Guida per l'utente

Amazon Athena

Amazon Athena: Guida per l'utente

Table of Contents

Che cos'è Amazon Athena?

Amazon Athena è un servizio interattivo di esecuzione di query che semplifica l'analisi di dati direttamente in Amazon Simple Storage Service (Amazon S3) con SQL standard. Con poche azioni AWS Management Console, puoi indirizzare Athena ai tuoi dati archiviati in Amazon S3 e iniziare a utilizzare SQL standard per eseguire query ad hoc e ottenere risultati in pochi secondi.

Per ulteriori informazioni, consulta Nozioni di base.

Amazon Athena facilita anche l'esecuzione di analisi dei dati in modo interattivo mediante Apache Spark senza dover pianificare, configurare o gestire le risorse. Quando esegui applicazioni Apache Spark su Athena, invii il codice Spark in elaborazione e ricevi direttamente i risultati. Utilizza l'esperienza semplificata dei notebook nella console Amazon Athena per sviluppare applicazioni Apache Spark utilizzando Python o API dei notebook Athena.

Per ulteriori informazioni, consulta Nozioni di base su Apache Spark su Amazon Athena.

Athena SQL e Apache Spark su Amazon Athena sono serverless, perciò non occorre installare o gestire alcuna infrastruttura e vengono addebitati solo i costi relativi alle query eseguite. Athena scala automaticamente, eseguendo query in parallelo, in modo che i risultati siano veloci, anche con set di dati di grandi dimensioni e query complesse.

Argomenti

Quando è opportuno utilizzare Athena?

Servizi di esecuzione di query come Amazon Athena, data warehouse come Amazon Redshift e sofisticati framework di elaborazione dati come Amazon EMR soddisfano esigenze e casi d'uso diversi. Le seguenti linee guida possono aiutarti a scegliere uno o più servizi in base alle tue esigenze.

Amazon Athena

Athena consente di analizzare dati non strutturati, semistrutturati e strutturati archiviati in Amazon S3. Tra gli esempi figurano CSV, JSON o formati di dati colonnari come Apache Parquet e Apache ORC. È possibile usare Athena per eseguire query ad-hoc con ANSI SQL, senza la necessità di aggregare o caricare i dati in Athena.

Athena si integra con Amazon QuickSight per una facile visualizzazione dei dati. È possibile utilizzare Athena per generare report o per analizzare i dati con strumenti di business intelligence o client SQL, collegati con un driver JDBC o ODBC. Per ulteriori informazioni, consulta What is Amazon QuickSight nella Amazon QuickSight User Guide eConnessione ad Amazon Athena con i driver ODBC e JDBC.

Athena si integra con AWS Glue Data Catalog, che offre un archivio di metadati persistente per i tuoi dati in Amazon S3. Ciò ti consente di creare tabelle e interrogare i dati in Athena sulla base di un archivio di metadati centrale disponibile in tutto il tuo account Amazon Web Services e integrato con le funzionalità ETL e di rilevamento dei dati di. AWS Glue Per ulteriori informazioni, consulta Integrazione con AWS Glue e Che cos'è AWS Glue nella Guida per gli sviluppatori di AWS Glue .

Amazon Athena semplifica l'esecuzione di query interattive sui dati direttamente in Amazon S3 senza dover formattare i dati o gestire l'infrastruttura. Ad esempio, Athena è utile se si desidera eseguire una query rapida sui registri Web per risolvere un problema di prestazioni sul sito. Con Athena puoi iniziare velocemente: devi semplicemente definire una tabella per i tuoi dati e iniziare a eseguire query utilizzando SQL standard.

È consigliabile utilizzare Amazon Athena se si desidera eseguire query SQL interattive ad hoc sui dati su Amazon S3, senza dover gestire alcuna infrastruttura o cluster. Amazon Athena è il modo più semplice per eseguire query ad hoc per i dati in Amazon S3 senza dover configurare o gestire alcun server.

Per un elenco delle funzionalità utilizzate da Athena o con Servizi AWS cui si integra, consulta. the section called “Servizio AWS integrazioni con Athena”

Amazon EMR

Amazon EMR rende semplice e conveniente eseguire framework di elaborazione altamente distribuiti come Hadoop, Spark e Presto rispetto alle distribuzioni locali. Amazon EMR è flessibile: puoi eseguire applicazioni e codice personalizzati e definire parametri specifici di elaborazione, memoria, archiviazione e applicazione per ottimizzare i requisiti analitici.

Oltre all'esecuzione di query SQL, Amazon EMR può eseguire un'ampia gamma di attività di elaborazione dei dati con scalabilità orizzontale per applicazioni come machine learning, analisi dei grafici, trasformazione dei dati, streaming di dati e praticamente tutto ciò che è possibile codificare. È consigliabile utilizzare Amazon EMR se si utilizza codice personalizzato per elaborare e analizzare set di dati estremamente grandi con i più recenti framework di elaborazione di Big Data come Spark, Hadoop, Presto o Hbase. Amazon EMR ti dà il pieno controllo sulla configurazione dei cluster e sul software installato su di essi.

Puoi utilizzare Amazon Athena per interrogare i dati elaborati utilizzando Amazon EMR. Amazon Athena supporta molti degli stessi formati di dati di Amazon EMR. Il catalogo dati di Athena è compatibile con il metastore Hive. Se utilizzi EMR e disponi già di un metastore Hive, puoi eseguire le istruzioni DDL su Amazon Athena e interrogare immediatamente i tuoi dati senza influire sui processi Amazon EMR.

Amazon Redshift

Un data warehouse come Amazon Redshift è la scelta migliore quando è necessario raccogliere dati provenienti da molte fonti diverse, come sistemi di inventario, sistemi finanziari e sistemi di vendita al dettaglio, in un formato comune e archiviarli per lunghi periodi di tempo. Se vuoi creare report aziendali sofisticati a partire da dati storici, un data warehouse come Amazon Redshift è la scelta migliore. Il motore di query in Amazon Redshift è stato ottimizzato per funzionare particolarmente bene nell'esecuzione di query complesse che uniscono un numero elevato di tabelle di database molto grandi. Quando devi eseguire query su dati altamente strutturati con molti join su diverse tabelle di grandi dimensioni, scegli Amazon Redshift.

Per maggiori informazioni su quando utilizzare Athena, consulta le seguenti risorse:

- Guida decisionale per i servizi di analisi su AWS nel Centro risorse per le nozioni di base
- Quando utilizzare Athena rispetto ad altri servizi di Big Data nelle domande frequenti su Amazon Athena
- Panoramica di Amazon Athena
- Funzioni di Amazon Athena
- Domande frequenti su Amazon Athena
- Post del blog di Amazon Athena

Servizio AWS integrazioni con Athena

È possibile utilizzare Athena per interrogare i dati Servizi AWS elencati in questa sezione. Per un elenco delle regioni supportate da ciascun servizio, consulta Regioni ed endpoint nella Riferimenti generali di Amazon Web Services.

Servizi AWS integrato con Athena

- AWS CloudFormation
- Amazon CloudFront
- AWS CloudTrail
- Amazon DataZone
- Elastic Load Balancing
- Amazon EMR Studio
- AWS Glue Data Catalog
- AWS Identity and Access Management (IAM)
- Amazon QuickSight
- Inventario Amazon S3
- AWS Step Functions
- Inventario AWS Systems Manager
- Amazon Virtual Private Cloud

Per ulteriori informazioni su ciascuna integrazione, consulta le sezioni riportate di seguito.

AWS CloudFormation

Prenotazione di capacità

Argomento di riferimento: AWS: :Athena:: CapacityReservation nella Guida per l'utente AWS CloudFormation

Specifica una prenotazione della capacità con il nome e il numero forniti delle unità di elaborazione dati richieste. Per ulteriori informazioni, consulta Gestione della capacità di elaborazione delle query la Amazon Athena User Guide e CreateCapacityReservationl'Amazon Athena API Reference.

Catalogo dati

Argomento di riferimento: AWS: :Athena:: DataCatalog nella Guida per l'utente AWS CloudFormation

Specifica un catalogo dati Athena, inclusi un nome, una descrizione, un tipo, parametri e tag. Per ulteriori informazioni, consulta Comprensione di tabelle, database e cataloghi dati la Amazon Athena User Guide e CreateDataCatalogl'Amazon Athena API Reference.

Query denominata

Argomento di riferimento: AWS: :Athena:: NamedQuery nella Guida per l'utente AWS CloudFormation

Specificate le query denominate con AWS CloudFormation ed eseguitele in Athena. Le query denominate consentono di mappare un nome di query a una query e quindi eseguirlo come query salvata dalla console Athena. Per ulteriori informazioni, consulta Utilizzo di query salvate la Amazon Athena User Guide e CreateNamedQueryl'Amazon Athena API Reference.

Dichiarazione preparata

Argomento di riferimento: AWS: :Athena:: PreparedStatement nella Guida per l'utente AWS CloudFormation

Specifica un'istruzione preparata da utilizzare con le query SQL in Athena. Un'istruzione preparata contiene segnaposto di parametro i cui valori vengono forniti al momento dell'esecuzione. Per ulteriori informazioni, consulta Utilizzo di query parametrizzate la Amazon Athena User Guide e CreatePreparedStatementl'Amazon Athena API Reference.

Gruppo di lavoro

Argomento di riferimento: AWS: :Athena:: WorkGroup nella Guida per l'utente AWS CloudFormation

Specificare i gruppi di lavoro Athena utilizzando. AWS CloudFormation Utilizzare gruppi di lavoro Athena per isolare le query per l'utente o il gruppo da altre query nello stesso account. Per ulteriori informazioni, consulta Uso dei gruppi di lavoro per controllare l'accesso alle query e i costi la Amazon Athena User Guide e CreateWorkGroupl'Amazon Athena API Reference.

Amazon CloudFront

Argomento di riferimento: Interrogazione dei log di Amazon CloudFront

Usa Athena per interrogare i log di Amazon CloudFront . Per ulteriori informazioni sull'utilizzo CloudFront, consulta l'Amazon CloudFront Developer Guide.

AWS CloudTrail

Argomento di riferimento: Interrogazione dei log AWS CloudTrail

L'utilizzo di Athena con CloudTrail i log è un modo efficace per migliorare l'analisi dell'attività di AWS servizio. Ad esempio, puoi utilizzare le query per identificare le tendenze e isolare con maggiore precisione le attività in base a un attributo specifico, ad esempio l'indirizzo IP di origine o un utente. Puoi creare tabelle per interrogare i log direttamente dalla CloudTrail console e utilizzarle per eseguire query in Athena. Per ulteriori informazioni, consulta Utilizzo della CloudTrail console per creare una tabella Athena per i log CloudTrail .

Amazon DataZone

Argomento di riferimento: Utilizzo di Amazon DataZone in Athena

Usa Amazon DataZone per condividere, cercare e scoprire dati su larga scala oltre i confini dell'organizzazione. DataZonesemplifica la tua esperienza con servizi di AWS analisi come Athena AWS Glue e. AWS Lake Formation Se disponi di grandi quantità di dati provenienti da fonti di dati diverse, puoi utilizzare Amazon DataZone per creare raggruppamenti di persone, dati e strumenti basati su casi d'uso aziendali.

In Athena, è possibile utilizzare l'editor di query per accedere e interrogare DataZone gli ambienti. Per ulteriori informazioni, consulta Utilizzo di Amazon DataZone in Athena.

Elastic Load Balancing

Argomento di riferimento: Esecuzione di query nei log di Application Load Balancer

Eseguire query sui log dell'Application Load Balancer consente di individuare l'origine del traffico, la latenza e i byte trasferiti a e da istanze Elastic Load Balancing e applicazioni di back-end. Per ulteriori informazioni, consulta Esecuzione di query nei log di Application Load Balancer.

Argomento di riferimento: Richiesta di log Classic Load Balancer

Esegui query sui log di Classic Load Balancer per analizzare e comprendere i modelli di traffico da e verso le istanze di Elastic Load Balancing e le applicazioni di back-end. È possibile visualizzare sorgente di traffico, latenza e byte trasferiti. Per ulteriori informazioni, consulta Creazione di tabelle per i log di ELB.

Amazon EMR Studio

Argomento di riferimento: Use the Amazon Athena SQL editor in EMR Studio

È possibile utilizzare Athena in Amazon EMR Studio per sviluppare ed eseguire query interattive. Ciò consente di utilizzare EMR Studio per l'analisi SQL su Athena dalla stessa interfaccia Amazon EMR utilizzata per Spark, Scala e altri carichi di lavoro. Con l'integrazione di Athena in EMR Studio, puoi svolgere le seguenti attività:

- Eseguire query SQL Athena
- Visualizzazione dei risultati della query
- Visualizzazione della cronologia delle query
- Visualizzare le query salvate
- Eseguire le query parametrizzate
- Visualizzare database, tabelle e viste per un catalogo di dati

Le seguenti funzionalità di Athena non sono disponibili in Amazon EMR Studio:

- Funzionalità di amministrazione come creazione o aggiornamento di gruppi di lavoro Athena, origini dati o prenotazioni della capacità
- Athena per Spark o notebook Spark
- DataZone integrazione
- Step Functions

L'integrazione di EMR Studio con Athena è disponibile ovunque siano disponibili Regioni AWS EMR Studio e Athena. Per ulteriori informazioni sull'utilizzo di Athena in EMR Studio, consulta la pagina Use the Amazon Athena SQL editor in EMR Studio nella Guida per la gestione di Amazon EMR.

AWS Glue Data Catalog

Argomento di riferimento: Integrazione con AWS Glue

Athena si integra con AWS Glue Data Catalog, che offre un archivio di metadati persistente per i tuoi dati in Amazon S3. Ciò ti consente di creare tabelle e interrogare i dati in Athena sulla base di un archivio di metadati centrale disponibile in tutto il tuo account Amazon Web Services e integrato con le funzionalità ETL e di rilevamento dei dati di. AWS Glue Per ulteriori informazioni, consulta la sezione Integrazione con AWS Glue e Che cos'è AWS Glue? nella Guida per gli sviluppatori di AWS Glue .

AWS Identity and Access Management (IAM)

Argomento di referenza: Operazioni per Amazon Athena

È possibile utilizzare le operazioni API Athena nelle policy di autorizzazione IAM. Per ulteriori informazioni, consulta Operazioni per Amazon Athena e Gestione dell'identità e dell'accesso in Athena.

Amazon QuickSight

Argomento di riferimento: Connessione ad Amazon Athena con i driver ODBC e JDBC

Athena si integra con Amazon QuickSight per una facile visualizzazione dei dati. È possibile utilizzare Athena per generare report o per analizzare i dati con strumenti di business intelligence o client SQL, collegati con un driver JDBC o ODBC. Per ulteriori informazioni su Amazon QuickSight, consulta What is Amazon QuickSight nella Amazon QuickSight User Guide. Per informazioni sull'utilizzo dei driver JDBC e ODBC con Athena, vedereConnessione ad Amazon Athena con i driver ODBC e JDBC.

Inventario Amazon S3

Argomento di riferimento: Esecuzione di query sull'inventario con Athena nella Guida per l'utente di Amazon Simple Storage Service

Puoi utilizzare Amazon Athena per eseguire query su un inventario Amazon S3 utilizzando SQL standard. Puoi usare l'inventario Amazon S3 per eseguire audit e report sullo stato di replica e crittografia degli oggetti per esigenze aziendali, normative e di conformità. Per ulteriori informazioni, consulta Inventario Amazon S3 nella Guida per l'utente di Amazon Simple Storage Service.

AWS Step Functions

Argomento di referenza: Chiama Athena con Step Functions nella Guida per gli sviluppatori di AWS Step Functions

Chiama Athena con. AWS Step Functions AWS Step Functions può controllare la selezione Servizi AWS direttamente utilizzando Amazon States Language. È possibile utilizzare Step Functions con Athena per avviare e interrompere l'esecuzione di query, ottenere risultati di query, eseguire query di dati ad-hoc o pianificate e recuperare i risultati da data lake in Amazon S3. Il ruolo Step Functions deve disporre delle autorizzazioni per utilizzare Athena. Per ulteriori informazioni, consulta la Guida per gli sviluppatori di AWS Step Functions.

Video: Orchestra le query di Amazon Athena utilizzando AWS Step Functions

Il video seguente mostra come usare Amazon Athena AWS Step Functions ed eseguire una query Athena pianificata regolarmente e generare un report corrispondente.

Orchestra le query di Amazon Athena utilizzando AWS Step Functions

Per un esempio che utilizza Step Functions e Amazon EventBridge per orchestrare, AWS Glue DataBrew Athena e Amazon QuickSight, consulta Orchestrating an job e AWS Glue DataBrew Amazon Athena query with nel Big Data Blog. AWS Step Functions AWS

AWS Systems Manager Inventory

Argomento di riferimento: Esecuzione di query sui dati di inventario da più Regioni e account nella Guida per l'utente di AWS Systems Manager

AWS Systems Manager Inventory si integra con Amazon Athena per aiutarti a interrogare i dati di inventario da Regioni AWS più account. Per ulteriori informazioni, consulta la Guida per l'utente AWS Systems Manager.

Amazon Virtual Private Cloud

Argomento di riferimento: Esecuzione di query sui log di flusso Amazon VPC

I flussi di log Amazon Virtual Private Cloud acquisiscono informazioni sul traffico IP da e verso le interfacce di rete in un VPC. Esegui query sui log in Athena per analizzare i modelli di traffico di rete e identificare le minacce e i rischi nella rete Amazon VPC. Per ulteriori informazioni su Amazon VPC, consulta la Guida per l'utente di Amazon VPC.

Configurazione

Se hai già effettuato la registrazione ad Amazon Web Services, puoi iniziare a utilizzare Amazon Athena immediatamente. Se non ti sei registrato AWS o hai bisogno di assistenza per iniziare, assicurati di completare le seguenti attività.

Registrati per un Account AWS

Se non ne hai uno Account AWS, completa i seguenti passaggi per crearne uno.

Per iscriverti a un Account AWS

1. Apri la pagina all'indirizzo https://portal.aws.amazon.com/billing/signup.
2. Segui le istruzioni online.

Nel corso della procedura di registrazione riceverai una telefonata, durante la quale sarà necessario inserire un codice di verifica attraverso la tastiera del telefono.

Quando ti iscrivi a un Account AWS, Utente root dell'account AWSviene creato un. L'utente root dispone dell'accesso a tutte le risorse e tutti i Servizi AWS nell'account. Come procedura consigliata in materia di sicurezza, assegnate l'accesso amministrativo a un utente e utilizzate solo l'utente root per eseguire attività che richiedono l'accesso da parte dell'utente root.

AWS ti invia un'e-mail di conferma dopo il completamento della procedura di registrazione. È possibile visualizzare l'attività corrente dell'account e gestire l'account in qualsiasi momento accedendo all'indirizzo https://aws.amazon.com/ e selezionando Il mio account.

Crea un utente con accesso amministrativo

Dopo esserti registrato Account AWS, proteggi Utente root dell'account AWS AWS IAM Identity Center, abilita e crea un utente amministrativo in modo da non utilizzare l'utente root per le attività quotidiane.

Proteggi i tuoi Utente root dell'account AWS

1. Accedi AWS Management Consolecome proprietario dell'account scegliendo Utente root e inserendo il tuo indirizzo Account AWS email. Nella pagina successiva, inserisci la password.

   Per informazioni sull'accesso utilizzando un utente root, consulta la pagina Signing in as the root user della Guida per l'utente di Accedi ad AWS .
2. Abilita l'autenticazione a più fattori (MFA) per l'utente root.

   Per istruzioni, consulta Abilitare un dispositivo MFA virtuale per l'utente Account AWS root (console) nella Guida per l'utente IAM.

Crea un utente con accesso amministrativo

1. Abilita Centro identità IAM.

   Per istruzioni, consulta Abilitazione di AWS IAM Identity Center nella Guida per l'utente di AWS IAM Identity Center .
2. In IAM Identity Center, concedi l'accesso amministrativo a un utente.

Per un tutorial sull'utilizzo di IAM Identity Center directory come fonte di identità, consulta Configurare l'accesso utente con le impostazioni predefinite IAM Identity Center directory nella Guida per l'AWS IAM Identity Center utente.

Accedi come utente con accesso amministrativo

- Per accedere con l'utente IAM Identity Center, utilizza l'URL di accesso che è stato inviato al tuo indirizzo e-mail quando hai creato l'utente IAM Identity Center.

  Per informazioni sull'accesso utilizzando un utente IAM Identity Center, consulta AWS Accedere al portale di accesso nella Guida per l'Accedi ad AWS utente.

Assegna l'accesso ad altri utenti

1. In IAM Identity Center, crea un set di autorizzazioni che segua la migliore pratica di applicazione delle autorizzazioni con privilegi minimi.

   Per istruzioni, consulta Creare un set di autorizzazioni nella Guida per l'utente.AWS IAM Identity Center
2. Assegna gli utenti a un gruppo, quindi assegna l'accesso Single Sign-On al gruppo.

   Per istruzioni, consulta Aggiungere gruppi nella Guida per l'utente.AWS IAM Identity Center

Concessione dell'accesso programmatico

Gli utenti hanno bisogno di un accesso programmatico se vogliono interagire con l' AWS AWS Management Console esterno di. Il modo per concedere l'accesso programmatico dipende dal tipo di utente che accede. AWS

Per fornire agli utenti l'accesso programmatico, scegli una delle seguenti opzioni.

Quale utente necessita dell'accesso programmatico?	Per	Come
Identità della forza lavoro	Utilizza credenziali temporane e per firmare le richieste	Segui le istruzioni per l'interfa ccia che desideri utilizzare.

Quale utente necessita dell'accesso programmatico?	Per	Come
(Utenti gestiti nel centro identità IAM)	programmatiche agli AWS CLI AWS SDK o alle API. AWS	• Per la AWS CLI, consulta Configurazione dell'uso AWS IAM Identity Center nella Guida AWS CLI per l'utente.AWS Command Line Interface • Per AWS SDK, strumenti e AWS API, consulta l'autenticazione IAM Identity Center nella Guida di riferimento agli AWS SDK e agli strumenti.
IAM	Utilizza credenziali temporane e per firmare le richieste programmatiche agli SDK o alle API AWS CLI. AWS AWS	Segui le istruzioni in Uso delle credenziali temporanee con AWS risorse nella Guida per l'utente IAM.

Quale utente necessita dell'accesso programmatico?	Per	Come
IAM	(Non consigliato) Utilizza credenziali a lungo termine per firmare le richieste programmatiche agli AWS CLI AWS SDK o alle API. AWS	Segui le istruzioni per l'interfaccia che desideri utilizzare. • Per la AWS CLI, consulta Autenticazione tramite credenziali utente IAM nella Guida per l'utente.AWS Command Line Interface • Per gli AWS SDK e gli strumenti, consulta Autenticazione tramite credenziali a lungo termine nella Guida di riferimento agli SDK e agli AWS strumenti. • Per le AWS API, consulta Gestione delle chiavi di accesso per gli utenti IAM nella Guida per l'utente IAM.

Allegare policy gestite per Athena

Le policy gestite da Athena concedono le autorizzazioni per utilizzare le funzionalità di Athena. È possibile collegare queste policy gestite a uno o più ruoli IAM che gli utenti possono assumere per utilizzare Athena.

Un ruolo IAM è un'identità IAM che puoi creare nel tuo account e che dispone di autorizzazioni specifiche. Un ruolo IAM è simile a quello di un utente IAM in quanto è un' AWS identità con policy di autorizzazioni che determinano ciò che l'identità può e non può fare. AWS Tuttavia, invece di essere associato in modo univoco a una persona, un ruolo è destinato a essere assunto da chiunque. Inoltre, un ruolo non ha credenziali a lungo termine standard associate (password o chiavi di accesso). Tuttavia, quando assumi un ruolo, vengono fornite le credenziali di sicurezza provvisorie per la sessione del ruolo.

Per ulteriori informazioni sui ruoli, consulta le sezioni IAM roles (Ruoli IAM) e Creating IAM roles (Creazione di ruoli IAM) nella Guida per l'utente di IAM.

Per creare un ruolo che garantisca l'accesso ad Athena, collega le policy gestite da Athena al ruolo. Sono disponibili due policy gestite per Athena: `AmazonAthenaFullAccess` e `AWSQuicksightAthenaAccess`. Queste policy concedono autorizzazioni ad Athena per eseguire query su Amazon S3, nonché per scrivere i risultati delle query in un bucket separato a tuo nome. Per visualizzare il contenuto di queste policy per Athena, consulta AWS politiche gestite per Amazon Athena.

Per i passaggi per collegare le policy gestite da Athena, segui la procedura Adding IAM identity permissions (console) (Aggiunta di autorizzazioni per identità IAM [console]) nella Guida per l'utente di IAM e aggiungi le policy gestite `AmazonAthenaFullAccess` e `AWSQuicksightAthenaAccess` al ruolo creato.

> **Note**
>
> Potresti necessitare di autorizzazioni aggiuntive per accedere ai set di dati sottostanti in Amazon S3. Se non sei il proprietario dell'account o il tuo accesso a un bucket è limitato per altri motivi, contatta il proprietario del bucket affinché ti conceda l'accesso utilizzando una policy del bucket basata sulle risorse; in alternativa, contatta il l'amministratore del tuo account affinché ti conceda l'accesso utilizzando una policy basata sul ruolo. Per ulteriori informazioni, consulta Accesso ad Amazon S3 da Athena. Se il set di dati o i risultati delle query di Athena sono crittografati, possono essere necessarie ulteriori autorizzazioni. Per ulteriori informazioni, consulta Crittografia a riposo.

Accesso ad Athena

Puoi accedere ad Athena utilizzando una connessione JDBC o ODBC AWS Management Console, l'API Athena, l'Athena CLI, l'SDK oppure. AWS AWS Tools for Windows PowerShell

- Per le nozioni di base sull'utilizzo di Athena SQL con la console, consulta la pagina Nozioni di base.
- Per iniziare a creare notebook compatibili con Jupyter e applicazioni Apache Spark che utilizzano Python, consulta. Utilizzo di Apache Spark in Amazon Athena
- Per informazioni su come utilizzare i driver JDBC oppure ODBC, consulta le sezione Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

- Per utilizzare l'API Athena, consulta la documentazione di riferimento dell'API Amazon Athena.
- Per utilizzare la CLI, installa la AWS CLI e digita `aws athena help` nella riga di comando per visualizzare i comandi disponibili. Per informazioni sui comandi disponibili, consulta la Guida di riferimento alla riga di comando di Amazon Athena.
- Per utilizzare AWS SDK for Java 2.x, consulta la sezione Athena dell'AWS SDK for Java 2.x API Reference, gli esempi di Athena Java V2 su GitHub .com e la Developer Guide.AWS SDK for Java 2.x
- Per utilizzare il AWS SDK for .NET, consulta lo spazio dei `Amazon.Athena` nomi nell'AWS SDK for .NET API Reference, gli esempi.NET Athena su GitHub .com e la Developer Guide.AWS SDK for .NET
- Per utilizzarlo AWS Tools for Windows PowerShell, consulta il riferimento al cmdlet AWS Tools for PowerShell - Amazon Athena, la pagina del AWS Tools for PowerShellportale e la Guida per l'utente.AWS Tools for Windows PowerShell
- Per informazioni sugli endpoint di servizio Athena ai quali puoi connetterti a livello programmatico, consulta Endpoint e quote di Amazon Athena nella Riferimenti generali di Amazon Web Services.

Utilizzo di Athena SQL

Puoi utilizzare Athena SQL per eseguire query sui tuoi dati direttamente in Amazon S3 utilizzando AWS Glue Data Catalog, un metastore Hive esterno o query federate con una varietà di connettori predefiniti ad altre origini dati.

Puoi anche:

- Connettiti agli strumenti di business intelligence e ad altre applicazioni utilizzando i driver JDBC e ODBC di Athena.
- Esegui query sui log dei servizi AWS.
- Esegui query sulle tabelle Apache Iceberg, incluse query temporali, e sui set di dati Apache Hudi.
- Esegui query sui dati geospaziali.
- Esegui query utilizzando l'inferenza di apprendimento automatico di Amazon SageMaker.
- Esegui query utilizzando funzioni definite dall'utente personalizzate.
- Velocizza l'elaborazione delle query di tabelle altamente partizionate e automatizza la gestione delle partizioni utilizzando la proiezione delle partizioni.

Argomenti

- Risoluzione dei problemi in Athena
- Esempi di codice

Comprensione di tabelle, database e cataloghi dati

In Athena, i cataloghi, i database e le tabelle sono container per le definizioni dei metadati che definiscono uno schema per i dati di origine sottostanti.

Athena utilizza i seguenti termini per fare riferimento alle gerarchie di oggetti dati:

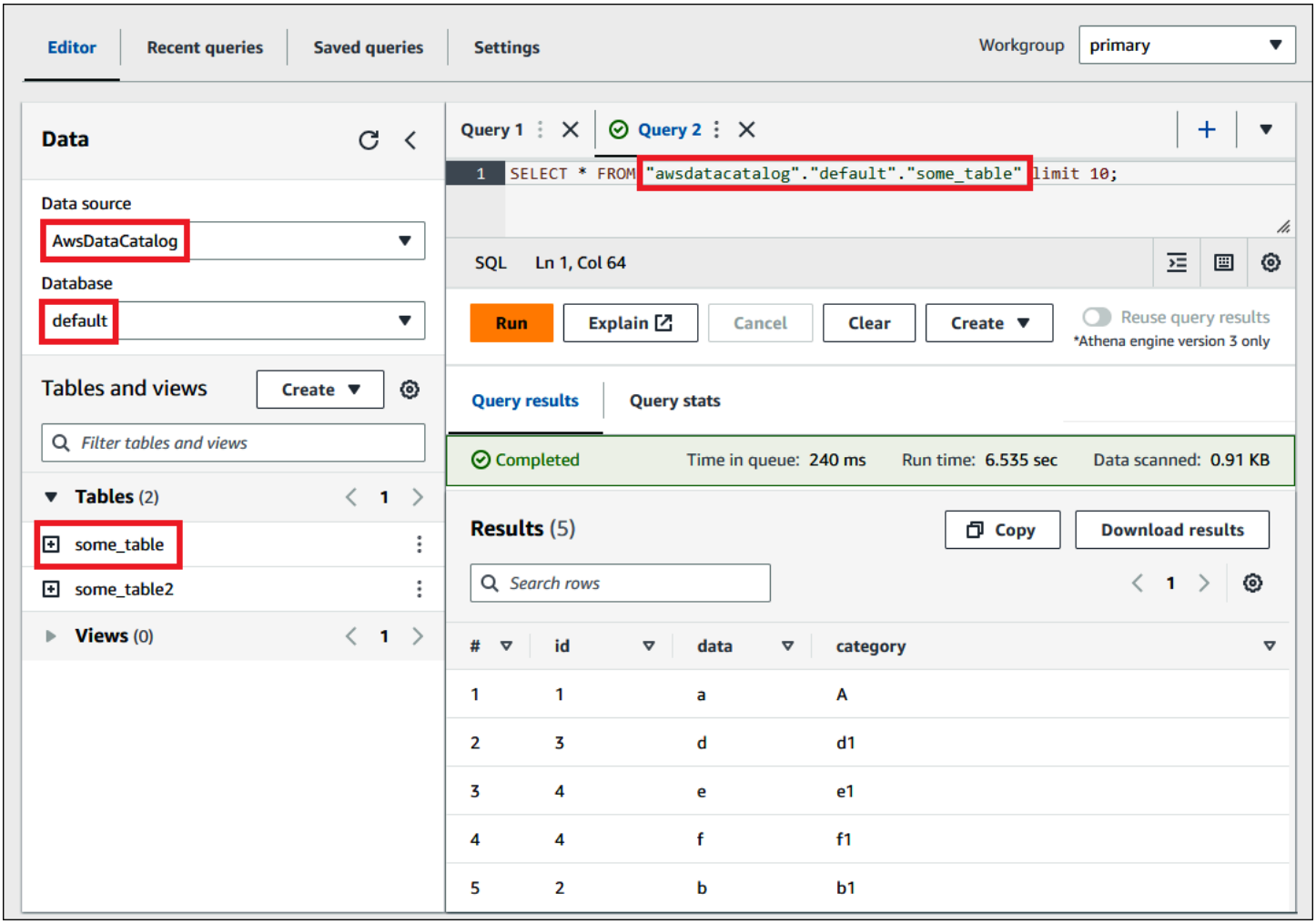
- Origine dati: un gruppo di database
- Database: un gruppo di tabelle
- Tabella: dati organizzati come gruppo di righe o colonne

A volte si fa riferimento a questi oggetti anche con nomi alternativi ma equivalenti, come i seguenti:

- Un'origine dati talvolta viene definita catalogo.
- Talvolta un database viene definito schema.

> **Note**
>
> Questa terminologia può variare nelle fonti di dati federate utilizzate con Athena. Per ulteriori informazioni, consulta Athena e i qualificatori dei nomi delle tabelle federate.

La seguente query di esempio nella console Athena utilizza l'origine dati `awsdatacatalog`, il database `default` e la tabella `some_table`.



Per ogni set di dati, deve esistere una tabella in Athena. I metadati nella tabella indicano ad Athena dove si trovano i dati in Amazon S3 e specificano la struttura dei dati, ad esempio, i nomi delle colonne, i tipi di dati e il nome della tabella. I database sono un raggruppamento logico di tabelle e inoltre contengono solo i metadati e le informazioni sullo schema per un set di dati.

Per ogni set di dati su cui si desidera eseguire query, Athena deve avere una tabella sottostante che utilizzerà per ottenere e restituire i risultati delle query. Pertanto, prima di eseguire query sui dati, occorre registrare una tabella in Athena. La registrazione si verifica quando si creano tabelle automaticamente o manualmente.

È possibile creare una tabella automaticamente utilizzando un AWS Glue crawler. Per ulteriori informazioni sui crawler AWS Glue e sui crawler, consulta Integrazione con. AWS Glue Quando AWS Glue crea una tabella, la registra nel proprio catalogo dati. AWS Glue Athena utilizza il catalogo dati AWS Glue per archiviare e recuperare questi metadati, utilizzandoli quando esegui query per analizzare il set di dati sottostante.

Indipendentemente dal modo in cui le tabelle vengono create, il processo di creazione delle tabelle registra il set di dati in Athena. Questa registrazione avviene in AWS Glue Data Catalog e consente ad Athena di eseguire interrogazioni sui dati. Nell'editor di query Athena, a questo catalogo (o origine dati) si fa riferimento con l'etichetta `AwsDataCatalog`.

Dopo aver creato una tabella, puoi usare l'istruzione SQL SELECT per interrogarla, incluso per ottenere percorsi file specifici per i dati di origine. I risultati della query sono archiviati in Amazon S3 nella posizione dei risultati delle query specificata.

Il catalogo AWS Glue dati è accessibile tramite il tuo account Amazon Web Services. Altri Servizi AWS possono condividere il catalogo AWS Glue dati, in modo che tu possa vedere i database e le tabelle creati in tutta l'organizzazione utilizzando Athena e viceversa.

- Creare una tabella manualmente:
  - Utilizzare la console Athena per eseguire la procedura guidata Crea tabella.
  - Utilizzare la console Athena per scrivere istruzioni DDL Hive nell'editor di query.
  - Utilizzare l'API Athena o la CLI per eseguire una stringa di query SQL con istruzioni DDL.
  - Utilizzare il driver JDBC o ODBC Athena.

Quando si creano tabelle e database manualmente, Athena utilizza istruzioni HiveQL (Data Definition Language) come `CREATE TABLE`, `CREATE DATABASE` e `DROP TABLE` dietro le quinte per creare tabelle e database in AWS Glue Data Catalog.

Per iniziare, puoi utilizzare un tutorial nella console Athena o consultare una step-by-step guida nella documentazione di Athena.

- Per utilizzare il tutorial nella console Athena, scegli l'icona delle informazioni in alto a destra della console, quindi scegli la scheda Tutorial.
- Per un step-by-step tutorial sulla creazione di una tabella e sulla scrittura di interrogazioni nell'editor di query Athena, consulta. Nozioni di base

Nozioni di base

Questo tutorial ti guiderà nell'utilizzo di Amazon Athena per eseguire le query dei dati. Durante il tutorial, creerai una tabella basata sui dati campione archiviati in Amazon Simple Storage Service, eseguire una query della tabella e controllare i risultati della query.

Il tutorial utilizza risorse in tempo reale, pertanto ti sarà addebitato il costo delle query da te eseguite. Non ti saranno addebitati i costi per i dati campione nella posizione utilizzata da questo tutorial, tuttavia se carichi tuoi file di dati in Amazon S3 vengono applicate delle tariffe.

Prerequisiti

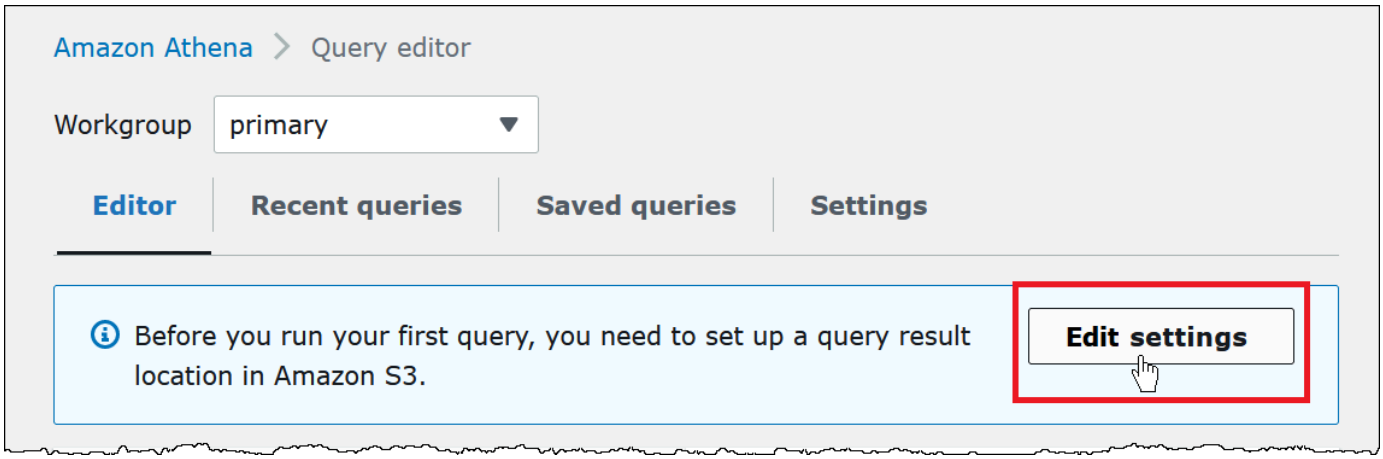
- Se non lo hai ancora fatto, registrati per creare un Account AWS.
- Utilizzando lo stesso account Regione AWS (ad esempio, US West (Oregon)) e lo stesso account che utilizzi per Athena, segui i passaggi per creare un bucket in Amazon S3 per contenere i risultati delle tue query Athena. Configurerai questo bucket come posizione di output della tua query.

Fase 1: crea un database

In primo luogo, devi creare un database in Athena.

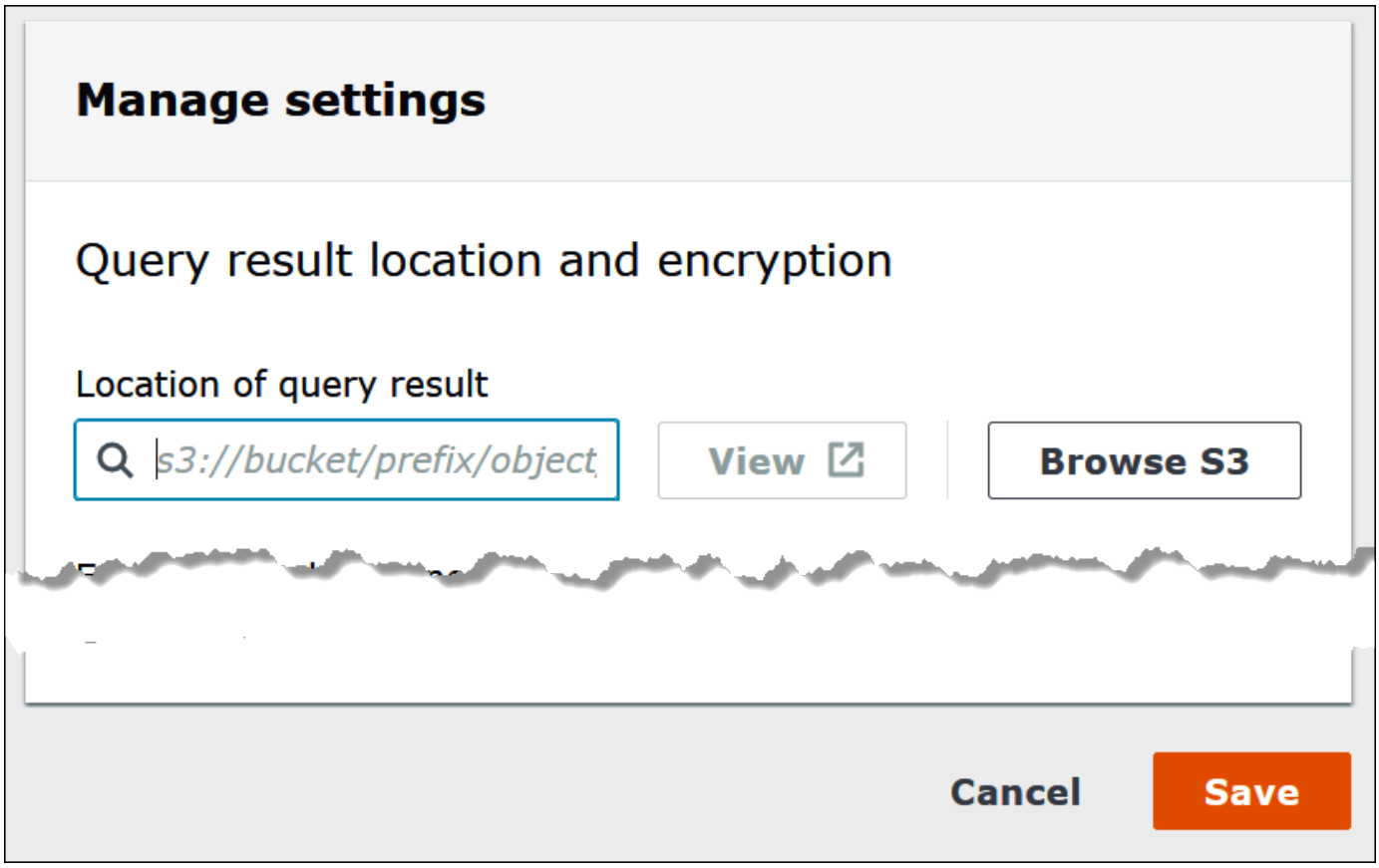
Per creare un database Athena

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se è la prima volta che visiti la console di Athena nella Regione AWS corrente, scegli Explore the query editor (Esplora l'editor di query) per aprire l'editor di query. Altrimenti, Athena apre la query nell'editor di query.
3. Scegli Edit Settings (Modifica impostazioni) per configurare una nuova posizione dei risultati di una query in Amazon S3.

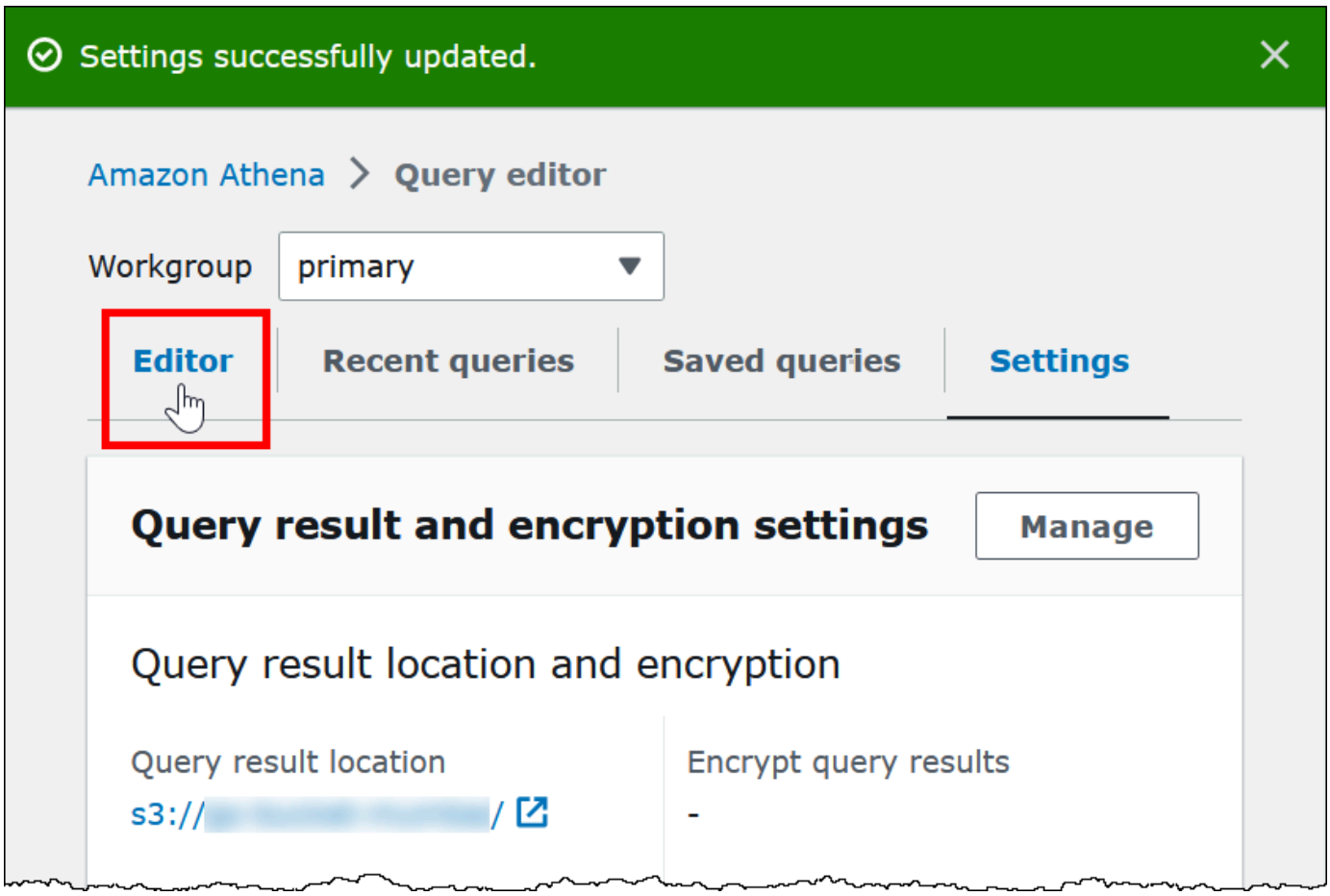


4. Per Manage settings (Gestisci impostazioni), effettua una delle seguenti operazioni:

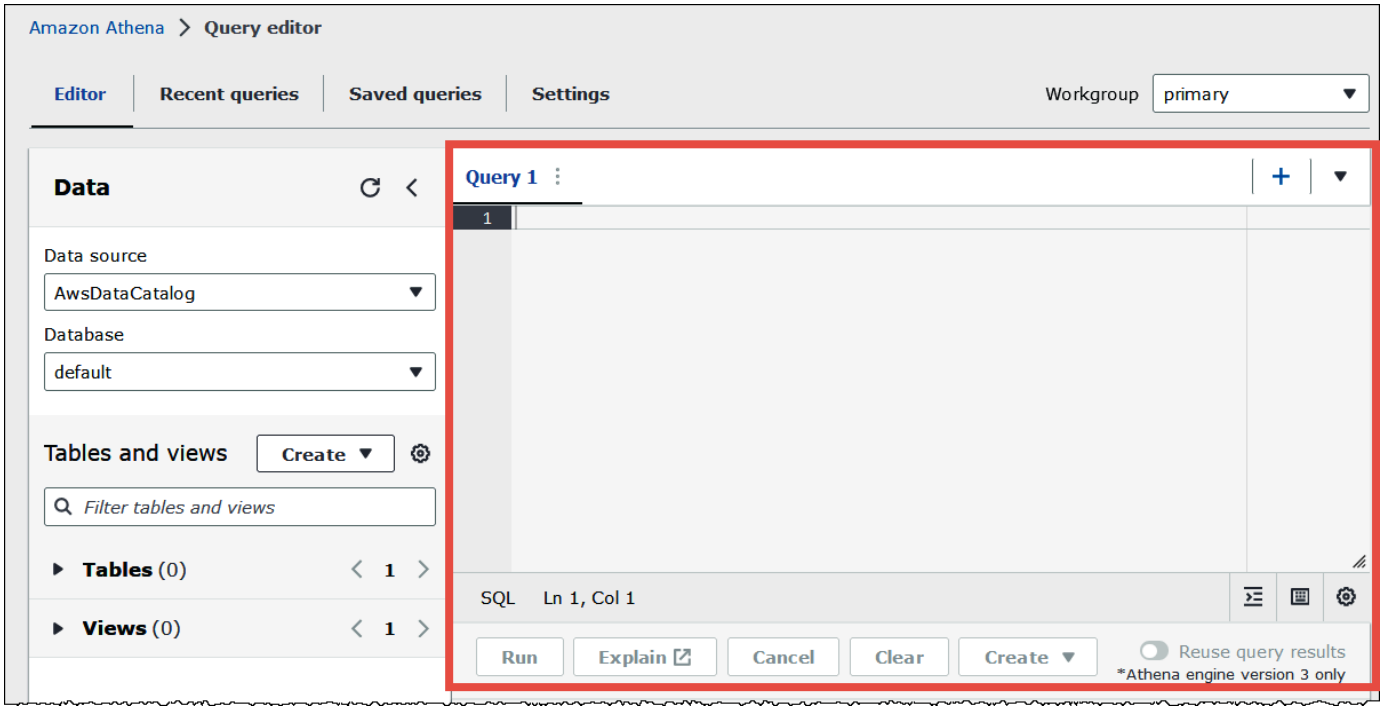
- Nella casella Location of query result (Posizione dei risultati delle query) inserisci il percorso del bucket creato in Amazon S3 per i risultati delle query. Aggiungi al percorso il prefisso `s3://`.
- Scegli Browse S3 (Sfoglia S3), scegli il bucket Amazon S3 creato per la tua regione corrente, quindi seleziona Choose (Scegli).



5. Selezionare Salva.
6. Scegli Editor per passare all'editor di query.



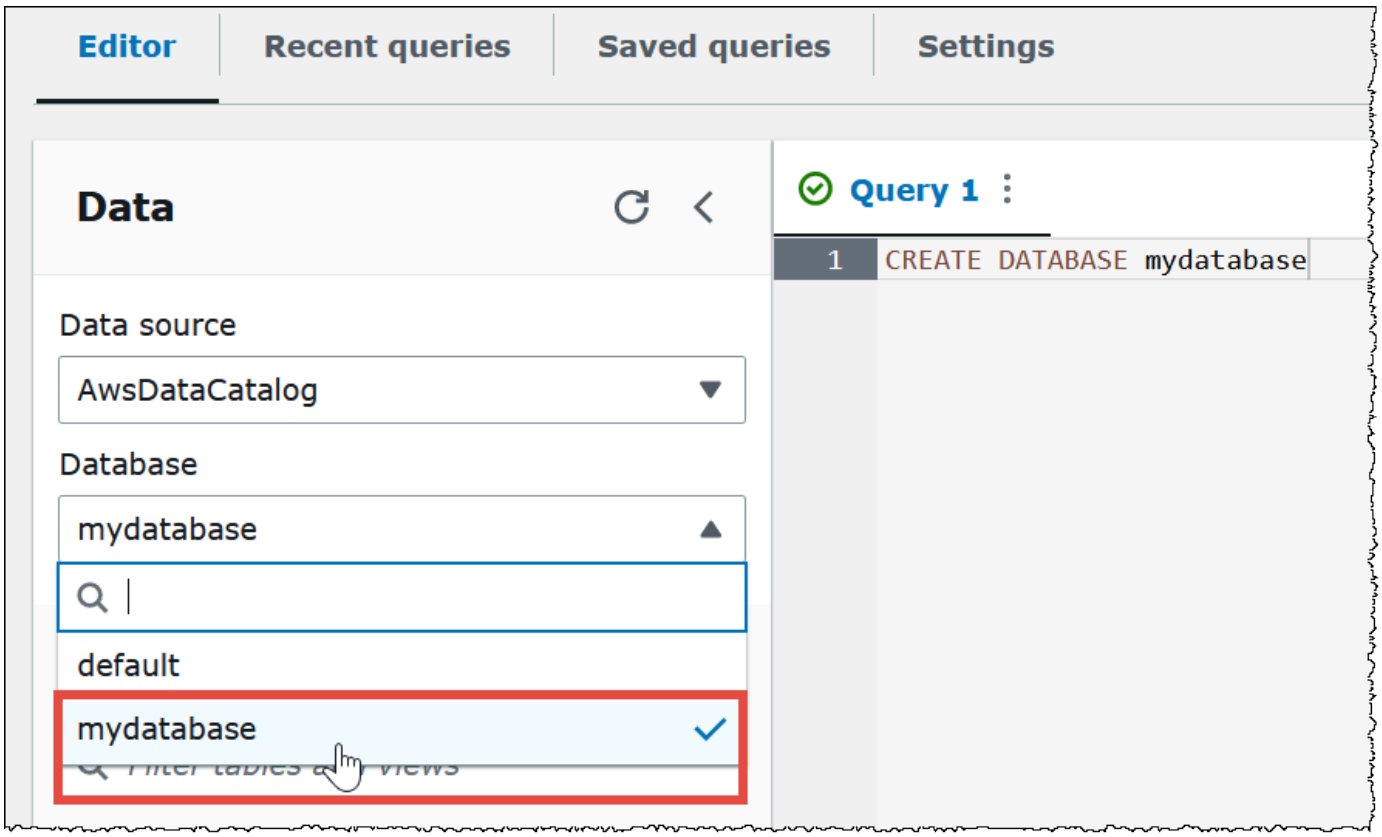
7. Sulla destra del pannello di navigazione, è possibile utilizzare l'editor di query Athena per inserire ed eseguire query e istruzioni.



8. Per creare un database denominato `mydatabase`, immettere la seguente istruzione CREATE DATABASE.

```
CREATE DATABASE mydatabase
```

9. Scegli Run (Esegui) o premi **`Ctrl+ENTER`**.
10. Dall'elenco Database sulla sinistra, scegli `mydatabase` per renderlo il database corrente.



Fase 2: creare una tabella

Ora che hai un database, puoi creare una tabella Athena. La tabella che creerai si baserà su esempi di dati di CloudFront log Amazon nella località `s3://athena-examples-`*`myregion`*`/cloudfront/plaintext/` in cui *`myregion`* è la tua attuale Regione AWS posizione.

I dati di log di esempio sono in formato TSV (Tab-Separated Values), il che significa che un carattere di tabulazione viene utilizzato come delimitatore per separare i campi. I dati vengono mostrati come nell'esempio seguente. Per la leggibilità, le tabulazioni nell'estratto sono state convertite in spazi e il campo finale è stato abbreviato.

```
2014-07-05 20:00:09 DFW3 4260 10.0.0.15 GET eabcd12345678.cloudfront.net /test-
image-1.jpeg 200 - Mozilla/5.0[...]
2014-07-05 20:00:09 DFW3 4252 10.0.0.15 GET eabcd12345678.cloudfront.net /test-
image-2.jpeg 200 - Mozilla/5.0[...]
2014-07-05 20:00:10 AMS1 4261 10.0.0.15 GET eabcd12345678.cloudfront.net /test-
image-3.jpeg 200 - Mozilla/5.0[...]
```

Per consentire ad Athena di leggere questi dati, puoi creare una `CREATE EXTERNAL TABLE` dichiarazione semplice come la seguente. L'istruzione che crea la tabella definisce le colonne che mappano i dati, specifica la modalità di delimitazione dei dati e specifica la posizione Amazon S3 che contiene i dati di esempio. Tieni presente che, poiché Athena prevede di scansionare tutti i file in una cartella, la `LOCATION` clausola specifica una posizione della cartella Amazon S3, non un file specifico.

Non utilizzate ancora questo esempio in quanto presenta un'importante limitazione che verrà spiegata a breve.

```
CREATE EXTERNAL TABLE IF NOT EXISTS cloudfront_logs (
  `Date` DATE,
  Time STRING,
  Location STRING,
  Bytes INT,
  RequestIP STRING,
  Method STRING,
  Host STRING,
  Uri STRING,
  Status INT,
  Referrer STRING,
  ClientInfo STRING
  )
  ROW FORMAT DELIMITED
  FIELDS TERMINATED BY '\t'
  LINES TERMINATED BY '\n'
  LOCATION 's3://athena-examples-my-region/cloudfront/plaintext/';
```

L'esempio crea una tabella denominata `cloudfront_logs` e specifica un nome e un tipo di dati per ogni campo. Questi campi diventano le colonne nella tabella. Poiché `date` è una parola riservata, viene eliminata con caratteri backtick (`). `ROW FORMAT DELIMITED`significa che Athena utilizzerà una libreria predefinita chiamata LazySimpleSerDeper eseguire l'effettivo lavoro di analisi dei dati. L'esempio specifica inoltre che i campi sono separati da tabulazioni (`FIELDS TERMINATED BY '\t'`) e che ogni record nel file termina con un carattere di nuova riga (`LINES TERMINATED BY '\n`). Infine, la clausola `LOCATION` specifica il percorso in Amazon S3 in cui si trovano i dati effettivi da leggere.

Se disponi di dati personalizzati separati da tabulazioni o virgole, puoi utilizzare un'`CREATE TABLE`istruzione come nell'esempio appena presentato, purché i campi non contengano informazioni

annidate. Tuttavia, se una colonna del genere contiene informazioni annidate `ClientInfo` che utilizzano un delimitatore diverso, è necessario un approccio diverso.

Estrazione di dati dal campo ClientInfo

Guardando i dati di esempio, ecco un esempio completo del campo `ClientInfo` finale:

```
Mozilla/5.0%20(Android;%20U;%20Windows%20NT%205.1;%20en-US;
%20rv:1.9.0.9)%20Gecko/2009040821%20IE/3.0.9
```

Come puoi vedere, questo campo è multivalore. Poiché l'`CREATE TABLE`istruzione di esempio appena presentata specifica le tabulazioni come delimitatori di campo, non è possibile suddividere i componenti separati all'interno del `ClientInfo` campo in colonne separate. Pertanto, è necessaria una nuova `CREATE TABLE` dichiarazione.

Per creare colonne dai valori all'interno del `ClientInfo` campo, puoi usare un'espressione regolare (regex) che contiene gruppi regex. I gruppi regex specificati diventano colonne di tabella separate. Per usare una regex nella tua istruzione `CREATE TABLE`, utilizzare la sintassi simile alla seguente. Questa sintassi indica ad Athena di utilizzare la libreria Regex SerDe e l'espressione regolare specificata.

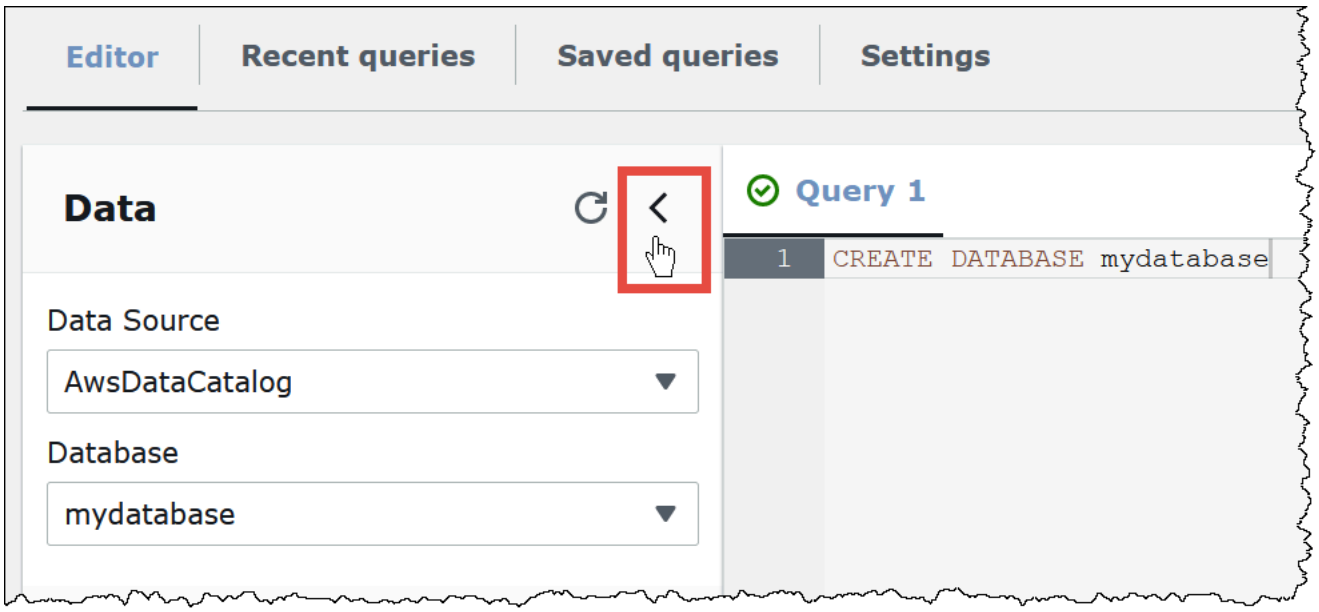
```
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
  WITH SERDEPROPERTIES ("input.regex" = "regular_expression")
```

Le espressioni regolari possono essere utili per la creazione di tabelle da dati CSV o TSV complessi, ma possono essere difficili da scrivere e gestire. Fortunatamente, ci sono altre librerie che è possibile utilizzare per formati come JSON, Parquet e ORC. Per ulteriori informazioni, consulta Formati di SerDe e di dati supportati.

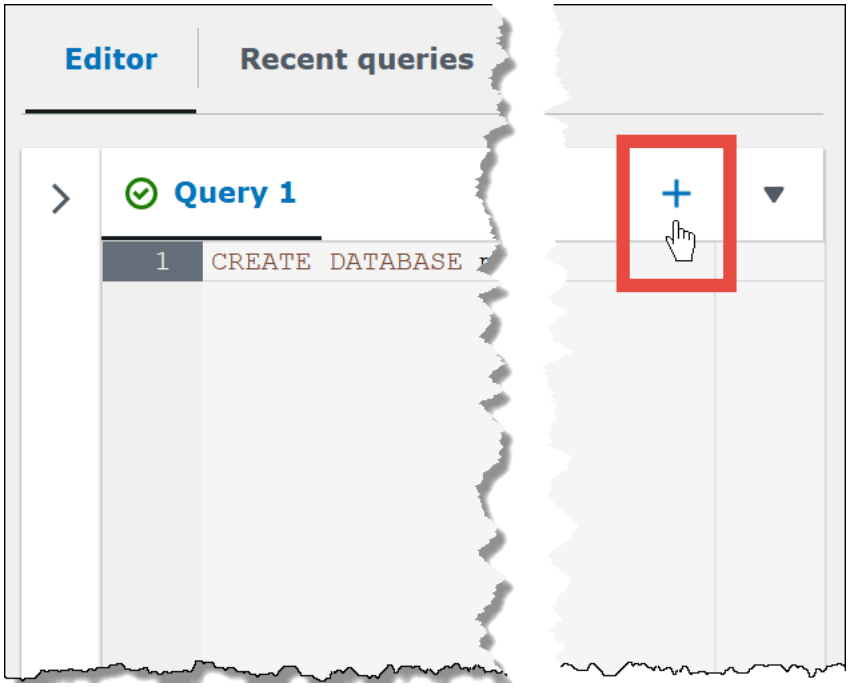
Ora è possibile creare la tabella nell'editor di query Athena. L'istruzione `CREATE TABLE` e la regex ti vengono forniti.

Per creare una tabella in Athena

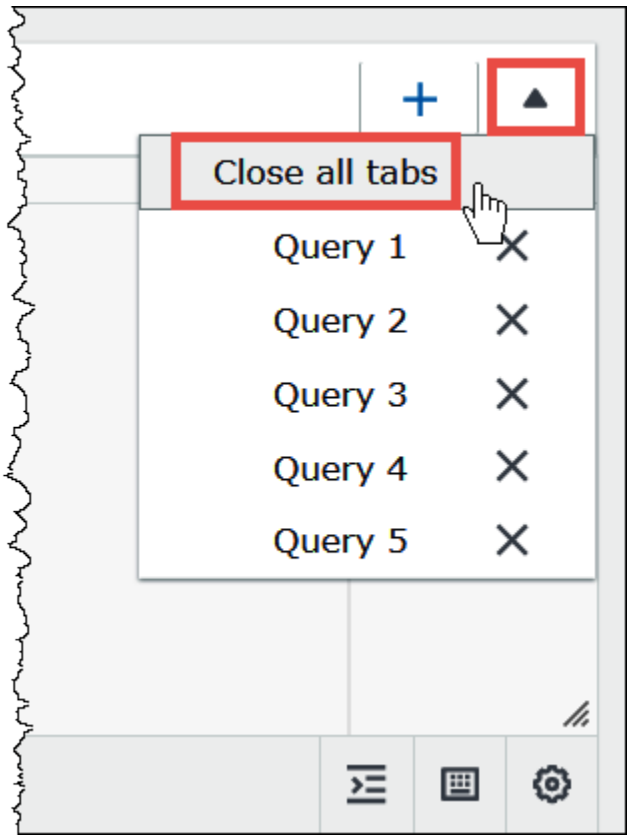
1. Nel pannello di navigazione, in Database, assicurarsi che `mydatabase` sia selezionato.
2. Per avere più spazio nell'editor di query, è possibile scegliere l'icona a forma di freccia e comprimere il pannello di navigazione.



3. Scegli il segno più (+) nell'editor delle query per creare una scheda per una nuova query. È possibile avere fino a dieci schede di query aperte contemporaneamente.



4. Per chiudere una o più schede di query, scegli la freccia accanto al segno più. Per chiudere tutte le schede contemporaneamente, scegliere la freccia, quindi scegli Close all tabs (Chiudi tutte le schede).



5. Nel riquadro delle query inserire la seguente istruzione `CREATE EXTERNAL TABLE`. La regex suddivide le informazioni sul sistema operativo, sul browser e sulla versione del browser dal campo `ClientInfo` nei dati di log.

> **Note**
>
> L'espressione regolare utilizzata nell'esempio seguente è progettata per funzionare con i dati di CloudFront log di esempio disponibili pubblicamente nella posizione `athena-examples` Amazon S3 ed è solo illustrativa. Per altre up-to-date espressioni regolari che interrogano file di log standard e in tempo reale CloudFront , consulta. Interrogazione dei log di Amazon CloudFront

```
CREATE EXTERNAL TABLE IF NOT EXISTS cloudfront_logs (
  `Date` DATE,
  Time STRING,
  Location STRING,
  Bytes INT,
  RequestIP STRING,
  Method STRING,
```

```
  Host STRING,
  Uri STRING,
  Status INT,
  Referrer STRING,
  os STRING,
  Browser STRING,
  BrowserVersion STRING
  )
  ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
  WITH SERDEPROPERTIES (
  "input.regex" = "^(?!#)([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s
+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+[^\(]+[\(]([^\;]+).*\%20([^
\/]+)[\/](.*)$"
  ) LOCATION 's3://athena-examples-myregion/cloudfront/plaintext/';
```

6. Nell'`LOCATION`istruzione, sostituisci *`myregion`* con Regione AWS quello che stai utilizzando attualmente (ad esempio,). `us-west-1`
7. Seleziona Esegui.

   Viene creata la tabella `cloudfront_logs` che appare nell'elenco Tables (Tabelle) del database `mydatabase`.

Fase 3: Esecuzione di query sui dati

Dopo aver creato la tabella `cloudfront_logs` in Athena in base ai dati in Amazon S3, è possibile eseguire le query sulla tabella e visualizzarne i risultati in Athena. Per ulteriori informazioni sull'utilizzo di SQL in Athena, consultare Documentazione di riferimento SQL per Athena.

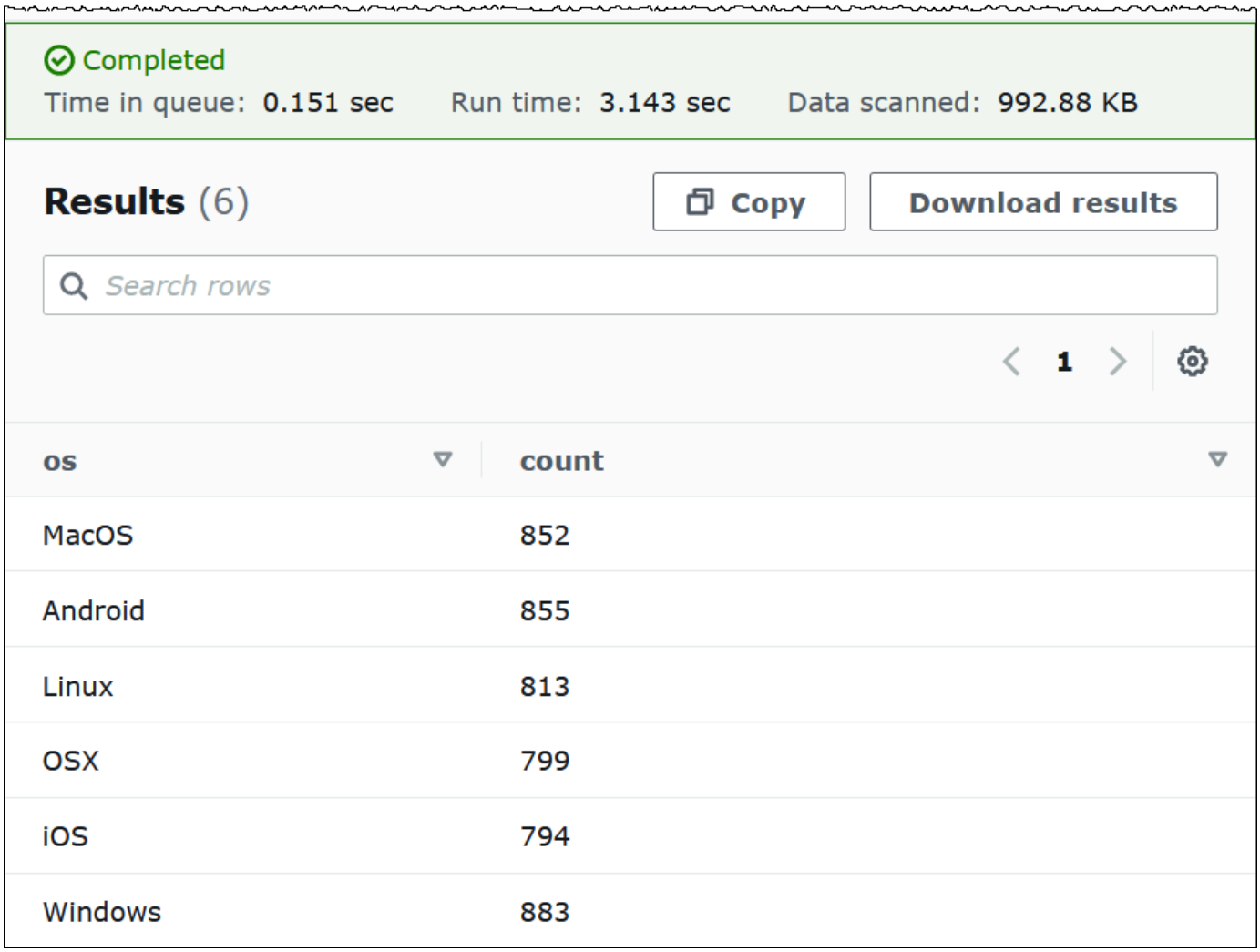
Per eseguire una query

1. Scegli il segno più (+) per aprire una nuova scheda di query e inserire la seguente istruzione SQL nel riquadro delle query.

   ```
   SELECT os, COUNT(*) count
   FROM cloudfront_logs
   WHERE date BETWEEN date '2014-07-05' AND date '2014-08-05'
   GROUP BY os
   ```

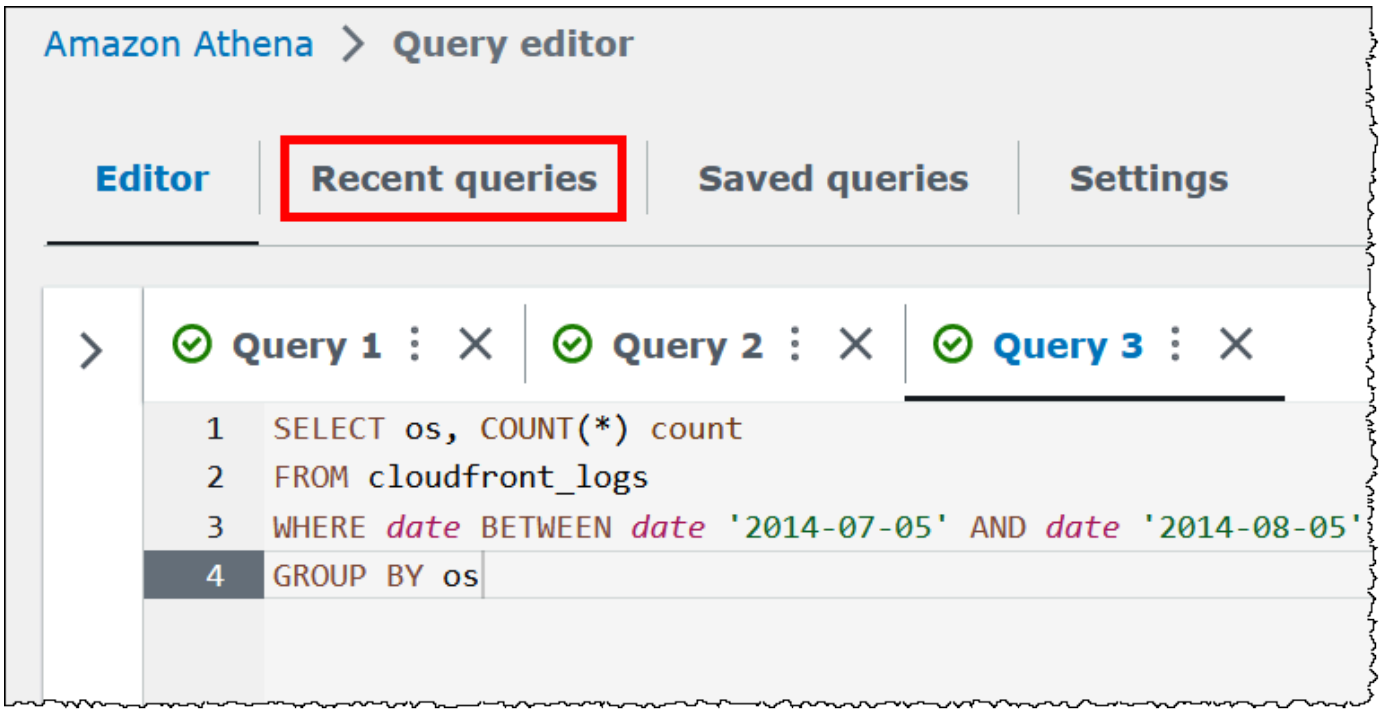
2. Seleziona Esegui.

   I risultati sono simili ai seguenti:

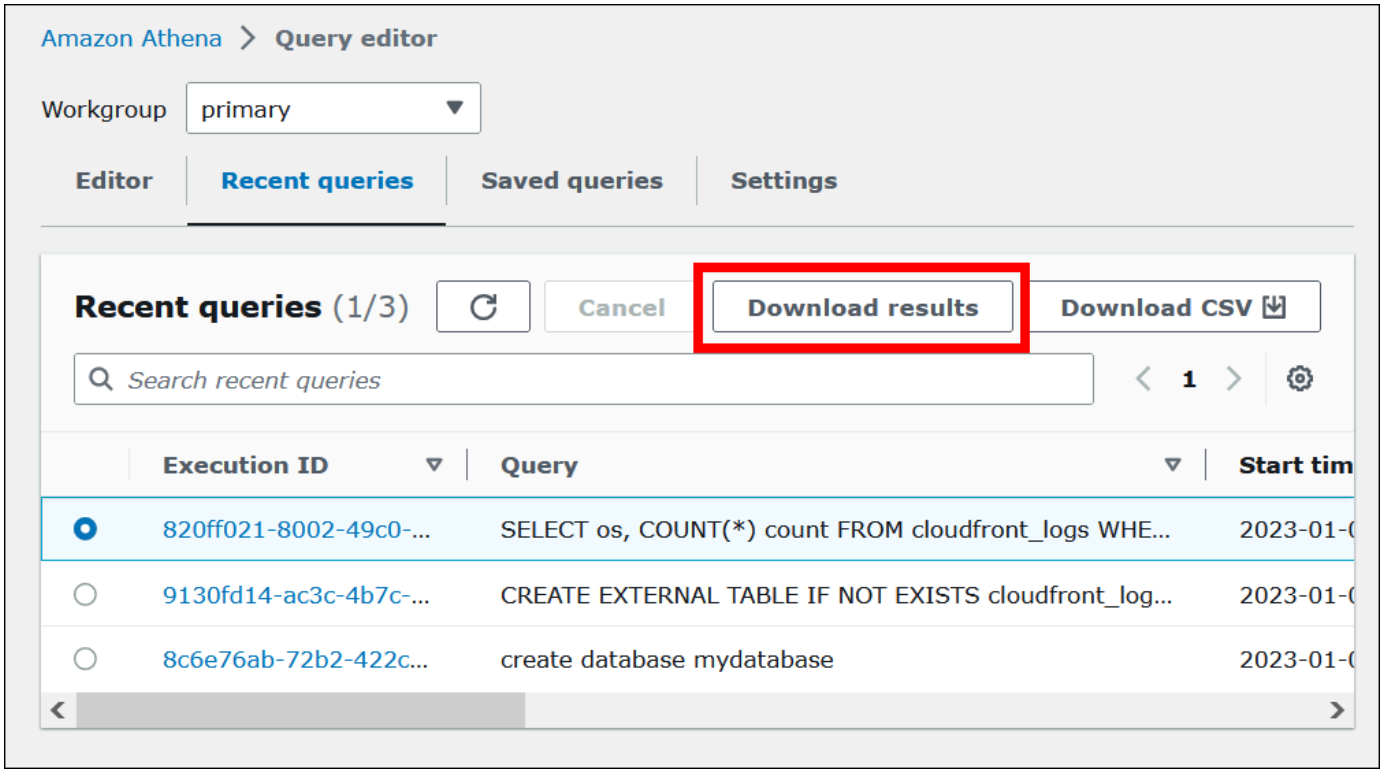


3. Per salvare i risultati della query in un file `.csv`, scegli Download results (Scarica risultati).

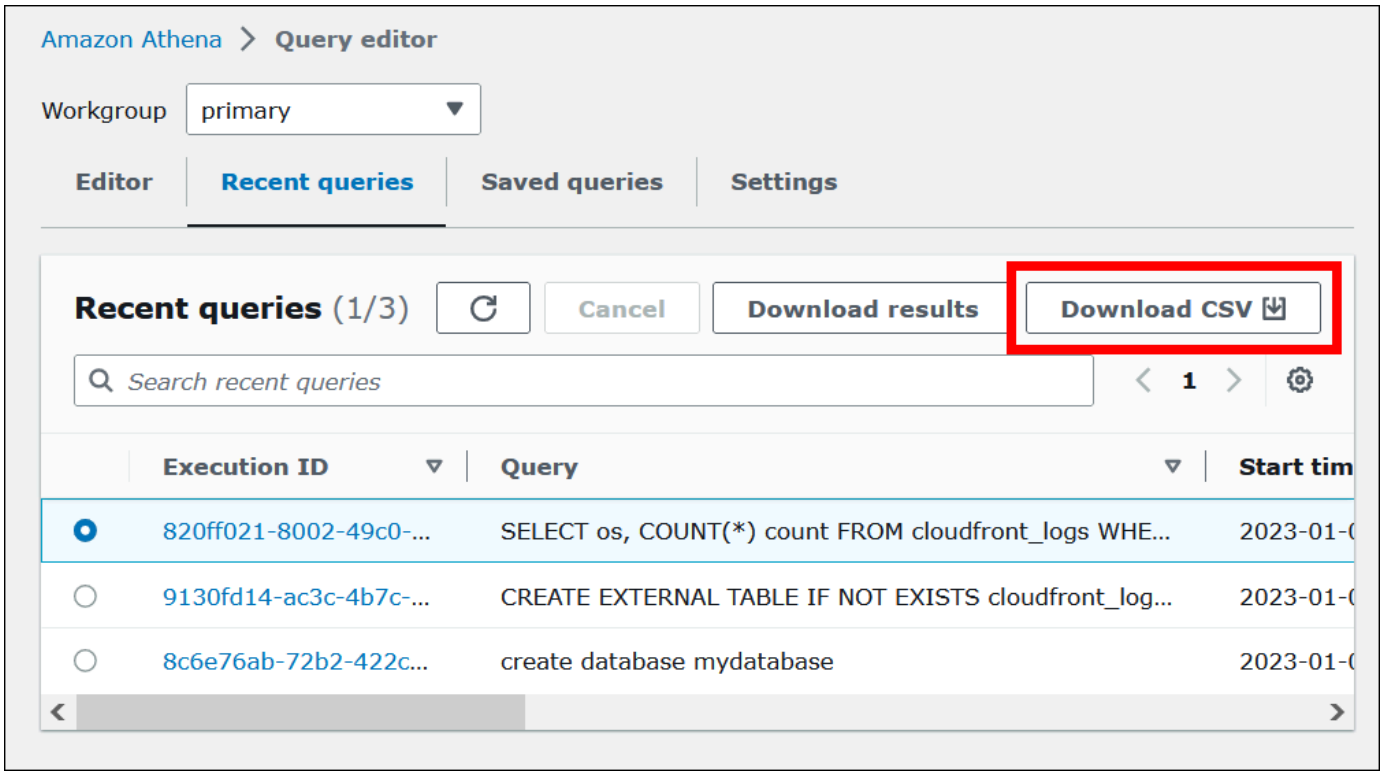
4. Per visualizzare o eseguire le query precedenti, scegli la scheda Recent queries (Query recenti).



5. Per scaricare i risultati di una query precedente dalla scheda Recent queries (Query recenti), seleziona la query, quindi scegli Download results (Scarica risultati). Le query vengono conservate per 45 giorni.



6. Per scaricare una o più stringhe di query SQL recenti in un file CSV, scegli Download CSV (Scarica CSV).



Per ulteriori informazioni, consulta Utilizzo dei risultati delle query, delle query recenti e dei file di output.

Salvataggio delle query

Puoi salvare le query create o modificate nell'editor di query con un nome. Athena memorizza queste query sulla scheda Query salvate. Puoi utilizzare la scheda Query salvate per richiamare, eseguire, rinominare o eliminare le query salvate. Per ulteriori informazioni, consulta Utilizzo di query salvate.

Scelte rapide da tastiera e suggerimenti di digitazione

L'editor di query Athena offre numerose scorciatoie da tastiera per azioni come l'esecuzione di una query, la formattazione di una query, le operazioni di riga e la ricerca e sostituzione. Per ulteriori informazioni e un elenco completo delle scorciatoie, consulta Come migliorare la produttività utilizzando le scorciatoie da tastiera nell'editor di query di Amazon Athena nel Blog sui Big Data di AWS .

L'editor di query Athena supporta suggerimenti di codice typeahead per un'esperienza di creazione di query più rapida. Per aiutarti a scrivere query SQL con maggiore precisione e maggiore efficienza, offre le seguenti funzionalità:

- Durante la digitazione, vengono visualizzati suggerimenti in tempo reale per parole chiave, variabili locali, frammenti ed elementi del catalogo.
- Quando digiti il nome di un database o di tabella seguito da un punto, l'editor visualizza in maniera pratica un elenco di tabelle o colonne tra cui scegliere.
- Quando passi il mouse su un suggerimento di snippet, una sinossi mostra una breve panoramica della sintassi e dell'utilizzo dello snippet.
- Per migliorare la leggibilità del codice, anche le parole chiave e le relative regole di evidenziazione sono state aggiornate per allinearle alla sintassi più recente di Trino e Hive.

Questa caratteristica viene attivata per impostazione predefinita. Per abilitare o disabilitare la funzionalità, utilizza le Preferenze editor di codice (icona a forma di ingranaggio) in basso a destra nella finestra dell'editor di query.

Connessione ad altre origini dati

Questo tutorial ha utilizzato un'origine dati Amazon S3 in formato CSV. Per informazioni sull'utilizzo di Athena con AWS Glue, vedere. Utilizzo AWS Glue per connettersi a sorgenti dati in Amazon S3 È possibile inoltre connettere Athena a una varietà di origini dati utilizzando i driver ODBC e JDBC, i metastore Hive esterni e i connettori di origini dati Athena. Per ulteriori informazioni, consulta Connessione alle origini dati.

Connessione alle origini dati

Puoi utilizzare Amazon Athena per eseguire query sui dati archiviati in posizioni e formati differenti in un set di dati. Il formato di questo set di dati potrebbe essere CSV, JSON, Avro, Parquet o un altro.

Le tabelle e i database che utilizzi in Athena per eseguire query sono basati su metadati. I metadati sono dati relativi ai dati sottostanti nel set di dati. Il modo in cui i metadati descrivono il set di dati viene chiamato schema. Ad esempio, un nome di tabella, i nomi di colonna nella tabella e il tipo di dati di ogni colonna sono uno schema, salvato come metadati, che descrive un set di dati sottostante. In Athena, un sistema per l'organizzazione dei metadati viene chiamato un catalogo dati o un metastore. La combinazione di un set di dati e del catalogo dati che lo descrive viene chiamata origine dati.

La relazione di metadati rispetto a un set di dati sottostante dipende dal tipo di origine dati utilizzato. Le origini dati relazionali come MySQL, PostgreSQL e SQL Server integrano strettamente i metadati con il set di dati. In questi sistemi, i metadati vengono spesso scritti al momento della scrittura dei dati. Altre fonti di dati, come quelle create con Hive, consentono di definire i metadati on-the-fly quando si legge il set di dati. Il set di dati può essere in diversi formati, ad esempio CSV, JSON, Parquet o Avro.

Athena supporta nativamente il. AWS Glue Data Catalog AWS Glue Data Catalog È un catalogo di dati basato su altri set di dati e fonti di dati come Amazon S3, Amazon Redshift e Amazon DynamoDB. Puoi inoltre connettere Athena ad altre origini dati utilizzando diversi connettori.

Argomenti

Integrazione con AWS Glue

AWS Glueè un ETL completamente gestito (estrazione, trasformazione e caricamento). Servizio AWS Una delle sue capacità chiave è quella di analizzare e classificare i dati. Puoi utilizzare AWS Glue i crawler per dedurre automaticamente lo schema di database e tabelle dai tuoi dati in Amazon S3 e archiviare i metadati associati in. AWS Glue Data Catalog

Athena lo utilizza AWS Glue Data Catalog per archiviare e recuperare i metadati delle tabelle per i dati Amazon S3 nel tuo account Amazon Web Services. I metadati della tabella consentono al motore di query Athena di sapere come trovare, leggere ed elaborare i dati che si desidera interrogare.

Per creare uno schema di database e tabelle in AWS Glue Data Catalog, puoi eseguire un AWS Glue crawler dall'interno di Athena su un'origine dati oppure puoi eseguire query DDL (Data Definition Language) direttamente nell'Athena Query Editor. Quindi, utilizzando il database e lo schema di tabella creato, puoi utilizzare le query DML (Data Manipulation) in Athena per eseguire query sui dati.

È possibile registrarne uno AWS Glue Data Catalog da un account diverso dal proprio. Dopo aver configurato le autorizzazioni IAM richieste per AWS Glue, puoi utilizzare Athena per eseguire query tra account. Per ulteriori informazioni, consulta Accesso tra account ai cataloghi dati AWS Glue.

Per ulteriori informazioni su AWS Glue Data Catalog, consulta Data Catalog and crawler nella Developer Guide. AWS GlueAWS Glue

Per un articolo illustrativo che mostra come utilizzare AWS Glue e Athena per elaborare i dati XML, consulta Elaborare e analizzare file XML altamente annidati e di grandi dimensioni utilizzando AWS Glue Amazon Athena nel Big Data Blog. AWS

Si applicano costi separati a. AWS Glue Per ulteriori informazioni, consultare Prezzi di AWS Glue.

Argomenti

- Utilizzo AWS Glue per connettersi a sorgenti dati in Amazon S3
- Registrazione e AWS Glue Data Catalog da un altro account
- Procedure consigliate per l'utilizzo di Athena con AWS Glue
- Utilizzo di AWS CLI per ricreare un AWS Glue database e le relative tabelle

Utilizzo AWS Glue per connettersi a sorgenti dati in Amazon S3

Athena può connettersi ai dati archiviati in Amazon S3 utilizzando AWS Glue Data Catalog per archiviare metadati come nomi di tabelle e colonne. Dopo aver effettuato la connessione, i database, le tabelle e le visualizzazioni appaiono nell'editor della query di Athena.

Per definire le informazioni sullo schema AWS Glue da utilizzare, puoi creare un AWS Glue crawler per recuperare le informazioni automaticamente oppure puoi aggiungere manualmente una tabella e inserire le informazioni sullo schema.

Creare un crawler AWS Glue

Puoi creare un crawler partendo dalla console Athena e quindi utilizzando la console AWS Glue in un modo integrato. Quando crei il crawler, specifica una posizione dei dati in Amazon S3 per eseguire la ricerca per indicizzazione.

Per creare un crawler AWS Glue partendo dalla console Athena

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.

2. Nell'editor di query, accanto a Tabelle e visualizzazioni, scegli Crea e quindi scegli Crawler AWS Glue .
3. Sulla paginaAggiungi crawler della console AWS Glue, segui i passaggi per creare un crawler. Per ulteriori informazioni, consulta Using AWS Glue Crawler in questa guida e Population the nella Developer Guide. AWS Glue Data CatalogAWS Glue

> **Note**
>
> Athena non riconosce i pattern di esclusione specificati per un AWS Glue crawler. Ad esempio, se disponi di un bucket Amazon S3 che contiene i file `.csv` e `.json` ed escludi i file `.json` dal crawler, Athena esegue query su entrambi i gruppi di file. Per evitare ciò, posizionare i file che si desidera escludere in una posizione diversa.

Aggiunta di una tabella utilizzando un modulo

Nella procedura seguente viene illustrato come utilizzare la console Athena per aggiungere una tabella utilizzando il modulo Crea tabella dai dati del bucket S3.

Per aggiungere una tabella e immettere le informazioni sullo schema utilizzando un modulo

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Nell'editor di query, accanto a Tables and views (Tabelle e visualizzazioni), scegli Create (Crea) e quindi scegli S3 bucket data. (Dati del bucket S3).
3. Nel modulo Create Table From S3 bucket data (Crea tabella dai dati del bucket S3), per Table name (Nome tabella), inserisci un nome per la tabella.
4. Per Database configuration (Configurazione database), scegli un database esistente o creane uno nuovo.
5. Per Posizione del set di dati di input, specificare il percorso in Amazon S3 alla cartella contenente il set di dati che si desidera elaborare. Non includere un nome di file nel percorso. Athena esegue la scansione di tutti i file contenuti nella cartella specificata. Se i dati sono già partizionati (ad esempio,

   s3://DOC-EXAMPLE-BUCKET/logs/year=2004/month=12/day=11/), inserisci solo il percorso di base (ad esempio, s3://DOC-EXAMPLE-BUCKET/logs/).
6. Per Data Format (Formato dei dati), scegli tra le seguenti opzioni:

- In Table type (Tipo di tabella), scegli Apache Hive, Apache Iceberg o Delta Lake. Per impostazione predefinita, Athena utilizza il tipo di tabella Apache Hive. Per informazioni sull'esecuzione di query sulle tabelle Apache Iceberg in Athena, consulta la pagina Utilizzo di tabelle Apache Iceberg. Per informazioni sull'utilizzo delle tabelle Delta Lake in Athena, consulta la pagina Esecuzione di query sulle tabelle Delta Lake di Linux Foundation.
- Per File format (Formato file), scegli il formato di file o log in cui si trovano i dati.
  - Per l'opzione Text File with Custom Delimiters (File di testo con delimitatori personalizzati), specificare un Field terminator (Carattere di terminazione del campo) (ovvero, un delimitatore di colonna). Facoltativamente, puoi specificare un Collection terminator (carattere di terminazione della raccolta) che segna la fine di un tipo di array o un Collection terminator (carattere di terminazione della raccolta) che segna la fine di un tipo di dati mappa.
- SerDe libreria — Una libreria SerDe (serializzatore-deserializzatore) analizza un particolare formato di dati in modo che Athena possa creare una tabella corrispondente. Per la maggior parte dei formati, viene scelta automaticamente una libreria predefinita. SerDe Per i formati seguenti, scegli una libreria in base alle tue esigenze:
  - Apache Web Logs: scegli la libreria RegexSerDeo GrokSerDe. Per RegexSerDe, fornisci un'espressione regolare nella casella di definizione Regex. Per GrokSerDe, fornite una serie di espressioni regolari denominate per la `input.format` SerDe proprietà. Le espressioni regolari denominate sono più facili da leggere e gestire rispetto alle espressioni regolari. Per ulteriori informazioni, consulta Esecuzione di query sui log Apache archiviati in Amazon S3.
  - CSV: scegli LazySimpleSerDese i tuoi dati separati da virgole non contengono valori racchiusi tra virgolette o se utilizzano il formato. `java.sql.Timestamp` Scegli SerDeOpenCSV se i tuoi dati includono virgolette o utilizzano il formato numerico UNIX per (ad esempio). `TIMESTAMP` `1564610311` Per ulteriori informazioni, consulta LazySimpleSerDe per CSV, TSV e file delimitati in modo personalizzato e OpenCSV per l'elaborazione di file SerDe CSV.
  - JSON: scegli la libreria JSON OpenX o SerDe Hive. Entrambi i formati prevedono che ogni documento JSON si trovi su una singola riga di testo e che i campi non siano separati da caratteri di nuova riga. OpenX SerDe offre alcune proprietà aggiuntive. Per ulteriori informazioni su queste proprietà, consultare OpenX JSON SerDe. Per informazioni su Hive SerDe, vedere. JSON Hive SerDe

  Per ulteriori informazioni sull'uso delle SerDe librerie in Athena, vedere. Formati di SerDe e di dati supportati

7. Per SerDe le proprietà, aggiungi, modifica o rimuovi proprietà e valori in base alla SerDe libreria che stai utilizzando e ai tuoi requisiti.

   - Per aggiungere una SerDe proprietà, scegli Aggiungi SerDe proprietà.
   - Nel campo Name (Nome), immetti il nome della proprietà.
   - Nel campo Value (Valore), immetti un valore per la proprietà.
   - Per rimuovere una SerDe proprietà, scegli Rimuovi.

8. Per Table properties (Proprietà tabella), scegli o modifica le proprietà della tabella in base alle tue esigenze.

   - Per Write compression (Compressione per la scrittura), scegli un'opzione di compressione. La disponibilità dell'opzione di compressione in scrittura e delle altre opzioni di compressione disponibili dipende dal formato dei dati. Per ulteriori informazioni, consulta Supporto della compressione in Athena.
   - Per Encryption (Crittografia), seleziona Encrypted data set (Set di dati crittografati) se i dati sottostanti sono crittografati in Amazon S3. Questa opzione imposta la proprietà della tabella `has_encrypted_data` su true nell'istruzione `CREATE TABLE`.

9. Per Column details (Dettagli della colonna), inserisci i nomi e i tipi di dati delle colonne che desideri aggiungere alla tabella.

   - Per aggiungere più colonne una alla volta, scegliere Add a column (Aggiungi una colonna).
   - Per aggiungere rapidamente più colonne, scegliere Bulk add columns (Aggiungi colonne in blocco). Nella casella di testo, immettere un elenco separato da virgole di colonne nel formato *`column_name data_type`*, *`column_name data_type`*[,...] e quindi scegliere Aggiungi.

10. (Facoltativo) Per Partition details (Dettagli delle partizioni), aggiungi uno o più nomi di colonna e tipi di dati. Il partizionamento tiene insieme i dati correlati in base ai valori delle colonne e consente di ridurre la quantità di dati analizzati per query. Per informazioni sulle partizioni, consulta Partizionamento dei dati in Athena.

11. (Facoltativo) In Bucketing puoi specificare una o più colonne contenenti le righe da raggruppare e inserire tali righe in più bucket. Ciò consente di eseguire query solo sul bucket che si desidera leggere quando si specifica il valore delle colonne con bucket.

