlu menggunak an koneksi jaringan. Untuk informasi selengkapnya, lihat Mengembangkan dan Menguji Skrip ETL Secara Lokal dengan Menggunakan Perpustakaan ETL AWS Glue.	28 Agustus 2019

Masalah yang diketahui	Penambahan informasi tentang masalah yang diketahui di AWS Glue. Untuk informasi selengkapnya, lihat Masalah yang Diketahui untuk AWS Glue.	28 Agustus 2019
Support untuk transformasi pembelajaran mesin di AWS Glue	Penambahan informasi tentang kemampuan machine learning yang disediakan oleh AWS Glue untuk membuat transformasi kustom. Anda dapat membuat transform asi ini saat membuat sebuah tugas. Untuk informasi selengkapnya, lihat Transform asi Machine Learning di AWS Glue.	8 Agustus 2019
Dukungan untuk Amazon Virtual Private Cloud bersama	Penambahan informasi tentang dukungan AWS Glue untuk Amazon Virtual Private Cloud bersama. Untuk informasi selengkapnya, lihat Amazon VPC Bersama.	6 Agustus 2019
Support untuk pembuatan versi di AWS Glue	Penambahan informasi tentang mendefinisikan `Glue version` di properti tugas. Versi AWS Glue menentukan versi Apache Spark dan Python yang didukung oleh AWS Glue. Untuk informasi selengkapnya, lihat Menambahkan Tugas di AWS Glue.	24 Juli 2019

Support untuk opsi konfigurasi tambahan untuk titik akhir pengembangan	Penambahan informasi tentang opsi konfigurasi untuk titik akhir pengembangan yang memiliki beban kerja memori-intensif. Anda dapat memilih dari dua konfigurasi baru yang memberikan lebih banyak memori untuk setiap pelaksana. Untuk informasi selengkapnya, lihat Bekerja dengan Titik Akhir Pengembangan pada Konsol AWS Glue.	24 Juli 2019
Support untuk melakukan aktivitas ekstrak, transfer, dan beban (ETL) menggunakan alur kerja	Penambahan informasi tentang menggunakan konstruksi baru yang disebut alur kerja untuk merancang aktivitas extract, transform, and load (ETL) yang kompleks yang dapat dijalankan dan dilacak sebagai entitas tunggal oleh AWS Glue. Untuk informasi selengkapnya, lihat Melakukan Aktivitas ETL Kompleks Menggunakan Alur Kerja di AWS Glue.	20 Juni 2019

Dukungan untuk Python 3.6 dalam pekerjaan shell Python	Penambahan informasi tentang dukungan untuk Python 3.6 di tugas shell Python. Anda dapat menentukan baik Python 2.7 atau Python 3.6 sebagai properti tugas. Untuk informasi selengkapnya, lihat Menambahkan Tugas Shell Python diAWS Glue.	5 Juni 2019
Dukungan untuk titik akhir virtual private cloud (VPC)	Penambahan informasi tentang menghubungkan langsung ke AWS Glue melalui titik akhir antarmuka di VPC Anda. Ketika Anda menggunakan titik akhir antarmuka VPC, komunikasi antara VPC dan Anda AWS Glue dilakukan sepenuhnya dan dengan aman di jaringan AWS . Untuk informasi selengkapnya, lihat Menggunakan AWS Glue dengan Titik Akhir VPC.	4 Juni 2019

Support untuk real-time, logging berkelanjutan untuk AWS Glue pekerjaan.	Menambahkan informasi tentang mengaktifkan dan melihat log pekerjaan Apache Spark real-time CloudWatch termasuk log driver, masing-masing log pelaksana, dan bilah kemajuan pekerjaan Spark. Untuk informasi selengkapnya, lihat Pencatatan Log Berkelanjutan untuk Tugas AWS Glue.	28 Mei 2019
Support untuk tabel Data Catalog yang ada sebagai sumber crawler	Penambahan informasi tentang menentukan daftar tabel Katalog Data yang ada sebagai sumber crawler. Crawler kemudian dapat mendeteksi perubahan pada skema tabel, memperbarui definisi tabel, dan mendaftarkan partisi baru saat data baru menjadi tersedia. Untuk informasi selengkapnya, lihat Properti Crawler.	10 Mei 2019