    - Per Buckets (Bucket), seleziona una o più colonne con un numero elevato di valori univoci (ad esempio, una chiave primaria) utilizzate di frequente per filtrare i dati nelle query.

- In Number of buckets (Numero di bucket), immetti un numero che consenta ai file di avere dimensioni ottimali. Per ulteriori informazioni, consulta la sezione I 10 migliori consigli per ottimizzare le prestazioni di Amazon Athena nel blog AWS Big Data.
- Per specificare le colonne con bucket, l'istruzione `CREATE TABLE` utilizza la sintassi seguente:

```
CLUSTERED BY (bucketed_columns) INTO number_of_buckets BUCKETS
```

> **Note**
>
> L'opzione Bucketing non è disponibile per i tipi di tabella Iceberg.

12. Nella casella Preview table query (Anteprima della query della tabella) viene visualizzata l'istruzione `CREATE TABLE` generata dalle informazioni immesse nel modulo. L'istruzione di anteprima non può essere modificata direttamente. Per modificare l'istruzione, modifica i campi del modulo sopra l'anteprima oppure crea direttamente l'istruzione nell'editor di query anziché utilizzare il modulo.
13. Scegli Create table (Crea tabella) per eseguire l'istruzione generata nell'editor di query e crea la tabella.

Registrazione e AWS Glue Data Catalog da un altro account

Puoi utilizzare la funzionalità di AWS Glue catalogo multiaccount di Athena per registrare un AWS Glue catalogo da un account diverso dal tuo. Dopo aver configurato le autorizzazioni IAM richieste per AWS Glue e registrato il catalogo come risorsa Athena `DataCatalog`, puoi utilizzare Athena per eseguire query tra account. Per informazioni sulla configurazione delle autorizzazioni richieste, consulta Accesso tra account ai cataloghi dati AWS Glue.

La procedura seguente illustra come utilizzare la console Athena per configurare AWS Glue Data Catalog in un account Amazon Web Services diverso dal tuo come origine dati.

Per registrare un annuncio AWS Glue Data Catalog da un altro account

1. Segui la procedura riportata in Accesso tra account ai cataloghi dati AWS Glue per verificare di disporre delle autorizzazioni per eseguire query sul catalogo dati nell'altro account.
2. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
3. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

4. Scegli Data sources (Origini dati).
5. Nell'angolo in alto a destra, scegli Create data source (Crea origine dei dati).
6. Nella pagina Scegli un'origine dati, per Origini dati, scegli S3 - AWS Glue Data Catalog, quindi scegli Avanti.
7. Nella pagina Inserisci dettagli origine dati, nella sezione AWS Glue Data Catalog, per Scegli un AWS Glue Data Catalog, seleziona AWS Glue Data Catalog in un altro account.
8. Per Dataset details (Dettagli del set di dati), fornisci le seguenti informazioni:
   - Nome origine dati: inserisci il nome che desideri utilizzare nelle query SQL per fare riferimento al catalogo dati nell'altro account.
   - Descrizione — (Facoltativo) Inserisci una descrizione del catalogo dati nell'altro account.
   - ID catalogo — Inserisci l'ID account Amazon Web Services a 12 cifre dell'account a cui appartiene il catalogo dati. L'ID dell'account Amazon Web Services è l'ID del catalogo.
9. (Facoltativo) Per Tag, inserisci le coppie chiave-valore da associare all'origine dati. Per ulteriori informazioni sui tag, consulta Assegnazione di tag alle risorse Athena.
10. Seleziona Successivo.
11. Nella pagina Review and create (Rivedi e crea), esamina le informazioni inserite, quindi scegli Create data source (Crea origine dei dati). La pagina Data source details (Dettagli sull'origine dei dati) elenca i database e i tag per il catalogo dati registrato.
12. Scegli Data sources (Origini dati). Il catalogo dati che hai registrato è elencato nella colonna Data source name (Nome origine dei dati).

13. Per visualizzare o modificare le informazioni sul catalogo dati, scegli il catalogo, quindi scegli Actions (Operazioni), Edit (Modifica).
14. Per eliminare il nuovo catalogo dati, scegli il catalogo, quindi scegli Actions (Operazioni), Delete (Elimina).

Per ulteriori informazioni, consulta Eseguire query su più account AWS Glue Data Catalog utilizzando Amazon Athena nel blog AWS sui Big Data.

Procedure consigliate per l'utilizzo di Athena con AWS Glue

Quando si utilizza Athena con AWS Glue Data Catalog, è possibile utilizzare AWS Glue per creare database e tabelle (schema) da interrogare in Athena oppure è possibile utilizzare Athena per creare schemi e quindi utilizzarli nei servizi correlati. AWS Glue Questo argomento fornisce alcune osservazioni e best practice da seguire quando si utilizza uno dei due metodi.

Athena utilizza in background Trino per elaborare le dichiarazioni DML e Hive per elaborare le dichiarazioni DDL che creano e modificano gli schemi. Con queste tecnologie, ci sono un paio di convenzioni da seguire affinché Athena AWS Glue e io lavoriamo bene insieme.

In questo argomento

- Utilizzo di processi ETL per ottimizzare le prestazioni delle query
- Conversione di tipi di dati SMALLINT e TINYINT su INT durante la conversione in ORC
- Automazione dei lavori per ETL AWS Glue

Nomi database, tabella e colonne

Quando crei uno schema AWS Glue per eseguire una query in Athena, considera quanto segue:

- I caratteri accettabili per i nomi di database, i nomi delle tabelle e i nomi delle colonne AWS Glue devono essere una stringa UTF-8 e devono essere scritti in minuscolo. Tieni presente che Athena riduce automaticamente tutti i nomi maiuscoli nelle query DDL quando crea database, tabelle o colonne. La lunghezza della stringa non deve essere inferiore a 1 o superiore a 255 byte. I caratteri che possono essere utilizzati includono gli spazi.
- Attualmente è possibile inserire degli spazi iniziali all'inizio dei nomi. Poiché questi spazi iniziali possono essere difficili da rilevare e possono causare problemi di usabilità dopo la creazione, evitate di creare inavvertitamente nomi di oggetti con spazi iniziali.
- Se utilizzi un AWS::Glue::Database AWS CloudFormation modello per creare un AWS Glue database e non specifichi un nome di database, genera AWS Glue automaticamente un nome di database nel formato *resource_name–random_string* che non è compatibile con Athena.
- È possibile utilizzare AWS Glue Catalog Manager per rinominare le colonne, ma non i nomi delle tabelle o dei database. Per ovviare a questa limitazione, è necessario utilizzare una definizione del vecchio database per creare un database con il nuovo nome. Quindi si utilizzano le definizioni delle tabelle del vecchio database per ricreare le tabelle nel nuovo database. Per fare ciò, puoi usare AWS CLI o AWS Glue SDK. Per le fasi, consulta Utilizzo di AWS CLI per ricreare un AWS Glue database e le relative tabelle.

Per ulteriori informazioni su database e tabelle in AWS Glue, consulta Databases and Tables nella AWS Glue Developer Guide.

Utilizzo dei AWS Glue crawler

AWS Glue i crawler aiutano a scoprire lo schema dei set di dati e a registrarli come tabelle nel Data Catalog. AWS Glue I crawler analizzano i dati e ne determinano lo schema. Inoltre, il crawler è in grado di rilevare e registrare le partizioni. Per ulteriori informazioni, consultare Definizione di crawler nella Guida per gli sviluppatori di AWS Glue . Le tabelle dei dati correttamente sottoposte a ricerca per indicizzazione possono essere interrogate da Athena.

> **Note**
>
> Athena non riconosce i pattern di esclusione specificati per un AWS Glue crawler. Ad esempio, se disponi di un bucket Amazon S3 che contiene i file `.csv` e `.json` ed escludi i file `.json` dal crawler, Athena esegue query su entrambi i gruppi di file. Per evitare ciò, posizionare i file che si desidera escludere in una posizione diversa.

Pianificazione di un crawler per mantenere sincronizzati AWS Glue Data Catalog e Amazon S3

AWS Glue i crawler possono essere configurati per essere eseguiti secondo una pianificazione o su richiesta. Per ulteriori informazioni, consulta la sezione relativa alle pianificazioni per processi e crawler nella guida per sviluppatori di AWS Glue .

Se disponi di dati per una tabella partizionata che arrivano a un orario fisso, puoi configurare un AWS Glue crawler da eseguire nei tempi previsti per rilevare e aggiornare le partizioni della tabella. In questo modo, non sarà più necessario eseguire un comando `MSCK REPAIR`, potenzialmente lungo e costoso, né eseguire manualmente un comando `ALTER TABLE ADD PARTITION`. Per ulteriori informazioni, consulta la sezione relativa alle partizioni tabella nella guida per sviluppatori AWS Glue .

Utilizzo di più origini dati con i crawler

Quando un AWS Glue crawler esegue la scansione di Amazon S3 e rileva più directory, utilizza un'euristica per determinare dove si trova la radice di una tabella nella struttura di directory e quali directory sono partizioni per la tabella. Nei casi in cui gli schemi rilevati in due o più directory siano analoghi, il crawler potrebbe trattarli come partizioni invece di tabelle separate. Un metodo per aiutare il crawler a rilevare singole tabelle è aggiungere la directory radice di ciascuna tabella come datastore per il crawler.

Le seguenti partizioni in Amazon S3 sono un esempio:

```
s3://DOC-EXAMPLE-BUCKET/folder1/table1/partition1/file.txt
s3://DOC-EXAMPLE-BUCKET/folder1/table1/partition2/file.txt
s3://DOC-EXAMPLE-BUCKET/folder1/table1/partition3/file.txt
s3://DOC-EXAMPLE-BUCKET/folder1/table2/partition4/file.txt
s3://DOC-EXAMPLE-BUCKET/folder1/table2/partition5/file.txt
```

Se gli schemi per `table1` e `table2` sono simili e una singola origine dati è impostata su `s3://DOC-EXAMPLE-BUCKET/folder1/` in AWS Glue, il crawler potrebbe creare una singola tabella con

due colonne di partizione: una che contiene `table1` e `table2`, e l'altra contenente `partition1` tramite `partition5`.

Per fare in modo che il AWS Glue crawler crei due tabelle separate, imposta il crawler in modo che abbia due origini dati e, come illustrato nella procedura seguente. `s3://DOC-EXAMPLE-BUCKET/folder1/table1/` `s3://DOC-EXAMPLE-BUCKET/folder1/table2`

Per aggiungere un data store S3 a un crawler esistente in AWS Glue

1. Accedi AWS Management Console e apri la AWS Glue console all'indirizzo https://console.aws.amazon.com/glue/.
2. Nel riquadro di navigazione, selezionare Crawlers (Crawler).
3. Scegli il link al tuo crawler, quindi scegli Edit (Modifica).
4. Per Fase 2: Scegli origini dei dati e classificatori, scegli Edit (Modifica).
5. Nella sezione Data sources (Origini dei dati), scegli Add a data source (Aggiungi un'origine dei dati).
6. Nella finestra di dialogo Add data source (Aggiungi origine dei dati), per S3 path (Percorso S3), scegli Browse (Sfoglia).
7. Scegli il bucket che vuoi utilizzare e poi seleziona Choose (Scegli).

   L'origine dei dati che hai aggiunto viene visualizzata nell'elenco Data sources (Origini dei dati).
8. Seleziona Successivo.
9. Nella pagina Configure security settings (Configura impostazioni di sicurezza), crea o scegli un ruolo IAM per il crawler, quindi scegli Next (Avanti).
10. Assicurati che il percorso S3 termini con una barra finale, quindi scegli Add an S3 data source (Aggiungi un'origine dei dati S3).
11. Nella pagina Set output and scheduling (Imposta l'output e la pianificazione), per Output configuration (Configurazione dell'output), scegli il database di destinazione.
12. Seleziona Successivo.
13. Nella pagina Review and update (Verifica e aggiorna), rivedi le scelte che hai fatto. Per modificare un passaggio, scegli Edit (Modifica).
14. Scegli Aggiorna.

Sincronizzazione dello schema di partizione per evitare "HIVE_PARTITION_SCHEMA_MISMATCH"

Per ogni tabella all'interno del AWS Glue Data Catalog con colonne di partizione, lo schema viene archiviato a livello di tabella e per ogni singola partizione all'interno della tabella. Lo schema per le partizioni viene popolato da un AWS Glue crawler basato sul campione di dati che legge all'interno della partizione. Per ulteriori informazioni, consulta Utilizzo di più origini dati con i crawler.

Quando Athena esegue una query, convalida lo schema della tabella e lo schema qualsiasi partizione necessaria per la query. La convalida confronta i tipi di dati di colonna in ordine e verifica che corrispondano per le colonne che si sovrappongono. Ciò impedisce operazioni inattese, come l'aggiunta o la rimozione di colonne dal centro di una tabella. Se Athena rileva che lo schema di una partizione differisce dallo schema della tabella, Athena potrebbe non essere in grado di elaborare la query e restituisce l'errore `HIVE_PARTITION_SCHEMA_MISMATCH`.

Vi sono vari modi per risolvere questo problema. In primo luogo, se i dati sono stati aggiunti accidentalmente, è possibile eliminare i file di dati che generano la differenza tra gli schemi, eliminare la partizione e ripetere il crawling dei dati. In secondo luogo, è possibile eliminare la singola partizione ed eseguire quindi `MSCK REPAIR` all'interno di Athena per ricreare la partizione impiegando lo schema della tabella. Questa seconda opzione funziona solo se si è certi che lo schema applicato continuerà a leggere i dati correttamente.

Aggiornamento dei metadati della tabella

Dopo una scansione, il AWS Glue crawler assegna automaticamente determinati metadati alla tabella per renderla compatibile con altre tecnologie esterne come Apache Hive, Presto e Spark. Occasionalmente, il crawler potrebbe assegnare le proprietà dei metadati in modo errato. Correggi manualmente le proprietà AWS Glue prima di interrogare la tabella usando Athena. Per ulteriori informazioni, consulta la sezione relativa alla visualizzazione e modifica dei dettagli tabella nella Guida per sviluppatori AWS Glue .

AWS Glue può assegnare erroneamente i metadati quando un file CSV contiene virgolette su ogni campo di dati, sbagliando la proprietà. `serializationLib` Per ulteriori informazioni, consulta Dati CSV tra virgolette.

Utilizzo dei file CSV

Nei file CSV, talvolta i valori dei dati destinati a ciascuna colonna sono scritti tra virgolette; inoltre nei file CSV potrebbero essere presenti dei valori di intestazione che non fanno parte dei dati da analizzare. Quando usi AWS Glue per creare uno schema da questi file, segui le istruzioni riportate in questa sezione.

Dati CSV tra virgolette

Potresti avere un file CSV che dispone di campi dati racchiusi tra virgolette doppie come l'esempio seguente:

```
"John","Doe","123-555-1231","John said \"hello\""
"Jane","Doe","123-555-9876","Jane said \"hello\""
```

Per eseguire una query in Athena su una tabella creata da un file CSV contenente valori tra virgolette, è necessario modificare le proprietà della tabella in modo da AWS Glue utilizzare OpenCSV. SerDe Per ulteriori informazioni su OpenCSV SerDe, vedere. OpenCSV per l'elaborazione di file SerDe CSV

Per modificare le proprietà delle tabelle nella console AWS Glue

1. Nel riquadro di navigazione della AWS Glue console, scegli Tabelle.
2. Scegli il collegamento per la tabella che desideri modificare, quindi scegli Actions (Operazioni), Edit table details (Modifica tabella).
3. Nella pagina Edit table (Modifica tabella), apporta le modifiche seguenti:
   - Per Serialization lib (Libreria serializzazione), inserisci `org.apache.hadoop.hive.serde2.OpenCSVSerde`.
   - Per Parametri Serde, inserisci i seguenti valori per le chiavi `escapeChar`, `quoteChar` e `separatorChar`:
     - Per `escapeChar`, inserire una barra rovesciata (****).
     - Per `quoteChar`, inserire una virgoletta doppia (**"**).
     - Per `separatorChar`, inserire una virgola (**,**).
4. Selezionare Salva.

Per ulteriori informazioni, consulta la sezione relativa alla visualizzazione e modifica dei dettagli tabella nella Guida per sviluppatori AWS Glue .

Aggiornamento delle proprietà AWS Glue delle tabelle a livello di codice

È possibile utilizzare l'operazione AWS Glue UpdateTableAPI o il comando CLI update-table per modificare il `SerDeInfo` blocco nella definizione della tabella, come nel seguente esempio JSON.

```
"SerDeInfo": {
```

```
   "name": "",
   "serializationLib": "org.apache.hadoop.hive.serde2.OpenCSVSerde",
   "parameters": {
      "separatorChar": ","
      "quoteChar": "\""
      "escapeChar": "\\"
      }
},
```

File CSV con intestazioni

Quando definisci una tabella in Athena con un'istruzione `CREATE TABLE`, puoi utilizzare la proprietà tabella `skip.header.line.count` per ignorare gli header nei dati CSV, come nell'esempio seguente.

```
...
STORED AS TEXTFILE
LOCATION 's3://DOC-EXAMPLE-BUCKET/csvdata_folder/';
TBLPROPERTIES ("skip.header.line.count"="1")
```

In alternativa, puoi rimuovere gli header CSV in anticipo in modo che le informazioni degli header non siano incluse nei risultati delle query di Athena di Athena. Un modo per raggiungere questo obiettivo è utilizzare i AWS Glue job, che eseguono operazioni di estrazione, trasformazione e caricamento (ETL). È possibile scrivere script AWS Glue utilizzando un linguaggio che è un'estensione del dialetto PySpark Python. Per ulteriori informazioni, consulta Authoring Jobs in AWS Glue nella AWS Glue Developer Guide.

L'esempio seguente mostra una funzione in AWS Glue uno script che scrive un frame dinamico utilizzando `from_options` e imposta l'opzione `writeHeader` format su false, che rimuove le informazioni di intestazione:

```
glueContext.write_dynamic_frame.from_options(frame = applymapping1, connection_type =
 "s3", connection_options = {"path": "s3://DOC-EXAMPLE-BUCKET/MYTABLEDATA/"}, format =
 "csv", format_options = {"writeHeader": False}, transformation_ctx = "datasink2")
```

AWS Glue indicizzazione e filtraggio delle partizioni

Quando Athena esegue una query su tabelle partizionate, recupera e filtra le partizioni della tabella disponibili nel sottoinsieme pertinente alla query. Quando vengono aggiunti nuovi dati e partizioni, è necessario più tempo per elaborare le partizioni e il runtime delle query può aumentare. Se si dispone di una tabella con un numero elevato di partizioni che cresce nel tempo, considerare l'uso

di indicizzazione e filtro delle partizioni AWS Glue . L'indicizzazione delle partizioni consente ad Athena di ottimizzare l'elaborazione delle partizioni e migliorare le prestazioni delle query su tabelle altamente partizionate. L'impostazione del filtro delle partizioni nelle proprietà di una tabella è un processo a due fasi:

1. Creazione di un indice di partizione in AWS Glue.
2. Abilitazione del filtro delle partizioni per la tabella.

Creazione di un indice di partizione

Per istruzioni su come creare un indice di partizione in AWS Glue, consulta Working with partition indexes nella Developer Guide. AWS Glue Per le limitazioni relative agli indici di partizione in AWS Glue, consulta la sezione Informazioni sugli indici di partizione in quella pagina.

Abilitazione del filtro delle partizioni

Per abilitare il filtro delle partizioni per la tabella, è necessario impostare una nuova proprietà della tabella in AWS Glue. Per istruzioni su come impostare le proprietà della tabella in AWS Glue, consulta la pagina Configurazione della proiezione delle partizioni. Quando modificate i dettagli della tabella in AWS Glue, aggiungete la seguente coppia chiave-valore alla sezione Proprietà della tabella:

- Per Key (Chiave), aggiungi `partition_filtering.enabled`
- Per Value (Valore), aggiungi `true`

È possibile disabilitare la proiezione delle partizioni su questa tabella in qualsiasi momento impostando `partition_filtering.enabled` su `false`.

Dopo aver completato le fasi precedenti, sarà possibile tornare alla console Athena per eseguire la query sui dati.

Per ulteriori informazioni sull'utilizzo dell'indicizzazione e del filtraggio delle partizioni, consulta Migliorare le prestazioni delle query di Amazon Athena utilizzando gli indici delle AWS Glue Data Catalog partizioni nel Big Data Blog.AWS

Utilizzo dei dati geospaziali

AWS Glue non supporta nativamente Woonown Text (WKT), Wooden Binary (WKB) o altri tipi di dati PostGIS. Il AWS Glue classificatore analizza i dati geospaziali e li classifica utilizzando i tipi di dati

supportati per il formato, ad esempio per CSV. `varchar` Come per altre AWS Glue tabelle, potrebbe essere necessario aggiornare le proprietà delle tabelle create da dati geospaziali per consentire ad Athena di analizzare questi tipi di dati così come sono. Per ulteriori informazioni, consulta Utilizzo dei AWS Glue crawler e Utilizzo dei file CSV. Athena potrebbe non essere in grado di analizzare alcuni tipi di dati geospaziali nelle tabelle così come sono. AWS Glue Per ulteriori informazioni sull'utilizzo dei dati geospaziali in Athena, consulta Esecuzione di query su dati geospaziali.

Utilizzo AWS Glue di job per ETL con Athena

AWS Glue i lavori eseguono operazioni ETL. Un AWS Glue job esegue uno script che estrae i dati dalle fonti, li trasforma e li carica in destinazioni. Per ulteriori informazioni, consulta Authoring Jobs in AWS Glue nella AWS Glue Developer Guide.

Creazione di tabelle utilizzando Athena per lavori ETL AWS Glue

Alle tabelle da te create in Athena devi aggiungere una proprietà di tabella, denominata `classification`, che identifica il formato dei dati. Ciò consente di AWS Glue utilizzare le tabelle per i lavori ETL. I valori di classificazione possono essere `avro`, `csv`, `json`, `orc`, `parquet` o `xml`. Di seguito è riportata un'istruzione `CREATE TABLE` di esempio in Athena:

```
CREATE EXTERNAL TABLE sampleTable (
  column1 INT,
  column2 INT
  ) STORED AS PARQUET
  TBLPROPERTIES (
  'classification'='parquet')
```

Se la proprietà della tabella non è stata aggiunta al momento della creazione della tabella, è possibile aggiungerla utilizzando la AWS Glue console.

Per aggiungere la proprietà della tabella di classificazione utilizzando la AWS Glue console

1. Accedere AWS Management Console e aprire la AWS Glue console all'indirizzo https://console.aws.amazon.com/glue/.
2. Nel pannello di navigazione della console, seleziona Tables (Tabelle).
3. Scegli il collegamento per la tabella che desideri modificare, quindi scegli Actions (Operazioni), Edit table details (Modifica tabella).
4. Scorri verso il basso fino alla sezione Table properties (Proprietà della tabella).
5. Scegli Aggiungi.

6. In Chiave, inserire **`classification`**.
7. Per Value (Valore), inserisci un tipo di dati (ad esempio, **`json`**).
8. Selezionare Salva.

   Nella sezione Table details (Dettagli della tabella), il tipo di dati che hai inserito appare nel campo Classification (Classificazione) della tabella.

Per ulteriori informazioni, consulta l'argomento relativo all'utilizzo delle tabelle nella Guida per sviluppatori AWS Glue .

Utilizzo di processi ETL per ottimizzare le prestazioni delle query

AWS Glue jobs possono aiutarti a trasformare i dati in un formato che ottimizza le prestazioni delle query in Athena. I formati dei dati influiscono moltissimo sulle prestazioni delle query e sui costi delle query in Athena.

Si consiglia di utilizzare i formati di dati Parquet e ORC. AWS Glue supporta la scrittura in entrambi questi formati di dati, il che può semplificare e velocizzare la trasformazione dei dati in un formato ottimale per Athena. Per ulteriori informazioni su questi formati e altri modi per migliorare le prestazioni, consulta i 10 migliori suggerimenti per l'ottimizzazione delle prestazioni per Amazon Athena.

Conversione di tipi di dati SMALLINT e TINYINT su INT durante la conversione in ORC

Per ridurre la probabilità che Athena non sia in grado di leggere i tipi di dati `TINYINT` e `SMALLINT` i tipi di dati prodotti da AWS Glue un processo ETL, `SMALLINT` converti `TINYINT` e `INT` verso quando usi la procedura guidata o scrivi uno script per un processo ETL.

Automazione dei lavori per ETL AWS Glue

È possibile configurare i processi AWS Glue ETL in modo che vengano eseguiti automaticamente in base ai trigger. Questa funzionalità è ideale quando i dati dall'esterno AWS vengono inviati a un bucket Amazon S3 in un formato non ottimale per l'esecuzione di query in Athena. Per ulteriori informazioni, consulta Attivazione dei processi AWS Glue nella Guida per sviluppatori di AWS Glue .

Utilizzo di AWS CLI per ricreare un AWS Glue database e le relative tabelle

Non è possibile rinominare direttamente un AWS Glue database, ma è possibile copiarne la definizione, modificarla e utilizzarla per ricreare il database con un nome diverso. Analogamente, è

possibile copiare le definizioni delle tabelle nel vecchio database, modificare le definizioni e utilizzare le definizioni modificate per ricreare le tabelle nel nuovo database.

> **Note**
>
> Il metodo presentato non copia il partizionamento delle tabelle.

La procedura seguente per Windows presuppone che l'utente AWS CLI sia configurato per l'output JSON. Per modificare il formato di output predefinito in AWS CLI, esegui. `aws configure`

Per copiare un AWS Glue database utilizzando il AWS CLI

1. Al prompt dei comandi, esegui il AWS CLI comando seguente per recuperare la definizione del AWS Glue database che desideri copiare.

   ```
   aws glue get-database --name database_name
   ```

   Per ulteriori informazioni sul comando `get-database`, consulta get-database.
2. Salva l'output JSON in un file con il nome del nuovo database (ad esempio, *new_database_name*`.json`) sul desktop.
3. Apri il file *new_database_name*`.json` in un editor di testo.
4. Nel file JSON, effettuate le seguenti operazioni:

   a. Rimuovete l'`{ "Database":`ingresso esterno e il corrispondente tutore di chiusura `}` alla fine del file.

   b. Cambia la `Name` voce con il nuovo nome del database.

   c. Rimuovi il campo `CatalogId`.
5. Salvare il file.
6. Al prompt dei comandi, eseguire il AWS CLI comando seguente per utilizzare il file di definizione del database modificato per creare il database con il nuovo nome.

   ```
   aws glue create-database --database-input "file://~/Desktop\new_database_name.json"
   ```

   Per ulteriori informazioni sul comando `create-database`, consulta create-database. Per informazioni sul caricamento AWS CLI dei parametri da un file, vedere Caricamento AWS CLI dei parametri da un file nella Guida per l'AWS Command Line Interface utente.

7. Per verificare che il nuovo database sia stato creato in AWS Glue, esegui il seguente comando:

```
aws glue get-database --name new_database_name
```

Ora siete pronti per ottenere la definizione di una tabella da copiare nel nuovo database, modificare la definizione e utilizzare la definizione modificata per ricreare la tabella nel nuovo database. Questa procedura non modifica il nome della tabella.

Per copiare una AWS Glue tabella utilizzando il AWS CLI

1. Al prompt dei comandi, esegui il AWS CLI comando seguente.

```
aws glue get-table --database-name database_name --name table_name
```

   Per ulteriori informazioni sul comando `get-table`, consulta get-table.

2. Salva l'output JSON in un file con il nome della tabella (ad esempio, *`table_name`*.json) sul desktop di Windows.
3. Apri il file in un editor di testo.
4. Nel file JSON, rimuovi la voce `{"Table":` esterna e la parentesi di chiusura corrispondente `}` alla fine del file.
5. Nel file JSON, rimuovi le seguenti voci e i relativi valori:
   - `DatabaseName`: questa voce non è obbligatoria perché il comando CLI `create-table` utilizza il parametro `--database-name`.
   - `CreateTime`
   - `UpdateTime`
   - `CreatedBy`
   - `IsRegisteredWithLakeFormation`
   - `CatalogId`
   - `VersionId`
6. Salvate il file di definizione della tabella.
7. Al prompt dei comandi, esegui il AWS CLI comando seguente per ricreare la tabella nel nuovo database:

```
aws glue create-table --database-name new_database_name --table-input "file://~/
Desktop\table_name.json"
```

Per ulteriori informazioni sul comando `create-table`, consulta create-table.

La tabella ora appare nel nuovo database di AWS Glue e può essere interrogata da Athena.

8. Ripeti i passaggi per copiare ogni tabella aggiuntiva nel nuovo database in AWS Glue.

Utilizzo di un connettore dati Athena per il metastore Hive esterno

Puoi utilizzare il connettore dati Amazon Athena per un metastore Hive esterno per interrogare set di dati in Amazon S3 che utilizzano un metastore Apache Hive. Non è necessaria alcuna migrazione dei metadati AWS Glue Data Catalog verso. Nella console di gestione di Athena, configura una funzione Lambda per comunicare con il metastore Hive nel VPC privato e quindi collegala al metastore. La connessione da Lambda al metastore Hive è protetta da un canale Amazon VPC privato e non utilizza la rete Internet pubblica. Puoi fornire il tuo codice di funzione Lambda, oppure puoi utilizzare l'implementazione predefinita del connettore dati Athena per un metastore Hive esterno.

Argomenti

Panoramica delle caratteristiche

Con il connettore dati Athena per un metastore Hive esterno, puoi svolgere le seguenti attività:

- Utilizzare la console Athena per registrare cataloghi personalizzati ed eseguire query che li utilizzano.
- Definire le funzioni Lambda per diversi metastore Hive esterni e unirle nelle query Athena.
- Usa AWS Glue Data Catalog i tuoi metastore Hive esterni nella stessa query Athena.
- Specificare un catalogo nel contesto di esecuzione della query come catalogo predefinito corrente. In questo modo viene rimosso il requisito di anteporre i nomi dei cataloghi ai nomi dei database nelle query. Invece di usare la sintassi *`catalog`*`.`*`database`*`.`*`table`*, è possibile usare *`database`*`.`*`table`*.
- Utilizzare una varietà di strumenti per eseguire query che fanno riferimento a metastore Hive esterni. Puoi utilizzare la console Athena, l' AWS SDK AWS CLI, le API Athena e i driver Athena JDBC e ODBC aggiornati. I driver aggiornati supportano i cataloghi personalizzati.

Supporto API

Il connettore Athena Data per il metastore Hive esterno include il supporto per le operazioni API di registrazione del catalogo e le operazioni API relative ai metadati.

- Registrazione catalogo: registrare cataloghi personalizzati per metastore Hive esterni e origini dati federate.
- Metadati: utilizza le API dei metadati per fornire informazioni su database e tabelle per AWS Glue qualsiasi catalogo registrato con Athena.
- Client SDK Athena JAVA - Utilizza le API di registrazione del catalogo, le API dei metadati e il supporto per i cataloghi nell'operazione `StartQueryExecution` nel client SDK Athena Java aggiornato.

Implementazione di riferimento

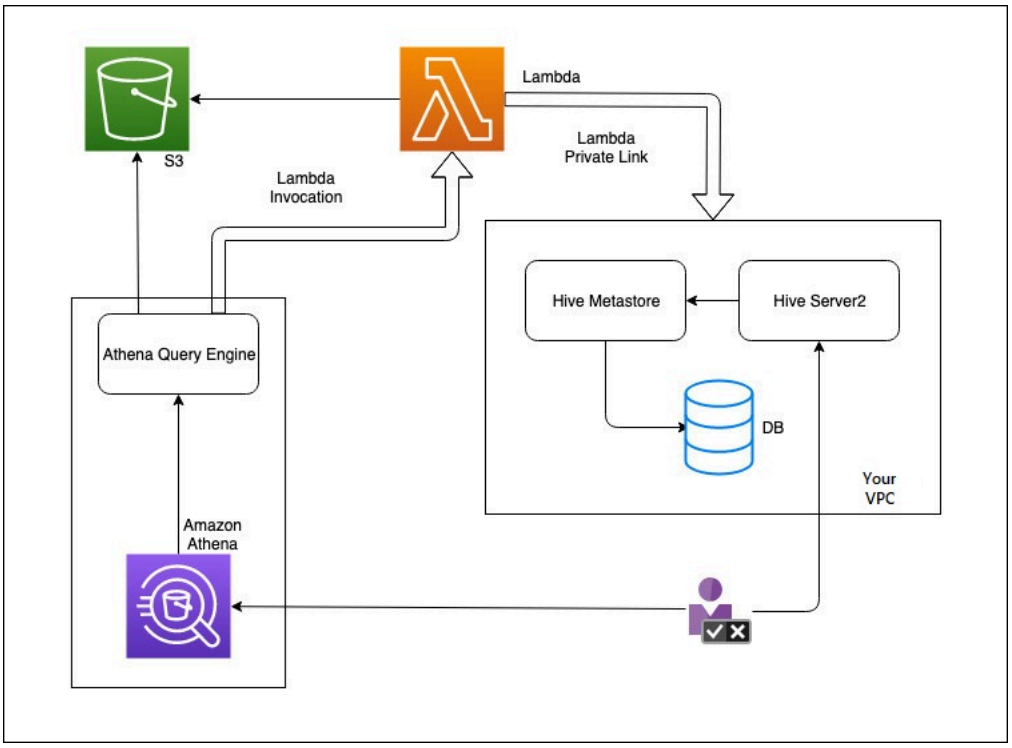
Athena fornisce un'implementazione di riferimento per la funzione Lambda che si connette a metastore Hive esterni. L'implementazione di riferimento è fornita GitHub come progetto open source presso il metastore Athena Hive.

L'implementazione di riferimento è disponibile come le seguenti due AWS SAM applicazioni in (SAR). AWS Serverless Application Repository È possibile utilizzare una di queste applicazioni in SAR per creare le proprie funzioni Lambda.

- **`AthenaHiveMetastoreFunction`**— File `.jar` della funzione Uber Lambda. Un JAR "uber" (noto anche come JAR grasso o JAR con dipendenze) è un `.jar` che contiene sia un programma Java che le sue dipendenze in un unico file.
- **`AthenaHiveMetastoreFunctionWithLayer`** – Livello Lambda e file `.jar` della funzione Lambda.

Flusso di lavoro

Il diagramma seguente mostra come Athena interagisce con il metastore Hive esterno.



In questo flusso di lavoro, il metastore Hive connesso al database si trova all'interno del VPC. È possibile utilizzare Hive Server2 per gestire il metastore Hive utilizzando l'interfaccia CLI di Hive.

Il flusso di lavoro per l'utilizzo di metastore Hive esterni da Athena include i passaggi seguenti.

1. Si crea una funzione Lambda che connette Athena al metastore Hive che si trova all'interno del VPC.
2. Si registra un nome di catalogo univoco per il metastore Hive e un nome di funzione corrispondente nell'account.
3. Quando si esegue una query DML o DDL Athena che utilizza il nome del catalogo, il motore di query Athena chiama il nome della funzione Lambda associato al nome del catalogo.

4. Utilizzando AWS PrivateLink, la funzione Lambda comunica con il metastore Hive esterno nel tuo VPC e riceve risposte alle richieste di metadati. Athena usa i metadati dal metastore Hive esterno proprio come usa i metadati dalla AWS Glue Data Catalog predefinita.

Considerazioni e limitazioni

Quando si utilizza Athena Data Connector per il metastore Hive esterno, considerare i seguenti punti:

- Puoi utilizzare CTAS per creare una tabella su un metastore Hive esterno.
- Puoi utilizzare INSERT INTO per inserire dati in un metastore Hive esterno.
- Il supporto DDL per il metastore Hive esterno è limitato alle seguenti istruzioni.
  - ALTER DATABASE SET DBPROPERTIES
  - ALTER TABLE ADD COLUMNS
  - ALTER TABLE ADD PARTITION
  - ALTER TABLE DROP PARTITION
  - ALTER TABLE RENAME PARTITION
  - ALTER TABLE REPLACE COLUMNS
  - ALTER TABLE SET LOCATION
  - ALTER TABLE SET TBLPROPERTIES
  - CREATE DATABASE
  - CREATE TABLE
  - CREATE TABLE AS
  - DESCRIBE TABLE
  - DROP DATABASE
  - DROP TABLE
  - SHOW COLUMNS
  - SHOW CREATE TABLE
  - SHOW PARTITIONS
  - SHOW SCHEMAS
  - SHOW TABLES
  - SHOW TBLPROPERTIES
- Il numero massimo di cataloghi registrati che è possibile avere è 1.000.

- L'autenticazione Kerberos per metastore Hive non è supportata.
- Per utilizzare il driver JDBC con un metastore Hive esterno o query federate, includi `MetadataRetrievalMethod=ProxyAPI` nella stringa di connessione JDBC. Per informazioni sul driver JDBC, consulta Connessione ad Amazon Athena con JDBC.
- Le colonne nascoste di Hive `$path`, `$bucket`, `$file_size`, `$file_modified_time`, `$partition` e `$row_id` non possono essere utilizzate per filtrare con il controllo granulare degli accessi.
- Le tabelle di sistema Hive nascoste, come *`example_table`*`$partitions` o *`example_table`*`$properties`, non sono supportate dal controllo granulare degli accessi.

Autorizzazioni

I connettori dati precompilati e personalizzati potrebbero richiedere l'accesso alle seguenti risorse per funzionare correttamente. Controlla le informazioni relative al connettore utilizzato per assicurarti di aver configurato correttamente il VPC. Per informazioni sulle autorizzazioni IAM obbligatorie per eseguire query e creare un connettore origine dati in Athena, consulta Consenti l'accesso a un connettore dati Athena per il metastore Hive esterno e Consentire l'accesso alla funzione Lambda a metastore Hive esterni.

- Amazon S3 — Oltre a scrivere i risultati delle query nella posizione dei risultati della query Athena in Amazon S3, i connettori di dati scrivono anche in un bucket di spill in Amazon S3. Sono richieste connettività e autorizzazioni a questa posizione Amazon S3. Per ulteriori informazioni, consulta Posizione spill in Amazon S3 più avanti in questo argomento.
- Athena – L'accesso è necessario per controllare lo stato della query e impedire l'overscan.
- AWS Glue— L'accesso è necessario se il connettore utilizza metadati supplementari o primari. AWS Glue
- AWS Key Management Service
- Policy — Il metastore Hive, Athena Query Federation e le UDF richiedono policy in aggiunta a AWS politica gestita: AmazonAthenaFullAccess. Per ulteriori informazioni, consulta Gestione dell'identità e dell'accesso in Athena.

Posizione spill in Amazon S3

A causa del limite relativo alle dimensioni delle risposte delle funzioni Lambda, le risposte superiori alla soglia si riversano in una posizione Amazon S3 specificata al momento della creazione della funzione Lambda. Athena legge direttamente queste risposte da Amazon S3.

> **Note**
>
> Athena non rimuove i file di risposta su Amazon S3. Ti consigliamo di impostare una policy di conservazione per eliminare automaticamente i file di risposta.

Connessione di Athena a un metastore Apache Hive

Per connettere Athena a un metastore Apache Hive, devi creare e configurare una funzione Lambda. Per un'implementazione di base, puoi eseguire tutte le fasi richieste a partire dalla console di gestione Athena.

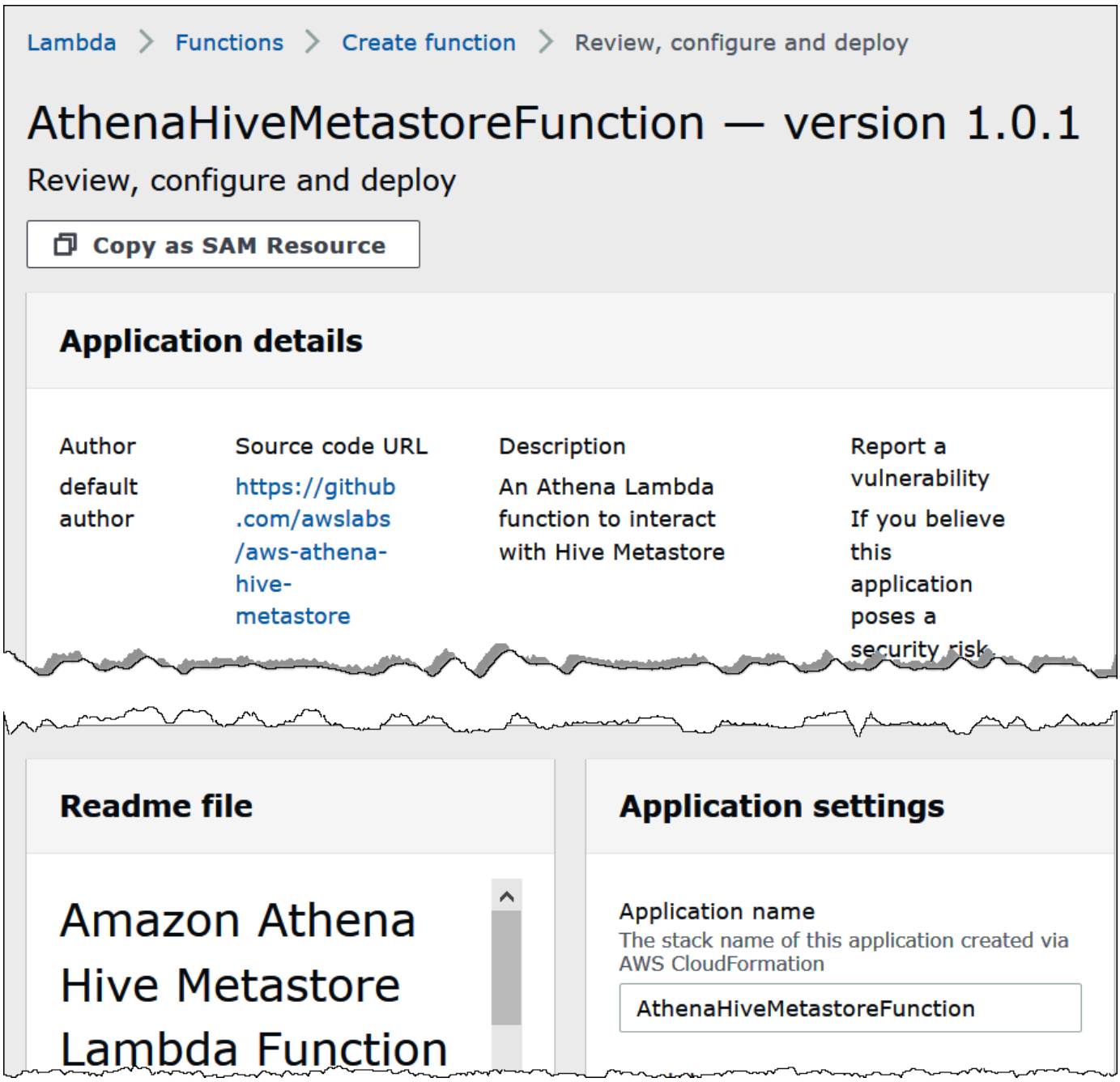
> **Note**
>
> La procedura seguente richiede l'autorizzazione per creare un ruolo IAM personalizzato per la funzione Lambda. Se non disponi dell'autorizzazione per creare un ruolo personalizzato, puoi utilizzare l'implementazione di riferimento Athena per creare una funzione Lambda separatamente, quindi utilizzare la AWS Lambda console per scegliere un ruolo IAM esistente per la funzione. Per ulteriori informazioni, consulta Collegamento di Athena a un metastore Hive utilizzando un ruolo di esecuzione IAM esistente.

Per connettere Athena a un metastore Hive

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Scegli Data sources (Origini dati).
4. Nell'angolo in alto a destra della console, scegli Create data source (Crea origine dati).
5. Nella pagina Choose a data source (Scegli un'origine dati), per Data source (Origini dati), scegli S3 - Apache Hive metastore (Metastore Apache Hive - S3).
6. Seleziona Successivo.
7. Nella sezione Data source details (Dettagli origine dati), per Data source name (Nome origine dati), inserisci il nome che desideri utilizzare nelle istruzioni SQL quando esegui una query sull'origine dati da Athena. Il nome può contenere fino a 127 caratteri e deve essere univoco all'interno dell'account. Non può essere modificato dopo la creazione. I caratteri validi sono a-z, A-z, 0-9, _ (trattino basso), @ (chiocciola) e - (trattino). I nomi `awsdatacatalog`, `hive`, `jmx` e `system` sono riservati ad Athena e non possono essere utilizzati per i nomi delle origini dati.
8. Per la funzione Lambda, scegli Crea funzione Lambda, quindi scegli Crea una nuova funzione Lambda in AWS Lambda

   La AthenaHiveMetastoreFunctionpagina si apre nella console. AWS Lambda La pagina include informazioni dettagliate sul connettore.

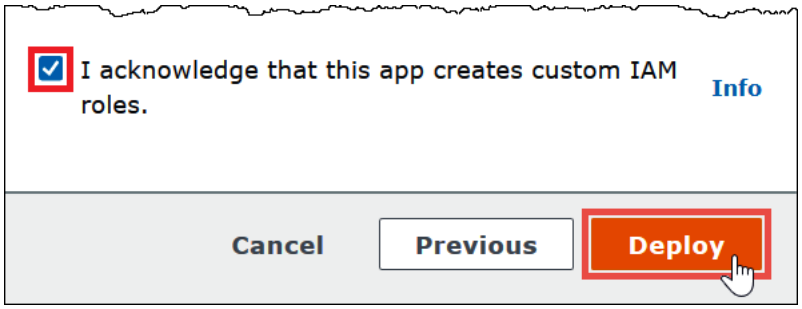


9. Sotto Impostazioni applicazioni inserisci i parametri per la funzione Lambda.

   - LambdaFuncName— Fornire un nome per la funzione. Ad esempio, myHiveMetastore.
   - SpillLocation— Specificare una posizione Amazon S3 in questo account per conservare i metadati di spillover se la dimensione della risposta della funzione Lambda supera i 4 MB.
   - HMSUris: inserisci l'URI dell'host del metastore Hive che utilizza il protocollo Thrift alla porta 9083. Utilizzo della sintassi `thrift://<host_name>:9083`.

- LambdaMemory— Specificare un valore compreso tra 128 MB e 3008 MB. Alla funzione Lambda vengono allocati cicli di CPU proporzionali alla quantità di memoria configurata. Il valore di default è 1024.
- LambdaTimeout— Specificare il tempo di esecuzione della chiamata Lambda massimo consentito in secondi da 1 a 900 (900 secondi corrispondono a 15 minuti). Il valore predefinito è 300 secondi (5 minuti).
- VPC SecurityGroupIds: inserisci un elenco separato da virgole di ID dei gruppi di sicurezza VPC per il metastore Hive.
- VPC SubnetIds: inserisci un elenco separato da virgole di ID di sottorete VPC per il metastore Hive.

10. Seleziona I acknowledge that this app creates custom IAM roles (Sono consapevole che questa app crea ruoli IAM personalizzati), quindi scegli Deploy (Implementa).



Al termine della distribuzione, la funzione viene visualizzata nell'elenco delle applicazioni Lambda. Ora che la funzione metastore Hive è stata distribuita sul tuo account, puoi configurare Athena per usarla.

11. Torna alla pagina Enter data source details (Inserisci i dettagli dell'origine dati) nella console Athena.
12. Nella sezione Lambda function (Funzione Lambda), scegli l'icona di aggiornamento accanto alla casella di ricerca della funzione Lambda. L'aggiornamento dell'elenco delle funzioni disponibili fa sì che la funzione appena creata venga visualizzata nell'elenco.
13. Scegli il nome della funzione appena creata nella console Lambda. Viene visualizzato l'ARN della funzione Lambda.
14. (Facoltativo) Per Tags (Tag), aggiungi coppie chiave-valore da associare a questa origine dati. Per ulteriori informazioni sui tag, consulta Assegnazione di tag alle risorse Athena.
15. Seleziona Successivo.
16. Nella pagina Review and create (Rivedi e crea), esamina i dettagli dell'origine dati, quindi scegli Create data source (Crea origine dati).

17. La sezione Data source details (Dettagli sull'origine dati) della pagina dell'origine dati mostra le informazioni relative al nuovo connettore.

    È ora possibile utilizzare il Data source name (Nome origine dati) specificato per fare riferimento al metastore Hive nelle query SQL in Athena. Nelle query SQL utilizzare la sintassi di esempio seguente, sostituendo `hms-catalog-1` con il nome del catalogo specificato in precedenza.

    ```
    SELECT * FROM hms-catalog-1.CustomerData.customers
    ```

18. Per informazioni sulla visualizzazione, la modifica o l'eliminazione delle origini dati create, consulta Gestione delle origini dati.

Utilizzo di AWS Serverless Application Repository per distribuire un connettore di origine dati Hive

Per distribuire un connettore origine dati Athena per Hive, puoi utilizzare AWS Serverless Application Repository invece di iniziare con la console Athena. Usa il AWS Serverless Application Repository per trovare il connettore che desideri utilizzare, fornisci i parametri richiesti dal connettore e quindi distribuisci il connettore al tuo account. Quindi, dopo aver distribuito il connettore, utilizza la console Athena per rendere disponibile l'origine dati ad Athena.

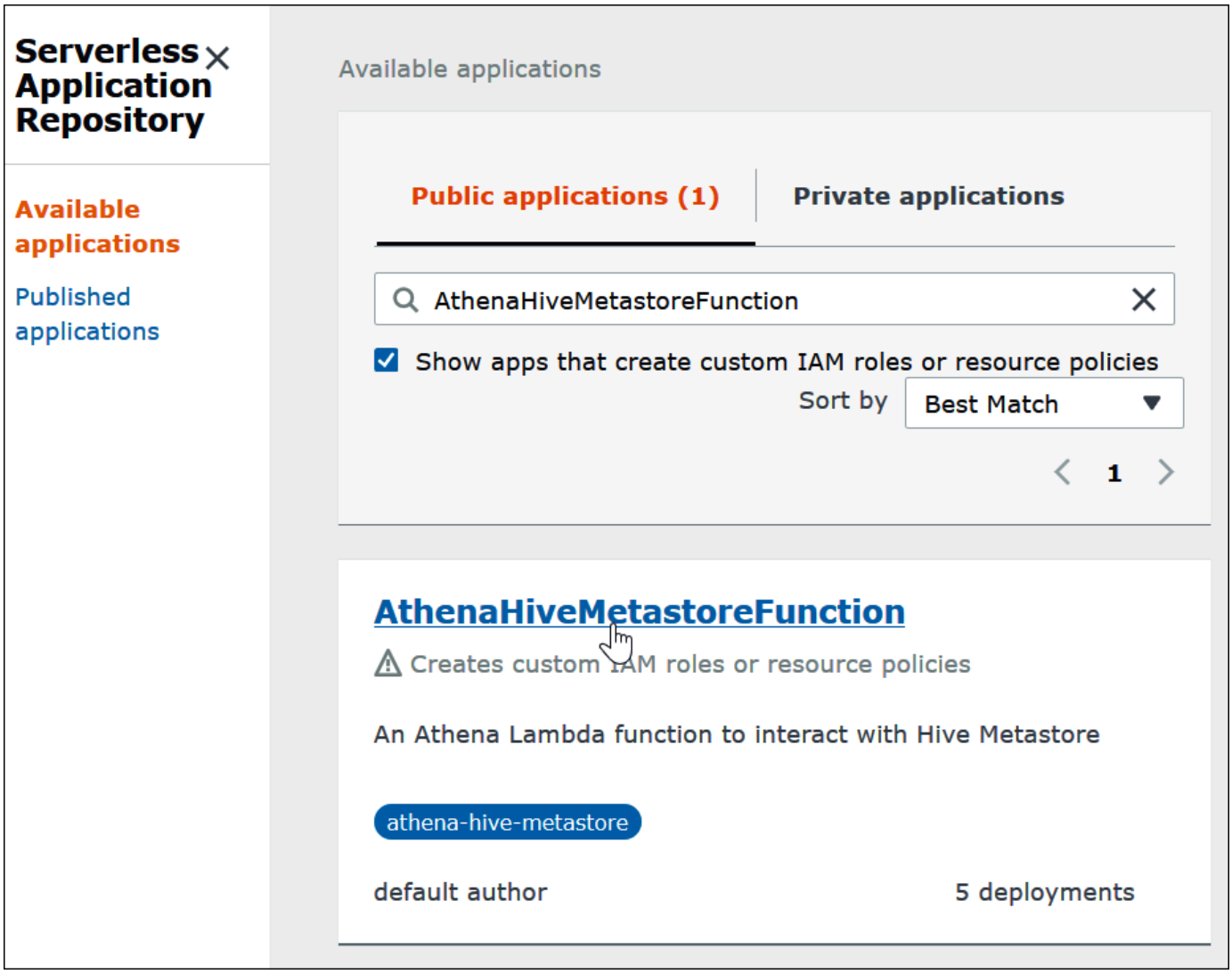
Da utilizzare AWS Serverless Application Repository per distribuire un connettore di origine dati per Hive sul tuo account

1. Accedi AWS Management Console e apri il Serverless App Repository.
2. Nel pannello di navigazione, scegli Available applications (Applicazioni disponibili).
3. Seleziona l'opzione Visualizzare le app che creano ruoli IAM personalizzati o policy delle risorse.
4. Nella casella di ricerca immetti **`Hive`**. I connettori visualizzati includono i due seguenti:

    - AthenaHiveMetastoreFunction— File `.jar` della funzione Uber Lambda.
    - AthenaHiveMetastoreFunctionWithLayer— Layer Lambda e file di funzioni Lambda sottile. `.jar`

    Le due applicazioni hanno la stessa funzionalità e differiscono solo nella loro implementazione. È possibile utilizzarle entrambe per creare una funzione Lambda che connette Athena al metastore Hive.

5. Scegli il nome del connettore da utilizzare. In questo tutorial si utilizza AthenaHiveMetastoreFunction.



6. Sotto Impostazioni applicazioni inserisci i parametri per la funzione Lambda.

   - LambdaFuncName— Fornite un nome per la funzione. Ad esempio, myHiveMetastore.
   - SpillLocation— Specificare una posizione Amazon S3 in questo account per conservare i metadati di derivazione se la dimensione della risposta della funzione Lambda supera i 4 MB.
   - HMSUris: inserisci l'URI dell'host del metastore Hive che utilizza il protocollo Thrift alla porta 9083. Utilizzo della sintassi `thrift://<host_name>:9083`.
   - LambdaMemory— Specificare un valore compreso tra 128 MB e 3008 MB. Alla funzione Lambda vengono allocati cicli di CPU proporzionali alla quantità di memoria configurata. Il valore di default è 1024.

- LambdaTimeout— Specificare il tempo di esecuzione della chiamata Lambda massimo consentito in secondi da 1 a 900 (900 secondi corrispondono a 15 minuti). Il valore predefinito è 300 secondi (5 minuti).
- VPC SecurityGroupIds: inserisci un elenco separato da virgole di ID dei gruppi di sicurezza VPC per il metastore Hive.
- VPC SubnetIds: inserisci un elenco separato da virgole di ID di sottorete VPC per il metastore Hive.

7. Nella parte inferiore destra della pagina Dettagli applicazione seleziona Sono consapevole che questa app crea ruoli IAM personalizzati, quindi scegli Distribuisci.

A questo punto, puoi configurare Athena per utilizzare la funzione Lambda per connetterti al metastore Hive. Per le fasi, consulta Configurazione di Athena per l'utilizzo di un connettore Hive Metastore distribuito.

Collegamento di Athena a un metastore Hive utilizzando un ruolo di esecuzione IAM esistente

Per connettere il metastore Hive esterno ad Athena con una funzione Lambda che utilizza un ruolo IAM esistente, è possibile utilizzare l'implementazione di riferimento di Athena del connettore Athena per il metastore Hive esterno.

Le tre fasi principali sono le seguenti:

1. Clona e Costruisci: clona l'implementazione di riferimento Athena e crea il file JAR che contiene il codice funzione Lambda.
2. AWS Lambda console: nella AWS Lambda console, crea una funzione Lambda, assegnale un ruolo di esecuzione IAM esistente e carica il codice della funzione che hai generato.
3. Console Amazon Athena: nella console Amazon Athena, crea un nome di origine dati che puoi usare per fare riferimento al tuo metastore Hive esterno nelle query Athena.

Se disponi già delle autorizzazioni per creare un ruolo IAM personalizzato, puoi utilizzare un flusso di lavoro più semplice che utilizza la console Athena e AWS Serverless Application Repository per creare e configurare una funzione Lambda. Per ulteriori informazioni, consulta Connessione di Athena a un metastore Apache Hive.

Prerequisiti

- Git deve essere installato sul sistema.
- È necessario avere Apache Maven installato.
- Hai un ruolo di esecuzione IAM che puoi assegnare alla funzione Lambda. Per ulteriori informazioni, consulta Consentire l'accesso alla funzione Lambda a metastore Hive esterni.

Clona e costruisci la funzione Lambda

Il codice della funzione per l'implementazione di riferimento Athena è un progetto Maven situato su awslabs/. GitHub aws-athena-hive-metastore Per informazioni dettagliate sul progetto, consulta il file README corrispondente su GitHub o l'argomento di questa documentazione. Implementazione di riferimento

Per clonare e costruire il codice della funzione Lambda

1. Inserire il seguente comando per clonare l'implementazione di riferimento Athena:

   ```
   git clone https://github.com/awslabs/aws-athena-hive-metastore
   ```

2. Eseguire il comando seguente per creare il file `.jar` per la funzione Lambda:

   ```
   mvn clean install
   ```

   Dopo che il progetto è stato creato correttamente, il seguente file `.jar` viene creato nella cartella di destinazione del tuo progetto:

   `hms-lambda-func-1.0-SNAPSHOT-withdep.jar`

   Nella sezione successiva, usi la AWS Lambda console per caricare questo file sul tuo account Amazon Web Services.

Creare e configurare la funzione Lambda nella console AWS Lambda

In questa sezione, si utilizza la AWS Lambda console per creare una funzione che utilizza un ruolo di esecuzione IAM esistente. Dopo aver configurato un VPC per la funzione, carica il codice della funzione e configura le variabili di ambiente per la funzione.

Creazione della funzione Lambda

In questo passaggio, crei una funzione nella AWS Lambda console che utilizza un ruolo IAM esistente.

Per creare una funzione Lambda che utilizza un ruolo IAM esistente

1. Accedi AWS Management Console e apri la AWS Lambda console all'indirizzo https://console.aws.amazon.com/lambda/.
2. Nel riquadro di navigazione, seleziona Funzioni.
3. Selezionare Create function (Crea funzione).
4. Scegli Author from scratch (Crea da zero).
5. In Nome funzione, inserisci il nome della funzione Lambda (ad esempio, **`EHMSBasedLambda`**).
6. Per Runtime, scegliere Java 8.
7. In Autorizzazioni espandere Modifica ruolo di esecuzione predefinito.
8. In Execution role (Ruolo di esecuzione), scegliere Use an existing role (Utilizza un ruolo esistente).
9. Per Ruolo esistente, scegli il ruolo di esecuzione IAM che la tua funzione Lambda utilizzerà per Athena (questo esempio utilizza un ruolo chiamato `AthenaLambdaExecutionRole`).
10. Espandere Advanced settings (Impostazioni avanzate).
11. Seleziona Enable Network (Abilita la rete).
12. Per VPC, scegli il VPC a cui la tua funzione avrà accesso.
13. Per Sottoreti, scegli le sottoreti VPC che utilizzerà Lambda.
14. Per Gruppi di sicurezza, scegli i gruppi di sicurezza VPC che utilizzerà Lambda.
15. Scegli Crea funzione. La AWS Lambda console apre la pagina di configurazione della funzione e inizia a creare la funzione.

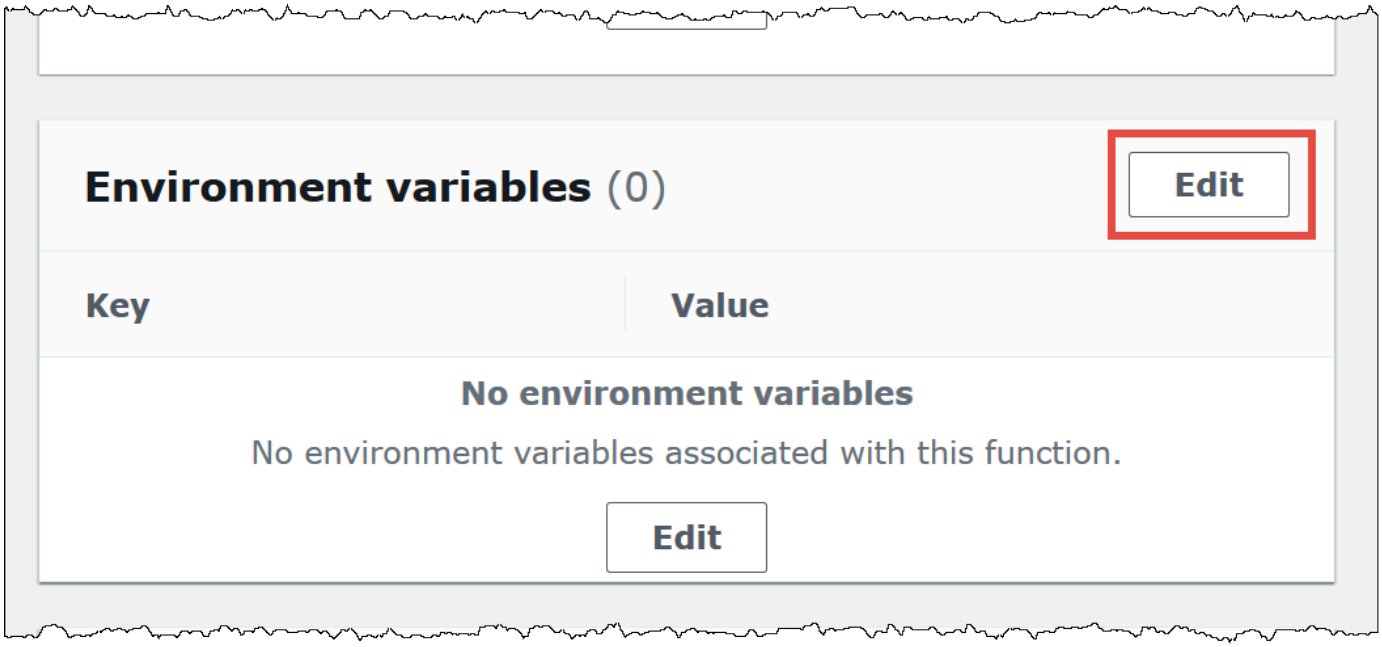
Carica il codice e configura la funzione Lambda

Quando la console informa che la funzione è stata creata correttamente, è possibile caricare il codice della funzione e configurarne le variabili di ambiente.

Per caricare il codice della funzione Lambda e configurare le variabili di ambiente

1. Nella console Lambda, assicurati di utilizzare la scheda Code (Codice) della pagina della funzione specificata.

2. Per Code source (Origine codice), scegli Upload from (Carica da), quindi scegli .zip or .jar file (file .zip o .jar).
3. Carica il file `hms-lambda-func-1.0-SNAPSHOT-withdep.jar` generato in precedenza.
4. Nella pagina della funzione Lambda, scegli la scheda Configuration (Configurazione).
5. Dal riquadro a sinistra, scegli Environment variables (Variabili di ambiente).
6. Nella sezione Variabili di ambiente, scegliere Modifica.

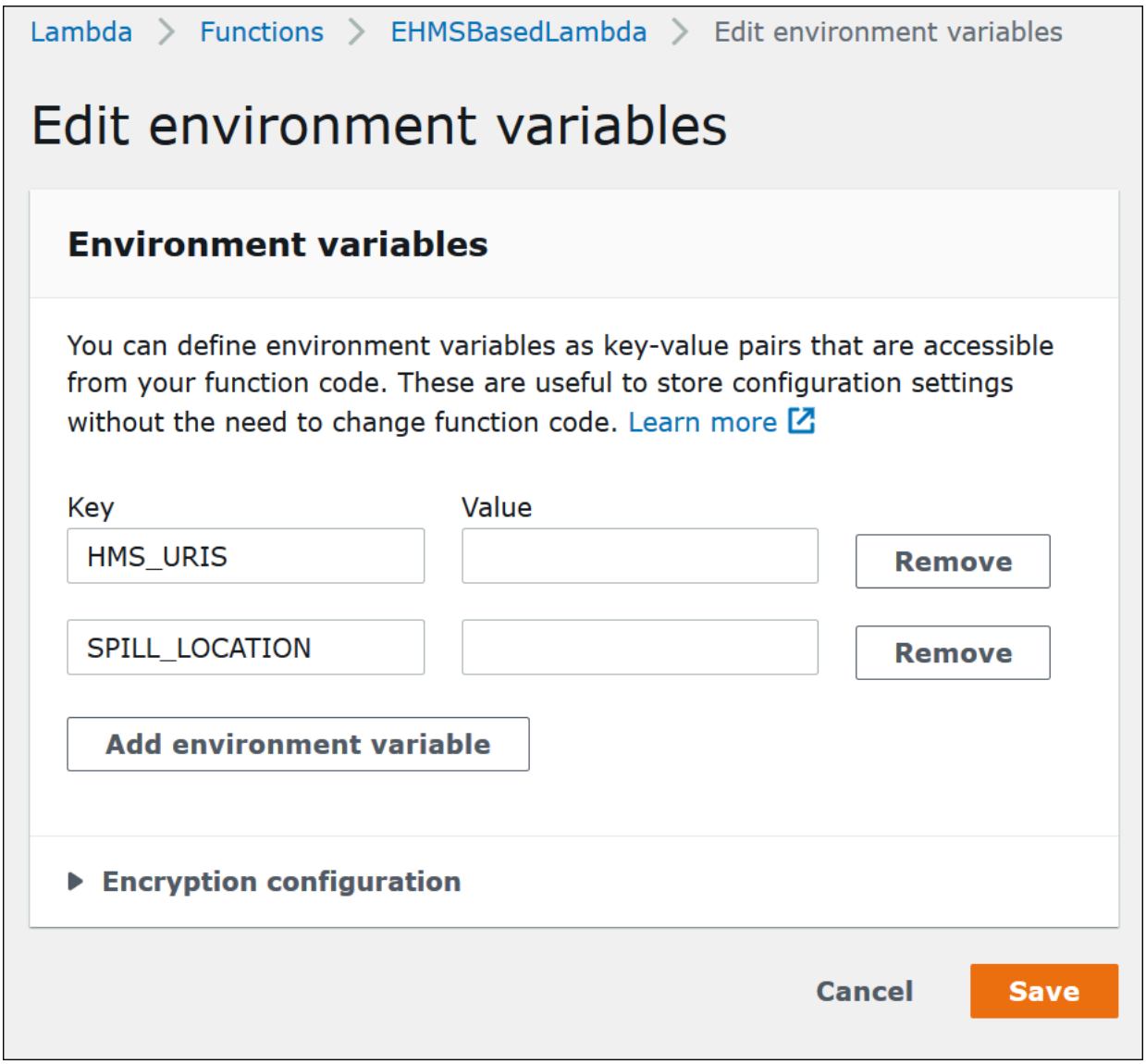


7. Nella pagina Edit environment variables (Modifica variabili di ambiente), utilizza l'opzione Add environment variable (Aggiungi variabile di ambiente) per aggiungere le chiavi e i valori delle variabili di ambiente seguenti:

   - HMS_URIS: usa la seguente sintassi per inserire l'URI dell'host del metastore Hive che utilizza il protocollo Thrift alla porta 9083.

     ```
     thrift://<host_name>:9083
     ```

   - SPILL_LOCATION: specifica una posizione Amazon S3 nell'account Amazon Web Services per contenere metadati di spillover se la dimensione di risposta della funzione Lambda supera i 4 MB.



8. Selezionare Salva.

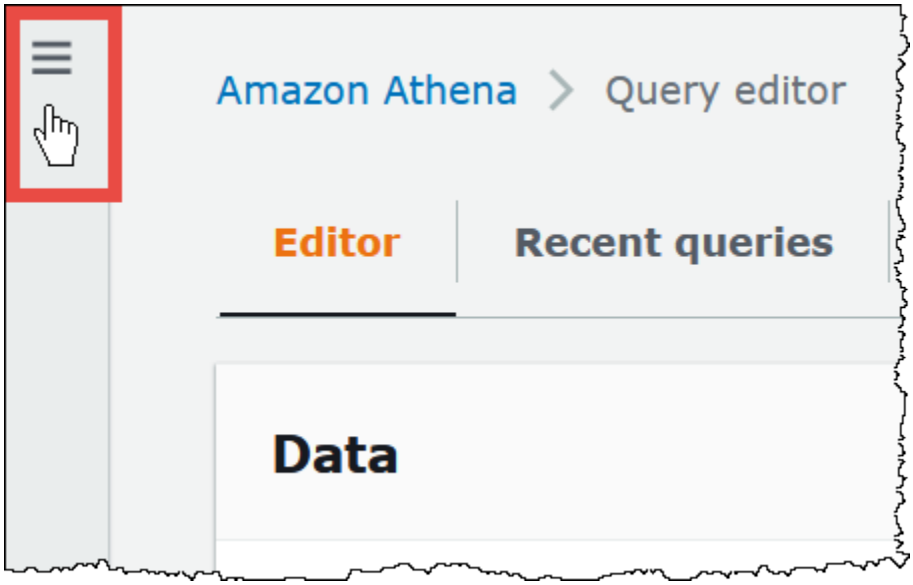
A questo punto, puoi configurare Athena per utilizzare la funzione Lambda per connetterti al metastore Hive. Per le fasi, consulta Configurazione di Athena per l'utilizzo di un connettore Hive Metastore distribuito.

Configurazione di Athena per l'utilizzo di un connettore Hive Metastore distribuito

Dopo aver distribuito un connettore di origine dati Lambda come `AthenaHiveMetastoreFunction` sul tuo account, puoi configurare Athena per usarlo. Per farlo, devi creare un nome di origine dati che faccia riferimento al metastore Hive esterno da utilizzare nelle query Athena.

Per connettere Athena al metastore Hive utilizzando una funzione Lambda esistente

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.



3. Scegli Data sources (Origini dati).
4. Nella pagina Data sources (Origini dati), scegli Create data source (Crea origine dati).
5. Nella pagina Choose a data source (Scegli un'origine dati), per Data source (Origini dati), scegli S3 - Apache Hive metastore (Metastore Apache Hive - S3).
6. Seleziona Successivo.
7. Nella sezione Data source details (Dettagli origine dati), per Data source name (Nome origine dati), inserisci il nome che desideri utilizzare nelle istruzioni SQL quando si esegue una query sull'origine dati da Athena (ad esempio, `MyHiveMetastore`). Il nome può contenere fino a 127 caratteri e deve essere univoco all'interno dell'account. Non può essere modificato dopo la creazione. I caratteri validi sono a-z, A-z, 0-9, _ (trattino basso), @ (chiocciola) e - (trattino). I nomi `awsdatacatalog`, `hive`, `jmx` e `system` sono riservati ad Athena e non possono essere utilizzati per i nomi delle origini dati.
8. Nella sezione Connection details (Dettagli di connessione), usa la casella Select or enter a Lambda function (Seleziona o inserisci una funzione Lambda) per scegliere il nome della funzione appena creata. Viene visualizzato l'ARN della funzione Lambda.
9. (Facoltativo) Per Tags (Tag), aggiungi coppie chiave-valore da associare a questa origine dati. Per ulteriori informazioni sui tag, consulta Assegnazione di tag alle risorse Athena.
10. Seleziona Successivo.

11. Nella pagina Review and create (Rivedi e crea), esamina i dettagli dell'origine dati, quindi scegli Create data source (Crea origine dati).
12. La sezione Data source details (Dettagli sull'origine dati) della pagina dell'origine dati mostra le informazioni relative al nuovo connettore.

    È ora possibile utilizzare il Data source name (Nome origine dati) specificato per fare riferimento al metastore Hive nelle query SQL in Athena.

    Nelle query SQL utilizzare la sintassi di esempio seguente, sostituendo `ehms-catalog` con il nome dell'origine dati specificato in precedenza.

    ```
    SELECT * FROM ehms-catalog.CustomerData.customers
    ```
13. Per visualizzare, modificare o eliminare le origini dati create, consulta Gestione delle origini dati.

Utilizzo del nome di un'origine dati di default nelle query di un metastore Hive esterno

Quando si eseguono query DML e DDL su metastore Hive esterne, è possibile semplificare la sintassi della query omettendo il nome del catalogo se tale nome è selezionato nell'editor di query. Alcune restrizioni si applicano a questa funzionalità.

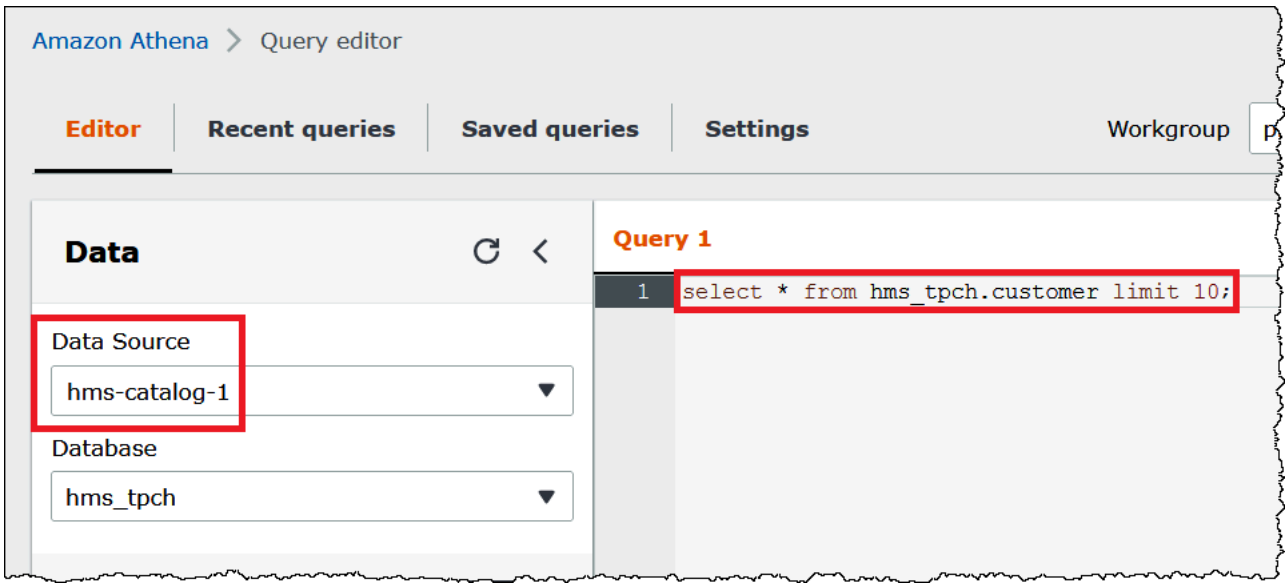
Istruzioni DML

Per eseguire query con cataloghi registrati

1. È possibile inserire il nome dell'origine dati prima del database utilizzando la sintassi `[[`*`data_source_name`*`].`*`database_name`*`].`*`table_name`*, come nell'esempio seguente.

    ```
    select * from  "hms-catalog-1".hms_tpch.customer limit 10;
    ```
2. Quando l'origine dati che si desidera utilizzare è già selezionata nell'editor di query, è possibile omettere il nome dalla query, come nell'esempio seguente.

    ```
    select * from hms_tpch.customer limit 10:
    ```



3. Quando si utilizzano più origini dati in una query, è possibile omettere solo il nome dell'origine dati di default ed è necessario specificare il nome completo per le origini dati non di default.

   Ad esempio, supponiamo che `AwsDataCatalog` sia selezionato come origine dati di default nell'editor di query. L'`FROM`istruzione nel seguente estratto di query qualifica completamente i primi due nomi di origini dati, ma omette il nome della terza origine dati perché si trova nel catalogo dati. AWS Glue

   ```
   ...
   FROM ehms01.hms_tpch.customer,
            "hms-catalog-1".hms_tpch.orders,
            hms_tpch.lineitem
   ...
   ```

Istruzioni DDL

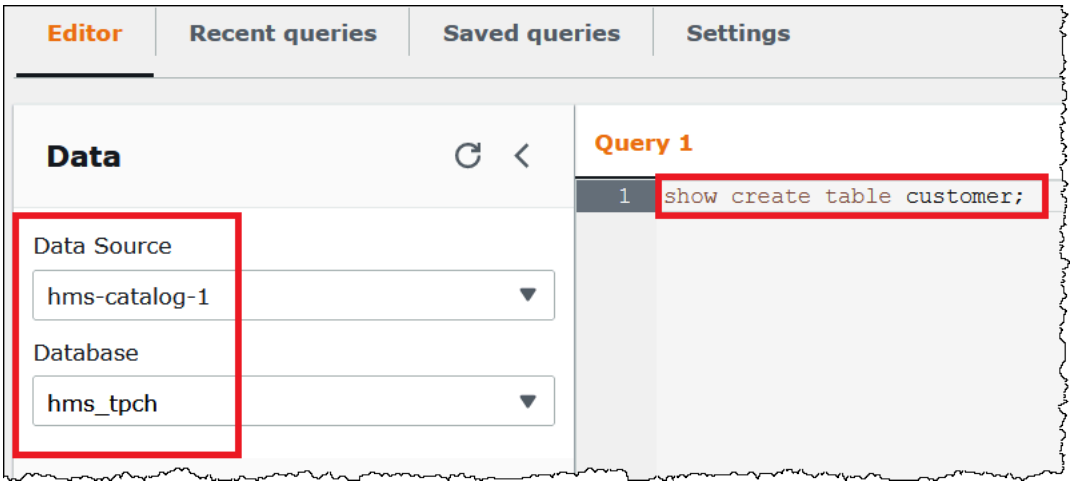
Le seguenti istruzioni DDL Athena supportano i prefissi dei nomi di catalogo. I prefissi dei nomi di catalogo in altre istruzioni DDL causano errori di sintassi.

```
SHOW TABLES [IN [catalog_name.]database_name] ['regular_expression']

SHOW TBLPROPERTIES [[catalog_name.]database_name.]table_name [('property_name')]

SHOW COLUMNS IN [[catalog_name.]database_name.]table_name
```

```
SHOW PARTITIONS [[catalog_name.]database_name.]table_name

SHOW CREATE TABLE [[catalog_name.][database_name.]table_name

DESCRIBE [EXTENDED | FORMATTED] [[catalog_name.][database_name.]table_name [PARTITION
 partition_spec] [col_name ( [.field_name] | [.'$elem$'] | [.'$key$'] | [.'$value$'] )]
```

Come per le istruzioni DML, è possibile omettere i prefissi dell'origine dati e del database dalla query quando l'origine dati e il database vengono selezionati nell'editor di query.

Nell'immagine seguente, l'origine dati `hms-catalog-1` e il database `hms_tpch` vengono selezionati nell'editor di query. L'istruzione`show create table customer` ha esito positivo anche se il prefisso `hms-catalog-1` e il nome del database `hms_tpch` vengono viene omessi dalla query stessa.



Definizione di un'origine dati di default in una stringa di connessione JDBC

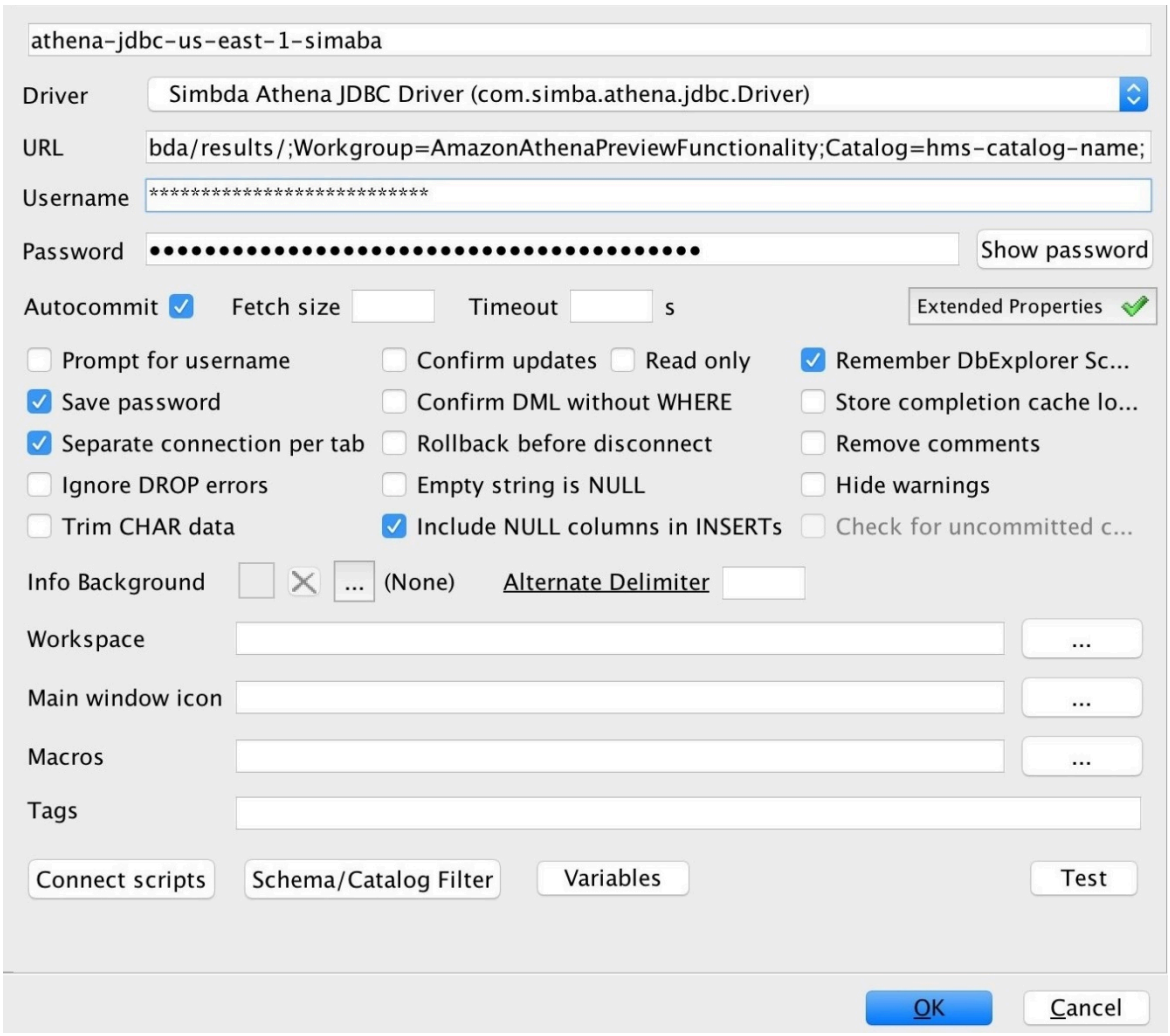
Quando si utilizza il driver Athena JDBC per collegare Athena a un metastore Hive esterno, è possibile utilizzare il parametro `Catalog` per specificare il nome dell'origine dati di default nella stringa di connessione in un editor SQL come SQL Workbench.

> **Note**
>
> Per scaricare i driver Athena JDBC più recenti, consultare Utilizzo di Athena con il driver JDBC.

La seguente stringa di connessione specifica l'origine dati predefinita. *`hms-catalog-name`*

```
    jdbc:awsathena://AwsRegion=us-east-1;S3OutputLocation=s3://DOC-EXAMPLE-BUCKET/
lambda/results/;Workgroup=AmazonAthenaPreviewFunctionality;Catalog=hms-catalog-name;
```

L'immagine seguente mostra un URL di connessione JDBC di esempio come configurato in SQL Workbench.



Utilizzo delle visualizzazioni Hive

È possibile utilizzare Athena per eseguire query sulle visualizzazioni Apache create in metastore Hive esterni. Athena traduce le tue visualizzazioni per te on-the-fly in fase di esecuzione senza modificare la vista originale o memorizzare la traduzione.

Ad esempio, supponiamo di avere una visualizzazione Hive come la seguente, che utilizza una sintassi non supportata in Athena come `LATERAL VIEW explode()`:

```
CREATE VIEW team_view AS
SELECT team, score
FROM matches
LATERAL VIEW explode(scores) m AS score
```

Athena traduce la stringa di query di visualizzazione Hive in un'istruzione simile alla seguente, che è in grado di eseguire:

```
SELECT team, score
FROM matches
CROSS JOIN UNNEST(scores) AS m (score)
```

Per informazioni sul collegamento di un metastore Hive esterno ad Athena, consulta la sezione Utilizzo di un connettore dati Athena per il metastore Hive esterno.

Considerazioni e limitazioni

Durante l'esecuzione di query sulle visualizzazioni Hive da Athena, considera i seguenti punti:

- Athena non supporta la creazione di visualizzazioni Hive. È possibile creare visualizzazioni Hive nel metastore esterno di Hive, sulle quali è possibile poi eseguire query da Athena.
- Athena non supporta le funzioni definite dall'utente personalizzate per le visualizzazioni Hive.
- A causa di un problema noto nella console Athena, le visualizzazioni Hive vengono mostrate nell'elenco delle tabelle anziché in quello delle visualizzazioni.
- Sebbene il processo di traduzione sia automatico, alcune funzioni Hive non sono supportate per le visualizzazioni Hive o richiedono una gestione speciale. Per ulteriori informazioni, consulta la sezione seguente.

Limitazioni nel supporto delle funzioni Hive

Questa sezione evidenzia le funzioni Hive che Athena non supporta per le visualizzazioni Hive o che richiedono un trattamento speciale. Attualmente, poiché Athena supporta principalmente le funzioni di Hive 2.2.0, le funzioni presenti solo nelle versioni successive (come Hive 4.0.0) non sono disponibili. Per un elenco completo delle funzioni Hive, consultaManuale del linguaggio Hive UDF.

Funzioni di aggregazione

Funzioni di aggregazione che richiedono una gestione speciale

La seguente funzione di aggregazione per le visualizzazioni Hive richiede una gestione speciale.

- Avg: al posto di `avg(INT i)`, utilizza `avg(CAST(i AS DOUBLE))`.

Funzioni aggregate non supportate

Le seguenti funzioni di aggregazione Hive non sono supportate in Athena per le visualizzazioni Hive.

```
covar_pop
histogram_numeric
ntile
percentile
percentile_approx
```

Funzioni di regressione come `regr_count`, `regr_r2` e `regr_sxx` non sono supportate in Athena per le visualizzazioni Hive.

Funzioni aggregate non supportate

Le seguenti funzioni di aggregazione Hive non sono supportate in Athena per le visualizzazioni Hive.

```
date_format(date/timestamp/string ts, string fmt)
day(string date)
dayofmonth(date)
extract(field FROM source)
hour(string date)
minute(string date)
month(string date)
quarter(date/timestamp/string)
second(string date)
weekofyear(string date)
year(string date)
```

Funzioni aggregate non supportate

Funzioni di regressione come `mask()`, e `mask_first_n()` non sono supportate in Athena per le visualizzazioni Hive.

Funzioni varie

Funzioni varie che richiedono una gestione speciale

Le seguenti funzioni varie per le visualizzazioni Hive richiedono una gestione speciale.

- md5: Athena supporta `md5(binary)` ma non `md5(varchar)`.
- Explode: Athena supporta `explode` quando viene utilizzato nella seguente sintassi:

```
LATERAL VIEW [OUTER] EXPLODE(<argument>)
```

- Explode: Athena supporta `posexplode` quando viene utilizzato nella seguente sintassi:

```
LATERAL VIEW [OUTER] POSEXPLODE(<argument>)
```

  Nell'output `(pos, val)`, Athena tratta la colonna `pos` come `BIGINT`. Per questo motivo, potrebbe essere necessario eseguire il cast della colonna `pos` a `BIGINT` per evitare una visualizzazione obsoleta. Nell'esempio seguente viene descritta tale tecnica.

```
SELECT CAST(c AS BIGINT) AS c_bigint, d
FROM table LATERAL VIEW POSEXPLODE(<argument>) t AS c, d
```

Funzioni varie non supportate

Le seguenti funzioni di aggregazione Hive non sono supportate in Athena per le visualizzazioni Hive.

```
aes_decrypt
aes_encrypt
current_database
current_user
inline
java_method
logged_in_user
reflect
sha/sha1/sha2
stack
version
```

Operatori

Operatori che richiedono una gestione speciale

I seguenti operatori per le visualizzazioni Hive richiedono una gestione speciale.

- Operatore mod o modulo (%): poiché il tipo `DOUBLE` esegue implicitamente un cast a `DECIMAL(x,y)`, la sintassi seguente può causare il messaggio di errore View is stale (La visualizzazione è non aggiornata):

  ```
  a_double % 1.0 AS column
  ```

  Per risolvere il problema, utilizza `CAST`, come nell'esempio seguente.

  ```
  CAST(a_double % 1.0 as DOUBLE) AS column
  ```

- Operatore divisione (/): in Hive,`int` diviso per `int` produce un `double`. In Athena, la stessa operazione produce un `int` abbreviato.

Operatori non supportati

Athena non supporta i seguenti operatori per le visualizzazioni Hive.

~A: bitwise `NOT`

A ^ b: bitwise `XOR`

A & b: bitwise `AND`

A | b: bitwise `OR`

A **<=>** b: Restituisce lo stesso risultato dell'operatore uguale (=) per operandi non nulli. Restituisce `TRUE` se entrambi sono `NULL`, `FALSE` se uno di questi è `NULL`.

Funzioni stringa

Funzioni stringa che richiedono una gestione speciale

Le seguenti funzioni stringa per le visualizzazioni Hive richiedono una gestione speciale.

- chr (bigint|doppia a): Hive consente argomenti negativi; Athena no.

- instr (string str, string substr): poiché la mappatura Athena per la funzione `instr` restituisce `BIGINT` anziché `INT`, utilizza la sintassi seguente:

  ```
  CAST(instr(string str, string substr) as INT)
  ```

  Senza questo passaggio, la visualizzazione sarà considerata obsoleta.

- lunghezza (stringa a): poiché la mappatura Athena per la funzione `length` restituisce `BIGINT` anziché `INT`, utilizza la seguente sintassi in modo che la visualizzazione non venga considerata obsoleta:

  ```
  CAST(length(string str) as INT)
  ```

Funzioni stringa non supportate

Le seguenti funzioni stringa di Hive non sono supportate nelle visualizzazioni Hive in Athena.

```
ascii(string str)
character_length(string str)
decode(binary bin, string charset)
encode(string src, string charset)
elt(N int,str1 string,str2 string,str3 string,...)
field(val T,val1 T,val2 T,val3 T,...)
find_in_set(string str, string strList)
initcap(string A)
levenshtein(string A, string B)
locate(string substr, string str[, int pos])
octet_length(string str)
parse_url(string urlString, string partToExtract [, string keyToExtract])
printf(String format, Obj... args)
quote(String text)
regexp_extract(string subject, string pattern, int index)
repeat(string str, int n)
sentences(string str, string lang, string locale)
soundex(string A)
space(int n)
str_to_map(text[, delimiter1, delimiter2])
substring_index(string A, string delim, int count)
```

Funzioni XPath non supportate

Le funzioni XPath di Hive come `xpath`, `xpath_short`, e `xpath_int` non sono supportate in Athena per le visualizzazioni Hive.

Risoluzione dei problemi

Quando si utilizzano le visualizzazioni Hive in Athena, potrebbero verificarsi i seguenti problemi:

- View ***<view name>*** is stale (La visualizzazione <nome visualizzazione> non è aggiornata): questo messaggio di solito indica una differenza nella tipologia di visualizzazione in Hive e in Athena. Se la stessa funzione nella documentazione delle funzioni e degli operatori di Hive LanguageManual UDF e Presto ha firme diverse, prova a inserire il tipo di dati non corrispondente.
- Function not registered (Funzione non registrata): Athena non supporta attualmente la funzione. Per ulteriori informazioni, consulta le informazioni descritte precedentemente in questo documento.

Utilizzo di AWS CLI with Hive metastores

È possibile utilizzare i comandi CLI `aws athena` per gestire i cataloghi dati di metastore Hive utilizzati con Athena. Dopo aver definito uno o più cataloghi da utilizzare con Athena, è possibile fare riferimento a tali cataloghi nei comandi DDL e DML di `aws athena`.

Utilizzo di per gestire i cataloghi metastore di Hive AWS CLI

Registrazione di un catalogo: C reate-data-catalog

Per registrare un catalogo dati, utilizzare il comando `create-data-catalog`. Utilizzare il parametro `name` per specificare il nome che si desidera utilizzare per il catalogo. Trasmetti l'ARN della funzione Lambda all'opzione `metadata-function` dell'argomento `parameters`. Per creare tag per il nuovo catalogo, utilizzare il parametro `tags` con una o più coppie di argomenti `Key=`*`key`*`,Value=`*`value`* separate da spazi.

Nell'esempio seguente viene registrato il catalogo del metastore Hive denominato `hms-catalog-1`. Il comando è stato formattato per la leggibilità.

```
$ aws athena create-data-catalog
 --name "hms-catalog-1"
 --type "HIVE"
 --description "Hive Catalog 1"
 --parameters "metadata-function=arn:aws:lambda:us-
east-1:111122223333:function:external-hms-service-v3,sdk-version=1.0"
```

```
 --tags Key=MyKey,Value=MyValue
 --region us-east-1
```

Mostra i dettagli del catalogo: G et-data-catalog

Per visualizzare i dettagli di un catalogo, passare il nome del catalogo al comando `get-data-catalog`, come nell'esempio seguente.

```
$ aws athena get-data-catalog --name "hms-catalog-1" --region us-east-1
```

Il seguente risultato di esempio è in formato JSON.

```
{
    "DataCatalog": {
        "Name": "hms-catalog-1",
        "Description": "Hive Catalog 1",
        "Type": "HIVE",
        "Parameters": {
            "metadata-function": "arn:aws:lambda:us-
east-1:111122223333:function:external-hms-service-v3",
            "sdk-version": "1.0"
        }
    }
}
```

Elenco dei cataloghi registrati: L ist-data-catalogs

Per elencare i cataloghi registrati, utilizzare il comando `list-data-catalogs` e, facoltativamente, specificare una regione, come nell'esempio seguente. I cataloghi elencati includono sempre AWS Glue.

```
$ aws athena list-data-catalogs --region us-east-1
```

Il seguente risultato di esempio è in formato JSON.

```
{
    "DataCatalogs": [
        {
            "CatalogName": "AwsDataCatalog",
            "Type": "GLUE"
        },
        {
```

```
            "CatalogName": "hms-catalog-1",
            "Type": "HIVE",
            "Parameters": {
                "metadata-function": "arn:aws:lambda:us-
east-1:111122223333:function:external-hms-service-v3",
                "sdk-version": "1.0"
            }
        }
    ]
}
```

Aggiornamento di un catalogo: U pdate-data-catalog

Per aggiornare un catalogo dati, utilizzare il comando `update-data-catalog`, come nell'esempio seguente. Il comando è stato formattato per la leggibilità.

```
$ aws athena update-data-catalog
 --name "hms-catalog-1"
 --type "HIVE"
 --description "My New Hive Catalog Description"
 --parameters "metadata-function=arn:aws:lambda:us-
east-1:111122223333:function:external-hms-service-new,sdk-version=1.0"
 --region us-east-1
```

Eliminazione di un catalogo: D elete-data-catalog

Per eliminare un catalogo dati, utilizzare il comando `delete-data-catalog`, come nell'esempio seguente.

```
$ aws athena delete-data-catalog --name "hms-catalog-1" --region us-east-1
```

Visualizzazione dei dettagli del database: get-database

Per visualizzare i dettagli di un database, passare il nome del catalogo e del database al comando `get-database`, come nell'esempio seguente.

```
$ aws athena get-database --catalog-name hms-catalog-1 --database-name mydb
```

Il seguente risultato di esempio è in formato JSON.

```
{
    "Database": {
```

```
        "Name": "mydb",
        "Description": "My database",
        "Parameters": {
            "CreatedBy": "Athena",
            "EXTERNAL": "TRUE"
        }
    }
}
```

Elencazione dei database in un catalogo: list-databases

Per elencare i database in un catalogo, utilizzare il comando `list-databases` e, facoltativamente, specificare una regione, come nell'esempio seguente.

```
$ aws athena list-databases --catalog-name AwsDataCatalog --region us-west-2
```

Il seguente risultato di esempio è in formato JSON.

```
{
    "DatabaseList": [
        {
            "Name": "default"
        },
        {
            "Name": "mycrawlerdatabase"
        },
        {
            "Name": "mydatabase"
        },
        {
            "Name": "sampledb",
            "Description": "Sample database",
            "Parameters": {
                "CreatedBy": "Athena",
                "EXTERNAL": "TRUE"
            }
        },
        {
            "Name": "tpch100"
        }
    ]
}
```

Visualizzazione dei dettagli della tabella: G et-table-metadata

Per visualizzare i metadati di una tabella, inclusi i nomi delle colonne e i tipi di dati, passare il nome del catalogo, del database e del nome della tabella al comando `get-table-metadata`, come nell'esempio seguente.

```
$ aws athena get-table-metadata --catalog-name AwsDataCatalog --database-name mydb --
table-name cityuseragent
```

Il seguente risultato di esempio è in formato JSON.

```
{
    "TableMetadata": {
        "Name": "cityuseragent",
            "CreateTime": 1586451276.0,
            "LastAccessTime": 0.0,
            "TableType": "EXTERNAL_TABLE",
            "Columns": [
                {
                    "Name": "city",
                    "Type": "string"
                },
                {
                    "Name": "useragent1",
                    "Type": "string"
                }
            ],
            "PartitionKeys": [],
            "Parameters": {
                "COLUMN_STATS_ACCURATE": "false",
                "EXTERNAL": "TRUE",
                "inputformat": "org.apache.hadoop.mapred.TextInputFormat",
                "last_modified_by": "hadoop",
                "last_modified_time": "1586454879",
                "location": "s3://DOC-EXAMPLE-BUCKET/",
                "numFiles": "1",
                "numRows": "-1",
                "outputformat":
 "org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat",
                "rawDataSize": "-1",
                "serde.param.serialization.format": "1",
                "serde.serialization.lib":
 "org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe",
```

```
                "totalSize": "61"
            }
        }
}
```

Visualizzazione dei metadati per tutte le tabelle di un database: L ist-table-metadata

Per visualizzare i metadati per tutte le tabelle di un database, passare il nome del catalogo e del nome del database al comando `list-table-metadata`. Il comando `list-table-metadata` è simile al comando `get-table-metadata`, ad eccezione del fatto che non si specifica un nome di tabella. Per limitare il numero di risultati, è possibile utilizzare l'opzione `--max-results`, come nell'esempio seguente.

```
$ aws athena list-table-metadata --catalog-name AwsDataCatalog --database-name sampledb
 --region us-east-1 --max-results 2
```

Il seguente risultato di esempio è in formato JSON.

```
{
    "TableMetadataList": [
        {
            "Name": "cityuseragent",
            "CreateTime": 1586451276.0,
            "LastAccessTime": 0.0,
            "TableType": "EXTERNAL_TABLE",
            "Columns": [
                {
                    "Name": "city",
                    "Type": "string"
                },
                {
                    "Name": "useragent1",
                    "Type": "string"
                }
            ],
            "PartitionKeys": [],
            "Parameters": {
                "COLUMN_STATS_ACCURATE": "false",
                "EXTERNAL": "TRUE",
                "inputformat": "org.apache.hadoop.mapred.TextInputFormat",
                "last_modified_by": "hadoop",
                "last_modified_time": "1586454879",
```

```
                "location": "s3://DOC-EXAMPLE-BUCKET/",
                "numFiles": "1",
                "numRows": "-1",
                "outputformat":
"org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat",
                "rawDataSize": "-1",
                "serde.param.serialization.format": "1",
                "serde.serialization.lib":
"org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe",
                "totalSize": "61"
            }
        },
        {
            "Name": "clearinghouse_data",
            "CreateTime": 1589255544.0,
            "LastAccessTime": 0.0,
            "TableType": "EXTERNAL_TABLE",
            "Columns": [
                {
                    "Name": "location",
                    "Type": "string"
                },
                {
                    "Name": "stock_count",
                    "Type": "int"
                },
                {
                    "Name": "quantity_shipped",
                    "Type": "int"
                }
            ],
            "PartitionKeys": [],
            "Parameters": {
                "EXTERNAL": "TRUE",
                "inputformat": "org.apache.hadoop.mapred.TextInputFormat",
                "location": "s3://DOC-EXAMPLE-BUCKET/",
                "outputformat":
"org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat",
                "serde.param.serialization.format": "1",
                "serde.serialization.lib":
"org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe",
                "transient_lastDdlTime": "1589255544"
            }
        }
```

```
    ],
    "NextToken":
 "eyJsYXN0RXZhbHVhdGVkS2V5Ijp7IkhBU0hfS0VZIjp7InMiOiJ0Ljk0YWZjYjk1MjJjNTQ1YmU4Y2I5OWE5NTg0MjFjY
}
```

Esecuzione di istruzioni DDL e DML

Quando si utilizza il AWS CLI per eseguire istruzioni DDL e DML, è possibile passare il nome del catalogo dei metastore Hive in due modi:

- Direttamente nelle istruzioni che lo supportano.
- Al parametro `--query-execution-context` `Catalog`.

Istruzioni DDL

L'esempio seguente passa il nome del catalogo direttamente come parte dell'istruzione DDL `show create table`. Il comando è stato formattato per la leggibilità.

```
$ aws athena start-query-execution
 --query-string "show create table hms-catalog-1.hms_tpch_partitioned.lineitem"
 --result-configuration "OutputLocation=s3://DOC-EXAMPLE-BUCKET/lambda/results"
```

Nell'esempio seguente l'istruzione DDL `show create table` usa il parametro `Catalog` di `--query-execution-context` per passare il nome del catalogo del metastore Hive `hms-catalog-1`. Il comando è stato formattato per la leggibilità.

```
$ aws athena start-query-execution
 --query-string "show create table lineitem"
 --query-execution-context "Catalog=hms-catalog-1,Database=hms_tpch_partitioned"
 --result-configuration "OutputLocation=s3://DOC-EXAMPLE-BUCKET/lambda/results"
```

Istruzioni DML

La seguente istruzione DML `select` di esempio passa direttamente il nome del catalogo nella query. Il comando è stato formattato per la leggibilità.

```
$ aws athena start-query-execution
 --query-string "select * from hms-catalog-1.hms_tpch_partitioned.customer limit 100"
 --result-configuration "OutputLocation=s3://DOC-EXAMPLE-BUCKET/lambda/results"
```

La seguente istruzione DML `select` di esempio utilizza il parametro `Catalog` di `--query-execution-context` per passare il nome del catalogo del metastore Hive `hms-catalog-1`. Il comando è stato formattato per la leggibilità.

```
$ aws athena start-query-execution
 --query-string "select * from customer limit 100"
 --query-execution-context "Catalog=hms-catalog-1,Database=hms_tpch_partitioned"
 --result-configuration "OutputLocation=s3://DOC-EXAMPLE-BUCKET/lambda/results"
```

Implementazione di riferimento

Athena fornisce un'implementazione di riferimento del suo connettore per il metastore Hive esterno su .com all'indirizzo https://github.com/awslabs/. GitHub aws-athena-hive-metastore

L'implementazione di riferimento è un progetto Apache Maven che presenta i seguenti moduli:

- **`hms-service-api`** – Contiene le operazioni API tra la funzione Lambda e i client del servizio Athena. Queste operazioni API sono definite nell'interfaccia `HiveMetaStoreService`. Poiché si tratta di un contratto di servizio, non è necessario modificare nulla in questo modulo.
- **`hms-lambda-handler`** – Un insieme di gestori Lambda predefiniti che elaborano tutte le chiamate API al metastore Hive. La classe `MetadataHandler` è il dispatcher per tutte le chiamate API. Non è necessario modificare questo pacchetto.
- **`hms-lambda-func`** – Una funzione Lambda di esempio con i seguenti componenti.
  - **`HiveMetaStoreLambdaFunc`** Una funzione Lambda di esempio che estende `MetadataHandler`.
  - **`ThriftHiveMetaStoreClient`**: Un client Thrift che comunica con il metastore Hive. Questo client è stato predisposto per Hive 2.3.0. Se si utilizza una versione di Hive diversa, potrebbe essere necessario aggiornare questa classe per assicurarsi che gli oggetti di risposta siano compatibili.
  - **`ThriftHiveMetaStoreClientFactory`** – Controlla il comportamento della funzione Lambda. Ad esempio, è possibile fornire il proprio set di provider di gestori sovrascrivendo il metodo `getHandlerProvider()`.
  - `hms.properties` – Configura funzione Lambda La maggior parte dei casi d'uso richiede solo l'aggiornamento delle due proprietà seguenti.
    - `hive.metastore.uris`: l'URI del metastore Hive nel formato `thrift://<host_name>:9083`.

- `hive.metastore.response.spill.location`: la posizione Amazon S3 in cui archiviare gli oggetti di risposta quando le loro dimensioni superano una determinata soglia (ad esempio, 4 MB). La soglia viene definita nella proprietà `hive.metastore.response.spill.threshold`. La modifica del valore predefinito non è consigliata.

> **Note**
>
> Queste due proprietà possono essere sovrascritte dalle variabili di ambiente Lambda `HMS_URIS` e `SPILL_LOCATION`. Utilizzare queste variabili invece di ricompilare il codice sorgente per la funzione Lambda quando si desidera utilizzare la funzione con una diversa posizione del metastore Hive o con una diversa posizione di spill.

- **`hms-lambda-layer`**: un progetto assembly di Maven che inserisce `hms-service-api`, `hms-lambda-handler` e le loro dipendenze in un file `.zip`. Il file `.zip` viene registrato come layer Lambda per l'utilizzo da parte di più funzioni Lambda.
- **`hms-lambda-rnp`**: registra le risposte di una funzione Lambda e le utilizza per riprodurre la risposta. Puoi utilizzare questo modello per simulare le risposte Lambda per i test.

Costruire gli artefatti in modo autonomo

La maggior parte dei casi d'uso non richiede la modifica dell'implementazione di riferimento. Tuttavia, se necessario, è possibile modificare il codice sorgente, creare manualmente gli artefatti e caricarli in una posizione Amazon S3.

Prima di creare gli artefatti, aggiornare le proprietà `hive.metastore.uris` e `hive.metastore.response.spill.location` nel file `hms.properties` nel modulo `hms-lambda-func`.

Per costruire gli artefatti, è necessario avere installato Apache Maven ed eseguire il comando `mvn install`. Questo genera il layer del file `.zip` nella cartella di output chiamata `target` nel modulo `hms-lambda-layer` e il file `.jar` della funzione Lambda nel modulo `hms-lambd-func`.

Utilizzo di Amazon Athena Federated Query

Se disponi di dati in origini diverse da Amazon S3, puoi utilizzare Athena Federated Query per eseguire query locali sui dati o compilare pipeline che estraggono i dati da più origini dati e archiviarli

in Amazon S3. Con Athena Federated Query, puoi eseguire query SQL su dati archiviati in origini dati relazionali, non relazionali, oggetto e personalizzate.

Athena utilizza connettori di origine dati che funzionano AWS Lambda per eseguire query federate. Un connettore origine dati è una parte di codice in grado di effettuare la conversione tra l'origine dati di destinazione e Athena. Puoi pensare a un connettore come a un'estensione del motore di query di Athena. Esistono connettori di origine dati Athena predefiniti per fonti di dati come Amazon Logs, CloudWatch Amazon DynamoDB, Amazon DocumentDB e Amazon RDS e fonti di dati relazionali conformi a JDBC come MySQL e PostgreSQL con licenza Apache 2.0. Puoi inoltre utilizzare l'SDK Athena Query Federation per scrivere connettori personalizzati. Per scegliere, configurare e distribuire un connettore origine dati nell'account, puoi utilizzare le console Athena e Lambda o AWS Serverless Application Repository. Dopo aver distribuito i connettori origine dati, il connettore è associato a un catalogo che puoi specificare nelle query SQL. Puoi combinare istruzioni SQL di più cataloghi ed estendere più origini dati con una singola query.

Quando una query viene inviata rispetto a un'origine dati, Athena richiama il connettore corrispondente per identificare le parti delle tabelle che devono essere lette, gestisce il parallelismo ed esegue il push down dei predicati del filtro. In base all'utente che invia la query, i connettori possono fornire o limitare l'accesso a elementi di dati specifici. I connettori utilizzano Apache Arrow come il formato per restituire i dati richiesti in una query, che consente l'implementazione dei connettori in linguaggi quali C, C++, Java, Python e Rust. Poiché i connettori vengono elaborati in Lambda, possono essere utilizzati per accedere ai dati da qualsiasi origine dati sul cloud o in locale che sia accessibile da Lambda.

Per scrivere il tuo connettore origine dati, puoi utilizzare l'SDK Athena Query Federation per personalizzare uno dei connettori precompilati forniti e gestiti da Amazon Athena. Puoi modificare una copia del codice sorgente dal repository e quindi utilizzare lo strumento di pubblicazione Connector per creare il tuo pacchetto. GitHub AWS Serverless Application Repository

> **Note**
>
> Sviluppatori di terze parti potrebbero aver utilizzato Athena Query Federation SDK per scrivere connettori di origini dati. Per problemi di supporto o di licenza relativi a questi connettori di origini dati, contattare il provider di connettori. Questi connettori non sono testati o supportati da AWS.

Per un elenco dei connettori origine dati scritti e testati da Athena, consulta Connettori di origine dati disponibili.

Per informazioni sulla scrittura di un connettore di origine dati personalizzato, vedi Esempio di connettore Athena attivo. GitHub

Considerazioni e limitazioni

- Versioni del motore: Athena Federated Query è supportato solo dalla versione 2 del motore Athena e versioni successive. Per ulteriori informazioni sulle versioni del motore Athena, consulta Controllo delle versioni del motore di Athena.
- Visualizzazioni: puoi creare ed eseguire query sulle viste su origini dati federate. Le viste federate vengono archiviate nell' AWS Glue origine dati sottostante e non nella stessa. Per ulteriori informazioni, consulta Esecuzione di query su visualizzazioni federate.
- Operazioni di scrittura: le operazioni di scrittura come INSERT INTO non sono supportate. Il tentativo di eseguire questa operazione potrebbe generare il messaggio di errore This operation is currently not supported for external catalogs (Questa operazione non è attualmente supportata per i cataloghi esterni).
- Prezzi — Per informazioni sui prezzi, consulta Prezzi di Amazon Athena.

  Driver JDBC — Per utilizzare il driver JDBC con query federate o un metastore Hive esterno, includere `MetadataRetrievalMethod=ProxyAPI` nella stringa di connessione JDBC. Per informazioni sul driver JDBC, consulta Connessione ad Amazon Athena con JDBC.
- Secrets Manager – Per utilizzare la funzione Query federata Athena con AWS Secrets Manager, devi configurare un endpoint privato Amazon VPC per Secrets Manager. Per ulteriori informazioni, consulta Creare un endpoint privato Secrets Manager VPC nella Guida dell'utente di AWS Secrets Manager .

I connettori origine dati potrebbero richiedere l'accesso alle risorse seguenti per funzionare correttamente. Se utilizzi un connettore precompilato, controlla le informazioni relative al connettore per assicurarti di aver configurato correttamente il VPC. Inoltre, assicurati che i principali IAM che eseguono le query e creano i connettori dispongano dei privilegi per le operazioni richieste. Per ulteriori informazioni, consulta Esempio di policy di autorizzazione IAM per consentire la query federata Athena.

- Amazon S3 — Oltre a scrivere i risultati delle query nella posizione dei risultati della query Athena in Amazon S3, i connettori di dati scrivono anche in un bucket di spill in Amazon S3. Sono richieste connettività e autorizzazioni a questa posizione Amazon S3.
- Athena — Le origini dati necessitano di connettività ad Athena e viceversa per controllare lo stato delle query e prevenire l'overscan.
- AWS Glue Data Catalog — Sono necessarie connettività e autorizzazioni se il connettore utilizza il catalogo dati per metadati supplementari o primari.

Video

Guarda i video seguenti per sapere di più sull'utilizzo di Athena Federated Query.

Video: analisi dei risultati di una query federata in Amazon Athena in Amazon QuickSight

Il video seguente mostra come analizzare i risultati di una query federata Athena in Amazon. QuickSight

Analizza i risultati di una query federata in Amazon Athena in Amazon QuickSight

Video: Game Analytics Pipeline

Il video seguente mostra come distribuire una Data Pipeline scalabile serverless per l'acquisizione, l'archiviazione e l'analisi dei dati di telemetria da giochi e servizi utilizzando le query federate di Amazon Athena.

Game Analytics Pipeline

Connettori di origine dati disponibili

In questa sezione sono elencati connettori origine dati Athena precompilati che puoi utilizzare per eseguire query su diverse origini dati esterne ad Amazon S3. Per utilizzare un connettore nelle query Athena, configurarlo e distribuirlo nell'account.

Considerazioni e limitazioni

- Alcuni connettori predefiniti richiedono la creazione di un VPC e un gruppo di sicurezza prima di poterli utilizzare. Per informazioni su come creare i tuoi VPC, consulta Creazione di un VPC per un connettore origine dati.

- Per utilizzare la funzionalità Athena Federated Query con AWS Secrets Manager, devi configurare un endpoint privato Amazon VPC per Secrets Manager. Per ulteriori informazioni, consulta Creare un endpoint privato Secrets Manager VPC nella Guida dell'utente di AWS Secrets Manager .
- Per i connettori che non supportano il pushdown del predicato, le query che includono un predicato richiedono un tempo di esecuzione molto più lungo. Per i set di dati di piccole dimensioni, vengono scansionati pochissimi dati e le query richiedono in media circa 2 minuti. Tuttavia, nel caso dei set di dati di grandi dimensioni, molte query possono andare in timeout.
- Alcune origini dati federati utilizzano una terminologia diversa da Athena per riferirsi a oggetti di dati. Per ulteriori informazioni, consulta Athena e i qualificatori dei nomi delle tabelle federate.
- Per i connettori che non supportano l'impaginazione quando si elencano le tabelle, il servizio web può scadere se il database contiene molte tabelle e metadati. I seguenti connettori forniscono il supporto dell'impaginazione per l'elencazione delle tabelle:
  - DocumentDB
  - DynamoDB
  - MySQL
  - OpenSearch
  - Oracle
  - PostgreSQL
  - Redshift
  - SQL Server

Informazioni aggiuntive

- Per informazioni sulla distribuzione di un connettore origine dati Athena, consulta Distribuzione di un connettore origine dati.
- Per informazioni sulle query che utilizzano connettori di origine dati Athena, consulta Esecuzione di query federate.
- Per informazioni dettagliate sui connettori delle sorgenti dati Athena, consulta Connettori disponibili su. GitHub

Connettori origine dati Athena

- Connettore Amazon Athena Azure Data Lake Storage (ADLS) Gen2
- Connettore Amazon Athena per Azure Synapse

- Connettore Amazon Athena per Cloudera Hive
- Connettore Amazon Athena per Cloudera Impala
- Connettore Amazon Athena CloudWatch
- Connettore Amazon Athena Metrics CloudWatch
- Connettore Amazon Athena CMDB AWS
- Connettore IBM Db2 di Amazon Athena
- Connettore Amazon Athena IBM Db2 AS/400 (Db2 iSeries)
- Connettore Amazon Athena DocumentDB
- Connettore Amazon Athena DynamoDB
- Connettore Google Amazon Athena BigQuery
- Connettore Google Cloud Storage per Amazon Athena
- Connettore Amazon Athena HBase
- Connettore Amazon Athena per Hortonworks
- Connettore Apache Kafka di Amazon Athena
- Connettore MSK di Amazon Athena
- Connettore MySQL di Amazon Athena
- Connettore Amazon Athena Neptune
- Connettore Amazon Athena OpenSearch
- Connettore Amazon Athena per Oracle
- Connettore PostgreSQL di Amazon Athena
- Connettore Amazon Athena Redis
- Connettore Redshift di Amazon Athena
- Connettore Amazon Athena per SAP HANA
- Connettore Amazon Athena per Snowflake
- Connettore Amazon Athena per Microsoft SQL Server
- Connettore Amazon Athena per Teradata
- Connettore Amazon Athena Timestream
- Connettore Amazon Athena TPC Benchmark DS (TPC-DS)
- Connettore Amazon Athena Vertica

> **Note**
>
> La AthenaJdbcConnector(ultima versione 2022.4.1) è stata obsoleta. In alternativa, utilizza un connettore specifico per il database come quelli per MySQL, Redshift o PostgreSQL.

Connettore Amazon Athena Azure Data Lake Storage (ADLS) Gen2

Il connettore Amazon Athena per Azure Data Lake Storage (ADLS) Gen2 consente ad Amazon Athena di eseguire query SQL sui dati archiviati in ADLS. Athena non può accedere direttamente ai file archiviati nel data lake.

- Flusso di lavoro: il connettore implementa l'interfaccia JDBC, che utilizza il driver `com.microsoft.sqlserver.jdbc.SQLServerDriver`. Il connettore passa le query al motore Azure Synapse, che quindi accede al data lake.
- Gestione dei dati e S3: normalmente il connettore Lambda esegue query sui dati direttamente senza trasferirli ad Amazon S3. Tuttavia, quando i dati restituiti dalla funzione Lambda superano i limiti Lambda, i dati vengono scritti nel bucket di spill di Amazon S3 specificato in modo che Athena possa leggere il superamento.
- Autenticazione AAD: AAD può essere usato come metodo di autenticazione per il connettore Azure Synapse. Per utilizzare AAD, la stringa di connessione JDBC utilizzata dal connettore deve contenere i parametri URL `authentication=ActiveDirectoryServicePrincipal`, `AADSecurePrincipalId` e `AADSecurePrincipalSecret`. Questi parametri possono essere passati direttamente o tramite Secrets Manager.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.

- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .
- Nelle condizioni di filtro, è necessario impostare i tipi di dati date e timestamp sul tipo di dati appropriato.

Termini

I seguenti termini si riferiscono al connettore Azure Data Lake Storage Gen2.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore Azure Data Lake Storage Gen2.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
datalakegentwo://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`DataLakeGen2MuxCompositeHandler`
Gestore dei metadati	`DataLakeGen2MuxMetadataHandler`
Gestore dei record	`DataLakeGen2MuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
$*catalog*_connection_string	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `mydatalakegentwocatalog` , il nome della variabile di ambiente è `mydatalakegentwocatalog_connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *AWS_LAMBDA_FUNCTION_NAME* `}`.

Le seguenti proprietà di esempio riguardano una funzione Lambda a DataLakeGen 2 MUX che supporta due istanze di database`datalakegentwo1`: (impostazione predefinita) e. `datalakegentwo2`

Proprietà	Valore
`default`	`datalakegentwo://jdbc:sqlserver://adlsgentwo1. hostname:port;databaseName= database_name ; ${secret1_name }`
`datalakegentwo_catalog1_connection_string`	`datalakegentwo://jdbc:sqlserver://adlsgentwo1. hostname:port;databaseName= database_name ; ${secret1_name }`
`datalakegentwo_catalog2_connection_string`	`datalakegentwo://jdbc:sqlserver://adlsgentwo2. hostname:port;databaseName= database_name ; ${secret2_name }`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti hardcoded in AWS Secrets Manager, consultate Move hardcoded secret in nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

```
{"username": "${username}", "password": "${password}"}
```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${secret1_name}`.

```
datalakegentwo://jdbc:sqlserver://hostname:port;databaseName=database_name;
${secret1_name}
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
datalakegentwo://
jdbc:sqlserver://
hostname:port;databaseName=database_name;user=user_name;password=password
```

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza di Azure Data Lake Storage Gen2.

Tipo di gestore	Classe
Gestore composito	`DataLakeGen2CompositeHandler`
Gestore dei metadati	`DataLakeGen2MetadataHandler`
Gestore dei record	`DataLakeGen2RecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza Azure Data Lake Storage Gen2 supportata da una funzione Lambda.

Proprietà	Valore
`default`	`datalakegentwo://jdbc:sqlserver://` *`hostname`*`:`*`port`*`;databaseName=;${` *`secret_name`* `}`

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per ADLS Gen2 e Arrow.

ADLS Gen2	Arrow
bit	TINYINT
tinyint	SMALLINT
smallint	SMALLINT
int	INT
bigint	BIGINT
decimal	DECIMAL
numeric	FLOAT8
smallmoney	FLOAT8
money	DECIMAL
float[24]	FLOAT4
float[53]	FLOAT8
real	FLOAT4
datetime	Date(MILLISECOND)
datetime2	Date(MILLISECOND)
smalldatetime	Date(MILLISECOND)
data	Date(DAY)
time	VARCHAR
datetimeoffset	Date(MILLISECOND)

ADLS Gen2	Arrow
char[n]	VARCHAR
varchar[n/max]	VARCHAR

Partizioni e suddivisioni

Azure Data Lake Storage Gen2 utilizza l'archiviazione blob Gen2 compatibile con Hadoop per l'archiviazione dei file di dati. I dati di questi file vengono interrogati dal motore Azure Synapse. Il motore Azure Synapse tratta i dati Gen2 archiviati nei file system come tabelle esterne. Le partizioni vengono implementate in base al tipo di dati. Se i dati sono già stati partizionati e distribuiti all'interno del sistema di archiviazione Gen2, il connettore recupera i dati come singola suddivisione.

Prestazioni

Il connettore Azure Data Lake Storage Gen2 mostra prestazioni più lente quando si eseguono più query contemporaneamente ed è soggetto alla limitazione della larghezza di banda della rete.

Il connettore Athena Azure Data Lake Storage Gen2 esegue il pushdown dei predicati per ridurre la quantità di dati scansionati dalla query. I predicati semplici e le espressioni complesse vengono inviati al connettore per ridurre la quantità di dati scansionati e ridurre il tempo di esecuzione delle query.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Athena Azure Data Lake Storage Gen2 può combinare queste espressioni e inviarle direttamente ad Azure Data Lake Storage Gen2 per funzionalità avanzate e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore Athena Azure Data Lake Storage Gen2 supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, NULL_IF, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%');
```

Interrogazioni pass-through

Il connettore Azure Data Lake Storage Gen2 supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare la query completa all'origine dati per l'esecuzione.

Per usare le query passthrough con Azure Data Lake Storage Gen2, puoi usare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a un'origine dati in Azure Data Lake Storage Gen2. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, vedi il file pom.xml per il connettore Azure Data Lake Storage Gen2 all'indirizzo .com. GitHub

Per ulteriori informazioni su questo connettore, visita il sito corrispondente all'indirizzo .com. GitHub

Connettore Amazon Athena per Azure Synapse

Il connettore Amazon Athena per Azure Synapse Analytics consente ad Amazon Athena di eseguire query SQL sui database Azure Synapse utilizzando JDBC.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .
- Nelle condizioni di filtro, è necessario impostare i tipi di dati `Date` e `Timestamp` sul tipo di dati appropriato.
- Per cercare valori negativi del tipo `Real` e `Float`, utilizza l'operatore <= o >=.
- I tipi di dati `binary`, `varbinary`, `image` e `rowversion` non sono supportati.

Termini

I seguenti termini si riferiscono al connettore Synapse.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.

- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore Synapse.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
synapse://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`SynapseMuxCompositeHandler`
Gestore dei metadati	`SynapseMuxMetadataHandler`
Gestore dei record	`SynapseMuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
$*catalog*_connection_string	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `mysynapsecatalog` , il nome della variabile di ambiente è `mysynapsecatalog_connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *AWS_LAMBDA_FUNCTION_NAME* `}`.

Le seguenti proprietà di esempio si riferiscono a una funzione Lambda Synapse MUX che supporta due istanze del database: `synapse1` (il valore predefinito) e `synapse2`.

Proprietà	Valore
`default`	`synapse://jdbc:synapse://synapse1.hostname:port;databaseName=` *<database_name>* `;${`*secret1_name*`}`
`synapse_catalog1_connection_string`	`synapse://jdbc:synapse://synapse1.hostname:port;databaseName=` *<database_name>* `;${`*secret1_name*`}`
`synapse_catalog2_connection_string`	`synapse://jdbc:synapse://synapse2.hostname:port;databaseName=` *<database_name>* `;${`*secret2_name*`}`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti codificati in AWS Secrets Manager, consulta Move i segreti codificati nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

  ```
  {"username": "${username}", "password": "${password}"}
  ```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto ${secret_name}.

```
synapse://jdbc:synapse://hostname:port;databaseName=<database_name>;${secret_name}
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
synapse://jdbc:synapse://
hostname:port;databaseName=<database_name>;user=<user>;password=<password>
```

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza Synapse.

Tipo di gestore	Classe
Gestore composito	`SynapseCompositeHandler`
Gestore dei metadati	`SynapseMetadataHandler`
Gestore dei record	`SynapseRecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza Synapse supportata da una funzione Lambda.

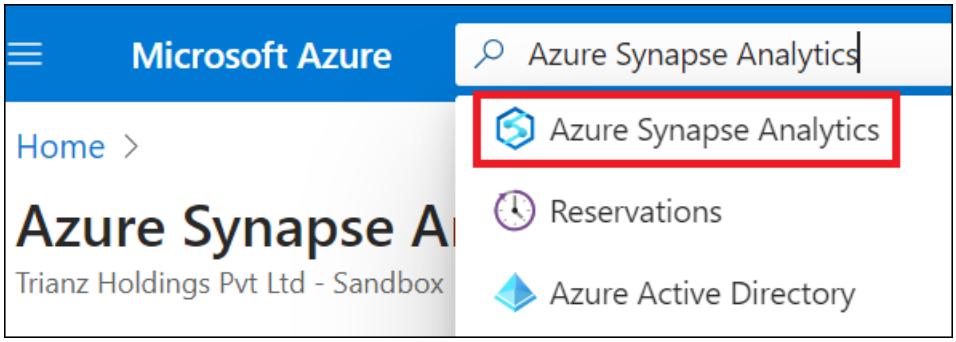
Propr	Valore
`defa`	`synapse://jdbc:sqlserver://hostname:port;databaseName=` *`<database_name>`* `;${`*`secret_name`*`}`

Configurazione dell'autenticazione Active Directory

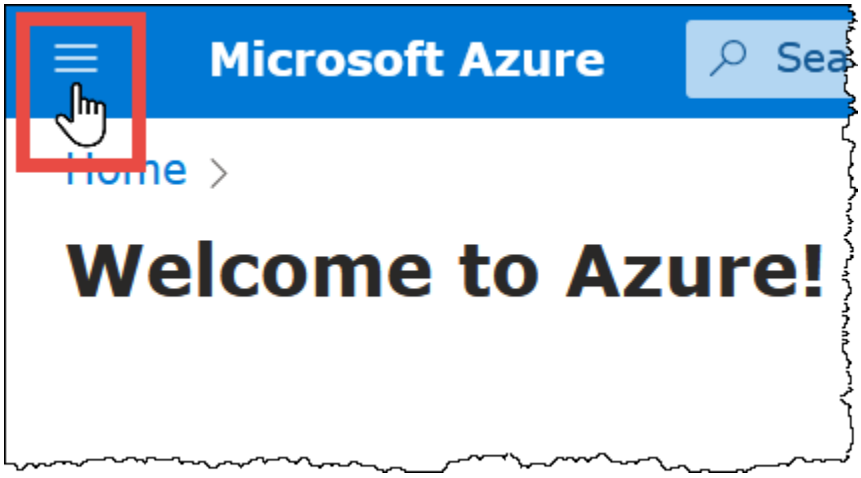
Il connettore Azure Synapse di Amazon Athena supporta l'autenticazione Microsoft Active Directory. Prima di iniziare, è necessario configurare un utente amministrativo nel portale di Microsoft Azure e quindi AWS Secrets Manager utilizzarlo per creare un segreto.

Impostazione dell'utente amministrativo di Active Directory

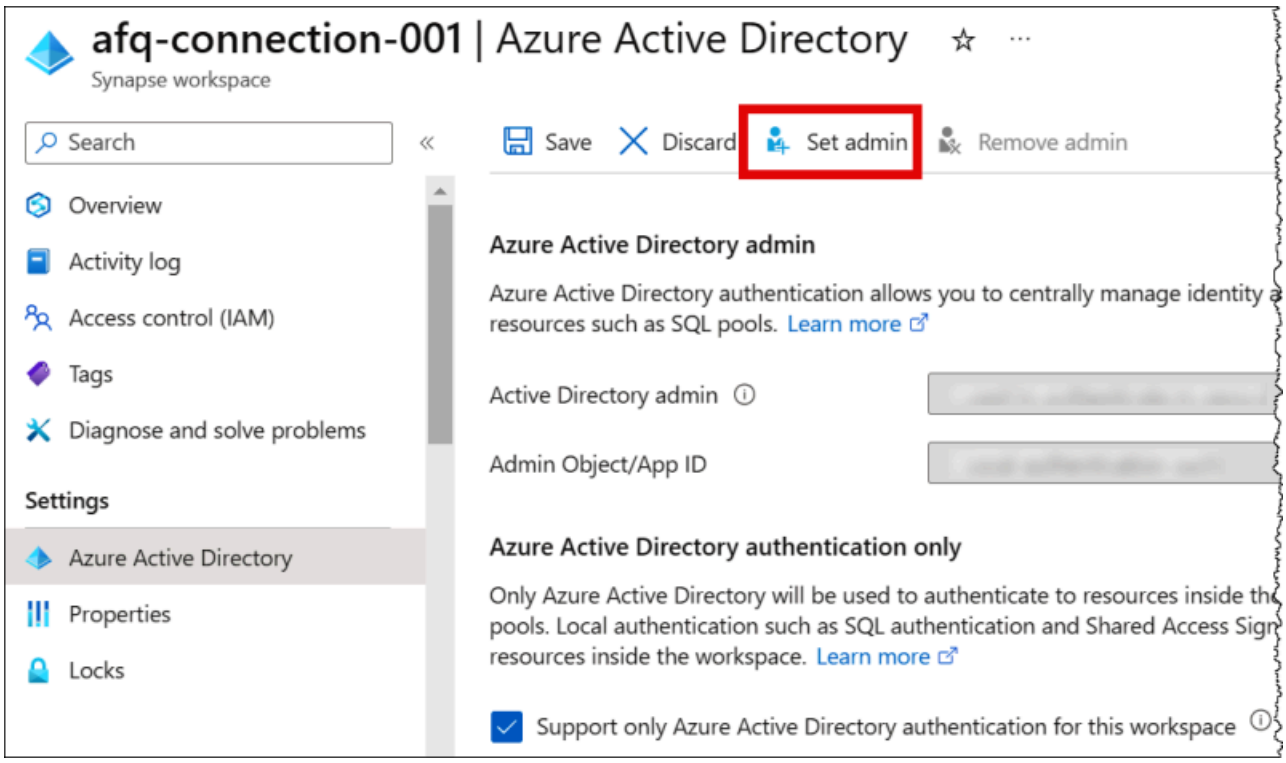
1. Utilizzando un account con privilegi amministrativi, accedi al portale Microsoft Azure all'indirizzo https://portal.azure.com/.
2. Nella casella di ricerca, inserisci Azure Synapse Analytics quindi seleziona Azure Synapse Analytics.



3. Apri il menu a sinistra.



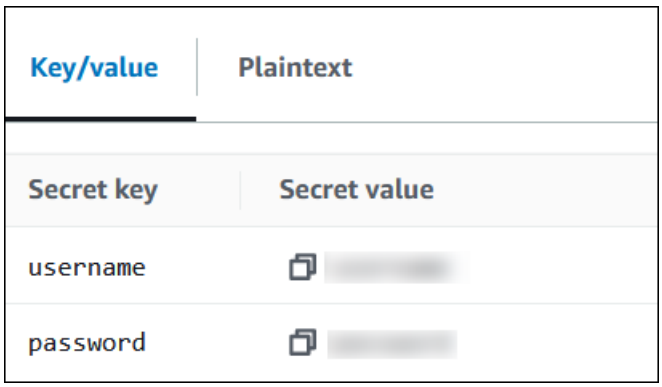
4. Nel riquadro di navigazione, seleziona Azure Active Directory.
5. Nella scheda Imposta admin, imposta Adim di Active Directory su un utente nuovo o esistente.



6. In AWS Secrets Manager, memorizza il nome utente e la password dell'amministratore. Per informazioni sulla creazione di un segreto in Secrets Manager, consulta la pagina Crea un segreto AWS Secrets Manager.

Visualizzazione del segreto in Gestione dei segreti

1. Apri la console di Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Nel riquadro di navigazione, scegli Secrets (Segreti).
3. Nella pagina Secrets (Segreti), scegli il collegamento al tuo segreto.
4. Nella pagina dei dettagli del segreto, scegli Retrieve secret value (Recupera il valore di un segreto).



Modifica della stringa di connessione

Per abilitare l'autenticazione Active Directory del connettore, modifica la stringa di connessione utilizzando la seguente sintassi:

```
synapse://jdbc:synapse://
hostname:port;databaseName=database_name;authentication=ActiveDirectoryPassword;
{secret_name}
```

Usando ActiveDirectoryServicePrincipal

Il connettore Azure Synapse di Amazon Athena supporta anche `ActiveDirectoryServicePrincipal`. Per l'abilitazione, modifica la stringa di connessione come segue.

```
synapse://jdbc:synapse://
hostname:port;databaseName=database_name;authentication=ActiveDirectoryServicePrincipal;
{secret_name}
```

Per `secret_name`, specificare l'ID dell'applicazione o del client come nome utente e il segreto di un'identità principale di servizio nella password.

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.

Parametro	Descrizione
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per Synapse e Apache Arrow.

Synapse	Arrow
bit	TINYINT
tinyint	SMALLINT
smallint	SMALLINT
int	INT
bigint	BIGINT
decimal	DECIMAL
numeric	FLOAT8
smallmoney	FLOAT8
money	DECIMAL
float[24]	FLOAT4
float[53]	FLOAT8
real	FLOAT4

Synapse	Arrow
datetime	Date(MILLISECOND)
datetime2	Date(MILLISECOND)
smalldatetime	Date(MILLISECOND)
data	Date(DAY)
time	VARCHAR
datetimeoffset	Date(MILLISECOND)
char[n]	VARCHAR
varchar[n/max]	VARCHAR
nchar[n]	VARCHAR
nvarchar[n/max]	VARCHAR

Partizioni e suddivisioni

Una partizione è rappresentata da una singola colonna di partizione di tipo `varchar`. Synapse supporta il partizionamento a intervalli, quindi il partizionamento viene implementato estraendo la colonna della partizione e l'intervallo della stessa dalle tabelle dei metadati di Synapse. Questi valori di intervallo vengono utilizzati per creare le suddivisioni.

Prestazioni

La selezione di un sottoinsieme di colonne rallenta notevolmente l'esecuzione delle query. Il connettore presenta una significativa limitazione della larghezza di banda della rete dovuta alla simultaneità.

Il connettore Athena Synapse esegue il pushdown del predicato per ridurre la quantità di dati scansionati dalla query. I predicati semplici e le espressioni complesse vengono inviati al connettore per ridurre la quantità di dati scansionati e ridurre il tempo di esecuzione delle query.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Synapse di Athena può combinare queste espressioni e inviarle direttamente a Synapse per migliorare le funzionalità e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore Synapse di Athena supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, NULL_IF, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%');
```

Interrogazioni pass-through

Il connettore Synapse supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Synapse, è possibile utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a una fonte di dati in Synapse. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

- Per un articolo che mostra come usare Amazon QuickSight e Amazon Athena Federated Query per creare dashboard e visualizzazioni sui dati archiviati nei database Microsoft Azure Synapse, consulta Eseguire analisi multi-cloud usando Amazon, Amazon QuickSight Athena Federated Query e Microsoft Azure Synapse nel Big Data Blog.AWS
- Per le informazioni sulla versione più recente del driver JDBC, consulta il file pom.xml per il connettore Synapse su .com. GitHub
- Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su .com. GitHub

Connettore Amazon Athena per Cloudera Hive

Il connettore Amazon Athena per Cloudera Hive consente ad Athena di eseguire query SQL sulla distribuzione Hadoop Cloudera Hive. Il connettore trasforma le query SQL di Athena nella sintassi HiveQL equivalente.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.
- Prima di utilizzare questo connettore, configura un VPC e un gruppo di sicurezza. Per ulteriori informazioni, consulta Creazione di un VPC per un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.

- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .

Termini

I seguenti termini si riferiscono al connettore Cloudera Hive.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore Cloudera Hive.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
hive://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`HiveMuxCompositeHandler`
Gestore dei metadati	`HiveMuxMetadataHandler`
Gestore dei record	`HiveMuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
`$`*`catalog`*`_connection_string`	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `myhivecatalog` , il nome della variabile di ambiente è `myhivecatalog_connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *`AWS_LAMBDA_FUNCTION_NAME`* `}`.

Le seguenti proprietà di esempio si riferiscono a una funzione Lambda Hive MUX che supporta due istanze del database: `hive1` (il valore predefinito) e `hive2`.

Proprietà	Valore
`default`	`hive://jdbc:hive2://hive1:10000/default?${Test/RDS/hive1}`
`hive2_catalog1_connection_string`	`hive://jdbc:hive2://hive1:10000/default?${Test/RDS/hive1}`
`hive2_catalog2_connection_string`	`hive://jdbc:hive2://hive2:10000/default?UID=sample&PWD=sample`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti codificati in AWS Secrets Manager, consulta Move i segreti codificati nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

```
{"username": "${username}", "password": "${password}"}
```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${Test/RDS/hive1}`.

```
hive://jdbc:hive2://hive1:10000/default?...&${Test/RDS/hive1}&...
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
hive://jdbc:hive2://hive1:10000/default?...&UID=sample2&PWD=sample2&...
```

Attualmente, il connettore Cloudera Hive riconosce le proprietà JDBC `UID` e `PWD`.

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza Cloudera Hive.

Tipo di gestore	Classe
Gestore composito	`HiveCompositeHandler`
Gestore dei metadati	`HiveMetadataHandler`
Gestore dei record	`HiveRecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza Cloudera Hive supportata da una funzione Lambda.

Proprietà	Valore
default	`hive://jdbc:hive2://hive1:10000/default?secret=${Test/RDS/hive1}`

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per JDBC, Cloudera Hive e Arrow.

JDBC	Cloudera Hive	Arrow
Booleano	Booleano	Bit

JDBC	Cloudera Hive	Arrow
Numero intero	TINYINT	Tiny
Breve	SMALLINT	Smallint
Numero intero	INT	Int
Long	BIGINT	Bigint
float	float4	Float4
Doppio	float8	Float8
Data	date	DateDay
Timestamp	timestamp	DateMilli
Stringa	VARCHAR	Varchar
Byte	bytes	Varbinary
BigDecimal	Decimale	Decimale
ARRAY	N/D (vedi nota)	Elenco

Note

Attualmente, Cloudera Hive non supporta i tipi di aggregati `ARRAY`, `MAP`, `STRUCT` e `UNIONTYPE`. Le colonne dei tipi di aggregati vengono trattate come colonne `VARCHAR` in SQL.

Partizioni e suddivisioni

Le partizioni vengono utilizzate per determinare come generare suddivisioni per il connettore. Athena costruisce una colonna sintetica di tipo `varchar` che rappresenta lo schema di partizionamento della tabella per aiutare il connettore a generare suddivisioni. Il connettore non modifica la definizione effettiva della tabella.

Prestazioni

Cloudera Hive supporta le partizioni statiche. Il connettore Cloudera Hive di Athena può recuperare dati da queste partizioni in parallelo. Se desideri interrogare set di dati molto grandi con una distribuzione uniforme delle partizioni, ti consigliamo vivamente il partizionamento statico. Il connettore Cloudera Hive è resiliente alla limitazione della larghezza di banda della rete dovuta alla simultaneità.

Il connettore Athena Cloudera Hive esegue il pushdown dei predicati per ridurre i dati analizzati dalla query. Per ridurre la quantità di dati analizzati e ridurre il tempo di esecuzione delle query le clausole `LIMIT`, i predicati semplici e le espressioni complesse vengono inviate al connettore.

Clausole LIMIT

Una dichiarazione `LIMIT N` riduce la quantità di dati analizzati dalla query. Con il pushdown `LIMIT N`, il connettore restituisce solo le righe `N` ad Athena.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Athena Cloudera Hive può combinare queste espressioni e inviarle direttamente a Cloudera Hive per funzionalità avanzate e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore Athena Cloudera Hive supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
```

```
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%')
LIMIT 10;
```

Interrogazioni pass-through

Il connettore Cloudera Hive supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Cloudera Hive, puoi utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a un'origine dati in Cloudera Hive. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consulta il file pom.xml per il connettore Cloudera Hive su .com. GitHub

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su .com. GitHub

Connettore Amazon Athena per Cloudera Impala

Il connettore Amazon Athena Cloudera Impala consente ad Athena di eseguire query SQL sulla distribuzione Cloudera Impala. Il connettore trasforma le query SQL di Athena nella sintassi Impala equivalente.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.
- Prima di utilizzare questo connettore, configura un VPC e un gruppo di sicurezza. Per ulteriori informazioni, consulta Creazione di un VPC per un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .

Termini

I seguenti termini si riferiscono al connettore Cloudera Impala.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`

- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore Cloudera Impala.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un cluster Impala.

```
impala://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`ImpalaMuxCompositeHandler`
Gestore dei metadati	`ImpalaMuxMetadataHandler`
Gestore dei record	`ImpalaMuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
`$catalog_connection_string`	Obbligatorio. Una stringa di connessione al cluster Impala per un catalogo Athena. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `myimpalacatalog` , il nome della variabile di ambiente è `myimpalacatalog_connection_string` .

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *`AWS_LAMBDA_FUNCTION_NAME`* `}`.

Le seguenti proprietà di esempio si riferiscono a una funzione Lambda Impala MUX che supporta due istanze del database: `impala1` (il valore predefinito) e `impala2`.

Proprietà	Valore
`default`	`impala://jdbc:impala://some.impala.host.name:21050/?${Test/impala1}`
`impala_catalog1_connection_string`	`impala://jdbc:impala://someother.impala.host.name:21050/?${Test/impala1}`
`impala_catalog2_connection_string`	`impala://jdbc:impala://another.impala.host.name:21050/?UID=sample&PWD=sample`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

> **Important**
>
> Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti codificati in AWS Secrets Manager, consulta Move i segreti codificati nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

  ```
  {"username": "${username}", "password": "${password}"}
  ```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${Test/impala1host}`.

```
impala://jdbc:impala://Impala1host:21050/?...&${Test/impala1host}&...
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
impala://jdbc:impala://Impala1host:21050/?...&UID=sample2&PWD=sample2&...
```

Attualmente, Cloudera Impala riconosce le proprietà JDBC `UID` e `PWD`.

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza Cloudera Impala.

Tipo di gestore	Classe
Gestore composito	`ImpalaCompositeHandler`
Gestore dei metadati	`ImpalaMetadataHandler`

Tipo di gestore	Classe
Gestore dei record	`ImpalaRecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza Cloudera Impala supportata da una funzione Lambda.

Proprietà	Valore
`default`	`impala://jdbc:impala://Impala1host:21050/?secret=${Test/impala1host}`

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-s`

Parametro	Descrizione
	`erver-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per JDBC, Cloudera Impala e Arrow.

JDBC	Cloudera Impala	Arrow
Booleano	Booleano	Bit
Numero intero	TINYINT	Tiny
Breve	SMALLINT	Smallint
Numero intero	INT	Int
Long	BIGINT	Bigint
float	float4	Float4
Doppio	float8	Float8
Data	date	DateDay
Timestamp	timestamp	DateMilli
Stringa	VARCHAR	Varchar
Byte	bytes	Varbinary
BigDecimal	Decimale	Decimale
ARRAY	N/D (vedi nota)	Elenco

> **Note**
>
> Attualmente, Cloudera Impala non supporta i tipi di aggregati `ARRAY`, `MAP`, `STRUCT` e `UNIONTYPE`. Le colonne dei tipi di aggregati vengono trattate come colonne `VARCHAR` in SQL.

Partizioni e suddivisioni

Le partizioni vengono utilizzate per determinare come generare suddivisioni per il connettore. Athena costruisce una colonna sintetica di tipo `varchar` che rappresenta lo schema di partizionamento della tabella per aiutare il connettore a generare suddivisioni. Il connettore non modifica la definizione effettiva della tabella.

Prestazioni

Cloudera Impala supporta le partizioni statiche. Il connettore Cloudera Impala di Athena può recuperare dati da queste partizioni in parallelo. Se desideri interrogare set di dati molto grandi con una distribuzione uniforme delle partizioni, ti consigliamo vivamente il partizionamento statico. Il connettore Cloudera Impala è resiliente alla limitazione della larghezza di banda della rete dovuta alla simultaneità.

Il connettore Athena Cloudera Impala esegue il pushdown dei predicati per ridurre i dati scansionati dalla query. Le clausole `LIMIT`, i predicati semplici e le espressioni complesse vengono inviati al connettore per ridurre la quantità di dati analizzati e il tempo di esecuzione delle query.

Clausole LIMIT

Una dichiarazione `LIMIT N` riduce la quantità di dati analizzati dalla query. Con il pushdown `LIMIT N`, il connettore restituisce solo le righe `N` ad Athena.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Athena Cloudera Impala può combinare queste espressioni e inviarle direttamente a Cloudera Impala per funzionalità avanzate e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore Athena Cloudera Impala supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_DISTINCT_FROM, NULL_IF, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%')
LIMIT 10;
```

Interrogazioni pass-through

Il connettore Cloudera Impala supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Cloudera Impala, puoi utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a un'origine dati in Cloudera Impala. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consulta il file pom.xml per il connettore Cloudera Impala su .com. GitHub

Per ulteriori informazioni su questo connettore, visita il sito corrispondente su .com. GitHub

Connettore Amazon Athena CloudWatch

Il CloudWatch connettore Amazon Athena consente la comunicazione CloudWatch con Amazon Athena in modo da poter interrogare i dati di log con SQL.

Il connettore mappa i tuoi schemi LogGroups AS e ciascuno LogStream come tabella. Il connettore mappa anche una `all_log_streams` vista speciale che contiene tutto LogStreams in. LogGroup Questa visualizzazione consente di interrogare tutti i log in una sola LogGroup volta invece di cercarli LogStream singolarmente.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Parametri

Usa le variabili di ambiente Lambda in questa sezione per configurare il CloudWatch connettore.

- spill_bucket: specifica il bucket Amazon S3 per i dati che superano i limiti della funzione Lambda.
- spill_prefix: (facoltativo) per impostazione predefinita, viene utilizzata una sottocartella nello `spill_bucket` specificato chiamata `athena-federation-spill`. Ti consigliamo di configurare un ciclo di vita dell'archiviazione di Amazon S3 in questa posizione per eliminare gli spill più vecchi di un numero predeterminato di giorni o ore.
- spill_put_request_headers: (facoltativo) una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-`

`amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

- kms_key_id: (facoltativo) per impostazione predefinita, tutti i dati riversati in Amazon S3 vengono crittografati utilizzando la modalità di crittografia autenticata AES-GCM e una chiave generata casualmente. Per fare in modo che la tua funzione Lambda utilizzi chiavi di crittografia più potenti generate da KMS come `a7e63k4b-8loc-40db-a2a1-4d0en2cd8331`, puoi specificare l'ID della chiave KMS.
- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.

Il connettore supporta anche il controllo della congestione AIMD per la gestione degli eventi di limitazione tramite il costrutto CloudWatch Amazon Athena Query Federation SDK. `ThrottlingInvoker` Puoi modificare il comportamento di limitazione predefinito impostando una delle seguenti variabili di ambiente facoltative:

- throttle_initial_delay_ms: il ritardo iniziale della chiamata applicato dopo il primo evento di congestione. Il valore predefinito è 10 millisecondi.
- throttle_max_delay_ms: il ritardo massimo tra le chiamate. Puoi derivare il TPS dividendolo per 1.000 ms. Il valore predefinito è 1000 millisecondi.
- throttle_decrease_factor: il fattore in base al quale Athena riduce la frequenza delle chiamate. Il valore predefinito è 0.5
- throttle_increase_ms: la velocità con cui Athena riduce il ritardo della chiamata. Il valore predefinito è 10 millisecondi.

Database e tabelle

Il CloudWatch connettore Athena mappa i tuoi schemi LogGroups AS (ovvero database) e ciascuno LogStream come tabella. Il connettore mappa anche una `all_log_streams` vista speciale che contiene tutto LogStreams in. LogGroup Questa visualizzazione consente di interrogare tutti i log in una sola LogGroup volta invece di cercarli LogStream singolarmente.

Ogni tabella mappata dal connettore CloudWatch Athena ha lo schema seguente. Questo schema corrisponde ai campi forniti da CloudWatch Logs.

- log_stream — Un VARCHAR che contiene il nome da LogStream cui proviene la riga.
- time: un INT64 che contiene l'ora epoch in cui è stata generata la riga del log.
- message: un VARCHAR che contiene il messaggio di log.

Esempi

L'esempio seguente mostra come eseguire un'SELECTinterrogazione su un oggetto specificato. LogStream

```
SELECT *
FROM "lambda:cloudwatch_connector_lambda_name"."log_group_path"."log_stream_name"
LIMIT 100
```

L'esempio seguente mostra come utilizzare la `all_log_streams` vista per eseguire un'interrogazione su tutti i dati LogStreams in uno specifico LogGroup.

```
SELECT *
FROM "lambda:cloudwatch_connector_lambda_name"."log_group_path"."all_log_streams"
LIMIT 100
```

Autorizzazioni richieste

Consulta la sezione `Policies` del file athena-cloudwatch.yaml per i dettagli completi delle policy IAM richieste da questo connettore. L'elenco che segue riporta un riepilogo delle autorizzazioni richieste.

- Accesso in scrittura ad Amazon S3: per trasferire i risultati di query di grandi dimensioni, il connettore richiede l'accesso in scrittura a una posizione in Amazon S3.
- Athena GetQueryExecution: il connettore utilizza questa autorizzazione per fallire rapidamente quando la query Athena upstream è terminata.
- CloudWatch Lettura/scrittura dei registri: il connettore utilizza questa autorizzazione per leggere i dati di registro e scrivere i registri di diagnostica.

Prestazioni

Il CloudWatch connettore Athena tenta di ottimizzare le query CloudWatch parallelizzando le scansioni dei flussi di log necessari per la query. Per determinati filtri temporali, il pushdown dei predicati viene eseguito sia all'interno della funzione Lambda che all'interno di Logs. CloudWatch

Per prestazioni ottimali, utilizza solo lettere minuscole per i nomi di gruppi di log e dei flussi di log. L'utilizzo di caratteri misti tra maiuscole e minuscole fa sì che il connettore esegua una ricerca senza distinzione tra maiuscole e minuscole, più impegnativa dal punto di vista computazionale.

Interrogazioni pass-through

Il CloudWatch connettore supporta le query passthrough che utilizzano la sintassi delle query di CloudWatch Logs Insights. Per ulteriori informazioni su CloudWatch Logs Insights, consulta Analyzing log data with CloudWatch Logs Insights nella Amazon CloudWatch Logs User Guide.

Per creare query passthrough con CloudWatch, utilizza la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            STARTTIME => 'start_time',
            ENDTIME => 'end_time',
            QUERYSTRING => 'query_string',
            LOGGROUPNAMES => 'log_group-names',
            LIMIT => 'max_number_of_results'
        ))
```

Il seguente esempio di query CloudWatch passthrough filtra per il `duration` campo quando questo non è uguale a 1000.

```
SELECT * FROM TABLE(
        system.query(
            STARTTIME => '1710918615308',
            ENDTIME => '1710918615972',
            QUERYSTRING => 'fields @duration | filter @duration != 1000',
            LOGGROUPNAMES => '/aws/lambda/cloudwatch-test-1',
            LIMIT => '2'
            ))
```

Informazioni sulla licenza

Il progetto Amazon Athena CloudWatch Connector è concesso in licenza con licenza Apache-2.0.

Risorse aggiuntive

Per ulteriori informazioni su questo connettore, visita il sito corrispondente su .com. GitHub

Connettore Amazon Athena Metrics CloudWatch

Il connettore Amazon Athena CloudWatch Metrics consente CloudWatch ad Amazon Athena di interrogare i dati di Metrics con SQL.

Per informazioni sulla pubblicazione delle metriche delle query CloudWatch da Athena stessa, consulta. Controllo dei costi e monitoraggio delle interrogazioni con metriche ed eventi CloudWatch

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Parametri

Usa le variabili di ambiente Lambda in questa sezione per configurare il connettore CloudWatch Metrics.

- spill_bucket: specifica il bucket Amazon S3 per i dati che superano i limiti della funzione Lambda.
- spill_prefix: (facoltativo) per impostazione predefinita, viene utilizzata una sottocartella nello `spill_bucket` specificato chiamata `athena-federation-spill`. Ti consigliamo di configurare un ciclo di vita dell'archiviazione di Amazon S3 in questa posizione per eliminare gli spill più vecchi di un numero predeterminato di giorni o ore.
- spill_put_request_headers: (facoltativo) una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.
- kms_key_id: (facoltativo) per impostazione predefinita, tutti i dati riversati in Amazon S3 vengono crittografati utilizzando la modalità di crittografia autenticata AES-GCM e una chiave generata casualmente. Per fare in modo che la tua funzione Lambda utilizzi chiavi di crittografia più potenti generate da KMS come `a7e63k4b-8loc-40db-a2a1-4d0en2cd8331`, puoi specificare l'ID della chiave KMS.
- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.

Il connettore supporta anche il controllo della congestione AIMD per la gestione degli eventi di limitazione tramite il costrutto CloudWatch Amazon Athena Query Federation SDK. `ThrottlingInvoker` Puoi modificare il comportamento di limitazione predefinito impostando una delle seguenti variabili di ambiente facoltative:

- throttle_initial_delay_ms: il ritardo iniziale della chiamata applicato dopo il primo evento di congestione. Il valore predefinito è 10 millisecondi.
- throttle_max_delay_ms: il ritardo massimo tra le chiamate. Puoi derivare il TPS dividendolo per 1.000 ms. Il valore predefinito è 1000 millisecondi.
- throttle_decrease_factor: il fattore in base al quale Athena riduce la frequenza delle chiamate. Il valore predefinito è 0.5
- throttle_increase_ms: la velocità con cui Athena riduce il ritardo della chiamata. Il valore predefinito è 10 millisecondi.

Database e tabelle

Il connettore Athena CloudWatch Metrics mappa i namespace, le dimensioni, le metriche e i valori delle metriche in due tabelle in un unico schema chiamato. `default`

La tabella metrics (parametri)

La tabella `metrics` contiene i parametri disponibili definiti in modo univoco da una combinazione di spazio del nome, set e nome. La tabella `metrics` contiene le colonne seguenti.

- namespace: un `VARCHAR` contenente lo spazio del nome.
- metric_name: un `VARCHAR` contenente il nome del parametro.
- dimensions: un `LIST` di oggetti `STRUCT` composti da `dim_name (VARCHAR)` e `dim_value (VARCHAR)`.
- statistic: un `LIST` di statistiche `VARCH` (ad esempio, `p90`, `AVERAGE`...) disponibili per il parametro.

La tabella metric_samples (campioni dei parametri)

La tabella `metric_samples` contiene i campioni del parametro disponibili per ciascun parametro all'interno della tabella `metrics`. La tabella `metric_samples` contiene le colonne seguenti.

- namespace: un `VARCHAR` contenente lo spazio del nome.
- metric_name: un `VARCHAR` contenente il nome del parametro.

- dimensions: un `LIST` di oggetti `STRUCT` composti da `dim_name (VARCHAR)` e `dim_value (VARCHAR)`.
- dim_name: un campo di cortesia `VARCHAR` che puoi utilizzare per filtrare facilmente in base al nome di una singola dimensione.
- dim_value: un campo di cortesia `VARCHAR` che puoi utilizzare per filtrare facilmente in base al valore di una singola dimensione.
- period: un campo `INT` che rappresenta il "periodo" del parametro in secondi (ad esempio, il parametro può avere un valore di 60 secondi).
- timestamp: un campo `BIGINT` che rappresenta l'ora epoch, espressa in secondi, alla quale il campione del parametro fa riferimento.
- value: un campo `FLOAT8` che contiene il valore del campione.
- statistic: un `VARCHAR` che contiene il tipo di statistica del campione (ad esempio, `AVERAGE` o `p90`).

Autorizzazioni richieste

Per tutti i dettagli sulle politiche IAM richieste da questo connettore, consulta la sezione del file.yaml. `Policies` athena-cloudwatch-metrics L'elenco che segue riporta un riepilogo delle autorizzazioni richieste.

- Accesso in scrittura ad Amazon S3: per trasferire i risultati di query di grandi dimensioni, il connettore richiede l'accesso in scrittura a una posizione in Amazon S3.
- Athena GetQueryExecution: il connettore utilizza questa autorizzazione per fallire rapidamente quando la query Athena upstream è terminata.
- CloudWatch Metriche ReadOnly: il connettore utilizza questa autorizzazione per interrogare i dati delle metriche.
- CloudWatch Scrittura dei registri: il connettore utilizza questo accesso per scrivere i registri di diagnostica.

Prestazioni

Il connettore Athena CloudWatch Metrics tenta di ottimizzare le query rispetto a CloudWatch Metrics parallelizzando le scansioni dei flussi di log necessari per la query. Per determinati filtri temporali, metrici, namespace e dimensioni, il pushdown dei predicati viene eseguito sia all'interno della funzione Lambda che all'interno dei log. CloudWatch

Informazioni sulla licenza

Il progetto Amazon Athena CloudWatch Metrics connector è concesso in licenza con la licenza Apache-2.0.

Risorse aggiuntive

Per ulteriori informazioni su questo connettore, visita il sito corrispondente su .com. GitHub

Connettore Amazon Athena CMDB AWS

Il connettore Amazon Athena AWS CMDB consente ad Athena di comunicare con vari AWS servizi in modo da poterli interrogare con SQL.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Parametri

Utilizza le variabili di ambiente Lambda in questa sezione per configurare il connettore AWS CMDB.

- spill_bucket: specifica il bucket Amazon S3 per i dati che superano i limiti della funzione Lambda.
- spill_prefix: (facoltativo) per impostazione predefinita, viene utilizzata una sottocartella nello `spill_bucket` specificato chiamata `athena-federation-spill`. Ti consigliamo di configurare un ciclo di vita dell'archiviazione di Amazon S3 in questa posizione per eliminare gli spill più vecchi di un numero predeterminato di giorni o ore.
- spill_put_request_headers: (facoltativo) una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.
- kms_key_id: (facoltativo) per impostazione predefinita, tutti i dati riversati in Amazon S3 vengono crittografati utilizzando la modalità di crittografia autenticata AES-GCM e una chiave generata casualmente. Per fare in modo che la tua funzione Lambda utilizzi chiavi di crittografia più potenti generate da KMS come `a7e63k4b-8loc-40db-a2a1-4d0en2cd8331`, puoi specificare l'ID della chiave KMS.

- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.
- default_ec2_image_owner: (facoltativo) una volta impostato, controlla il proprietario predefinito dell'immagine Amazon EC2 che filtra le Amazon Machine Image (AMI). Se non imposti questo valore e la tua query sulla tabella delle immagini EC2 non include un filtro per il proprietario, i risultati includeranno tutte le immagini pubbliche.

Database e tabelle

Il connettore Athena AWS CMDB rende disponibili i seguenti database e tabelle per interrogare l'inventario delle risorse. AWS Per ulteriori informazioni sulle colonne disponibili in ogni tabella, esegui un'istruzione `DESCRIBE` *`database`*`.`*`table`* utilizzando la console o l'API Athena.

- ec2: questo database contiene risorse correlate ad Amazon EC2, tra cui le seguenti.

- ebs_volumes: contiene i dettagli dei tuoi volumi Amazon EBS.
- ec2_instances: contiene i dettagli delle tue istanze EC2.
- ec2_images: contiene i dettagli delle immagini delle tue istanze EC2.
- routing_tables: contiene i dettagli delle tue tabelle di instradamento del VPC.
- security_groups: contiene i dettagli dei tuoi gruppi di sicurezza.
- subnets: contiene i dettagli delle tue sottoreti VPC.
- vpcs: contiene i dettagli dei tuoi VPC.

- emr: questo database contiene risorse correlate ad Amazon EMR, tra cui le seguenti.

- emr_clusters: contiene i dettagli dei tuoi cluster EMR.

- rds: questo database contiene risorse correlate ad Amazon RDS, tra cui le seguenti.

- rds_instances: contiene i dettagli delle tue istanze RDS.

- s3: questo database contiene risorse correlate a RDS, tra cui le seguenti.

- bucket: contiene i dettagli dei tuoi bucket Amazon S3.
- objects: contiene i dettagli dei tuoi oggetti Amazon S3, escluso il loro contenuto.

Autorizzazioni richieste

Per tutti i dettagli sulle politiche IAM richieste da questo connettore, consulta la `Policies` sezione del athena-aws-cmdb file.yaml. L'elenco che segue riporta un riepilogo delle autorizzazioni richieste.

- Accesso in scrittura ad Amazon S3: per trasferire i risultati di query di grandi dimensioni, il connettore richiede l'accesso in scrittura a una posizione in Amazon S3.
- Athena GetQueryExecution: il connettore utilizza questa autorizzazione per fallire rapidamente quando la query Athena upstream è terminata.
- S3 List: il connettore utilizza questa autorizzazione per elencare bucket e oggetti di Amazon S3.
- EC2 Describe: il connettore utilizza questa autorizzazione per descrivere risorse come le istanze Amazon EC2, i gruppi di sicurezza, i VPC e i volumi Amazon EBS.
- EMR Describe / List: il connettore utilizza questa autorizzazione per descrivere i cluster EMR.
- RDS Describe: il connettore utilizza questa autorizzazione per descrivere le istanze RDS.

Prestazioni

Attualmente, il connettore Athena AWS CMDB non supporta le scansioni parallele. Il pushdown dei predicati viene eseguito all'interno della funzione Lambda. Ove possibile, i predicati parziali vengono inviati ai servizi interrogati. Ad esempio, una query per i dettagli di una specifica istanza Amazon EC2 chiama l'API EC2 con l'ID di istanza specifico per eseguire un'operazione descrittiva mirata.

Informazioni sulla licenza

Il progetto del connettore Amazon Athena AWS CMDB è concesso in licenza con licenza Apache-2.0.

Risorse aggiuntive

Per ulteriori informazioni su questo connettore, visita il sito corrispondente su .com. GitHub

Connettore IBM Db2 di Amazon Athena

Il connettore Amazon Athena per Db2 consente ad Amazon Athena di eseguire query SQL sui database IBM Db2 utilizzando JDBC.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.
- Prima di utilizzare questo connettore, configura un VPC e un gruppo di sicurezza. Per ulteriori informazioni, consulta Creazione di un VPC per un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .
- Nelle condizioni di filtro, è necessario impostare i tipi di dati date e timestamp sul tipo di dati appropriato.

Termini

I seguenti termini si riferiscono al connettore Db2.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.

- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore Db2.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
dbtwo://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`Db2MuxCompositeHandler`
Gestore dei metadati	`Db2MuxMetadataHandler`
Gestore dei record	`Db2MuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
$*catalog*_connection_string	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `mydbtwocatalog` , il nome della variabile di ambiente è `mydbtwocatalog_connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *AWS_LAMBDA_FUNCTION_NAME* `}`.

Le seguenti proprietà di esempio si riferiscono a una funzione Lambda Db2 MUX che supporta due istanze del database: `dbtwo1` (il valore predefinito) e `dbtwo2`.

Proprietà	Valore
`default`	`dbtwo://jdbc:db2://dbtwo1.hostname:port/` *database_name* `:${`*secret1_name*`}`
`dbtwo_catalog1_connection_string`	`dbtwo://jdbc:db2://dbtwo1. hostname:port/` *database_name* `:${`*secret1_name*`}`
`dbtwo_catalog2_connection_string`	`dbtwo://jdbc:db2://dbtwo2. hostname:port/` *database_name* `:${`*secret2_name*`}`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

> **Important**
>
> Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti codificati in AWS Secrets Manager, consulta Move i segreti codificati nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

  ```
  {"username": "${username}", "password": "${password}"}
  ```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${secret_name}`.

```
dbtwo://jdbc:db2://hostname:port/database_name:${secret_name}
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
dbtwo://jdbc:db2://hostname:port/database_name:user=user_name;password=password;
```

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza Db2.

Tipo di gestore	Classe
Gestore composito	`Db2CompositeHandler`
Gestore dei metadati	`Db2MetadataHandler`
Gestore dei record	`Db2RecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza Db2 supportata da una funzione Lambda.

Proprietà	Valore
`default`	dbtwo://jdbc:db2://hostname:port/*`database_name`* :${secret_name}

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.

Parametro	Descrizione
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per JDBC e Arrow.

Db2	Arrow
CHAR	VARCHAR
VARCHAR	VARCHAR
DATE	DATEDAY
TIME	VARCHAR
TIMESTAMP	DATEMILLI
DATETIME	DATEMILLI
BOOLEAN	BOOL
SMALLINT	SMALLINT
INTEGER	INT
BIGINT	BIGINT
DECIMAL	DECIMAL
REAL	FLOAT8
DOUBLE	FLOAT8

Db2	Arrow
DECFLOAT	FLOAT8

Partizioni e suddivisioni

Una partizione è rappresentata da una o più colonne di partizione di tipo `varchar`. Il connettore Db2 crea partizioni utilizzando i seguenti schemi organizzativi.

- Distribuzione per hash
- Partizione per intervallo
- Organizzazione per dimensioni

Il connettore recupera i dettagli delle partizioni come il numero di partizioni e il nome della colonna da una o più tabelle di metadati Db2. Le suddivisioni vengono create in base al numero di partizioni distinte ricevute.

Prestazioni

Il connettore Athena Db2 esegue il pushdown dei predicati per ridurre i dati analizzati dalla query. Le clausole `LIMIT`, i predicati semplici e le espressioni complesse vengono inviati al connettore per ridurre la quantità di dati analizzati e per ridurre il tempo di esecuzione delle query.

Clausole LIMIT

Una dichiarazione `LIMIT N` riduce la quantità di dati analizzati dalla query. Con il pushdown `LIMIT N`, il connettore restituisce solo le righe `N` ad Athena.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Athena Db2 può combinare queste espressioni e inviarle direttamente a Db2 per migliorare le funzionalità e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore Athena Db2 supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT

- Ugualglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_DISTINCT_FROM, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%')
LIMIT 10;
```

Interrogazioni pass-through

Il connettore Db2 supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Db2, è possibile utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a una fonte di dati in Db2. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consultate il file pom.xml per il connettore Db2 su .com. GitHub

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su .com. GitHub

Connettore Amazon Athena IBM Db2 AS/400 (Db2 iSeries)

Il connettore Amazon Athena per Db2 AS/400 consente ad Amazon Athena di eseguire query SQL sui database IBM Db2 AS/400 (Db2 iSeries) utilizzando JDBC.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.
- Prima di utilizzare questo connettore, configura un VPC e un gruppo di sicurezza. Per ulteriori informazioni, consulta Creazione di un VPC per un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .
- Nelle condizioni di filtro, è necessario impostare i tipi di dati date e timestamp sul tipo di dati appropriato.

Termini

I seguenti termini si riferiscono al connettore Db2 AS/400.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Usa le variabili di ambiente Lambda in questa sezione per configurare il connettore Db2 AS/400.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
db2as400://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`Db2MuxCompositeHandler`
Gestore dei metadati	`Db2MuxMetadataHandler`
Gestore dei record	`Db2MuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
`$`*`catalog`*`_connection_string`	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `mydb2as400catalog` , il nome della variabile di ambiente è `mydb2as400catalog_connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *`AWS_LAMBDA_FUNCTION_NAME`* `}`.

Le seguenti proprietà di esempio si riferiscono a una funzione Lambda Db2 MUX che supporta due istanze del database: `db2as4001` (il valore predefinito) e `db2as4002`.

Proprietà	Valore
`default`	`db2as400://jdbc:as400://` *`<ip_address>`* `;`*`<properties>`* `;:${`*`<secret name>`*`};`
`db2as400_catalog1_connection_string`	`db2as400://jdbc:as400://db2as4001.hostname/:${` *`secret1_name`* `}`
`db2as400_catalog2_connection_string`	`db2as400://jdbc:as400://db2as4002.hostname/:${` *`secret2_name`* `}`

Proprietà	Valore
`db2as400_catalog3_connection_string`	`db2as400://jdbc:as400://` *`<ip_address>`* `;user=`*`<username>`* `;password=` *`<password>`* `;`*`<properties>`* `;`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti codificati in AWS Secrets Manager, consulta Move i segreti codificati nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

  ```
  {"username": "${username}", "password": "${password}"}
  ```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${secret_name}`.

```
db2as400://jdbc:as400://<ip_address>;<properties>;:${<secret_name>};
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
db2as400://jdbc:as400://<ip_address>;user=<username>;password=<password>;<properties>;
```

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti metadati e gestori di record a connessione singola per connetterti a una singola istanza Db2 AS/400.

Tipo di gestore	Classe
Gestore composito	`Db2CompositeHandler`
Gestore dei metadati	`Db2MetadataHandler`
Gestore dei record	`Db2RecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza Db2 AS/400 supportata da una funzione Lambda.

Proprietà	Valore
default	db2as400://jdbc:as400:// *<ip_address>* ;*<properties>* ;: ${*<secret_name>* };

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
spill_bucket	Obbligatorio. Nome del bucket di spill.
spill_prefix	Obbligatorio. Prefisso della chiave del bucket di spill.
spill_put_request_ headers	(Facoltativo) Una mappa codificata in JSON delle intestazi oni e dei valori della richiesta per la richiesta putObject di Amazon S3 utilizzata per lo spill (ad esempio, {"x-amz-s erver-side-encryption" : "AES256"}). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per JDBC e Apache Arrow.

Db2 AS/400	Arrow
CHAR	VARCHAR
VARCHAR	VARCHAR
DATE	DATEDAY
TIME	VARCHAR

Db2 AS/400	Arrow
TIMESTAMF	DATEMILLI
DATETIME	DATEMILLI
BOOLEAN	BOOL
SMALLINT	SMALLINT
INTEGER	INT
BIGINT	BIGINT
DECIMAL	DECIMAL
REAL	FLOAT8
DOUBLE	FLOAT8
DECFLOAT	FLOAT8

Partizioni e suddivisioni

Una partizione è rappresentata da una o più colonne di partizione di tipo `varchar`. Il connettore Db2 AS/400 crea partizioni utilizzando i seguenti schemi di organizzazione.

- Distribuzione per hash
- Partizione per intervallo
- Organizzazione per dimensioni

Il connettore recupera i dettagli delle partizioni come il numero di partizioni e il nome delle colonne da una o più tabelle di metadati Db2 AS/400. Le suddivisioni vengono create in base al numero di partizioni distinte ricevute.

Prestazioni

Per migliorare le prestazioni, utilizzate il predicate pushdown per eseguire query da Athena, come illustrato negli esempi seguenti.

```
SELECT * FROM "lambda:<LAMBDA_NAME>"."<SCHEMA_NAME>"."<TABLE_NAME>"
 WHERE integercol = 2147483647
```

```
SELECT * FROM "lambda: <LAMBDA_NAME>"."<SCHEMA_NAME>"."<TABLE_NAME>"
 WHERE timestampcol >= TIMESTAMP '2018-03-25 07:30:58.878'
```

Interrogazioni pass-through

Il connettore Db2 AS/400 supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Db2 AS/400, è possibile utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a una fonte di dati in Db2 AS/400. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consultate il file pom.xml per il connettore Db2 AS/400 su .com. GitHub

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su .com. GitHub

Connettore Amazon Athena DocumentDB

Il connettore DocumentDB per Amazon Athena consente ad Amazon Athena di comunicare con le istanze DocumentDB in modo da poter eseguire query sui dati DocumentDB con SQL. Il connettore funziona anche con qualsiasi endpoint compatibile con MongoDB.

A differenza dei tradizionali archivi di dati relazionali, le raccolte Amazon DocumentDB non hanno uno schema prestabilito. DocumentDB non dispone di un archivio dei metadati. Ogni voce in una raccolta DocumentDB può avere campi e tipi di dati diversi.

Il connettore DocumentDB supporta due meccanismi per la generazione di informazioni sullo schema della tabella: inferenza dello schema di base e metadati. AWS Glue Data Catalog

L'inferenza dello schema è l'impostazione predefinita. Questa opzione esegue la scansione di un numero limitato di documenti della raccolta, forma un'unione di tutti i campi e forza i campi che hanno tipi di dati non sovrapposti. Questa opzione funziona bene per le raccolte che hanno voci per lo più uniformi.

Per le raccolte con una maggiore varietà di tipi di dati, il connettore supporta il recupero dei metadati da AWS Glue Data Catalog. Se il connettore vede un AWS Glue database e una tabella che corrispondono al database DocumentDB e ai nomi delle raccolte, ottiene le informazioni sullo schema dalla tabella corrispondente AWS Glue . Quando crei la AWS Glue tabella, ti consigliamo di renderla un superset di tutti i campi a cui potresti voler accedere dalla tua raccolta DocumentDB.

Se hai abilitato Lake Formation nel tuo account, il ruolo IAM per il tuo connettore Lambda federato Athena che hai distribuito nell'accesso in lettura deve avere accesso in lettura in AWS Serverless Application Repository Lake Formation a. AWS Glue Data Catalog

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore DocumentDB.

- spill_bucket: specifica il bucket Amazon S3 per i dati che superano i limiti della funzione Lambda.
- spill_prefix: (facoltativo) per impostazione predefinita, viene utilizzata una sottocartella nello `spill_bucket` specificato chiamata `athena-federation-spill`. Ti consigliamo di configurare un ciclo di vita dell'archiviazione di Amazon S3 in questa posizione per eliminare gli spill più vecchi di un numero predeterminato di giorni o ore.
- spill_put_request_headers: (facoltativo) una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.
- kms_key_id: (facoltativo) per impostazione predefinita, tutti i dati riversati in Amazon S3 vengono crittografati utilizzando la modalità di crittografia autenticata AES-GCM e una chiave generata casualmente. Per fare in modo che la tua funzione Lambda utilizzi chiavi di crittografia più potenti generate da KMS come `a7e63k4b-8loc-40db-a2a1-4d0en2cd8331`, puoi specificare l'ID della chiave KMS.
- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.
- disable_glue — (Facoltativo) Se presente e impostato su true, il connettore non tenta di recuperare metadati supplementari da. AWS Glue
- glue_catalog: (facoltativo) utilizza questa opzione per specificare un catalogo AWS Glue multi-account. Per impostazione predefinita, il connettore tenta di ottenere metadati dal proprio account. AWS Glue
- default_docdb: se presente, specifica una stringa di connessione DocumentDB da utilizzare quando non esiste alcuna variabile di ambiente specifica del catalogo.
- disable_projection_and_casing: (facoltativo) disabilita la proiezione e la formattazione delle maiuscole. Usa questa opzione se desideri eseguire query sulle tabelle Amazon DocumentDB che utilizzano nomi di colonna con distinzione tra maiuscole e minuscole. Il parametro `disable_projection_and_casing` utilizza i seguenti valori per specificare il comportamento di formattazione di maiuscole e minuscole e della mappatura delle colonne:
  - false: si tratta dell'impostazione di default. La proiezione è abilitata e il connettore prevede che tutti i nomi delle colonne siano in minuscolo.

- true: disabilita la proiezione e la combinazione di maiuscole e minuscole. Quando utilizzi il parametro `disable_projection_and_casing`, tieni presente i seguenti punti:
  - L'utilizzo del parametro può comportare un consumo maggiore di larghezza di banda. Inoltre, se la funzione Lambda non si trova nella stessa Regione AWS dell'origine dati, i costi di trasferimento standard tra Regioni AWS saranno più elevati a causa del maggiore utilizzo della larghezza di banda. Per ulteriori informazioni sui costi di trasferimento tra regioni, consulta la sezione Costi di trasferimento AWS dei dati per le architetture server e serverless nel blog di Partner Network. AWS
  - Dato che viene trasferito un numero maggiore di byte e poiché un numero maggiore di byte richiede un tempo di deserializzazione maggiore, la latenza complessiva può aumentare.
- enable_case_insensitive_match — (Facoltativo) When, esegue ricerche senza distinzione tra maiuscole e minuscole su nomi di schemi e `true` tabelle in Amazon DocumentDB. Il valore predefinito è `false`. Utilizzalo se la tua query contiene nomi di schemi o tabelle in maiuscolo.

Specifica delle stringhe di connessione

Puoi specificare una o più proprietà che definiscono i dettagli di connessione DocumentDB per le istanze DocumentDB utilizzate con il connettore. A tale scopo, imposta una variabile di ambiente Lambda che corrisponda al nome del catalogo che desideri utilizzare in Athena. Ad esempio, supponiamo di voler utilizzare le seguenti query per interrogare due diverse istanze DocumentDB da Athena:

```
SELECT * FROM "docdb_instance_1".database.table
```

```
SELECT * FROM "docdb_instance_2".database.table
```

Prima di poter utilizzare queste due istruzioni SQL, devi aggiungere due variabili di ambiente alla funzione Lambda: `docdb_instance_1` e `docdb_instance_2`. Il valore di ciascuno deve essere una stringa di connessione DocumentDB nel formato seguente:

```
mongodb://:@:/?ssl=true&ssl_ca_certs=rds-combined-ca-bundle.pem&replicaSet=rs0
```

Utilizzo dei segreti

Facoltativamente, è possibile utilizzare AWS Secrets Manager per intero o in parte il valore per i dettagli della stringa di connessione. Per utilizzare la funzione Athena Federated Query con Secrets

Manager, il VPC collegato alla funzione Lambda dovrebbe disporre dell'accesso a Internet o di un endpoint VPC per connettersi a Secrets Manager.

Se si utilizza la sintassi `${my_secret}` per inserire il nome di un segreto di Secrets Manager nella stringa di connessione, il connettore sostituisce esattamente `${my_secret}` con il valore di testo semplice di Secrets Manager. I segreti devono essere archiviati come segreti di testo semplice con valore *<username>*`:`*<password>*. I segreti memorizzati come `{username:`*<username>*`,password:`*<password>*`}` non verranno passati correttamente alla stringa di connessione.

I segreti possono essere utilizzati esclusivamente anche per l'intera stringa di connessione e il nome utente e la password possono essere definiti all'interno del segreto.

Ad esempio, supponiamo di impostare la variabile di ambiente Lambda per `docdb_instance_1` sul seguente valore:

```
mongodb://${docdb_instance_1_creds}@myhostname.com:123/?ssl=true&ssl_ca_certs=rds-
combined-ca-bundle.pem&replicaSet=rs0
```

L'SDK Athena Query Federation tenta automaticamente di recuperare un segreto denominato `docdb_instance_1_creds` da Secrets Manager e inietta quel valore al posto di `${docdb_instance_1_creds}`. Qualsiasi parte della stringa di connessione racchiusa entro la combinazione di caratteri `${ }` viene interpretata come un segreto da Secrets Manager. Se specifichi un nome del segreto che il connettore non riesce a trovare in Secrets Manager, il connettore non sostituisce il testo.

Configurazione di database e tabelle in AWS Glue

Poiché la funzionalità di inferenza dello schema integrata nel connettore esegue la scansione di un numero limitato di documenti e supporta solo un sottoinsieme di tipi di dati, è consigliabile utilizzarla invece AWS Glue per i metadati.

Per abilitare una AWS Glue tabella da utilizzare con Amazon DocumentDB, devi disporre di un database e di una tabella per il AWS Glue database e la raccolta DocumentDB per i quali desideri fornire metadati supplementari.

Per utilizzare una tabella per metadati supplementari AWS Glue

1. Quando modificate la tabella e il database nella AWS Glue console, aggiungete la seguente proprietà della tabella.

- docdb-metadata-flag— Questa proprietà indica al connettore DocumentDB che il connettore può utilizzare la tabella per metadati supplementari. Puoi fornire qualsiasi valore per `docdb-metadata-flag`, purché la proprietà `docdb-metadata-flag` sia presente nell'elenco delle proprietà della tabella.

2. (Facoltativo) Aggiungi la proprietà della tabella sourceTable. Questa proprietà definisce il nome della tabella di origine in Amazon DocumentDB. Utilizza questa proprietà se le regole di denominazione delle AWS Glue tabelle impediscono di creare una AWS Glue tabella con lo stesso nome della tabella Amazon DocumentDB. Ad esempio, le lettere maiuscole non sono consentite nei nomi delle tabelle AWS Glue , ma sono permesse nei nomi delle tabelle Amazon DocumentDB.
3. (Facoltativo) Aggiungi la proprietà della tabella columnMapping. Questa proprietà definisce le mappature dei nomi delle colonne. Utilizza questa proprietà se le regole di denominazione delle AWS Glue colonne impediscono di creare una AWS Glue tabella con gli stessi nomi di colonna di quelli della tabella Amazon DocumentDB. Ciò può essere utile perché le lettere maiuscole sono consentite nei nomi delle colonne Amazon DocumentDB, ma non sono permesse nei nomi delle colonne AWS Glue .

   Il valore della proprietà `columnMapping` dovrebbe essere un insieme di mappature nel formato `col1=Col1,col2=Col2`.

   > **Note**
   >
   > Il mapping di colonne si applica solo ai nomi delle colonne di primo livello e non ai campi annidati.

   Dopo aver aggiunto la proprietà AWS Glue `columnMapping` table, puoi rimuovere la variabile di ambiente `disable_projection_and_casing` Lambda.
4. Assicurati di utilizzare i tipi di dati AWS Glue appropriati elencati in questo documento.

Supporto dei tipi di dati

Questa sezione elenca i tipi di dati utilizzati dal connettore DocumentDB per l'inferenza dello schema e i tipi di dati quando vengono utilizzati i AWS Glue metadati.

Tipi di dati di inferenza dello schema

La funzionalità di inferenza dello schema del connettore DocumentDB tenta di dedurre i valori come appartenenti a uno dei seguenti tipi di dati. Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per Amazon DocumentDB, Java e Apache Arrow.

Apache Arrow	Java o DocDB
VARCHAR	Stringa
INT	Numero intero
BIGINT	Long
BIT	Booleano
FLOAT4	Float
FLOAT8	Doppio
TIMESTAMPSEC	Data
VARCHAR	ObjectId
LIST	Elenco
STRUCT	Documento

AWS Glue tipi di dati

Se si utilizzano AWS Glue metadati supplementari, è possibile configurare i seguenti tipi di dati. La tabella mostra i tipi di dati corrispondenti per AWS Glue e Apache Arrow.

AWS Glue	Apache Arrow
int	INT
bigint	BIGINT
double	FLOAT8

AWS Glue	Apache Arrow
float	FLOAT4
booleano	BIT
binary	VARBINARY
string	VARCHAR
Elenco	LIST
Struct	STRUCT

Autorizzazioni richieste

Consulta la sezione `Policies` del file athena-docdb.yaml per i dettagli completi delle policy IAM richieste da questo connettore. L'elenco che segue riporta un riepilogo delle autorizzazioni richieste.

- Accesso in scrittura ad Amazon S3: per trasferire i risultati di query di grandi dimensioni, il connettore richiede l'accesso in scrittura a una posizione in Amazon S3.
- Athena GetQueryExecution: il connettore utilizza questa autorizzazione per fallire rapidamente quando la query Athena upstream è terminata.
- AWS Glue Data Catalog— Il connettore DocumentDB richiede l'accesso in sola lettura per AWS Glue Data Catalog ottenere informazioni sullo schema.
- CloudWatch Registri: il connettore richiede l'accesso ai CloudWatch registri per l'archiviazione dei registri.
- AWS Secrets Manager accesso in lettura: se si sceglie di archiviare i dettagli degli endpoint di DocumentDB in Secrets Manager, è necessario concedere al connettore l'accesso a tali segreti.
- Accesso VPC: il connettore richiede la capacità di collegare e scollegare le interfacce al VPC in modo che possa connettersi ad esso e comunicare con le istanze DocumentDB.

Prestazioni

Il connettore Amazon DocumentDB per Athena al momento non supporta le scansioni in parallelo, ma tenta di eseguire il pushdown dei predicati come parte delle query DocumentDB e i predicati

sugli indici della raccolta DocumentDB risultano in una quantità di dati scansionati significativamente inferiore.

La funzione Lambda esegue il pushdown dei predicati per ridurre la quantità di dati scansionati dalla query. Tuttavia, la selezione di un sottoinsieme di colonne a volte comporta un runtime delle query più lungo. Le clausole `LIMIT` riducono la quantità di dati scansionati, ma se non viene fornito un predicato, le query `SELECT` con una clausola `LIMIT` eseguiranno la scansione di almeno 16 MB di dati.

Interrogazioni pass-through

Il connettore Amazon DocumentDB Athena supporta le query passthrough ed è basato su NoSQL. Per informazioni sull'esecuzione di query in Amazon DocumentDB, consulta la sezione Querying nella Amazon DocumentDB Developer Guide.

Per utilizzare le query passthrough con Amazon DocumentDB, utilizza la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            database => 'database_name',
            collection => 'collection_name',
            filter => '{query_syntax}'
        ))
```

L'esempio seguente interroga il *`example`* database all'interno della *`TPCDS`* raccolta, filtrando tutti i libri con il titolo Bill of Rights.

```
SELECT * FROM TABLE(
        system.query(
            database => 'example',
            collection => 'tpcds',
            filter => '{title: "Bill of Rights"}'
        ))
```

Risorse aggiuntive

- Per un articolo sull'utilizzo di Amazon Athena Federated Query per connettere un database MongoDB ad Amazon per creare dashboard e visualizzazioni, consulta QuickSight Visualizzare i dati MongoDB da Amazon usando Amazon Athena Federated Query nel Big Data Blog. QuickSight AWS
- Per ulteriori informazioni su questo connettore, visita il sito corrispondente su .com. GitHub

Connettore Amazon Athena DynamoDB

Il connettore Amazon Athena DynamoDB consente ad Amazon Athena di comunicare con DynamoDB in modo da poter eseguire query sulle tabelle con SQL. Operazioni di scrittura come INSERT INTO non sono supportate.

Se hai abilitato Lake Formation nel tuo account, il ruolo IAM per il tuo connettore Lambda federato Athena che hai distribuito nell'accesso in lettura deve avere accesso in lettura in AWS Serverless Application Repository Lake Formation a. AWS Glue Data Catalog

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore DynamoDB.

- spill_bucket: specifica il bucket Amazon S3 per i dati che superano i limiti della funzione Lambda.
- spill_prefix: (facoltativo) per impostazione predefinita, viene utilizzata una sottocartella nello `spill_bucket` specificato chiamata `athena-federation-spill`. Ti consigliamo di configurare un ciclo di vita dell'archiviazione di Amazon S3 in questa posizione per eliminare gli spill più vecchi di un numero predeterminato di giorni o ore.
- spill_put_request_headers: (facoltativo) una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.
- kms_key_id: (facoltativo) per impostazione predefinita, tutti i dati riversati in Amazon S3 vengono crittografati utilizzando la modalità di crittografia autenticata AES-GCM e una chiave generata casualmente. Per fare in modo che la tua funzione Lambda utilizzi chiavi di crittografia più potenti generate da KMS come `a7e63k4b-8loc-40db-a2a1-4d0en2cd8331`, puoi specificare l'ID della chiave KMS.
- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante

KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.

- disable_glue — (Facoltativo) Se presente e impostato su true, il connettore non tenta di recuperare metadati supplementari da. AWS Glue
- glue_catalog: (facoltativo) utilizza questa opzione per specificare un catalogo AWS Glue multi-account. Per impostazione predefinita, il connettore tenta di ottenere metadati dal proprio account. AWS Glue
- disable_projection_and_casing: (facoltativo) disabilita la proiezione e la formattazione delle maiuscole. Utilizzatelo se desiderate interrogare le tabelle DynamoDB i cui nomi di colonna contengono maiuscole e minuscole e non desiderate specificare `columnMapping` una proprietà sulla tabella. AWS Glue

  Il parametro `disable_projection_and_casing` utilizza i seguenti valori per specificare il comportamento di formattazione di maiuscole e minuscole e della mappatura delle colonne:

  - auto: disabilita la proiezione e la formattazione di maiuscole e minuscole quando viene rilevato un tipo precedentemente non supportato e la mappatura dei nomi delle colonne non è impostata sulla tabella. Si tratta dell'impostazione di default.
  - always: disabilita la proiezione e la formattazione di maiuscole e minuscole in modo incondizionato. È utile quando i nomi delle colonne DynamoDB contengono lettere maiuscole e minuscole ma non desideri specificare alcuna mappatura dei nomi delle colonne.

  Quando utilizzi il parametro `disable_projection_and_casing`, tieni presente i seguenti punti:

  - L'utilizzo del parametro può comportare un consumo maggiore di larghezza di banda. Inoltre, se la funzione Lambda non si trova nella stessa Regione AWS dell'origine dati, i costi di trasferimento standard tra Regioni AWS saranno più elevati a causa del maggiore utilizzo della larghezza di banda. Per ulteriori informazioni sui costi di trasferimento tra regioni, consulta la sezione Costi di trasferimento AWS dati per le architetture server e serverless nel blog di Partner Network. AWS
  - Dato che viene trasferito un numero maggiore di byte e poiché un numero maggiore di byte richiede un tempo di deserializzazione maggiore, la latenza complessiva può aumentare.

Configurazione di database e tabelle in AWS Glue

Poiché la capacità di inferenza dello schema integrata nel connettore è limitata, potresti volerla utilizzare AWS Glue per i metadati. A tale scopo, è necessario disporre di un database e di una tabella. AWS Glue Per abilitarli all'uso con DynamoDB, è necessario modificarne le proprietà.

Per modificare le proprietà del database nella AWS Glue console

1. Accedere AWS Management Console e aprire la AWS Glue console all'indirizzo https://console.aws.amazon.com/glue/.
2. Nel riquadro di navigazione, espandi Data Catalog, quindi scegli Database.

   Nella pagina Databases (Database), puoi modificare un database esistente oppure scegliere Add database (Aggiungi database) per crearne uno.
3. Nell'elenco dei database, scegli il link del database da modificare.
4. Scegli Modifica.
5. Nella pagina Aggiorna un database, in Impostazioni del database, per Posizione, aggiungi la stringa**`dynamo-db-flag`**. Questa parola chiave indica che il database contiene tabelle che il connettore Athena DynamoDB utilizza per metadati supplementari ed è necessaria per database diversi da. AWS Glue `default` La proprietà `dynamo-db-flag` è utile per filtrare i database negli account che hanno numerosi database.
6. Scegli Update Database (Aggiorna database).

Per modificare le proprietà delle tabelle nella console AWS Glue

1. Accedere AWS Management Console e aprire la AWS Glue console all'indirizzo https://console.aws.amazon.com/glue/.
2. Nel riquadro di navigazione, espandi Data Catalog, quindi scegli Tabelle.
3. Nella pagina Tabelle, nell'elenco delle tabelle, scegli il nome collegato della tabella che desideri modificare.
4. Scegli Actions (Operazioni), Edit (Modifica).
5. Nella pagina Edit table (Modifica tabella), nella sezione Table properties (Proprietà della tabella), aggiungi le seguenti proprietà della tabella in base alle necessità. Se si utilizza il crawler AWS Glue DynamoDB, queste proprietà vengono impostate automaticamente.

   - DynamoDB: stringa che indica al connettore DynamoDB per Athena che la tabella può essere utilizzata per metadati supplementari. Inserisci la stringa `dynamodb` nelle proprietà della tabella in un campo denominato classification (classificazione), con una corrispondenza esatta.

> **Note**
>
> La pagina Imposta le proprietà della tabella che fa parte del processo di creazione della tabella nella AWS Glue console contiene una sezione Formato dati con un campo Classificazione. Qui non puoi immettere o scegliere `dynamodb`. Invece, dopo aver creato la tabella, completa le operazioni per modificare la tabella e inserire `classification` e `dynamodb` come coppia chiave-valore nella sezione Proprietà della tabella.

- sourceTable: proprietà facoltativa della tabella che definisce il nome della tabella di origine in DynamoDB. Usalo se le regole di denominazione delle AWS Glue tabelle ti impediscono di creare una AWS Glue tabella con lo stesso nome della tabella DynamoDB. Ad esempio, le lettere maiuscole non sono consentite nei nomi delle AWS Glue tabelle, ma sono consentite nei nomi delle tabelle DynamoDB.
- columnMapping: proprietà facoltativa della tabella che definisce le mappature dei nomi delle colonne. Usalo se le regole di denominazione delle AWS Glue colonne ti impediscono di creare una AWS Glue tabella con gli stessi nomi di colonna della tabella DynamoDB. Ad esempio, le lettere maiuscole non sono consentite nei nomi delle AWS Glue colonne, ma sono consentite nei nomi delle colonne di DynamoDB. Il valore della proprietà dovrebbe essere nel formato col1=Col1,col2=Col2. Il mapping di colonne si applica solo ai nomi delle colonne di primo livello e non ai campi nidificati.
- defaultTimeZone— Proprietà opzionale della tabella che viene applicata a `datetime` valori `date` o valori che non hanno un fuso orario esplicito. L'impostazione di questo valore è una buona pratica per evitare discrepanze tra il fuso orario predefinito dell'origine dei dati e il fuso orario della sessione Athena.
- datetimeFormatMapping— Proprietà opzionale della tabella che specifica il `datetime` formato `date` or da utilizzare per l'analisi dei dati da una colonna del tipo di dati AWS Glue `date` or`timestamp`. Se questa proprietà non viene specificata, il connettore tenta di dedurre un formato ISO-8601. Se il connettore non è in grado di dedurre il formato di `date` o `datetime` oppure di analizzare la stringa non elaborata, il valore viene omesso dal risultato.