Support untuk opsi konfigurasi tambahan untuk pekerjaan intensif memori	Penambahan informasi tentang pilihan konfigurasi untuk tugas Apache Spark dengan beban kerja intensif-memori. Anda dapat memilih dari dua konfigurasi baru yang memberikan lebih banyak memori untuk setiap pelaksana. Untuk informasi selengkapnya, lihat Menambahkan Tugas di AWS Glue.	5 April 2019
Support untuk pengklasifikasi kustom CSV	Penambahan informasi tentang penggunaan pengklasifikasi CSV kustom untuk menyimpulkan skema berbagai jenis data CSV. Untuk informasi selengkapnya, lihat Menulis Pengklasifikasi Kustom.	26 Maret 2019
Support untuk tag AWS sumber daya	Menambahkan informasi tentang penggunaan tag AWS sumber daya untuk membantu Anda mengelola dan mengontrol akses ke AWS Glue sumber daya Anda. Anda dapat menetapkan tag AWS sumber daya untuk pekerjaan, pemicu, titik akhir, dan crawler di. AWS Glue Untuk informasi selengkapnya, lihat Tag AWS di AWS Glue.	20 Maret 2019

Support of Data Catalog untuk pekerjaan Spark SQL	Menambahkan informasi tentang mengonfigurasi AWS Glue pekerjaan dan titik akhir pengembangan Anda untuk menggunakan Metastore Apache Hive AWS Glue Data Catalog sebagai eksternal. Hal ini memungkinkan tugas dan titik akhir pengemban gan untuk secara langsung menjalankan kueri Apache Spark SQL terhadap tabel yang disimpan dalam AWS Glue Data Catalog. Untuk informasi selengkapnya, lihat Support AWS Glue Data Catalog untuk Tugas Spark SQL.	14 Maret 2019
Support untuk pekerjaan Python shell	Penambahan informasi tentang tugas shell Python dan bidang baru Kapasitas maksimal. Untuk informasi selengkapnya, lihat Menambahkan Tugas Shell Python di AWS Glue.	18 Januari 2019

Support untuk notifikasi ketika ada perubahan pada database dan tabel	Penambahan informasi tentang peristiwa yang dihasilkan untuk perubahan basis data, tabel, dan panggilan API partisi. Anda dapat mengonfigurasi tindakan di CloudWatch Acara untuk menanggapi peristiwa ini. Untuk informasi selengkap nya, lihat Mengotomatisasi AWS Glue dengan CloudWatc h Acara.	16 Januari 2019
Support untuk mengenkripsi kata sandi koneksi	Penambahan informasi tentang mengenkripsi kata sandi yang digunakan dalam objek koneksi. Untuk informasi selengkapnya, lihat Mengenkri psi Kata sandi Koneksi.	11 Desember 2018
Dukungan untuk izin tingkat sumber daya dan kebijakan berbasis sumber daya	Penambahan informasi tentang menggunakan kebijakan berbasis sumber daya dan kebijakan dengan AWS Glue. Untuk informasi selengkapnya, lihat topik di Keamanan di AWS Glue.	15 Oktober 2018
Support untuk SageMaker notebook	Menambahkan informasi tentang penggunaan SageMaker notebook dengan titik akhir AWS Glue pengembangan. Untuk informasi selengkapnya, lihat Mengelola Notebook.	5 Oktober 2018

Support untuk enkripsi	Penambahan informasi tentang menggunakan enkripsi dengan AWS Glue. Untuk informasi selengkapnya, lihat Enkripsi Saat Data Tidak Berpindah, Enkripsi in Transit, dan Menyiapkan Enkripsi di AWS Glue.	24 Agustus 2018
Support untuk metrik pekerjaan Apache Spark	Penambahan informasi tentang penggunaan metrik Apache Spark untuk melakukan debugging dan pemrofilan tugas ETL yang lebih baik. Anda dapat dengan mudah melacak metrik waktu aktif seperti byte yang dibaca dan ditulis, penggunaan memori dan beban CPU driver dan pelaksana, dan data yang diacak antara pelaksana dari konsol AWS Glue. Untuk informasi selengkapnya, lihat Memantau AWS Glue Menggunakan CloudWatch Metrik, Pemantauan Pekerjaan , dan Debugging, dan Bekerja dengan Pekerjaan di AWS Glue Konsol.	13 Juli 2018

Support DynamoDB sebagai sumber data	Penambahan informasi tentang cara melakukan perayapan DynamoDB dan menggunakannya sebagai sumber data tugas ETL. Untuk informasi selengkapnya, lihat Katalogisasi Tabel dengan Crawler dan Parameter Koneksi.	10 Juli 2018
Pembaruan untuk membuat prosedur server notebook	Pembaruan informasi tentang cara membuat server notebook pada instans Amazon EC2 yang dikaitkan dengan titik akhir pengembangan. Untuk informasi selengkapnya, lihat Membuat Server Notebook yang Dikaitkan dengan Titik Akhir Pengembangan.	9 Juli 2018
Pembaruan kini tersedia melalui RSS	Anda sekarang dapat berlangganan umpan RSS untuk menerima notifikasi tentang pembaruan untuk Panduan Developer AWS Glue .	25 Juni 2018
Support pemberitahuan penundaan untuk pekerjaan	Penambahan informasi tentang mengkonfigurasi ambang penundaan saat tugas berjalan. Untuk informasi selengkapnya, lihat Menambahkan Tugas di AWS Glue.	25 Mei 2018

Konfigurasikan crawler untuk menambahkan kolom baru	Menambahkan informasi tentang opsi konfigurasi baru untuk crawler, MergeNewC olumns. Untuk informasi selengkapnya, lihat Mengkonfigurasi Crawler.	7 Mei 2018
Support timeout pekerjaan	Penambahan informasi tentang menetapkan ambang batas habis waktu saat tugas berjalan. Untuk informasi selengkapnya, lihat Menambahkan Tugas di AWS Glue.	10 April 2018
Support Scala ETL script dan memicu pekerjaan berdasarkan status run tambahan	Penambahan informasi tentang menggunakan Scala sebagai bahasa pemrograman ETL. Selain itu, API pemicu sekarang mendukung pengaktifan ketika ada salah satu syarat terpenuhi (selain semua syarat). Selain itu, tugas dapat dipicu berdasarkan tugas "gagal" atau "berhenti" (selain eksekusi tugas "berhasil").	12 Januari 2018

Pembaruan sebelumnya

Tabel berikut menjelaskan perubahan penting dalam setiap rilis Panduan Developer AWS Glue sebelum Januari 2018.

Perubahan	Deskripsi	Tanggal
Support sumber data XML dan opsi konfigurasi crawler yang baru	Penambahan informasi tentang cara mengklasifikasikan sumber data XML dan opsi crawler baru untuk perubahan partisi.	16 November 2017
Transformasi baru, dukungan untuk mesin basis data Amazon RDS tambahan, dan titik akhir pengemban gan tambahan	Penambahan informasi tentang transform asi peta dan filter, dukungan untuk Amazon RDS Microsoft SQL Server, dan Amazon RDS Oracle, dan fitur baru untuk titik akhir pengembangan.	29 September 2017
Rilis awal AWS Glue	Ini adalah rilis awal dari Panduan Developer AWS Glue .	14 Agustus, 2017

AWS Glosarium

Untuk AWS terminologi terbaru, lihat AWS glosarium di Referensi.Glosarium AWS

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.