  Il valore `datetimeFormatMapping` deve essere nel formato `col1=someformat1,col2=someformat2`. Di seguito sono riportati alcuni formati di esempio:

```
yyyyMMdd'T'HHmmss
```

```
ddMMyyyy'T'HH:mm:ss
```

Se la tua colonna ha valori `date` o `datetime` senza fuso orario e desideri utilizzare la colonna nella clausola `WHERE`, imposta la proprietà `datetimeFormatMapping` per la colonna.

6. Se definisci le colonne in modalità manuale, assicurati di utilizzare i tipi di dati appropriati. Se hai usato un crawler, convalida le colonne e i tipi rilevati dal crawler.
7. Selezionare Salva.

Autorizzazioni richieste

Consulta la sezione `Policies` del file athena-dynamodb.yaml per i dettagli completi delle policy IAM richieste da questo connettore. L'elenco che segue riporta un riepilogo delle autorizzazioni richieste.

- Accesso in scrittura ad Amazon S3: per trasferire i risultati di query di grandi dimensioni, il connettore richiede l'accesso in scrittura a una posizione in Amazon S3.
- Athena GetQueryExecution: il connettore utilizza questa autorizzazione per fallire rapidamente quando la query Athena upstream è terminata.
- AWS Glue Data Catalog— Il connettore DynamoDB richiede l'accesso in sola lettura per ottenere informazioni sullo AWS Glue Data Catalog schema.
- CloudWatch Registri: il connettore richiede l'accesso ai CloudWatch registri per l'archiviazione dei log.
- Accesso in lettura a DynamoDB: il connettore utilizza le operazioni API `DescribeTable`, `ListSchemas`, `ListTables`, `Query` e `Scan`.

Prestazioni

Il connettore DynamoDB per Athena supporta le scansioni in parallelo e tenta di eseguire il pushdown dei predicati come parte delle query DynamoDB. Un predicato hash key con valori `X` distinti genera `X` chiamate di query a DynamoDB. Tutti gli altri scenari di predicato danno come risultato un numero `Y` di chiamate di scansione, dove `Y` viene determinato euristicamente in base alla dimensione della tabella e alla velocità di trasmissione effettiva allocata. Tuttavia, la selezione di un sottoinsieme di colonne comporta un runtime di esecuzione delle query più lungo.

Viene eseguito il pushdown delle clausole `LIMIT` e dei predicati semplici, il che può ridurre la quantità di dati scansionati e, di conseguenza, il runtime di esecuzione delle query.

Clausole LIMIT

Una dichiarazione `LIMIT N` riduce la quantità di dati analizzati dalla query. Con il pushdown `LIMIT N`, il connettore restituisce solo le righe `N` ad Athena.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Per migliorare le funzionalità e ridurre la quantità di dati scansionati, il connettore Athena DynamoDB può combinare queste espressioni e inviarle direttamente a DynamoDB.

I seguenti operatori del connettore DynamoDB di Athena supportano il pushdown dei predicati:

- Boolean: AND
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_NULL

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10 and col_b < 10
LIMIT 10
```

Per un articolo sull'utilizzo del pushdown del predicato per migliorare le prestazioni nelle query federate, incluso DynamoDB, consulta Come migliorare le query federate con il pushdown del predicato in Amazon Athena nel Blog sui Big Data di AWS .

Interrogazioni pass-through

Il connettore DynamoDB supporta le query passthrough e utilizza la sintassi PartiQL. L'operazione API GetItemDynamoDB non è supportata. Per informazioni sull'interrogazione di DynamoDB utilizzando PartiQL, consulta le istruzioni PartiQL select per DynamoDB nella Amazon DynamoDB Developer Guide.

Per utilizzare le query passthrough con DynamoDB, utilizzate la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query_string'
        ))
```

Il seguente esempio di query passthrough DynamoDB utilizza PartiQL per restituire un elenco di dispositivi Fire TV Stick che hanno una proprietà successiva al 24/12/22. `DateWatched`

```
SELECT * FROM TABLE(
        system.query(
           query => 'SELECT Devices
                       FROM WatchList
                       WHERE Devices.FireStick.DateWatched[0] > '12/24/22''
        ))
```

Risoluzione dei problemi

Filtri multipli su una colonna con chiave di ordinamento

Messaggio di errore: KeyConditionExpressionsdeve contenere solo una condizione per chiave

Causa: questo problema può verificarsi nella versione 3 del motore Athena nelle query che hanno un filtro con limite sia inferiore sia superiore su una colonna con chiave di ordinamento DynamoDB. Poiché DynamoDB non supporta più di una condizione di filtro su una chiave di ordinamento, quando il connettore tenta di inviare una query a cui sono applicate entrambe le condizioni, viene generato un errore.

Soluzione: aggiorna il connettore alla versione 2023.11.1 o successiva. Per istruzioni sull'aggiornamento di un connettore, consulta Aggiornamento di un connettore origine dati.

Costi

I costi di utilizzo del connettore dipendono dalle AWS risorse sottostanti utilizzate. Dato che le query che utilizzano le scansioni possono consumare un numero elevato di unità di capacità di lettura (RCU), consulta attentamente le informazioni sui prezzi di Amazon DynamoDB.

Risorse aggiuntive

- Per un'introduzione all'utilizzo del connettore Amazon Athena DynamoDB, consulta Accesso, esecuzione di query e join di tabelle Amazon DynamoDB con Athena nella guida AWS Prescriptive Guidance Patterns.

- Per un articolo su come utilizzare il connettore Athena DynamoDB per interrogare i dati in DynamoDB con SQL e visualizzare approfondimenti in Amazon, QuickSight consulta il post del AWS blog Big Data Visualizza gli approfondimenti di Amazon DynamoDB in Amazon usando il connettore Amazon Athena DynamoDB e. QuickSight AWS Glue
- Per un articolo sull'utilizzo del connettore Amazon Athena DynamoDB con Amazon DynamoDB, Athena e Amazon per creare una dashboard di governance semplice, consulta il post del blog Big AWS Data Interroga tabelle QuickSight Amazon DynamoDB cross-account usando Amazon Athena Federated Query.
- Per ulteriori informazioni su questo connettore, visita il sito corrispondente su .com. GitHub

Connettore Google Amazon Athena BigQuery

Il connettore Amazon Athena per Google BigQueryconsente ad Amazon Athena di eseguire query SQL sui tuoi dati Google. BigQuery

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le funzioni Lambda hanno un valore di timeout massimo di 15 minuti. Ogni split esegue una query su BigQuery e deve terminare con un tempo sufficiente per memorizzare i risultati da consentire ad Athena di leggerli. Se la funzione Lambda scade, la query ha esito negativo.
- Google distingue tra maiuscole e BigQuery minuscole. Il connettore tenta di correggere la formattazione di maiuscole e minuscole dei nomi dei set di dati e delle tabelle, ma non apporta tale tipo di correzione agli ID di progetto. Ciò è necessario perché Athena applica la formattazione minuscola a tutti i metadati. Queste correzioni effettuano molte chiamate aggiuntive verso Google BigQuery.
- I tipi di dati binari non sono supportati.
- A causa della BigQuery concorrenza e dei limiti di quota di Google, il connettore potrebbe riscontrare problemi con i limiti di quota di Google. Per evitare questi problemi, imponi a Google BigQuery il maggior numero possibile di vincoli. Per informazioni sulle BigQuery quote, consulta Quote e limiti nella documentazione di Google. BigQuery

Parametri

Utilizza le variabili di ambiente Lambda in questa sezione per configurare il connettore Google BigQuery .

- spill_bucket: specifica il bucket Amazon S3 per i dati che superano i limiti della funzione Lambda.
- spill_prefix: (facoltativo) per impostazione predefinita, viene utilizzata una sottocartella nello `spill_bucket` specificato chiamata `athena-federation-spill`. Ti consigliamo di configurare un ciclo di vita dell'archiviazione di Amazon S3 in questa posizione per eliminare gli spill più vecchi di un numero predeterminato di giorni o ore.
- spill_put_request_headers: (facoltativo) una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.
- kms_key_id: (facoltativo) per impostazione predefinita, tutti i dati riversati in Amazon S3 vengono crittografati utilizzando la modalità di crittografia autenticata AES-GCM e una chiave generata casualmente. Per fare in modo che la tua funzione Lambda utilizzi chiavi di crittografia più potenti generate da KMS come `a7e63k4b-8loc-40db-a2a1-4d0en2cd8331`, puoi specificare l'ID della chiave KMS.
- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.
- gcp_project_id: l'ID del progetto (non il nome del progetto) che contiene i set di dati da cui il connettore deve leggere (ad esempio, `semiotic-primer-1234567`).
- secret_manager_gcp_creds_name — Il nome del segreto all'interno AWS Secrets Manager del quale sono contenute le credenziali in formato JSON (ad esempio,). BigQuery `GoogleCloudPlatformCredentials`
- big_query_endpoint — (Facoltativo) L'URL di un endpoint privato. BigQuery Utilizza questo parametro quando desideri accedere tramite un endpoint privato. BigQuery

Divisioni e visualizzazioni

Poiché il BigQuery connettore utilizza l'API BigQuery Storage Read per interrogare le tabelle e l'API BigQuery Storage non supporta le visualizzazioni, il connettore utilizza il BigQuery client con un'unica suddivisione per le visualizzazioni.

Prestazioni

Per interrogare le tabelle, il BigQuery connettore utilizza l'API BigQuery Storage Read, che utilizza un protocollo basato su RPC che fornisce un accesso rapido allo storage BigQuery gestito. Per ulteriori informazioni sull'API BigQuery Storage Read, consulta Utilizzare l'API BigQuery Storage Read per leggere i dati delle tabelle nella documentazione di Google Cloud.

La selezione di un sottoinsieme di colonne velocizza notevolmente il runtime delle query e riduce i dati scansionati. Il connettore è soggetto a errori di query all'aumentare della simultaneità e generalmente è lento.

Il BigQuery connettore Google Athena esegue il pushdown dei predicati per ridurre i dati scansionati dalla query. `LIMIT`clausole, `ORDER BY` clausole, predicati semplici ed espressioni complesse vengono inserite nel connettore per ridurre la quantità di dati analizzati e ridurre il tempo di esecuzione delle query.

Clausole LIMIT

Una dichiarazione `LIMIT N` riduce la quantità di dati analizzati dalla query. Con il pushdown `LIMIT N`, il connettore restituisce solo le righe `N` ad Athena.

Query top N

Una query top `N` principale specifica un ordinamento dei set di risultati e un limite al numero di righe restituite. Puoi utilizzare questo tipo di query per determinare i valori massimi top `N` o i valori minimi top `N` per i set di dati. Con il pushdown top `N`, il connettore restituisce solo le righe ordinate `N` ad Athena.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il BigQuery connettore Google Athena può combinare queste espressioni e inviarle direttamente a Google BigQuery per funzionalità avanzate e ridurre la quantità di dati scansionati.

I seguenti operatori del BigQuery connettore Google Athena supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_DISTINCT_FROM, NULL_IF, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%')
ORDER BY col_a DESC
LIMIT 10;
```

Interrogazioni pass-through

Il BigQuery connettore Google supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Google BigQuery, puoi utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a una fonte di dati in Google. BigQuery La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
```

```
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Il progetto Amazon Athena Google BigQuery Connector è concesso in licenza con la licenza Apache-2.0.

Utilizzando questo connettore, riconosci l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetti i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su GitHub .com.

Connettore Google Cloud Storage per Amazon Athena

Il connettore Google Cloud Storage di Amazon Athena consente ad Amazon Athena di eseguire query su file Parquet e CSV archiviati in un bucket Google Cloud Storage (GCS). Dopo aver raggruppato uno o più file Parquet o CSV in una cartella partizionata o non partizionata in un bucket GCS, sarà possibile organizzarli in una tabella di database AWS Glue.

Se hai abilitato Lake Formation nel tuo account, il ruolo IAM per il tuo connettore Lambda federato Athena che hai distribuito nell'accesso in lettura deve avere accesso in lettura in AWS Serverless Application Repository Lake Formation a. AWS Glue Data Catalog

Per un articolo che mostra come utilizzare Athena per eseguire query su file Parquet o CSV in un bucket GCS, consulta il post sul blog AWS Big Data Use Amazon Athena per interrogare i dati archiviati in Google Cloud Platform.

Prerequisiti

- Configura un AWS Glue database e una tabella che corrispondano al tuo bucket e alle tue cartelle in Google Cloud Storage. Per i passaggi, consulta Configurazione di database e tabelle in AWS Glue più avanti in questo documento.
- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .
- Attualmente, il connettore supporta solo il `VARCHAR` tipo per le colonne di partizione (`string`o `varchar` in uno schema AWS Glue tabellare). Altri tipi di campi di partizione generano errori quando vengono interrogati in Athena.

Termini

I seguenti termini si riferiscono al connettore GCS.

- Gestore: un gestore Lambda che accede al bucket GCS. Un gestore può gestire i metadati o i record di dati.
- Metadata handler (Gestore dei metadati): un gestore Lambda che recupera i metadati dal bucket GCS.
- Record handler (Gestore dei record): un gestore Lambda che recupera i record di dati dal bucket GCS.
- Composite handler (Gestore composito): un gestore Lambda che recupera sia i metadati sia i record di dati dal bucket GCS.

Tipi di file supportati

Il connettore GCS supporta i tipi di file Parquet e CSV.

> **Note**
>
> Assicurati di non inserire entrambi i file CSV e Parquet nello stesso bucket o percorso GCS. Ciò potrebbe causare un errore di runtime quando si prova a leggere i file Parquet come CSV o viceversa.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore GCS.

- spill_bucket: specifica il bucket Amazon S3 per i dati che superano i limiti della funzione Lambda.

- spill_prefix: (facoltativo) per impostazione predefinita, viene utilizzata una sottocartella nello `spill_bucket` specificato chiamata `athena-federation-spill`. Ti consigliamo di configurare un ciclo di vita dell'archiviazione di Amazon S3 in questa posizione per eliminare gli spill più vecchi di un numero predeterminato di giorni o ore.
- spill_put_request_headers: (facoltativo) una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.
- kms_key_id: (facoltativo) per impostazione predefinita, tutti i dati riversati in Amazon S3 vengono crittografati utilizzando la modalità di crittografia autenticata AES-GCM e una chiave generata casualmente. Per fare in modo che la tua funzione Lambda utilizzi chiavi di crittografia più potenti generate da KMS come `a7e63k4b-8loc-40db-a2a1-4d0en2cd8331`, puoi specificare l'ID della chiave KMS.
- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.
- secret_manager_gcp_creds_name — Il nome del segreto AWS Secrets Manager che contiene le tue credenziali GCS in formato JSON (ad esempio,). `GoogleCloudPlatformCredentials`

Configurazione di database e tabelle in AWS Glue

Poiché la funzionalità integrata di inferenza dello schema del connettore GCS è limitata, si consiglia di utilizzarla AWS Glue per i metadati. Le procedure seguenti mostrano come creare un database e una tabella a AWS Glue cui è possibile accedere da Athena.

Creazione di un database in AWS Glue

È possibile utilizzare la AWS Glue console per creare un database da utilizzare con il connettore GCS.

Per creare un database in AWS Glue

1. Accedere AWS Management Console e aprire la AWS Glue console all'indirizzo https://console.aws.amazon.com/glue/.
2. Nel pannello di navigazione seleziona Databases (Database).

3. Scegli Aggiungi database.
4. In Name (Nome) immetti un nome per il database che desideri utilizzare con il connettore GCS.
5. Per Posizione, specificare `s3://google-cloud-storage-flag.` Questa posizione indica al connettore GCS che il AWS Glue database contiene tabelle per i dati GCS da interrogare in Athena. Il connettore riconosce i database in Athena che hanno questo flag e ignora quelli che non lo hanno.
6. Scegliere Crea database.

Creazione di una tabella in AWS Glue

Ora puoi creare una tabella per il database. Quando si crea una AWS Glue tabella da utilizzare con il connettore GCS, è necessario specificare metadati aggiuntivi.

Per creare una tabella nella console AWS Glue

1. Nella AWS Glue console, dal pannello di navigazione, scegli Tabelle.
2. Nella pagina Tables (Tabelle), scegli Add table (Aggiungi tabella).
3. Nella pagina Set table properties (Imposta proprietà tabella) immetti le seguenti informazioni.
   - Name (Nome): un nome univoco per la tabella.
   - Database: scegli il database AWS Glue creato per il connettore GCS.
   - Include path (Includi percorso): nella sezione Data store (Archivio dati), in Include path (Includi percorso), inserisci la posizione URI per GCS preceduta da `gs://` (ad esempio, `gs://gcs_table/data/`). Se disponi di una o più cartelle di partizione, non includerle nel percorso.

     > **Note**
     >
     > Quando si immette il percorso di tabella non `s3://`, la console AWS Glue riporta un errore. Puoi ignorare questo errore. La tabella verrà creata correttamente.

   - Data format (Formato dati): per Classification (Classificazione), seleziona CSV o Parquet.
4. Seleziona Successivo.
5. Nella pagina Choose or define schema (Scegli o definisci schema), la definizione di uno schema di tabella è altamente consigliata, ma non obbligatoria. Se non viene definito uno schema, il connettore GCS prova a dedurne uno per tuo conto.

Esegui una di queste operazioni:

- Se desideri che il connettore GCS provi a dedurre uno schema per tuo conto, scegli Next (Successivo), quindi Create (Crea).
- Per definire uno schema personale, seguire la procedura descritta nella sezione successiva.

Definizione di uno schema di tabella in AWS Glue

La definizione di uno schema di tabella in AWS Glue richiede più passaggi, ma offre un maggiore controllo sul processo di creazione della tabella.

Per definire uno schema per la tabella in AWS Glue

1. Nella pagina Choose or define schema (Scegli o definisci schema), seleziona Add (Aggiungi).
2. Utilizza la finestra di dialogo Add schema entry (Aggiungi voce allo schema) per fornire un nome di colonna e un tipo di dati.
3. Per designare la colonna come colonna di partizione, seleziona l'opzione Set as partition key (Imposta come chiave di partizione).
4. Seleziona Save (Salva) per salvare la colonna.
5. Scegli Add (Aggiungi) per aggiungere un'altra colonna.
6. Dopo aver aggiunto le colonne, seleziona Next (Successivo).
7. Nella pagina Review and create (Rivedi e crea), verifica la tabella, quindi scegli Create (Crea).
8. Se lo schema contiene informazioni sulle partizioni, completa la procedura descritta nella sezione successiva per aggiungere un modello di partizione alle proprietà della tabella in AWS Glue.

Aggiungere uno schema di partizione alle proprietà della tabella in AWS Glue

Se i bucket GCS hanno delle partizioni, è necessario aggiungere il modello di partizione alle proprietà della tabella in AWS Glue.

Per aggiungere informazioni sulle partizioni alle proprietà della tabella AWS Glue

1. Nella pagina dei dettagli della tabella in cui hai creato AWS Glue, scegli Azioni, Modifica tabella.
2. Nella pagina Edit table (Modifica tabella), scorri verso il basso fino alla sezione Table properties (Proprietà della tabella).
3. Scegli Add (Aggiungi) per aggiungere una chiave di partizione.

4. In Chiave, inserire **`partition.pattern`**. Questa chiave definisce il modello del percorso della cartella.
5. In Value (Valore), inserisci un modello di percorso della cartella come **`StateName=${statename}/ZipCode=${zipcode}/`**, dove **`statename`** e **`zipcode`** racchiusi tra **`${}`** sono i nomi delle colonne delle partizioni. Il connettore GCS supporta schemi di partizione Hive e non Hive.
6. Quando hai terminato, seleziona Save (Salva).
7. Per visualizzare le proprietà della tabella appena creata, scegli la scheda Advanced properties (Proprietà avanzate).

A questo punto, è possibile passare alla console Athena. Il database e la tabella in cui hai creato AWS Glue sono disponibili per l'interrogazione in Athena.

Supporto dei tipi di dati

Le tabelle seguenti mostrano i tipi di dati supportati per CSV e Parquet.

CSV

Natura dei dati	Tipo di dati dedotto
I dati hanno l'aspetto di un numero	BIGINT
I dati hanno l'aspetto di una stringa	VARCHAR
I dati hanno l'aspetto di una virgola mobile (mobile, doppia o decimale)	DOUBLE
I dati hanno l'aspetto di una data	Timestamp
Dati che contengono valori vero/falso	BOOL

Parquet

PARQUET	Athena (freccia)
BINARY	VARCHAR

PARQUET	Athena (freccia)
BOOLEAN	BOOL
DOUBLE	DOUBLE
ENUM	VARCHAR
FIXED_LEN _BYTE_ARRAY	DECIMAL
FLOAT	FLOAT (32 bit)
INT32	1. INT32 2. DATEDAY (quando il tipo logico della colonna Parquet è DATE)
INT64	1. INT64 2. TIMESTAMP (quando il tipo logico della colonna Parquet è TIMESTAMP)
INT96	Timestamp
MAP	MAP
STRUCT	STRUCT
LIST	LIST

Autorizzazioni richieste

Consulta la sezione `Policies` del file athena-gcs.yaml per i dettagli completi delle policy IAM richieste da questo connettore. L'elenco che segue riporta un riepilogo delle autorizzazioni richieste.

- Accesso in scrittura ad Amazon S3: per trasferire i risultati di query di grandi dimensioni, il connettore richiede l'accesso in scrittura a una posizione in Amazon S3.
- Athena GetQueryExecution: il connettore utilizza questa autorizzazione per fallire rapidamente quando la query Athena upstream è terminata.
- AWS Glue Data Catalog— Il connettore GCS richiede l'accesso in sola lettura per ottenere informazioni sullo schema. AWS Glue Data Catalog

- CloudWatch Registri: il connettore richiede l'accesso ai CloudWatch registri per l'archiviazione dei registri.

Prestazioni

Quando lo schema della tabella contiene campi di partizione e la proprietà della tabella `partition.pattern` è configurata correttamente, è possibile includere il campo di partizione nella clausola `WHERE` delle query. Per tali query, il connettore GCS utilizza le colonne delle partizioni per perfezionare il percorso della cartella GCS ed evitare la scansione di file non necessari nelle cartelle GCS.

Per i set di dati Parquet, la selezione di un sottoinsieme di colonne comporta un minor numero di dati da scansionare. Ciò si traduce in genere in un runtime di esecuzione delle query più breve quando viene applicata la proiezione di colonne.

Per i set di dati CSV, la proiezione di colonne non è supportata e non riduce la quantità di dati da scansionare.

Le clausole `LIMIT` riducono la quantità di dati scansionati, ma se non viene fornito un predicato, le query `SELECT` con una clausola `LIMIT` eseguiranno la scansione di almeno 16 MB di dati. Il connettore GCS esegue la scansione di un maggior numero di dati per i set di dati più grandi rispetto ai set di dati più piccoli indipendentemente dalla clausola `LIMIT` applicata. Ad esempio, la query `SELECT * LIMIT 10000` esegue la scansione di un maggior numero di dati per un set di dati sottostante più grande rispetto a uno più piccolo.

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per ulteriori informazioni su questo connettore, visita il sito corrispondente su GitHub .com.

Connettore Amazon Athena HBase

Il connettore HBase di Amazon Athena consente ad Amazon Athena di comunicare con le istanze Apache HBase in modo da poter eseguire query sui dati HBase con SQL.

A differenza dei tradizionali archivi di dati relazionali, le raccolte HBase non hanno uno schema prestabilito. HBase non dispone di un archivio dei metadati. Ogni voce in una raccolta HBase può avere campi e tipi di dati diversi.

Il connettore HBase supporta due meccanismi per la generazione di informazioni sullo schema della tabella: inferenza dello schema di base e AWS Glue Data Catalog metadati.

L'inferenza dello schema è l'impostazione predefinita. Questa opzione esegue la scansione di un numero limitato di documenti della raccolta, forma un'unione di tutti i campi e forza i campi che hanno tipi di dati non sovrapposti. Questa opzione funziona bene per le raccolte che hanno voci per lo più uniformi.

Per le raccolte con una maggiore varietà di tipi di dati, il connettore supporta il recupero dei metadati da AWS Glue Data Catalog. Se il connettore rileva un AWS Glue database e una tabella che corrispondono allo spazio dei nomi HBase e ai nomi delle raccolte, ottiene le informazioni sullo schema dalla tabella corrispondente. AWS Glue Quando crei la AWS Glue tabella, ti consigliamo di renderla un superset di tutti i campi a cui potresti voler accedere dalla tua raccolta HBase.

Se hai abilitato Lake Formation nel tuo account, il ruolo IAM per il tuo connettore Lambda federato Athena che hai distribuito nell'accesso in lettura deve avere accesso in lettura in AWS Serverless Application Repository Lake Formation a. AWS Glue Data Catalog

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore HBase.

- spill_bucket: specifica il bucket Amazon S3 per i dati che superano i limiti della funzione Lambda.
- spill_prefix: (facoltativo) per impostazione predefinita, viene utilizzata una sottocartella nello `spill_bucket` specificato chiamata `athena-federation-spill`. Ti consigliamo di configurare un ciclo di vita dell'archiviazione di Amazon S3 in questa posizione per eliminare gli spill più vecchi di un numero predeterminato di giorni o ore.

- spill_put_request_headers: (facoltativo) una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.
- kms_key_id: (facoltativo) per impostazione predefinita, tutti i dati riversati in Amazon S3 vengono crittografati utilizzando la modalità di crittografia autenticata AES-GCM e una chiave generata casualmente. Per fare in modo che la tua funzione Lambda utilizzi chiavi di crittografia più potenti generate da KMS come `a7e63k4b-8loc-40db-a2a1-4d0en2cd8331`, puoi specificare l'ID della chiave KMS.
- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.
- disable_glue — (Facoltativo) Se presente e impostato su true, il connettore non tenta di recuperare metadati supplementari da. AWS Glue
- glue_catalog: (facoltativo) utilizza questa opzione per specificare un catalogo AWS Glue multi-account. Per impostazione predefinita, il connettore tenta di ottenere metadati dal proprio account. AWS Glue
- default_hbase: se presente, specifica una stringa di connessione HBase da utilizzare quando non esiste alcuna variabile di ambiente specifica del catalogo.
- enable_case_insensitive_match — (Facoltativo) When, esegue ricerche senza distinzione tra maiuscole e minuscole sui nomi delle tabelle in HBase`true`. Il valore predefinito è `false`. Utilizza se la tua query contiene nomi di tabella in maiuscolo.

Specifica delle stringhe di connessione

Puoi specificare una o più proprietà che definiscono i dettagli di connessione HBase per le istanze HBase utilizzate con il connettore. A tale scopo, imposta una variabile di ambiente Lambda che corrisponda al nome del catalogo che desideri utilizzare in Athena. Ad esempio, supponiamo di voler utilizzare le seguenti query per interrogare due diverse istanze HBase da Athena:

```
SELECT * FROM "hbase_instance_1".database.table
```

```
SELECT * FROM "hbase_instance_2".database.table
```

Prima di poter utilizzare queste due istruzioni SQL, devi aggiungere due variabili di ambiente alla funzione Lambda: `hbase_instance_1` e `hbase_instance_2`. Il valore di ciascuno deve essere una stringa di connessione HBase nel formato seguente:

```
master_hostname:hbase_port:zookeeper_port
```

Utilizzo dei segreti

Facoltativamente, è possibile utilizzare AWS Secrets Manager parzialmente o totalmente il valore per i dettagli della stringa di connessione. Per utilizzare la funzione Athena Federated Query con Secrets Manager, il VPC collegato alla funzione Lambda dovrebbe disporre dell'accesso a Internet o di un endpoint VPC per connettersi a Secrets Manager.

Se si utilizza la sintassi `${my_secret}` per inserire il nome di un segreto di Secrets Manager nella stringa di connessione, il connettore sostituisce il nome del segreto con i valori del nome utente e della password di Secrets Manager.

Ad esempio, supponiamo di impostare la variabile di ambiente Lambda per `hbase_instance_1` sul seguente valore:

```
${hbase_host_1}:${hbase_master_port_1}:${hbase_zookeeper_port_1}
```

L'SDK Athena Query Federation tenta automaticamente di recuperare un segreto denominato `hbase_instance_1_creds` da Secrets Manager e inietta quel valore al posto di `${hbase_instance_1_creds}`. Qualsiasi parte della stringa di connessione racchiusa entro la combinazione di caratteri `${ }` viene interpretata come un segreto da Secrets Manager. Se specifichi un nome del segreto che il connettore non riesce a trovare in Secrets Manager, il connettore non sostituisce il testo.

Configurazione di database e tabelle in AWS Glue

L'inferenza dello schema integrata del connettore supporta solo i valori serializzati in HBase come stringhe (ad esempio, `String.valueOf(int)`). Poiché la capacità integrata di inferenza dello schema del connettore è limitata, in alternativa potresti voler usare AWS Glue per i metadati. Per abilitare una AWS Glue tabella da utilizzare con HBase, è necessario disporre di un AWS Glue database e di una tabella con nomi che corrispondano allo spazio dei nomi e alla tabella HBase per i quali si desidera fornire metadati supplementari. L'uso delle convenzioni di denominazione delle famiglie di colonne HBase è facoltativo ma non obbligatorio.

Per utilizzare una tabella per metadati supplementari AWS Glue

1. Quando modifichi la tabella e il database nella AWS Glue console, aggiungi le seguenti proprietà della tabella:

   - hbase-metadata-flag— Questa proprietà indica al connettore HBase che il connettore può utilizzare la tabella per metadati supplementari. Puoi fornire qualsiasi valore per `hbase-metadata-flag`, purché la proprietà `hbase-metadata-flag` sia presente nell'elenco delle proprietà della tabella.
   - hbase-native-storage-flag— Utilizzate questo flag per attivare o disattivare le due modalità di serializzazione dei valori supportate dal connettore. Per impostazione predefinita, quando questo campo non è presente, il connettore presuppone che tutti i valori siano archiviati in HBase come stringhe. Pertanto, tenterà di analizzare tipi di dati come `INT`, `BIGINT` e `DOUBLE` da HBase come stringhe. Se questo campo è impostato con un valore qualsiasi della tabella AWS Glue, il connettore passa alla modalità di archiviazione «nativa» e tenta di leggere `INT` `BIGINTBIT`, e `DOUBLE` come byte utilizzando le seguenti funzioni:

     ```
     ByteBuffer.wrap(value).getInt()
     ByteBuffer.wrap(value).getLong()
     ByteBuffer.wrap(value).get()
     ByteBuffer.wrap(value).getDouble()
     ```

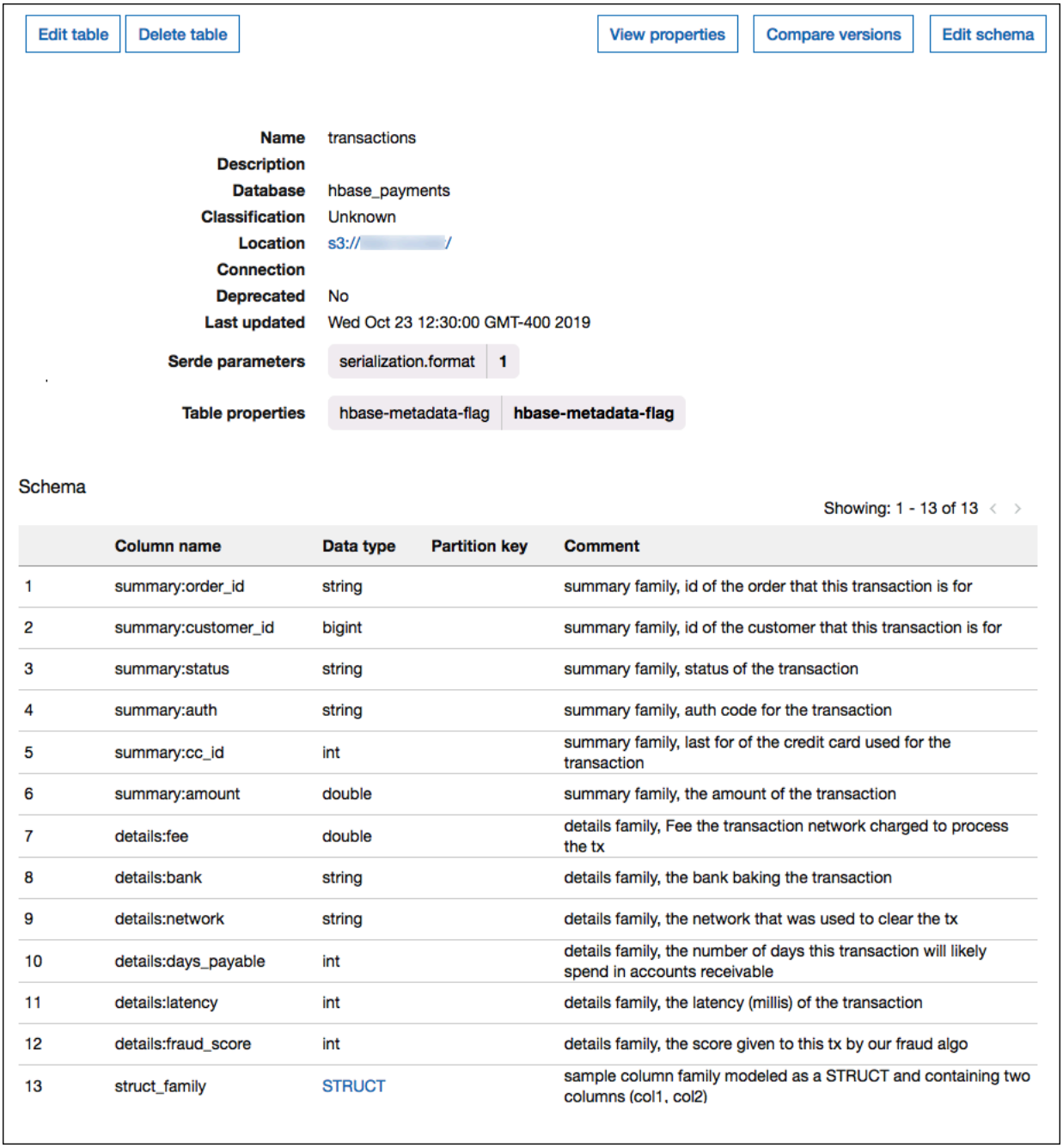
2. Assicurati di utilizzare i tipi di dati appropriati elencati AWS Glue in questo documento.

Modellazione delle famiglie di colonne

Il connettore HBase per Athena supporta due modi per modellare le famiglie di colonne HBase: denominazioni completamente qualificate (appiattite) come `family:column`, oppure l'utilizzo degli oggetti `STRUCT`.

Nel modello `STRUCT`, il nome del campo `STRUCT` dovrebbe corrispondere alla famiglia della colonna, mentre i figli di `STRUCT` dovrebbero corrispondere ai nomi delle colonne della famiglia. Tuttavia, poiché le letture colonnari e il pushdown dei predicati non sono ancora completamente supportati per tipi complessi come `STRUCT`, l'utilizzo di `STRUCT` al momento non è consigliato.

L'immagine seguente mostra una tabella configurata in AWS Glue che utilizza una combinazione dei due approcci.



	Column name	Data type	Partition key	Comment
1	summary:order_id	string		summary family, id of the order that this transaction is for
2	summary:customer_id	bigint		summary family, id of the customer that this transaction is for
3	summary:status	string		summary family, status of the transaction
4	summary:auth	string		summary family, auth code for the transaction
5	summary:cc_id	int		summary family, last for of the credit card used for the transaction
6	summary:amount	double		summary family, the amount of the transaction
7	details:fee	double		details family, Fee the transaction network charged to process the tx
8	details:bank	string		details family, the bank baking the transaction
9	details:network	string		details family, the network that was used to clear the tx
10	details:days_payable	int		details family, the number of days this transaction will likely spend in accounts receivable
11	details:latency	int		details family, the latency (millis) of the transaction
12	details:fraud_score	int		details family, the score given to this tx by our fraud algo
13	struct_family	STRUCT		sample column family modeled as a STRUCT and containing two columns (col1, col2)

Supporto dei tipi di dati

Il connettore recupera tutti i valori HBase come tipo di byte di base. Quindi, in base a come hai definito le tabelle in AWS Glue Data Catalog, mappa i valori in uno dei tipi di dati Apache Arrow nella tabella seguente.

AWS Glue tipo di dati	Tipo di dati Apache Arrow
int	INT
bigint	BIGINT
double	FLOAT8
float	FLOAT4
booleano	BIT
binary	VARBINARY
string	VARCHAR

> **Note**
>
> Se non si utilizza AWS Glue per integrare i metadati, l'inferenza dello schema del connettore utilizza solo i tipi di `BIGINT` dati e. `FLOAT8` `VARCHAR`

Autorizzazioni richieste

Consulta la sezione `Policies` del file athena-hbase.yaml per i dettagli completi delle policy IAM richieste da questo connettore. L'elenco che segue riporta un riepilogo delle autorizzazioni richieste.

- Accesso in scrittura ad Amazon S3: per trasferire i risultati di query di grandi dimensioni, il connettore richiede l'accesso in scrittura a una posizione in Amazon S3.
- Athena GetQueryExecution: il connettore utilizza questa autorizzazione per fallire rapidamente quando la query Athena upstream è terminata.
- AWS Glue Data Catalog— Il connettore HBase richiede l'accesso in sola lettura a per ottenere informazioni sullo schema. AWS Glue Data Catalog
- CloudWatch Registri: il connettore richiede l'accesso ai CloudWatch registri per l'archiviazione dei registri.
- AWS Secrets Manager accesso in lettura: se si sceglie di archiviare i dettagli degli endpoint HBase in Secrets Manager, è necessario concedere al connettore l'accesso a tali segreti.

- Accesso VPC: il connettore richiede la capacità di collegare e scollegare le interfacce al VPC in modo che possa connettersi ad esso e comunicare con le istanze HBase.

Prestazioni

Il connettore HBase per Athena tenta di eseguire in parallelo le query sull'istanza HBase leggendo ogni server regionale in parallelo. Il connettore Athena HBase esegue il pushdown del predicato per ridurre la quantità di dati scansionati dalla query.

La funzione Lambda esegue inoltre il pushdown delle proiezioni per ridurre la quantità di dati scansionati dalla query. Tuttavia, la selezione di un sottoinsieme di colonne a volte comporta un runtime delle query più lungo. Le clausole `LIMIT` riducono la quantità di dati scansionati, ma se non viene fornito un predicato, le query `SELECT` con una clausola `LIMIT` eseguiranno la scansione di almeno 16 MB di dati.

HBase è soggetto a errori di query e tempi di esecuzione delle query variabili. Potrebbe essere necessario ritentare più volte affinché le query abbiano esito positivo. Il connettore HBase è resiliente alla limitazione della larghezza di banda della rete dovuta alla simultaneità.

Interrogazioni pass-through

Il connettore HBase supporta le query passthrough ed è basato su NoSQL. Per informazioni sull'interrogazione di Apache HBase utilizzando il filtro, consulta Filter language nella documentazione di Apache.

Per utilizzare le query passthrough con HBase, utilizzate la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            database => 'database_name',
            collection => 'collection_name',
            filter => '{query_syntax}'
        ))
```

L'esempio seguente: filtri di query passthrough HBase per i dipendenti di 24 o 30 anni all'interno della raccolta del database. `employee` `default`

```
SELECT * FROM TABLE(
        system.query(
            DATABASE => 'default',
            COLLECTION => 'employee',
```

```
            FILTER => 'SingleColumnValueFilter(''personaldata'', ''age'', =,
''binary:30'')' ||
                      ' OR SingleColumnValueFilter(''personaldata'', ''age'', =,
''binary:24'')'
       ))
```

Informazioni sulla licenza

Il progetto del connettore HBase per Amazon Athena è concesso in licenza ai sensi della Licenza Apache-2.0.

Risorse aggiuntive

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su GitHub .com.

Connettore Amazon Athena per Hortonworks

Il connettore Amazon Athena per Hortonworks consente ad Amazon Athena di eseguire query SQL sulla piattaforma dati Cloudera Hortonworks. Il connettore trasforma le query SQL di Athena nella sintassi HiveQL equivalente.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .

Termini

I seguenti termini si riferiscono al connettore Hortonworks Hive.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.

- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore Hortonworks Hive.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
hive://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`HiveMuxCompositeHandler`

Gestore	Classe
Gestore dei metadati	`HiveMuxMetadataHandler`
Gestore dei record	`HiveMuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
`$`*`catalog`*`_connection_string`	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `myhivecatalog` , il nome della variabile di ambiente è `myhivecatalog_connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *`AWS_LAMBDA_FUNCTION_NAME`* `}`.

Le seguenti proprietà di esempio si riferiscono a una funzione Lambda Hive MUX che supporta due istanze del database: `hive1` (il valore predefinito) e `hive2`.

Proprietà	Valore
`default`	`hive://jdbc:hive2://hive1:10000/default?${Test/RDS/hive1}`
`hive_catalog1_connection_string`	`hive://jdbc:hive2://hive1:10000/default?${Test/RDS/hive1}`
`hive_catalog2_connection_string`	`hive://jdbc:hive2://hive2:10000/default?UID=sample&PWD=sample`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti codificati in AWS Secrets Manager, consulta Move i segreti codificati nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

  ```
  {"username": "${username}", "password": "${password}"}
  ```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${Test/RDS/hive1host}`.

```
hive://jdbc:hive2://hive1host:10000/default?...&${Test/RDS/hive1host}&...
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
hive://jdbc:hive2://hive1host:10000/default?...&UID=sample2&PWD=sample2&...
```

Attualmente, il connettore Hortonworks Hive riconosce le proprietà JDBC `UID` e `PWD`.

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza Hortonworks Hive.

Tipo di gestore	Classe
Gestore composito	`HiveCompositeHandler`
Gestore dei metadati	`HiveMetadataHandler`
Gestore dei record	`HiveRecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza Hortonworks Hive supportata da una funzione Lambda.

Proprietà	Valore
`default`	`hive://jdbc:hive2://hive1host:10000/default?secret=${Test/RDS/hive1host}`

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per JDBC, Hortonworks Hive e Arrow.

JDBC	Hortonworks Hive	Arrow
Booleano	Booleano	Bit
Numero intero	TINYINT	Tiny
Breve	SMALLINT	Smallint
Numero intero	INT	Int
Long	BIGINT	Bigint
float	float4	Float4
Doppio	float8	Float8

JDBC	Hortonworks Hive	Arrow
Data	date	DateDay
Timestamp	timestamp	DateMilli
Stringa	VARCHAR	Varchar
Byte	bytes	Varbinary
BigDecimal	Decimale	Decimale
ARRAY	N/D (vedi nota)	Elenco

> **Note**
>
> Attualmente, Hortonworks Hive non supporta i tipi di aggregati `ARRAY`, `MAP`, `STRUCT` e `UNIONTYPE`. Le colonne dei tipi di aggregati vengono trattate come colonne `VARCHAR` in SQL.

Partizioni e suddivisioni

Le partizioni vengono utilizzate per determinare come generare suddivisioni per il connettore. Athena costruisce una colonna sintetica di tipo `varchar` che rappresenta lo schema di partizionamento della tabella per aiutare il connettore a generare suddivisioni. Il connettore non modifica la definizione effettiva della tabella.

Prestazioni

Hortonworks Hive supporta le partizioni statiche. Il connettore Hortonworks Hive di Athena può recuperare dati da queste partizioni in parallelo. Se desideri interrogare set di dati molto grandi con una distribuzione uniforme delle partizioni, ti consigliamo vivamente il partizionamento statico. La selezione di un sottoinsieme di colonne velocizza notevolmente il runtime delle query e riduce i dati scansionati. Il connettore Hortonworks Hive è resiliente alla limitazione della larghezza di banda della rete dovuta alla simultaneità.

Il connettore Athena Hortonworks Hive esegue il pushdown dei predicati per ridurre i dati scansionati dalla query. Le clausole `LIMIT`, i predicati semplici e le espressioni complesse vengono inviate al connettore per ridurre la quantità di dati scansionati e ridurre il tempo di esecuzione delle query.

Clausole LIMIT

Una dichiarazione `LIMIT N` riduce la quantità di dati analizzati dalla query. Con il pushdown `LIMIT N`, il connettore restituisce solo le righe `N` ad Athena.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Athena Hortonworks Hive può combinare queste espressioni e inviarle direttamente a Hortonworks Hive per funzionalità avanzate e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore Athena Hortonworks Hive supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%')
LIMIT 10;
```

Interrogazioni pass-through

Il connettore Hortonworks Hive supporta le interrogazioni passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Hortonworks Hive, puoi utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a una fonte di dati in Hortonworks Hive. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consulta il file pom.xml per il connettore Hortonworks Hive su .com. GitHub

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su .com. GitHub

Connettore Apache Kafka di Amazon Athena

Il connettore Amazon Athena per Apache Kafka consente ad Amazon Athena di eseguire query SQL sui tuoi argomenti di Apache Kafka. Utilizza questo connettore per visualizzare gli argomenti di Apache Kafka come tabelle e i messaggi come righe in Athena.

Prerequisiti

Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.

- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .
- Nelle condizioni di filtro, è necessario impostare i tipi di dati date e timestamp sul tipo di dati appropriato.
- I tipi di dati data e ora non sono supportati per il tipo di file CSV e vengono trattati come valori varchar.
- La mappatura in campi JSON annidati non è supportata. Il connettore mappa solo i campi di primo livello.
- Il connettore non supporta tipi complessi. I tipi complessi vengono interpretati come stringhe.
- Per estrarre o lavorare con valori JSON complessi, utilizza le funzioni relative a JSON disponibili in Athena. Per ulteriori informazioni, consulta Estrazione di dati JSON dalle stringhe.
- Il connettore non supporta l'accesso ai metadati dei messaggi Kafka.

Termini

- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Endpoint Kafka: una stringa di testo che stabilisce una connessione a un'istanza Kafka.

Compatibilità dei cluster

Il connettore Kafka può essere utilizzato con i seguenti tipi di cluster.

- Kafka standalone: una connessione diretta a Kafka (autenticata o non autenticata).
- Confluent: una connessione diretta a Confluent Kafka. Per informazioni sull'utilizzo di Athena con i dati di Confluent Kafka, consulta Visualizza i dati Confluent in Amazon usando QuickSight Amazon Athena nel blog di Business Intelligence.AWS

Connessione a Confluent

Per la connessione a Confluent sono necessari i seguenti passaggi:

1. Genera una chiave API da Confluent.

2. Memorizza il nome utente e la password per la chiave API di Confluent in AWS Secrets Manager.
3. Fornisci il nome del segreto per la variabile di ambiente `secrets_manager_secret` nel connettore Kafka.
4. Segui la procedura nella sezione Configurazione del connettore Kafka di questo documento.

Metodi di autenticazione di supportati

Il connettore supporta i seguenti metodi di autenticazione.

- SSL
- SASL/SCRAM
- SASL/PLAIN
- SASL/PLAINTEXT
- NO_AUTH
- Kafka e Confluent Platform autogestiti: SSL, SASL/SCRAM, SASL/PLAINTEXT, NO_AUTH
- Kafka e Confluent Cloud autogestiti: SASL/PLAIN

Per ulteriori informazioni, consulta Configurazione dell'autenticazione del connettore Kafka di Athena.

Formati di dati di input supportati

Il connettore supporta i seguenti formati di dati di input.

- JSON
- CSV

Parametri

Utilizza le variabili di ambiente Lambda menzionate in questa sezione per configurare il connettore Kafka di Athena.

- auth_type: specifica il tipo di autenticazione del cluster. Il connettore supporta i seguenti tipi di autenticazione:
  - NO_AUTH: connettiti direttamente a Kafka (ad esempio, a un cluster Kafka implementato su un'istanza EC2 che non utilizza l'autenticazione).

- SASL_SSL_PLAIN: questo metodo utilizza il protocollo di sicurezza `SASL_SSL` e il meccanismo SASL `PLAIN`. Per ulteriori informazioni, consulta Configurazione SASL nella documentazione Kafka di Apache.
- SASL_PLAINTEXT_PLAIN: questo metodo utilizza il protocollo di sicurezza `SASL_PLAINTEXT` e il meccanismo SASL `PLAIN`. Per ulteriori informazioni, consulta Configurazione SASL nella documentazione Kafka di Apache.
- SASL_SSL_SCRAM_SHA512: puoi utilizzare questo tipo di autenticazione per controllare l'accesso ai cluster Amazon Kafka. Questo metodo memorizza il nome utente e la password in. AWS Secrets Manager Il segreto deve essere associato al cluster Kafka. Per ulteriori informazioni, consulta Autenticazione tramite SASL/SCRAM nella documentazione di Apache Kafka.
- SASL_PLAINTEXT_SCRAM_SHA512: questo metodo utilizza il protocollo di sicurezza `SASL_PLAINTEXT` e il meccanismo `SCRAM_SHA512 SASL`. Questo metodo utilizza il nome utente e la password memorizzati in AWS Secrets Manager. Per ulteriori informazioni, consulta la sezione Configurazione SASL nella documentazione Apache Kafka.
- SSL: l'autenticazione SSL utilizza i file archivio delle chiavi e di archivio di trust per connettersi al cluster Apache Kafka. Devi generare i file del trust store e del key store, caricarli in un bucket Amazon S3 e fornire il riferimento ad Amazon S3 quando implementi il connettore. Il key store, il trust store e la chiave SSL sono archiviati in AWS Secrets Manager. Il client deve fornire la chiave AWS segreta quando il connettore viene distribuito. Per ulteriori informazioni, consulta Crittografia e autenticazione tramite SSL nella documentazione di Apache Kafka.

  Per ulteriori informazioni, consulta Configurazione dell'autenticazione del connettore Kafka di Athena.

- certificates_s3_reference: la posizione Amazon S3 che contiene i certificati (i file key store e trust store).
- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.
- kafka_endpoint: i dettagli dell'endpoint da fornire a Kafka.
- secrets_manager_secret: il nome del segreto AWS in cui vengono salvate le credenziali.
- Parametri di spill: le funzioni Lambda archiviano temporaneamente ("riversano") i dati che non rientrano nella memoria di Amazon S3. Tutte le istanze del database a cui accede la stessa

funzione Lambda riversano i dati nella stessa posizione. Utilizza i parametri nella tabella seguente per specificare la posizione di spill.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Il nome del bucket Amazon S3 in cui la funzione Lambda può riversare i dati.
`spill_prefix`	Obbligatorio. Il prefisso all'interno del bucket di spill in cui la funzione Lambda può riversare dati.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

- ID di sottoreti: uno o più ID di sottorete corrispondenti alla sottorete che la funzione Lambda può utilizzare per accedere all'origine dati.
  - Cluster di Kafka pubblico o cluster standard di Confluent Cloud: associa il connettore a una sottorete privata dotata di un gateway NAT.
  - Cluster Confluent Cloud con connettività privata: associa il connettore a una sottorete privata che abbia un percorso diretto al cluster Confluent Cloud.
    - Per AWS Transit Gateway, le sottoreti devono trovarsi in un VPC collegato allo stesso gateway di transito utilizzato da Confluent Cloud.
    - Per il Peering VPC, le sottoreti devono trovarsi in un VPC in peering con Confluent Cloud VPC.
    - Per AWS PrivateLink, le sottoreti devono trovarsi in un VPC con un percorso verso gli endpoint VPC che si connettono a Confluent Cloud.

> **Note**
>
> Se si implementa il connettore in un VPC per accedere a risorse private e si desidera connettersi anche a un servizio accessibile al pubblico come Confluent, è necessario

associare il connettore a una sottorete privata con un gateway NAT. Per ulteriori informazioni, consulta Gateway NAT nella Guida per l'utente di Amazon VPC.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti supportati per Kafka e Apache Arrow.

Kafka	Arrow
CHAR	VARCHAR
VARCHAR	VARCHAR
TIMESTAMP	MILLISECOND
DATE	GIORNO
BOOLEAN	BOOL
SMALLINT	SMALLINT
INTEGER	INT
BIGINT	BIGINT
DECIMAL	FLOAT8
DOUBLE	FLOAT8

Partizioni e suddivisioni

Gli argomenti di Kafka sono suddivisi in partizioni. Ogni partizione è ordinata. Ogni messaggio in una partizione ha un ID incrementale chiamato offset. Ogni partizione Kafka è ulteriormente suddivisa in più suddivisioni per l'elaborazione in parallelo. I dati sono disponibili per il periodo di conservazione configurato nei cluster Kafka.

Best practice

Come procedura consigliata, utilizza il pushdown del predicato quando esegui query su Athena, come negli esempi seguenti.

```
SELECT *
FROM "kafka_catalog_name"."glue_schema_registry_name"."glue_schema_name"
WHERE integercol = 2147483647
```

```
SELECT *
FROM "kafka_catalog_name"."glue_schema_registry_name"."glue_schema_name"
WHERE timestampcol >= TIMESTAMP '2018-03-25 07:30:58.878'
```

Configurazione del connettore Kafka

Prima di poter utilizzare il connettore, è necessario configurare il cluster Apache Kafka, utilizzare il registro degli schemi di AWS Glue per definire lo schema e configurare l'autenticazione del connettore.

Quando lavori con lo AWS Glue Schema Registry, tieni presente i seguenti punti:

- Assicurati che il testo nel campo Description (Descrizione) del registro degli schemi di AWS Glue includa la stringa `{AthenaFederationKafka}`. Questa stringa di marker è obbligatoria per i AWS Glue registri che usi con il connettore Amazon Athena Kafka.
- Per prestazioni ottimali, utilizza solo lettere minuscole per i nomi dei database e delle tabelle. L'utilizzo di caratteri misti tra maiuscole e minuscole fa sì che il connettore esegua una ricerca senza distinzione tra maiuscole e minuscole, più impegnativa dal punto di vista computazionale.

Per configurare l'ambiente Apache Kafka e lo Schema Registry AWS Glue

1. Configurazione dell'ambiente Apache Kafka.
2. Carica il file di descrizione dell'argomento di Kafka (ovvero il relativo schema) in formato JSON nel registro degli schemi. AWS Glue Per ulteriori informazioni, consulta Integrating with AWS Glue Schema Registry nella Developer Guide. AWS Glue Per schemi di esempio, consulta la sezione seguente.

Esempi di schemi per il registro degli schemi di AWS Glue

Utilizza il formato degli esempi in questa sezione quando carichi lo schema nel registro degli schemi di AWS Glue.

Esempio di schema di tipo JSON

Nell'esempio seguente, lo schema da creare nel registro degli AWS Glue schemi viene specificato `json` come valore `dataFormat` e da utilizzare `datatypejson` per. `topicName`

> **Note**
>
> Il valore di `topicName` deve utilizzare le stesse maiuscole e minuscole del nome dell'argomento in Kafka.

```
{
  "topicName": "datatypejson",
  "message": {
    "dataFormat": "json",
    "fields": [
      {
        "name": "intcol",
        "mapping": "intcol",
        "type": "INTEGER"
      },
      {
        "name": "varcharcol",
        "mapping": "varcharcol",
        "type": "VARCHAR"
      },
      {
        "name": "booleancol",
        "mapping": "booleancol",
        "type": "BOOLEAN"
      },
      {
        "name": "bigintcol",
        "mapping": "bigintcol",
        "type": "BIGINT"
      },
      {
```

```
        "name": "doublecol",
        "mapping": "doublecol",
        "type": "DOUBLE"
      },
      {
        "name": "smallintcol",
        "mapping": "smallintcol",
        "type": "SMALLINT"
      },
      {
        "name": "tinyintcol",
        "mapping": "tinyintcol",
        "type": "TINYINT"
      },
      {
        "name": "datecol",
        "mapping": "datecol",
        "type": "DATE",
        "formatHint": "yyyy-MM-dd"
      },
      {
        "name": "timestampcol",
        "mapping": "timestampcol",
        "type": "TIMESTAMP",
        "formatHint": "yyyy-MM-dd HH:mm:ss.SSS"
      }
    ]
  }
}
```

Esempio di schema di tipo CSV

Nell'esempio seguente, lo schema da creare nel registro degli AWS Glue schemi viene specificato `csv` come valore `dataFormat` e da utilizzare `datatypecsvbulk` per. `topicName` Il valore di `topicName` deve utilizzare le stesse maiuscole e minuscole del nome dell'argomento in Kafka.

```
{
  "topicName": "datatypecsvbulk",
  "message": {
    "dataFormat": "csv",
    "fields": [
      {
        "name": "intcol",
```

```
        "type": "INTEGER",
        "mapping": "0"
      },
      {
        "name": "varcharcol",
        "type": "VARCHAR",
        "mapping": "1"
      },
      {
        "name": "booleancol",
        "type": "BOOLEAN",
        "mapping": "2"
      },
      {
        "name": "bigintcol",
        "type": "BIGINT",
        "mapping": "3"
      },
      {
        "name": "doublecol",
        "type": "DOUBLE",
        "mapping": "4"
      },
      {
        "name": "smallintcol",
        "type": "SMALLINT",
        "mapping": "5"
      },
      {
        "name": "tinyintcol",
        "type": "TINYINT",
        "mapping": "6"
      },
      {
        "name": "floatcol",
        "type": "DOUBLE",
        "mapping": "7"
      }
    ]
  }
}
```

Configurazione dell'autenticazione del connettore Kafka di Athena

Puoi utilizzare diversi metodi per l'autenticazione al cluster Apache Kafka, tra cui SSL, SASL/SCRAM, SASL/PLAIN e SASL/PLAINTEXT.

La tabella seguente mostra i tipi di autenticazione per il connettore, il protocollo di sicurezza e il meccanismo SASL per ciascuno di questi metodi. Per ulteriori informazioni, consulta la sezione Sicurezza nella documentazione di Apache Kafka.

auth_type	security.protocol	sasl.mechanism	Compatibilità con tipo di cluster
`SASL_SSL_PLAIN`	`SASL_SSL`	`PLAIN`	• Kafka autogestito • Confluent Platform • Confluent Cloud
`SASL_PLAINTEXT_PLAIN`	`SASL_PLAINTEXT`	`PLAIN`	• Kafka autogestito • Confluent Platform
`SASL_SSL_SCRAM_SHA512`	`SASL_SSL`	`SCRAM-SHA-512`	• Kafka autogestito • Confluent Platform
`SASL_PLAINTEXT_SCRAM_SHA512`	`SASL_PLAINTEXT`	`SCRAM-SHA-512`	• Kafka autogestito • Confluent Platform
`SSL`	`SSL`	N/D	• Kafka autogestito • Confluent Platform

SSL

Se il cluster è autenticato tramite SSL, devi generare i file del trust store e del key store e caricarli nel bucket Amazon S3. È necessario fornire questo riferimento Amazon S3 quando implementi il connettore. Il key store, il trust store e la chiave SSL sono archiviati in AWS Secrets Manager. La chiave AWS segreta viene fornita quando si distribuisce il connettore.

Per informazioni sulla creazione di un segreto in Secrets Manager, consulta la pagina Crea un segreto AWS Secrets Manager.

Per utilizzare questo tipo di autenticazione, è necessario impostare le variabili di ambiente come illustrato nella tabella seguente.

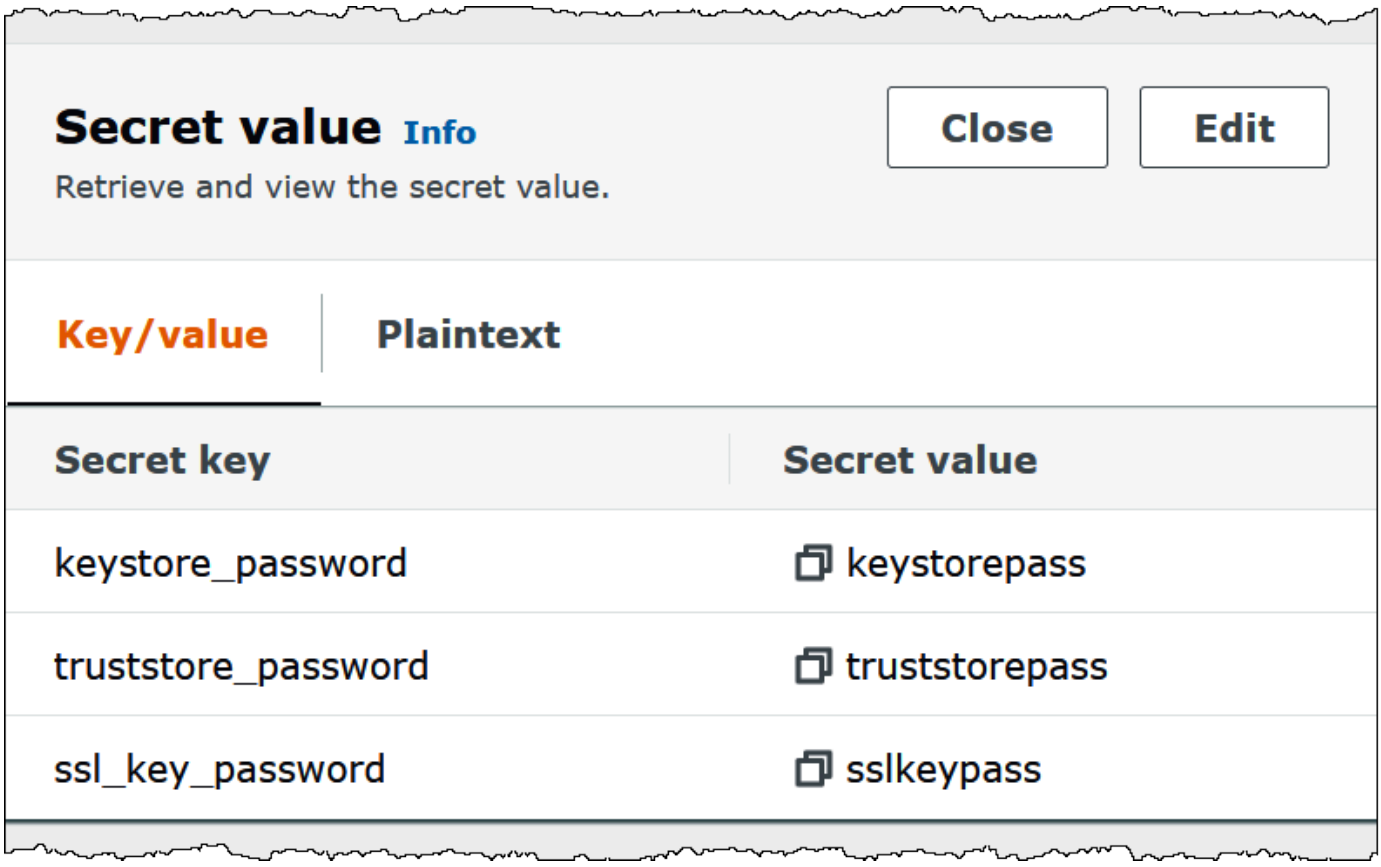
Parametro	Valore
`auth_type`	`SSL`
`certificates_s3_reference`	La posizione di Amazon S3 contenente i certificati.
`secrets_manager_secret`	Il nome della tua chiave AWS segreta.

Dopo avere creato un segreto in Gestione dei segreti, puoi visualizzarlo nella console Gestione dei segreti.

Visualizzazione del segreto in Gestione dei segreti

1. Apri la console di Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Nel riquadro di navigazione, scegli Secrets (Segreti).
3. Nella pagina Secrets (Segreti), scegli il collegamento al tuo segreto.
4. Nella pagina dei dettagli del segreto, scegli Retrieve secret value (Recupera il valore di un segreto).

   L'immagine seguente mostra un esempio di segreto con tre coppie chiave/valore: `keystore_password`, `truststore_password` e `ssl_key_password`.



Per ulteriori informazioni sull'utilizzo di SSL con Kafka, consulta Crittografia e autenticazione tramite SSL nella documentazione di Apache Kafka.

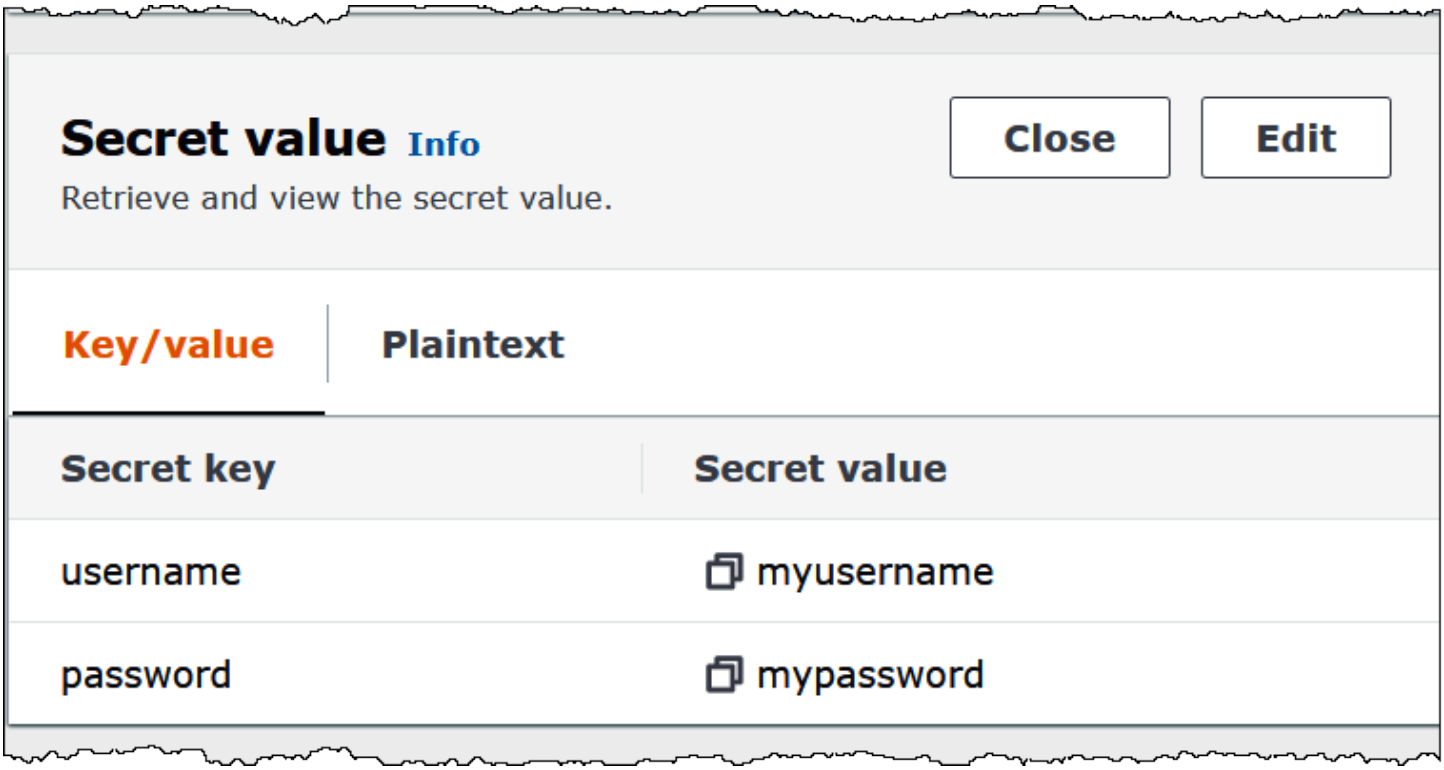
SASL/SCRAM

Se il cluster utilizza l'autenticazione SCRAM, fornisci la chiave Gestione dei segreti associata al cluster quando implementi il connettore. Le credenziali AWS dell'utente (chiave segreta e chiave di accesso) vengono utilizzate per autenticarsi al cluster.

Imposta le variabili di ambiente come illustrato nella tabella seguente.

Parametro	Valore
`auth_type`	`SASL_SSL_SCRAM_SHA512`
`secrets_manager_secret`	Il nome della tua chiave AWS segreta.

L'immagine seguente mostra un esempio di segreto nella console Gestione dei segreti con due coppie chiave/valore: una per `username` e una per `password`.



Per ulteriori informazioni sull'utilizzo di SASL/SCRAM con Kafka, consulta Autenticazione tramite SASL/SCRAM nella documentazione di Apache Kafka.

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su GitHub .com.

Risorse aggiuntive

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su GitHub .com.

Connettore MSK di Amazon Athena

Il connettore Amazon Athena per Amazon MSK consente ad Amazon Athena di eseguire query SQL sui tuoi argomenti di Apache Kafka. Utilizza questo connettore per visualizzare gli argomenti di Apache Kafka come tabelle e i messaggi come righe in Athena. Per ulteriori informazioni, consulta Analizzare i dati di streaming in tempo reale in Amazon MSK con Amazon Athena AWS nel blog Big Data.

Prerequisiti

Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .
- Nelle condizioni di filtro, è necessario impostare i tipi di dati date e timestamp sul tipo di dati appropriato.
- I tipi di dati data e ora non sono supportati per il tipo di file CSV e vengono trattati come valori varchar.
- La mappatura in campi JSON annidati non è supportata. Il connettore mappa solo i campi di primo livello.
- Il connettore non supporta tipi complessi. I tipi complessi vengono interpretati come stringhe.
- Per estrarre o lavorare con valori JSON complessi, utilizza le funzioni relative a JSON disponibili in Athena. Per ulteriori informazioni, consulta Estrazione di dati JSON dalle stringhe.
- Il connettore non supporta l'accesso ai metadati dei messaggi Kafka.

Termini

- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Endpoint Kafka: una stringa di testo che stabilisce una connessione a un'istanza Kafka.

Compatibilità dei cluster

Il connettore MSK può essere utilizzato con i seguenti tipi di cluster.

- Cluster fornito da MSK: la capacità del cluster viene specificata, monitorata e dimensionata manualmente.

- Cluster MSK serverless: fornisce capacità on demand che si adatta automaticamente al dimensionamento degli I/O delle applicazioni.
- Kafka standalone: una connessione diretta a Kafka (autenticata o non autenticata).

Metodi di autenticazione di supportati

Il connettore supporta i seguenti metodi di autenticazione.

- SASL/IAM
- SSL
- SASL/SCRAM
- SASL/PLAIN
- SASL/PLAINTEXT
- NO_AUTH

  Per ulteriori informazioni, consulta Configurazione dell'autenticazione per il connettore Athena MSK.

Formati di dati di input supportati

Il connettore supporta i seguenti formati di dati di input.

- JSON
- CSV

Parametri

Utilizza le variabili di ambiente Lambda menzionate in questa sezione per configurare il connettore Athena MSK.

- auth_type: specifica il tipo di autenticazione del cluster. Il connettore supporta i seguenti tipi di autenticazione:
  - NO_AUTH: connettiti direttamente a Kafka senza autenticazione (ad esempio, a un cluster Kafka implementato su un'istanza EC2 che non utilizza l'autenticazione).
  - SASL_SSL_PLAIN: questo metodo utilizza il protocollo di sicurezza `SASL_SSL` e il meccanismo SASL `PLAIN`.

- SASL_PLAINTEXT_PLAIN: questo metodo utilizza il protocollo di sicurezza `SASL_PLAINTEXT` e il meccanismo SASL `PLAIN`.

  > **Note**
  >
  > I tipi di autenticazione `SASL_SSL_PLAIN` e `SASL_PLAINTEXT_PLAIN` sono supportati da Apache Kafka ma non da Amazon MSK.

- SASL_SSL_AWS_MSK_IAM: il controllo degli accessi IAM per Amazon MSK ti consente di gestire sia l'autenticazione sia l'autorizzazione per il cluster MSK. AWS Le credenziali dell'utente (chiave segreta e chiave di accesso) vengono utilizzate per connettersi al cluster. Per ulteriori informazioni, consulta la pagina IAM access control (Controllo degli accessi IAM) nella Guida per gli sviluppatori di Amazon Managed Streaming per Apache Kafka.
- SASL_SSL_SCRAM_SHA512: puoi utilizzare questo tipo di autenticazione per controllare l'accesso ai cluster Amazon MSK. Questo metodo memorizza il nome utente e la password su AWS Secrets Manager. Il segreto deve essere associato al cluster Amazon MSK. Per ulteriori informazioni, consulta la pagina Setting up SASL/SCRAM authentication for an Amazon MSK cluster (Configurazione dell'autenticazione SASL/SCRAM per un cluster Amazon MSK) nella Guida per gli sviluppatori di Amazon Managed Streaming per Apache Kafka.
- SSL: l'autenticazione SSL utilizza i file di key store e trust store per connettersi al cluster Amazon MSK. Devi generare i file del trust store e del key store, caricarli in un bucket Amazon S3 e fornire il riferimento ad Amazon S3 quando implementi il connettore. Il key store, il trust store e la chiave SSL sono archiviati in AWS Secrets Manager. Il client deve fornire la chiave AWS segreta quando il connettore viene distribuito. Per ulteriori informazioni, consulta la pagina Mutual TLS authentication (Autenticazione TLS reciproca) nella Guida per gli sviluppatori di Amazon Managed Streaming per Apache Kafka.

  Per ulteriori informazioni, consulta Configurazione dell'autenticazione per il connettore Athena MSK.

- certificates_s3_reference: la posizione Amazon S3 che contiene i certificati (i file key store e trust store).
- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.

- kafka_endpoint: i dettagli dell'endpoint da fornire a Kafka. Ad esempio, per un cluster Amazon MSK, fornisci un URL di bootstrap per il cluster.
- secrets_manager_secret: il nome del segreto AWS in cui vengono salvate le credenziali. Questo parametro non è obbligatorio per l'autenticazione IAM.
- Parametri di spill: le funzioni Lambda archiviano temporaneamente ("riversano") i dati che non rientrano nella memoria di Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione. Utilizza i parametri nella tabella seguente per specificare la posizione di spill.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Il nome del bucket Amazon S3 in cui la funzione Lambda può riversare i dati.
`spill_prefix`	Obbligatorio. Il prefisso all'interno del bucket di spill in cui la funzione Lambda può riversare dati.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti supportati per Kafka e Apache Arrow.

Kafka	Arrow
CHAR	VARCHAR
VARCHAR	VARCHAR
TIMESTAMP	MILLISECOND

Kafka	Arrow
DATE	GIORNO
BOOLEAN	BOOL
SMALLINT	SMALLINT
INTEGER	INT
BIGINT	BIGINT
DECIMAL	FLOAT8
DOUBLE	FLOAT8

Partizioni e suddivisioni

Gli argomenti di Kafka sono suddivisi in partizioni. Ogni partizione è ordinata. Ogni messaggio in una partizione ha un ID incrementale chiamato offset. Ogni partizione Kafka è ulteriormente suddivisa in più suddivisioni per l'elaborazione in parallelo. I dati sono disponibili per il periodo di conservazione configurato nei cluster Kafka.

Best practice

Come procedura consigliata, utilizza il pushdown del predicato quando esegui query su Athena, come negli esempi seguenti.

```
SELECT *
FROM "msk_catalog_name"."glue_schema_registry_name"."glue_schema_name"
WHERE integercol = 2147483647
```

```
SELECT *
FROM "msk_catalog_name"."glue_schema_registry_name"."glue_schema_name"
WHERE timestampcol >= TIMESTAMP '2018-03-25 07:30:58.878'
```

Configurazione del connettore MSK

Prima di poter utilizzare il connettore, è necessario configurare il cluster Amazon MSK, utilizzare il registro degli schemi di AWS Glue per definire lo schema e configurare l'autenticazione per il connettore.

> **Note**
>
> Se si implementa il connettore in un VPC per accedere a risorse private e si desidera connettersi anche a un servizio accessibile al pubblico come Confluent, è necessario associare il connettore a una sottorete privata con un gateway NAT. Per ulteriori informazioni, consulta Gateway NAT nella Guida per l'utente di Amazon VPC.

Quando lavori con lo AWS Glue Schema Registry, tieni presente i seguenti punti:

- Assicurati che il testo nel campo Description (Descrizione) del registro degli schemi di AWS Glue includa la stringa `{AthenaFederationMSK}`. Questa stringa di marker è obbligatoria per AWS Glue i registri utilizzati con il connettore Amazon Athena MSK.
- Per prestazioni ottimali, utilizza solo lettere minuscole per i nomi dei database e delle tabelle. L'utilizzo di caratteri misti tra maiuscole e minuscole fa sì che il connettore esegua una ricerca senza distinzione tra maiuscole e minuscole, più impegnativa dal punto di vista computazionale.

Per configurare l'ambiente Amazon MSK e il registro AWS Glue degli schemi

1. Configura il tuo ambiente Amazon MSK. Per ulteriori informazioni e per conoscere la procedura, consulta le sezioni Setting up Amazon MSK (Configurazione di Amazon MSK) e Getting started using Amazon MSK (Nozioni di base sull'utilizzo di Amazon MSK) nella Guida per gli sviluppatori di Amazon Managed Streaming for Apache Kafka.
2. Carica il file di descrizione dell'argomento di Kafka (ovvero il relativo schema) in formato JSON nel registro degli schemi. AWS Glue Per ulteriori informazioni, consulta Integrating with AWS Glue Schema Registry nella Developer Guide. AWS Glue Per schemi di esempio, consulta la sezione seguente.

Esempi di schemi per il registro degli schemi di AWS Glue

Utilizza il formato degli esempi in questa sezione quando carichi lo schema nel registro degli schemi di AWS Glue.

Esempio di schema di tipo JSON

Nell'esempio seguente, lo schema da creare nel registro degli AWS Glue schemi viene specificato `json` come valore `dataFormat` e da utilizzare `datatypejson` per. `topicName`

> **Note**
>
> Il valore di `topicName` deve utilizzare le stesse maiuscole e minuscole del nome dell'argomento in Kafka.

```
{
  "topicName": "datatypejson",
  "message": {
    "dataFormat": "json",
    "fields": [
      {
        "name": "intcol",
        "mapping": "intcol",
        "type": "INTEGER"
      },
      {
        "name": "varcharcol",
        "mapping": "varcharcol",
        "type": "VARCHAR"
      },
      {
        "name": "booleancol",
        "mapping": "booleancol",
        "type": "BOOLEAN"
      },
      {
        "name": "bigintcol",
        "mapping": "bigintcol",
        "type": "BIGINT"
      },
      {
        "name": "doublecol",
        "mapping": "doublecol",
        "type": "DOUBLE"
      },
      {
```

```
        "name": "smallintcol",
        "mapping": "smallintcol",
        "type": "SMALLINT"
      },
      {
        "name": "tinyintcol",
        "mapping": "tinyintcol",
        "type": "TINYINT"
      },
      {
        "name": "datecol",
        "mapping": "datecol",
        "type": "DATE",
        "formatHint": "yyyy-MM-dd"
      },
      {
        "name": "timestampcol",
        "mapping": "timestampcol",
        "type": "TIMESTAMP",
        "formatHint": "yyyy-MM-dd HH:mm:ss.SSS"
      }
    ]
  }
}
```

Esempio di schema di tipo CSV

Nell'esempio seguente, lo schema da creare nel registro degli AWS Glue schemi viene specificato `csv` come valore `dataFormat` e da utilizzare `datatypecsvbulk` per. `topicName` Il valore di `topicName` deve utilizzare le stesse maiuscole e minuscole del nome dell'argomento in Kafka.

```
{
  "topicName": "datatypecsvbulk",
  "message": {
    "dataFormat": "csv",
    "fields": [
      {
        "name": "intcol",
        "type": "INTEGER",
        "mapping": "0"
      },
      {
        "name": "varcharcol",
```

```
        "type": "VARCHAR",
        "mapping": "1"
      },
      {
        "name": "booleancol",
        "type": "BOOLEAN",
        "mapping": "2"
      },
      {
        "name": "bigintcol",
        "type": "BIGINT",
        "mapping": "3"
      },
      {
        "name": "doublecol",
        "type": "DOUBLE",
        "mapping": "4"
      },
      {
        "name": "smallintcol",
        "type": "SMALLINT",
        "mapping": "5"
      },
      {
        "name": "tinyintcol",
        "type": "TINYINT",
        "mapping": "6"
      },
      {
        "name": "floatcol",
        "type": "DOUBLE",
        "mapping": "7"
      }
    ]
  }
}
```

Configurazione dell'autenticazione per il connettore Athena MSK

Puoi utilizzare diversi metodi per l'autenticazione al cluster Amazon MSK, tra cui IAM, SSL, SCRAM e Kafka standalone.

La tabella seguente mostra i tipi di autenticazione per il connettore, il protocollo di sicurezza e il meccanismo SASL per ciascuno di questi metodi. Per ulteriori informazioni, consulta la pagina Authentication and authorization for Apache Kafka APIs (Autenticazione e autorizzazione per le API Apache Kafka) nella Guida per gli sviluppatori di Amazon Managed Streaming per Apache Kafka.

auth_type	security.protocol	sasl.mechanism
SASL_SSL_PLAIN	SASL_SSL	PLAIN
SASL_PLAINTEXT_PLAIN	SASL_PLAINTEXT	PLAIN
SASL_SSL_AWS_MSK_IAM	SASL_SSL	AWS_MSK_IAM
SASL_SSL_SCRAM_SHA512	SASL_SSL	SCRAM-SHA-512
SSL	SSL	N/D

> **Note**
>
> I tipi di autenticazione `SASL_SSL_PLAIN` e `SASL_PLAINTEXT_PLAIN` sono supportati da Apache Kafka ma non da Amazon MSK.

SASL/IAM

Se il cluster utilizza l'autenticazione IAM, è necessario configurare la policy IAM per l'utente al momento della configurazione del cluster. Per ulteriori informazioni, consulta la pagina IAM access control (Controllo degli accessi IAM) nella Guida per gli sviluppatori di Amazon Managed Streaming per Apache Kafka.

Per utilizzare questo tipo di autenticazione, imposta la variabile di ambiente Lambda `auth_type` per il connettore su `SASL_SSL_AWS_MSK_IAM`.

SSL

Se il cluster è autenticato tramite SSL, devi generare i file del trust store e del key store e caricarli nel bucket Amazon S3. È necessario fornire questo riferimento Amazon S3 quando implementi il connettore. Il key store, il trust store e la chiave SSL sono archiviati in AWS Secrets Manager. La chiave AWS segreta viene fornita quando si distribuisce il connettore.

Per informazioni sulla creazione di un segreto in Secrets Manager, consulta la pagina Crea un segreto AWS Secrets Manager.

Per utilizzare questo tipo di autenticazione, è necessario impostare le variabili di ambiente come illustrato nella tabella seguente.

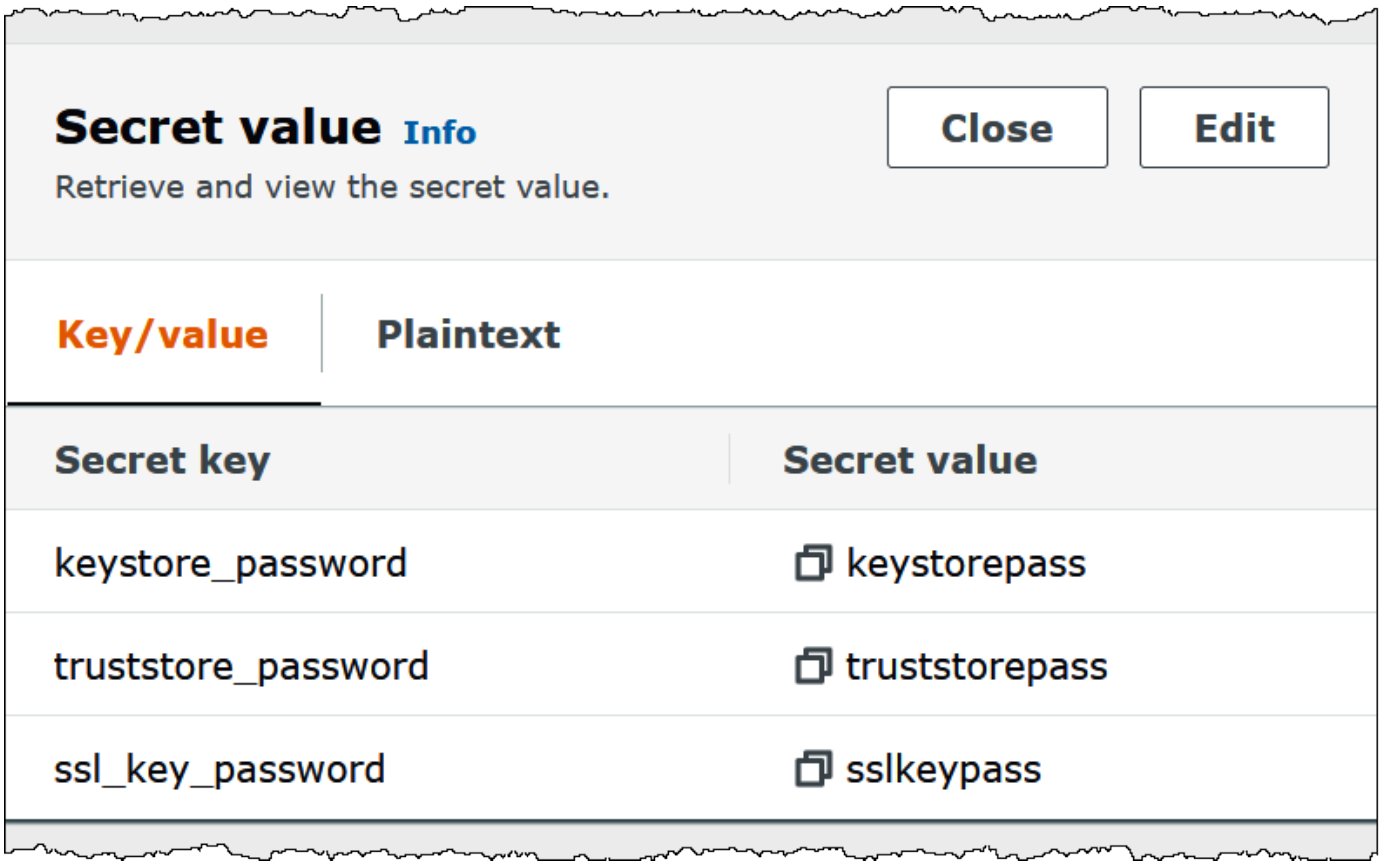
Parametro	Valore
`auth_type`	`SSL`
`certificates_s3_reference`	La posizione di Amazon S3 contenente i certificati.
`secrets_manager_secret`	Il nome della tua chiave AWS segreta.

Dopo avere creato un segreto in Gestione dei segreti, puoi visualizzarlo nella console Gestione dei segreti.

Visualizzazione del segreto in Gestione dei segreti

1. Apri la console di Secrets Manager all'indirizzo https://console.aws.amazon.com/secretsmanager/.
2. Nel riquadro di navigazione, scegli Secrets (Segreti).
3. Nella pagina Secrets (Segreti), scegli il collegamento al tuo segreto.
4. Nella pagina dei dettagli del segreto, scegli Retrieve secret value (Recupera il valore di un segreto).

   L'immagine seguente mostra un esempio di segreto con tre coppie chiave/valore: `keystore_password`, `truststore_password` e `ssl_key_password`.



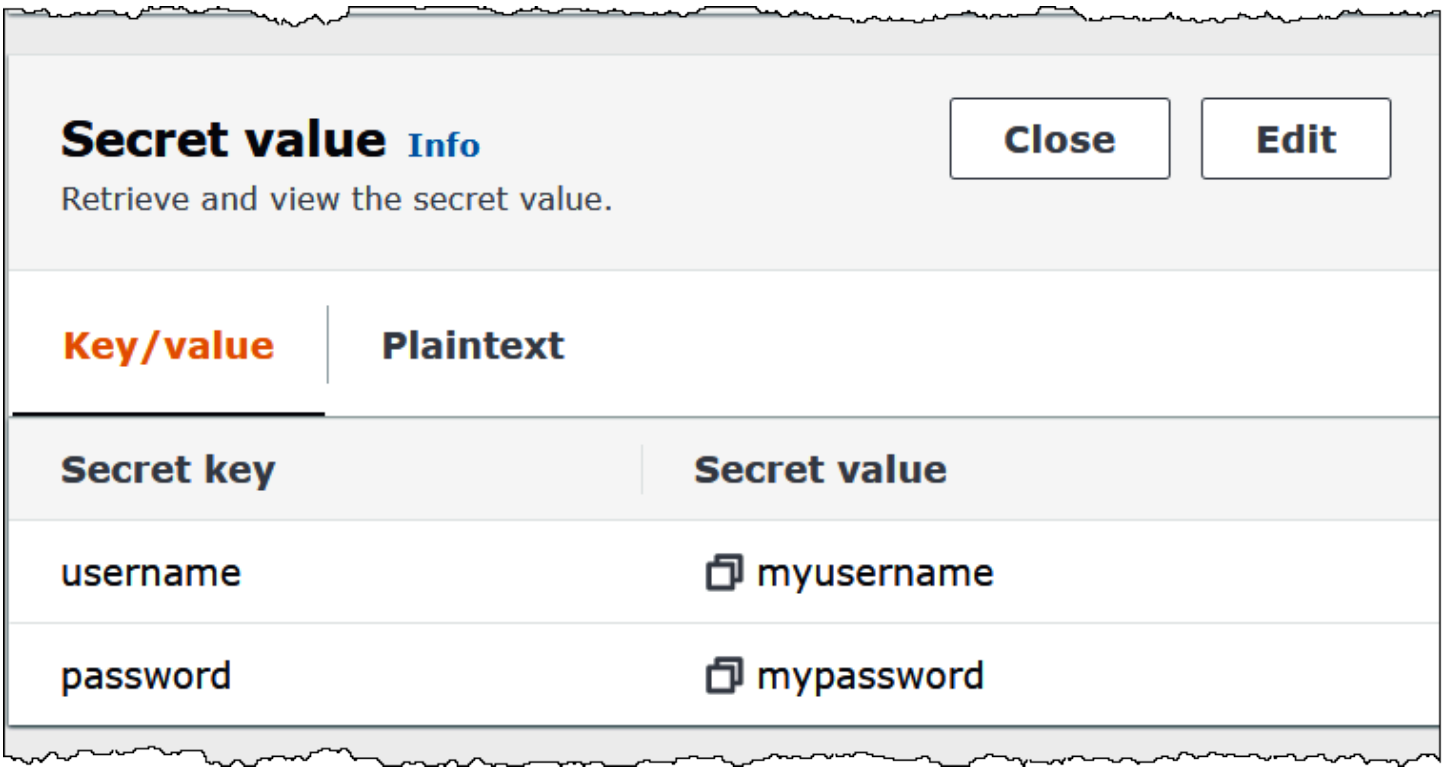
SASL/SCRAM

Se il cluster utilizza l'autenticazione SCRAM, fornisci la chiave Gestione dei segreti associata al cluster quando implementi il connettore. Le credenziali AWS dell'utente (chiave segreta e chiave di accesso) vengono utilizzate per autenticarsi al cluster.

Imposta le variabili di ambiente come illustrato nella tabella seguente.

Parametro	Valore
`auth_type`	`SASL_SSL_SCRAM_SHA512`
`secrets_manager_secret`	Il nome della tua chiave AWS segreta.

L'immagine seguente mostra un esempio di segreto nella console Gestione dei segreti con due coppie chiave/valore: una per `username` e una per `password`.



Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su GitHub .com.

Risorse aggiuntive

Per ulteriori informazioni su questo connettore, visita il sito corrispondente su GitHub .com.

Connettore MySQL di Amazon Athena

Il connettore Lambda MySQL di Amazon Athena consente ad Amazon Athena di accedere ai database MySQL.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.
- Prima di utilizzare questo connettore, configura un VPC e un gruppo di sicurezza. Per ulteriori informazioni, consulta Creazione di un VPC per un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .
- Poiché Athena converte le query in lettere minuscole, i nomi delle tabelle MySQL devono essere in minuscolo. Ad esempio, le query di Athena su una tabella denominata `myTable` avranno esito negativo.

Termini

I seguenti termini si riferiscono al connettore MySQL.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore MySQL.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
mysql://${jdbc_connection_string}
```

> **Note**
>
> Se ricevi l'errore java.sql.SQLException: Zero date value prohibited (java.sql.SQLException: Valore data zero proibito) quando esegui una query `SELECT` su una tabella MySQL, aggiungi il seguente parametro alla stringa di connessione:
>
> ```
> zeroDateTimeBehavior=convertToNull
> ```
>
> Per ulteriori informazioni, vedi Errore «Valore di data zero proibito» durante il tentativo di selezione dalla tabella MySQL su .com. GitHub

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`MySqlMuxCompositeHandler`
Gestore dei metadati	`MySqlMuxMetadataHandler`
Gestore dei record	`MySqlMuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
$*catalog*_connection_string	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è mymysqlcatalog , il nome della variabile di ambiente è mymysqlcatalog_connection_string .
default	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è lambda:${ *AWS_LAMBDA_FUNCTION_NAME* }.

Le seguenti proprietà di esempio riguardano una funzione Lambda MySql MUX che supporta due istanze di databasemysql1: (impostazione predefinita) e. mysql2

Proprietà	Valore
default	mysql://jdbc:mysql://mysql2.host:3333/default? user=sample2&password=sample2
mysql_catalog1_connection_string	mysql://jdbc:mysql://mysql1.host:3306/default?${Test/RDS/MySql1}
mysql_catalog2_connection_string	mysql://jdbc:mysql://mysql2.host:3333/default? user=sample2&password=sample2

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti hardcoded in AWS Secrets Manager, consultate Move hardcoded secret in nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

  ```
  {"username": "${username}", "password": "${password}"}
  ```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${Test/RDS/MySql1}`.

```
mysql://jdbc:mysql://mysql1.host:3306/default?...&${Test/RDS/MySql1}&...
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
mysql://jdbc:mysql://mysql1host:3306/default?...&user=sample2&password=sample2&...
```

Attualmente, il connettore MySQL riconosce le proprietà JDBC `user` e `password`.

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza MySQL.

Tipo di gestore	Classe
Gestore composito	`MySqlCompositeHandler`
Gestore dei metadati	`MySqlMetadataHandler`
Gestore dei record	`MySqlRecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza MySQL supportata da una funzione Lambda.

Proprietà	Valore
`default`	`mysql://mysql1.host:3306/default?secret=Test/RDS/` MySql1

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per JDBC e Arrow.

JDBC	Arrow
Booleano	Bit
Numero intero	Tiny
Breve	Smallint
Numero intero	Int
Long	Bigint
float	Float4
Doppio	Float8
Data	DateDay
Timestamp	DateMilli
Stringa	Varchar

JDBC	Arrow
Byte	Varbinary
BigDecimal	Decimale
ARRAY	Elenco

Partizioni e suddivisioni

Le partizioni vengono utilizzate per determinare come generare suddivisioni per il connettore. Athena costruisce una colonna sintetica di tipo `varchar` che rappresenta lo schema di partizionamento della tabella per aiutare il connettore a generare suddivisioni. Il connettore non modifica la definizione effettiva della tabella.

Prestazioni

MySQL supporta le partizioni native. Il connettore MySQL di Athena può recuperare dati da queste partizioni in parallelo. Se desideri interrogare set di dati molto grandi con una distribuzione uniforme delle partizioni, ti consigliamo vivamente il partizionamento nativo.

Il connettore Athena MySQL esegue il pushdown dei predicati per ridurre i dati analizzati dalla query. Le clausole `LIMIT`, i predicati semplici e le espressioni complesse vengono inviati al connettore per ridurre la quantità di dati analizzati e per ridurre il tempo di esecuzione delle query.

Clausole LIMIT

Una dichiarazione `LIMIT N` riduce la quantità di dati analizzati dalla query. Con il pushdown `LIMIT N`, il connettore restituisce solo le righe `N` ad Athena.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Athena MySQL può combinare queste espressioni e inviarle direttamente a MySQL per migliorare le funzionalità e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore Athena MySQL supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT

- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_DISTINCT_FROM, NULL_IF, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%')
LIMIT 10;
```

Per un articolo sull'utilizzo del pushdown del predicato per migliorare le prestazioni nelle query federate, incluso MySQL, consulta Come migliorare le query federate con il pushdown del predicato in Amazon Athena nel Blog sui Big Data di AWS .

Interrogazioni pass-through

Il connettore MySQL supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con MySQL, è possibile utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a un'origine dati in MySQL. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
```

```
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consulta il file pom.xml per il connettore MySQL su .com. GitHub

Per ulteriori informazioni su questo connettore, visita il sito corrispondente su .com. GitHub

Connettore Amazon Athena Neptune

Amazon Neptune è un servizio di database a grafo gestito rapido e affidabile che rende più semplice la creazione e l'esecuzione di applicazioni che funzionano con set di dati altamente connessi. Il motore di database a grafo dedicato e a prestazioni elevate di Neptune archivia miliardi di relazioni e invia query al grafo con una latenza di millisecondi. Per ulteriori informazioni, consulta la Guida per l'utente di Neptune.

Il connettore Amazon Athena Neptune consente ad Athena di comunicare con l'istanza del database a grafo Neptune, rendendo i dati del grafico Neptune accessibili tramite query SQL.

Se hai abilitato Lake Formation nel tuo account, il ruolo IAM per il tuo connettore Lambda federato Athena che hai distribuito nell'accesso in lettura deve avere accesso in lettura in AWS Serverless Application Repository Lake Formation a. AWS Glue Data Catalog

Prerequisiti

L'utilizzo del connettore Neptune richiede i seguenti tre passaggi.

- Configurazione di un cluster Neptune
- Configurare un AWS Glue Data Catalog
- Implementazione del connettore sul tuo Account AWS. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati. Per ulteriori dettagli specifici sulla distribuzione del connettore Neptune, consulta Distribuire il connettore Amazon Athena Neptune su .com. GitHub

Limitazioni

Attualmente, il Neptune Connector presenta le seguenti limitazioni.

- La proiezione di colonne, inclusa la chiave primaria (ID), non è supportata.

Configurazione di un cluster Neptune

Se non disponi di un cluster Amazon Neptune esistente e di un set di dati con grafici delle proprietà che desideri utilizzare, devi configurarne uno.

Assicurati che siano disponibili un gateway Internet e un gateway NAT nel VPC che ospita il cluster Neptune. Le sottoreti private utilizzate dalla funzione Lambda del connettore Neptune devono avere un instradamento verso Internet tramite questo gateway NAT. La funzione Lambda del connettore Neptune utilizza il gateway NAT per comunicare. AWS Glue

Per istruzioni su come configurare un nuovo cluster Neptune e caricarlo con un set di dati di esempio, consulta Sample Neptune Cluster Setup su .com. GitHub

Configurazione di un AWS Glue Data Catalog

A differenza dei tradizionali archivi di dati relazionali, gli edge e i nodi DB dei grafici di Neptune non utilizzano uno schema prestabilito. Ogni voce può avere campi e tipi di dati diversi. Tuttavia, poiché il connettore Neptune recupera i metadati da AWS Glue Data Catalog, è necessario creare AWS Glue un database con tabelle con lo schema richiesto. Dopo aver creato il database e le tabelle AWS Glue , il connettore può compilare l'elenco delle tabelle disponibili per le query da Athena.

Come abilitare la corrispondenza delle colonne senza distinzione tra maiuscole

Per risolvere i nomi delle colonne della tabella Neptune con l'uso corretto delle maiuscole e delle minuscole anche quando i nomi delle colonne sono tutti AWS Glue in minuscolo, puoi configurare il connettore Neptune per la corrispondenza senza distinzione tra maiuscole e minuscole.

Per abilitare questa funzionalità, imposta la variabile di ambiente `enable_caseinsensitivematch` su `true` nella funzione Lambda del connettore Neptune.

Specificazione del parametro glabel per i nomi delle tabelle con maiuscole e minuscole AWS Glue

Poiché AWS Glue supporta solo nomi di tabella minuscoli, è importante specificare il parametro `glabel` AWS Glue table quando si crea una tabella per Neptune e il nome della AWS Glue tabella Neptune include il maiuscolo.

Nella definizione della AWS Glue tabella, includete il `glabel` parametro e impostate il suo valore sul nome della tabella con l'involucro originale. Ciò garantisce che l'involucro corretto venga preservato quando AWS Glue interagisce con il tavolo Neptune. Nell'esempio seguente il valore di `glabel` è impostato sul nome della tabella `Airport`.

```
glabel = Airport
```

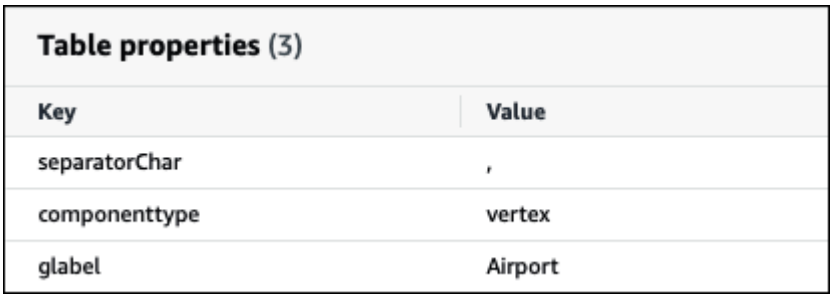


Table properties (3)

Key	Value
separatorChar	,
componenttype	vertex
glabel	Airport

Per ulteriori informazioni sulla configurazione di un dispositivo AWS Glue Data Catalog per l'utilizzo con Neptune, consulta AWS Glue Configurare il catalogo su .com. GitHub

Prestazioni

Il connettore Athena Neptune esegue il pushdown del predicato per ridurre la quantità di dati scansionati dalla query. Tuttavia, i predicati che utilizzano la chiave primaria comportano un errore di query. Le clausole `LIMIT` riducono la quantità di dati scansionati, ma se non viene fornito un predicato, le query `SELECT` con una clausola `LIMIT` eseguiranno la scansione di almeno 16 MB di dati. Il connettore Neptune è resiliente alla limitazione della larghezza di banda della rete dovuta alla simultaneità.

Interrogazioni pass-through

Il connettore Neptune supporta le query passthrough. È possibile utilizzare questa funzionalità per eseguire query Gremlin sui grafici delle proprietà e per eseguire query SPARQL sui dati RDF.

Per creare query passthrough con Neptune, utilizzate la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            DATABASE => 'database_name',
            COLLECTION => 'collection_name',
            QUERY => 'query_string'
        ))
```

L'esempio seguente: filtri di interrogazione passthrough di Neptune per gli aeroporti con il codice. ATL Le virgolette singole raddoppiate servono per scappare.

```
SELECT * FROM TABLE(
        system.query(
            DATABASE => 'graph-database',
            COLLECTION => 'airport',
            QUERY => 'g.V().has(''airport'', ''code'', ''ATL'').valueMap()'
        ))
```

Risorse aggiuntive

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su GitHub .com.

Connettore Amazon Athena OpenSearch

OpenSearch Servizio

Il OpenSearch connettore Amazon Athena consente ad Amazon Athena di comunicare con le OpenSearch tue istanze in modo da poter utilizzare SQL per interrogare i tuoi dati. OpenSearch

> **Note**
>
> A causa di un problema noto, il OpenSearch connettore non può essere utilizzato con un VPC.

Se hai abilitato Lake Formation nel tuo account, il ruolo IAM per il tuo connettore Lambda federato Athena che hai distribuito nell'accesso in lettura deve avere accesso in lettura in AWS Serverless Application Repository Lake Formation a. AWS Glue Data Catalog

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Termini

I seguenti termini si riferiscono al connettore. OpenSearch

- Dominio: un nome che questo connettore associa all'endpoint dell'istanza. OpenSearch Il dominio viene utilizzato anche come nome del database. Per OpenSearch le istanze definite all'interno del OpenSearch servizio Amazon, il dominio è individuabile automaticamente. Per altre istanze, devi fornire una mappatura tra il nome di dominio e l'endpoint.
- Indice: una tabella di database definita nell'istanza. OpenSearch
- Mappatura: se un indice è una tabella del database, la mappatura è il relativo schema (ossia le definizioni dei campi e degli attributi).

  Questo connettore supporta sia il recupero dei metadati dall' OpenSearch istanza che da. AWS Glue Data Catalog Se il connettore trova un AWS Glue database e una tabella che corrispondono ai nomi di OpenSearch dominio e di indice, tenta di utilizzarli per la definizione dello schema. Ti consigliamo di creare la AWS Glue tabella in modo che sia un superset di tutti i campi dell' OpenSearch indice.
- Documento: un record all'interno di una tabella del database.
- Flusso di dati: dati basati su tempo composti da più indici di supporto. Per ulteriori informazioni, consulta Data stream nella OpenSearch documentazione e Introduzione ai flussi di dati nella Amazon OpenSearch Service Developer Guide.

> **Note**
>
> Poiché gli indici dei flussi di dati sono creati e gestiti internamente da open search, il connettore sceglie dal primo indice disponibile la mappatura dello schema. Per questo motivo, consigliamo vivamente di configurare una AWS Glue tabella come fonte di metadati supplementare. Per ulteriori informazioni, consulta Configurazione di database e tabelle in AWS Glue.

Parametri

Usa le variabili di ambiente Lambda in questa sezione per configurare il OpenSearch connettore.

- spill_bucket: specifica il bucket Amazon S3 per i dati che superano i limiti della funzione Lambda.
- spill_prefix: (facoltativo) per impostazione predefinita, viene utilizzata una sottocartella nello `spill_bucket` specificato chiamata `athena-federation-spill`. Ti consigliamo di configurare un ciclo di vita dell'archiviazione di Amazon S3 in questa posizione per eliminare gli spill più vecchi di un numero predeterminato di giorni o ore.

- spill_put_request_headers: (facoltativo) una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.
- kms_key_id: (facoltativo) per impostazione predefinita, tutti i dati riversati in Amazon S3 vengono crittografati utilizzando la modalità di crittografia autenticata AES-GCM e una chiave generata casualmente. Per fare in modo che la tua funzione Lambda utilizzi chiavi di crittografia più potenti generate da KMS come `a7e63k4b-8loc-40db-a2a1-4d0en2cd8331`, puoi specificare l'ID della chiave KMS.
- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.
- disable_glue — (Facoltativo) Se presente e impostato su true, il connettore non tenta di recuperare metadati supplementari da. AWS Glue
- query_timeout_cluster: il periodo di timeout, in secondi, per le query sullo stato del cluster utilizzate nella generazione di scansioni in parallelo.
- query_timeout_search: il periodo di timeout, in secondi, per le query di ricerca utilizzate per il recupero di documenti da un indice.
- auto_discover_endpoint: un valore booleano. Il valore predefinito è `true`. Quando utilizzi il OpenSearch servizio Amazon e imposti questo parametro su true, il connettore può scoprire automaticamente i tuoi domini ed endpoint chiamando le operazioni API appropriate per descrivere o elencare sul Servizio. OpenSearch Per qualsiasi altro tipo di OpenSearch istanza (ad esempio, self-hosted), devi specificare gli endpoint di dominio associati nella variabile. `domain_mapping` Se`auto_discover_endpoint=true`, il connettore utilizza AWS le credenziali per l'autenticazione al Servizio. OpenSearch In caso contrario, il connettore recupera le credenziali del nome utente e della password tramite la variabile. AWS Secrets Manager `domain_mapping`
- domain_mapping: viene utilizzato solo quando `auto_discover_endpoint` è impostato su false e definisce la mappatura tra i nomi di dominio e gli endpoint associati. La `domain_mapping` variabile può contenere più OpenSearch endpoint nel seguente formato:

```
domain1=endpoint1,domain2=endpoint2,domain3=endpoint3,...
```

Ai fini dell'autenticazione su un OpenSearch endpoint, il connettore supporta stringhe sostitutive inserite utilizzando il formato `${SecretName}:` con nome utente e password recuperati da. AWS Secrets Manager I due punti (:) alla fine dell'espressione fungono da separatore dal resto dell'endpoint.

> **Important**
>
> Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti hardcoded in, consultate Move hardcoded secret in nella Guida per l' AWS Secrets Manager utente. AWS Secrets ManagerAWS Secrets Manager

Nell'esempio seguente viene utilizzato il segreto `opensearch-creds`.

```
movies=https://${opensearch-creds}:search-movies-ne...qu.us-east-1.es.amazonaws.com
```

In fase di runtime, `${opensearch-creds}` viene visualizzato come nome utente e password, come nell'esempio seguente.

```
movies=https://myusername@mypassword:search-movies-ne...qu.us-east-1.es.amazonaws.com
```

Nel parametro `domain_mapping`, ogni coppia dominio-endpoint può utilizzare un segreto diverso. Il segreto stesso deve essere specificato nel formato *`nome_utente`*@*`password`*. Sebbene la password possa contenere segni `@`, il primo `@` funge da separatore da *`user_name`* (nome utente).

È anche importante tenere a mente che questo connettore utilizza la virgola (,) e il segno uguale (=) come separatori per le coppie dominio-endpoint. Per questo motivo, non è consigliabile utilizzarli da nessuna parte all'interno del secreto archiviato.

Configurazione di database e tabelle in AWS Glue

Il connettore ottiene le informazioni sui metadati utilizzando AWS Glue o. OpenSearch È possibile impostare una AWS Glue tabella come fonte di definizione dei metadati supplementare. Per abilitare questa funzionalità, definisci un AWS Glue database e una tabella che corrispondano al dominio e

all'indice della fonte che stai integrando. Il connettore può inoltre sfruttare le definizioni dei metadati memorizzate nell' OpenSearch istanza recuperando la mappatura per l'indice specificato.

Definizione dei metadati per gli array in OpenSearch

OpenSearch non dispone di un tipo di dati di array dedicato. Qualsiasi campo può contenere zero o più valori purché siano dello stesso tipo di dati. Se si desidera utilizzare OpenSearch come fonte di definizione dei metadati, è necessario definire una `_meta` proprietà per tutti gli indici utilizzati con Athena per i campi che devono essere considerati un elenco o un array. Se non completi questo passaggio, le query restituiscono solo il primo elemento nel campo elenco. Quando specifichi la proprietà `_meta`, i nomi dei campi devono essere completamente qualificati per le strutture JSON nidificate (ad esempio, `address.street`, dove `street` è un campo nidificato all'interno di una struttura `address`).

L'esempio seguente definisce gli elenchi `actor` e `genre` nella tabella `movies`.

```
PUT movies/_mapping
{
  "_meta": {
    "actor": "list",
    "genre": "list"
  }
}
```

Tipi di dati

Il OpenSearch connettore può estrarre le definizioni dei metadati da entrambe le istanze o dall'istanza AWS Glue . OpenSearch Il connettore utilizza la mappatura illustrata nella seguente tabella per convertire le definizioni in tipi di dati Apache Arrow, inclusi i punti indicati nella sezione seguente.

OpenSearch	Apache Arrow	AWS Glue
text, keyword, binary	VARCHAR	string
Long	BIGINT	bigint
scaled_float	BIGINT	SCALED_FLOAT(...)
integer	INT	int

OpenSearch	Apache Arrow	AWS Glue
short	SMALLINT	smallint
byte	TINYINT	tinyint
double	FLOAT8	double
float, half_float	FLOAT4	float
booleano	BIT	booleano
date, date_nanos	DATEMILLI	timestamp
Struttura JSON	STRUCT	STRUCT
_meta (per informazioni, consulta la sezione Definizione dei metadati per gli array in OpenSearch.)	LIST	ARRAY

Note sui tipi di dati

- Attualmente, il connettore supporta solo OpenSearch i AWS Glue tipi di dati elencati nella tabella precedente.
- Un `scaled_float` è un numero a virgola mobile scalato in base a un fattore di scala fisso doppio e rappresentato come `BIGINT` in Apache Arrow. Ad esempio, 0,756 con un fattore di scala pari a 100 viene arrotondato a 76.
- *Per definire un scaled_float in AWS Glue, è necessario selezionare il tipo di array colonna e dichiarare il campo utilizzando il formato SCALED_FLOAT (scaling_factor).*

  Gli esempi seguenti sono validi:

```
SCALED_FLOAT(10.51)
SCALED_FLOAT(100)
SCALED_FLOAT(100.0)
```

Gli esempi seguenti non sono validi:

```
SCALED_FLOAT(10.)
SCALED_FLOAT(.5)
```

- Durante la conversione da `date_nanos` a `DATEMILLI`, i nanosecondi vengono arrotondati al millisecondo più vicino. I valori validi per `date` e `date_nanos` includono, senza limitazione, i seguenti formati:

```
"2020-05-18T10:15:30.123456789"
"2020-05-15T06:50:01.123Z"
"2020-05-15T06:49:30.123-05:00"
1589525370001 (epoch milliseconds)
```

- An OpenSearch `binary` è una rappresentazione in formato stringa di un valore binario codificato utilizzando e convertito in un. `Base64` `VARCHAR`

Esecuzione di query SQL

Di seguito sono riportati alcuni esempi di query DDL che puoi utilizzare con questo connettore. Negli esempi, *`function_name`* corrisponde al nome della funzione Lambda, *`domain`* è il nome del dominio su cui desideri eseguire una query e *`index`* è il nome del tuo indice.

```
SHOW DATABASES in `lambda:function_name`
```

```
SHOW TABLES in `lambda:function_name`.domain
```

```
DESCRIBE `lambda:function_name`.domain.index
```

Prestazioni

Il OpenSearch connettore Athena supporta scansioni parallele basate su shard. Il connettore utilizza le informazioni sullo stato del cluster recuperate dall' OpenSearch istanza per generare più richieste per una query di ricerca di documenti. Le richieste vengono suddivise per ogni partizione ed eseguite in simultanea.

Il connettore inoltre esegue il pushdown dei predicati come parte delle sue query di ricerca di documenti. I seguenti query e predicato di esempio mostrano come il connettore utilizza il pushdown del predicato.

Query

```
SELECT * FROM "lambda:elasticsearch".movies.movies
WHERE year >= 1955 AND year <= 1962 OR year = 1996
```

Predicate

```
(_exists_:year) AND year:([1955 TO 1962] OR 1996)
```

Interrogazioni pass-through

Il OpenSearch connettore supporta le query passthrough e utilizza il linguaggio Query DSL. Per ulteriori informazioni sull'esecuzione di query con Query DSL, consulta Query DSL nella documentazione di Elasticsearch o Query DSL nella documentazione. OpenSearch

Per utilizzare le query passthrough con il connettore, utilizzate la seguente sintassi: OpenSearch

```
SELECT * FROM TABLE(
        system.query(
            schema => 'schema_name',
            index => 'index_name',
            query => "{query_string}"
        ))
```

Il seguente OpenSearch esempio di filtri di interrogazione passthrough per i dipendenti con status occupazionale attivo nell'`employee`indice dello schema. `default`

```
SELECT * FROM TABLE(
        system.query(
            schema => 'default',
            index => 'employee',
            query => "{ ''bool'':{''filter'':{''term'':{''status'': ''active''}}}}"
        ))
```

Risorse aggiuntive

- Per un articolo sull'utilizzo del OpenSearch connettore Amazon Athena per interrogare i dati in Amazon OpenSearch Service e Amazon S3 in un'unica query, consulta Interrogare i dati in Amazon Service usando SQL di OpenSearch Amazon Athena nel Big Data Blog.AWS
- Per ulteriori informazioni su questo connettore, visita il sito corrispondente su .com. GitHub

Connettore Amazon Athena per Oracle

Il connettore Amazon Athena per Oracle consente ad Amazon Athena di eseguire query SQL sui dati archiviati in Oracle in esecuzione on-premise oppure su Amazon EC2 o Amazon RDS. Puoi anche utilizzare il connettore per eseguire query sui dati Oracle Exadata.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .

Termini

I seguenti termini si riferiscono al connettore Oracle.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.

- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore Oracle.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
oracle://${jdbc_connection_string}
```

> **Note**
>
> Se la password contiene caratteri speciali (ad esempio `some.password`), racchiudi la password tra virgolette doppie quando la passi alla stringa di connessione (ad esempio `"some.password"`). In caso contrario, è possibile che venga generato come errore URL Oracle specificato non valido.

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`OracleMuxCompositeHandler`
Gestore dei metadati	`OracleMuxMetadataHandler`
Gestore dei record	`OracleMuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
`$`*`catalog`*`_connection_string`	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `myoraclecatalog` , il nome della variabile di ambiente è `myoraclecatalog_connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *`AWS_LAMBDA_FUNCTION_NAME`* `}`.

Le seguenti proprietà di esempio si riferiscono a una funzione Lambda Oracle MUX che supporta due istanze del database: `oracle1` (il valore predefinito) e `oracle2`.

Proprietà	Valore
`default`	`oracle://jdbc:oracle:thin:${Test/RDS/Oracle1}@//oracle1.hostname:port/servicename`
`oracle_catalog1_connection_string`	`oracle://jdbc:oracle:thin:${Test/RDS/Oracle1}@//oracle1.hostname:port/servicename`

Proprietà	Valore
`oracle_catalog2_connection_string`	`oracle://jdbc:oracle:thin:${Test/RDS/Oracle2}@//oracle2.hostname:port/servicename`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti codificati in AWS Secrets Manager, consulta Move i segreti codificati nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

  ```
  {"username": "${username}", "password": "${password}"}
  ```

> **Note**
>
> Se la password contiene caratteri speciali (ad esempio `some.password`), racchiudi la password tra virgolette doppie quando la memorizzi in Secrets Manager (ad esempio `"some.password"`). In caso contrario, è possibile che venga generato come errore URL Oracle specificato non valido.

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${Test/RDS/Oracle}`.

```
oracle://jdbc:oracle:thin:${Test/RDS/Oracle}@//hostname:port/servicename
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
oracle://jdbc:oracle:thin:username/password@//hostname:port/servicename
```

Attualmente, il connettore Oracle riconosce le proprietà JDBC `UID` e `PWD`.

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza Oracle.

Tipo di gestore	Classe
Gestore composito	`OracleCompositeHandler`
Gestore dei metadati	`OracleMetadataHandler`
Gestore dei record	`OracleRecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.
`IsFIPSEnabled`	Facoltativo. Imposta su `true` quando è abilitata la modalità FIPS. Il valore predefinito è `false`.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

Il connettore supporta connessioni basate su SSL alle istanze Amazon RDS. Il supporto è limitato al protocollo Transport Layer Security (TLS) e all'autenticazione del server da parte del client. L'autenticazione reciproca non è supportata in Amazon RDS. La seconda riga della tabella seguente mostra la sintassi per l'utilizzo di SSL.

La seguente proprietà di esempio si riferisce a una singola istanza Oracle supportata da una funzione Lambda.

Proprietà	Valore
`default`	`oracle://jdbc:oracle:thin:${Test/RDS/Oracle}@//hostname:port/servicename`
	`oracle://jdbc:oracle:thin:${Test/RDS/Oracle}@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCPS) (HOST=<HOST_NAME>)(PORT=))(CONNECT_DATA=(SID=))(SECURITY=(SSL_SERVER_CERT_DN=)))`

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per JDBC, Oracle e Arrow.

JDBC	Oracle	Arrow
Booleano	booleano	Bit
Numero intero	N/D	Tiny
Breve	smallint	Smallint
Numero intero	integer	Int
Long	bigint	Bigint
float	float4	Float4
Doppio	float8	Float8
Data	date	DateDay
Timestamp	timestamp	DateMilli
Stringa	text	Varchar

JDBC	Oracle	Arrow
Byte	bytes	Varbinary
BigDecimal	numeric(p,s)	Decimale
ARRAY	N/D (vedi nota)	Elenco

Partizioni e suddivisioni

Le partizioni vengono utilizzate per determinare come generare suddivisioni per il connettore. Athena costruisce una colonna sintetica di tipo `varchar` che rappresenta lo schema di partizionamento della tabella per aiutare il connettore a generare suddivisioni. Il connettore non modifica la definizione effettiva della tabella.

Prestazioni

Oracle supporta le partizioni native. Il connettore Oracle di Athena può recuperare dati da queste partizioni in parallelo. Se desideri interrogare set di dati molto grandi con una distribuzione uniforme delle partizioni, ti consigliamo vivamente il partizionamento nativo. La selezione di un sottoinsieme di colonne velocizza notevolmente il runtime delle query e riduce i dati scansionati. Il connettore Oracle è resiliente alla limitazione della larghezza di banda della rete dovuta alla simultaneità. Tuttavia, i tempi di esecuzione delle query tendono a essere lunghi.

Il connettore Athena Oracle esegue il pushdown del predicato per ridurre la quantità di dati scansionati dalla query. I predicati semplici e le espressioni complesse vengono inviati al connettore per ridurre la quantità di dati scansionati e ridurre il tempo di esecuzione delle query.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Oracle di Athena può combinare queste espressioni e inviarle direttamente a Oracle per funzionalità avanzate e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore Oracle di Athena supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_NULL

- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%');
```

Interrogazioni pass-through

Il connettore Oracle supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare la query completa alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Oracle, puoi utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a un'origine dati in Oracle. La query seleziona tutte le colonne della `customer` tabella.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su GitHub .com.

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consulta il file pom.xml per il connettore Oracle su .com. GitHub

Per ulteriori informazioni su questo connettore, visita il sito corrispondente su GitHub .com.

Connettore PostgreSQL di Amazon Athena

Il connettore PostgreSQL per Amazon Athena consente ad Amazon Athena di accedere ai database PostgreSQL.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .

Termini

I seguenti termini si riferiscono al connettore PostgreSQL.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.

- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore PostgreSQL.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
postgres://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`PostGreSqlMuxCompositeHandler`
Gestore dei metadati	`PostGreSqlMuxMetadataHandler`
Gestore dei record	`PostGreSqlMuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
$*catalog*_connection_string	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `mypostgrescatalog` , il nome della variabile di ambiente è `mypostgrescatalog_connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *AWS_LAMBDA_FUNCTION_NAME* `}`.

Le seguenti proprietà di esempio riguardano una funzione Lambda PostGreSql MUX che supporta due istanze di database`postgres1`: (impostazione predefinita) e. `postgres2`

Proprietà	Valore
`default`	`postgres://jdbc:postgresql://postgres1.host:5432/default?${Test/RDS/PostGres1}`
`postgres_catalog1_connection_string`	`postgres://jdbc:postgresql://postgres1.host:5432/default?${Test/RDS/PostGres1}`
`postgres_catalog2_connection_string`	`postgres://jdbc:postgresql://postgres2.host:5432/default?user=sample&password=sample`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti hardcoded in AWS Secrets Manager, consultate Move hardcoded secret in nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

  ```
  {"username": "${username}", "password": "${password}"}
  ```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${Test/RDS/PostGres1}`.

```
postgres://jdbc:postgresql://postgres1.host:5432/default?...&${Test/RDS/PostGres1}&...
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
postgres://jdbc:postgresql://postgres1.host:5432/
default?...&user=sample2&password=sample2&...
```

Attualmente, il connettore PostgreSQL riconosce le proprietà JDBC `user` e `password`.

Abilitazione SSL

Per supportare SSL nella tua connessione PostgreSQL, aggiungi alla stringa di connessione quanto segue:

```
&sslmode=verify-ca&sslfactory=org.postgresql.ssl.DefaultJavaSSLFactory
```

Esempio

La stringa di connessione dell'esempio seguente non utilizza SSL.

```
postgres://jdbc:postgresql://example-asdf-aurora-postgres-endpoint:5432/asdf?
user=someuser&password=somepassword
```

Per abilitare SSL, modifica la stringa come segue.

```
postgres://jdbc:postgresql://example-asdf-aurora-postgres-
endpoint:5432/asdf?user=someuser&password=somepassword&sslmode=verify-
ca&sslfactory=org.postgresql.ssl.DefaultJavaSSLFactory
```

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza PostgreSQL.

Tipo di gestore	Classe
Gestore composito	`PostGreSqlCompositeHandler`
Gestore dei metadati	`PostGreSqlMetadataHandler`
Gestore dei record	`PostGreSqlRecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza PostgreSQL supportata da una funzione Lambda.

Proprietà	Valore
`default`	`postgres://jdbc:postgresql://postgres1.host:5432/default?secret=${Test/RDS/PostgreSQL1}`

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

La tabella seguente mostra i tipi di dati corrispondenti per JDBC, PostGre SQL e Arrow.

JDBC	PostGreSQL	Arrow
Booleano	Booleano	Bit

JDBC	PostGreSQL	Arrow
Numero intero	N/D	Tiny
Breve	smallint	Smallint
Numero intero	integer	Int
Long	bigint	Bigint
float	float4	Float4
Doppio	float8	Float8
Data	date	DateDay
Timestamp	timestamp	DateMilli
Stringa	text	Varchar
Byte	bytes	Varbinary
BigDecimal	numeric(p,s)	Decimale
ARRAY	N/D (vedi nota)	Elenco

Note

Il tipo ARRAY è supportato per il connettore PostgreSQL con i seguenti vincoli: le matrici multidimensionali (`<data_type>[][]` o matrici nidificate) non sono supportate. Le colonne con tipi di dati ARRAY non supportati vengono convertite in una matrice di elementi di stringa (`array<varchar>`).

Partizioni e suddivisioni

Le partizioni vengono utilizzate per determinare come generare suddivisioni per il connettore. Athena costruisce una colonna sintetica di tipo `varchar` che rappresenta lo schema di partizionamento della tabella per aiutare il connettore a generare suddivisioni. Il connettore non modifica la definizione effettiva della tabella.

Prestazioni

PostgreSQL supporta le partizioni native. Il connettore PostgreSQL di Athena può recuperare dati da queste partizioni in parallelo. Se desideri interrogare set di dati molto grandi con una distribuzione uniforme delle partizioni, ti consigliamo vivamente il partizionamento nativo.

Il connettore Athena PostgreSQL esegue il pushdown dei predicati per ridurre i dati scansionati dalla query. Le clausole `LIMIT`, i predicati semplici e le espressioni complesse vengono inviati al connettore per ridurre la quantità di dati scansionati e per ridurre il tempo di esecuzione delle query. Tuttavia, la selezione di un sottoinsieme di colonne comporta un runtime di esecuzione delle query più lungo.

Clausole LIMIT

Una dichiarazione `LIMIT N` riduce la quantità di dati analizzati dalla query. Con il pushdown `LIMIT N`, il connettore restituisce solo le righe `N` ad Athena.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore PostgreSQL di Athena può combinare queste espressioni e inviarle direttamente a PostgreSQL per funzionalità avanzate e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore PostgreSQL di Athena supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_DISTINCT_FROM, NULL_IF, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
```

```
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%')
LIMIT 10;
```

Interrogazioni pass-through

Il connettore PostgreSQL supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con PostgreSQL, puoi usare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a un'origine dati in PostgreSQL. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consulta il file pom.xml per il connettore PostgreSQL su .com. GitHub

Per ulteriori informazioni su questo connettore, visita il sito corrispondente su .com. GitHub

Connettore Amazon Athena Redis

Il connettore Amazon Athena Redis consente ad Amazon Athena di comunicare con le istanze Redis in modo da poter eseguire query sui dati Redis con SQL. Puoi usare AWS Glue Data Catalog per mappare le tue coppie chiave-valore Redis in tabelle virtuali.

A differenza dei tradizionali archivi di dati relazionali, Redis non ha il concetto di tabella o colonna. Al contrario, Redis offre modelli di accesso chiave-valore in cui la chiave è essenzialmente una `string` e il valore è una `string`, un `z-set` oppure una hmap.

È possibile utilizzare il AWS Glue Data Catalog per creare schemi e configurare tabelle virtuali. Le proprietà speciali della tabella indicano al connettore Redis per Athena come mappare le chiavi e i valori di Redis in una tabella. Per ulteriori informazioni, consultare Configurazione di database e tabelle in AWS Glue riportata di seguito in questo documento.

Se hai abilitato Lake Formation nel tuo account, il ruolo IAM per il tuo connettore Lambda federato Athena che hai distribuito nell'accesso in lettura deve avere accesso in lettura in AWS Serverless Application Repository Lake Formation a. AWS Glue Data Catalog

Il connettore Amazon Athena Redis supporta Amazon MemoryDB per Redis e Amazon per Redis. ElastiCache

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.
- Prima di utilizzare questo connettore, configura un VPC e un gruppo di sicurezza. Per ulteriori informazioni, consulta Creazione di un VPC per un connettore origine dati.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore Redis.

- spill_bucket: specifica il bucket Amazon S3 per i dati che superano i limiti della funzione Lambda.
- spill_prefix: (facoltativo) per impostazione predefinita, viene utilizzata una sottocartella nello `spill_bucket` specificato chiamata `athena-federation-spill`. Ti consigliamo di configurare un ciclo di vita dell'archiviazione di Amazon S3 in questa posizione per eliminare gli spill più vecchi di un numero predeterminato di giorni o ore.
- spill_put_request_headers: (facoltativo) una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.
- kms_key_id: (facoltativo) per impostazione predefinita, tutti i dati riversati in Amazon S3 vengono crittografati utilizzando la modalità di crittografia autenticata AES-GCM e una chiave generata casualmente. Per fare in modo che la tua funzione Lambda utilizzi chiavi di crittografia più potenti

generate da KMS come `a7e63k4b-8loc-40db-a2a1-4d0en2cd8331`, puoi specificare l'ID della chiave KMS.

- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.
- glue_catalog: (facoltativo) utilizza questa opzione per specificare un catalogo AWS Glue multi-account. Per impostazione predefinita, il connettore tenta di ottenere metadati dal proprio account. AWS Glue

Configurazione di database e tabelle in AWS Glue

Per abilitare una AWS Glue tabella da utilizzare con Redis, puoi impostare le seguenti proprietà della tabella: `redis-endpointredis-value-type`, e `redis-keys-zset` o`redis-key-prefix`.

Inoltre, qualsiasi AWS Glue database che contiene tabelle Redis deve avere una `redis-db-flag` proprietà URI del database. Per impostare la proprietà `redis-db-flag` URI, usa la AWS Glue console per modificare il database.

L'elenco seguente descrive le proprietà delle tabelle.

- redis-endpoint — *(Obbligatorio) La : password della : porta del nome host del server Redis che contiene i dati per questa tabella (ad esempioathena-federation-demo.cache.amazonaws.com:6379). In alternativa, puoi memorizzare l'endpoint, o parte dell'endpoint, AWS Secrets Manager utilizzando $ {Secret_Name} come valore della proprietà della tabella.*

> **Note**
>
> Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

- redis-keys-zset— (Obbligatorio se non `redis-key-prefix` viene utilizzato) Un elenco di chiavi separate da virgole il cui valore è un set (ad esempio,). `active-orders,pending-orders`

Ciascuno dei valori del set viene considerato come una chiave facente parte della tabella. È necessario impostare la proprietà `redis-keys-zset` o la proprietà `redis-key-prefix`.

- redis-key-prefix— (Obbligatorio se non `redis-keys-zset` viene utilizzato) Un elenco separato da virgole di prefissi chiave per la scansione dei valori nella tabella (ad esempio,). `accounts-*,acct-` È necessario impostare la proprietà `redis-key-prefix` o la proprietà `redis-keys-zset`.
- redis-value-type— (Obbligatorio) Definisce in che modo i valori delle chiavi definite da uno `redis-key-prefix` o dall'altro `redis-keys-zset` vengono mappati sulla tabella. Un valore letterale è associato a una singola colonna. Anche un zset è associato a una singola colonna, ma ogni chiave può archiviare più righe. Un hash consente a ciascuna chiave di costituire una riga con più colonne (ad esempio, hash, letterale o zset).
- redis-ssl-flag— (Facoltativo) Quando`True`, crea una connessione Redis che utilizza SSL/TLS. Il valore predefinito è `False`.
- redis-cluster-flag— (Facoltativo) Quando`True`, abilita il supporto per le istanze Redis in cluster. Il valore predefinito è `False`.
- redis-db-number— (Facoltativo) Si applica solo alle istanze autonome e non raggruppate.) Imposta questo numero (ad esempio 1, 2 o 3) per leggere da un database Redis non predefinito. L'impostazione predefinita è il database logico Redis 0. Questo numero non si riferisce a un database in Athena o AWS Glue, ma a un database logico Redis. Per ulteriori informazioni, consulta la sezione SELECT index (Indice SELECT) nella documentazione di Redis.

Tipi di dati

Il connettore Redis supporta i seguenti tipi di dati. I flussi Redis non sono supportati.

- Stringa
- Hash
- Set ordinato (ZSet)

Tutti i valori di Redis vengono recuperati come tipo di dati `string`. Quindi vengono convertiti in uno dei seguenti tipi di dati Apache Arrow in base a come sono state definite le tabelle nel AWS Glue Data Catalog.

AWS Glue tipo di dati	Tipo di dati Apache Arrow
int	INT
string	VARCHAR
bigint	BIGINT
double	FLOAT8
float	FLOAT4
smallint	SMALLINT
tinyint	TINYINT
booleano	BIT
binary	VARBINARY

Autorizzazioni richieste

Consulta la sezione `Policies` del file athena-redis.yaml per i dettagli completi delle policy IAM richieste da questo connettore. L'elenco che segue riporta un riepilogo delle autorizzazioni richieste.

- Accesso in scrittura ad Amazon S3: per trasferire i risultati di query di grandi dimensioni, il connettore richiede l'accesso in scrittura a una posizione in Amazon S3.
- Athena GetQueryExecution: il connettore utilizza questa autorizzazione per fallire rapidamente quando la query Athena upstream è terminata.
- AWS Glue Data Catalog— Il connettore Redis richiede l'accesso in sola lettura a per ottenere informazioni sullo schema. AWS Glue Data Catalog
- CloudWatch Registri: il connettore richiede l'accesso ai CloudWatch registri per l'archiviazione dei registri.
- AWS Secrets Manager accesso in lettura: se scegli di archiviare i dettagli degli endpoint Redis in Secrets Manager, devi concedere al connettore l'accesso a tali segreti.
- Accesso VPC: il connettore richiede la capacità di collegare e scollegare le interfacce al VPC in modo che possa connettersi ad esso e comunicare con le istanze Redis.

Prestazioni

Il connettore Redis per Athena tenta di eseguire in parallelo le query sull'istanza Redis in base al tipo di tabella che hai definito (ad esempio, chiavi zset chiavi di prefisso).

Il connettore Athena Redis esegue il pushdown del predicato per ridurre la quantità di dati scansionati dalla query. Tuttavia, le query contenenti un predicato sulla chiave primaria comportano un errore di timeout. Le clausole `LIMIT` riducono la quantità di dati scansionati, ma se non viene fornito un predicato, le query `SELECT` con una clausola `LIMIT` eseguiranno la scansione di almeno 16 MB di dati. Il connettore Redis è resiliente alla limitazione della larghezza di banda della rete dovuta alla simultaneità.

Informazioni sulla licenza

Il progetto del connettore Redis per Amazon Athena è concesso in licenza ai sensi della Licenza Apache-2.0.

Risorse aggiuntive

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su GitHub .com.

Connettore Redshift di Amazon Athena

Il connettore Amazon Athena Redshift consente ad Amazon Athena di accedere ai tuoi database Amazon Redshift e Amazon Redshift Serverless, incluse le viste Redshift Serverless. Puoi connetterti a entrambi i servizi utilizzando le impostazioni di configurazione della stringa di connessione JDBC descritte in questa pagina.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .

- Poiché Redshift non supporta le partizioni esterne, tutti i dati specificati da una query vengono recuperati ogni volta.

Termini

I seguenti termini sono correlati al connettore Redshift.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore Redshift.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
redshift://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`RedshiftMuxCompositeHandler`
Gestore dei metadati	`RedshiftMuxMetadataHandler`
Gestore dei record	`RedshiftMuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
$*catalog*_connection_string	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `myredshiftcatalog` , il nome della variabile di ambiente è `myredshiftcatalog_connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *AWS_LAMBDA_FUNCTION_NAME* `}`.

Le seguenti proprietà di esempio si riferiscono a una funzione Lambda Redshift MUX che supporta due istanze del database: `redshift1` (il valore predefinito) e `redshift2`.

Proprietà	Valore
`default`	`redshift://jdbc:redshift://redshift1.host:5439/dev?user=sample2&password=sample2`

Proprietà	Valore
`redshift_catalog1_connection_string`	`redshift://jdbc:redshift://redshift1.host:3306/default?${Test/RDS/Redshift1}`
`redshift_catalog2_connection_string`	`redshift://jdbc:redshift://redshift2.host:3333/default?user=sample2&password=sample2`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti codificati in AWS Secrets Manager, consulta Move i segreti codificati nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

```
{"username": "${username}", "password": "${password}"}
```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto ${Test/RDS/ `Redshift1`}.

```
redshift://jdbc:redshift://redshift1.host:3306/default?...&${Test/RDS/Redshift1}&...
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
redshift://jdbc:redshift://redshift1.host:3306/
default?...&user=sample2&password=sample2&...
```

Attualmente, il connettore Redshift riconosce le proprietà JDBC `user` e `password`.

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per JDBC e Apache Arrow.

JDBC	Arrow
Booleano	Bit
Numero intero	Tiny
Breve	Smallint
Numero intero	Int
Long	Bigint
float	Float4
Doppio	Float8
Data	DateDay
Timestamp	DateMilli
Stringa	Varchar
Byte	Varbinary
BigDecimal	Decimale
ARRAY	Elenco

Partizioni e suddivisioni

Redshift non supporta le partizioni esterne. Per ulteriori informazioni sui problemi relativi alle prestazioni, consulta la sezione Prestazioni.

Prestazioni

Il connettore Athena Redshift esegue il pushdown dei predicati per ridurre i dati scansionati dalla query. `LIMIT`clausole, `ORDER BY` clausole, predicati semplici ed espressioni complesse vengono inserite nel connettore per ridurre la quantità di dati scansionati e ridurre il tempo di esecuzione delle query. Tuttavia, la selezione di un sottoinsieme di colonne comporta un runtime di esecuzione delle query più lungo. Amazon Redshift è particolarmente soggetto a rallentamenti nell'esecuzione delle query quando più query vengono eseguite simultaneamente.

Clausole LIMIT

Una dichiarazione `LIMIT N` riduce la quantità di dati analizzati dalla query. Con il pushdown `LIMIT N`, il connettore restituisce solo le righe `N` ad Athena.

Query top N

Una query top `N` principale specifica un ordinamento dei set di risultati e un limite al numero di righe restituite. Puoi utilizzare questo tipo di query per determinare i valori massimi top `N` o i valori minimi top `N` per i set di dati. Con il pushdown top `N`, il connettore restituisce solo le righe ordinate `N` ad Athena.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Redshift di Athena può combinare queste espressioni e inviarle direttamente a Redshift per migliorare le funzionalità e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore Redshift di Athena supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_DISTINCT_FROM, NULL_IF, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%')
ORDER BY col_a DESC
LIMIT 10;
```

Per un articolo sull'utilizzo del pushdown del predicato per migliorare le prestazioni nelle query federate, incluso Amazon Redshift, consulta Come migliorare le query federate con il pushdown del predicato in Amazon Athena nel Blog sui Big Data di AWS .

Interrogazioni pass-through

Il connettore Redshift supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Redshift, puoi utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a un'origine dati in Redshift. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consulta il file pom.xml per il connettore Redshift su .com. GitHub

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su .com. GitHub

Connettore Amazon Athena per SAP HANA

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.

- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .
- In SAP HANA, i nomi degli oggetti vengono convertiti in lettere maiuscole quando vengono archiviati nel database SAP HANA. Tuttavia, poiché i nomi tra virgolette fanno distinzione tra maiuscole e minuscole, è possibile che due tabelle abbiano lo stesso nome in lettere minuscole e maiuscole (ad esempio, `EMPLOYEE` e `employee`).

  In Athena Federated Query, i nomi delle tabelle dello schema vengono forniti alla funzione Lambda in lettere minuscole. Per risolvere il problema, puoi fornire suggerimenti sulle query `@schemaCase` per recuperare i dati dalle tabelle i cui nomi che fanno distinzione tra maiuscole e minuscole. Di seguito sono riportate due query di esempio con suggerimenti per le query.

```
SELECT *
FROM "lambda:saphanaconnector".SYSTEM."MY_TABLE@schemaCase=upper&tableCase=upper"
```

```
SELECT *
FROM "lambda:saphanaconnector".SYSTEM."MY_TABLE@schemaCase=upper&tableCase=lower"
```

Termini

I seguenti termini si riferiscono al connettore SAP HANA.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.

- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore SAP HANA.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
saphana://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`SaphanaMuxCompositeHandler`
Gestore dei metadati	`SaphanaMuxMetadataHandler`
Gestore dei record	`SaphanaMuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
$*catalog*_connection_string	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `mysaphanacatalog` , il nome della variabile di ambiente è `mysaphanacatalog_connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *AWS_LAMBDA_FUNCTION_NAME* `}`.

Le seguenti proprietà di esempio si riferiscono a una funzione Lambda Saphana MUX che supporta due istanze del database: `saphana1` (il valore predefinito) e `saphana2`.

Proprietà	Valore
`default`	`saphana://jdbc:sap://saphana1.host:port/?${Test/RDS/Saphana1}`
`saphana_catalog1_connection_string`	`saphana://jdbc:sap://saphana1.host:port/?${Test/RDS/Saphana1}`
`saphana_catalog2_connection_string`	`saphana://jdbc:sap://saphana2.host:port/?user=sample2&password=sample2`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti codificati in AWS Secrets Manager, consulta Move i segreti codificati nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

  ```
  {"username": "${username}", "password": "${password}"}
  ```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${Test/RDS/Saphana1}`.

```
saphana://jdbc:sap://saphana1.host:port/?${Test/RDS/Saphana1}&...
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
saphana://jdbc:sap://saphana1.host:port/?user=sample2&password=sample2&...
```

Attualmente, il connettore SAP HANA riconosce le proprietà JDBC `user` e `password`.

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza SAP HANA.

Tipo di gestore	Classe
Gestore composito	`SaphanaCompositeHandler`
Gestore dei metadati	`SaphanaMetadataHandler`
Gestore dei record	`SaphanaRecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza SAP HANA supportata da una funzione Lambda.

Proprietà	Valore
`default`	`saphana://jdbc:sap://saphana1.host:port/?secret=Test/RDS/Saphana1`

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per JDBC e Apache Arrow.

JDBC	Arrow
Booleano	Bit
Numero intero	Tiny
Breve	Smallint
Numero intero	Int
Long	Bigint
float	Float4
Doppio	Float8
Data	DateDay
Timestamp	DateMilli
Stringa	Varchar

JDBC	Arrow
Byte	Varbinary
BigDecimal	Decimale
ARRAY	Elenco

Conversioni dei tipi di dati

Oltre alle conversioni da JDBC a Arrow, il connettore esegue alcune altre conversioni per rendere compatibili l'origine SAP HANA e i tipi di dati Athena. Queste conversioni contribuiscono a garantire che le query vengano eseguite in modo corretto. Nella tabella seguente vengono illustrate queste conversioni.

Tipo di dati di origine (SAP HANA)	Tipo di dati convertito (Athena)
DECIMAL	BIGINT
INTEGER	INT
DATE	DATEDAY
TIMESTAMP	DATEMILLI

Tutti gli altri tipi di dati non supportati vengono convertiti in `VARCHAR`.

Partizioni e suddivisioni

Una partizione è rappresentata da una singola colonna di partizione di tipo `Integer`. La colonna contiene i nomi delle partizioni definite in una tabella SAP HANA. Se una tabella non ha nomi di partizione, viene restituito *, che equivale a una singola partizione. Una partizione equivale a una suddivisione.

Nome	Type	Descrizione
PART_ID	Numero intero	Partizione denominata in SAP HANA.

Prestazioni

SAP HANA supporta le partizioni native. Il connettore SAP HANA di Athena può recuperare dati da queste partizioni in parallelo. Se desideri interrogare set di dati molto grandi con una distribuzione uniforme delle partizioni, ti consigliamo vivamente il partizionamento nativo. La selezione di un sottoinsieme di colonne velocizza notevolmente il runtime delle query e riduce i dati scansionati. Il connettore presenta una significativa limitazione della larghezza di banda della rete e talvolta errori di query dovuti alla simultaneità.

Il connettore Athena SAP HANA esegue il pushdown dei predicati per ridurre i dati scansionati dalla query. Le clausole `LIMIT`, i predicati semplici e le espressioni complesse vengono inviati al connettore per ridurre la quantità di dati analizzati e per ridurre il tempo di esecuzione delle query.

Clausole LIMIT

Una dichiarazione `LIMIT N` riduce la quantità di dati analizzati dalla query. Con il pushdown `LIMIT N`, il connettore restituisce solo le righe `N` ad Athena.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Athena SAP HANA può combinare queste espressioni e inviarle direttamente a SAP HANA per migliorare le funzionalità e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore Athena SAP HANA supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_DISTINCT_FROM, NULL_IF, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
```

```
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%')
LIMIT 10;
```

Interrogazioni pass-through

Il connettore SAP HANA supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con SAP HANA, puoi utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a un'origine dati in SAP HANA. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consulta il file pom.xml per il connettore SAP HANA su .com. GitHub

Per ulteriori informazioni su questo connettore, visita il sito corrispondente su .com. GitHub

Connettore Amazon Athena per Snowflake

Il connettore Amazon Athena per Snowflake consente ad Amazon Athena di eseguire query SQL sui dati archiviati nei database SQL Snowflake o sulle istanze RDS utilizzando JDBC.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .
- Al momento, le viste di Snowflake non sono supportate.
- In Snowflake, poiché i nomi degli oggetti fanno distinzione tra maiuscole e minuscole, due tabelle possono avere lo stesso nome in lettere minuscole e maiuscole (ad esempio, `EMPLOYEE` e `employee`). In Athena Federated Query, i nomi delle tabelle dello schema vengono forniti alla funzione Lambda in lettere minuscole. Per risolvere il problema, puoi fornire suggerimenti sulle query `@schemaCase` per recuperare i dati dalle tabelle i cui nomi che fanno distinzione tra maiuscole e minuscole. Di seguito sono riportate due query di esempio con suggerimenti per le query.

```
SELECT *
FROM "lambda:snowflakeconnector".SYSTEM."MY_TABLE@schemaCase=upper&tableCase=upper"
```

```
SELECT *
FROM "lambda:snowflakeconnector".SYSTEM."MY_TABLE@schemaCase=upper&tableCase=lower"
```

Termini

I seguenti termini si riferiscono al connettore Snowflake.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.

- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore Snowflake.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
snowflake://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`SnowflakeMuxCompositeHandler`

Gestore	Classe
Gestore dei metadati	`SnowflakeMuxMetadataHandler`
Gestore dei record	`SnowflakeMuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
`$`*`catalog`*`_connection_string`	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `mysnowflakecatalog` , il nome della variabile di ambiente è `mysnowflakecatalog _connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *`AWS_LAMBDA_FUNCTION_NAME`* `}`.

Le seguenti proprietà di esempio si riferiscono a una funzione Lambda Snowflake MUX che supporta due istanze del database: `snowflake1` (il valore predefinito) e `snowflake2`.

Proprietà	Valore
`default`	`snowflake://jdbc:snowflake://snowflake1.host:port/?warehouse=warehousename&db=db1&schema=schema1&${Test/RDS/Snowflake1}`
`snowflake_catalog1_connection_string`	`snowflake://jdbc:snowflake://snowflake1.host:port/?warehouse=warehousename&db=db1&schema=schema1${Test/RDS/Snowflake1}`

Proprietà	Valore
`snowflake_catalog2_connection_string`	`snowflake://jdbc:snowflake://snowflake2.host:port/?warehouse=warehousename&db=db1&schema=schema1&user=sample2&password=sample2`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti codificati in AWS Secrets Manager, consulta Move i segreti codificati nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

  ```
  {"username": "${username}", "password": "${password}"}
  ```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${Test/RDS/Snowflake1}`.

```
snowflake://jdbc:snowflake://snowflake1.host:port/?
warehouse=warehousename&db=db1&schema=schema1${Test/RDS/Snowflake1}&...
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
snowflake://jdbc:snowflake://snowflake1.host:port/
warehouse=warehousename&db=db1&schema=schema1&user=sample2&password=sample2&...
```

Attualmente, Snowflake riconosce le proprietà JDBC `user` e `password`. Accetta anche il nome utente e la password nel formato *`nome utente`*`/`*`password`* senza le chiavi `user` o `password`.

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza Snowflake.

Tipo di gestore	Classe
Gestore composito	`SnowflakeCompositeHandler`
Gestore dei metadati	`SnowflakeMetadataHandler`
Gestore dei record	`SnowflakeRecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza Snowflake supportata da una funzione Lambda.

Proprietà	Valore
`default`	`snowflake://jdbc:snowflake://snowflake1.host:port/?secret=Test/RDS/Snowflake1`

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per JDBC e Apache Arrow.

JDBC	Arrow
Booleano	Bit
Numero intero	Tiny

JDBC	Arrow
Breve	Smallint
Numero intero	Int
Long	Bigint
float	Float4
Doppio	Float8
Data	DateDay
Timestamp	DateMilli
Stringa	Varchar
Byte	Varbinary
BigDecimal	Decimale
ARRAY	Elenco

Conversioni dei tipi di dati

Oltre alle conversioni da JDBC ad Arrow, il connettore esegue alcune altre conversioni per rendere compatibili l'origine Snowflake e i tipi di dati Athena. Queste conversioni contribuiscono a garantire che le query vengano eseguite in modo corretto. Nella tabella seguente vengono illustrate queste conversioni.

Tipo di dati di origine (Snowflake)	Tipo di dati convertito (Athena)
TIMESTAMP	TIMESTAMPMILLI
DATE	TIMESTAMPMILLI
INTEGER	INT

Tipo di dati di origine (Snowflake)	Tipo di dati convertito (Athena)
DECIMAL	BIGINT
TIMESTAMP_NTZ	TIMESTAMPMILLI

Tutti gli altri tipi di dati non supportati vengono convertiti in `VARCHAR`.

Partizioni e suddivisioni

Le partizioni vengono utilizzate per determinare come generare suddivisioni per il connettore. Athena costruisce una colonna sintetica di tipo `varchar` che rappresenta lo schema di partizionamento della tabella per aiutare il connettore a generare suddivisioni. Il connettore non modifica la definizione effettiva della tabella.

Per creare questa colonna sintetica e le partizioni, Athena richiede la definizione di una chiave primaria. Tuttavia, poiché Snowflake non impone vincoli relativi alla chiave primaria, è necessario imporre l'unicità autonomamente. In caso contrario, Athena passerà automaticamente a una singola divisione.

Prestazioni

Per assicurare prestazioni ottimali, utilizza i filtri nelle query ogniqualvolta possibile. Inoltre, consigliamo vivamente di utilizzare il partizionamento nativo per recuperare set di dati di grandissime dimensioni con una distribuzione uniforme delle partizioni. La selezione di un sottoinsieme di colonne velocizza notevolmente il runtime delle query e riduce i dati scansionati. Il connettore Snowflake è resiliente alla limitazione della larghezza di banda della rete dovuta alla simultaneità.

Il connettore Athena Snowflake esegue il pushdown dei predicati per ridurre i dati scansionati dalla query. Le clausole `LIMIT`, i predicati semplici e le espressioni complesse vengono inviati al connettore per ridurre la quantità di dati scansionati e per ridurre il tempo di esecuzione delle query.

Clausole LIMIT

Una dichiarazione `LIMIT N` riduce la quantità di dati analizzati dalla query. Con il pushdown `LIMIT N`, il connettore restituisce solo le righe `N` ad Athena.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Snowflake di Athena può combinare

queste espressioni e inviarle direttamente a Snowflake per migliorare le funzionalità e ridurre la quantità di dati scansionati.

I seguenti operatori del connettore Athena Snowflake supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_DISTINCT_FROM, NULL_IF, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%')
LIMIT 10;
```

Interrogazioni pass-through

Il connettore Snowflake supporta le interrogazioni passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Snowflake, puoi utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a un'origine dati in Snowflake. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
```

```
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consulta il file pom.xml per il connettore Snowflake su .com. GitHub

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su .com. GitHub

Connettore Amazon Athena per Microsoft SQL Server

Il connettore Amazon Athena per Microsoft SQL Server consente ad Amazon Athena di eseguire query SQL sui dati archiviati in Microsoft SQL Server utilizzando JDBC.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .
- Nelle condizioni di filtro, è necessario impostare i tipi di dati `Date` e `Timestamp` sul tipo di dati appropriato.
- Per cercare valori negativi del tipo `Real` e `Float`, utilizza l'operatore <= o >=.

- I tipi di dati `binary`, `varbinary`, `image` e `rowversion` non sono supportati.

Termini

I seguenti termini si riferiscono al connettore SQL Server.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore SQL Server.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
sqlserver://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`SqlServerMuxCompositeHandler`
Gestore dei metadati	`SqlServerMuxMetadataHandler`
Gestore dei record	`SqlServerMuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
`$`*`catalog`*`_connection_string`	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `mysqlservercatalog` , il nome della variabile di ambiente è `mysqlservercatalog_connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${`*`AWS_LAMBDA_FUNCTION_NAME`*`}`.

Le seguenti proprietà di esempio riguardano una funzione Lambda SqlServer MUX che supporta due istanze di database`sqlserver1`: (impostazione predefinita) e. `sqlserver2`

Proprietà	Valore
`default`	`sqlserver://jdbc:sqlserver://sqlserver1.`*`hostname`*`:`*`port`*`;databaseName=`*`<database_name>`*`;${`*`secret1_name`*`}`

Proprietà	Valore
`sqlserver_catalog1_connection_string`	`sqlserver://jdbc:sqlserver://sqlserver1.` *`hostname`*`:`*`port`*`;databaseName=` *`<database_name>`* `;${`*`secret1_name`*` }`
`sqlserver_catalog2_connection_string`	`sqlserver://jdbc:sqlserver://sqlserver2.` *`hostname`*`:`*`port`*`;databaseName=` *`<database_name>`* `;${`*`secret2_name`*` }`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti hardcoded in AWS Secrets Manager, consultate Move hardcoded secret in nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

```
{"username": "${username}", "password": "${password}"}
```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${secret_name}`.

```
sqlserver://jdbc:sqlserver://hostname:port;databaseName=<database_name>;${secret_name}
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
sqlserver://
jdbc:sqlserver://
hostname:port;databaseName=<database_name>;user=<user>;password=<password>
```

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza SQL Server.

Tipo di gestore	Classe
Gestore composito	`SqlServerCompositeHandler`
Gestore dei metadati	`SqlServerMetadataHandler`
Gestore dei record	`SqlServerRecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza SQL Server supportata da una funzione Lambda.

Proprietà	Valore
`default`	`sqlserver://jdbc:sqlserver://` *`hostname`*`:`*`port`*`;database Name=` *`<database_name>`* `;${`*`secret_name`*` }`

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per SQL Server e Apache Arrow.

SQL Server	Arrow
bit	TINYINT

SQL Server	Arrow
tinyint	SMALLINT
smallint	SMALLINT
int	INT
bigint	BIGINT
decimal	DECIMAL
numeric	FLOAT8
smallmoney	FLOAT8
money	DECIMAL
float[24]	FLOAT4
float[53]	FLOAT8
real	FLOAT4
datetime	Date(MILLISECOND)
datetime2	Date(MILLISECOND)
smalldatetime	Date(MILLISECOND)
data	Date(DAY)
time	VARCHAR
datetimeoffset	Date(MILLISECOND)
char[n]	VARCHAR
varchar[n/max]	VARCHAR
nchar[n]	VARCHAR

SQL Server	Arrow
nvarchar[n/max]	VARCHAR
text	VARCHAR
ntext	VARCHAR

Partizioni e suddivisioni

Una partizione è rappresentata da una singola colonna di partizione di tipo `varchar`. Nel caso del connettore SQL Server, una funzione di partizione determina il modo in cui le partizioni vengono applicate alla tabella. Le informazioni sulla funzione e sul nome della colonna di partizione vengono recuperate dalla tabella dei metadati di SQL Server. La partizione viene quindi recuperata con una query personalizzata. Le suddivisioni vengono create in base al numero di partizioni distinte ricevute.

Prestazioni

La selezione di un sottoinsieme di colonne velocizza notevolmente il runtime delle query e riduce i dati scansionati. Il connettore SQL Server è resiliente alla limitazione della larghezza di banda della rete dovuta alla simultaneità.

Il connettore Athena SQL Server esegue il pushdown del predicato per ridurre la quantità di dati scansionati dalla query. I predicati semplici e le espressioni complesse vengono inviati al connettore per ridurre la quantità di dati scansionati e ridurre il tempo di esecuzione delle query.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Athena SQL Server può combinare queste espressioni e inviarle direttamente a SQL Server per migliorare le funzionalità e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore SQL Server di Athena supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, IS_DISTINCT_FROM, NULL_IF, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE

- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%');
```

Interrogazioni pass-through

Il connettore SQL Server supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query all'origine dati per l'esecuzione.

Per utilizzare le query passthrough con SQL Server, è possibile utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a un'origine dati in SQL Server. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, vedere il file pom.xml per il connettore SQL Server all'indirizzo .com. GitHub

Per ulteriori informazioni su questo connettore, visita il sito corrispondente su GitHub .com.

Connettore Amazon Athena per Teradata

Il connettore Amazon Athena per Teradata consente ad Amazon Athena di eseguire query SQL sui dati archiviati nei database Teradata.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Limitazioni

- Le operazioni di scrittura DDL non sono supportate.
- In una configurazione multiplex, il bucket di spill e il prefisso sono condivisi tra tutte le istanze del database.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .

Termini

I seguenti termini si riferiscono al connettore Teradata.

- Istanza del database: qualsiasi istanza del database distribuita on-premise, su Amazon EC2 o su Amazon RDS.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.

- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`
- Gestore multiplex: un gestore Lambda in grado di accettare e utilizzare più connessioni al database.

Prerequisito del livello Lambda

Per utilizzare il connettore per Teradata con Athena, è necessario creare un livello Lambda che includa il driver Teradata JDBC. Un livello Lambda è un archivio di file `.zip` contenente un codice aggiuntivo per una funzione Lambda. Quando distribuisci il connettore Teradata nel tuo account, specifichi l'ARN del livello. Ciò collega il livello Lambda con il driver Teradata JDBC al connettore Teradata, in modo da poterlo utilizzare con Athena.

Per ulteriori informazioni sui livelli Lambda, consulta Creazione e condivisione dei livelli Lambda nella Guida per gli sviluppatori di AWS Lambda .

Per creare un livello Lambda per il connettore Teradata

1. Vai alla pagina di download del driver JDBC Teradata all'indirizzo https://downloads.teradata.com/download/connectivity/jdbc-driver.
2. Scarica il driver JDBC Teradata. Il sito Web richiede di creare un account e di accettare un contratto di licenza per scaricare il file.
3. Estrai il file `terajdbc4.jar` dall'archivio dei file che hai scaricato.
4. Crea la seguente struttura di cartelle in cui posizionare il file `.jar`.

   ```
   java\lib\terajdbc4.jar
   ```
5. Crea un file `.zip` dell'intera struttura di cartelle che contiene il file `terajdbc4.jar`.
6. Accedi AWS Management Console e apri la AWS Lambda console all'indirizzo https://console.aws.amazon.com/lambda/.
7. Scegli Layers (Livelli) nel pannello di navigazione, quindi scegli Create layer (Crea un livello).
8. Per Name (Nome), immettere un nome per il livello (ad esempio, `TeradataJava11LambdaLayer`).

9. Assicurati che l'opzione Upload a .zip file (Carica un file .zip) sia selezionata.
10. Scegli Upload (Carica) e quindi carica la cartella compressa che contiene il driver JDBC Teradata.
11. Scegli Crea.
12. Nella pagina dei dettagli del layer, copia il layer ARN scegliendo l'icona degli appunti nella parte superiore.
13. Salva l'ARN come riferimento.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore Teradata.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
teradata://${jdbc_connection_string}
```

Utilizzo di un gestore multiplex

Puoi utilizzare un gestore multiplex per connetterti a più istanze del database con una singola funzione Lambda. Le richieste vengono indirizzate in base al nome del catalogo. Utilizza le seguenti classi in Lambda.

Gestore	Classe
Gestore composito	`TeradataMuxCompositeHandler`
Gestore dei metadati	`TeradataMuxMetadataHandler`
Gestore dei record	`TeradataMuxRecordHandler`

Parametri del gestore multiplex

Parametro	Descrizione
`$`*`catalog`*`_connection_string`	Obbligatorio. Una stringa di connessione di un'istanza del database. Appone alla variabile di ambiente un prefisso con il nome del catalogo utilizzato in Athena. Ad esempio, se il catalogo registrato presso Athena è `myteradatacatalog` , il nome della variabile di ambiente è `myteradatacatalog_connection_string` .
`default`	Obbligatorio. La stringa di connessione predefinita. Questa stringa viene utilizzata quando il catalogo è `lambda:${` *`AWS_LAMBDA_FUNCTION_NAME`* `}`.

Le seguenti proprietà di esempio si riferiscono a una funzione Lambda Teradata MUX che supporta due istanze del database: `teradata1` (il valore predefinito) e `teradata2`.

Proprietà	Valore
`default`	`teradata://jdbc:teradata://teradata2.host/TMODE=ANSI,CHARSET=UTF8,DATABASE=TEST,user=sample2&password=sample2`
`teradata_catalog1_connection_string`	`teradata://jdbc:teradata://teradata1.host/TMODE=ANSI,CHARSET=UTF8,DATABASE=TEST,${Test/RDS/Teradata1}`
`teradata_catalog2_connection_string`	`teradata://jdbc:teradata://teradata2.host/TMODE=ANSI,CHARSET=UTF8,DATABASE=TEST,user=sample2&password=sample2`

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti hardcoded in AWS Secrets Manager, consulta Move hardcoded secret AWS Secrets Manager in nella Guida per l'utente.AWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

  È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

  Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

  ```
  {"username": "${username}", "password": "${password}"}
  ```

Esempio di stringa di connessione con nome del segreto

La stringa seguente ha il nome del segreto `${Test/RDS/Teradata1}`.

```
teradata://jdbc:teradata1.host/TMODE=ANSI,CHARSET=UTF8,DATABASE=TEST,${Test/RDS/
Teradata1}&...
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
teradata://jdbc:teradata://teradata1.host/
TMODE=ANSI,CHARSET=UTF8,DATABASE=TEST,...&user=sample2&password=sample2&...
```

Attualmente, Teradata riconosce le proprietà JDBC `user` e `password`. Accetta anche il nome utente e la password nel formato *`username`*`/`*`password`* (nome utente / password) senza le chiavi `user` o `password`.

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza Teradata.

Tipo di gestore	Classe
Gestore composito	`TeradataCompositeHandler`
Gestore dei metadati	`TeradataMetadataHandler`
Gestore dei record	`TeradataRecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

La seguente proprietà di esempio si riferisce a una singola istanza Teradata supportata da una funzione Lambda.

Proprietà	Valore
`default`	`teradata://jdbc:teradata://teradata1.host/TMODE=ANSI,C`

Proprietà	Valore
	`HARSET=UTF8,DATABASE=TEST,secret=Test/RDS/Teradata1`

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati corrispondenti per JDBC e Apache Arrow.

JDBC	Arrow
Booleano	Bit
Numero intero	Tiny
Breve	Smallint
Numero intero	Int
Long	Bigint

JDBC	Arrow
float	Float4
Doppio	Float8
Data	DateDay
Timestamp	DateMilli
Stringa	Varchar
Byte	Varbinary
BigDecimal	Decimale
ARRAY	Elenco

Partizioni e suddivisioni

Una partizione è rappresentata da una singola colonna di partizione di tipo `Integer`. La colonna contiene i nomi delle partizioni definite in una tabella Teradata. Se una tabella non ha nomi di partizione, viene restituito *, che equivale a una singola partizione. Una partizione equivale a una suddivisione.

Nome	Type	Descrizione
partition	Numero intero	Partizione denominata in Teradata.

Prestazioni

Teradata supporta le partizioni native. Il connettore Teradata di Athena può recuperare dati da queste partizioni in parallelo. Se desideri interrogare set di dati molto grandi con una distribuzione uniforme delle partizioni, ti consigliamo vivamente il partizionamento nativo. La selezione di un sottoinsieme di colonne rallenta notevolmente l'esecuzione delle query. Il connettore mostra alcune limitazioni della larghezza di banda della rete dovute alla simultaneità.

Il connettore Athena Teradata esegue il pushdown del predicato per ridurre la quantità di dati scansionati dalla query. I predicati semplici e le espressioni complesse vengono inviati al connettore per ridurre la quantità di dati scansionati e ridurre il tempo di esecuzione delle query.

Predicati

Un predicato è un'espressione nella clausola `WHERE` di una query SQL che valuta a un valore booleano e filtra le righe in base a più condizioni. Il connettore Teradata di Athena può combinare queste espressioni e inviarle direttamente a Teradata per funzionalità avanzate e per ridurre la quantità di dati scansionati.

I seguenti operatori del connettore Athena Teradata supportano il pushdown dei predicati:

- Booleano: AND, OR, NOT
- Uguaglianza: EQUAL, NOT_EQUAL, LESS_THAN, LESS_THAN_OR_EQUAL, GREATER_THAN, GREATER_THAN_OR_EQUAL, NULL_IF, IS_NULL
- Aritmetica: ADD, SUBTRACT, MULTIPLY, DIVIDE, MODULUS, NEGATE
- Altro: LIKE_PATTERN, IN

Esempio di pushdown combinato

Per le funzionalità di esecuzione di query avanzate, combina i tipi di pushdown, come nell'esempio seguente:

```
SELECT *
FROM my_table
WHERE col_a > 10
    AND ((col_a + col_b) > (col_c % col_d))
    AND (col_e IN ('val1', 'val2', 'val3') OR col_f LIKE '%pattern%');
```

Interrogazioni pass-through

Il connettore Teradata supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Teradata, è possibile utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
```

```
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a una fonte di dati in Teradata. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consultate il file pom.xml per il connettore Teradata su .com. GitHub

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su .com. GitHub

Connettore Amazon Athena Timestream

Il connettore Amazon Athena per Timestream consente ad Amazon Athena di comunicare con Amazon Timestream, rendendo i dati di serie temporali accessibili mediante Amazon Athena. Facoltativamente, puoi utilizzarlo AWS Glue Data Catalog come fonte di metadati supplementari.

Amazon Timestream è un database di serie temporali veloce, scalabile, completamente gestito e appositamente costruito che semplifica l'archiviazione e l'analisi giornaliera di trilioni di punti dati delle serie temporali. Timestream consente di risparmiare tempo e denaro nella gestione del ciclo di vita dei dati di serie temporali mantenendo i dati recenti in memoria e spostando i dati cronologici su un livello di archiviazione ottimizzato in base a policy definite dall'utente.

Se hai abilitato Lake Formation nel tuo account, il ruolo IAM per il tuo connettore Lambda federato Athena che hai distribuito nell'accesso in lettura deve avere accesso in lettura in AWS Serverless Application Repository Lake Formation a. AWS Glue Data Catalog

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore Timestream.

- spill_bucket: specifica il bucket Amazon S3 per i dati che superano i limiti della funzione Lambda.
- spill_prefix: (facoltativo) per impostazione predefinita, viene utilizzata una sottocartella nello `spill_bucket` specificato chiamata `athena-federation-spill`. Ti consigliamo di configurare un ciclo di vita dell'archiviazione di Amazon S3 in questa posizione per eliminare gli spill più vecchi di un numero predeterminato di giorni o ore.
- spill_put_request_headers: (facoltativo) una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.
- kms_key_id: (facoltativo) per impostazione predefinita, tutti i dati riversati in Amazon S3 vengono crittografati utilizzando la modalità di crittografia autenticata AES-GCM e una chiave generata casualmente. Per fare in modo che la tua funzione Lambda utilizzi chiavi di crittografia più potenti generate da KMS come `a7e63k4b-8loc-40db-a2a1-4d0en2cd8331`, puoi specificare l'ID della chiave KMS.
- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.
- glue_catalog: (facoltativo) utilizza questa opzione per specificare un catalogo AWS Glue multi-account. Per impostazione predefinita, il connettore tenta di ottenere metadati dal proprio account. AWS Glue

Configurazione di database e tabelle in AWS Glue

Facoltativamente, puoi utilizzarli AWS Glue Data Catalog come fonte di metadati supplementari. Per abilitare una AWS Glue tabella da utilizzare con Timestream, è necessario disporre di un database e di una tabella con nomi che corrispondano al AWS Glue database e alla tabella Timestream per i quali si desidera fornire metadati supplementari.

> **Note**
>
> Per prestazioni ottimali, utilizza solo lettere minuscole per i nomi dei database e delle tabelle. L'utilizzo di caratteri misti tra maiuscole e minuscole fa sì che il connettore esegua una ricerca senza distinzione tra maiuscole e minuscole, più impegnativa dal punto di vista computazionale.

Per configurare la AWS Glue tabella da utilizzare con Timestream, è necessario impostarne le proprietà in. AWS Glue

Per utilizzare una AWS Glue tabella per metadati supplementari

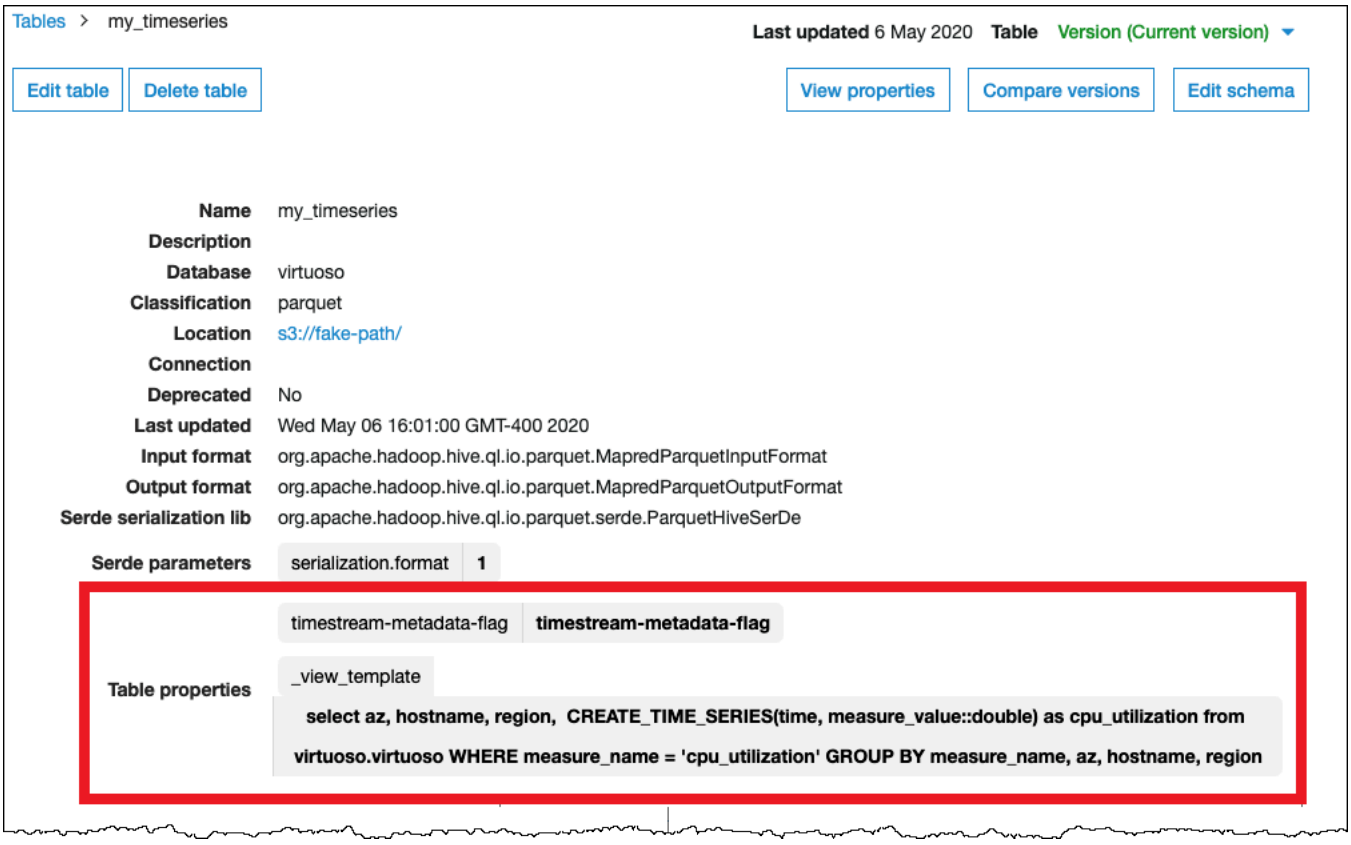
1. Modifica la tabella nella AWS Glue console per aggiungere le seguenti proprietà della tabella:
   - timestream-metadata-flag— Questa proprietà indica al connettore Timestream che il connettore può utilizzare la tabella per metadati supplementari. Puoi fornire qualsiasi valore per `timestream-metadata-flag`, purché la proprietà `timestream-metadata-flag` sia presente nell'elenco delle proprietà della tabella.
   - _view_template: quando utilizzi AWS Glue per i metadati supplementari, puoi utilizzare questa proprietà della tabella e specificare qualsiasi SQL Timestream come visualizzazione. Il connettore Timestream per Athena utilizza il codice SQL della visualizzazione insieme al codice SQL di Athena per eseguire la query. È utile se desideri utilizzare una funzionalità di Timestream SQL altrimenti non disponibile in Athena.
2. Assicurati di utilizzare i tipi di dati appropriati elencati in questo AWS Glue documento.

Tipi di dati

Attualmente, il connettore Timestream supporta solo un sottoinsieme dei tipi di dati disponibili in Timestream, in particolare i valori scalari `varchar`, `double` e `timestamp`.

Per interrogare il tipo di dati `timeseries`, devi configurare una visualizzazione nelle proprietà delle tabelle AWS Glue che utilizza la funzione di Timestream `CREATE_TIME_SERIES`. Inoltre, per la visualizzazione devi fornire uno schema che utilizzi la sintassi `ARRAY<STRUCT<time:timestamp,measure_value::double:double>>` come tipo per qualsiasi colonna delle tue serie temporali. Assicurati di sostituire `double` con il tipo scalare appropriato per la tabella.

L'immagine seguente mostra un esempio di proprietà della AWS Glue tabella configurate per impostare una visualizzazione su una serie temporale.



Autorizzazioni richieste

Consulta la sezione `Policies` del file athena-timestream.yaml per i dettagli completi delle policy IAM richieste da questo connettore. L'elenco che segue riporta un riepilogo delle autorizzazioni richieste.

- Accesso in scrittura ad Amazon S3: per trasferire i risultati di query di grandi dimensioni, il connettore richiede l'accesso in scrittura a una posizione in Amazon S3.
- Athena GetQueryExecution: il connettore utilizza questa autorizzazione per fallire rapidamente quando la query Athena upstream è terminata.

- AWS Glue Data Catalog— Il connettore Timestream richiede l'accesso in sola lettura a per ottenere informazioni sullo schema. AWS Glue Data Catalog
- CloudWatch Registri: il connettore richiede l'accesso ai CloudWatch registri per l'archiviazione dei registri.
- Accesso a Timestream: per eseguire query su Timestream.

Prestazioni

Ti consigliamo di utilizzare la clausola `LIMIT` per limitare i dati restituiti (non i dati scansionati) a meno di 256 MB per garantire l'efficienza delle query interattive.

Il connettore Timestream di Athena esegue il pushdown del predicato per ridurre la quantità di dati scansionati dalla query. Le clausole `LIMIT` riducono la quantità di dati scansionati, ma se non fornisci un predicato, le query `SELECT` con una clausola `LIMIT` eseguiranno la scansione di almeno 16 MB di dati. La selezione di un sottoinsieme di colonne velocizza notevolmente il runtime delle query e riduce i dati scansionati. Il connettore Timestream è resiliente alla limitazione della larghezza di banda della rete dovuta alla simultaneità.

Interrogazioni pass-through

Il connettore Timestream supporta le query passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Timestream, puoi utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a un'origine dati in Timestream. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Il progetto del connettore Timestream per Amazon Athena è concesso in licenza ai sensi della Licenza Apache-2.0.

Risorse aggiuntive

Per ulteriori informazioni su questo connettore, visita il sito corrispondente su GitHub .com.

Connettore Amazon Athena TPC Benchmark DS (TPC-DS)

Il connettore TPC-DS per Amazon Athena consente ad Amazon Athena di comunicare con un'origine di dati TPC Benchmark DS generati casualmente per l'utilizzo nel benchmarking e nei test funzionali di Athena Federation. Il connettore TPC-DS Athena genera un database compatibile con TPC-DS in corrispondenza di uno dei quattro fattori di scala. Non consigliamo di utilizzare questo connettore in alternativa ai test sulle prestazioni dei data lake basati su Amazon S3.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.

Parametri

Utilizza le variabili di ambiente Lambda illustrate in questa sezione per configurare il connettore TPC-DS.

- spill_bucket: specifica il bucket Amazon S3 per i dati che superano i limiti della funzione Lambda.
- spill_prefix: (facoltativo) per impostazione predefinita, viene utilizzata una sottocartella nello `spill_bucket` specificato chiamata `athena-federation-spill`. Ti consigliamo di configurare un ciclo di vita dell'archiviazione di Amazon S3 in questa posizione per eliminare gli spill più vecchi di un numero predeterminato di giorni o ore.
- spill_put_request_headers: (facoltativo) una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.
- kms_key_id: (facoltativo) per impostazione predefinita, tutti i dati riversati in Amazon S3 vengono crittografati utilizzando la modalità di crittografia autenticata AES-GCM e una chiave generata

casualmente. Per fare in modo che la tua funzione Lambda utilizzi chiavi di crittografia più potenti generate da KMS come `a7e63k4b-8loc-40db-a2a1-4d0en2cd8331`, puoi specificare l'ID della chiave KMS.

- disable_spill_encryption: (facoltativo) se impostato su `True`, disabilita la crittografia dello spill. L'impostazione predefinita è `False`: in questo modo, i dati riversati su S3 vengono crittografati utilizzando AES-GCM tramite una chiave generata casualmente o una chiave generata mediante KMS. La disabilitazione della crittografia dello spill può migliorare le prestazioni, soprattutto se la posizione dello spill utilizza la crittografia lato server.

Esecuzione di test su database e tabelle

Il connettore TPC-DS per Athena genera un database compatibile con TPC-DS in uno dei quattro fattori di scala `tpcds1`, `tpcds10`, `tpcds100`, `tpcds250` o `tpcds1000`.

Riepilogo delle tabelle

Per un elenco completo delle tabelle e delle colonne dei dati del test, esegui le query `SHOW TABLES` o `DESCRIBE TABLE`. Per comodità, viene fornito il seguente riepilogo delle tabelle.

1. call_center
2. catalog_page
3. catalog_returns
4. catalog_sales
5. customer
6. customer_address
7. customer_demographics
8. date_dim
9. dbgen_version
10. household_demographics
11. income_band
12. Inventory
13. elemento
14. promotion
15. motivo
16. ship_mode

17. memorizzazione
18. store_returns
19. store_sales
20. time_dim
21. warehouse
22. web_page
23. web_returns
24. web_sales
25. web_site

Per le query TPC-DS compatibili con lo schema e i dati generati, consulta la directory athena-tpcds/src/main/resources/queries/ su. GitHub

Query di esempio

La query SELECT di esempio seguente interroga il catalogo `tpcds` in merito ai dati demografici dei clienti in contee specifiche.

```
SELECT
  cd_gender,
  cd_marital_status,
  cd_education_status,
  count(*) cnt1,
  cd_purchase_estimate,
  count(*) cnt2,
  cd_credit_rating,
  count(*) cnt3,
  cd_dep_count,
  count(*) cnt4,
  cd_dep_employed_count,
  count(*) cnt5,
  cd_dep_college_count,
  count(*) cnt6
FROM
  "lambda:tpcds".tpcds1.customer c, "lambda:tpcds".tpcds1.customer_address ca,
 "lambda:tpcds".tpcds1.customer_demographics
WHERE
  c.c_current_addr_sk = ca.ca_address_sk AND
    ca_county IN ('Rush County', 'Toole County', 'Jefferson County',
```

```
                  'Dona Ana County', 'La Porte County') AND
    cd_demo_sk = c.c_current_cdemo_sk AND
    exists(SELECT *
           FROM "lambda:tpcds".tpcds1.store_sales, "lambda:tpcds".tpcds1.date_dim
           WHERE c.c_customer_sk = ss_customer_sk AND
             ss_sold_date_sk = d_date_sk AND
             d_year = 2002 AND
             d_moy BETWEEN 1 AND 1 + 3) AND
    (exists(SELECT *
            FROM "lambda:tpcds".tpcds1.web_sales, "lambda:tpcds".tpcds1.date_dim
            WHERE c.c_customer_sk = ws_bill_customer_sk AND
              ws_sold_date_sk = d_date_sk AND
              d_year = 2002 AND
              d_moy BETWEEN 1 AND 1 + 3) OR
      exists(SELECT *
             FROM "lambda:tpcds".tpcds1.catalog_sales, "lambda:tpcds".tpcds1.date_dim
             WHERE c.c_customer_sk = cs_ship_customer_sk AND
               cs_sold_date_sk = d_date_sk AND
               d_year = 2002 AND
               d_moy BETWEEN 1 AND 1 + 3))
GROUP BY cd_gender,
  cd_marital_status,
  cd_education_status,
  cd_purchase_estimate,
  cd_credit_rating,
  cd_dep_count,
  cd_dep_employed_count,
  cd_dep_college_count
ORDER BY cd_gender,
  cd_marital_status,
  cd_education_status,
  cd_purchase_estimate,
  cd_credit_rating,
  cd_dep_count,
  cd_dep_employed_count,
  cd_dep_college_count
LIMIT 100
```

Autorizzazioni richieste

Consulta la sezione `Policies` del file athena-tpcds.yaml per i dettagli completi delle policy IAM richieste da questo connettore. L'elenco che segue riporta un riepilogo delle autorizzazioni richieste.

- Accesso in scrittura ad Amazon S3: per trasferire i risultati di query di grandi dimensioni, il connettore richiede l'accesso in scrittura a una posizione in Amazon S3.
- Athena GetQueryExecution: il connettore utilizza questa autorizzazione per fallire rapidamente quando la query Athena upstream è terminata.

Prestazioni

Il connettore TPC-DS per Athena tenta di condurre le query in parallelo in base al fattore di scala scelto. Il pushdown dei predicati viene eseguito all'interno della funzione Lambda.

Informazioni sulla licenza

Il progetto del connettore TPC-DS per Amazon Athena è concesso in licenza ai sensi della Licenza Apache-2.0.

Risorse aggiuntive

Per ulteriori informazioni su questo connettore, visitate il sito corrispondente su .com. GitHub

Connettore Amazon Athena Vertica

Vertica è una piattaforma di database colonnare che può essere implementata nel cloud o on-premise che supporta data warehouse con scalabilità exabyte. Puoi utilizzare il connettore Amazon Athena Vertica nelle query federate per interrogare le origini dati Vertica da Athena. Ad esempio, è possibile eseguire query analitiche su un data warehouse in Vertica e su un data lake in Amazon S3.

Prerequisiti

- Implementa il connettore sul tuo Account AWS utilizzando la console Athena o AWS Serverless Application Repository. Per ulteriori informazioni, consulta Distribuzione di un connettore origine dati o Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati.
- Prima di utilizzare questo connettore, configura un VPC e un gruppo di sicurezza. Per ulteriori informazioni, consulta Creazione di un VPC per un connettore origine dati.

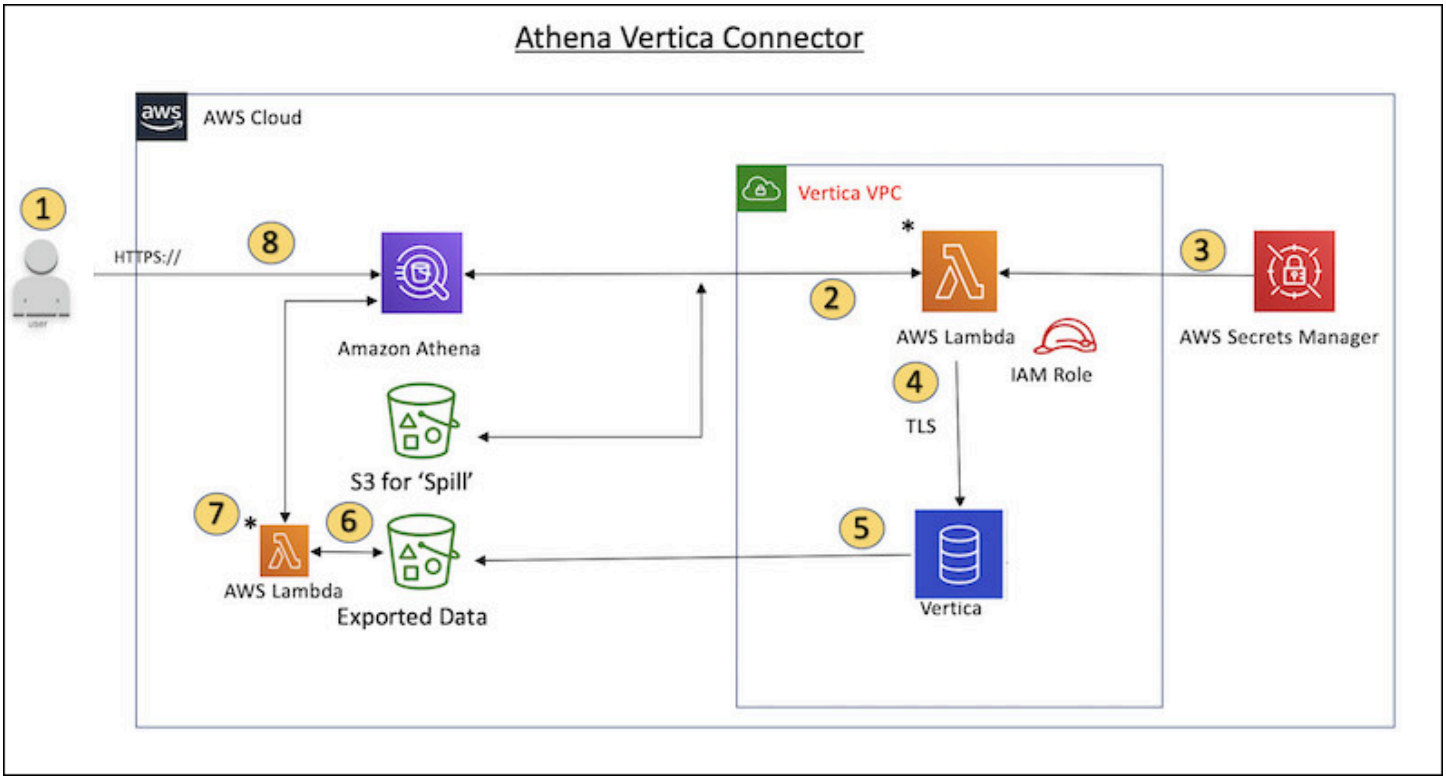
Limitazioni

- Poiché il connettore Vertica per Athena utilizza Amazon S3 Select per leggere i file Parquet da Amazon S3, le sue prestazioni potrebbero essere lente. Per eseguire una query su tabelle di grandi dimensioni, ti suggeriamo di utilizzare una query CREATE TABLE AS (SELECT ...) (Crea tabella come [seleziona...]) e predicati SQL.

- Attualmente, a causa di un problema noto in Athena Federated Query, il connettore fa sì che Vertica esporti tutte le colonne della tabella interrogata in Amazon S3, ma nei risultati sulla console di Athena risultano visibili solo le colonne interrogate.
- Le operazioni di scrittura DDL non sono supportate.
- Eventuali limiti Lambda pertinenti. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .

Flusso di lavoro

Il diagramma seguente mostra il flusso di lavoro di una query che utilizza il connettore Vertica.



1. Viene eseguita una query SQL su una o più tabelle in Vertica.
2. Il connettore analizza la query SQL per inviare la porzione pertinente a Vertica tramite la connessione JDBC.
3. Le stringhe di connessione utilizzano il nome utente e la password memorizzati AWS Secrets Manager per accedere a Vertica.
4. Il connettore esegue il wrapping della query SQL con un comando EXPORT di Vertica, come nell'esempio seguente.

```
EXPORT TO PARQUET (directory = 's3://DOC-EXAMPLE-BUCKET/folder_name,
```

```
   Compression='Snappy', fileSizeMB=64) OVER() as
SELECT
PATH_ID,
...
SOURCE_ITEMIZED,
SOURCE_OVERRIDE
FROM DELETED_OBJECT_SCHEMA.FORM_USAGE_DATA
WHERE PATH_ID <= 5;
```

5. Vertica elabora la query SQL e invia il set di risultati a un bucket Amazon S3. Per una migliore velocità di trasmissione effettiva, Vertica utilizza l'opzione EXPORT per eseguire in parallelo l'operazione di scrittura di più file Parquet.
6. Athena effettua la scansione del bucket Amazon S3 per determinare il numero di file da leggere per il set di risultati.
7. Athena effettua più chiamate alla funzione Lambda e utilizza Amazon S3 Select per leggere i file Parquet dal set di risultati. Le chiamate multiple consentono ad Athena di eseguire la lettura dei file Amazon S3 in parallelo e raggiungere una velocità di trasmissione effettiva fino a 100 GB al secondo.
8. Athena elabora i dati restituiti da Vertica con i dati scansionati dal data lake e restituisce il risultato.

Termini

I seguenti termini si riferiscono al connettore Vertica.

- Istanza del database: qualsiasi istanza del database Vertica distribuita su Amazon EC2.
- Gestore: un gestore Lambda che accede all'istanza del database. Un gestore può gestire i metadati o i record di dati.
- Gestore dei metadati: un gestore Lambda che recupera i metadati dall'istanza del database.
- Gestore dei record: un gestore Lambda che recupera i record di dati dall'istanza del database.
- Gestore composito: un gestore Lambda che recupera sia i metadati sia i record di dati dall'istanza del database.
- Proprietà o parametro: una proprietà del database utilizzata dai gestori per estrarre le informazioni del database. Queste proprietà vengono configurate come variabili di ambiente Lambda.
- Stringa di connessione: una stringa di testo utilizzata per stabilire una connessione a un'istanza del database.
- Catalogo: un AWS Glue catalogo non registrato con Athena che è un prefisso obbligatorio per la proprietà. `connection_string`

Parametri

Il connettore Vertica per Amazon Athena presenta diverse opzioni di configurazione attraverso le variabili di ambiente Lambda. Puoi utilizzare le seguenti variabili di ambiente Lambda per configurare il connettore.

- AthenaCatalogName— Nome della funzione Lambda
- ExportBucket— Il bucket Amazon S3 in cui vengono esportati i risultati delle query Vertica.
- SpillBucket— Il nome del bucket Amazon S3 in cui questa funzione può trasferire dati.
- SpillPrefix— Il prefisso della `SpillBucket` posizione in cui questa funzione può trasferire dati.
- SecurityGroupIds— Uno o più ID che corrispondono al gruppo di sicurezza da applicare alla funzione Lambda (ad esempio, `sg1sg2`, `osg3`).
- SubnetIds— Uno o più ID di sottorete che corrispondono alla sottorete che la funzione Lambda può utilizzare per accedere all'origine dati (ad esempio, `subnet1` o). `subnet2`
- SecretNameOrPrefix— Il nome o il prefisso di un set di nomi in Secrets Manager a cui questa funzione ha accesso (ad esempio,`vertica-*`)
- VerticaConnectionString— I dettagli della connessione Vertica da utilizzare di default se non è definita alcuna connessione specifica al catalogo. La stringa può utilizzare facoltativamente la AWS Secrets Manager sintassi (ad esempio,). `${secret_name}`
- ID VPC: l'ID del VPC da collegare alla funzione Lambda.

Stringa di connessione

Utilizza una stringa di connessione JDBC nel formato seguente per connetterti a un'istanza del database.

```
vertica://jdbc:vertica://host_name:port/database?user=vertica-
username&password=vertica-password
```

Utilizzo di un gestore a singola connessione

Puoi utilizzare i seguenti gestori di metadati e record a singola connessione per connetterti a una singola istanza Vertica.

Tipo di gestore	Classe
Gestore composito	`VerticaCompositeHandler`
Gestore dei metadati	`VerticaMetadataHandler`
Gestore dei record	`VerticaRecordHandler`

Parametri del gestore a singola connessione

Parametro	Descrizione
`default`	Obbligatorio. La stringa di connessione predefinita.

I gestori a singola connessione supportano una sola istanza del database e devono fornire un parametro di stringa di connessione del tipo `default`. Tutte le altre stringhe di connessione vengono ignorate.

Specifica delle credenziali

Per fornire un nome utente e una password per il database nella stringa di connessione JDBC, puoi utilizzare le proprietà della stringa di connessione o AWS Secrets Manager.

- Stringa di connessione: puoi specificare un nome utente e una password come proprietà nella stringa di connessione JDBC.

  > **Important**
  >
  > Come best practice di sicurezza, non utilizzare credenziali codificate nelle variabili di ambiente o nelle stringhe di connessione. Per informazioni su come trasferire i segreti codificati in AWS Secrets Manager, consulta Move i segreti codificati nella Guida per l'utente. AWS Secrets ManagerAWS Secrets Manager

- AWS Secrets Manager— Per utilizzare la funzionalità Athena Federated Query con, AWS Secrets Manager il VPC collegato alla funzione Lambda deve avere accesso a Internet o un endpoint VPC per connettersi a Secrets Manager.

È possibile inserire il nome di un segreto nella AWS Secrets Manager stringa di connessione JDBC. Il connettore sostituisce il nome del segreto con i valori `username` e `password` di Secrets Manager.

Per le istanze del database Amazon RDS, questo supporto è strettamente integrato. Se usi Amazon RDS, ti consigliamo vivamente di utilizzare AWS Secrets Manager la rotazione delle credenziali. Se il tuo database non utilizza Amazon RDS, archivia le credenziali come JSON nel seguente formato:

```
{"username": "${username}", "password": "${password}"}
```

Esempio di stringa di connessione con nomi dei segreti

La stringa seguente ha i nomi del segreto ${`vertica-username`} e ${`vertica-password`}.

```
vertica://jdbc:vertica://host_name:port/database?user=${vertica-username}&password=
${vertica-password}
```

Il connettore utilizza il nome del segreto per recuperare i segreti e fornire il nome utente e la password, come nell'esempio seguente.

```
vertica://jdbc:vertica://host_name:port/database?user=sample-user&password=sample-
password
```

Attualmente, il connettore Vertica riconosce le proprietà JDBC `vertica-username` e `vertica-password`.

Parametri di spill

L'SDK Lambda può riversare i dati su Amazon S3. Tutte le istanze del database a cui accede la stessa funzione Lambda riversano i dati nella stessa posizione.

Parametro	Descrizione
`spill_bucket`	Obbligatorio. Nome del bucket di spill.
`spill_prefix`	Obbligatorio. Prefisso della chiave del bucket di spill.

Parametro	Descrizione
`spill_put_request_headers`	(Facoltativo) Una mappa codificata in JSON delle intestazioni e dei valori della richiesta per la richiesta `putObject` di Amazon S3 utilizzata per lo spill (ad esempio, `{"x-amz-server-side-encryption" : "AES256"}`). Per altre possibili intestazioni, consulta il riferimento PutObjectall'API di Amazon Simple Storage Service.

Supporto dei tipi di dati

Nella tabella seguente vengono illustrati i tipi di dati supportati per il connettore Vertica.

Booleano
BigInt
Breve
Numero intero
Long
Float
Doppio
Data
Varchar
Byte
BigDecimal
TimeStamp come Varchar

Prestazioni

La funzione Lambda esegue il pushdown della proiezione per ridurre la quantità di dati scansionati dalla query. Le clausole `LIMIT` riducono la quantità di dati scansionati, ma se non viene fornito un predicato, le query `SELECT` con una clausola `LIMIT` eseguiranno la scansione di almeno 16 MB di dati. Il connettore Vertica è resiliente alla limitazione della larghezza di banda della rete dovuta alla simultaneità.

Interrogazioni passthrough

Il connettore Vertica supporta le interrogazioni passthrough. Le query passthrough utilizzano una funzione di tabella per inviare l'intera query alla fonte di dati per l'esecuzione.

Per utilizzare le query passthrough con Vertica, è possibile utilizzare la seguente sintassi:

```
SELECT * FROM TABLE(
        system.query(
            query => 'query string'
        ))
```

La seguente query di esempio invia una query a un'origine dati in Vertica. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10'
        ))
```

Informazioni sulla licenza

Utilizzando questo connettore, l'utente riconosce l'inclusione di componenti di terze parti, un elenco dei quali è disponibile nel file pom.xml relativo a questo connettore, e accetta i termini delle rispettive licenze di terze parti fornite nel file LICENSE.txt su .com. GitHub

Risorse aggiuntive

Per le informazioni sulla versione più recente del driver JDBC, consultate il file pom.xml per il connettore Vertica su .com. GitHub

Per ulteriori informazioni su questo connettore, consulta il sito corrispondente su GitHub .com e Interrogazione di un'origine dati Vertica in Amazon Athena utilizzando l'Athena Federated Query SDK nel Big Data Blog.AWS

Distribuzione di un connettore origine dati

La preparazione alla creazione di query federate è un processo in due parti: la distribuzione di un connettore origine dati della funzione Lambda e la connessione della funzione Lambda a un'origine dati. Nella prima parte del processo, viene assegnato un nome alla funzione Lambda che è possibile scegliere in seguito nella console Athena. Nella seconda parte, viene assegnato un nome al connettore a cui è possibile fare riferimento nelle query SQL.

> **Note**
>
> Per utilizzare la funzionalità Athena Federated Query con AWS Secrets Manager, devi configurare un endpoint privato Amazon VPC per Secrets Manager. Per ulteriori informazioni, consulta Creare un endpoint privato Secrets Manager VPC nella Guida dell'utente di AWS Secrets Manager .

Argomenti

Utilizzo della console Athena

Per scegliere, assegnare un nome e distribuire un connettore origine dati, utilizza le console Athena e Lambda in un processo integrato.

Per distribuire un connettore origine dati

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Nel pannello di navigazione scegli Data sources (Origini dati).
4. Nella pagina Data sources (Origini dati), scegli Create data source (Crea origine dati).
5. In Choose a data source (Scegli un'origine dati), scegli l'origine dati su cui eseguire una query con Athena, tenendo conto delle seguenti linee guida:

   - Scegli un'opzione di query federata che corrisponde all'origine dati. Athena dispone di connettori di origine dati predefiniti che è possibile configurare per origini, tra cui MySQL, Amazon DocumentDB e PostgreSQL.
   - Scegli AWS Glue Data Catalog- S3 se desideri interrogare i dati in Amazon S3 senza utilizzare un metastore Apache Hive o una delle altre opzioni di origine dati di query federate in questa pagina. Athena utilizza AWS Glue Data Catalog per archiviare i metadati e le informazioni sullo schema per le origini dati in Amazon S3. Si tratta dell'opzione di default (non federata). Per ulteriori informazioni, consulta Utilizzo AWS Glue per connettersi a sorgenti dati in Amazon S3.
   - Scegli S3 - Apache Hive metastore (Metastore Apache Hive - S3) per interrogare set di dati in Amazon S3 che utilizzano un metastore Apache Hive. Per ulteriori informazioni su questa opzione, consulta Connessione di Athena a un metastore Apache Hive.
   - Scegli Custom or shared connector (Connettore personalizzato o condiviso) se desideri creare un connettore origine dati personalizzato da utilizzare con Athena. Per informazioni sulla scrittura di un connettore origine dati, consulta Sviluppo di un connettore origine dati utilizzando l'SDK Query Federation di Athena.

   Questo tutorial sceglie Amazon CloudWatch Logs come origine dati federata.

6. Seleziona Successivo.
7. Nella pagina Enter data source details (Inserisci i dettagli dell'origine dati), per Data source name (Nome origine dati), inserisci il nome che desideri utilizzare nelle istruzioni SQL quando esegui una query sull'origine dati da Athena (ad esempio, `CloudWatchLogs`). Il nome può contenere fino a 127 caratteri e deve essere univoco all'interno dell'account. Non può essere modificato dopo la creazione. I caratteri validi sono a-z, A-z, 0-9, _ (trattino basso), @ (chiocciola) e - (trattino). I nomi `awsdatacatalog`, `hive`, `jmx` e `system` sono riservati ad Athena e non possono essere utilizzati per i nomi delle origini dati.
8. Per Lambda function (Funzione Lambda), scegli Create Lambda function (Crea funzione Lambda). La pagina delle funzioni per il connettore che hai scelto si apre nella console. AWS Lambda La pagina include informazioni dettagliate sul connettore.
9. Sotto Impostazioni applicazione, leggere la descrizione per ogni impostazione dell'applicazione e quindi inserire i valori corrispondenti alle proprie esigenze.

   Le impostazioni dell'applicazione visualizzate variano a seconda del connettore di origini dati. Le impostazioni minime richieste includono:

   - AthenaCatalogName— Un nome, in minuscolo, per la funzione Lambda che indica l'origine dati a cui è destinata, ad esempio. `cloudwatchlogs`
   - SpillBucket— Un bucket Amazon S3 nel tuo account per archiviare i dati che superano i limiti di dimensione della risposta della funzione Lambda.

   > **Note**
   >
   > I dati fuoriusciti non vengono riutilizzati nelle esecuzioni successive e possono essere eliminati in modo sicuro dopo 12 ore. Athena non elimina questi dati al posto tuo. Per gestire questi oggetti, prendi in considerazione l'aggiunta di una policy del ciclo di vita degli oggetti che elimina i dati precedenti dal bucket spill di Amazon S3. Per ulteriori informazioni, consulta Gestione del ciclo di vita dello storage nella Guida per l'utente di Amazon S3.

10. Seleziona I acknowledge that this app creates custom IAM roles and resource policies (Sono consapevole che questa app crea ruoli IAM personalizzati e policy della risorsa). Per ulteriori informazioni, scegliere il link Info (Informazioni) .
11. Seleziona Deploy (Implementa). Al termine dell'implementazione, la funzione Lambda viene visualizzata nella sezione Resources (Risorse) nella console Lambda.

Connessione all'origine dati

Dopo aver implementato il connettore origine dati nell'account, puoi connetterlo ad Athena.

Per eseguire la connessione di un'origine dati ad Athena utilizzando un connettore implementato nell'account

1. Torna alla pagina Enter data source details (Inserisci i dettagli dell'origine dati) nella console Athena.
2. Nella sezione Connection details (Dettagli di connessione), scegli l'icona di aggiornamento accanto alla casella di ricerca Select or enter a Lambda function (Seleziona o inserisci una funzione Lambda).
3. Scegli il nome della funzione appena creata nella console Lambda. Viene visualizzato l'ARN della funzione Lambda.
4. (Facoltativo) Per Tags (Tag), aggiungi coppie chiave-valore da associare a questa origine dati. Per ulteriori informazioni sui tag, consulta Assegnazione di tag alle risorse Athena.
5. Seleziona Successivo.
6. Nella pagina Review and create (Rivedi e crea), esamina i dettagli dell'origine dati, quindi scegli Create data source (Crea origine dati).
7. La sezione Data source details (Dettagli sull'origine dati) della pagina dell'origine dati mostra le informazioni relative al nuovo connettore. È ora possibile utilizzare il connettore nelle query Athena.

   Per informazioni sull'utilizzo di connettori dati nelle query, consulta Esecuzione di query federate.

Utilizzo di AWS Serverless Application Repository per distribuire un connettore origine dati

Per implementare un connettore origine dati puoi utilizzare AWS Serverless Application Repository o in alternativa puoi avviare la console Athena. Utilizza AWS Serverless Application Repository per trovare il connettore che desideri utilizzare, fornisci i parametri richiesti dal connettore e quindi implementa il connettore nell'account. Quindi, dopo aver distribuito il connettore, utilizza la console Athena per rendere disponibile l'origine dati ad Athena.

Implementazione del connettore sull'account

Per utilizzare AWS Serverless Application Repository al fine di implementare un connettore origine dati nell'account

1. Accedere a AWS Management Console e aprire Serverless App Repository (Repository app senza server).
2. Nel pannello di navigazione, scegli Available applications (Applicazioni disponibili).
3. Seleziona l'opzione Visualizzare le app che creano ruoli IAM personalizzati o policy delle risorse.
4. Nella casella di ricerca digita il nome del connettore. Per un elenco dei connettori dati Athena predefiniti, consulta Connettori di origine dati disponibili.
5. Scegliere il nome del connettore. In seguito a questa operazione, si apre la pagina della funzione Lambda Dettagli dell'applicazione nella console AWS Lambda.
6. Sul lato destro della pagina dei dettagli, inserisci le informazioni richieste in Application settings (Impostazioni dell'applicazione). Le impostazioni minime richieste includono quanto segue. Per informazioni sulle opzioni configurabili rimanenti per i connettori dati creati da Athena, consulta l'argomento Available Connectors (Connettori disponibili) corrispondente su GitHub.
   - AthenaCatalogName: un nome in minuscolo per la funzione Lambda che indica l'origine dati a cui si rivolge, ad esempio `cloudwatchlogs`.
   - SpillBucket: specifica un bucket Amazon S3 nel tuo account per ricevere dati da payload della risposta di grandi dimensioni che superano i limiti delle dimensioni della risposta della funzione Lambda.
7. Seleziona I acknowledge that this app creates custom IAM roles and resource policies (Sono consapevole che questa app crea ruoli IAM personalizzati e policy della risorsa). Per ulteriori informazioni, scegliere il link Info (Informazioni) .
8. Nella parte inferiore destra della pagina Application settings (Impostazioni dell'applicazione), scegli Deploy (Implementa). Al termine dell'implementazione, la funzione Lambda viene visualizzata nella sezione Resources (Risorse) nella console Lambda.

Rendere disponibile il connettore in Athena

A questo punto, puoi usare la console Athena per rendere disponibile il connettore origine dati per Athena.

Per rendere disponibile il connettore origine dati per Athena

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Nel pannello di navigazione scegli Data sources (Origini dati).
4. Nella pagina Data sources (Origini dati), scegli Create data source (Crea origine dati).
5. In Choose a data source (Scegli un'origine dati), scegli l'origine dati per la quale è stato creato un connettore in AWS Serverless Application Repository. Questo tutorial utilizza Amazon CloudWatch Logs come origine dati federata.
6. Seleziona Successivo.
7. Nella pagina Enter data source details (Inserisci i dettagli dell'origine dati), per Data source name (Nome origine dati), inserisci il nome che desideri utilizzare nelle istruzioni SQL quando esegui una query sull'origine dati da Athena (ad esempio, `CloudWatchLogs`). Il nome può contenere fino a 127 caratteri e deve essere univoco all'interno dell'account. Non può essere modificato dopo la creazione. I caratteri validi sono a-z, A-z, 0-9, _ (trattino basso), @ (chiocciola) e - (trattino). I nomi `awsdatacatalog`, `hive`, `jmx` e `system` sono riservati ad Athena e non possono essere utilizzati per i nomi delle origini dati.
8. Nella sezione Connection details (Dettagli di connessione), usa la casella Select or enter a Lambda function (Seleziona o inserisci una funzione Lambda) per scegliere il nome della funzione appena creata. Viene visualizzato l'ARN della funzione Lambda.
9. (Facoltativo) Per Tags (Tag), aggiungi coppie chiave-valore da associare a questa origine dati. Per ulteriori informazioni sui tag, consulta Assegnazione di tag alle risorse Athena.

10. Seleziona Successivo.
11. Nella pagina Review and create (Rivedi e crea), esamina i dettagli dell'origine dati, quindi scegli Create data source (Crea origine dati).
12. La sezione Data source details (Dettagli sull'origine dati) della pagina dell'origine dati mostra le informazioni relative al nuovo connettore. È ora possibile utilizzare il connettore nelle query Athena.

    Per informazioni sull'utilizzo di connettori dati nelle query, consulta Esecuzione di query federate.

Creazione di un VPC per un connettore origine dati

Alcuni connettori origine dati Athena richiedono un VPC e un gruppo di sicurezza. In questo argomento viene illustrato come creare un VPC con una sottorete e un gruppo di sicurezza per il VPC. Come parte di questo processo, si recuperano gli ID per il VPC, la sottorete e il gruppo di sicurezza creati. Questi ID sono necessari quando si configura il connettore per l'uso con Athena.

Creazione di un VPC da utilizzare con un connettore origine dati Athena.

1. Accedi AWS Management Console e apri la console Amazon VPC all'indirizzo https://console.aws.amazon.com/vpc/.
2. Seleziona Crea VPC.
3. Nella pagina Crea VPC, in Impostazioni VPC, per Risorse da creare, scegli VPC e altro.
4. In Generazione automatica di tag di nome, per Generazione automatica, inserisci un valore che verrà utilizzato per generare tag di nome per tutte le risorse nel tuo VPC.
5. Seleziona Crea VPC.
6. Al termine del processo, scegli Visualizza VPC.
7. Nella sezione Details (Dettagli), alla voce VPC ID (ID VPC), copia l'ID VPC per riferimento futuro.

Ora è tutto pronto per recuperare l'ID della sottorete per il VPC appena creato.

Recupero dell'ID della sottorete VPC

1. Nel pannello di navigazione della console del VPC, seleziona Subnets (Sottoreti).
2. Seleziona il nome di una sottorete la cui colonna VPC ha l'ID VPC che hai annotato.
3. Nella sezione Dettagli, alla voce ID sottorete, copia il tuo ID sottorete per un riferimento successivo.

Successivamente, crea un gruppo di sicurezza per la VPC.

Creazione di un gruppo di sicurezza per il proprio VPC

1. Nel pannello di navigazione della console del VPC; seleziona Sicurezza, Gruppi di sicurezza.
2. Scegliere Create Security Group (Crea gruppo di sicurezza).
3. Nella pagina Create Security Group (Crea gruppo di sicurezza) immettere le seguenti informazioni:
   - In Security group name (Nome gruppo di sicurezza), inserisci un nome per il tuo gruppo di sicurezza.
   - In Description (Descrizione), inserisci una breve descrizione per il tuo gruppo di sicurezza. È richiesta una descrizione.
   - Per VPC, scegli l'ID VPC del VPC che hai creato per il connettore di origine dati.
   - In Inbound rules (Regole in entrata) e Outbound rules (Regole in uscita), aggiungi tutte le regole in entrata e in uscita necessarie.
4. Scegliere Create Security Group (Crea gruppo di sicurezza).
5. Nella pagina Details (Dettagli) per il gruppo di sicurezza, copia la voce nel campo Security group ID (ID gruppo di sicurezza) per riferimento successivo.

Abilitazione delle query federate tra account

La query federata consente di eseguire query su origini dati diverse da Amazon S3 utilizzando connettori di origine dati implementati su AWS Lambda. La funzione di query federata tra account consente alla funzione Lambda e alle origini dati che devono essere sottoposte a query di trovarsi in account diversi.

In qualità di amministratore dei dati, puoi abilitare query federate tra più account condividendo il connettore dati con l'account di un analista di dati o, in qualità di analista di dati, utilizzando un ARN Lambda condiviso da un amministratore di dati da aggiungere al tuo account. Quando vengono apportate modifiche di configurazione a un connettore nell'account di origine, la configurazione aggiornata viene automaticamente applicata alle istanze condivise del connettore negli account di altri utenti.

Considerazioni e limitazioni

- La funzione di query federata tra account è disponibile per i connettori dati di metastore non Hive che utilizzano un'origine dati basata su Lambda.
- La funzionalità non è disponibile per il tipo di origine AWS Glue Data Catalog dati. Per informazioni sull'accesso a AWS Glue Data Catalog s da più account, vedereAccesso tra account ai cataloghi dati AWS Glue.
- Se la risposta della funzione Lambda del connettore supera il limite di dimensione della risposta Lambda di 6 MB, Athena crittografa, raggruppa e trasferisce automaticamente la risposta a un bucket Amazon S3 da te configurato. L'entità che esegue la query Athena deve avere accesso al luogo dello sversamento affinché Athena possa leggere i dati fuoriusciti. Ti consigliamo di impostare una policy del ciclo di vita di Amazon S3 per eliminare gli oggetti dal luogo di fuoriuscita, poiché i dati non sono necessari dopo il completamento della query.
- L'utilizzo di query federate su più server Regioni AWS non è supportato.

Autorizzazioni richieste

- Affinché l'account A dell'amministratore dei dati condivida una funzione Lambda con l'account B dell'analista dati, l'account B richiede la funzione di richiamo Lambda e l'accesso al bucket spill. Di conseguenza, l'account A dovrebbe aggiungere una policy basata sulle risorse alla funzione Lambda e un accesso principale al relativo bucket spill in Amazon S3.
  1. La seguente policy concede a Lambda le autorizzazioni di richiamo funzione all'account B su una funzione Lambda nell'account A.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CrossAccountInvocationStatement",
      "Effect": "Allow",
      "Principal": {
        "AWS": ["arn:aws:iam::account-B-id:user/username"]
      },
      "Action": "lambda:InvokeFunction",
      "Resource": "arn:aws:lambda:aws-region:account-A-id:function:lambda-function-
name"
    }
  ]
```

```
}
```

2. La seguente policy consente l'accesso del bucket spill al principale nell'account B.

```
{
    "Version": "2008-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": ["arn:aws:iam::account-B-id:user/username"]
            },
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
             ],
            "Resource": [
                "arn:aws:s3:::spill-bucket",
                "arn:aws:s3:::spill-bucket/*"
            ]
        }
     ]
 }
```

3. Se la funzione Lambda crittografa lo spill bucket con una AWS KMS chiave anziché la crittografia predefinita offerta dall'SDK di federazione, la politica AWS KMS chiave nell'Account A deve concedere l'accesso all'utente nell'Account B, come nell'esempio seguente.

```
{
    "Sid": "Allow use of the key",
    "Effect": "Allow",
    "Principal":
    {
       "AWS": ["arn:aws:iam::account-B-id:user/username"]
    },
    "Action": [ "kms:Decrypt" ],
    "Resource": "*" // Resource policy that gets placed on the KMS key.
 }
```

- Affinché l'Account A condivida il connettore con l'Account B, l'Account B deve creare un ruolo chiamato `AthenaCrossAccountCreate-account-A-id` che l'Account A assume chiamando l' AWS azione API Security Token Service. AssumeRole

La seguente policy, che consente l'operazione `CreateDataCatalog`, deve essere creata nell'account B e aggiunta al ruolo `AthenaCrossAccountCreate-`*`account-A-id`* creato dall'account B per l'account A.

```
{
 "Effect": "Allow",
 "Action": "athena:CreateDataCatalog",
 "Resource": "arn:aws:athena:*:account-B-id:datacatalog/*"
}
```

Condivisione di un'origine dati nell'account A con l'account B

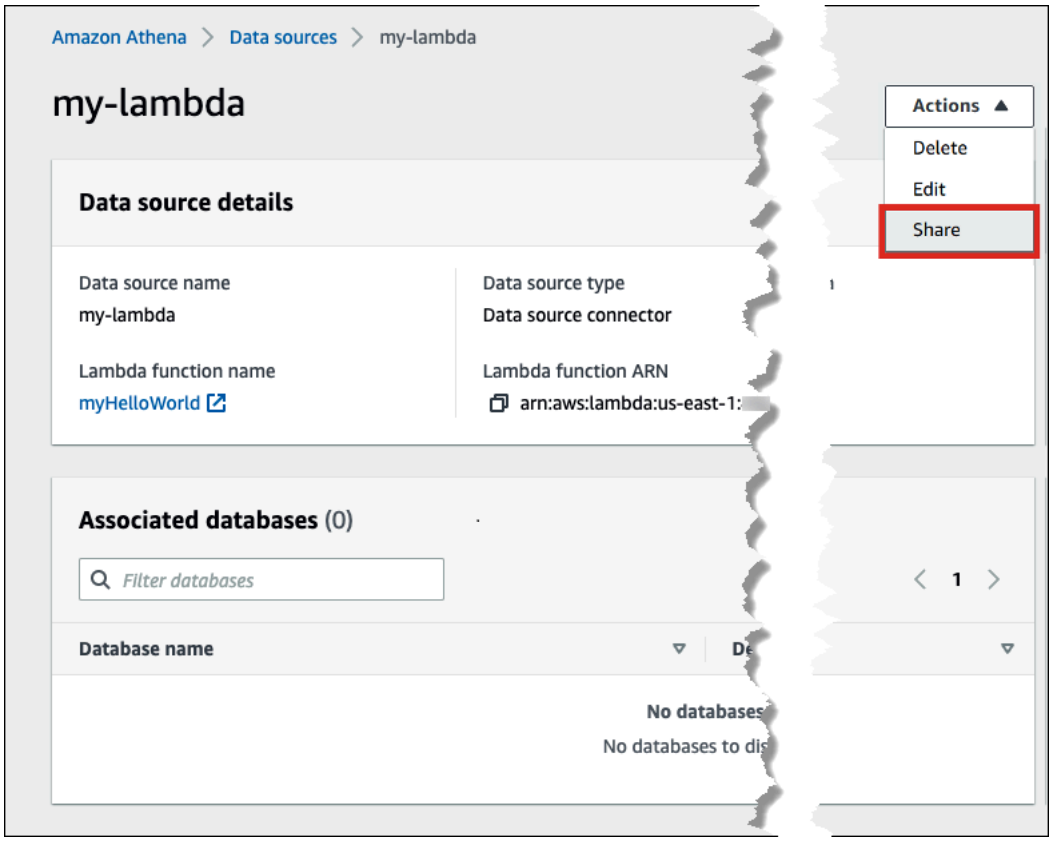
Dopo che le autorizzazioni sono state impostate, è possibile utilizzare la pagina Data sources (Origini dati) nella console Athena per condividere un connettore dati nel tuo account (account A) con un altro account (account B). L'account A mantiene il pieno controllo e la proprietà del connettore. Quando l'account A apporta modifiche di configurazione al connettore, la configurazione aggiornata si applica al connettore condiviso nell'account B.

Per condividere un'origine dati Lambda nell'account A con l'account B

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

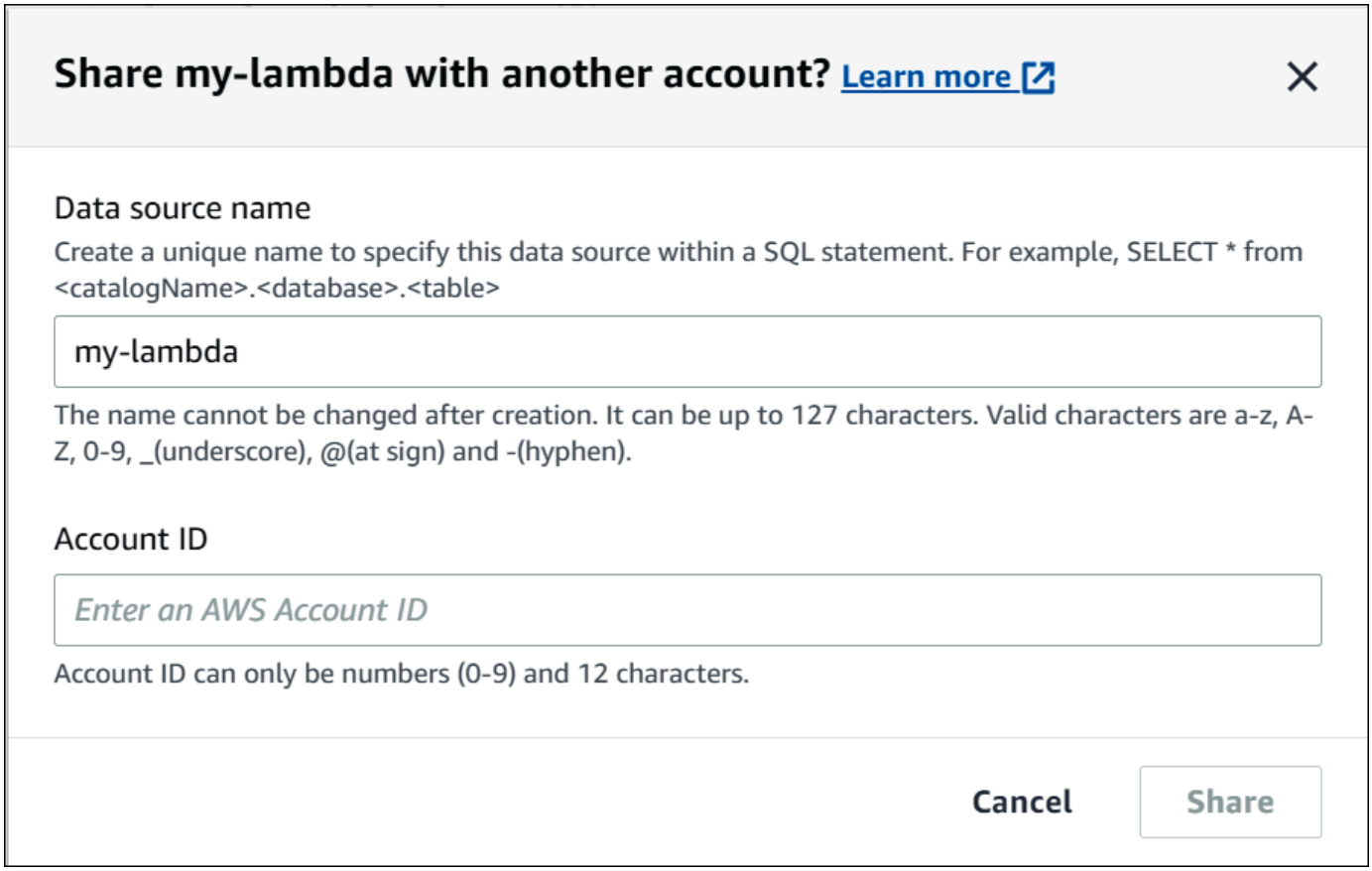
3. Scegli Data sources (Origini dati).
4. Sulla pagina Data sources (Origini dati), scegli il collegamento del connettore da condividere.

5. Nella pagina dei dettagli relativa a un'origine dati Lambda, scegli l'opzione Share (Condividi) nell'angolo in alto a destra.



6. Nella finestra di dialogo Share ***Lambda-name*** with another account (Condividi nome-Lambda con un altro account), inserisci le informazioni richieste.

   - Per Data source name (Nome origine dati), inserisci il nome dell'origine dati copiata come desideri che appaia nell'altro account.
   - Per Account ID (ID account), inserisci l'ID dell'account con cui desideri condividere l'origine dati (in questo caso, l'account B).



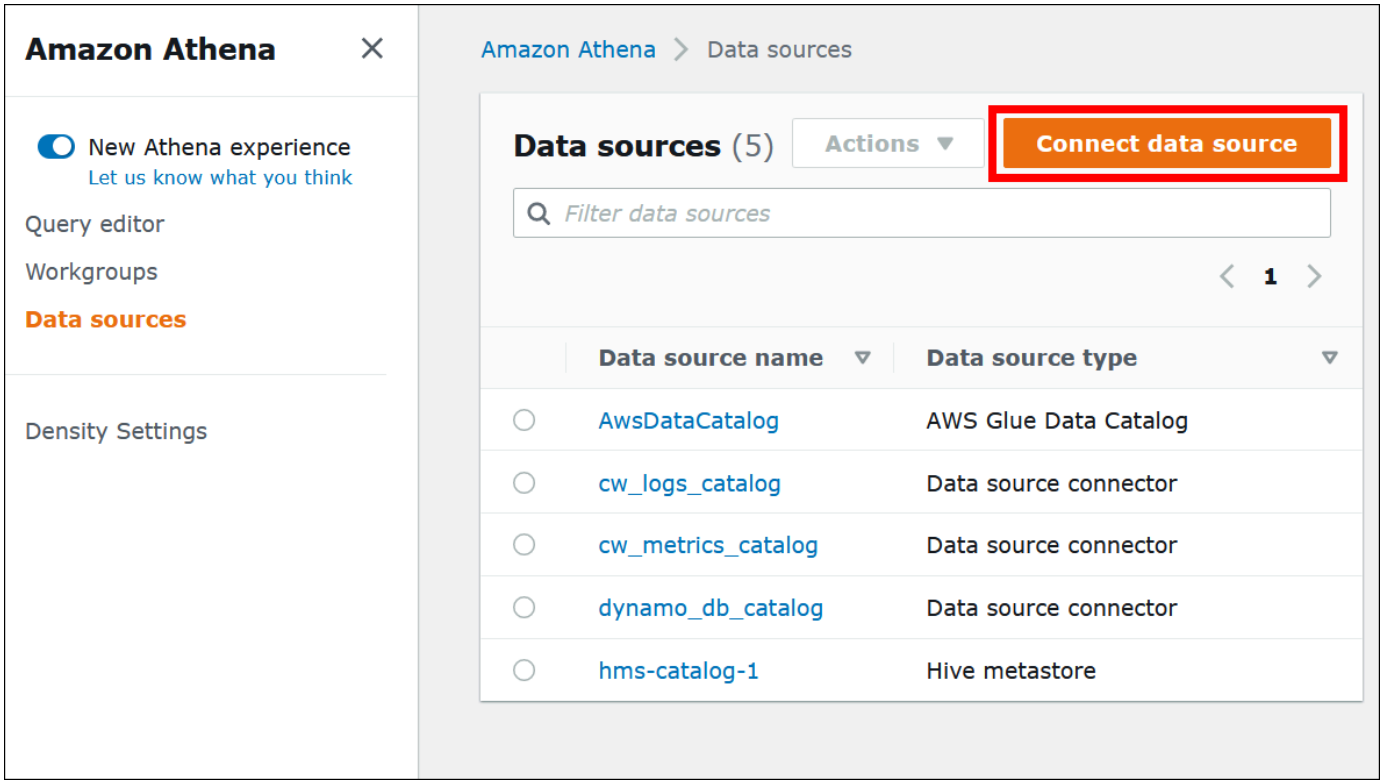
7. Scegli Condividi. Il connettore dati condiviso specificato viene creato nell'account B. Le modifiche di configurazione apportate al connettore nell'account A si applicano al connettore nell'account B.

Aggiunta di un'origine dati condivisa dall'account A all'account B

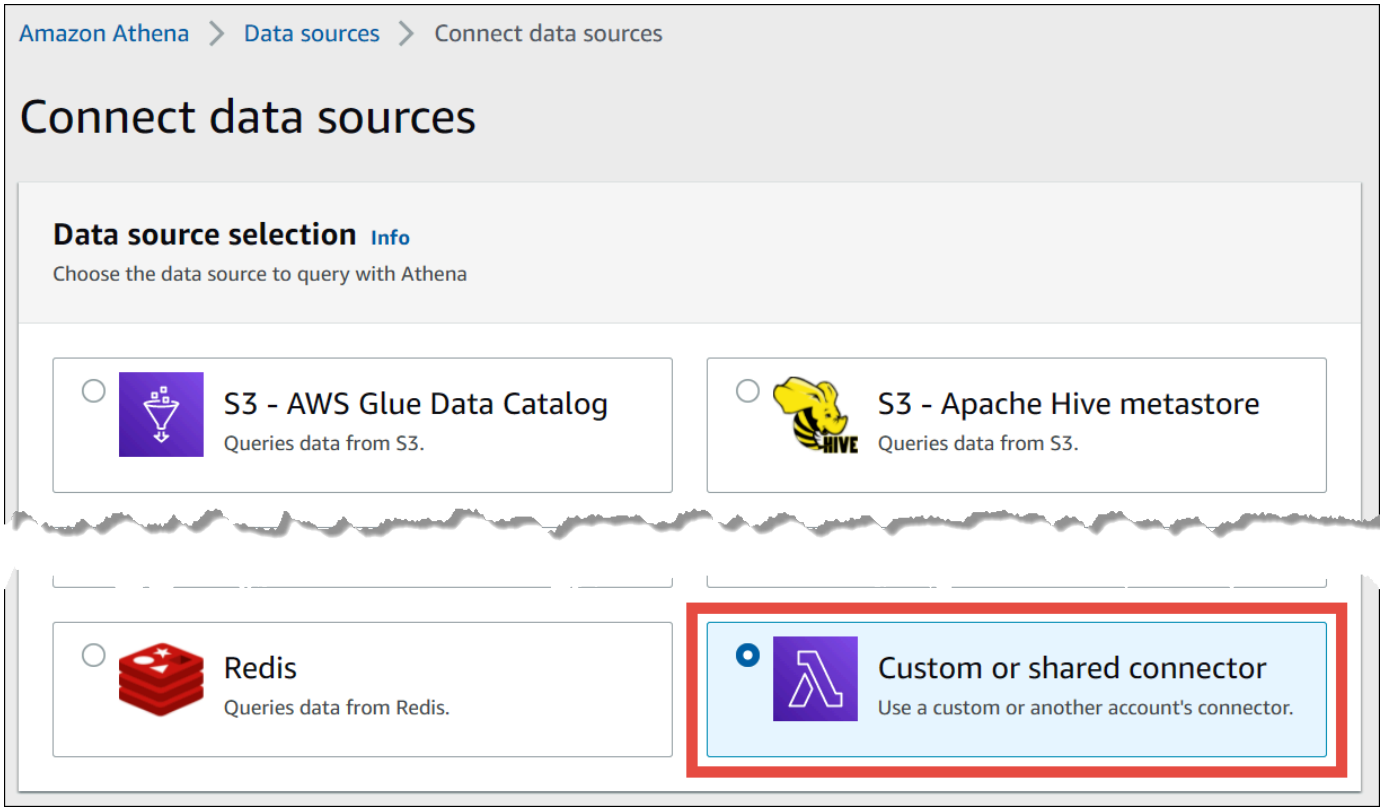
In qualità di analista di dati, potresti ricevere l'ARN di un connettore da aggiungere al tuo account da un amministratore dei dati. Puoi utilizzare la pagina Data sources (Origini dati) della console Athena per aggiungere l'ARN Lambda fornito dall'amministratore al tuo account.

Per aggiungere l'ARN Lambda di un connettore dati condiviso al tuo account

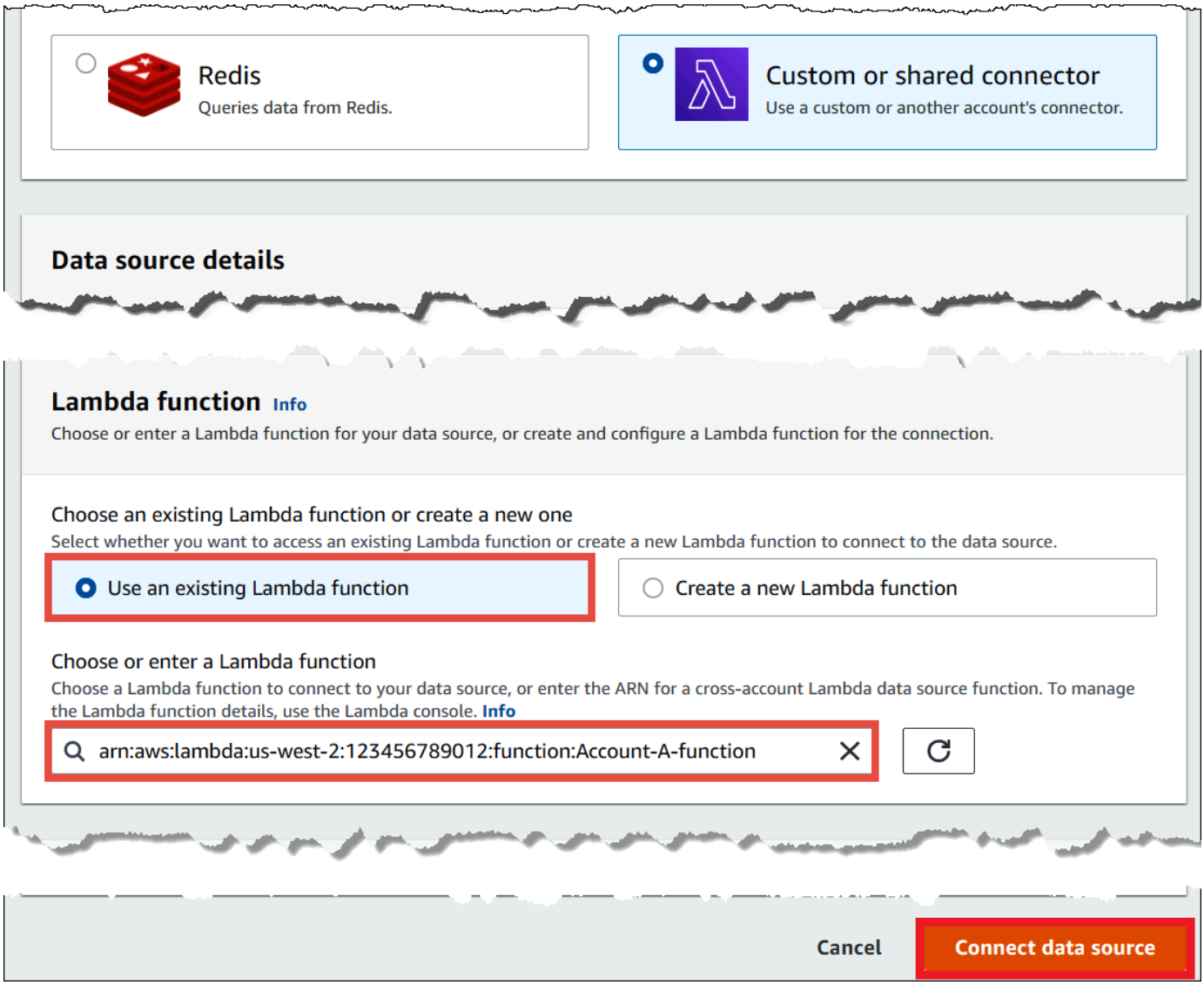
1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se si utilizza la nuova console e il pannello di navigazione non è visibile, scegli il menu di espansione a sinistra.
3. Scegli Data sources (Origini dati).
4. Nella scheda Data sources (Origini dati), scegli Connect data source (Connetti origine dati).



5. Scegli Custom or shared connector (Connettore personalizzato o condiviso).



6. Nella sezione Lambda function (Funzione Lambda), assicurarsi che l'opzione Use an existing Lambda function (Utilizza una funzione Lambda esistente) sia selezionata.



7. Per Choose or enter a Lambda function (Scegli o inserisci una funzione Lambda), inserire l'ARN Lambda dell'account A.
8. Scegliere Connect data source (Connetti origine dati).

Risoluzione dei problemi

Se viene visualizzato un messaggio di errore che indica che l'account A non dispone delle autorizzazioni per assumere un ruolo nell'account B, assicurarsi che il nome del ruolo creato nell'account B sia scritto correttamente e che la policy appropriata sia collegata.

Aggiornamento di un connettore origine dati

Athena consiglia di aggiornare regolarmente i connettori di origine dati utilizzati alla versione più recente per sfruttare le nuove funzionalità e miglioramenti. Per iniziare, devi individuare il numero della versione più recente.

Come trovare la versione più recente di Athena Query Federation

Il numero di versione più recente dei connettori di origine dati di Athena Data corrisponde all'ultima versione di Athena Query Federation. In alcuni casi, le versioni di GitHub possono essere leggermente più recenti di quelle disponibili su AWS Serverless Application Repository (SAR).

Come trovare il numero di versione più recente di Athena Query Federation

1. Visita l'URL GitHub https://github.com/awslabs/aws-athena-query-federation/releases/latest.
2. Prendi nota del numero della versione nell'intestazione della pagina principale nel seguente formato:

   Rilascio v *anno*.*week_of_year*.*iteration_of_week* di Athena Query Federation

   Ad esempio, il numero di rilascio per la versione di rilascio v2023.8.3 di Athena Query Federation è 2023.8.3.

Individuazione e annotazione dei nomi delle risorse

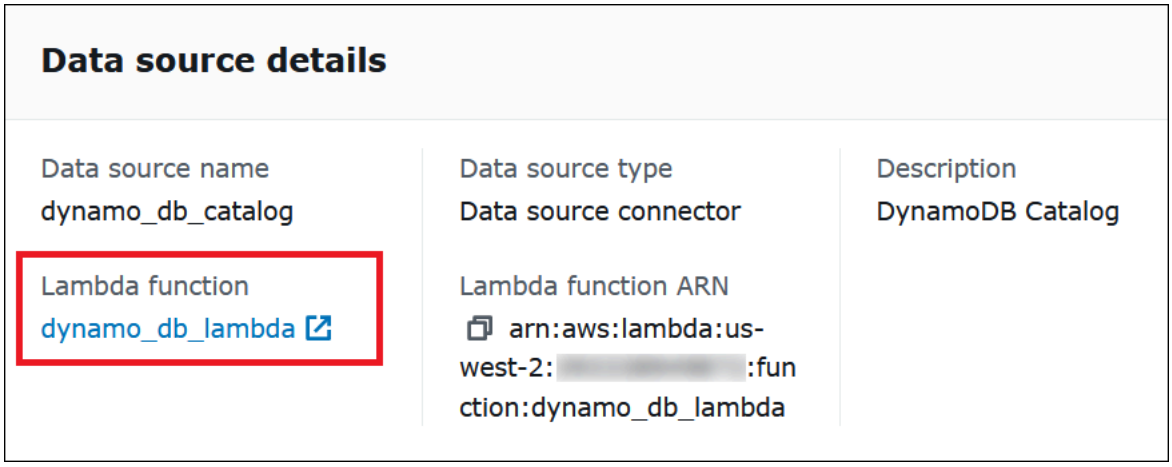
Per prepararti all'aggiornamento, individua e prendi nota delle seguenti informazioni:

1. il nome della funzione Lambda del connettore.
2. Variabili di ambiente della funzione Lambda.
3. il nome dell'applicazione Lambda, che gestisce la funzione Lambda del connettore.

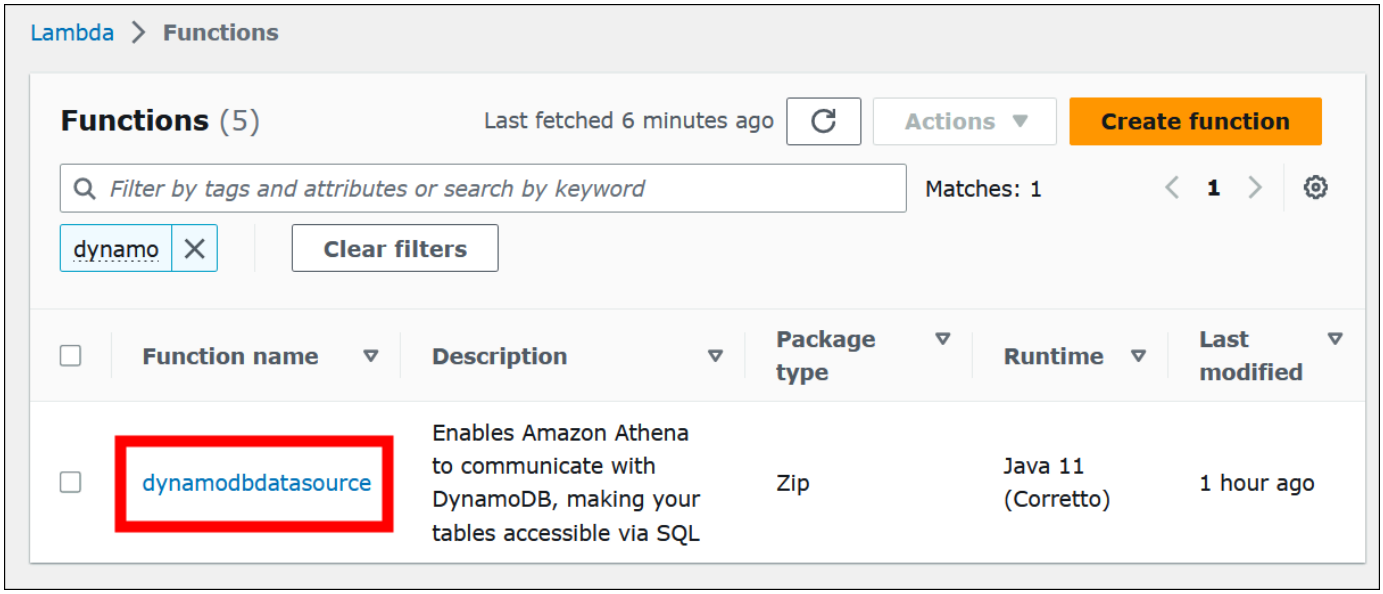
Come trovare i nomi delle risorse dalla console Athena

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Nel pannello di navigazione scegli Data sources (Origini dati).
4. Nella colonna Nome origine dati, seleziona il link dell'origine dati del tuo connettore.
5. Nella sezione Dettagli origine dati, in Funzione Lambda, seleziona il link alla tua funzione Lambda.



6. Nella pagina Funzioni, nella colonna Nome funzione, annota il nome della funzione per il connettore.



7. Scegli il link del nome della funzione.
8. Nella sezione Panoramica funzioni, seleziona la scheda Configurazione.
9. Nel riquadro a sinistra, seleziona Variabili di ambiente.
10. Nella sezione Variabili ambiente, prendi nota delle chiavi e dei valori corrispondenti.
11. Scorri fino alla parte superiore della pagina.
12. Nel messaggio Questa funzione appartiene a un'applicazione. Fai clic qui per gestirlo, seleziona il link Fai clic qui.
13. Nella pagina serverlessrepo-***your_application_name***, prendi nota del nome dell'applicazione senza serverlessrepo. Ad esempio, se il nome dell'applicazione è serverlessrepo-DynamoDbTestApp, il nome della tua applicazione sarà DynamoDbTestApp.
14. Resta sulla pagina della console Lambda dell'applicazione, quindi continua con i passaggi descritti in Individuazione della versione del connettore in uso.

Individuazione della versione del connettore in uso

Segui questi passaggi per trovare la versione del connettore in uso.

Come trovare la versione del connettore in uso

1. Nella pagina della console Lambda dell'applicazione Lambda, seleziona la scheda Implementazioni.
2. Nella scheda Distribuzioni espandi il modello SAM.

3. Cerca CodeUri.
4. Nel campo Chiave sotto CodeUri, individua la seguente stringa:

```
applications-connector_name-
versions-year.week_of_year.iteration_of_week/hash_number
```

   L'esempio seguente mostra una stringa del connettore CloudWatch:

```
applications-AthenaCloudwatchConnector-versions-2021.42.1/15151159...
```

5. Registra il valore per *year.week_of_year.iteration_of_week* (ad esempio 2021.42.1). Questa è la versione del tuo connettore.

Implementazione della nuova versione del connettore

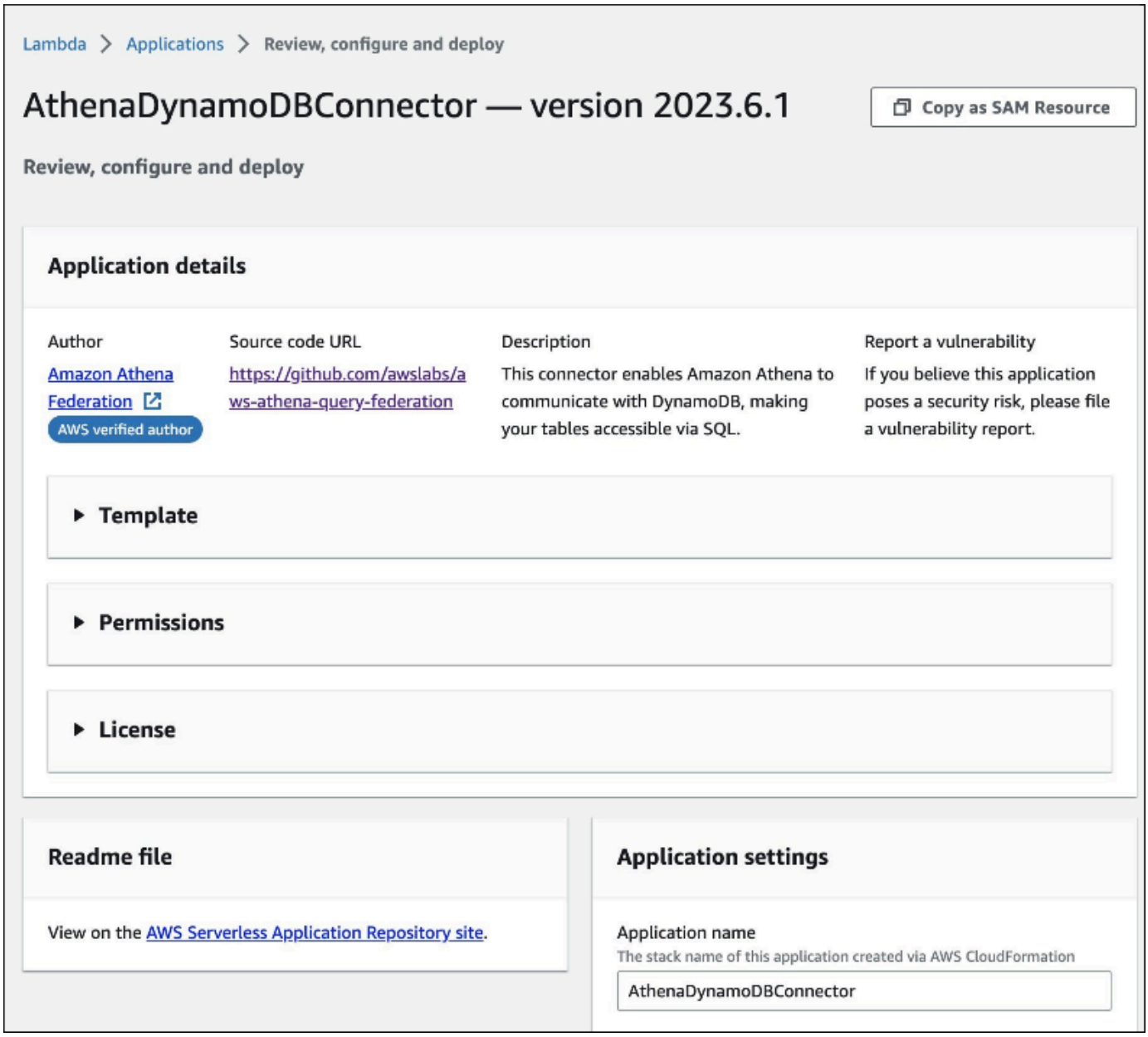
Segui questi passaggi per implementare una nuova versione del connettore.

Come implementare una nuova versione del connettore

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Nel pannello di navigazione scegli Data sources (Origini dati).
4. Nella pagina Data sources (Origini dati), scegli Create data source (Crea origine dati).
5. Seleziona l'origine dati che intendi aggiornare, quindi seleziona Avanti.

6. Nella sezione Dettagli connessione, seleziona Crea funzione Lambda. Si apre la console Lambda dove potrai implementare l'applicazione aggiornata.



7. Poiché di fatto non stai creando una nuova origine dati, puoi chiudere la scheda della console Athena.
8. Nella pagina della console Lambda del connettore, esegui le seguenti operazioni:

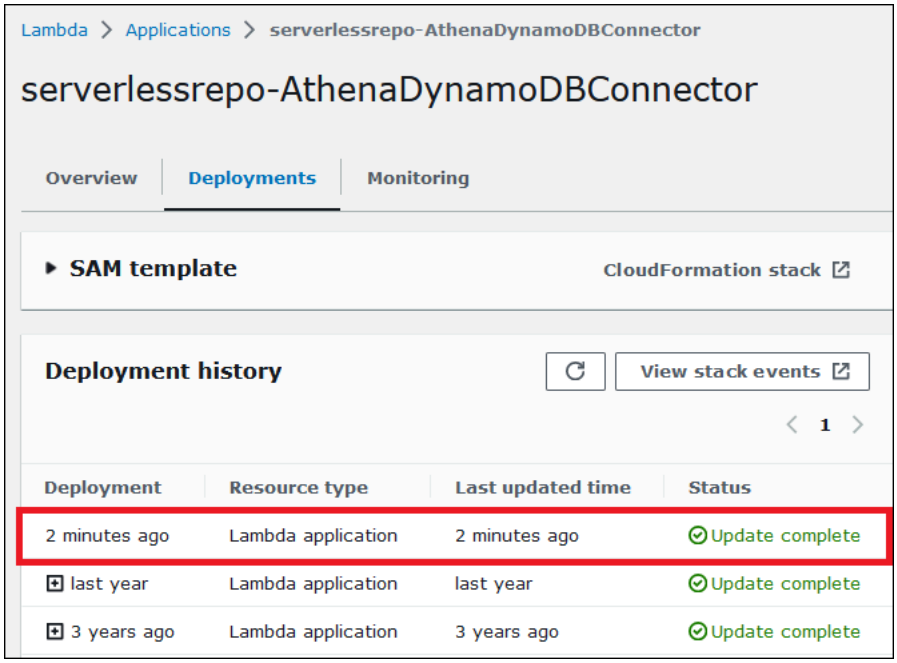
   a. Assicurati di aver rimosso il prefisso serverlessrepo- dal nome dell'applicazione, quindi copia il nome dell'applicazione nel campo Nome applicazione.

   b. Copia il nome della funzione Lambda nel campo AthenaCatalogName. Alcuni connettori invocano questo campo LambdaFunctionName.

c. Copia le variabili di ambiente registrate nei campi corrispondenti.

9. Seleziona l'opzione Sono consapevole che questa app crea ruoli IAM personalizzati e policy delle risorse, quindi scegli Implementa.
10. Per verificare che l'applicazione sia stata aggiornata, seleziona la scheda Implementazioni.

    La sezione Cronologia delle implementazioni mostra che l'aggiornamento è completo.



11. Per confermare il nuovo numero di versione, puoi espandere il modello SAM come fatto prima, individuare CodeUri e controllare il numero di versione del connettore nel campo Chiave.

Ora puoi usare il connettore aggiornato per creare query federate Athena.

Esecuzione di query federate

Dopo aver configurato uno o più connettori dati e averli distribuiti nell'account, puoi utilizzarli nelle query Athena.

Esecuzione di query su una singola origine dati

Negli esempi di questa sezione si presuppone che Connettore Amazon Athena CloudWatch sia stato configurato e distribuito nel tuo account. Utilizza lo stesso approccio per eseguire query quando utilizzi altri connettori.

Per creare una query Athena che utilizza il connettore CloudWatch

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Nell'editor di query Athena, creare una query SQL che utilizza la sintassi seguente nella clausola `FROM`.

```
MyCloudwatchCatalog.database_name.table_name
```

Esempi

L'esempio seguente utilizza il CloudWatch connettore Athena per connettersi alla `all_log_streams` vista nel gruppo `/var/ecommerce-engine/order-processor` CloudWatch Logs Log. La visualizzazione `all_log_streams` è una visualizzazione di tutti i flussi di log nel gruppo di log. La query di esempio limita il numero di righe restituite a 100.

```
SELECT *
FROM "MyCloudwatchCatalog"."/var/ecommerce-engine/order-processor".all_log_streams
LIMIT 100;
```

Nell'esempio seguente vengono analizzate le informazioni della stessa visualizzazione dell'esempio precedente. Nell'esempio vengono estratti l'ID ordine e il livello di log e filtrati tutti i messaggi che dispongono del livello `INFO`.

```
SELECT
    log_stream as ec2_instance,
    Regexp_extract(message '.*orderId=(\d+) .*', 1) AS orderId,
    message AS order_processor_log,
    Regexp_extract(message, '(.*):.*', 1) AS log_level
FROM MyCloudwatchCatalog."/var/ecommerce-engine/order-processor".all_log_streams
WHERE Regexp_extract(message, '(.*):.*', 1) != 'INFO'
```

Esecuzione di query su più origini dati

Come esempio più complesso, immagina un'azienda di e-commerce che utilizza le seguenti fonti di dati per archiviare i dati relativi agli acquisti dei clienti:

- Amazon RDS per MySQL per archiviare i dati del catalogo dei prodotti
- Amazon DocumentDB per archiviare dati dell'account del cliente come e-mail e indirizzi di spedizione

- Amazon DynamoDB per archiviare i dati di spedizione e tracciamento degli ordini

Immagina che un analista di dati per questa applicazione di e-commerce scopra che i tempi di spedizione in alcune regioni sono stati influenzati dalle condizioni meteorologiche locali. L'analista desidera sapere quanti ordini subiscono ritardi, dove si trovano i clienti interessati e quali prodotti sono maggiormente interessati. Invece di esaminare separatamente le fonti di informazioni, l'analista utilizza Athena per unire i dati in un'unica query federata.

Example

```
SELECT
     t2.product_name AS product,
     t2.product_category AS category,
     t3.customer_region AS region,
     count(t1.order_id) AS impacted_orders
FROM my_dynamodb.default.orders t1
JOIN my_mysql.products.catalog t2 ON t1.product_id = t2.product_id
JOIN my_documentdb.default.customers t3 ON t1.customer_id = t3.customer_id
WHERE
     t1.order_status = 'PENDING'
     AND t1.order_date between '2022-01-01' AND '2022-01-05'
GROUP BY 1, 2, 3
ORDER BY 4 DESC
```

Esecuzione di query su visualizzazioni federate

Quando si interrogano fonti federate, è possibile utilizzare le viste per offuscare le fonti di dati sottostanti o nascondere join complessi ad altri analisti che eseguono query sui dati.

Considerazioni e limitazioni

- Le viste federate richiedono la versione 3 del motore Athena.
- Le viste federate vengono archiviate nella fonte di dati sottostante AWS Glue, non con essa.
- Le viste create con cataloghi federati devono utilizzare una sintassi dei nomi completa, come nell'esempio seguente:

  ```
  "ddbcatalog"."default"."customers"
  ```

- Gli utenti che eseguono query su fonti federate devono disporre dell'autorizzazione per poterle eseguire.

- L'autorizzazione `athena:GetDataCatalog` è richiesta per le visualizzazioni federate. Per ulteriori informazioni, consulta Esempio di policy di autorizzazione IAM per consentire la query federata Athena.

Esempi

L'esempio seguente crea una visualizzazione chiamata `customers` sui dati archiviati in un'origine dati federata.

Example

```
CREATE VIEW customers AS
SELECT *
FROM my_federated_source.default.table
```

La seguente query di esempio mostra una query che fa riferimento alla visualizzazione `customers` anziché all'origine dati federata sottostante.

Example

```
SELECT id, SUM(order_amount)
FROM customers
GROUP by 1
ORDER by 2 DESC
LIMIT 50
```

L'esempio seguente crea una visualizzazione chiamata `order_summary` che combina i dati provenienti da un'origine dati federata e da un'origine dati Amazon S3. Dall'origine federata, che è già stata creata in Athena, la visualizzazione utilizza `person` e le tabelle `profile`. Da Amazon S3, la visualizzazione utilizza le tabelle `purchase` e `payment`. Per fare riferimento ad Amazon S3, l'istruzione utilizza la parola chiave `awsdatacatalog`. Tieni presente che l'origine dati federata utilizza la sintassi del nome completa *`federated_source_name`*.*`federated_source_database`*.*`federated_source_table`*

Example

```
CREATE VIEW default.order_summary AS
SELECT *
FROM federated_source_name.federated_source_database."person" p
```

```
    JOIN federated_source_name.federated_source_database."profile" pr ON pr.id = p.id
    JOIN awsdatacatalog.default.purchase i ON p.id = i.id
    JOIN awsdatacatalog.default.payment pay ON pay.id = p.id
```

Risorse aggiuntive

- Per un esempio di vista federata disaccoppiata dalla fonte di origine e disponibile per l'analisi su richiesta in un modello multi-utente, consulta Enstensione di data mesh con Amazon Athena e visualizzazioni federate nel Blog sui Big Data di AWS .
- Per ulteriori informazioni sull'utilizzo delle visualizzazioni in Athena, consulta Utilizzo delle visualizzazioni.

Esecuzione di query passthrough federate

In Athena, puoi eseguire query su origini dati federate utilizzando il linguaggio di query dell'origine dati stessa e inviare l'intera query all'origine dati per l'esecuzione. Queste interrogazioni sono chiamate interrogazioni passthrough. Per eseguire interrogazioni passthrough, si utilizza una funzione di tabella nella query Athena. Includi la query passthrough da eseguire sull'origine dati in uno degli argomenti della funzione di tabella. Le query Pass Through restituiscono una tabella che è possibile analizzare utilizzando Athena SQL.

Connettori supportati

I seguenti connettori di origine dati Athena supportano le query passthrough.

- Archiviazione Azure Data Lake
- Azure Synapse
- Cloudera Hive
- Cloudera Impala
- CloudWatch
- Db2
- Db2 iSeries
- DocumentDB
- DynamoDB
- HBase
- Google BigQuery

- Hortonworks
- MySQL
- Neptune
- OpenSearch
- Oracle
- PostgreSQL
- Redshift
- SAP HANA
- Snowflake
- SQL Server
- Teradata
- Timestream
- Vertica

Considerazioni e limitazioni

Quando utilizzate le interrogazioni passthrough in Athena, tenete presente i seguenti punti:

- Il passthrough delle query è supportato solo per le istruzioni `SELECT` Athena o le operazioni di lettura.
- Le query passthrough devono essere eseguite nel contesto del catalogo della query esterna (ovvero la query che chiama la funzione di tabella).
- Le prestazioni delle query possono variare a seconda della configurazione dell'origine dati.
- Le query passthrough non sono supportate per le visualizzazioni.

Sintassi

La sintassi generale della query passthrough di Athena è la seguente.

```
SELECT * FROM TABLE(system.function_name(arg1 => 'arg1Value'[, arg2 =>
 'arg2Value', ...]))
```

Per la maggior parte delle fonti di dati, il primo e unico argomento è `query` seguito dall'operatore freccia => e dalla stringa di query.

```
SELECT * FROM TABLE(system.query(query => 'query string'))
```

Per semplicità, è possibile omettere l'argomento denominato opzionale `query` e l'operatore => freccia.

```
SELECT * FROM TABLE(system.query('query string'))
```

Se l'origine dati richiede più della stringa di query, utilizzate argomenti denominati nell'ordine previsto dall'origine dati. Ad esempio, l'espressione *`arg1`* `=>` `'`*`arg1Value`*`'` contiene il primo argomento e il relativo valore. Il nome *`arg1`* è specifico dell'origine dati e può differire da connettore a connettore.

```
SELECT * FROM TABLE(
        system.query(
            arg1 => 'arg1Value',
            arg2 => 'arg2Value',
            arg3 => 'arg3Value'
        ));
```

Per informazioni sulla sintassi esatta da utilizzare con un particolare connettore, consulta la pagina dei singoli connettori.

Utilizzo delle virgolette

I valori degli argomenti, inclusa la stringa di query passata, devono essere racchiusi tra virgolette singole, come nell'esempio seguente.

```
SELECT * FROM TABLE(system.query(query => 'SELECT * FROM testdb.persons LIMIT 10'))
```

Quando la stringa di query è racchiusa tra virgolette doppie, la query ha esito negativo. La seguente query ha esito negativo e restituisce il messaggio di errore COLUMN_NOT_FOUND: riga 1:43: La colonna 'select * from testdb.persons limit 10' non può essere risolta.

```
SELECT * FROM TABLE(system.query(query => "SELECT * FROM testdb.persons LIMIT 10"))
```

Per evitare una singola virgoletta, aggiungi una singola citazione all'originale (ad esempio, a). `terry's_group` `terry''s_group`

Esempi

La seguente query di esempio invia una query a una fonte di dati. La query seleziona tutte le colonne della `customer` tabella, limitando i risultati a 10.

```
SELECT * FROM TABLE(
        system.query(
            query => 'SELECT * FROM customer LIMIT 10;'
        ))
```

L'istruzione seguente esegue la stessa query, ma elimina l'argomento denominato opzionale `query` e l'operatore freccia. =>

```
SELECT * FROM TABLE(
        system.query(
            'SELECT * FROM customer LIMIT 10;'
        ))
```

Athena e i qualificatori dei nomi delle tabelle federate

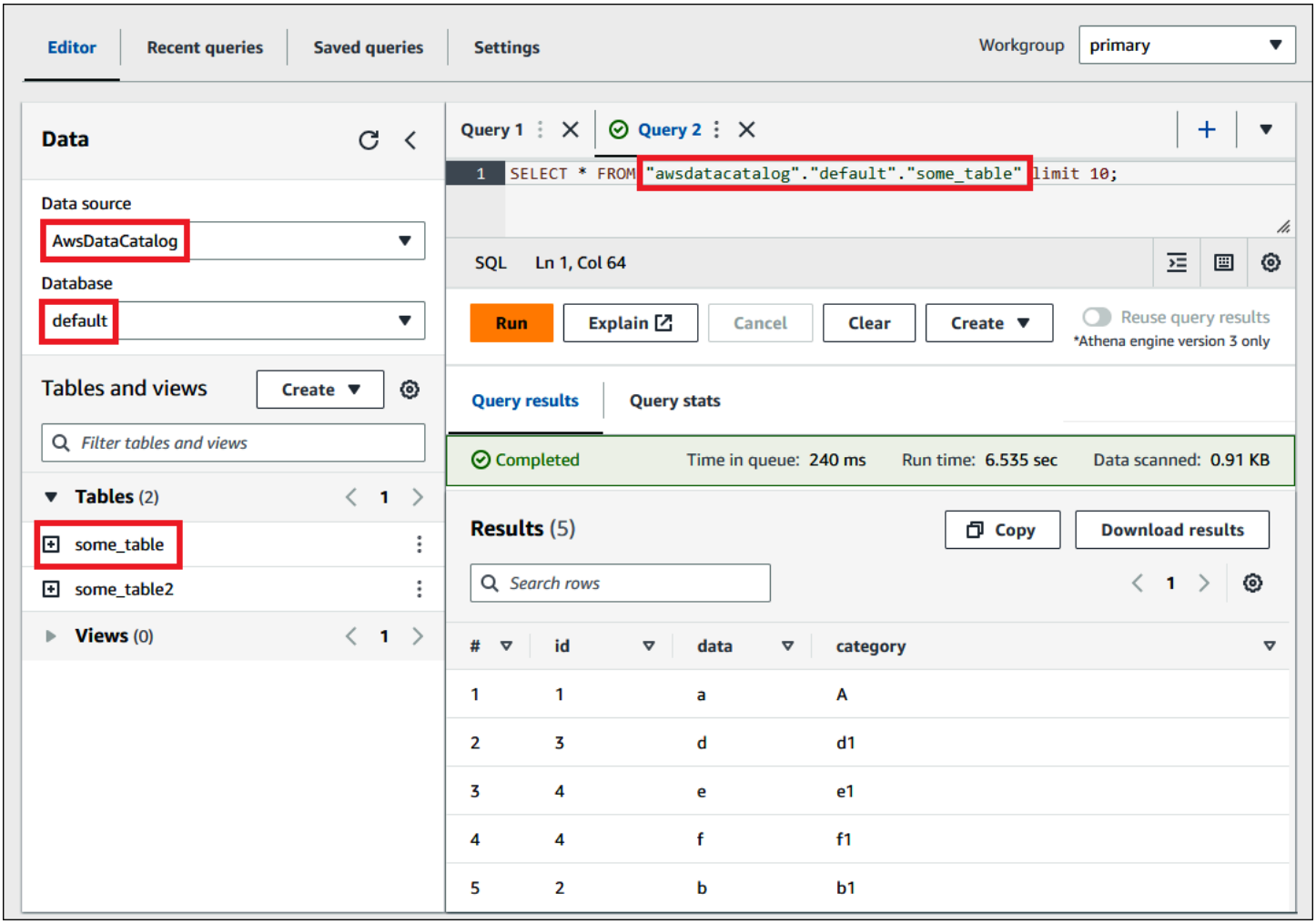
Athena utilizza i seguenti termini per fare riferimento alle gerarchie di oggetti dati:

- Origine dati: un gruppo di database
- Database: un gruppo di tabelle
- Tabella: dati organizzati come gruppo di righe o colonne

A volte si fa riferimento a questi oggetti anche con nomi alternativi ma equivalenti, come i seguenti:

- Un'origine dati talvolta viene definita catalogo.
- Un database a volte viene chiamato schema.

La seguente query di esempio nella console Athena utilizza l'origine dati `awsdatacatalog`, il database `default` e la tabella `some_table`.



Termini nelle fonti di dati federate

Quando esegui una query su origini dati federate, tieni presente che l'origine dati sottostante potrebbe non utilizzare la stessa terminologia di Athena. Tieni presente questa distinzione quando scrivi le tue query federate. Le sezioni seguenti descrivono in che modo i termini relativi agli oggetti dati in Athena corrispondono a quelli delle origini dati federate.

Amazon Redshift

Un database Amazon Redshift è un gruppo di schemi Redshift che contiene un gruppo di tabelle Redshift.

Athena	Redshift
Origine dati Redshift	Una funzione Lambda del connettore Redshift configurata per puntare a un `database` Redshift.
`data_source.database.table`	`database.schema.table`

Query di esempio

```
SELECT * FROM
Athena_Redshift_connector_data_source.Redshift_schema_name.Redshift_table_name
```

Per ulteriori informazioni su questo connettore, consulta Connettore Redshift di Amazon Athena.

Cloudera Hive

Un server o cluster Cloudera Hive è un gruppo di database Cloudera Hive che contiene un gruppo di tabelle Cloudera Hive.

Athena	Hive
Origine dati Cloudera Hive	Funzione Lambda del connettore Cloudera Hive configurata per puntare a un `server` Cloudera Hive.
`data_source.database.table`	`server.database.table`

Query di esempio

```
SELECT * FROM
Athena_Cloudera_Hive_connector_data_source.Cloudera_Hive_database_name.Cloudera_Hive_table_name
```

Per ulteriori informazioni su questo connettore, consulta Connettore Amazon Athena per Cloudera Hive.

Cloudera Impala

Un server o cluster Impala è un gruppo di database Impala che contiene un gruppo di tabelle Impala.

Athena	Impala
Origine dati Impala	Funzione Lambda del connettore Impala configurata per puntare a un `server` Impala.
`data_source.database.table`	`server.database.table`

Query di esempio

```
SELECT * FROM
Athena_Impala_connector_data_source.Impala_database_name.Impala_table_name
```

Per ulteriori informazioni su questo connettore, consulta Connettore Amazon Athena per Cloudera Impala.

MySQL

Un server MySQL è un gruppo di database MySQL che contiene un gruppo di tabelle MySQL.

Athena	MySQL
Origine dati MySQL	Funzione Lambda del connettore MySQL configurata per puntare a un `server` MySQL.
`data_source.database.table`	`server.database.table`

Query di esempio

```
SELECT * FROM
Athena_MySQL_connector_data source.MySQL_database_name.MySQL_table_name
```

Per ulteriori informazioni su questo connettore, consulta Connettore MySQL di Amazon Athena.

Oracle

Un server (o database) Oracle è un gruppo di schemi Oracle che contiene un gruppo di tabelle Oracle.

Athena	Oracle
Origine dati Oracle	Funzione Lambda del connettore Oracle configurata per puntare a un `server` Oracle.
`data_source.database.table`	`server.schema.table`

Query di esempio

```
SELECT * FROM
Athena_Oracle_connector_data_source.Oracle_schema_name.Oracle_table_name
```

Per ulteriori informazioni su questo connettore, consulta Connettore Amazon Athena per Oracle.

Postgres

Un server (o cluster) Postgres è un gruppo di database Postgres. Un database Postgres è un gruppo di schemi Postgres che contiene un gruppo di tabelle Postgres.

Athena	Postgres
Origine dati Postgres	Funzione Lambda del connettore Postgres configurata per puntare a un `server` e un `database` Postgres.
`data_source.database.table`	`server.database.schema.table`

Query di esempio

```
SELECT * FROM
Athena_Postgres_connector_data_source.Postgres_schema_name.Postgres_table_name
```

Per ulteriori informazioni su questo connettore, consulta Connettore PostgreSQL di Amazon Athena.

Sviluppo di un connettore origine dati utilizzando l'SDK Query Federation di Athena

Per scrivere i propri connettori origine dati, puoi utilizzare l'SDK Athena Query Federation.. L'SDK Athena Query Federation definisce un insieme di interfacce e protocolli di collegamento che puoi utilizzare per consentire ad Athena di delegare parti del suo piano di esecuzione della query al codice scritto e distribuito. L'SDK include una suite di connettori e un connettore di esempio.

È inoltre possibile personalizzare i connettori preconfigurati di Amazon Athena per il proprio uso personale. Puoi modificare una copia del codice sorgente da GitHub e quindi utilizzare lo strumento di pubblicazione Connector per creare il tuo pacchetto. AWS Serverless Application Repository Dopo aver distribuito il connettore in questo modo, puoi utilizzarlo nelle query Athena.

Per informazioni su come scaricare l'SDK e istruzioni dettagliate per scrivere un connettore personalizzato, consulta Example Athena connector on. GitHub

Connettori origine dati di Athena per Apache Spark

Alcuni connettori di origine dati Athena sono disponibili come connettori Spark DSV2. I nomi dei connettori Spark DSV2 hanno come suffisso `-dsv2` (ad esempio `athena-dynamodb-dsv2`).

Di seguito sono riportati i connettori DSV2 attualmente disponibili, il relativo nome della classe `.format()` Spark e i link alla documentazione corrispondente di Amazon Athena Federated Query:

Connettore DSV2	Spark .format() nome classe	Documentazione
athena-cloudwatch-dsv2	`com.amazonaws.athena.connectors.dsv2.cloudwatch.CloudwatchTableProvider`	CloudWatch
athena-cloudwatch-metrics-dsv2	`com.amazonaws.athena.connectors.dsv2.cloudwatch.metrics.CloudwatchMetricsTableProvider`	CloudWatch metriche
athena-aws-	`com.amazonaws.athena.connectors.dsv2`	CMDB

Connettore DSV2	Spark .format() nome classe	Documentazione
cmdb-dsv2	`.aws.cmdb.AwsCmdbTableProvider`	
athena-dynamodb-dsv2	`com.amazonaws.athena.connectors.dsv2.dynamodb.DDBTableProvider`	DynamoDB

Per scaricare **`.jar`** i file per i connettori DSV2, visita la pagina GitHub DSV2 di Amazon Athena Query Federation e consulta la sezione Releases, Release, Assets. <version>

Specificazione del jar a Spark

Per utilizzare i connettori Athena DSV2 con Spark, devi inviare il file `.jar` del connettore all'ambiente Spark che stai utilizzando. Le sezioni seguenti descrivono casi specifici.

Athena per Spark

Per informazioni sull'aggiunta di file `.jar` personalizzati e configurazioni personalizzate ad Amazon Athena per Apache Spark, consulta Aggiungere file JAR e configurazione Spark personalizzata.

General Spark

Per passare il file `.jar` del connettore a Spark, usa il comando `spark-submit` e specifica il file `.jar` nell'opzione `--jars`, come nell'esempio seguente:

```
spark-submit \
  --deploy-mode cluster \
  --jars https://github.com/awslabs/aws-athena-query-federation-dsv2/releases/
download/some_version/athena-dynamodb-dsv2-some_version.jar
```

Spark di Amazon EMR

Per eseguire un comando `spark-submit` con il parametro `--jars` su Amazon EMR, devi aggiungere un passaggio al cluster Spark di Amazon EMR. Per informazioni dettagliate su come utilizzare `spark-submit` su Amazon EMR, consulta la Aggiungi fase Spark nella Guida di rilascio di Amazon EMR.

AWS Glue ETL Spark

Per AWS Glue ETL, puoi passare l'URL GitHub .com del `.jar` file all'`--extra-jars`argomento del comando. `aws glue start-job-run` La AWS Glue documentazione descrive il `--extra-jars` parametro come se accettasse un percorso Amazon S3, ma il parametro può anche accettare un URL HTTPS. Per ulteriori informazioni, consulta la Documentazione di riferimento dei parametri dei processi nella Guida per gli sviluppatori di AWS Glue .

Esecuzione di query del connettore su Spark

Per inviare l'equivalente della tua query esistente federata di Athena su Apache Spark, utilizza la funzione `spark.sql()`. Ad esempio, supponi che intendi utilizzare la seguente query Athena su Apache Spark.

```
SELECT somecola, somecolb, somecolc
FROM ddb_datasource.some_schema_or_glue_database.some_ddb_or_glue_table
WHERE somecola > 1
```

Per eseguire la stessa query su Spark utilizzando il connettore DynamoDB DSV2 di Amazon Athena, utilizza il codice seguente:

```
dynamoDf = (spark.read
    .option("athena.connectors.schema", "some_schema_or_glue_database")
    .option("athena.connectors.table", "some_ddb_or_glue_table")
    .format("com.amazonaws.athena.connectors.dsv2.dynamodb.DDBTableProvider")
    .load())

dynamoDf.createOrReplaceTempView("ddb_spark_table")

spark.sql('''
SELECT somecola, somecolb, somecolc
FROM ddb_spark_table
WHERE somecola > 1
''')
```

Come specificare parametri

Le versioni DSV2 dei connettori di origine dati Athena utilizzano gli stessi parametri dei connettori corrispondenti di origine dati Athena. Per informazioni sui parametri, consulta la documentazione del connettore corrispondente di origine dati Athena.

Nel PySpark codice, usa la seguente sintassi per configurare i parametri.

```
spark.read.option("athena.connectors.conf.parameter", "value")
```

Ad esempio, il codice seguente imposta il parametro `disable_projection_and_casing` del connettore DynamoDB di Amazon Athena su `always`.

```
dynamoDf = (spark.read
    .option("athena.connectors.schema", "some_schema_or_glue_database")
    .option("athena.connectors.table", "some_ddb_or_glue_table")
    .option("athena.connectors.conf.disable_projection_and_casing", "always")
    .format("com.amazonaws.athena.connectors.dsv2.dynamodb.DDBTableProvider")
    .load())
```

Policy IAM per l'accesso ai cataloghi dati

Per controllare l'accesso ai cataloghi dati, usa le autorizzazioni IAM a livello di risorsa o le policy IAM basate sull'identità.

La seguente procedura è specifica di Athena.

Per informazioni specifiche su IAM, segui i collegamenti elencati alla fine di questa sezione. Per informazioni sulle policy di esempio relative al catalogo dati JSON, consulta Policy di esempio del catalogo dati.

Per utilizzare l'editor visivo nella console IAM per creare una policy del catalogo dati.

1. Accedi AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione sulla sinistra, selezionare Policies (Policy) e fare clic su Create Policy (Crea policy).
3. Nella scheda Visual editor (Editor visivo), selezionare Choose a Service (Scegli un servizio). Quindi selezionare Athena per aggiungerlo alla policy.
4. Scegliere Select actions (Seleziona operazioni), quindi scegliere le operazioni da aggiungere alla policy. L'editor visivo mostra le operazioni disponibili in Athena. Per ulteriori informazioni, consulta Operazioni, risorse e chiavi di condizione per Amazon Athena nella Documentazione di riferimento per l'autorizzazione ai servizi.
5. Scegliere add actions (aggiungi operazioni) per immettere un'operazione oppure usare i caratteri jolly (*) per specificare più operazioni.

Come impostazione predefinita, la policy che si sta creando utilizza le operazioni selezionate. Se si selezionano una o più operazioni che supportano le autorizzazioni a livello di risorsa per la risorsa `datacatalog` in Athena, l'editor elenca la risorsa `datacatalog`

6. Selezionare Resources (Risorse) per specificare i cataloghi dati per la policy. Per le policy di esempio relative al catalogo dati JSON, consultare Policy di esempio del catalogo dati.
7. Specificare la risorsa `datacatalog` come segue:

```
arn:aws:athena:<region>:<user-account>:datacatalog/<datacatalog-name>
```

8. Scegliere Review policy (Rivedi policy) e digitare i valori per Name (Nome) e Description (Descrizione) (facoltativa) per la policy che si sta creando. Esaminare il riepilogo della policy per accertarsi di disporre delle autorizzazioni desiderate.
9. Seleziona Crea policy per salvare la nuova policy.
10. Collegare questa policy basata sull'identità a un utente, un gruppo o un ruolo e specificare le risorse `datacatalog` a cui possono accedere.

Per ulteriori informazioni, consulta gli argomenti seguenti nella Referenza sull'autorizzazione del servizio e la Guida per l'utente di IAM:

- Operazioni, risorse e chiavi di condizione per Amazon Athena
- Creazione di policy con l'editor visivo
- Aggiunta e rimozione delle policy IAM
- Controllo dell'accesso alle risorse

Per le policy di esempio relative al catalogo dati JSON, consultare Policy di esempio del catalogo dati.

Per informazioni sulle AWS Glue autorizzazioni e sulle autorizzazioni del AWS Glue crawler, consulta Configurazione delle autorizzazioni IAM AWS Glue e dei prerequisiti del crawler nella Guida per gli sviluppatori.AWS Glue

Per un elenco completo delle operazioni Amazon Athena, consulta i nomi delle operazioni API nella documentazione di riferimento dell'API Amazon Athena.

Policy di esempio del catalogo dati

Questa sezione include policy di esempio che puoi utilizzare per abilitare varie operazioni sui cataloghi dati.

Un catalogo dati è una risorsa IAM gestita da Athena. Pertanto, se la policy del catalogo dati utilizza operazioni che accettano `datacatalog` come input, devi specificare l'ARN del catalogo dati nel modo seguente:

```
"Resource": [arn:aws:athena:<region>:<user-account>:datacatalog/<datacatalog-name>]
```

`<datacatalog-name>` è il nome del catalogo dati. Ad esempio, per un catalogo dati denominato `test_datacatalog`, specificalo come risorsa nel modo seguente:

```
"Resource": ["arn:aws:athena:us-east-1:123456789012:datacatalog/test_datacatalog"]
```

Per un elenco completo delle operazioni Amazon Athena, consulta i nomi delle operazioni API nella documentazione di riferimento dell'API Amazon Athena. Per ulteriori informazioni sulle policy IAM, consulta Creazione di policy con l'editor visivo nella Guida per l'utente di IAM. Per ulteriori informazioni sulla creazione di policy IAM per i gruppi di lavoro, consulta Policy IAM per l'accesso ai cataloghi dati.

- Example Policy for Full Access to All Data Catalogs
- Example Policy for Full Access to a Specified Data Catalog
- Example Policy for Querying a Specified Data Catalog
- Example Policy for Management Operations on a Specified Data Catalog
- Example Policy for Listing Data Catalogs
- Example Policy for Metadata Operations on Data Catalogs

Example Esempio di policy per l'accesso completo a tutti i cataloghi dati

La policy seguente consente l'accesso completo a tutte le risorse del catalogo dati che potrebbero essere presenti nell'account. Ti consigliamo di utilizzare questa policy per gli utenti nell'account che devono amministrare e gestire cataloghi dati per tutti gli altri utenti.

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

```
      {
         "Effect":"Allow",
         "Action":[
            "athena:*"
         ],
         "Resource":[
            "*"
         ]
      }
   ]
}
```

Example Esempio di policy per l'accesso completo a un catalogo dati specifico

La policy seguente consente l'accesso completo a una specifica risorsa del catalogo dati denominata `datacatalogA`. Puoi usare questa policy per gli utenti con controllo completo su un determinato catalogo dati.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "athena:ListDataCatalogs",
            "athena:ListWorkGroups",
            "athena:GetDatabase",
            "athena:ListDatabases",
            "athena:ListTableMetadata",
            "athena:GetTableMetadata"
         ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "athena:StartQueryExecution",
            "athena:GetQueryResults",
            "athena:DeleteNamedQuery",
            "athena:GetNamedQuery",
            "athena:ListQueryExecutions",
            "athena:StopQueryExecution",
            "athena:GetQueryResultsStream",
```

```
            "athena:ListNamedQueries",
            "athena:CreateNamedQuery",
            "athena:GetQueryExecution",
            "athena:BatchGetNamedQuery",
            "athena:BatchGetQueryExecution",
            "athena:DeleteWorkGroup",
            "athena:UpdateWorkGroup",
            "athena:GetWorkGroup",
            "athena:CreateWorkGroup"
         ],
         "Resource":[
            "arn:aws:athena:us-east-1:123456789012:workgroup/*"
         ]
      },
      {
        "Effect":"Allow",
        "Action":[
            "athena:CreateDataCatalog",
            "athena:DeleteDataCatalog",
            "athena:GetDataCatalog",
            "athena:GetDatabase",
            "athena:GetTableMetadata",
            "athena:ListDatabases",
            "athena:ListTableMetadata",
            "athena:UpdateDataCatalog"
        ],
        "Resource":"arn:aws:athena:us-east-1:123456789012:datacatalog/datacatalogA"
      }
   ]
}
```

Example Esempio di policy per l'esecuzione di query su un catalogo dati specifico

Nella policy seguente, a un utente è consentito eseguire query nel gruppo `datacatalogA` specificato. All'utente non è consentito eseguire attività di gestione per il catalogo dati, ad esempio l'aggiornamento o l'eliminazione.

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

```
            "athena:StartQueryExecution"
         ],
         "Resource":[
            "arn:aws:athena:us-east-1:123456789012:workgroup/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "athena:GetDataCatalog"
         ],
         "Resource":[
            "arn:aws:athena:us-east-1:123456789012:datacatalog/datacatalogA"
         ]
      }
   ]
}
```

Example Esempio di policy per le operazioni di gestione in un catalogo dati specifico

Nel policy seguente, un utente può creare, eliminare, ottenere i dettagli e aggiornare un catalogo dati `datacatalogA`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:CreateDataCatalog",
                "athena:GetDataCatalog",
                "athena:DeleteDataCatalog",
                "athena:UpdateDataCatalog"
            ],
            "Resource": [
                "arn:aws:athena:us-east-1:123456789012:datacatalog/datacatalogA"
            ]
        }
    ]
}
```

Example Esempio di policy per elencare cataloghi dati

La policy seguente consente a tutti gli utenti per elencare tutti i cataloghi dati:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:ListDataCatalogs"
            ],
            "Resource": "*"
        }
    ]
}
```

Example Esempio di policy per le operazioni relative a metadati sui cataloghi dati

La seguente policy consente operazioni relative ai metadati sui cataloghi dati:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:GetDatabase",
                "athena:GetTableMetadata",
                "athena:ListDatabases",
                "athena:ListTableMetadata"
            ],
            "Resource": "*"
        }
    ]
}
```

Gestione delle origini dati

Puoi utilizzare la pagina Data Sources (Origini dati) della console Athena per gestire le origini dati create.

Per visualizzare un'origine dati

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Nel pannello di navigazione scegli Data sources (Origini dati).
4. Dall'elenco delle origini dati scegli il nome dell'origine dati da visualizzare.

> **Note**
>
> Gli elementi nella colonna Data source name (Nome origine dati) corrispondono all'output dell'operazione API ListDataCatalogs e al comando della CLI list-data-catalogs.

Per modificare un'origine dati

1. Nella pagina Data sources (Origini dati), procedi come segue:
   - Selezionare il pulsante accanto al nome del catalogo, quindi scegli Actions (Operazioni), Edit (Modifica).
   - Scegli il nome del dell'origine dati. Quindi, nella pagina dei dettagli, scegli Actions (Operazioni), Edit (Modifica).
2. Nella pagina Edit (Modifica), è possibile scegliere una funzione Lambda diversa per l'origine dati, modificare la descrizione o aggiungere tag personalizzati. Per ulteriori informazioni sui tag, consulta Assegnazione di tag alle risorse Athena.

3. Seleziona Salva.
4. Per modificare l'origine dati AwsDataCatalog, scegli il link AwsDataCatalog per aprire la relativa pagina dei dettagli. Quindi, nella pagina dei dettagli, seleziona il link alla console AWS Glue in cui è possibile modificare il catalogo.

Per condividere un'origine dati

Per informazioni sulla condivisione di origini dati, vedi i link seguenti.

- Per origini dati basate su LAMBDA non Hive, vedi Abilitazione delle query federate tra account.
- Per i AWS Glue Data Catalog, vedi Accesso tra account ai cataloghi dati AWS Glue.

Per eliminare un'origine dati

1. Nella pagina Data sources (Origini dati), procedi come segue:

   - Seleziona il pulsante accanto al nome del catalogo, quindi scegli Actions (Operazioni), Edit (Modifica).
   - Scegli il nome dell'origine dati, quindi, nella pagina dei dettagli, scegli Actions (Operazioni), Delete (Elimina).

   > **Note**
   >
   > AwsDataCatalog è l'origine dati di default nel tuo account e non può essere eliminata.

   Viene visualizzato un avviso per ricordare che quando si elimina un'origine dati, il catalogo dati, le tabelle e le visualizzazioni corrispondenti vengono rimossi dall'editor della query. Le query salvate che hanno utilizzato l'origine dati non vengono più eseguite in Athena.
2. Per confermare l'eliminazione, digita il nome dell'origine dati, quindi seleziona Delete (Elimina).

Utilizzo di Amazon DataZone in Athena

Puoi usare Amazon DataZone per condividere, cercare e rilevare dati su larga scala oltre i confini dell'organizzazione. DataZone semplifica la tua esperienza con servizi di analisi AWS come Athena, AWS Glue e AWS Lake Formation. Ad esempio, se disponi di petabyte di dati in diverse origini dati,

puoi utilizzare Amazon DataZone per creare raggruppamenti di persone, dati e strumenti basati su casi d'uso aziendali. Per ulteriori informazioni, consulta Cos'è Amazon DataZone?.

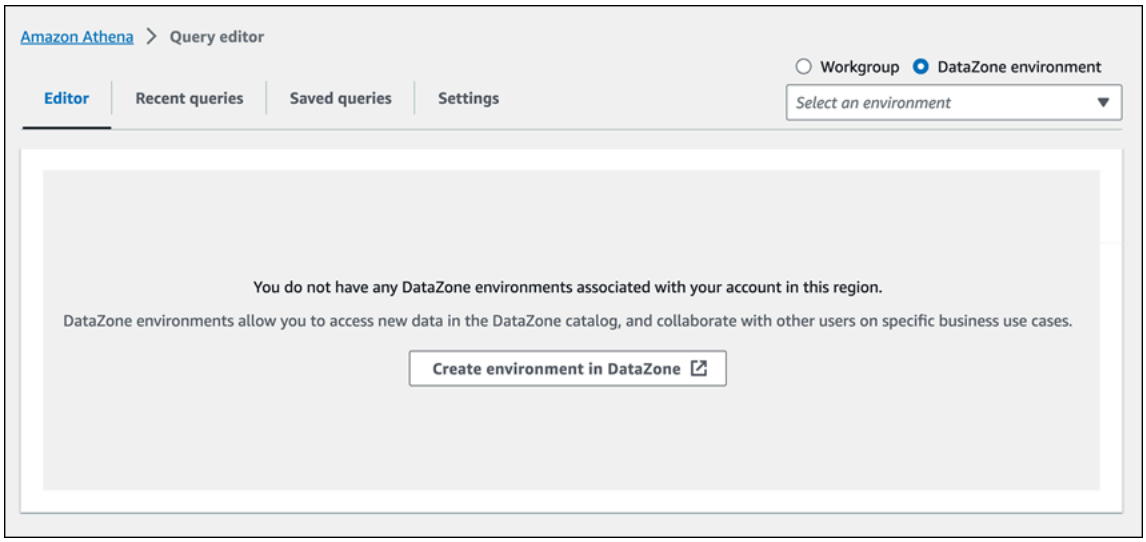
In Athena, puoi utilizzare l'editor di query per accedere e interrogare gli ambienti DataZone. Un ambiente DataZone specifica una combinazione di progetto e dominio DataZone. Quando utilizzi un ambiente DataZone dalla console Athena, assumi il ruolo IAM dell'ambiente DataZone e vedi solo i database e le tabelle che appartengono a quell'ambiente. Le autorizzazioni sono determinate dai ruoli specificati in DataZone.

In Athena, per scegliere un ambiente DataZone puoi utilizzare il selettore dell'ambiente DataZone nella pagina dell'editor di query.

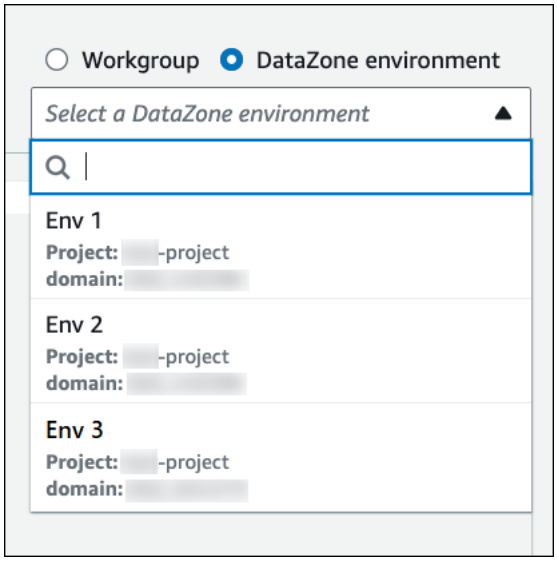
Apertura di un ambiente DataZone in Athena

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Nella parte superiore destra della console Athena, accanto a Gruppo di lavoro, scegli Ambiente DataZone.

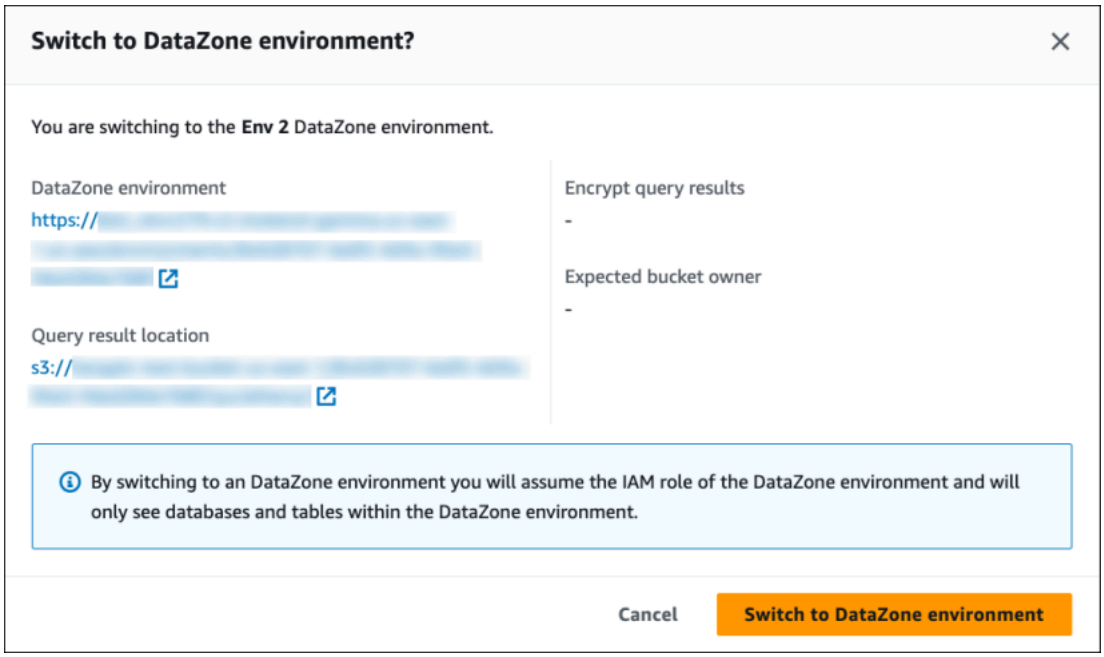
> **Note**
>
> L'opzione Ambiente DataZone è presente solo quando in DataZone sono disponibili uno o più domini.



3. Utilizza il selettore Ambiente DataZone per scegliere un ambiente DataZone.



4. Nella finestra di dialogo Passa all'ambiente DataZone, verifica che l'ambiente sia quello desiderato, quindi scegli Passa all'ambiente DataZone.



Per ulteriori informazioni su come iniziare a usare DataZone e Athena, consulta il tutorial Nozioni di base nella Guida per l'utente di Amazon DataZone.

Connessione ad Amazon Athena con i driver ODBC e JDBC

Per esplorare e visualizzare i dati utilizzando gli strumenti di business intelligence, scarica, installa e configura un driver ODBC (Open Database Connectivity) o JDBC (Java Database Connectivity).

Argomenti

Consulta anche i seguenti argomenti del AWS Knowledge Center e del blog sui AWS Big Data:

- Come posso utilizzare le credenziali del mio ruolo IAM o passare a un altro ruolo IAM quando mi connetto ad Athena utilizzando il driver JDBC?
- Configurazione della fiducia tra ADFS AWS e utilizzo delle credenziali di Active Directory per la connessione ad Amazon Athena con driver ODBC

Connessione ad Amazon Athena con JDBC

Amazon Athena offre due driver JDBC, le versioni 2.x e 3.x. Il driver Athena JDBC 3.x è il driver di nuova generazione che offre prestazioni e compatibilità migliori. Il driver JDBC 3.x supporta la lettura dei risultati delle query direttamente da Amazon S3, il che migliora le prestazioni delle applicazioni che utilizzano risultati di query di grandi dimensioni. Il nuovo driver ha anche un minor numero di dipendenze da terze parti, il che semplifica l'integrazione con strumenti di BI e applicazioni personalizzate. Nella maggior parte dei casi, è possibile utilizzare il nuovo driver senza apportare modifiche minime alla configurazione esistente.

- Per scaricare il driver JDBC 3.x, consulta Driver JDBC 3.x di Athena.
- Per scaricare il driver JDBC 2.x, consulta Driver JDBC 2.x di Athena.

Argomenti

Driver JDBC 3.x di Athena

È possibile utilizzare un il driver JDBC di Athena per connetterti ad Amazon Athena da numerosi strumenti client SQL e dalle e applicazioni di terze parti.

Requisiti di sistema

- Ambiente di runtime Java 8 (o superiore)

- Almeno 20 MB di spazio su disco disponibile

Considerazioni e limitazioni

Di seguito sono riportate alcune considerazioni e limitazioni per il driver JDBC 3.x di Athena.

- Registrazione: il driver 3.x utilizza SLF4J, un livello di astrazione che consente l'utilizzo di uno qualsiasi di diversi sistemi di registrazione durante il runtime.
- Crittografia: quando utilizzi il fetcher Amazon S3 con l'opzione di crittografia `CSE_KMS`, il client di Amazon S3 non può decrittografare i risultati archiviati in un bucket di Amazon S3. Se è necessaria la crittografia `CSE_KMS`, è possibile continuare a utilizzare il fetcher di streaming. È previsto il supporto per la crittografia `CSE_KMS` con il fetcher Amazon S3.

Download del driver JDBC 3.x

Questa sezione contiene informazioni sul download e sulla licenza per il driver JDBC 3.x.

> **Important**
>
> Quando utilizzi il driver JDBC 3.x, assicurati di rispondere ai seguenti requisiti:
>
> - Open port 444 (apri la porta 444): mantieni la porta 444, che Athena utilizza per trasmettere i risultati delle query, aperta al traffico in uscita. Quando usi un PrivateLink endpoint per connetterti ad Athena, assicurati che il gruppo di sicurezza collegato all'endpoint sia aperto PrivateLink al traffico in entrata sulla porta 444.
> - athena: GetQueryResultsStream policy — Aggiungi l'azione `athena:GetQueryResultsStream` policy ai principali IAM che utilizzano il driver JDBC. Questa operazione di policy non è esposta direttamente con l'API. Viene utilizzata solo con i driver ODBC e JDBC come parte del supporto per i risultati di streaming. Per un esempio di policy, consulta AWS politica gestita: AWSQuicksightAthenaAccess.

Per scaricare il driver JDBC 3.x di Amazon Athena, visita i seguenti link.

Jar uber del driver JDBC

Il seguente download impachetta il driver e tutte le sue dipendenze nello stesso file `.jar`. Questo download viene in genere utilizzato per client SQL di terze parti.

https://downloads.athena.us-east-1.amazonaws.com/drivers/JDBC/3.2.0/athena-jdbc-3.2.0-with-dependencies.jar

Jar lean del driver JDBC

Il seguente download è un file `.zip` che contiene il file `.jar` lean per il driver e separa i file `.jar` per le dipendenze del driver. Questo download viene in genere utilizzato per applicazioni personalizzate che potrebbero avere dipendenze in conflitto con quelle utilizzate dal driver. Questo download è utile se si desidera scegliere quali dipendenze dei driver includere nel jar lean e quali escludere se l'applicazione personalizzata ne contiene già una o più.

https://downloads.athena.us-east-1.amazonaws.com/drivers/JDBC/3.2.0/athena-jdbc-3.2.0-lean-jar-and-separate-dependencies-jars.zip

Licenza

Il seguente link contiene il contratto di licenza per il driver JDBC 3.x.

Licenza

Argomenti

Guida introduttiva al driver JDBC 3.x

Usa le informazioni in questa sezione per iniziare a usare il driver JDBC 3.x di Amazon Athena.

Argomenti

Istruzioni di installazione

È possibile utilizzare il driver JDBC 3.x in un'applicazione personalizzata o da un client SQL di terze parti.

In un'applicazione personalizzata

Scarica il file `.zip` che contiene il file jar del driver e le relative dipendenze. Ogni dipendenza ha il proprio file `.jar`. Aggiungi il driver jar come dipendenza nella tua applicazione personalizzata. Aggiungi selettivamente le dipendenze del file jar del driver a seconda che tu abbia già aggiunto tali dipendenze all'applicazione da un'altra origine.

In un client SQL di terze parti

Scarica il file jar uber del driver e aggiungilo al client SQL di terze parti seguendo le istruzioni per quel client.

Esecuzione del driver

Per eseguire il driver, puoi utilizzare un'applicazione personalizzata o un client SQL di terze parti.

In un'applicazione personalizzata

Utilizza l'interfaccia JDBC per interagire con il driver JDBC da un programma. Il codice riportato di seguito mostra un'applicazione Java personalizzata di esempio.

```
public static void main(String args[]) throws SQLException {
    Properties connectionParameters = new Properties();
    connectionParameters.setProperty("Workgroup", "primary");
    connectionParameters.setProperty("Region", "us-east-2");
    connectionParameters.setProperty("Catalog", "AwsDataCatalog");
    connectionParameters.setProperty("Database","sampledatabase");
    connectionParameters.setProperty("OutputLocation","s3://DOC-EXAMPLE-BUCKET");
    connectionParameters.setProperty("CredentialsProvider","DefaultChain");
    String url = "jdbc:athena://";
    AthenaDriver driver = new AthenaDriver();
    Connection connection = driver.connect(url, connectionParameters);
    Statement statement = connection.createStatement();
    String query = "SELECT * from sample_table LIMIT 10";
    ResultSet resultSet = statement.executeQuery(query);
    printResults(resultSet); // A custom-defined method for iterating over a
                             // result set and printing its contents
}
```

In un client SQL di terze parti

Consulta la documentazione del client SQL che stai utilizzando. In genere, si utilizza l'interfaccia utente grafica del client SQL per immettere e inviare la query e i risultati della query vengono visualizzati nella stessa interfaccia.

Configurazione del driver

Puoi utilizzare i parametri di connessione per configurare il driver JDBC di Amazon Athena. Per i parametri di connessione supportati, consulta Parametri di connessione JDBC 3.x di Amazon Athena.

In un'applicazione personalizzata

Per impostare i parametri di connessione per il driver JDBC in un'applicazione personalizzata, completa una delle seguenti operazioni:

- Aggiungi i nomi dei parametri e i relativi valori a un oggetto `Properties`. Quando chiami `Connection#connect`, passa quell'oggetto insieme all'URL. Per un esempio, consulta l'applicazione Java di esempio in Esecuzione del driver.
- Nella stringa di connessione (l'URL), utilizza il seguente formato per aggiungere i nomi dei parametri e i relativi valori direttamente dopo il prefisso del protocollo.

  ```
  <parameterName>=<parameterValue>;
  ```

  Utilizza un punto e virgola alla fine di ogni coppia nome/valore del parametro e non lasciare spazi vuoti dopo il punto e virgola, come nell'esempio seguente.

  ```
  String url = "jdbc:athena://WorkGroup=primary;Region=us-east-1;...;";AthenaDriver
   driver = new AthenaDriver();Connection connection = driver.connect(url, null);
  ```

  > **Note**
  >
  > Se viene specificato un parametro sia nella stringa di connessione che nell'oggetto `Properties`, il valore nella stringa di connessione ha la precedenza. Non è consigliabile specificare lo stesso parametro in entrambe le posizioni.

- Aggiungi i valori dei parametri come argomenti ai metodi di `AthenaDataSource`, come nell'esempio seguente.

```
AthenaDataSource dataSource = new AthenaDataSource();
    dataSource.setWorkGroup("primary");
    dataSource.setRegion("us-east-2");
    ...
    Connection connection = dataSource.getConnection();
    ...
```

In un client SQL di terze parti

Segui le istruzioni del client SQL che stai utilizzando. In genere, il client fornisce un'interfaccia utente grafica per inserire i nomi dei parametri e i relativi valori.

Aggiornamento dal driver JDBC v2 di Athena

La maggior parte dei parametri di connessione JDBC versione 3 sono retrocompatibili con il driver JDBC versione 2 (Simba). Ciò significa che una stringa di connessione della versione 2 può essere riutilizzata con la versione 3 del driver. Tuttavia, alcuni parametri di connessione sono cambiati. Queste modifiche sono descritte qui. Quando esegui l'aggiornamento al driver JDBC versione 3, aggiorna la configurazione esistente se necessario.

Classe del driver

Alcuni strumenti di BI richiedono di fornire la classe del driver dal file `.jar` del driver JDBC. La maggior parte degli strumenti trova questa classe automaticamente. Il nome completo della classe nel driver della versione 3 è `com.amazon.athena.jdbc.AthenaDriver`. Nel driver della versione 2, la classe era `com.simba.athena.jdbc.Driver`.

Stringa di connessione

Il driver della versione 3 utilizzato `jdbc:athena://` per il protocollo all'inizio dell'URL della stringa di connessione JDBC. Il driver della versione 3 supporta anche il protocollo della versione 2`jdbc:awsathena://`, ma l'utilizzo del protocollo della versione 2 è obsoleto. Per evitare comportamenti indefiniti, la versione 3 non accetta stringhe di connessione che iniziano con `jdbc:awsathena://` se la versione 2 (o qualsiasi altro driver che accetta stringhe di connessione che iniziano con`jdbc:awsathena://`) è stata registrata nella classe. DriverManager

Provider di credenziali

Il driver della versione 2 utilizza nomi completi per identificare diversi provider di credenziali, ad esempio `com.simba.athena.amazonaws.auth.DefaultAWSCredentialsProviderChain`.

Il driver della versione 3 utilizza nomi più brevi, ad esempio, `DefaultChain`. I nuovi nomi sono descritti nelle sezioni corrispondenti per ogni provider di credenziali.

I provider di credenziali personalizzate scritti per il driver della versione 2 devono essere modificati affinché il driver della versione 3 possa implementare l'AwsCredentialsProviderinterfaccia della AWSCredentialsProvidernuova versione AWS SDK for Java anziché quella precedente. AWS SDK for Java

Non `PropertiesFileCredentialsProvider` è supportato nel driver JDBC 3.x. Il provider è stato utilizzato nel driver JDBC 2.x ma appartiene alla versione precedente dell'SDK for AWS Java che sta per terminare il supporto. Per ottenere la stessa funzionalità nel driver JDBC 3.x, utilizzate invece il provider. Credenziali del profilo di configurazione AWS

Livello di log

La tabella seguente mostra le differenze nei parametri `LogLevel` nei driver JDBC versione 2 e versione 3.

Versione driver JDBC	Nome del parametro	Tipo parametro	Valore predefinito	Valori possibili	Esempio stringa connessione
v2	`LogLevel`	Facoltativo	0	0-6	`LogLevel=6;`
v3	`LogLevel`	Facoltativo	TRACE	OFF, ERROR, WARN, INFO, DEBUG, TRACE	`LogLevel=INFO;`

Recupero dell'ID query

Nel driver della versione 2, si spacchetta un'istanza `Statement` in `com.interfaces.core.IStatementQueryInfoProvider`, un'interfaccia con due metodi, `#getPReparedQueryId` e `#getQueryId`. È possibile utilizzare questi metodi per ottenere l'ID di esecuzione di una query che è stata eseguita.

Nel driver della versione 3, si spacchettano le istanze `Statement`, `PreparedStatement` e `ResultSet` sull'interfaccia `com.amazon.athena.jdbc.AthenaResultSet`. L'interfaccia ha un metodo: `#getQueryExecutionId`.

Parametri di connessione JDBC 3.x di Amazon Athena

I parametri di connessione supportati sono qui suddivisi in tre sezioni: Parametri di connessione di base, Parametri di connessione avanzati e Parametri di connessione per l'autenticazione. Le sezioni Parametri di connessione avanzati e Parametri di connessione di autenticazione contengono sottosezioni che raggruppano i parametri correlati.

Argomenti

Parametri di connessione di base

Nelle sezioni seguenti sono descritti i parametri di connessione di base per il driver JDBC 3.x.

Regione

Regione AWS Dove verranno eseguite le interrogazioni. Per un elenco delle regioni, consulta Endpoint e quote di Amazon Athena.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Regione	AwsRegion (obsoleto)	Obbligatorio (ma se non fornito, verrà cercato utilizzando il) DefaultAwsRegionProviderChain	nessuno

Catalogo

Il catalogo che contiene i database e le tabelle a cui si accederà con il driver. Per informazioni sui cataloghi, vedere DataCatalog.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Catalogo	nessuno	Facoltativo	AwsDataCatalog

Database

Il database in cui verranno eseguite le query. Le tabelle che non sono esplicitamente qualificate con un nome di database vengono risolte in questo database. Per informazioni sui database, consulta Database.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Database	Schema	Facoltativo	default

Gruppo di lavoro

Il gruppo di lavoro in cui verranno eseguite le query. Per informazioni sui gruppi di lavoro, vedere. WorkGroup

Nome del parametro	Alias	Tipo parametro	Valore predefinito
WorkGroup	nessuno	Facoltativo	principale

Percorso di output

Il percorso in Amazon S3 in cui verranno archiviati i risultati della query. Per informazioni sulla posizione di output, vedere ResultConfiguration.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
OutputLocation	S3 OutputLocation (obsoleto)	Obbligatorio (a meno che il gruppo di lavoro non specifichi una posizione di output)	nessuno

Parametri di connessione avanzati

Nelle sezioni seguenti sono descritti i parametri di connessione avanzati per il driver JDBC 3.x.

Argomenti

Parametri di crittografia dei risultati

Notare i seguenti punti:

- La AWS KMS chiave deve essere specificata quando `EncryptionOption` è `SSE_KMS` o`CSE_KMS`.
- La AWS KMS chiave non può essere specificata quando non `EncryptionOption` è specificata o quando lo `EncryptionOption` è`SSE_S3`.

Opzione di crittografia

Il tipo di crittografia da utilizzare per i risultati delle query quando vengono archiviati in Amazon S3. Per informazioni sulla crittografia dei risultati delle query, EncryptionConfigurationconsulta Amazon Athena API Reference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valori possibili
EncryptionOption	S3 OutputEncOption (obsoleto)	Facoltativo	nessuno	SSE_S3, SSE_KMS, CSE_KMS

Chiave KMS

L'ARN o l'ID della chiave KMS, se `SSE_KMS` o `CSE_KMS` viene scelto come opzione di crittografia. Per ulteriori informazioni, consulta EncryptionConfigurationAmazon Athena API Reference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
KmsKey	S3 OutputEncKMSKey (obsoleto)	Facoltativo	nessuno

Parametri di recupero dei risultati

Sistema di recupero (fetcher) dei risultati

Il fetcher che verrà utilizzato per scaricare i risultati delle query.

Il fetcher dei risultati predefinito, `S3`, scarica i risultati delle query direttamente da Amazon S3 senza utilizzare le API Athena. Si tratta dell'opzione più rapida nella maggior parte dei casi. Questa opzione non è disponibile se i risultati della query sono crittografati con `CSE_KMS` o se la policy che consente all'utente di accedere ai risultati della query consente solo le chiamate da Athena tramite `s3:CalledVia`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valori possibili
ResultFetcher	nessuno	Facoltativo	S3	GetQueryResultsS3, GetQueryResultsStream

> **Note**
>
> Nel driver JDBC 2.x, l'`UseResultsetStreaming = 1`impostazione configura il driver per l'utilizzo dell'API di streaming del set di risultati. Nel driver JDBC 3.x, l'impostazione equivalente è. `ResultFetcher=GetQueryResultsStream`

Dimensioni di recupero

Il valore di questo parametro viene utilizzato come valore minimo per i buffer interni e come dimensione della pagina di destinazione durante il recupero dei risultati. Il valore 0 (zero) indica che il driver deve utilizzare i propri valori predefiniti come descritto di seguito. Il valore massimo è 1.000.000.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
FetchSize	RowsToFetchPerBlock (obsoleto)	Facoltativo	0

- Il fetcher `GetQueryResults` utilizzerà sempre una dimensione di pagina pari a 1.000, che è il valore massimo supportato dalla chiamata API. Quando la dimensione di recupero è superiore a 1.000, vengono effettuate più chiamate API successive per riempire il buffer al di sopra del minimo.
- Il fetcher `GetQueryResultsStream` utilizzerà la dimensione di recupero configurata come dimensione della pagina, o 10.000 per impostazione predefinita.
- Il fetcher `S3` utilizzerà la dimensione di recupero configurata come dimensione della pagina, o 10.000 per impostazione predefinita.

Parametri di riutilizzo dei risultati della query

Come abilitare il riutilizzo dei risultati

Specifica se i risultati della query precedente possono essere riutilizzati quando la query viene eseguita. Per informazioni sul riutilizzo dei risultati delle query, vedere. ResultReuseByAgeConfiguration

Nome del parametro	Alias	Tipo parametro	Valore predefinito
EnableResultReuseByAge	nessuno	Facoltativo	FALSE

Età massima per il riutilizzo di risultati

L'età massima, in minuti, dei risultati di una query precedente che Athena dovrebbe considerare per il riutilizzo. Per informazioni sull'età massima per il riutilizzo dei risultati, vedere. ResultReuseByAgeConfiguration

Nome del parametro	Alias	Tipo parametro	Valore predefinito
MaxResultReuseAgeInMinutes	nessuno	Facoltativo	60

Parametri di polling per l'esecuzione delle query

Intervallo minimo di polling per l'esecuzione delle query

Il tempo minimo, in millisecondi, da attendere prima di eseguire il polling di Athena per verificare lo stato di esecuzione della query.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
MinQueryExecutionPollingIntervalMillis	MinQueryExecutionPollingInterval (obsoleto)	Facoltativo	100

Intervallo massimo di polling per l'esecuzione delle query

Il tempo massimo, in millisecondi, da attendere prima di eseguire il polling di Athena per verificare lo stato di esecuzione della query.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
MaxQueryExecutionPollingIntervalMillis	MaxQueryExecutionPollingInterval (obsoleto)	Facoltativo	5000

Moltiplicatore dell'intervallo di polling per l'esecuzione delle query

Il fattore di incremento del periodo di polling. Per impostazione predefinita, il polling inizierà con il valore per `MinQueryExecutionPollingIntervalMillis` e raddoppierà con ogni polling fino a raggiungere il valore per `MaxQueryExecutionPollingIntervalMillis`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
QueryExecutionPollingIntervalMultiplier	nessuno	Facoltativo	2

Parametri di sovrascrittura dell'endpoint

Sovrascrizione degli endpoint Athena

L'endpoint che il driver utilizzerà per effettuare chiamate API verso Athena.

Notare i seguenti punti:

- Se i protocolli `https://` o `http://` non sono specificati nell'URL fornito, il driver inserisce il prefisso `https://`.
- Se questo parametro non è specificato, il driver utilizza un endpoint predefinito.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
AthenaEndpoint	EndpointOverride (obsoleto)	Facoltativo	nessuno

Sostituzione degli endpoint di streaming Athena

L'endpoint che il driver utilizzerà per scaricare i risultati delle query quando utilizza il servizio di streaming Athena. Il servizio di streaming Athena è disponibile tramite la porta 444.

Notare i seguenti punti:

- Se i protocolli `https://` o `http://` non sono specificati nell'URL fornito, il driver inserisce il prefisso `https://`.
- Se una porta non è specificata nell'URL fornito, il driver inserisce la porta 444 del servizio di streaming.
- Se il `AthenaStreamingEndpoint` parametro non è specificato, il driver utilizza la sovrascrittura di `AthenaEndpoint`. Se non viene specificata né la sovrascrittura `AthenaStreamingEndpoint` né `AthenaEndpoint`, il driver utilizzerà un endpoint di streaming predefinito.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
AthenaStreamingEndpoint	StreamingEndpointOverride (obsoleto)	Facoltativo	nessuno

LakeFormation sovrascrittura dell'endpoint

L'endpoint che il driver utilizzerà per il servizio Lake Formation quando utilizza l'API AWS Lake Formation AssumeDecoratedRoleWithSAML per recuperare le credenziali temporanee. Se questo parametro non viene specificato, il driver utilizza un endpoint predefinito.

Notare i seguenti punti:

- Se i protocolli `https://` o `http://` non sono specificati nell'URL fornito, il driver inserisce il prefisso `https://`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
LakeFormationEndpoint	LfEndpointOverride (obsoleto)	Facoltativo	nessuno

Sovrascrittura endpoint S3

L'endpoint che il driver utilizzerà per scaricare i risultati delle query quando utilizza il fetcher Amazon S3. Se questo parametro non viene specificato, il driver utilizzerà un endpoint Amazon S3 predefinito.

Notare i seguenti punti:

- Se i protocolli `https://` o `http://` non sono specificati nell'URL fornito, il driver inserisce il prefisso `https://`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
S3Endpoint	Nessuno	Facoltativo	nessuno

Sovrascrittura di endpoint STS

L'endpoint che il driver utilizzerà per il AWS STS servizio quando utilizza l'API AWS STS AssumeRoleWithSAML per recuperare credenziali temporanee. Se questo parametro non è specificato, il driver utilizza un endpoint predefinito. AWS STS

Notare i seguenti punti:

- Se i protocolli `https://` o `http://` non sono specificati nell'URL fornito, il driver inserisce il prefisso `https://`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
StsEndpoint	StsEndpointOverrid e(obsoleto)	Facoltativo	nessuno

Parametri di configurazione proxy

Host proxy

L'URL dell'host proxy. Utilizza questo parametro se desideri che le richieste Athena passino attraverso un proxy.

> **Note**
>
> Assicurati di includere il protocollo `https://` o `http://` all'inizio dell'URL per `ProxyHost`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
ProxyHost	nessuno	Facoltativo	nessuno

Porta proxy

La porta da utilizzare sull'host proxy. Utilizza questo parametro se desideri che le richieste Athena passino attraverso un proxy.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
ProxyPort	nessuno	Facoltativo	nessuno

Nome utente proxy

Il nome utente per eseguire l'autenticazione con il server proxy. Utilizza questo parametro se desideri che le richieste Athena passino attraverso un proxy.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
ProxyUsername	ProxyUID (obsoleto)	Facoltativo	nessuno

Password proxy

La password per eseguire l'autenticazione con il server proxy. Utilizza questo parametro se desideri che le richieste Athena passino attraverso un proxy.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
ProxyPassword	ProxyPWD (obsoleto)	Facoltativo	nessuno

Host esenti da proxy

Un set di nomi host a cui il driver si connette senza utilizzare un proxy quando il proxy è abilitato (ovvero quando sono impostati i parametri di connessione `ProxyHost` e `ProxyPort`). Gli host devono essere separati con una barra verticale (`|`), ad esempio, `host1.com|host2.com`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
ProxyExemptHosts	NonProxyHosts	Facoltativo	nessuno

Proxy abilitato per i gestori delle identità

Speciifica se deve essere utilizzato un proxy quando il driver si connette a un gestore delle identità.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
ProxyEnabledForIdP	UseProxyForIdP	Facoltativo	FALSE

Parametri di registrazione

In questa sezione sono descritti i parametri relativi alla registrazione.

Livello di log

Specifica il livello di registrazione del driver. Non viene registrato nulla a meno che non sia impostato anche il parametro `LogPath`.

> **Note**
>
> Si consiglia di impostare solo il parametro `LogPath`, a meno che non si abbiano requisiti speciali. L'impostazione del solo parametro `LogPath` abilita la registrazione e utilizza il livello di log `TRACE` predefinito. Il livello di log `TRACE` fornisce la registrazione più dettagliata.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valori possibili
LogLevel	nessuno	Facoltativo	TRACE	OFF, ERROR, WARN, INFO, DEBUG, TRACE

Log Path (Percorso log)

Il percorso di una directory sul computer che esegue il driver in cui verranno archiviati i log dei driver. Verrà creato un file di log con un nome univoco all'interno della directory specificata. Se impostato, abilita la registrazione del driver.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
LogPath	nessuno	Facoltativo	nessuno

Nome applicazione

Il nome dell'applicazione che utilizza il driver. Se viene specificato un valore per questo parametro, il valore sarà incluso nella stringa dell'agente utente o delle chiamate API che il driver effettua ad Athena.

> **Note**
>
> È inoltre possibile impostare il nome dell'applicazione richiamando `setApplicationName` sull'oggetto `DataSource`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
ApplicationName	nessuno	Facoltativo	nessuno

Test di connessione

Se impostato su TRUE, il driver esegue un test di connessione ogni volta che viene creata una connessione JDBC, anche se non viene eseguita alcuna query sulla connessione.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
ConnectionTest	nessuno	Facoltativo	TRUE

> **Note**
>
> Un test di connessione invia una query `SELECT 1` ad Athena per verificare che la connessione sia stata configurata correttamente. Ciò significa che due file verranno archiviati in Amazon S3 (il set di risultati e i metadati) e potranno essere applicati costi aggiuntivi in base alla policy dei prezzi di Amazon Athena.

Numero di tentativi

Il numero massimo di volte in cui il driver deve inviare nuovamente una richiesta recuperabile ad Athena.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
NumRetries	MaxErrorRetry (obsoleto)	Facoltativo	nessuno

Parametri di connessione per l'autenticazione

Il driver JDBC 3.x di Athena supporta diversi metodi di autenticazione. I parametri di connessione richiesti dipendono dal metodo di autenticazione utilizzato.

Argomenti

- Credenziali IAM
- Credenziali predefinite
- Credenziali del profilo di configurazione AWS

- Credenziali del profilo dell'istanza
- Credenziali personalizzate
- Credenziali JWT
- Credenzial Azure AD
- Credenziali Okta
- Credenziali Ping
- Credenziali AD FS
- Credenziali Browser Azure AD
- Credenziali Browser SAML

Credenziali IAM

Puoi utilizzare le tue credenziali IAM con il driver JDBC per connetterti ad Amazon Athena utilizzando i parametri della stringa di connessione riportati di seguito.

Utente

L'ID della tua chiave di AWS accesso. Per ulteriori informazioni sulle chiavi di accesso, consulta Credenziali di sicurezza di AWS nella Guida per l'utente di IAM.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Utente	AccessKeyId	Richiesto	nessuno

Password

L'ID della tua chiave AWS segreta. Per ulteriori informazioni sulle chiavi di accesso, consulta Credenziali di sicurezza di AWS nella Guida per l'utente di IAM.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Password	SecretAccessKey	Facoltativo	nessuno

Token di sessione

Se si utilizzano AWS credenziali temporanee, è necessario specificare un token di sessione. Per informazioni sulle credenziali temporanee, consulta la sezione relativa alle Credenziali di sicurezza temporanee in IAM nella Guida per l'utente IAM.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
SessionToken	nessuno	Facoltativo	nessuno

Credenziali predefinite

Puoi utilizzare le credenziali predefinite che configuri sul tuo sistema client per connetterti ad Amazon Athena impostando i parametri di connessione riportati di seguito. Per informazioni sull'utilizzo delle credenziali predefinite, consulta Utilizzo della catena di provider delle credenziali predefinita nella Guida per gli sviluppatori di AWS SDK for Java .

Provider di credenziali

Il provider di credenziali che sarà utilizzato per autenticare le richieste a AWS. Imposta il valore di questo parametro su `DefaultChain`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valore da utilizzare
CredentialsProvider	AWSCredentialsProviderClass (obsoleto)	Richiesto	nessuno	`DefaultChain`

Credenziali del profilo di configurazione AWS

È possibile utilizzare le credenziali memorizzate in un profilo di AWS configurazione impostando i seguenti parametri di connessione. AWS i profili di configurazione sono in genere archiviati nei file della `~/.aws` directory). Per informazioni sui profili di configurazione AWS , consulta Utilizzo dei profili nella Guida per gli sviluppatori di AWS SDK for Java .

Provider di credenziali

Il provider di credenziali che sarà utilizzato per autenticare le richieste a AWS. Imposta il valore di questo parametro su `ProfileCredentials`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valore da utilizzare
CredentialsProvider	AWSCredentialsProviderClass(obsoleto)	Richiesto	nessuno	`ProfileCredentials`

Nome del profilo

Il nome del profilo di AWS configurazione le cui credenziali devono essere utilizzate per autenticare la richiesta ad Athena.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
ProfileName	nessuno	Richiesto	nessuno

> **Note**
>
> Il nome del profilo può essere specificato anche come valore del parametro `CredentialsProviderArguments`, sebbene questo uso sia obsoleto.

Credenziali del profilo dell'istanza

Questo tipo di autenticazione viene utilizzato sulle istanze Amazon EC2. Un profilo dell'istanza è un profilo collegato a un'istanza Amazon EC2. L'utilizzo di un provider di credenziali di profilo di istanza delega la gestione delle AWS credenziali all'Amazon EC2 Instance Metadata Service. Ciò elimina la necessità per gli sviluppatori di archiviare le credenziali in modo permanente sull'istanza Amazon EC2 o di preoccuparsi della rotazione o della gestione delle credenziali temporanee.

Provider di credenziali

Il provider di credenziali che sarà utilizzato per autenticare le richieste a AWS. Imposta il valore di questo parametro su `InstanceProfile`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valore da utilizzare
CredentialsProvider	AWSCredentialsProviderClass (obsoleto)	Richiesto	nessuno	`InstanceProfile`

Credenziali personalizzate

È possibile utilizzare questo tipo di autenticazione per fornire le proprie credenziali utilizzando una classe Java che implementa l'interfaccia. AwsCredentialsProvider

Provider di credenziali

Il provider di credenziali che sarà utilizzato per autenticare le richieste a AWS. Imposta il valore di questo parametro sul nome completo della classe personalizzata che implementa l'interfaccia. AwsCredentialsProvider Durante il runtime, tale classe deve trovarsi nel percorso della classe Java dell'applicazione che utilizza il driver JDBC.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valore da utilizzare
CredentialsProvider	AWSCredentialsProviderClass (obsoleto)	Richiesto	nessuno	Il nome completo della classe dell'implementazione personalizzata di `AwsCredentialsProvider`

Argomenti del provider di credenziali

Un elenco di argomenti di stringa separati da virgole per il costruttore del provider di credenziali personalizzate.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
CredentialsProvideArguments	AwsCredentialsProviderArguments (obsoleto)	Facoltativo	nessuno

Credenziali JWT

Con questo tipo di autenticazione, puoi utilizzare un token Web JSON (JWT) ottenuto da un gestore delle identità esterno come parametro di connessione per l'autenticazione con Athena. Il provider di credenziali esterno deve già essere federato con. AWS

Provider di credenziali

Il provider di credenziali che sarà utilizzato per autenticare le richieste a AWS. Imposta il valore di questo parametro su `JWT`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valore da utilizzare
CredentialsProvider	AWSCredentialsProviderClass (obsoleto)	Richiesto	nessuno	`JWT`

Token di identità Web JWT

Il token JWT ottenuto da un gestore delle identità federato esterno. Questo token verrà utilizzato per l'autenticazione con Athena.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
JwtWebIdentityToken	web_identity_token (obsoleto)	Richiesto	nessuno

ARN del ruolo JWT

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Per informazioni sull'assunzione di ruoli, AssumeRoleconsulta l'API Reference.AWS Security Token Service

Nome del parametro	Alias	Tipo parametro	Valore predefinito
JwtRoleArn	role_arn (obsoleto)	Richiesto	nessuno

Nome della sessione del ruolo JWT

Il nome della sessione quando si utilizzano le credenziali JWT per l'autenticazione. Il nome può essere un qualsiasi nome a tua scelta.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
JwtRoleSessionName	role_session_name (obsoleto)	Richiesto	nessuno

Credenzial Azure AD

Un meccanismo di autenticazione basato su SAML che consente l'autenticazione con Athena usando il gestore delle identità Azure AD. Questo metodo presuppone che sia già stata configurata una federazione tra Athena e Azure AD.

> **Note**
>
> Alcuni nomi di parametri in questa sezione hanno alias. Gli alias sono equivalenti funzionali dei nomi dei parametri e sono stati forniti per garantire la compatibilità con le versioni precedenti con il driver JDBC 2.x. Poiché i nomi dei parametri sono stati migliorati per seguire una convenzione di denominazione più chiara e coerente, si consiglia di utilizzarli al posto degli alias, ormai obsoleti.

Provider di credenziali

Il provider di credenziali che sarà utilizzato per autenticare le richieste a AWS. Imposta il valore di questo parametro su `AzureAD`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valore da utilizzare
CredentialsProvider	AWSCredentialsProviderClass (obsoleto)	Richiesto	nessuno	`AzureAD`

Utente

L'indirizzo e-mail dell'utente di Azure AD da usare per l'autenticazione con Azure AD.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Utente	UID (obsoleto)	Richiesto	nessuno

Password

La password per l'utente Azure AD.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Password	PWD (obsoleto)	Richiesto	nessuno

ID tenant Azure AD

L'ID tenant dell'applicazione Azure AD.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
AzureAdTenantId	tenant_id (obsoleto)	Richiesto	nessuno

ID client di Azure AD

L'ID client dell'applicazione Azure AD.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
AzureAdClientId	client_id (obsoleto)	Richiesto	nessuno

Segreto client Azure

Il segreto del client dell'applicazione Azure AD.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
AzureAdClientSecret	client_secret (obsoleto)	Richiesto	nessuno

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Per informazioni sui ruoli ARN, consulta l'AWS Security Token Service API AssumeRoleReference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
PreferredRole	preferred_role (obsoleto)	Facoltativo	nessuno

Durata della sessione del ruolo

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni, consulta AssumeRolel'AWS Security Token Service API Reference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
RoleSessionDuration	Duration (obsoleto)	Facoltativo	3600

Lake Formation abilitato

Speciifica se utilizzare l'azione API AssumeDecoratedRoleWithSAML Lake Formation per recuperare le credenziali IAM temporanee anziché l'AssumeRoleWithazione API SAML. AWS STS

Nome del parametro	Alias	Tipo parametro	Valore predefinito
LakeFormationEnabled	nessuno	Facoltativo	FALSE

Credenziali Okta

Un meccanismo di autenticazione basato su SAML che consente l'autenticazione con Athena usando il gestore delle identità Okta. Questo metodo presuppone che sia già stata configurata una federazione tra Athena e Okta.

Provider di credenziali

Il provider di credenziali che sarà utilizzato per autenticare le richieste a AWS. Imposta il valore di questo parametro su `Okta`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valore da utilizzare
CredentialsProvider	AWSCredentialsProviderClass (obsoleto)	Richiesto	nessuno	`Okta`

Utente

L'indirizzo e-mail dell'utente Okta da utilizzare per l'autenticazione con Okta.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Utente	UID (obsoleto)	Richiesto	nessuno

Password

La password dell'utente Okta.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Password	PWD (obsoleto)	Richiesto	nessuno

Nome host Okta

L'URL della tua organizzazione Okta. Puoi estrarre il parametro `idp_host` dall'URL del Link di incorporamento nella tua applicazione Okta. Per le fasi, consulta Recupero delle informazioni di configurazione ODBC da Okta. Il primo segmento successivo a `https://`, fino a `okta.com` incluso, è il tuo host IdP (ad esempio, `trial-1234567.okta.com` per un URL che inizia con `https://trial-1234567.okta.com`).

Nome del parametro	Alias	Tipo parametro	Valore predefinito
OktaHostName	IdP_Host (obsoleto)	Richiesto	nessuno

ID applicazione Okta

L'identificatore in due parti dell'applicazione. Puoi estrarre l'ID applicazione dall'URL Link di incorporamento nella tua applicazione Okta. Per le fasi, consulta Recupero delle informazioni di configurazione ODBC da Okta. L'ID dell'applicazione è costituito dagli ultimi due segmenti dell'URL, inclusa la barra nel mezzo. I segmenti sono due stringhe di 20 caratteri con un mix di numeri e lettere maiuscole e minuscole (ad esempio `Abc1de2fghi3J45kL678/abc1defghij2klmNo3p4`).

Nome del parametro	Alias	Tipo parametro	Valore predefinito
OktaAppId	App_ID (obsoleto)	Richiesto	nessuno

Nome applicazione Okta

Il nome dell'applicazione Okta.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
OktaAppName	App_Name (obsoleto)	Richiesto	nessuno

Tipo MFA Okta

Se hai configurato Okta per richiedere l'autenticazione a più fattori (MFA), devi specificare il tipo di MFA Okta e i parametri aggiuntivi in base al secondo fattore che desideri utilizzare.

Il tipo di MFA Okta è il secondo tipo di fattore di autenticazione (dopo la password) da utilizzare per l'autenticazione con Okta. I secondi fattori supportati includono le notifiche push inviate tramite l'app Okta Verify e le password monouso temporanee (TOTP) generate da Okta Verify, Google Authenticator o inviate tramite SMS. Le policy di sicurezza della singola organizzazione determinano se richiedere o meno l'MFA per l'accesso degli utenti.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valori possibili
OktaMfaType	okta_mfa_type (obsoleto)	Obbligatorio, se Okta è configurato per richiedere l'MFA	nessuno	`oktaverifywithpush`, `oktaverifywithtotp`, `googleauthenticator`, `smsauthentication`

Numero di telefono Okta

Il numero di telefono a cui Okta invierà una password monouso temporanea tramite SMS quando viene scelto il tipo di MFA `smsauthentication`. Il numero di telefono deve essere un numero di telefono statunitense o canadese.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
OktaPhoneNumber	okta_phone_number (obsoleto)	Obbligatorio, se è OktaMfaType `smsauthentication`	nessuno

Tempo di attesa MFA Okta

Il tempo, in secondi, da attendere perché l'utente confermi una notifica push di Okta prima che il driver generi un'eccezione di timeout.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
OktaMfaWaitTime	okta_mfa_wait_time (obsoleto)	Facoltativo	60

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Per informazioni sui ruoli ARN, consulta l'AWS Security Token Service API AssumeRoleReference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
PreferredRole	preferred_role (obsoleto)	Facoltativo	nessuno

Durata della sessione del ruolo

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni, consulta AssumeRolel'AWS Security Token Service API Reference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
RoleSessionDuration	Duration (obsoleto)	Facoltativo	3600

Lake Formation abilitato

Speciifica se utilizzare l'azione API AssumeDecoratedRoleWithSAML Lake Formation per recuperare le credenziali IAM temporanee anziché l'AssumeRoleWithazione API SAML. AWS STS

Nome del parametro	Alias	Tipo parametro	Valore predefinito
LakeFormationEnabled	nessuno	Facoltativo	FALSE

Credenziali Ping

Un meccanismo di autenticazione basato su SAML che consente l'autenticazione con Athena tramite il gestore delle identità Ping Federate. Questo metodo presuppone che sia già stata configurata una federazione tra Athena e Ping Federate.

Provider di credenziali

Il provider di credenziali che sarà utilizzato per autenticare le richieste a AWS. Imposta il valore di questo parametro su `Ping`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valore da utilizzare
CredentialsProvider	AWSCredentialsProviderClass (obsoleto)	Richiesto	nessuno	`Ping`

Utente

L'indirizzo e-mail dell'utente Ping Federate da utilizzare per l'autenticazione con Ping Federate.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Utente	UID (obsoleto)	Richiesto	nessuno

Password

La password per l'utente Ping Federate.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Password	PWD (obsoleto)	Richiesto	nessuno

PingHostName

L'indirizzo del tuo server Ping. Per trovare il tuo indirizzo, visita il seguente URL e visualizza il campo Endpoint applicazione SSO.

```
https://your-pf-host-#:9999/pingfederate/your-pf-app#/spConnections
```

Nome del parametro	Alias	Tipo parametro	Valore predefinito
PingHostName	IdP_Host (obsoleto)	Richiesto	nessuno

PingPortNumber

Il numero di porta da utilizzare per connettersi all'host IdP.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
PingPortNumber	IdP_Port (obsoleto)	Richiesto	nessuno

PingPartnerSpId

L'indirizzo del provider di servizi. Per trovare l'indirizzo del provider di servizi, visita il seguente URL e visualizza il campo Endpoint applicazione SSO.

```
https://your-pf-host-#:9999/pingfederate/your-pf-app#/spConnections
```

Nome del parametro	Alias	Tipo parametro	Valore predefinito
PingPartnerSpId	Partner_SPID (obsoleto)	Richiesto	nessuno

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Per informazioni sui ruoli ARN, consulta l'AWS Security Token Service API AssumeRoleReference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
PreferredRole	preferred_role (obsoleto)	Facoltativo	nessuno

Durata della sessione del ruolo

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni, consulta AssumeRolel'AWS Security Token Service API Reference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
RoleSessionDuration	Duration (obsoleto)	Facoltativo	3600

Lake Formation abilitato

Speciifica se utilizzare l'azione API AssumeDecoratedRoleWithSAML Lake Formation per recuperare le credenziali IAM temporanee anziché l'AssumeRoleWithazione API SAML. AWS STS

Nome del parametro	Alias	Tipo parametro	Valore predefinito
LakeFormationEnabled	nessuno	Facoltativo	FALSE

Credenziali AD FS

Un meccanismo di autenticazione basato su SAML che consente l'autenticazione su Athena tramite Microsoft Active Directory Federation Services (AD FS). Questo metodo presuppone che l'utente abbia già configurato una federazione tra Athena e AD FS.

Provider di credenziali

Il provider di credenziali che sarà utilizzato per autenticare le richieste a AWS. Imposta il valore di questo parametro su `ADFS`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valore da utilizzare
CredentialsProvider	AWSCredentialsProviderClass (obsoleto)	Richiesto	nessuno	`ADFS`

Utente

L'indirizzo e-mail dell'utente AD FS da utilizzare per l'autenticazione con AD FS.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Utente	UID (obsoleto)	Necessario per l'autenticazione basata su moduli. Facoltativo per l'autenticazione integrata di Windows.	nessuno

Password

La password per l'utente AD FS.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
Password	PWD (obsoleto)	Richiesta per l'autenticazione basata su moduli. Facoltativo per l'autenticazione integrata di Windows.	nessuno

Nome host ADFS

L'indirizzo del tuo server AD FS.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
AdfsHostName	IdP_Host (obsoleto)	Richiesto	nessuno

Numero di porta ADFS

Il numero di porta da utilizzare per connettersi al server AD FS.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
AdfsPortNumber	IdP_Port (obsoleto)	Richiesto	nessuno

Relating party ADFS

Il relying party attendibile. Utilizza questo parametro per sovrascrivere l'URL dell'endpoint del relying party AD FS.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
AdfsRelyingParty	LoginToRP (obsoleto)	Facoltativo	`urn:amazon:webservices`

ADFS WIA abilitato

booleano. Utilizza questo parametro per abilitare l'autenticazione integrata di Windows (WIA) con AD FS.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
AdfsWiaEnabled	none	Facoltativo	`FALSE`

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Per informazioni sui ruoli ARN, consulta l'AWS Security Token Service API *AssumeRole*Reference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
PreferredRole	preferred_role (obsoleto)	Facoltativo	nessuno

Durata della sessione del ruolo

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni, consulta `AssumeRole` nella documentazione di riferimento dell'API AWS Security Token Service .

Nome del parametro	Alias	Tipo parametro	Valore predefinito
RoleSessionDuration	Duration (obsoleto)	Facoltativo	`3600`

Lake Formation abilitato

Specifica se utilizzare l'azione dell'API `AssumeDecoratedRoleWithSAML`Lake Formation per recuperare le credenziali IAM temporanee anziché l'`AssumeRoleWithSAML` AWS STS azione API.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
LakeFormationEnabled	`none`	Facoltativo	FALSE

Credenziali Browser Azure AD

Browser Azure AD è un meccanismo di autenticazione basato su SAML che funziona con il gestore delle identità Azure AD e supporta l'autenticazione a più fattori. Diversamente dal meccanismo di autenticazione standard Azure AD, per questo meccanismo non sono necessari nome utente, password o segreto del client nei parametri di connessione. Analogamente al meccanismo di autenticazione standard di Azure AD, Browser Azure AD presuppone anche che l'utente abbia già configurato la federazione tra Athena e Azure AD.

Provider di credenziali

Il provider di credenziali che sarà utilizzato per autenticare le richieste a AWS. Imposta il valore di questo parametro su `BrowserAzureAD`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valore da utilizzare
CredentialsProvider	AWSCredentialsProviderClass (deprecato)	Richiesto	nessuno	`BrowserAzureAD`

ID tenant Azure AD

L'ID tenant dell'applicazione Azure AD.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
AzureAdTenantId	tenant_id (obsoleto)	Richiesto	nessuno

ID client di Azure AD

L'ID client dell'applicazione Azure AD

Nome del parametro	Alias	Tipo parametro	Valore predefinito
AzureAdClientId	client_id (obsoleto)	Richiesto	nessuno

Timeout di risposta del gestore dell'identità

Il tempo, in secondi, prima che il plug-in termini l'attesa della risposta SAML da Azure AD.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
IdpResponseTimeout	idp_response_timeout (obsoleto)	Facoltativo	120

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Per informazioni sui ruoli ARN, consulta l'AWS Security Token Service API AssumeRoleReference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
PreferredRole	preferred_role (obsoleto)	Facoltativo	nessuno

Durata della sessione del ruolo

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni, consulta AssumeRolel'AWS Security Token Service API Reference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
RoleSessionDuration	Duration (obsoleto)	Facoltativo	3600

Lake Formation abilitato

Speciifica se utilizzare l'azione API AssumeDecoratedRoleWithSAML Lake Formation per recuperare le credenziali IAM temporanee anziché l'AssumeRoleWithazione API SAML. AWS STS

Nome del parametro	Alias	Tipo parametro	Valore predefinito
LakeFormationEnabled	nessuno	Facoltativo	FALSE

Credenziali Browser SAML

Browser SAML è un plug-in di autenticazione generico basato su SAML che può funzionare con i provider di identità basati su SAML e che supporta l'autenticazione a più fattori.

Provider di credenziali

Il provider di credenziali che sarà utilizzato per autenticare le richieste a AWS. Imposta il valore di questo parametro su `BrowserSaml`.

Nome del parametro	Alias	Tipo parametro	Valore predefinito	Valore da utilizzare
CredentialsProvider	AWSCredentialsProviderClass (obsoleto)	Richiesto	nessuno	`BrowserSaml`

URL di accesso Single Sign-On

L'URL Single Sign-On per la tua applicazione sul gestore delle identità basato su SAML.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
SsoLoginUrl	login_url (obsoleto)	Richiesto	nessuno

Porta dell'ascoltatore

Il numero della porta utilizzato per ascoltare la risposta SAML. Questo valore deve corrispondere all'URL con cui hai configurato il gestore di identità basato su SAML (ad esempio, `http://localhost:7890/athena`).

Nome del parametro	Alias	Tipo parametro	Valore predefinito
ListenPort	listen_port (obsoleto)	Facoltativo	`7890`

Timeout di risposta del gestore dell'identità

Il tempo, in secondi, prima che il plug-in termini l'attesa della risposta SAML da Azure AD.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
IdpResponseTimeout	idp_response_timeout (obsoleto)	Facoltativo	120

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Per informazioni sui ruoli ARN, consulta l'AWS Security Token Service API AssumeRoleReference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
PreferredRole	preferred_role (obsoleto)	Facoltativo	nessuno

Durata della sessione del ruolo

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni, consulta AssumeRolel'AWS Security Token Service API Reference.

Nome del parametro	Alias	Tipo parametro	Valore predefinito
RoleSessionDuration	Duration (obsoleto)	Facoltativo	3600

Lake Formation abilitato

Speciifica se utilizzare l'azione API AssumeDecoratedRoleWithSAML Lake Formation per recuperare le credenziali IAM temporanee anziché l'AssumeRoleWithazione API SAML. AWS STS

Nome del parametro	Alias	Tipo parametro	Valore predefinito
LakeFormationEnabled	nessuno	Facoltativo	FALSE

Altra configurazione di JDBC 3.x

Nelle sezioni seguenti sono descritte alcune impostazioni di configurazione aggiuntive per il driver JDBC 3.x.

Timeout di rete

Il periodo di attesa, in millisecondi, per una risposta del driver durante una chiamata API ad Athena. Trascorso questo tempo, il driver genererà un'eccezione di timeout.

Il timeout di rete non può essere impostato come parametro di connessione. Per impostarlo, chiama il metodo `setNetworkTimeout` su un oggetto `Connection` JDBC. Questo valore può essere modificato durante il ciclo di vita della connessione JDBC. Il valore predefinito di questo parametro è `infinity`.

L'esempio seguente imposta il timeout di rete su 5.000 millisecondi.

```
...
AthenaDriver driver = new AthenaDriver();
Connection connection = driver.connect(url, connectionParameters);
connection.setNetworkTimeout(null, 5000);
...
```

Timeout delle query

Il periodo di attesa, in secondi, per il completamento di una query su Athena dopo l'invio di una query. Trascorso questo tempo, il driver prova ad annullare la query inviata e genera un'eccezione di timeout.

Il timeout della query non può essere impostato come parametro di connessione. Per impostarlo, chiama il metodo `setQueryTimeout` su un oggetto `Statement` JDBC. Questo valore può essere modificato durante il ciclo di vita di un'istruzione JDBC. Il valore predefinito di questo parametro è `0` (zero). Un valore pari a `0` indica che le query possono essere eseguite fino al completamento (soggetto a Service Quotas (Quote di Servizio)).

L'esempio seguente imposta il timeout della query su 5 secondi.

```
...
AthenaDriver driver = new AthenaDriver();
Connection connection = driver.connect(url, connectionParameters);
Statement statement = connection.createStatement();
statement.setQueryTimeout(5);
...
```

Note di rilascio di Amazon Athena JDBC 3.x

Queste note di rilascio forniscono dettagli su miglioramenti e correzioni nel driver Amazon Athena JDBC 3.x.

3.2.0

Rilasciato il 26 aprile 2020

Miglioramenti

- Prestazioni del funzionamento del catalogo: le prestazioni sono state migliorate per le operazioni di catalogo che non utilizzano caratteri jolly.
- Modifica dell'intervallo minimo di polling: l'impostazione predefinita dell'intervallo di polling minimo è stata modificata per ridurre il numero di chiamate API che il driver effettua ad Athena. I completamenti delle query vengono comunque rilevati il prima possibile.
- Scoperibilità degli strumenti di BI: il driver è stato reso più facilmente individuabile per gli strumenti di business intelligence.
- Mappatura dei tipi di dati: la mappatura dei tipi di dati sui tipi di dati Athena `binary` e `struct` DDL è stata migliorata. `array`
- AWS Versione SDK: la versione AWS SDK utilizzata nel driver è stata aggiornata alla 2.25.34.

Correzioni

- Elenchi di tabelle di cataloghi federati: è stato risolto un problema che causava la restituzione di un elenco vuoto di tabelle da parte dei cataloghi federati.
- GetSchemas: è stato risolto un problema a causa del quale il metodo JDBC DatabaseMetaData#getSchemas recuperava i database solo dal catalogo predefinito anziché da tutti i cataloghi.
- getColumns — È stato risolto un problema che causava la restituzione di un catalogo nullo quando il metodo JDBC DatabaseMetaData #getColumns veniva chiamato con un nome di catalogo nullo.

3.1.0

Rilasciato il 15/02/2020

Miglioramenti

- Supporto aggiunto per l'autenticazione integrata di Windows di Microsoft Active Directory Federation Services (AD FS) e l'autenticazione basata su moduli.
- Per motivi di retrocompatibilità con la versione 2.x, il sottoprotocollo `awsathena` JDBC è ora accettato ma genera un avviso di obsolescenza. `athena`Utilizzate invece il sottoprotocollo JDBC.
- `AwsDataCatalog`è ora l'impostazione predefinita per il parametro del catalogo ed `default` è l'impostazione predefinita per il parametro del database. Queste modifiche assicurano che vengano restituiti i valori corretti per il catalogo e il database correnti anziché null.

- In conformità con la specifica JDBC, `IS_AUTOINCREMENT` e `IS_GENERATEDCOLUMN` ora restituisce una stringa vuota anziché. `NO`
- Il tipo di `int` dati Athena ora viene mappato allo stesso tipo JDBC di Athena anziché a. `integer` `other`
- Quando i metadati della colonna di Athena non contengono i campi `precision` opzionali `scale` e, il driver ora restituisce zero per i valori corrispondenti in `ResultSet` una colonna.
- La versione AWS SDK è stata aggiornata alla 2.21.39.

Correzioni

- È stato risolto un problema `GetQueryResultsStream` che causava un'eccezione quando i risultati di testo semplice di Athena avevano un conteggio delle colonne non coerente con il conteggio delle colonne nei metadati dei risultati di Athena.

3.0.0

Rilasciato il 16/11/23

Il driver Athena JDBC 3.x è il driver di nuova generazione che offre prestazioni e compatibilità migliori. Il driver JDBC 3.x supporta la lettura dei risultati delle query direttamente da Amazon S3, il che migliora le prestazioni delle applicazioni che utilizzano risultati di query di grandi dimensioni. Il nuovo driver ha anche un minor numero di dipendenze da terze parti, il che semplifica l'integrazione con strumenti di BI e applicazioni personalizzate.

Versioni precedenti del driver Athena JDBC 3.x

Si consiglia vivamente di utilizzare la versione più recente del driver JDBC 3.x. La versione più recente del driver contiene i miglioramenti e le correzioni più recenti. Utilizzate una versione precedente solo se l'applicazione presenta incompatibilità con la versione più recente.

Jar uber del driver JDBC

Il seguente download impachetta il driver e tutte le sue dipendenze nello stesso file `.jar`. Questo download viene in genere utilizzato per client SQL di terze parti.

- https://downloads.athena.us-east-1.amazonaws.com/drivers/JDBC/3.1.0/athena-jdbc-3.1.0-with-dependencies.jar

- https://downloads.athena.us-east-1.amazonaws.com/drivers/JDBC/3.0.0/athena-jdbc-3.0.0-with-dependencies.jar

Jar lean del driver JDBC

Il seguente download è un file `.zip` che contiene il file `.jar` lean per il driver e separa i file `.jar` per le dipendenze del driver. Questo download viene in genere utilizzato per applicazioni personalizzate che potrebbero avere dipendenze in conflitto con quelle utilizzate dal driver. Questo download è utile se si desidera scegliere quali dipendenze dei driver includere nel jar lean e quali escludere se l'applicazione personalizzata ne contiene già una o più.

- https://downloads.athena.us-east-1.amazonaws.com/drivers/JDBC/3.1.0/athena-jdbc-3.1.0-lean-jar-and-separate-dependencies-jars.zip
- https://downloads.athena.us-east-1.amazonaws.com/drivers/JDBC/3.0.0/athena-jdbc-3.0.0-lean-jar-and-separate-dependencies-jars.zip

Driver JDBC 2.x di Athena

È possibile usare una connessione JDBC per connettere Athena agli strumenti di business intelligence e ad altre applicazioni, come SQL Workbench. Per completare questa operazione, utilizza i link di Amazon S3 in questa pagina per scaricare, installare e configurare il driver JDBC 2.x di Athena. Per informazioni sulla creazione dell'URL di connessione JDBC, consulta il file scaricabile Guida all'installazione e alla configurazione del driver JDBC. Per informazioni sulle autorizzazioni, consulta Accesso tramite connessioni JDBC e ODBC. Per inoltrare commenti sul driver JDBC, invia un'e-mail a athena-feedback@amazon.com. A partire dalla versione 2.0.24, sono disponibili due versioni del driver: una che include l' AWS SDK e una che non lo include.

Important

Quando utilizzi il driver JDBC, assicurati di rispondere ai seguenti requisiti:

- Open port 444 (apri la porta 444): mantieni la porta 444, che Athena utilizza per trasmettere i risultati delle query, aperta al traffico in uscita. Quando usi un PrivateLink endpoint per connetterti ad Athena, assicurati che il gruppo di sicurezza collegato all'endpoint sia aperto PrivateLink al traffico in entrata sulla porta 444. Se la porta 444 è bloccata, è possibile che venga visualizzato il messaggio di errore [Simba] [AthenajDBC] (100123) Si è verificato un errore. Eccezione durante l'inizializzazione della colonna.

- athena: GetQueryResultsStream policy — Aggiungi l'azione `athena:GetQueryResultsStream` politica ai principali IAM che utilizzano il driver JDBC. Questa operazione di policy non è esposta direttamente con l'API. Viene utilizzata solo con i driver ODBC e JDBC come parte del supporto per i risultati di streaming. Per un esempio di policy, consulta AWS politica gestita: AWSQuicksightAthenaAccess.
- Utilizzo del driver JDBC per più cataloghi dati: per utilizzare il driver JDBC per più cataloghi dati con Athena (ad esempio, quando si utilizza un metastore Hive esterno o query federate), includi `MetadataRetrievalMethod=ProxyAPI` nella stringa di connessione JDBC.
- Driver 4.1: a partire dal 2023, il supporto dei driver per JDBC versione 4.1 non è più disponibile. Non verranno rilasciati ulteriori aggiornamenti. Se utilizzi un driver JDBC 4.1, ti consigliamo vivamente di passare al driver 4.2.

Driver JDBC 2.x con SDK AWS

La versione 2.1.5 del driver JDBC è conforme allo standard di dati JDBC API 4.2 e richiede JDK 8.0 o versione successiva. Per informazioni sul controllo della versione di Java Runtime Environment (JRE) utilizzata, consultare la documentazione Java.

Utilizza il collegamento seguente per scaricare il file `.jar` del driver JDBC 4.2.

- AthenaJDBC42-2.1.5.1000.jar

Il seguente `.zip` file da scaricare contiene il file per JDBC 4.2 e include l'SDK e la documentazione di accompagnamento, le note di rilascio, le licenze e gli accordi. `.jar` AWS

- SimbaAthena

Driver JDBC 2.x senza l'SDK AWS

La versione 2.1.5 del driver JDBC è conforme allo standard di dati JDBC API 4.2 e richiede JDK 8.0 o successivo. Per informazioni sul controllo della versione di Java Runtime Environment (JRE) utilizzata, consultare la documentazione Java.

Utilizza il seguente link per scaricare il file del driver JDBC 4.2 senza l'SDK. `.jar` AWS

- AthenaJDBC42-2.1.5.1001.jar

Il seguente download del file `.zip` contiene il file `.jar` per JDBC 4.2 e la relativa documentazione, le note di rilascio, le licenze e gli accordi. Non include l'SDK. AWS

- SimbaAthena

Note di rilascio, accordo di licenza e avvisi del driver JDBC 2.x

Dopo aver scaricato la versione di cui hai bisogno, leggi le note di rilascio ed esamina il contratto di licenza e gli avvisi.

- Note di rilascio
- Contratto di licenza
- Avvisi
- Licenze di terze parti

Documentazione del driver JDBC 2.x

Scarica la seguente documentazione per il driver:

- Guida all'installazione e alla configurazione del driver JDBC. Utilizza questa guida per installare e configurare il driver.
- Guida alla migrazione del driver JDBC. Utilizza questa guida per eseguire la migrazione da versioni precedenti alla versione corrente.

Connessione ad Amazon Athena con ODBC

Amazon Athena offre due driver ODBC, le versioni 1.x e 2.x. Il driver Athena ODBC 2.x è una nuova alternativa che supporta i sistemi Linux, macOS ARM, macOS Intel e Windows a 64 bit. Il driver Athena 2.x supporta tutti i plugin di autenticazione supportati dal driver ODBC 1.x; inoltre quasi tutti i parametri di connessione sono retrocompatibili.

- Per scaricare il driver ODBC 2.x, consulta Amazon Athena ODBC 2.x.
- Per scaricare il driver ODBC 1.x, consulta Driver ODBC 1.x di Athena.

Argomenti

- Amazon Athena ODBC 2.x

- Driver ODBC 1.x di Athena
- Utilizzo del connettore Power BI di Amazon Athena

Amazon Athena ODBC 2.x

Puoi utilizzare una connessione ODBC per connetterti ad Amazon Athena da molti strumenti e applicazioni del client SQL di terze parti. Configura la connessione ODBC sul computer client.

Considerazioni e limitazioni

- Per informazioni sulla migrazione dal driver ODBC 1.x di Athena al driver ODBC 2.x di Athena, consulta Come migrare al driver ODBC 2.x.
- Quando utilizzi il fetcher S3 con l'opzione di crittografia `CSE_KMS`, il client di Amazon S3 non può decrittografare il risultato archiviato nel bucket di Amazon S3. Come soluzione alternativa, utilizza l'opzione API di streaming Athena per recuperare il set di risultati.

Come scaricare il driver ODBC 2.x

Per scaricare il driver ODBC Amazon Athena 2.x, visita i link in questa pagina.

> **Important**
>
> Quando utilizzi il driver ODBC 2.x, assicurati di valutare i seguenti requisiti:
>
> - Open port 444 (apri la porta 444): mantieni la porta 444, che Athena utilizza per trasmettere i risultati delle query, aperta al traffico in uscita. Quando usi un PrivateLink endpoint per connetterti ad Athena, assicurati che il gruppo di sicurezza collegato all'endpoint sia aperto PrivateLink al traffico in entrata sulla porta 444.
> - athena: GetQueryResultsStream policy — Aggiungi l'azione `athena:GetQueryResultsStream` politica ai principi IAM che utilizzano il driver ODBC. Questa operazione di policy non è esposta direttamente con l'API. Viene utilizzata solo con i driver ODBC e JDBC come parte del supporto per i risultati di streaming. Per un esempio di policy, consulta AWS politica gestita: AWSQuicksightAthenaAccess.

Linux

Versione driver	Collegamento per il download
ODBC 2.0.3.0 per Linux a 64 bit	

macOS (ARM)

Versione driver	Collegamento per il download
ODBC 2.0.3.0 per macOS 64 bit (ARM)	Driver ODBC per macOS a 64 bit 2.0.3.0 (ARM) Driver ODBC per macOS bit 2.0.3.0 (ARM)

macOS (Intel)

Versione driver	Collegamento per il download
ODBC 2.0.3.0 per macOS a 64 bit (Intel)	Driver ODBC per macOS a 64 bit 2.0.3.0 (Intel) Driver ODBC per macOS bit 2.0.3.0 (Intel)

Windows

Versione driver	Collegamento per il download
ODBC 2.0.3.0 per Windows a 64 bit	

Argomenti

- Note di rilascio per ODBC 2.x di Amazon Athena

Guida introduttiva al driver ODBC 2.x

Usa le informazioni in questa sezione per iniziare a usare il driver Amazon Athena ODBC 2.x. Il driver è supportato sui sistemi operativi Windows, Linux e macOS.

Argomenti

- Windows
- Linux
- macOS

Windows

Se desideri utilizzare un computer client Windows per accedere ad Amazon Athena, è necessario il driver Amazon Athena ODBC.

Requisiti di sistema di Windows

Installa il driver Amazon Athena ODBC sui computer client che accederanno direttamente ai database Amazon Athena anziché utilizzare un browser Web.

Il sistema Windows che utilizzi deve soddisfare i seguenti requisiti:

- Hai i diritti di amministratore
- Uno dei seguenti sistemi operativi:
  - Windows 11, 10 o 8,1
  - Windows Server 2019, 2016 o 2012
- Almeno 100 MB di spazio su disco disponibile.
- Microsoft Visual C++ Redistributable for Visual Studio per Windows a 64 bit è installato.

Installazione del driver ODBC di Amazon Athena

Per scaricare e installare il driver Amazon Athena ODBC per Windows

1. Scarica il file di installazione di `AmazonAthenaODBC-2.x.x.x.msi`.
2. Avvia il file di installazione, quindi scegli Avanti.

3. Per accettare i termini del contratto di licenza, seleziona la casella di controllo, quindi scegli Avanti.
4. Per modificare la posizione di installazione, scegliete Sfoglia, individuate la cartella desiderata, quindi scegliete OK.
5. Per accettare la posizione di installazione, scegliete Avanti.
6. Scegli Installa.
7. Al termine dell'installazione, seleziona Termina.

Modi per impostare le opzioni di configurazione del driver

Per controllare il comportamento del driver Amazon Athena ODBC in Windows, puoi impostare le opzioni di configurazione del driver nei seguenti modi:

- Nel programma ODBC Data Source Administrator quando configuri un nome di origine dati (DSN).
- Aggiungendo o modificando le chiavi del registro di Windows nella seguente posizione:

```
HKEY_LOCAL_MACHINE\SOFTWARE\ODBC\ODBC.INI\YOUR_DSN_NAME
```

- Impostando le opzioni del driver nella stringa di connessione quando ci si connette a livello di programmazione.

Configurazione del nome di un'origine dati in Windows

Dopo aver scaricato e installato il driver ODBC, è necessario aggiungere un nome di origine dati (DSN) al computer client o all'istanza Amazon EC2. Gli strumenti client SQL utilizzano questa origine dati per connettersi ed eseguire query in Amazon Athena.

Per creare una voce DSN di sistema

1. Dal menu Start di Windows, fai clic con il pulsante destro del mouse su Origini dati ODBC (64 bit), quindi seleziona Altro, Esegui come amministratore.
2. In Amministratore origine dati ODBC, seleziona la scheda Driver.
3. Nella colonna Nome, verifica che Amazon Athena ODBC (x64) sia presente.
4. Esegui una di queste operazioni:

- Per configurare il driver per tutti gli utenti del computer, scegli la scheda Sistema DSN. Poiché le applicazioni che utilizzano un account diverso per caricare i dati potrebbero non essere in grado di rilevare i DSN utente.

  > **Note**
  >
  > L'utilizzo dell'opzione DSN di sistema richiede privilegi amministrativi.

- Per configurare il driver solo per il tuo account utente, scegli la scheda DSN utente.

5. Scegli Aggiungi. Si aprirà la finestra di dialogo Crea nuova origine dati.
6. Seleziona ODBC (x64) di Amazon Athena, quindi seleziona Termina.
7. Nella finestra di dialogo Configurazione ODBC di Amazon Athena, inserisci le seguenti informazioni. Per informazioni dettagliate su queste opzioni, consulta Parametri di connessione ODBC 2.x principali.

   - Per Nome origine dati, inserisci un nome che desideri utilizzare per identificare l'origine dati.
   - Per Descrizione, inserisci una descrizione per aiutarti a identificare l'origine dati.
   - Per Regione, inserisci il nome di Athena della Regione AWS che utilizzerai (ad esempio, **`us-west-1`**).
   - Per Catalogo, inserisci il nome del catalogo Amazon Athena. L'impostazione predefinita è AwsDataCatalog, utilizzata da. AWS Glue
   - Per Database, inserisci il nome del database Amazon Athena. Il valore predefinitoo è default.
   - Per Nome gruppo, inserisci il nome del gruppo di lavoro Amazon Athena. L'impostazione predefinita è primaria.
   - Per S3 Output Location, inserisci la posizione in Amazon S3 in cui verranno archiviati i risultati della query (ad esempio**`s3://DOC-EXAMPLE-BUCKET/`**,).
   - (Facoltativo) Per Opzioni di crittografia seleziona un'opzione di crittografia. Il valore predefinito è `NOT_SET`.
   - (Facoltativo) Per Chiave KMS scegli una chiave KMS di crittografia, se necessario.
8. Per specificare le opzioni di configurazione per l'autenticazione IAM, seleziona Opzioni di autenticazione.
9. Immetti le seguenti informazioni:

- Per Tipo autenticazione, seleziona Credenziali IAM. Questa è l'impostazione predefinita. Per ulteriori informazioni sui tipi di autenticazioni disponibili, consulta Opzioni di autenticazione.
- In Nome utente digita un nome utente.
- Per Password immetti una password.
- Per Session Token, inserisci un token di sessione se desideri utilizzare credenziali temporanee AWS . Per informazioni sulle credenziali temporanee, consulta Using temporary credenziali with AWS resources nella IAM User Guide.

10. Scegli OK.
11. Nella parte inferiore della finestra di dialogo di Configurazione ODBC di Amazon Athena, seleziona Test. Se il computer client si connette correttamente ad Amazon Athena, la casella di Test connessione riporta Connessione riuscita. In caso contrario, la casella riporta Connessione non riuscita con le informazioni di errore corrispondenti.
12. Scegli OK per chiudere il test di connessione. L'origine dati creata ora viene visualizzata nell'elenco dei nomi di origine dati.

Utilizzo di una connessione senza DSN in Windows

È possibile utilizzare una connessione senza DSN per connettersi a un database senza un DSN (Data Source Name). L'esempio seguente mostra una stringa di connessione per il driver ODBC Amazon Athena (x64) che si connette ad Amazon Athena.

```
DRIVER={Amazon Athena ODBC (x64)};Catalog=AwsDataCatalog;AwsRegion=us-
west-1;Schema=test_schema;S3OutputLocation=
s3://DOC-EXAMPLE-BUCKET/;AuthenticationType=IAM Credentials;UID=YOUR_UID;PWD=YOUR_PWD;
```

Linux

Se desideri utilizzare un computer client Linux per accedere ad Amazon Athena, è necessario il driver Amazon Athena ODBC.

Requisiti di sistema Linux

Ogni computer client Linux su cui si installa il driver deve soddisfare i seguenti requisiti.

- Hai accesso root.

- Usa una delle seguenti distribuzioni Linux:
  - Red Hat Enterprise Linux (RHEL) 7 o 8
  - CentOS 7 o 8.
- Disponi di 100 MB di spazio su disco.
- Usa la versione 2.3.1 o successiva di unixODBC.
- Usa la versione 2.26 o successiva della libreria GNU C (glibc).

Installazione del connettore dati ODBC su Linux

Utilizza la seguente procedura per installare il driver Amazon Athena ODBC su un sistema operativo Linux.

Per installare il driver Amazon Athena ODBC su Linux

1. Utilizza uno dei comandi seguenti:

   ```
   sudo rpm -Uvh AmazonAthenaODBC-2.X.Y.Z.rpm
   ```

   oppure

   ```
   sudo yum --nogpgcheck localinstall AmazonAthenaODBC-2.X.Y.Z.rpm
   ```

2. Al termine dell'installazione, inserisci uno dei seguenti comandi per verificare che il driver sia installato:

   - ```
     yum list | grep amazon-athena-odbc-driver
     ```

     Output:

     ```
     amazon-athena-odbc-driver.x86_64 2.0.2.1-1.amzn2int installed
     ```

   - ```
     rpm -qa | grep amazon
     ```

     Output:

     ```
     amazon-athena-odbc-driver-2.0.2.1-1.amzn2int.x86_64
     ```

Configurazione del nome di un'origine dati su Linux

Dopo aver installato il driver, puoi trovare esempi `.odbc.ini` e `.odbcinst.ini` file nella seguente posizione:

- `/opt/athena/odbc/ini/`.

Usa i `.ini` file in questa posizione come esempi per configurare il driver ODBC e il nome dell'origine dati (DSN) di Amazon Athena.

> **Note**
>
> Per impostazione predefinita, i gestori di driver ODBC utilizzano i file di configurazione nascosti `.odbc.ini` e `.odbcinst.ini`, che si trovano nella home directory.

Per specificare il percorso dei `.odbcinst.ini` file `.odbc.ini` and utilizzando unixODBC, effettuate le seguenti operazioni.

Per specificare le posizioni dei **`.ini`** file ODBC utilizzando unixODBC

1. Imposta `ODBCINI` il percorso completo e il nome del `odbc.ini` file, come nell'esempio seguente.

   ```
   export ODBCINI=/opt/athena/odbc/ini/odbc.ini
   ```

2. Imposta `ODBCSYSINI` il percorso completo della directory che contiene il `odbcinst.ini` file, come nell'esempio seguente.

   ```
   export ODBCSYSINI=/opt/athena/odbc/ini
   ```

3. Immettete il seguente comando per verificare che stiate utilizzando il gestore driver unixODBC e i file corretti: `odbc*.ini`

   ```
   username % odbcinst -j
   ```

   Output di esempio

   ```
   unixODBC 2.3.1
   ```

```
DRIVERS............: /opt/athena/odbc/ini/odbcinst.ini
SYSTEM DATA SOURCES: /opt/athena/odbc/ini/odbc.ini
FILE DATA SOURCES..: /opt/athena/odbc/ini/ODBCDataSources
USER DATA SOURCES..: /opt/athena/odbc/ini/odbc.ini
SQLULEN Size.......: 8
SQLLEN Size........: 8
SQLSETPOSIROW Size.: 8
```

4. Se desideri utilizzare un nome di origine dati (DSN) per connetterti al tuo data store, configura il `odbc.ini` file per definire i nomi delle origini dati (DSN). Imposta le proprietà del `odbc.ini` file per creare un DSN che specifichi le informazioni di connessione per il tuo data store, come nell'esempio seguente.

```
[ODBC Data Sources]
athena_odbc_test=Amazon Athena ODBC (x64)

[ATHENA_WIDE_SETTINGS]  # Special DSN-name to signal driver about logging
 configuration.
LogLevel=0              # To enable ODBC driver logs, set this to 1.
UseAwsLogger=0          # To enable AWS-SDK logs, set this to 1.
LogPath=/opt/athena/odbc/logs/ # Path to store the log files. Permissions to the
 location are required.

[athena_odbc_test]
Driver=/opt/athena/odbc/lib/libathena-odbc.so
AwsRegion=us-west-1
Workgroup=primary
Catalog=AwsDataCatalog
Schema=default
AuthenticationType=IAM Credentials
UID=
PWD=
S3OutputLocation=s3://DOC-EXAMPLE-BUCKET/
```

5. Configura il `odbcinst.ini` file, come nell'esempio seguente.

```
[ODBC Drivers]
Amazon Athena ODBC (x64)=Installed

[Amazon Athena ODBC (x64)]
Driver=/opt/athena/odbc/lib/libathena-odbc.so
Setup=/opt/athena/odbc/lib/libathena-odbc.so
```

6. Dopo aver installato e configurato il driver Amazon Athena ODBC, usa lo strumento da `isql` riga di comando unixODBC per verificare la connessione, come nell'esempio seguente.

```
username % isql -v "athena_odbc_test"
+---------------------------------------+
| Connected!                            |
|                                       |
| sql-statement                         |
| help [tablename]                      |
| quit                                  |
|                                       |
+---------------------------------------+
SQL>
```

macOS

Se desideri utilizzare un computer client macOS per accedere ad Amazon Athena, è necessario il driver Amazon Athena ODBC.

Requisiti di sistema macOS

Ogni computer macOS su cui si installa il driver deve soddisfare i seguenti requisiti.

- Usa macOS versione 14 o successiva.
- Tieni a disposizione 100 MB di spazio su disco.
- Utilizzare la versione 3.52.16 o successiva di IODBC.

Installazione del connettore dati ODBC su macOS

Utilizza la seguente procedura per scaricare e installare il driver Amazon Athena ODBC per i sistemi operativi macOS.

Per scaricare e installare il driver Amazon Athena ODBC per macOS

1. Scarica il file del pacchetto. `.pkg`
2. Fare doppio clic sul file `.pkg`.
3. Segui i passaggi della procedura guidata per installare il driver.
4. Nella pagina del Contratto di licenza, premi Continua, quindi scegli Accetto.

5. Scegli Installa.
6. Al termine dell'installazione, seleziona Termina.
7. Inserisci il seguente comando per verificare che il driver sia installato:

```
> pkgutil --pkgs | grep athenaodbc
```

A seconda del sistema in uso, l'output può essere simile a uno dei seguenti.

```
com.amazon.athenaodbc-x86_64.Config
com.amazon.athenaodbc-x86_64.Driver
```

oppure

```
com.amazon.athenaodbc-arm64.Config
com.amazon.athenaodbc-arm64.Driver
```

Configurazione del nome di un'origine dati su macOS

Dopo aver installato il driver, puoi trovare esempi `.odbc.ini` e `.odbcinst.ini` file nelle seguenti posizioni:

- Computer con processore Intel: `/opt/athena/odbc/x86_64/ini/`
- Computer con processore ARM: `/opt/athena/odbc/arm64/ini/`

Usa i `.ini` file in questa posizione come esempi per configurare il driver ODBC e il nome dell'origine dati (DSN) di Amazon Athena.

> **Note**
>
> Per impostazione predefinita, i gestori di driver ODBC utilizzano i file di configurazione nascosti `.odbc.ini` e`.odbcinst.ini`, che si trovano nella home directory.

Per specificare il percorso dei `.odbcinst.ini` file `.odbc.ini` and utilizzando il driver manager IODBC, effettuate le seguenti operazioni.

Per specificare le posizioni dei **`.ini`** file ODBC utilizzando il gestore driver IODBC

1. Imposta `ODBCINI` sul percorso completo e sul nome file del file `odbc.ini`.

   - Per i computer macOS con processori Intel, utilizzare la seguente sintassi.

     ```
     export ODBCINI=/opt/athena/odbc/x86_64/ini/odbc.ini
     ```

   - Per i computer macOS con processori ARM, usa la seguente sintassi.

     ```
     export ODBCINI=/opt/athena/odbc/arm64/ini/odbc.ini
     ```

2. Imposta `ODBCSYSINI` sul percorso completo e sul nome file del file `odbcinst.ini`.

   - Per i computer macOS con processori Intel, utilizzare la seguente sintassi.

     ```
     export ODBCSYSINI=/opt/athena/odbc/x86_64/ini/odbcinst.ini
     ```

   - Per i computer macOS con processori ARM, usa la seguente sintassi.

     ```
     export ODBCSYSINI=/opt/athena/odbc/arm64/ini/odbcinst.ini
     ```

3. Se desideri utilizzare un nome di origine dati (DSN) per connetterti al tuo data store, configura il `odbc.ini` file per definire i nomi delle sorgenti di dati (DSN). Imposta le proprietà del `odbc.ini` file per creare un DSN che specifichi le informazioni di connessione per il tuo data store, come nell'esempio seguente.

   ```
   [ODBC Data Sources]
   athena_odbc_test=Amazon Athena ODBC (x64)

   [ATHENA_WIDE_SETTINGS] # Special DSN-name to signal driver about logging
    configuration.
   LogLevel=0             # set to 1 to enable ODBC driver logs
   UseAwsLogger=0         # set to 1 to enable AWS-SDK logs
   LogPath=/opt/athena/odbc/logs/ # Path to store the log files. Permissions to the
    location are required.

   [athena_odbc_test]
   Description=Amazon Athena ODBC (x64)
   # For ARM:
   Driver=/opt/athena/odbc/arm64/lib/libathena-odbc-arm64.dylib
   # For Intel:
   ```

```
# Driver=/opt/athena/odbc/x86_64/lib/libathena-odbc-x86_64.dylib
AwsRegion=us-west-1
Workgroup=primary
Catalog=AwsDataCatalog
Schema=default
AuthenticationType=IAM Credentials
UID=
PWD=
S3OutputLocation=s3://DOC-EXAMPLE-BUCKET/
```

4. Configura il `odbcinst.ini` file, come nell'esempio seguente.

```
[ODBC Drivers]
Amazon Athena ODBC (x64)=Installed

[Amazon Athena ODBC (x64)]
# For ARM:
Driver=/opt/athena/odbc/arm64/lib/libathena-odbc-arm64.dylib
Setup=/opt/athena/odbc/arm64/lib/libathena-odbc-arm64.dylib
# For Intel:
# Driver=/opt/athena/odbc/x86_64/lib/libathena-odbc-x86_64.dylib
# Setup=/opt/athena/odbc/x86_64/lib/libathena-odbc-x86_64.dylib
```

5. Dopo aver installato e configurato il driver Amazon Athena ODBC, utilizza lo strumento da `iodbctest` riga di comando per verificare la connessione, come nell'esempio seguente.

```
username@ % iodbctest
iODBC Demonstration program
This program shows an interactive SQL processor
Driver Manager: 03.52.1623.0502

Enter ODBC connect string (? shows list): ?

DSN                             | Driver
-------------------------------------------------------------------------------
athena_odbc_test                | Amazon Athena ODBC (x64)

Enter ODBC connect string (? shows list): DSN=athena_odbc_test;
Driver: 2.0.2.1 (Amazon Athena ODBC Driver)

SQL>
```

Parametri di connessione Athena ODBC 2.x

Le opzioni della finestra di dialogo di Configurazione ODBC di Amazon Athena includono Opzioni autenticazione, Opzioni avanzate, Opzioni di log, Sovrascrittura endpoint e Opzioni proxy. Per maggiori informazioni su ognuno, consulta i link corrispondenti.

- Parametri di connessione ODBC 2.x principali
- Opzioni di autenticazione
- Opzioni avanzate
- Opzioni di registrazione
- Sostituzioni degli endpoint
- Opzioni proxy

Parametri di connessione ODBC 2.x principali

Le seguenti sezioni descrivono ciascuno dei principali parametri di connessione.

Nome origine dati

Specifica il nome della tua origine dati.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
DSN	Facoltativo per tipi di connessione senza DSN	`none`	`DSN=AmazonAthena0dbcUsWest1;`

Descrizione

Contiene descrizione della tua origine dati.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
Descrizione	Facoltativo	`none`	`Description=Connection`

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
			`to Amazon Athena us-west-1;`

Catalogo

Specifica il nome del catalogo dati. Per ulteriori informazioni sui cataloghi, consulta DataCatalogAmazon Athena API Reference.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
Catalogo	Facoltativo	`AwsDataCatalog`	`Catalog=AwsDataCatalog;`

Regione

Specifica il. Regione AWS Per informazioni su Regioni AWS, vedere Regioni e zone di disponibilità.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AwsRegion	Obbligatorio	`none`	`AwsRegion =us-west-1;`

Database

Specifica il nome del database. Per ulteriori informazioni sui database, consulta Database nella Documentazione di riferimento dell'API di Amazon Athena.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
Schema	Facoltativo	`default`	`Schema=default;`

Gruppo di lavoro

Specifica il nome del gruppo di lavoro. Per ulteriori informazioni sui gruppi di lavoro, consulta WorkGroupAmazon Athena API Reference.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
Gruppo di lavoro	Facoltativo	`primary`	`Workgroup =primary;`

Percorso di output

Specifica il percorso su Amazon S3 in cui sono archiviati i risultati della query. Per ulteriori informazioni sulla posizione di output, consulta ResultConfigurationAmazon Athena API Reference.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
S3 OutputLocation	Obbligatorio	none	`S3OutputL ocation=s3:// DOC-EXAMPLE-BU CKET/;`

Opzioni di crittografia

Nome parametro di dialogo: opzioni di crittografia

Specifica l'opzione di crittografia. Per ulteriori informazioni sulle opzioni di crittografia, consulta EncryptionConfigurationAmazon Athena API Reference.

Nome stringa connessione	Tipo parametro	Valore predefini to	Valori possibili	Esempio stringa connessione
S3 OutputEnc Option	Facoltativo	none	`NOT_SET`, `SSE_S3`,	`S3OutputE ncOption= SSE_S3;`

Nome stringa connessione	Tipo parametro	Valore predefinito	Valori possibili	Esempio stringa connessione
			`SSE_KMS`, `CSE_KMS`	

Chiave KMS

Specifica una chiave KMS per la crittografia. Per ulteriori informazioni sulla configurazione della crittografia per le chiavi KMS, consulta EncryptionConfigurationAmazon Athena API Reference.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
Chiave KMS S3 OutputEnc	Facoltativo	`none`	`S3OutputEncKMSKey=your_key;`

Test di connessione

ODBC Data Source Administrator offre un'opzione di Test che puoi utilizzare per eseguire il test della tua connessione ODBC 2.x ad Amazon Athena. Per le fasi, consulta Configurazione del nome di un'origine dati in Windows. Quando si verifica una connessione, il driver ODBC richiama l'azione API GetWorkGroupAthena. La chiamata utilizza il tipo di autenticazione e il provider di credenziali corrispondente che hai indicato per recuperare le credenziali. Non è previsto alcun costo per il test di connessione quando utilizzi il driver ODBC 2.x. Il test non genera risultati di query nel bucket Amazon S3.

Opzioni di autenticazione

Puoi connetterti ad Amazon Athena utilizzando i seguenti tipi di autenticazione. Per tutti i tipi, il nome della stringa di connessione è `AuthenticationType`, il tipo di parametro è `Required` e il valore predefinito è `IAM Credentials`. Per informazioni sui parametri previsti per il tipo di autenticazione, visita il link corrispondente. Per i parametri di autenticazione comuni, consulta Parametri comuni di autenticazione.

Tipo di autenticazione	Esempio stringa connessione
Credenziali IAM	`AuthenticationType=IAM Credentials;`
Profilo IAM	`AuthenticationType=IAM Profile;`
AD FS	`AuthenticationType=ADFS;`
Azure AD	`AuthenticationType=AzureAD;`
Browser Azure AD	`AuthenticationType=BrowserAzureAD;`
Browser SAML	`AuthenticationType=BrowserSAML;`
Browser SSO OIDC	`AuthenticationType=BrowserSSOOIDC;`
Credenziali predefinite	`AuthenticationType=Default Credentials;`
Credenziali esterne	`AuthenticationType=External Credentials;`
Profilo dell'istanza	`AuthenticationType=Instance Profile;`
JWT	`AuthenticationType=JWT;`
Okta	`AuthenticationType=Okta;`
Ping	`AuthenticationType=Ping;`

Credenziali IAM

Puoi utilizzare le tue credenziali IAM per connetterti ad Amazon Athena con il driver ODBC utilizzando i parametri della stringa di connessione descritti in questa sezione.

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Richiesto	`IAM Credentials`	`AuthenticationType=IAM Credentials;`

ID utente

Il tuo ID della chiave di AWS accesso. Per ulteriori informazioni sulle chiavi di accesso, consulta le credenziali di sicurezza AWS nella Guida per l'utente IAM.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
UID	Richiesto	`none`	`UID=AKIAIOSFODNN7EXAMPLE;`

Password

L'ID della tua chiave AWS segreta. Per ulteriori informazioni sulle chiavi di accesso, consulta le credenziali di sicurezza AWS nella Guida per l'utente IAM.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
PWD	Richiesto	`none`	`PWD=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKE;`

Token di sessione

Se utilizzi AWS credenziali temporanee, devi specificare un token di sessione. Per informazioni sulle credenziali temporanee, consulta la sezione relativa alle Credenziali di sicurezza temporanee in IAM nella Guida per l'utente IAM.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
SessionToken	Facoltativo	none	`SessionToken=AQoDYXdzEJr...<remainder of session token>;`

Profilo IAM

Puoi configurare un profilo denominato per connetterti ad Amazon Athena utilizzando il driver ODBC. Per utilizzare le credenziali disponibili nel profilo dell'istanza Amazon EC2 di hosting, imposta il parametro `credential_source` su `Ec2InstanceMetadata`. Se vuoi utilizzare un provider di credenziali personalizzato in un profilo denominato, specifica un valore per il parametro `plugin_name` nella configurazione del profilo.

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Obbligatorio	`IAM Credentials`	`AuthenticationType=IAM Profile;`

Profilo AWS

Il nome del profilo da utilizzare per la connessione ODBC. Per ulteriori informazioni sui profili, consulta Utilizzo di profili designati nella Guida per l'utente di AWS Command Line Interface.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
ProfiloAWS	Obbligatorio	none	`AWSProfile=default;`

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Il parametro del ruolo preferito viene utilizzato quando il provider di credenziali personalizzate viene specificato dal parametro `plugin_name` nella configurazione del profilo. Per ulteriori informazioni sui ruoli ARN, consulta AssumeRole nella Documentazione di riferimento dell'API di AWS Security Token Service

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
preferred_role	Facoltativo	none	`preferred_role=arn:aws:IAM::123456789012:id/user1;`

Durata della sessione

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni sulla durata della sessione, consulta AssumeRole nella Documentazione di riferimento dell'API di AWS Security Token Service. Il parametro di durata della sessione viene utilizzato quando il provider di credenziali personalizzate viene specificato dal parametro `plugin_name` nella configurazione del profilo.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
durata	Facoltativo	`900`	`duration=900;`

Nome del plugin

Specifica il nome di un provider di credenziali personalizzate utilizzato in un profilo denominato. Questo parametro può assumere gli stessi valori del campo Tipo di autenticazione dell'amministratore dell'origine dati ODBC, ma viene utilizzato solo dalla configurazione `AWSProfile`.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
plugin_name	Facoltativo	`none`	`plugin_name=AzureAD;`

AD FS

AD FS è un plug-in di autenticazione basato su SAML che funziona con il provider di identità Active Directory Federation Service (AD FS). Il plug-in supporta l'autenticazione integrata di Windows e l'autenticazione basata su moduli. Se utilizzi l'autenticazione integrata di Windows, puoi omettere il nome utente e la password. Per ulteriori informazioni sulla configurazione di AD FS e Athena, consulta Configurazione dell'accesso federato ad Amazon Athena per utenti Microsoft AD FS utilizzando un client ODBC.

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Obbligatorio	`IAM Credentials`	`AuthenticationType=ADFS;`

ID utente

Il tuo nome utente per connetterti al server AD FS. Per l'autenticazione integrata di Windows, puoi omettere il nome utente. Se per la configurazione di AD FS è necessario un nome utente, bisogna fornirlo nel parametro di connessione.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
UID	Facoltativo per l'autenticazione integrata di Windows	none	`UID=domain\username;`

Password

La tua password per connetterti al server AD FS. Come per il campo del nome utente, puoi omettere il nome utente se utilizzi l'autenticazione integrata di Windows. Se per la configurazione di AD FS è necessaria una password, bisogna fornirla nel parametro di connessione.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
PWD	Facoltativo per l'autenticazione integrata di Windows	none	`PWD=password_3EXAMPLE;`

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Se l'asserzione SAML prevede più ruoli, puoi specificare questo parametro per scegliere il ruolo da assumere. Questo ruolo deve essere presente nell'asserzione SAML. Per ulteriori informazioni sui ruoli ARN, consulta AssumeRole nella Documentazione di riferimento dell'API di AWS Security Token Service

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
preferred_role	Facoltativo	none	`preferred_role=arn:aws:IAM::123456789012:id/user1;`

Durata della sessione

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni sulla durata della sessione, consulta AssumeRole nella Documentazione di riferimento dell'API di AWS Security Token Service.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
durata	Facoltativo	`900`	`duration=900;`

Host IdP

Il nome dell'host del servizio AD FS.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
idp_host	Require	`none`	`idp_host=<server-name>.<company.com>;`

Porta IdP

La porta da utilizzare per connettersi all'host AD FS.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
idp_port	Obbligatorio	`none`	`idp_port=443;`

LoginToRP

Il relying party attendibile. Utilizza questo parametro per sovrascrivere l'URL dell'endpoint del relying party AD FS.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
LoginToRP	Facoltativo	`urn:amazon:webservices`	`LoginToRP=trustedparty;`

Azure AD

Azure AD è un plug-in di autenticazione basato su SAML che funziona con il provider di identità Azure AD. Questo plugin non supporta l'autenticazione a più fattori (MFA). Se hai bisogno del supporto alla MFA, considera la possibilità di utilizzare il plug-in `BrowserAzureAD`.

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Obbligatorio	`IAM Credentials`	`AuthenticationType=AzureAD;`

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Per informazioni sui ruoli ARN, consulta AssumeRole nella Documentazione di riferimento dell'API di AWS Security Token Service

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
preferred_role	Facoltativo	none	`preferred_role=arn:aws:iam::123456789012:id/user1;`

Durata della sessione

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni, consulta AssumeRole nella Documentazione di riferimento dell'API di AWS Security Token Service.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
durata	Facoltativo	`900`	`duration=900;`

ID tenant

Specifica l'ID del tenant dell'applicazione.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
idp_tenant	Obbligatorio	none	`idp_tenant=123zz112z-z12d-1z1f-11zz-f111aa111234;`

ID client

Specifica l'ID del client dell'applicazione.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
client_id	Obbligatorio	none	`client_id=9178ac27-a1bc-1a2b-1a2b-a123abcd1234;`

Client secret

Specifica il secret del client.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
client_secret	Obbligatorio	none	`client_secret=zG12q~.xzG1xxxZ1wX1.~ZzXXX1XxkHZizeT1zzZ;`

Browser Azure AD

Browser Azure AD è un plug-in di autenticazione basato su SAML che funziona con il provider di identità Azure AD e supporta l'autenticazione a più fattori. Diversamente dal plug-in standard Azure AD, per questo plug-in non sono necessari nome utente, password o secret del client nei parametri di connessione.

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Obbligatorio	`IAM Credentials`	`AuthenticationType=BrowserAzureAD;`

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Se l'asserzione SAML prevede più ruoli, puoi specificare questo parametro per scegliere il ruolo da assumere. Il ruolo specificato deve essere presente nell'asserzione SAML. Per ulteriori informazioni sui ruoli ARN, consulta AssumeRole nella Documentazione di riferimento dell'API di AWS Security Token Service

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
preferred_role	Facoltativo	none	`preferred_role=arn:aws:IAM::123456789012:id/user1;`

Durata della sessione

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni sulla durata della sessione, consulta AssumeRole nella Documentazione di riferimento dell'API di AWS Security Token Service.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
durata	Facoltativo	`900`	`duration=900;`

ID tenant

Specifica l'ID del tenant dell'applicazione.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
idp_tenant	Obbligatorio	none	`idp_tenant=123zz112z-z12d-1z1f-11zz-f111aa111234;`

ID client

Specifica l'ID del client dell'applicazione.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
client_id	Obbligatorio	none	`client_id=9178ac27-a1bc-1a2b-1a2b-a123abcd1234;`

Timeout

La durata, in secondi, prima che il plugin termini l'attesa della risposta SAML da Azure AD.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
timeout	Facoltativo	`120`	`timeout=90;`

Abilitazione della cache dei file Azure

Abilita una cache delle credenziali temporanee. Questo parametro di connessione consente di memorizzare nella cache le credenziali temporanee e di riutilizzarle tra più processi. Utilizza questa opzione per ridurre il numero di finestre del browser aperte quando utilizzi strumenti di BI come Microsoft Power BI.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
browser_azure_cache	Facoltativo	`1`	`browser_azure_cache=0;`

Browser SAML

Browser SAML è un plug-in di autenticazione basato su SAML che può funzionare con i provider di identità basati su SAML e può supportare l'autenticazione a più fattori. Per informazioni dettagliate sulla configurazione, consulta Configurazione di Single Sign-On tramite ODBC, SAML 2.0 e il provider di identità Okta.

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Obbligatorio	`IAM Credentials`	`AuthenticationType=BrowserSAML;`

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Se l'asserzione SAML prevede più ruoli, puoi specificare questo parametro per scegliere il ruolo da assumere. Questo ruolo deve essere presente nell'asserzione SAML. Per ulteriori informazioni sui ruoli ARN, consulta AssumeRole nella Documentazione di riferimento dell'API di AWS Security Token Service

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
preferred_role	Facoltativo	none	`preferred_role=arn:aws:IAM::123456789012:id/user1;`

Durata della sessione

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni, consulta AssumeRole nella Documentazione di riferimento dell'API di AWS Security Token Service.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
durata	Facoltativo	`900`	`duration=900;`

URL di accesso

L'URL Single Sign-On che viene visualizzato per la tua applicazione.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
login_url	Obbligatorio	none	`login_url=https://trial-1234567.okta.com/app/trial-123`

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
			`4567_okta browsersaml_1/ zzz4izzzAzDFB zZz1234/sso/ saml;`

Porta dell'ascoltatore

Il numero della porta utilizzato per ascoltare la risposta SAML. Questo valore deve corrispondere all'URL di IAM Identity Center con cui hai configurato l'IdP (ad esempio, `http://localhost:7890/athena`).

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
listen_port	Facoltativo	`7890`	`listen_po rt=7890;`

Timeout

La durata, in secondi, prima che il plugin termini l'attesa della risposta SAML dal provider di identità.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
timeout	Facoltativo	`120`	`timeout=90;`

Browser SSO OIDC

Browser SSO OIDC è un plug-in di autenticazione che funziona con. AWS IAM Identity Center Per informazioni sull'attivazione e l'utilizzo di IAM Identity Center, consulta Fase 1: Abilita IAM Identity Center nella Guida per l'utente di AWS IAM Identity Center .

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Richiesto	`IAM Credentials`	`AuthenticationType=BrowserSSOOIDC;`

URL di avvio di IAM Identity Center

L'URL del portale di accesso. AWS L'azione dell'StartDeviceAuthorizationAPI IAM Identity Center utilizza questo valore per il `startUrl` parametro.

Per copiare l'URL del portale di AWS accesso

1. Accedere AWS Management Console e aprire la AWS IAM Identity Center console all'indirizzo https://console.aws.amazon.com/singlesignon/.
2. Nel pannello di navigazione scegli Impostazioni.
3. Nella pagina Impostazioni, sotto la voce Origine identità, scegli l'icona degli appunti per l'accesso AWS all'URL del portale.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
sso_oidc_start_url	Richiesto	none	`sso_oidc_start_url=https://app_id.awsapps.com/start;`

Regione IAM Identity Center

Regione AWS Dove è configurato il tuo SSO. I client `SSOOIDCClient` e `SSOClient` AWS SDK utilizzano questo valore per il `region` parametro.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
sso_oidc_region	Richiesto	none	`sso_oidc_region=us-east-1;`

Ambiti

L'elenco di ambiti definiti dal client. Una volta autorizzato, questo elenco limita le autorizzazioni quando viene concesso un token di accesso. L'azione dell'RegisterClientAPI IAM Identity Center utilizza questo valore per il `scopes` parametro.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
sso_oidc_scopes	Facoltativo	none	`sso_oidc_scopes=scope1,scope2,scope3;`

ID account

L'identificatore del Account AWS che viene assegnato all'utente. L'GetRoleCredentialsAPI IAM Identity Center utilizza questo valore per il `accountId` parametro.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
sso_oidc_account_id	Richiesto	none	`sso_oidc_account_id=123456789123;`

Nome ruolo

Il nome descrittivo del ruolo assegnato all'utente. Il nome specificato per questo set di autorizzazioni viene visualizzato nel portale di AWS accesso come ruolo disponibile. L'azione GetRoleCredentialsAPI IAM Identity Center utilizza questo valore per il `roleName` parametro.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
sso_oidc_role_name	Richiesto	none	`sso_oidc_role_name=AthenaReadAccess;`

Timeout

Il numero di secondi in cui l'API SSO di polling verifica la presenza del token di accesso.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
sso_oidc_timeout	Facoltativo	`120`	`sso_oidc_timeout=60;`

Abilitazione della cache di file

Abilita una cache delle credenziali temporanee. Questo parametro di connessione consente di memorizzare nella cache le credenziali temporanee e di riutilizzarle tra più processi. Utilizza questa opzione per ridurre il numero di finestre del browser aperte quando utilizzi strumenti di BI come Microsoft Power BI.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
sso_oidc_cache	Facoltativo	1	`sso_oidc_cache=0;`

Credenziali predefinite

Puoi utilizzare le credenziali predefinite che configuri sul tuo sistema client per connetterti ad Amazon Athena. Per informazioni sull'utilizzo delle credenziali predefinite, consulta Utilizzo della catena di provider delle credenziali predefinita nella Guida per gli sviluppatori di AWS SDK for Java.

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Obbligatorio	`IAM Credentials`	`AuthenticationType=Default Credentials;`

Credenziali esterne

Le credenziali esterne sono un plug-in di autenticazione generico che puoi utilizzare per connetterti a qualsiasi provider di identità esterno basato su SAML. Per utilizzare il plugin, devi passare un file eseguibile che restituisce una risposta SAML.

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Obbligatorio	`IAM Credentials`	`AuthenticationType=External Credentials;`

Percorso eseguibile

Il percorso dell'eseguibile con la logica del provider di credenziali personalizzato basato su SAML. L'output dell'eseguibile deve essere la risposta SAML analizzata dal provider di identità.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
ExecutablePath	Obbligatorio	none	ExecutablePath=C:\Users *user_name* *external_ credential.exe*

Elenco degli argomenti

L'elenco degli argomenti che si desidera trasferire all'eseguibile.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
ArgumentList	Facoltativo	none	`ArgumentList=` *arg1 arg2 arg3*

Profilo dell'istanza

Questo tipo di autenticazione viene utilizzato sulle istanze EC2 e viene fornito mediante il servizio di metadati Amazon EC2.

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Obbligatorio	`IAM Credentia ls`	`AuthenticationType =Instance Profile;`

JWT

Il plug-in JWT (JSON Web Token) fornisce un'interfaccia che utilizza JSON Web Tokens per assumere un ruolo IAM di Amazon. La configurazione dipende dal provider di identità. Per informazioni sulla configurazione della federazione per Google Cloud eAWS, consulta Configurazione

della federazione delle identità dei carichi di lavoro con AWS o Azure nella documentazione di Google Cloud.

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Obbligatorio	`IAM Credentials`	`AuthenticationType=JWT;`

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Per ulteriori informazioni sui ruoli ARN, consulta AssumeRole nella Documentazione di riferimento dell'API di AWS Security Token Service

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
preferred_role	Facoltativo	none	`preferred_role=arn:aws:IAM::123456789012:id/user1;`

Durata della sessione

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni sulla durata della sessione, consulta AssumeRole nella Documentazione di riferimento dell'API di AWS Security Token Service.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
durata	Facoltativo	`900`	`duration=900;`

Token web JSON

Il token web JSON utilizzato per recuperare le credenziali temporanee IAM utilizzando l'azione dell'API AWS STS AssumeRoleWithWebIdentity. Per informazioni sulla generazione di token web JSON per gli utenti di Google Cloud Platform (GCP), consulta Utilizzo dei token JWT OAuth nella documentazione di Google Cloud.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
`web_identity_token`	Obbligatorio	`none`	`web_identity_token=eyJhbGc...<remainder of token>;`

Nome della sessione del ruolo

Un nome per la sessione. Una tecnica comune consiste nell'utilizzare il nome o l'identificatore dell'utente dell'applicazione come nome della sessione di ruolo. Ciò associa comodamente le credenziali di sicurezza temporanee utilizzate dall'applicazione a un utente corrispondente.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
role_session_name	Obbligatorio	`none`	`role_session_name=familiarname;`

Okta

Okta è un plug-in di autenticazione basato su SAML che funziona con il gestore dell'identità digitale Okta. Per informazioni sulla configurazione della federazione per Okta e Amazon Athena, consulta Configurazione di SSO per ODBC utilizzando il plug-in Okta e il gestore dell'identità digitale (IdP) Okta.

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Richiesto	`IAM Credentials`	`AuthenticationType=Okta;`

ID utente

Il tuo nome utente Okta.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
UID	Richiesto	none	`UID=jane.doe@org.com;`

Password

La tua password utente Okta.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
PWD	Richiesto	none	`PWD=oktauserpasswordexample;`

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Per ulteriori informazioni sui ruoli ARN, consulta l'AWS Security Token Service API AssumeRoleReference.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
preferred_role	Facoltativo	none	`preferred_role=arn:aws:IAM::123456789012:id/user1;`

Durata della sessione

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni, consulta AssumeRolel'AWS Security Token Service API Reference.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
durata	Facoltativo	`900`	`duration=900;`

Host IdP

L'URL della tua organizzazione Okta. Puoi estrarre il parametro `idp_host` dall'URL del Link di incorporamento nella tua applicazione Okta. Per le fasi, consulta Recupero delle informazioni di configurazione ODBC da Okta. Il primo segmento successivo a `https://`, fino a `okta.com` incluso, è il tuo host IdP (ad esempio, `http://trial-1234567.okta.com`).

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
idp_host	Richiesto	None	`idp_host=dev-99999999.okta.com;`

Porta IdP

Il numero di porta da utilizzare per connettersi all'host IdP.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
idp_port	Richiesto	None	idp_port=443;

ID app Okta

L'identificatore in due parti dell'applicazione. Puoi estrarre il parametro app_id dall'URL del Link di incorporamento nella tua applicazione Okta. Per le fasi, consulta Recupero delle informazioni di configurazione ODBC da Okta. L'ID dell'applicazione è costituito dagli ultimi due segmenti dell'URL, inclusa la barra nel mezzo. I segmenti sono due stringhe di 20 caratteri con un mix di numeri e lettere maiuscole e minuscole (ad esempio Abc1de2fghi3J45kL678/abc1defghij2klmNo3p4).

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
app_id	Richiesto	None	app_id=0oa25kx8ze9A3example/alnexamplea0piaWa0g7;

Nome dell'app Okta

Il nome dell'applicazione Okta.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
Nome_App	Richiesto	None	app_name=amazon_aws_redshift;

Tempo di attesa Okta

Specifica in secondi la durata di attesa del codice di autenticazione a più fattori (MFA).

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
okta_mfa_wait_time	Facoltativo	`10`	`okta_mfa_wait_time=20;`

Tipo MFA Okta

Il tipo di fattore MFA. I tipi supportati sono Google Authenticator, SMS (Okta), Okta Verify with Push e Okta Verify with TOTP. Le policy di sicurezza della singola organizzazione determinano se richiedere o meno l'MFA per l'accesso degli utenti.

Nome stringa connessione	Tipo parametro	Valore predefinito	Valori possibili	Esempio stringa connessione
okta_mfa_type	`Optional`	`None`	`googleauthenticator`, `smsauthentication`, `oktaverifywithpush`, `oktaverifywithtotp`	`okta_mfa_type=oktaverifywithpush;`

Numero di telefono Okta

Il numero di telefono da utilizzare per l' AWS SMS autenticazione. Questo parametro è richiesto solo per la registrazione a più fattori. Se il tuo numero di cellulare è già registrato o se l'autenticazione con AWS SMS non è contemplata dalla policy di sicurezza, ignora pure questo campo.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
okta_mfa_ phone_number	Necessario per la registrazione MFA, altrimenti facoltativo	None	`okta_mfa_phone_num ber=19991234567;`

Abilitazione della cache di file Okta

Abilita una cache temporanea delle credenziali. Questo parametro di connessione consente di memorizzare nella cache le credenziali temporanee e di riutilizzarle tra i diversi processi aperti dalle applicazioni di BI. Utilizza questa opzione per evitare la soglia di limitazione dell'API Okta.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
okta_cache	Facoltativo	`0`	`okta_cache=1;`

Ping

Ping è un plug-in basato su SAML che funziona con il provider di PingFederateidentità.

Tipo di autenticazione

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
AuthenticationType	Richiesto	`IAM Credentials`	`Authentic ationType =Ping;`

ID utente

Il nome utente del PingFederate server.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
UID	Richiesto	none	`UID=pingusername@domain.com;`

Password

La password per il PingFederate server.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
PWD	Richiesto	none	`PWD=pingpassword;`

Ruolo preferito

L'Amazon Resource Name (ARN) del ruolo assegnato al ruolo da assumere. Se l'asserzione SAML prevede più ruoli, puoi specificare questo parametro per scegliere il ruolo da assumere. Questo ruolo deve essere presente nell'asserzione SAML. Per ulteriori informazioni sui ruoli ARN, consulta l'AWS Security Token Service API AssumeRoleReference.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
preferred_role	Facoltativo	none	`preferred_role=arn:aws:iam::123456789012:id/user1;`

Durata della sessione

La durata, in secondi, della sessione dei ruoli. Per ulteriori informazioni sulla durata della sessione, consulta AssumeRolel'AWS Security Token Service API Reference.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
durata	Facoltativo	`900`	`duration=900;`

Host IdP

L'indirizzo del tuo server Ping. Per trovare il tuo indirizzo, visita il seguente URL e visualizza il campo Endpoint applicazione SSO.

```
https://your-pf-host-#:9999/pingfederate/your-pf-app#/spConnections
```

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
idp_host	Richiesto	`none`	`idp_host=ec2-1-83-65-12.compute-1.amazonaws.com;`

Porta IdP

Il numero di porta da utilizzare per connettersi all'host IdP.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
idp_port	Richiesto	`None`	`idp_port=443;`

SPID del partner

L'indirizzo del provider di servizi. Per trovare l'indirizzo del provider di servizi, visita il seguente URL e visualizza il campo Endpoint applicazione SSO.

```
https://your-pf-host-#:9999/pingfederate/your-pf-app#/spConnections
```

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
partner_spid	Richiesto	None	`partner_spid=https://us-east-1.signin.aws.amazon.com/platform/saml/<...>;`

Parametro URI di Ping

Passa un argomento URI per una richiesta di autenticazione a Ping. Utilizza questo parametro per aggirare la limitazione del ruolo singolo di Lake Formation. Configura Ping per riconoscere il parametro inviato e verificare che il ruolo inviato esista nell'elenco dei ruoli assegnati all'utente. Quindi, invia un singolo ruolo nell'asserzione SAML.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
ping_uri_param	Facoltativo	None	`ping_uri_param=role=my_iam_role;`

Parametri comuni di autenticazione

In questa sezione i parametri sono comuni con i tipi di autenticazione, come indicato.

Utilizzo di Proxy per IdP

Abilita la comunicazione tra il driver e l'IdP tramite il proxy. Questa opzione è disponibile per i seguenti plugin di autenticazione:

- AD FS
- Azure AD
- Browser Azure AD
- Browser SSO OIDC
- JWT

- Okta
- Ping

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
UseProxyForIdP	Facoltativo	`0`	`UseProxyForIdP=1;`

Utilizzo di Lake Formation

Utilizza l'azione API AssumeDecoratedRoleWithSAML di Lake Formation per recuperare credenziali IAM temporanee anziché l'azione API AssumeRoleWithSAML di AWS STS. Questa opzione è disponibile per i plugin di autenticazione Azure AD, Browser Azure AD, Browser SAML, Okta, Ping e AD FS.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
LakeformationEnabled	Facoltativo	`0`	`LakeformationEnabled=1;`

SSL non sicuro (IdP)

Disattiva SSL durante la comunicazione con l'IdP. Questa opzione è disponibile per i plugin di autenticazione Azure AD, Browser Azure AD, Okta, Ping e AD FS

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
ssl_insecure	Facoltativo	`0`	`SSL_Insecure=1;`

Sostituzioni degli endpoint

Sovrascrizione degli endpoint Athena

La classe `endpointOverride ClientConfiguration` utilizza questo valore per sostituire l'endpoint HTTP predefinito per il client Amazon Athena. Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++ .

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
EndpointOverride	Facoltativo	none	`EndpointOverride=athena.us-west-2.amazonaws.com;`

Sostituzione degli endpoint di streaming Athena

Il metodo `ClientConfiguration.endpointOverride` utilizza questo valore per sovrascrivere l'endpoint HTTP predefinito per il client di streaming Amazon Athena. Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++ . Il servizio Athena Streaming è disponibile tramite la porta 444.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
Streaming EndpointOverride	Facoltativo	none	`StreamingEndpointOverride=athena.us-west-1.amazonaws.com:444;`

AWS STS sovrascrittura dell'endpoint

Il `ClientConfiguration.endpointOverride` metodo utilizza questo valore per sovrascrivere l'endpoint HTTP predefinito per il client. AWS STS Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++ .

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
StsEndpointOverride	Facoltativo	none	`StsEndpointOverride=sts.us-west-1.amazonaws.com;`

Sovrascittura endpoint di Lake Formation

Il metodo `ClientConfiguration.endpointOverride` utilizza questo valore per sovrascrivere l'endpoint HTTP predefinito per il client Lake Formation. Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++ .

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
LakeFormationEndpointOverride	Facoltativo	none	`LakeFormationEndpointOverride=lakeformation.us-west-1.amazonaws.com;`

Sovrascrittura degli endpoint SSO

Il metodo `ClientConfiguration.endpointOverride` utilizza questo valore per sovrascrivere l'endpoint HTTP predefinito per il client SSO. Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++ .

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
SSO EndpointOverride	Facoltativo	none	`SSOEndpointOverride=portal.sso.us-east-2.amazonaws.com;`

Sovrascrittura degli endpoint OIDC SSO

Il metodo `ClientConfiguration.endpointOverride` utilizza questo valore per sovrascrivere l'endpoint HTTP predefinito per il client OIDC SSO. Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++ .

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
SSOIDICO EndpointOverride	Facoltativo	`none`	`SSOOIDCEndpointOverride=oidc.us-east-2.amazonaws.com`

Opzioni avanzate

Dimensioni di recupero

Numero massimo di risultati (righe) da restituire in questa richiesta. Per informazioni sui parametri, vedere GetQuery MaxResults. Per l'API di streaming, il valore massimo è pari a 10000000.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
RowsToFetchPerBlock	Facoltativo	`1000` per il non streaming `20000` per lo streaming	`RowsToFetchPerBlock=20000;`

Come abilitare il riutilizzo dei risultati

Specifica se i risultati della query precedente possono essere riutilizzati quando la query viene eseguita. Per informazioni sui parametri, vedere ResultReuseByAgeConfiguration.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
EnableResultReuse	Facoltativo	`0`	`EnableResultReuse=1;`

Età massima per il riutilizzo di risultati

Specifica, in minuti, l'età massima dei risultati di una query precedente che Athena debba considerare per il riutilizzo. Per informazioni sui parametri, vedere ResultReuseByAgeConfiguration.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
ReusedResultMaxAgeInMinutes	Facoltativo	`60`	`ReusedResultMaxAgeInMinutes=90;`

Come abilitare l'API di streaming

Sceglie se utilizzare l'API di streaming Athena per recuperare il set di risultati.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
UseResultsetStreaming	Facoltativo	`0`	`UseResultsetStreaming=1;`

Come abilitare il fetcher S3

Recupera il set di risultati generato da Athena dal bucket Amazon S3 interagendo direttamente con Amazon S3.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
EnableS3Fetcher	Facoltativo	1	`EnableS3Fetcher=1;`

Utilizzo di più thread S3

Recupera i dati da Amazon S3 utilizzando più thread. Quando questa opzione è abilitata, il file di risultati archiviato nel bucket Amazon S3 viene recuperato in parallelo utilizzando più thread.

Abilita questa opzione solo se disponi di una buona larghezza di banda della rete. Ad esempio, nelle nostre misurazioni su un'istanza EC2 c5.2xlarge, un client S3 a thread singolo ha raggiunto 1 Gbps, mentre i client S3 a thread multiplo hanno raggiunto 4 Gbps di velocità di trasmissione effettiva di rete.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
UseMultipleThread S3	Facoltativo	`0`	`UseMultipleS3Threads=1;`

Utilizzo di schema e catalogo singolo

Per impostazione predefinita, il driver ODBC esegue query su Athena per ottenere l'elenco dei cataloghi e degli schemi disponibili. Questa opzione prevede che il driver utilizzi il catalogo e lo schema specificati dalla finestra di dialogo di configurazione di ODBC Data Source Administrator o dai parametri di connessione.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
UseSingleCatalogAndSchema	Facoltativo	`0`	`UseSingleCatalogAndSchema=1;`

Usa la query per elencare le tabelle

Per i tipi di `LAMBDA` catalogo, consente al driver ODBC di inviare una SHOW TABLES query per ottenere un elenco di tabelle disponibili. Questa è l'impostazione di default. Se questo parametro è impostato su 0, il driver ODBC utilizza l'API ListTableMetadataAthena per ottenere un elenco di tabelle disponibili. Tieni presente che, per i tipi di `LAMBDA` catalogo, l'utilizzo `ListTableMetadata` porta alla regressione delle prestazioni.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
UseQueryToListTables	Facoltativo	`1`	`UseQueryToListTables=1;`

Usa WCHAR per i tipi di stringhe

Per impostazione predefinita, il driver ODBC utilizza `SQL_CHAR` e `SQL_VARCHAR` per Athena i `char` tipi di dati string`varchar`,,`string`, `array` `map<>struct<>`, e. `row` L'impostazione di questo parametro per `1` forzare l'utilizzo da parte del driver `SQL_WCHAR` e `SQL_WVARCHAR` per i tipi di dati a stringa. I tipi di caratteri wide e wide variable vengono utilizzati per garantire che i caratteri di diverse lingue possano essere memorizzati e recuperati correttamente.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
Usa W CharForStringTypes	Facoltativo	`0`	`UseWCharForStringTypes=1;`

Esecuzione di query su cataloghi esterni

Specifica se il driver deve eseguire query sui cataloghi esterni ad Athena. Per ulteriori informazioni, consulta Come migrare al driver ODBC 2.x.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
QueryExternalCatalogs	Facoltativo	`0`	`QueryExternalCatalogs=1;`

Verifica di SSL

Controlla se verificare i certificati SSL quando utilizzi l' AWS SDK. Questo valore viene passato al parametro `ClientConfiguration.verifySSL`. Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++ .

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
VerifySSL	Facoltativo	`1`	`VerifySSL=0;`

Dimensione del blocco dei risultati S3

Speciifica, in byte, la dimensione del blocco da scaricare per una singola richiesta API Amazon GetObjectS3. Il valore predefinito è pari a 67108864 (64 MB). I valori minimo e massimo consentiti sono 10485760 (10 MB) e 2146435072 (circa 2 GB).

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
S3 ResultBlockSize	Facoltativo	`67108864`	`S3ResultBlockSize=268435456;`

Lunghezza della colonna di stringhe

Speciifica la lunghezza delle colonne con il tipo di `string` dati. Poiché Athena utilizza il tipo di dati stringa Apache Hive, che non ha una precisione definita, la lunghezza predefinita riportata da Athena è 2147483647 (). `INT_MAX` Poiché gli strumenti di BI di solito preallocano la memoria per

le colonne, ciò può comportare un elevato consumo di memoria. Per evitare ciò, il driver ODBC Athena limita la precisione riportata per le colonne del tipo di `string` dati ed espone il parametro di `StringColumnLength` connessione in modo che il valore predefinito possa essere modificato.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
StringColumnLength	Facoltativo	255	`StringColumnLength =65535;`

Lunghezza delle colonne di tipo complesso

Speciifica la lunghezza delle colonne con tipi di dati complessi come `mapstruct`, `earray`. Ad esempio StringColumnLength, Athena riporta una precisione pari a 0 per le colonne con tipi di dati complessi. Il driver ODBC Athena imposta la precisione predefinita per le colonne con tipi di dati complessi ed espone il parametro di `ComplexTypeColumnLength` connessione in modo che il valore predefinito possa essere modificato.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
ComplexTy peColumnLength	Facoltativo	65535	`ComplexTypeColumnL ength=123456;`

Certificato CA attendibile

Indica al client HTTP dove trovare l'archivio attendibile con i certificati SSL. Questo valore viene passato al parametro `ClientConfiguration.caFile`. Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++ .

Nome stringa connessione	Tipo parametro	Valore predefini to	Esempio stringa connessione
TrustedCerts	Facoltativo	`%INSTALL_ PATH%/bin`	`TrustedCerts=C:\\Program Files\ \Amazon Athena ODBC Driver\\bin \\cacert.pem;`

Periodo minimo del polling

Specifica in millisecondi il valore minimo di attesa prima di eseguire il polling di Athena per verificare lo stato di esecuzione della query.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
MinQueryExecutionPollingInterval	Facoltativo	`100`	`MinQueryExecutionPollingInterval=200;`

Periodo massimo del polling

Specifica in millisecondi il valore massimo di attesa prima di eseguire il polling di Athena per verificare lo stato di esecuzione della query.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
MaxQueryExecutionPollingInterval	Facoltativo	`60000`	`MaxQueryExecutionPollingInterval=1000;`

Moltiplicatore polling

Specifica il fattore di incremento del periodo del polling. Per impostazione predefinita, il polling inizia con il valore del periodo minimo del polling e raddoppia con ogni polling fino a raggiungere il valore del periodo massimo di polling.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
QueryExecutionPollingIntervalMultiplier	Facoltativo	2	`QueryExecutionPollingIntervalMultiplier=2;`

Durata massima del polling

Specifica il valore massimo in millisecondi entro cui un driver può eseguire il polling su Athena per verificare lo stato di esecuzione delle query.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
MaxPollDuration	Facoltativo	`1800000`	`MaxPollDuration=1800000;`

Timeout di connessione

La quantità di tempo (in millisecondi) di attesa prima che venga stabilita una connessione HTTP. Questo valore è impostato per il client Athena `ClientConfiguration.connectTimeoutMs`. Se non è specificato, viene utilizzato il valore predefinito curl. Per informazioni sui parametri di connessione, consulta Configurazione client nella Guida per gli sviluppatori di AWS SDK for Java .

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
ConnectionTimeout	Facoltativo	`0`	`ConnectionTimeout=2000;`

Timeout richiesta

Specifica il timeout di lettura dei socket per i client HTTP. Questo valore è impostato per il parametro `ClientConfiguration.requestTimeoutMs` del client Athena. Per informazioni, consulta Configurazione client nella Guida per gli sviluppatori di AWS SDK for Java .

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
RequestTimeout	Facoltativo	`10000`	`RequestTimeout=30000;`

Opzioni proxy

Host proxy

Se vuoi che gli utenti passino attraverso un proxy, utilizza questo parametro per impostare l'host del proxy. Questo parametro corrisponde al parametro `ClientConfiguration.proxyHost` nell'SDK di AWS. Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
ProxyHost	Facoltativo	none	`ProxyHost =127.0.0.1;`

Porta proxy

Utilizza questo parametro per impostare la porta del proxy. Questo parametro corrisponde al parametro `ClientConfiguration.proxyPort` nell'SDK di AWS. Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
ProxyPort	Facoltativo	none	`ProxyPort=8888;`

Nome utente proxy

Utilizza questo parametro per impostare il nome utente del proxy. Questo parametro corrisponde al parametro `ClientConfiguration.proxyUserName` nell'SDK di AWS. Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
ProxyUID	Facoltativo	none	`ProxyUID= username;`

Password proxy

Utilizza questo parametro per impostare la password del proxy. Questo parametro corrisponde al parametro `ClientConfiguration.proxyPassword` nell'SDK di AWS. Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
ProxyPWD	Facoltativo	`none`	`ProxyPWD=password;`

Host non proxy

Utilizza questo parametro opzionale per specificare un host a cui il driver si connette senza utilizzare un proxy. Questo parametro corrisponde al parametro `ClientConfiguration.nonProxyHosts` nell'SDK di AWS. Per ulteriori informazioni, consulta Configurazione client AWS nella Guida per gli sviluppatori di AWS SDK for C++.

Il parametro di connessione `NonProxyHost` viene passato all'opzione curl `CURLOPT_NOPROXY`. Per informazioni sul formato `CURLOPT_NOPROXY`, consulta CURLOPT_NOPROXY nella documentazione di curl.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
NonProxyHost	Facoltativo	`none`	`NonProxyHost=.amazonaws.com,localhost,.example.net,.example.com;`

Come utilizzare il proxy

Abilita il traffico utente attraverso il proxy specificato.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
UseProxy	Facoltativo	none	`UseProxy=1;`

Opzioni di registrazione

I diritti di amministratore sono necessari per modificare le impostazioni qui descritte. Per apportare le modifiche, è possibile utilizzare la finestra di dialogo ODBC Data Source Administrator Opzioni di log o modificare direttamente il registro di Windows.

Livello di log

Questa opzione abilita i registri dei driver ODBC. In Windows, è possibile utilizzare il registro o una finestra di dialogo per abilitare o disabilitare la registrazione. L'opzione si trova nel seguente percorso del registro:

```
Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Amazon Athena\ODBC\Driver
```

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
LogLevel	Facoltativo	`0`	`LogLevel=1;`

Log Path (Percorso log)

Specifica il percorso del file in cui sono archiviati i log dei driver ODBC. È possibile utilizzare il registro o una finestra di dialogo per impostare questo valore. L'opzione si trova nel seguente percorso del registro:

```
Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Amazon Athena\ODBC\Driver
```

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
LogPath	Facoltativo	none	`LogPath=C:\Users\ username\projects \internal\trunk\;`

Utilizzo di AWS Logger

Specifica se la registrazione SDK AWS è abilitata. Specificare 1 per abilitare, 0 per disabilitare.

Nome stringa connessione	Tipo parametro	Valore predefinito	Esempio stringa connessione
Usa AWS Logger	Facoltativo	`0`	`UseAwsLogger=1;`

Come migrare al driver ODBC 2.x

Poiché la maggior parte dei parametri di connessione ODBC 2.x di Athena sono retrocompatibili con il driver ODBC 1.x, con il driver Athena ODBC 2.x puoi riutilizzare la maggior parte della stringa di connessione esistente. Tuttavia, i seguenti parametri di connessione richiedono delle modifiche.

Livello di log

Sebbene il driver ODBC attuale offra una gamma di opzioni di log disponibili, a partire da `LOG_OFF (0)` a `LOG_TRACE (6)`, il driver ODBC di Amazon Athena presenta solo due valori: 0 (disabilitato) e 1 (abilitato).

Per ulteriori informazioni sulla registrazione di log con il driver ODBC 2.x, consulta Opzioni di registrazione.

	Driver ODBC 1.x	Driver ODBC 2.x
Nome stringa connessione	`LogLevel`	`LogLevel`
Tipo parametro	Facoltativo	Facoltativo

	Driver ODBC 1.x	Driver ODBC 2.x
Valore predefinito	`0`	`0`
Valori possibili	`0-6`	`0,1`
Esempio stringa connessione	`LogLevel=6;`	`LogLevel=1;`

MetadataRetrievalMethod

L'attuale driver ODBC offre varie opzioni per recuperare i metadati da Athena. Il driver ODBC di Amazon Athena rende `MetadataRetrievalMethod` obsoleto e utilizza sempre l'API di Amazon Athena per estrarre i metadati.

Athena introduce il flag `QueryExternalCatalogs` per eseguire query su cataloghi esterni. Per eseguire query sui cataloghi esterni con l'attuale driver ODBC, imposta `MetadataRetrievalMethod` su `ProxyAPI`. Per eseguire query sui cataloghi esterni con l'attuale driver ODBC di Athena, imposta `QueryExternalCatalogs` su `1`.

	Driver ODBC 1.x	Driver ODBC 2.x
Nome stringa connessione	`MetadataRetrievalMethod`	`QueryExternalCatalogs`
Tipo parametro	Facoltativo	Facoltativo
Valore predefinito	`Auto`	`0`
Valori possibili	`Auto`, `AWS Glue`, `ProxyAPI`, `Query`	`0`,`1`
Esempio stringa connessione	`MetadataRetrievalMethod=ProxyAPI;`	`QueryExternalCatalogs=1;`

Test di connessione

Quando esegui il test di connessione di un driver ODBC 1.x, il driver esegue una query `SELECT 1` che genera due file nel bucket Amazon S3: uno per il set di risultati e uno per i metadati. La connessione del test viene addebitata in base alla policy dei Prezzi di Amazon Athena.

Quando esegui il test di connessione di un driver ODBC 2.x, il driver richiama l'operazione API GetWorkGroup di Athena. La chiamata utilizza il tipo di autenticazione e il provider di credenziali corrispondente che hai indicato per recuperare le credenziali. Non è previsto alcun costo per il test di connessione quando utilizzi il driver ODBC 2.x, inoltre il test non genera risultati di query nel bucket Amazon S3.

Risoluzione dei problemi del driver ODBC 2.x

Se riscontri problemi con il driver ODBC di Amazon Athena, puoi metterti in contatto con AWS Support (nella AWS Management Console seleziona Assistenza,Centro assistenza).

Assicurati di includere le seguenti informazioni e di fornire eventuali dettagli aggiuntivi che aiutino il team di assistenza a comprendere il tuo caso d'uso.

- Descrizione: (obbligatorio) una descrizione che includa informazioni dettagliate sul caso d'uso e sulla differenza tra il comportamento previsto e quello osservato. Includi tutte le informazioni che possano aiutare i tecnici dell'assistenza a risolvere facilmente il problema. Se il problema è intermittente, specifica le date, i timestamp o gli intervalli in cui si è presentato il problema.
- Informazioni sulla versione: (obbligatorio) informazioni sulla versione del driver, sul sistema operativo e sulle applicazioni utilizzate. Ad esempio, "Driver ODBC versione 1.2.3, Windows 10 (x64), Power BI".
- File di log: (obbligatorio) il numero minimo di file di log del driver ODBC necessari per comprendere il problema. Per ulteriori informazioni sulle opzioni di registrazione di log con il driver ODBC 2.x, consulta Opzioni di registrazione.
- Stringa di connessione: (obbligatorio) la stringa di connessione ODBC o una schermata della finestra di dialogo che mostra i parametri di connessione utilizzati. Per informazioni sui parametri di connessione, consulta Parametri di connessione Athena ODBC 2.x.
- Fasi del problema: (facoltativo) se possibile, includi le fasi o un programma indipendente che possa aiutare a riprodurre il problema.
- Informazioni sugli errori delle query: (facoltativo) se sono presenti errori relativi alle query DML o DDL, includi le seguenti informazioni:
  - una versione completa o semplificata della query DML o DDL che non è andata a buon fine.

  - l'ID dell'account e la Regione AWS in uso, nonché l'ID di esecuzione della query.
- Errori SAML: (facoltativo) se riscontri un problema relativo all'autenticazione con asserzione SAML, includi le seguenti informazioni:
  - il provider di identità e il plug-in di autenticazione utilizzato.
  - Un esempio con il token SAML.

Note di rilascio per ODBC 2.x di Amazon Athena

Queste note di rilascio forniscono dettagli su miglioramenti, funzionalità, problemi noti e modifiche al flusso di lavoro nel driver ODBC 2.x di Amazon Athena.

2,03,0

Rilasciato il 2024-04-08

Il driver Amazon Athena ODBC v2.0.3.0 contiene i seguenti miglioramenti e correzioni.

Miglioramenti

- Aggiunto il supporto MFA per il plug-in di autenticazione Okta su piattaforme Linux e Mac.
- Sia la `athena-odbc.dll` libreria che il programma di `AmazonAthenaODBC-2.x.x.x.msi` installazione per Windows sono ora firmati.
- È stato aggiornato il `cacert.pem` file del certificato CA installato con il driver.
- È stato migliorato il tempo necessario per elencare le tabelle nei cataloghi Lambda. Per i tipi di `LAMBDA` catalogo, il driver ODBC può ora inviare una SHOW TABLES query per ottenere un elenco di tabelle disponibili. Per ulteriori informazioni, consulta Usa la query per elencare le tabelle.
- È stato introdotto il parametro di `UseWCharForStringTypes` connessione per riportare i tipi di dati di stringa utilizzando `SQL_WCHAR` and`SQL_WVARCHAR`. Per ulteriori informazioni, consulta Usa WCHAR per i tipi di stringhe.

Correzioni

- È stato corretto un avviso di danneggiamento del registro che si verificava quando veniva utilizzato OdbcDsn PowerShell lo strumento Get-.
- È stata aggiornata la logica di analisi per gestire i commenti all'inizio delle stringhe di query.
- I tipi di dati relativi a data e ora ora consentono lo zero nel campo dell'anno.

Per scaricare il nuovo driver ODBC v2, consulta Come scaricare il driver ODBC 2.x. Per informazioni sulla connessione, consulta Amazon Athena ODBC 2.x.

2.0.2.2

Rilasciato il 2024-02-13

Il driver Amazon Athena ODBC v2.0.2.2 contiene i seguenti miglioramenti e correzioni.

Miglioramenti

- Sono stati aggiunti due parametri di connessione, `StringColumnLength` e`ComplexTypeColumnLength`, che puoi utilizzare per modificare la lunghezza predefinita delle colonne per stringhe e tipi di dati complessi. Per ulteriori informazioni, consulta Lunghezza della colonna di stringhe e Lunghezza delle colonne di tipo complesso.
- È stato aggiunto il supporto per i sistemi operativi Linux e macOS (Intel e ARM). Per ulteriori informazioni, consulta Linux e macOS.
- AWS-SDK-CPP è stato aggiornato alla versione del tag 1.11.245.
- La libreria curl è stata aggiornata alla versione 8.6.0.

Correzioni

- È stato risolto un problema che causava la segnalazione di valori errati nei metadati del set di risultati per tipi di dati simili a stringhe nella colonna precision.

Per scaricare il nuovo driver ODBC v2, consulta la pagina Come scaricare il driver ODBC 2.x. Per informazioni sulla connessione, consulta Amazon Athena ODBC 2.x.

2.0.2.1

Data di rilascio: 07/12/2023

Il driver ODBC v2.0.2.1 di Amazon Athena contiene i miglioramenti e le correzioni seguenti.

Miglioramenti

- Migliore sicurezza dei thread dei driver ODBC per tutte le interfacce.
- Quando la registrazione è abilitata, ora i valori datetime vengono registrati con una precisione di millisecondi.

- Durante l'autenticazione con il plug-in Browser SSO OIDC, ora il terminale si apre per mostrare all'utente il codice del dispositivo.

Correzioni

- È stato risolto un problema di rilascio della memoria che si verificava durante l'analisi dei risultati dall'API di streaming.
- Le richieste per le interfacce `SQLTablePrivileges()`, `SQLSpecialColumns()`, `SQLProcedureColumns()` e `SQLProcedures()` ora restituiscono set di risultati vuoti.

Per scaricare il nuovo driver ODBC v2, consulta la pagina Come scaricare il driver ODBC 2.x. Per informazioni sulla connessione, consulta Amazon Athena ODBC 2.x.

2,02,0

Data di rilascio: 17/10/2023

Il driver ODBC v2.0.2.0 di Amazon Athena contiene i seguenti miglioramenti e correzioni.

Miglioramenti

- Funzionalità della cache di file aggiunta per i plug-in di autenticazione basati su browser di Browser Azure AD, Browser SSO OIDC e Okta.

  Gli strumenti di BI come Power BI e i plug-in basati su browser utilizzano più finestre del browser. Il nuovo parametro di connessione alla cache dei file consente di memorizzare nella cache le credenziali temporanee e di riutilizzarle tra i molteplici processi aperti dalle applicazioni di BI.
- Le applicazioni possono ora interrogare le informazioni sul set di risultati dopo la preparazione di un'istruzione.
- I timeout predefiniti di connessione e richiesta sono stati aumentati in modo da poter essere utilizzati con reti client più lente. Per ulteriori informazioni, consulta Timeout di connessione e Timeout richiesta.
- Sono state aggiunte le sovrascritture degli endpoint per SSO e SSO OIDC. Per ulteriori informazioni, consulta Sostituzioni degli endpoint.
- È stato aggiunto un parametro di connessione per passare un argomento URI per una richiesta di autenticazione a Ping. Puoi usare questo parametro per aggirare la limitazione del ruolo singolo di Lake Formation. Per ulteriori informazioni, consulta Parametro URI di Ping.

Correzioni

- È stato risolto un problema di overflow di numeri interi che si verificava quando si utilizzava il meccanismo di associazione basato sulle righe.
- Timeout rimosso dall'elenco dei parametri di connessione richiesti per il plug-in di autenticazione Browser SSO OIDC.
- Sono state aggiunte le interfacce mancanti per `SQLStatistics()`, `SQLPrimaryKeys()`, `SQLForeignKeys()` e `SQLColumnPrivileges()` ed è stata aggiunta la possibilità di restituire set di risultati vuoti su richiesta.

Per scaricare il nuovo driver ODBC v2, consulta Come scaricare il driver ODBC 2.x. Per informazioni sulla connessione, consulta Amazon Athena ODBC 2.x.

2,01.1

Data di rilascio: 10/08/2023

Il driver ODBC v2.0.1.1 di Amazon Athena contiene i seguenti miglioramenti e correzioni.

Miglioramenti

- È stata aggiunta la registrazione di log URI al plug-in di autenticazione Okta.
- È stato aggiunto il parametro di ruolo preferito al plug-in del provider di credenziali esterne.
- Aggiunta in corso della gestione del prefisso del profilo nel nome del profilo del file di configurazione AWS .

Correzioni

- È stato corretto un problema di Regione AWS utilizzo che si verificava durante l'utilizzo con Lake Formation e AWS STS i client.
- Le chiavi di partizione mancanti sono state ripristinate nell'elenco delle colonne della tabella.
- È stato aggiunto il tipo di autenticazione `BrowserSSOOIDC` mancante al profilo AWS .

Per scaricare il nuovo driver ODBC v2, consulta Come scaricare il driver ODBC 2.x.

2.0.1.0

Data di rilascio: 29/06/2023

Amazon Athena rilascia il driver ODBC v2.0.1.0.

Athena ha rilasciato un nuovo driver ODBC che migliora l'esperienza di connessione, esecuzione di quey e visualizzazione dei dati da applicazioni di sviluppo SQL compatibili e di business intelligence. L'ultima versione del driver Athena ODBC supporta le funzionalità del driver esistente ed è semplice da aggiornare. La nuova versione include il supporto per l'autenticazione degli utenti tramite AWS IAM Identity Center. Offre anche la possibilità di leggere i risultati delle query da Amazon S3, in modo da renderli disponibili prima.

Per ulteriori informazioni, consulta Amazon Athena ODBC 2.x.

Driver ODBC 1.x di Athena

Usa i link in questa pagina per scaricare il contratto di licenza del driver ODBC 1.x di Amazon Athena, i driver ODBC e la documentazione ODBC. Per informazioni sulla stringa di connessione ODBC, consulta il file PDF della guida all'installazione e alla configurazione del driver ODBC, scaricabile da questa pagina. Per informazioni sulle autorizzazioni, consulta Accesso tramite connessioni JDBC e ODBC.

Important

Quando utilizzi il driver ODBC 1.x, assicurati di valutare i seguenti requisiti:

- Open port 444 (apri la porta 444): mantieni la porta 444, che Athena utilizza per trasmettere i risultati delle query, aperta al traffico in uscita. Quando usi un PrivateLink endpoint per connetterti ad Athena, assicurati che il gruppo di sicurezza collegato all'endpoint sia aperto PrivateLink al traffico in entrata sulla porta 444.
- athena: GetQueryResultsStream policy — Aggiungi l'azione `athena:GetQueryResultsStream` politica ai principi IAM che utilizzano il driver ODBC. Questa operazione di policy non è esposta direttamente con l'API. Viene utilizzata solo con i driver ODBC e JDBC come parte del supporto per i risultati di streaming. Per un esempio di policy, consulta AWS politica gestita: AWSQuicksightAthenaAccess.

Windows

Versione driver	Collegamento per il download
ODBC 1.2.3.1000 per Windows a 32 bit	
ODBC 1.2.3.1000 per Windows a 64 bit	

Linux

Versione driver	Collegamento per il download
ODBC 1.2.3.1000 per Linux a 32 bit	
ODBC 1.2.3.1000 per Linux a 64 bit	

OSX

Versione driver	Collegamento per il download
ODBC 1.2.3.1000 per OSX	

Documentazione

Contenuti	Collegamento alla documentazione
Contratto di licenza del driver ODBC di Amazon Athena	Contratto di licenza
Documentazione per ODBC 1.2.3.1000	
Note di rilascio per ODBC 1.2.3.1000	

Note del driver ODBC

Connessione senza utilizzo di un proxy

Se si desidera specificare alcuni host a cui il driver si connette senza utilizzare un proxy, è possibile utilizzare la prorietà `NonProxyHost` opzionale nella stringa di connessione ODBC.

La proprietà `NonProxyHost` specifica un elenco separato da virgole di host a cui il connettore può accedere senza passare attraverso il server proxy quando è abilitata una connessione proxy, come nell'esempio seguente:

```
.amazonaws.com,localhost,.example.net,.example.com
```

Il parametro di connessione `NonProxyHost` viene passato all'opzione curl `CURLOPT_NOPROXY`. Per informazioni sul formato `CURLOPT_NOPROXY`, consulta CURLOPT_NOPROXY nella documentazione di curl.

Configurazione dell'accesso federato ad Amazon Athena per utenti Microsoft AD FS utilizzando un client ODBC

Per configurare l'accesso federato ad Amazon Athena per gli utenti di Microsoft Active Directory Federation Services (AD FS) utilizzando un client ODBC, devi innanzitutto stabilire una relazione di fiducia tra AD FS e il tuo account AWS. Grazie a questa relazione di fiducia, gli utenti AD possono federarsi a AWS utilizzando le proprie credenziali AD e assumere le autorizzazioni di un ruolo AWS Identity and Access Management (IAM) per accedere a risorse AWS come l'API Athena.

Per creare questa relazione di fiducia, aggiungi AD FS come provider SAML al tuo Account AWS e crei un ruolo IAM che gli utenti federati possono assumere. Per quanto riguarda AD FS, aggiungi AWS come relying party e scrivi le regole di attestazione SAML per inviare gli attributi utente corretti a AWS per l'autorizzazione (in particolare, Athena e Amazon S3).

La configurazione dell'accesso di AD FS ad Athena prevede i seguenti passaggi principali:

1. Configurazione di un provider e di un ruolo IAM SAML

2. Configurazione di AD FS

3. Creazione di utenti e gruppi Active Directory

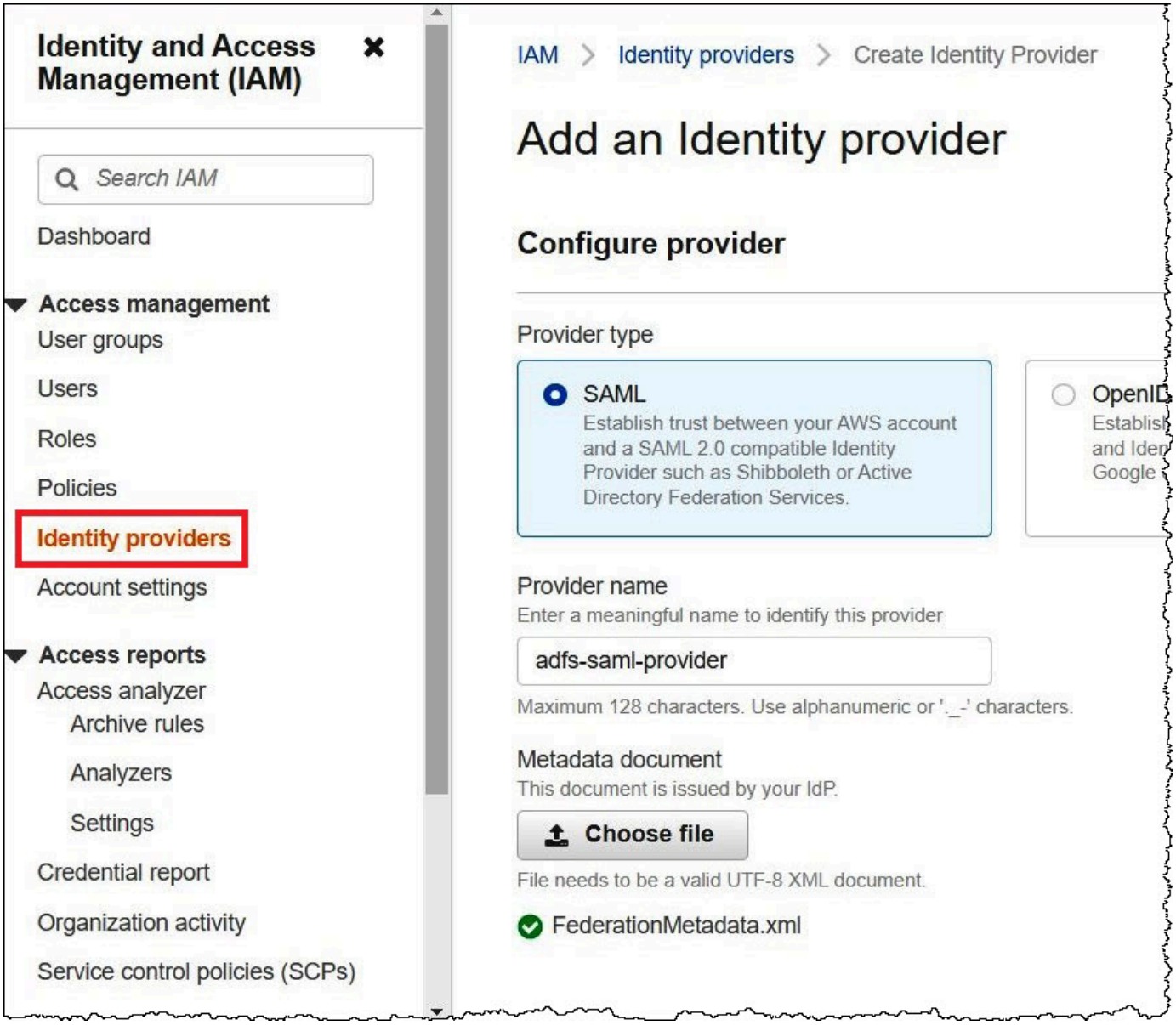
4. Configurazione della connessione ODBC di AD FS ad Athena

1. Configurazione di un provider e di un ruolo IAM SAML

In questa sezione, aggiungi AD FS come provider SAML al tuo account AWS e crei un ruolo IAM che i tuoi utenti federati possono assumere.

Configurazione di un provider SAML

1. Accedi alla AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione, scegli Identity providers (Provider di identità).
3. Scegli Aggiungi provider.
4. Per Tipo di provider, scegliere SAML.



5. Per Nome provider, inserire **`adfs-saml-provider`**.
6. In un browser, inserisci il seguente indirizzo per scaricare il file XML di federazione per il server AD FS. Per eseguire questo passaggio, il browser deve disporre dell'accesso al server AD FS.

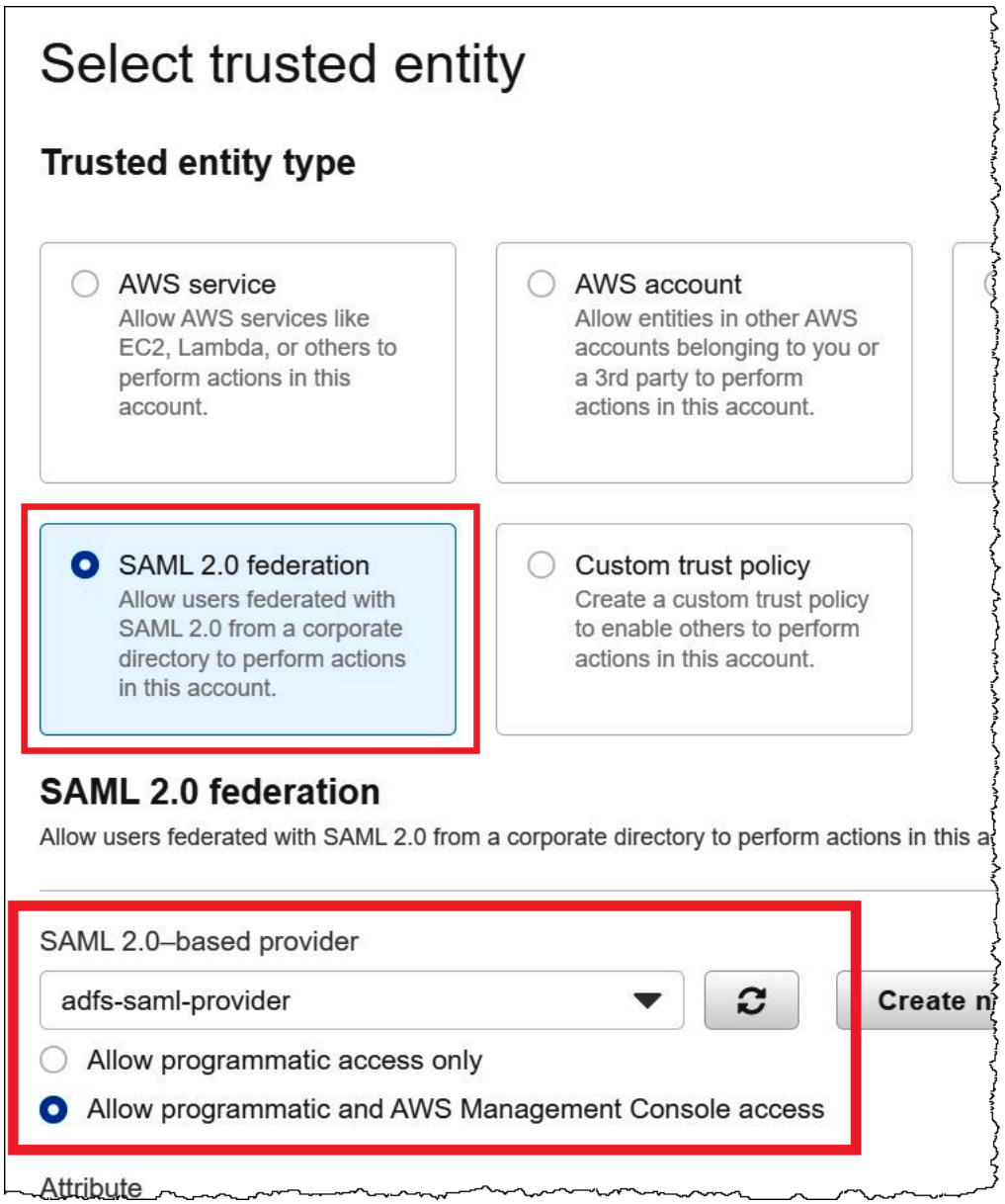
   ```
   https://adfs-server-name/federationmetadata/2007-06/federationmetadata.xml
   ```

7. Nella console IAM, in Metadata document (Documento di metadati), seleziona Choose file (Scegli file), quindi carica il file dei metadati di federazione su AWS.
8. Per concludere, scegli Add provider (Aggiungi provider).

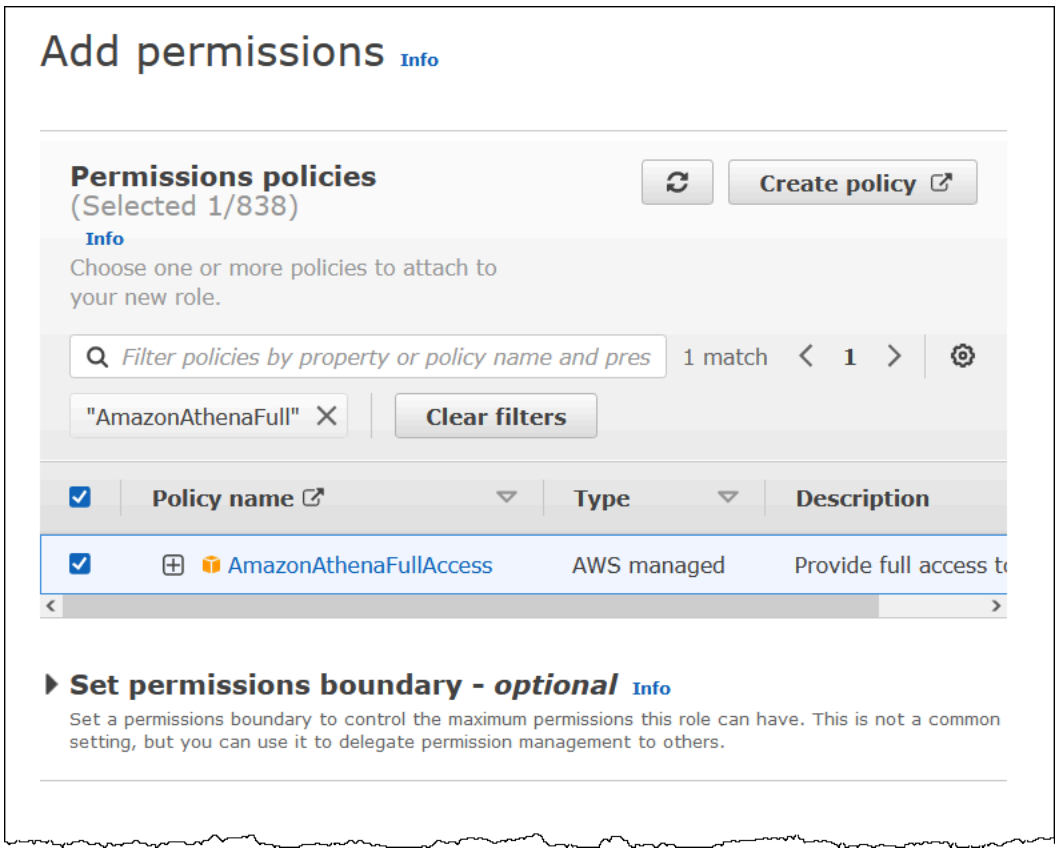
Successivamente, crei il ruolo IAM che può essere assunto dai tuoi utenti federati.

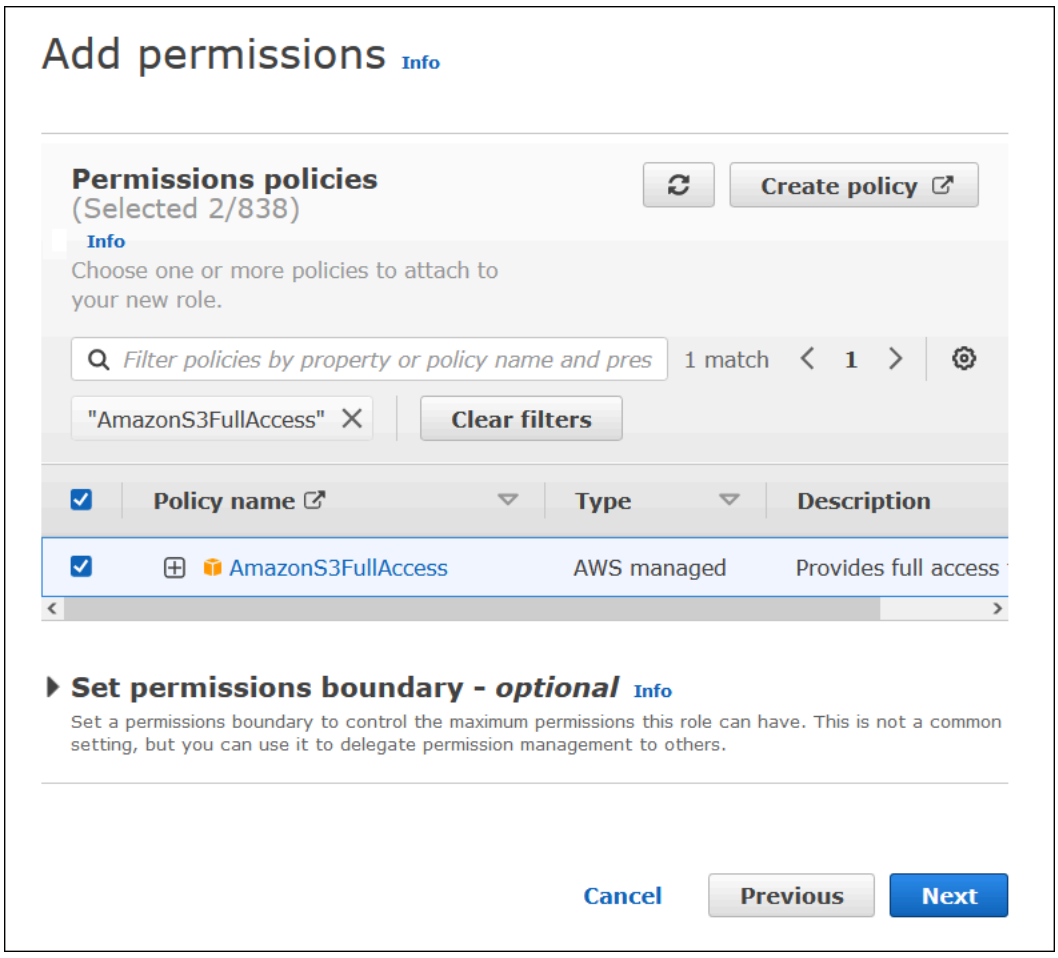
Creazione di un ruolo IAM per utenti federati

1. Nel pannello di navigazione della console IAM, scegli Roles (Ruoli).
2. Scegliere Crea ruolo.
3. In Trusted entity type (Tipo di entità attendibile), scegli SAML 2.0 federation (Federazione SAML 2.0).
4. In SAML 2.0-based provider (Provider basato su SAML 2.0), scegli il provider adfs-saml-provider che hai creato.
5. Seleziona Consenti accesso programmatico e AWS alla Console di gestione, quindi seleziona Successivo.



6. Nella pagina Add permissions (Aggiungi autorizzazioni), filtra le policy di autorizzazione IAM necessarie per questo ruolo, quindi seleziona le caselle di controllo corrispondenti. Questo tutorial mostra come collegare le policy `AmazonAthenaFullAccess` e `AmazonS3FullAccess`.


Add permissions Info
Permissions policies
(Selected 1/838)
Info
Choose one or more policies to attach to your new role.
Create policy
Filter policies by property or policy name and pres
1 match
1
"AmazonAthenaFull"
Clear filters
Policy name
Type
Description
AmazonAthenaFullAccess
AWS managed
Provide full access t
Set permissions boundary - optional Info
Set a permissions boundary to control the maximum permissions this role can have. This is not a common setting, but you can use it to delegate permission management to others.

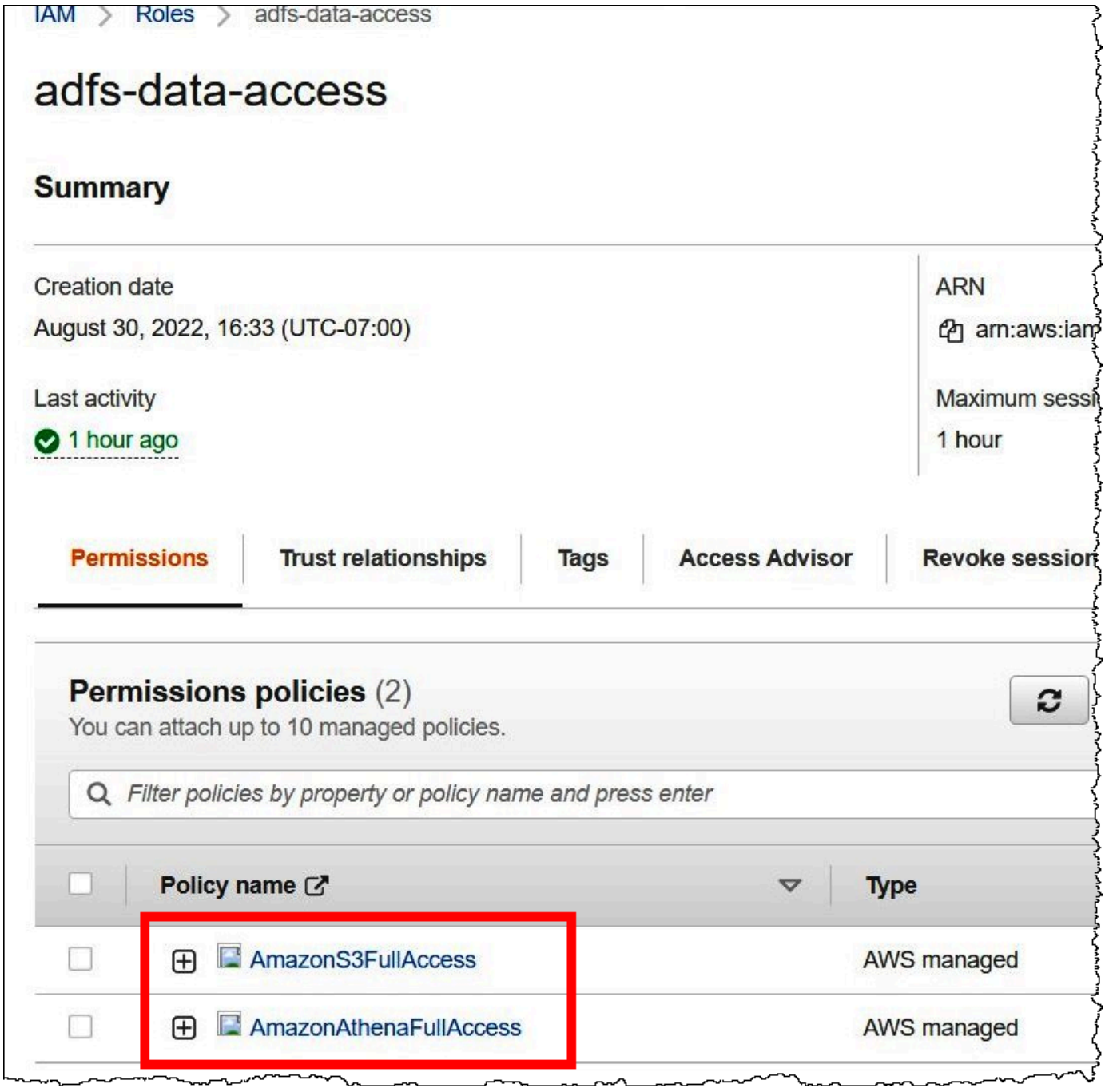


7. Seleziona Successivo.
8. Nella pagina Name, review, and create (Assegna un nome, verifica e crea), per Role name (Nome ruolo) inserisci un nome per il ruolo. Questo tutorial usa il nome adfs-data-access.

   Nel Passaggio 1: Selezione delle entità attendibili, il campo Principal (Principale) dovrebbe essere compilato automaticamente con `"Federated:" "arn:aws:iam::account_id:saml-provider/adfs-saml-provider"`. Il campo `Condition` deve contenere `"SAML:aud"` e `"https://signin.aws.amazon.com/saml"`.

Passaggio 2: Aggiunta delle autorizzazioni mostra le policy che hai collegato al ruolo.

9. Scegliere Crea ruolo. Un banner informativo conferma la creazione del ruolo.
10. Nella pagina Roles (Ruoli), scegli il nome del ruolo appena creato. La pagina di riepilogo del ruolo mostra le policy collegate.



2. Configurazione di AD FS

Ora è possibile aggiungere AWS come relying party e scrivere le regole di attestazione SAML in modo da poter inviare gli attributi utente corretti a AWS per l'autorizzazione.

La federazione basata su SAML coinvolge due parti: l'IdP (Active Directory) e il relying party (AWS), che è il servizio o l'applicazione che utilizza l'autenticazione dall'IdP.

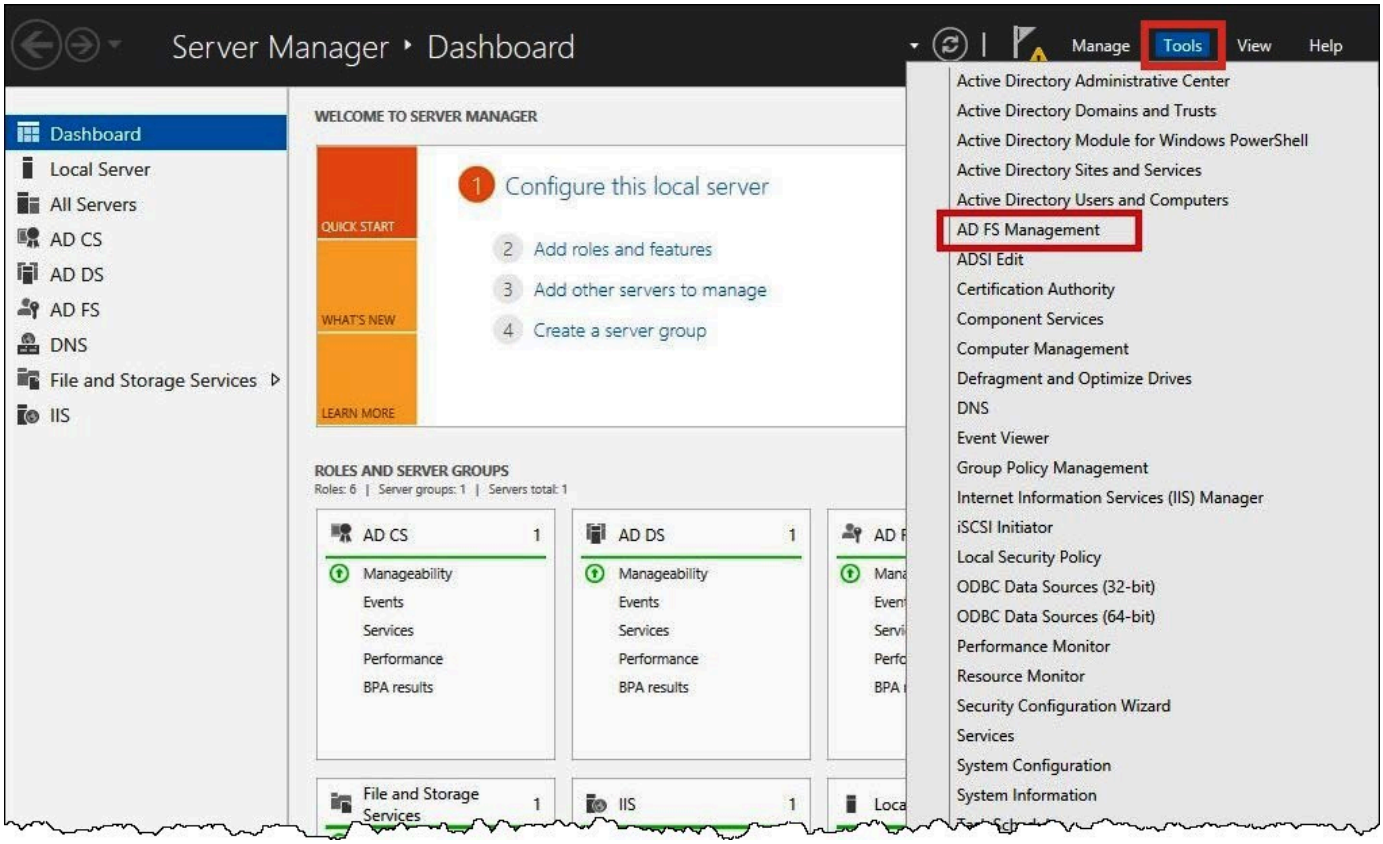
Per configurare AD FS, innanzitutto è necessario aggiungere la fiducia nel relying party, dopodiché occorre configurare le regole di attestazione SAML per il relying party. AD FS utilizza le regole di attestazione per creare un'asserzione SAML che viene inviata a un relying party. L'asserzione SAML afferma che le informazioni sull'utente AD sono vere e che l'utente è stato autenticato.

Aggiunta della fiducia in un relying party

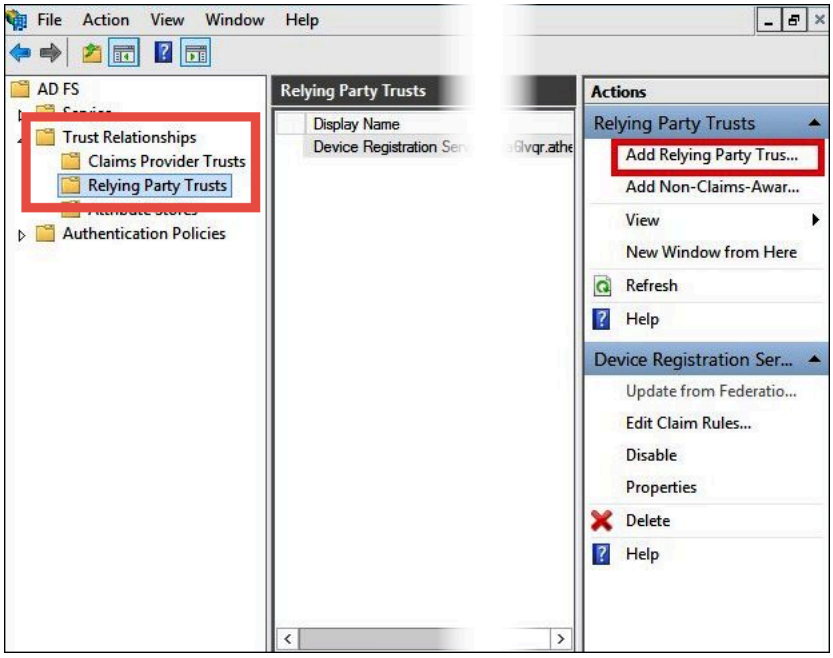
Per aggiungere la fiducia in un relying party in AD FS, si utilizza il gestore server AD FS.

Aggiunta della fiducia in un relying party in AD FS

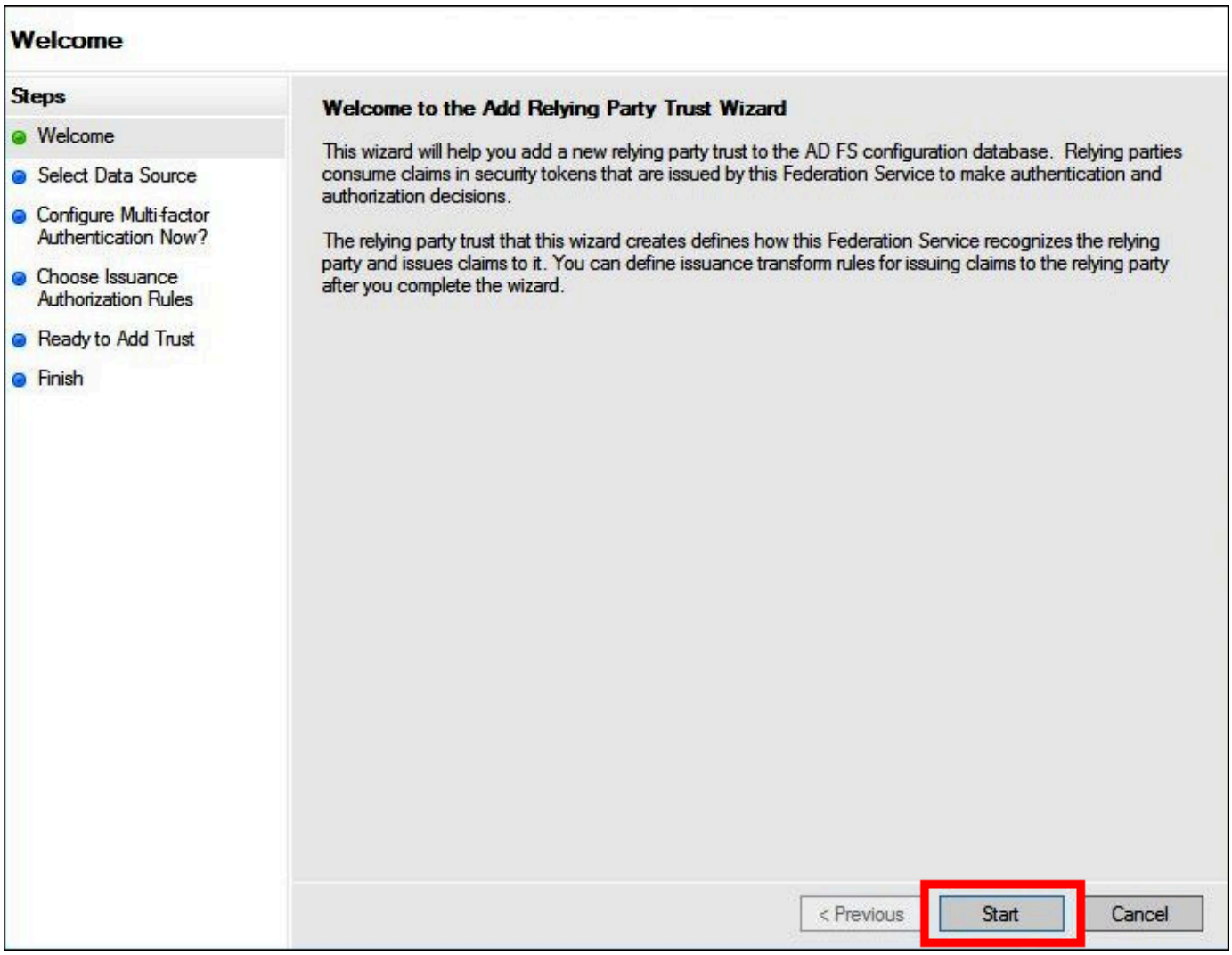
1. Accedi al server AD FS.
2. Nel menu Start, apri Server Manager (Gestore server).
3. Scegli Tools (Strumenti), quindi scegli AD FS Management (Gestione AD FS).



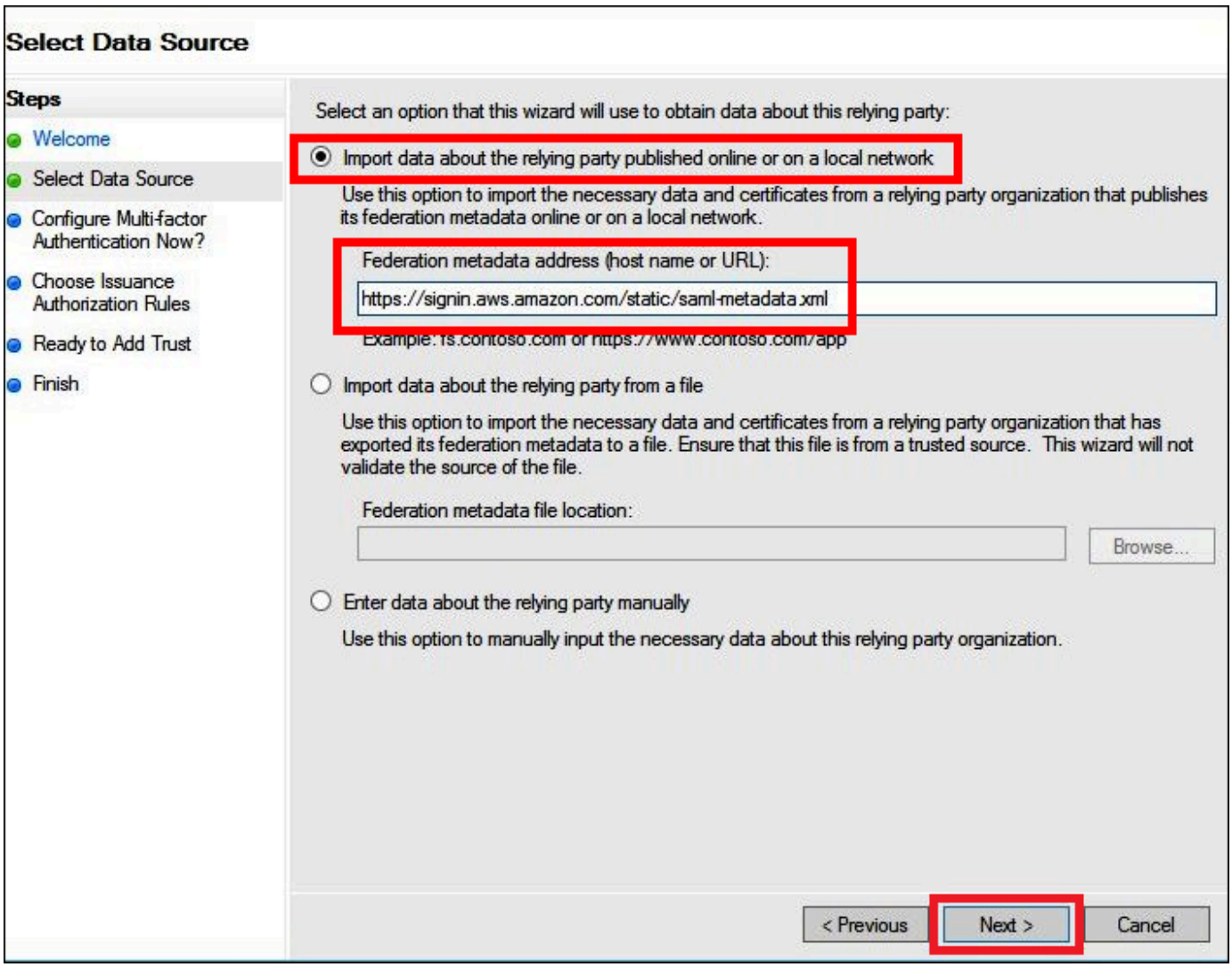
4. Nel riquadro di navigazione, in Trust Relationships (Relazioni di fiducia), scegli Relying Party Trust (Fiducia nel relying party).
5. In Actions (Operazioni), scegli Add Relying Party Trust (Aggiungi fiducia in un relying party).



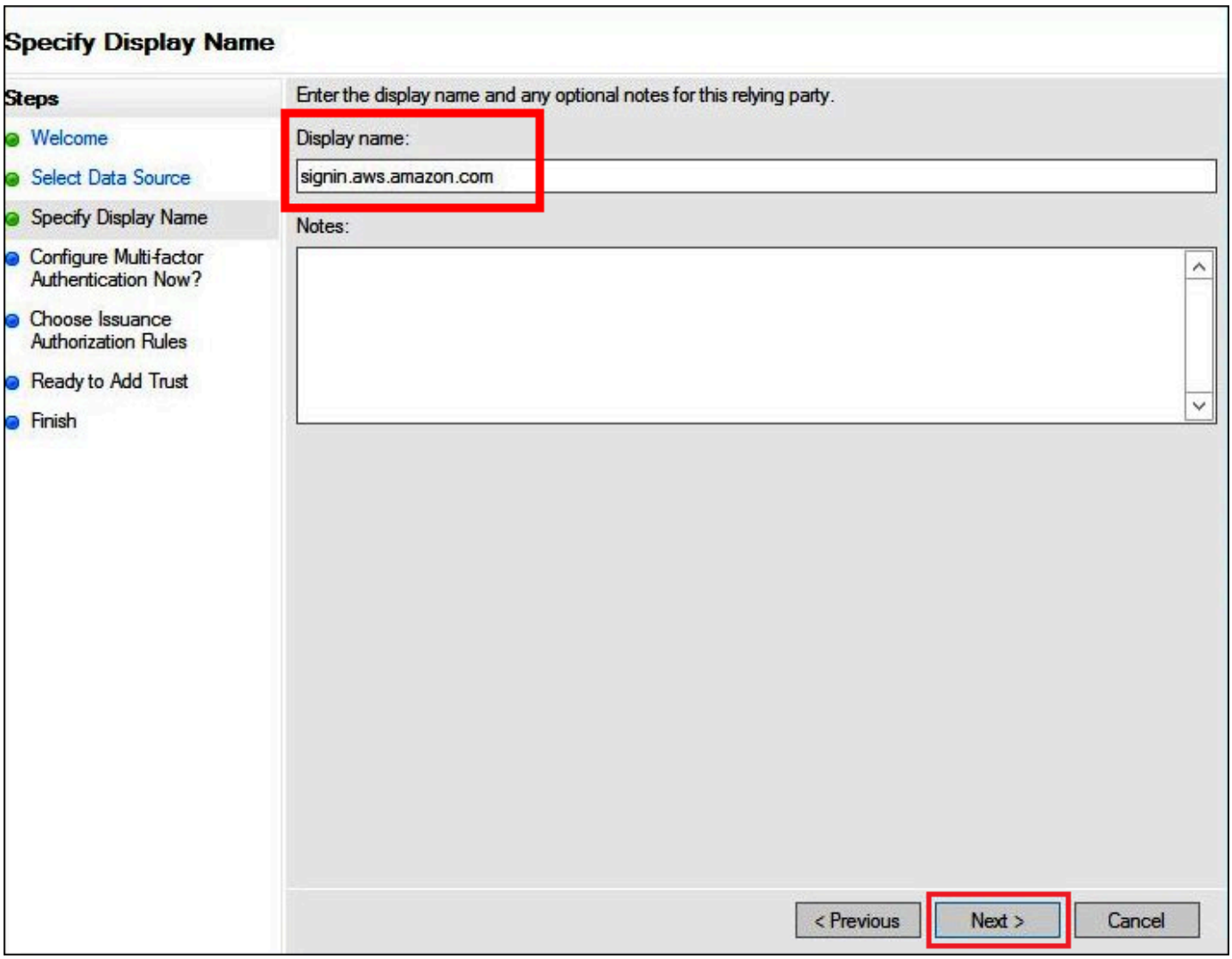
6. Nella pagina Add Relying Party Trust Wizard (Aggiunta guidata relazione di trust), scegliere Start (Avvia).



7. Nella pagina Select Data Source (Seleziona origine dati), seleziona l'opzione Import data about the relying party published online or on a local network (Importa dati sul relying party pubblicati online o in una rete locale).
8. In Federation metadata address (host name or URL) (Indirizzo dei metadati di federazione [nome host o URL]), inserisci l'URL **`https://signin.aws.amazon.com/static/saml-metadata.xml`**.
9. Seleziona Successivo.

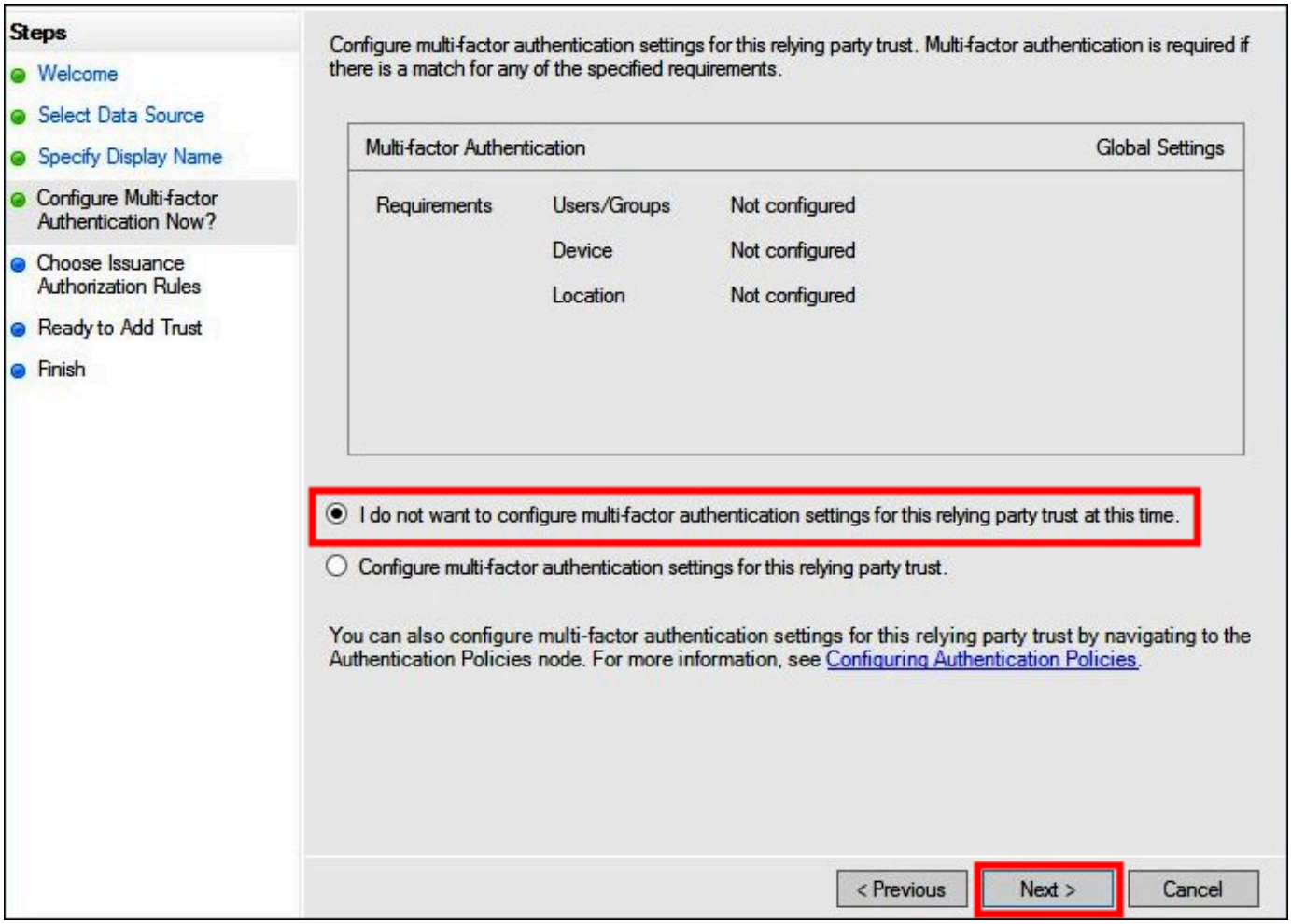


10. Nella pagina Specify Display Name (Specifica il nome da visualizzare), in Display name (Nome da visualizzare) inserisci un nome da visualizzare per il tuo relying party, quindi scegli Next (Successivo).



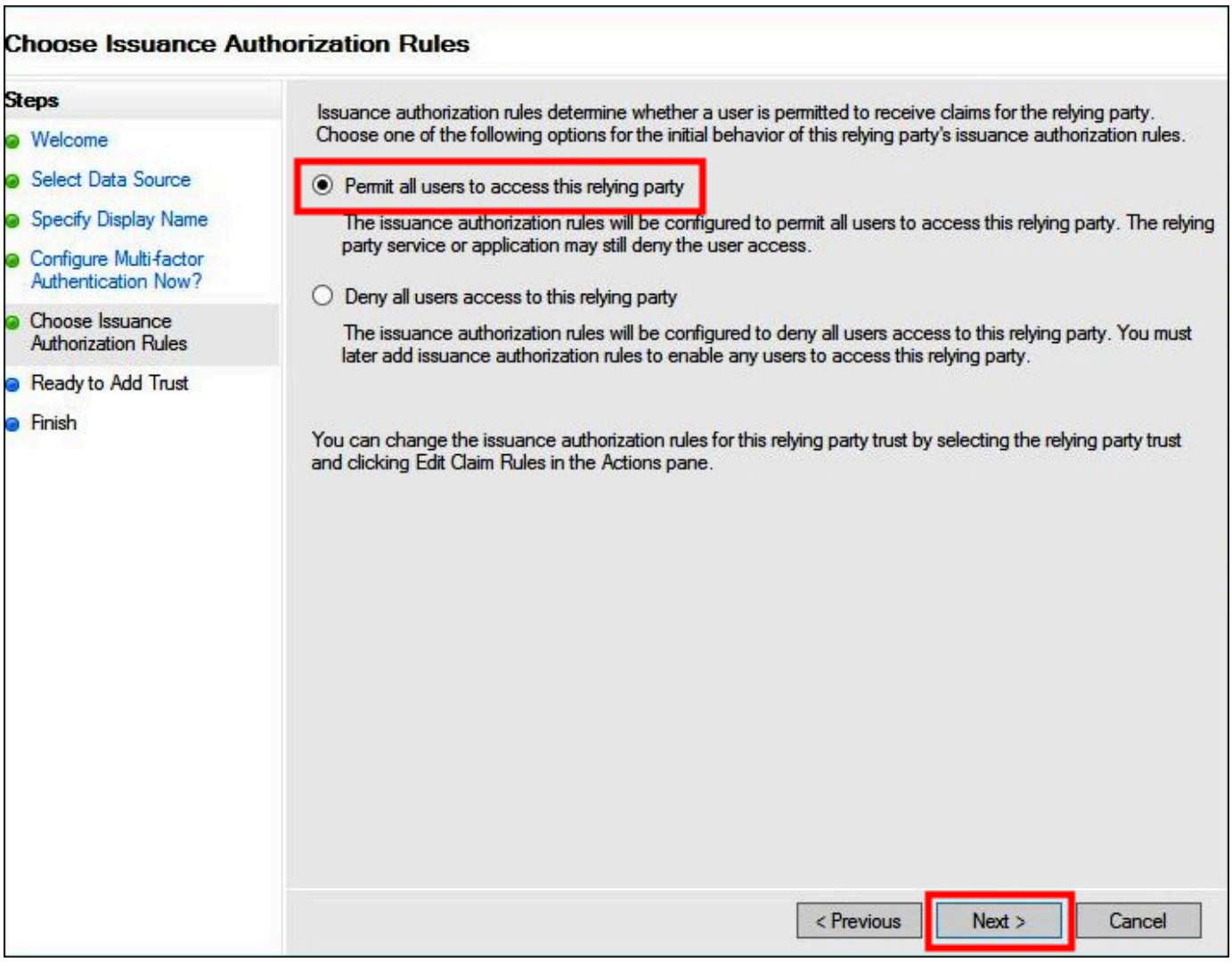
11. Nella pagina Configure Multi-factor Authentication Now (Configura ora l'autenticazione a più fattori), in questo tutorial viene selezionato I do not want to configure multi-factor authentication for this relying party trust at this time (Non voglio configurare l'autenticazione a più fattori per questo relying party in questo momento).

    Per maggiore sicurezza, ti consigliamo di configurare l'autenticazione a più fattori per favorire la protezione delle risorse AWS. Poiché utilizza un set di dati di esempio, questo tutorial non abilita l'autenticazione a più fattori.

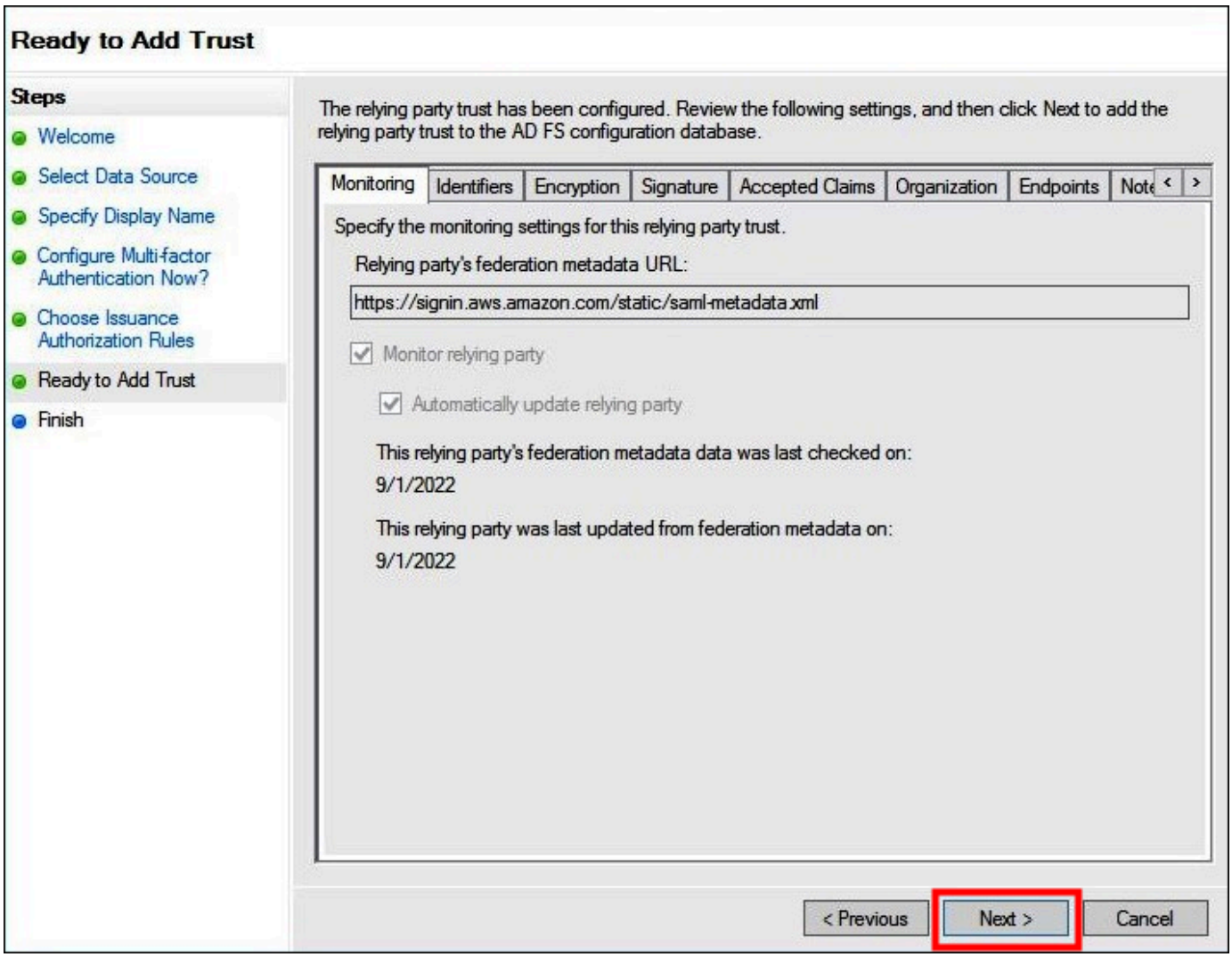


12. Seleziona Successivo.
13. Nella pagina Choose Issuance Authorization Rules (Scegli regole di autorizzazione emissione), seleziona Permit all users to access this relying party (Autorizza tutti gli utenti ad accedere a questo relying party).

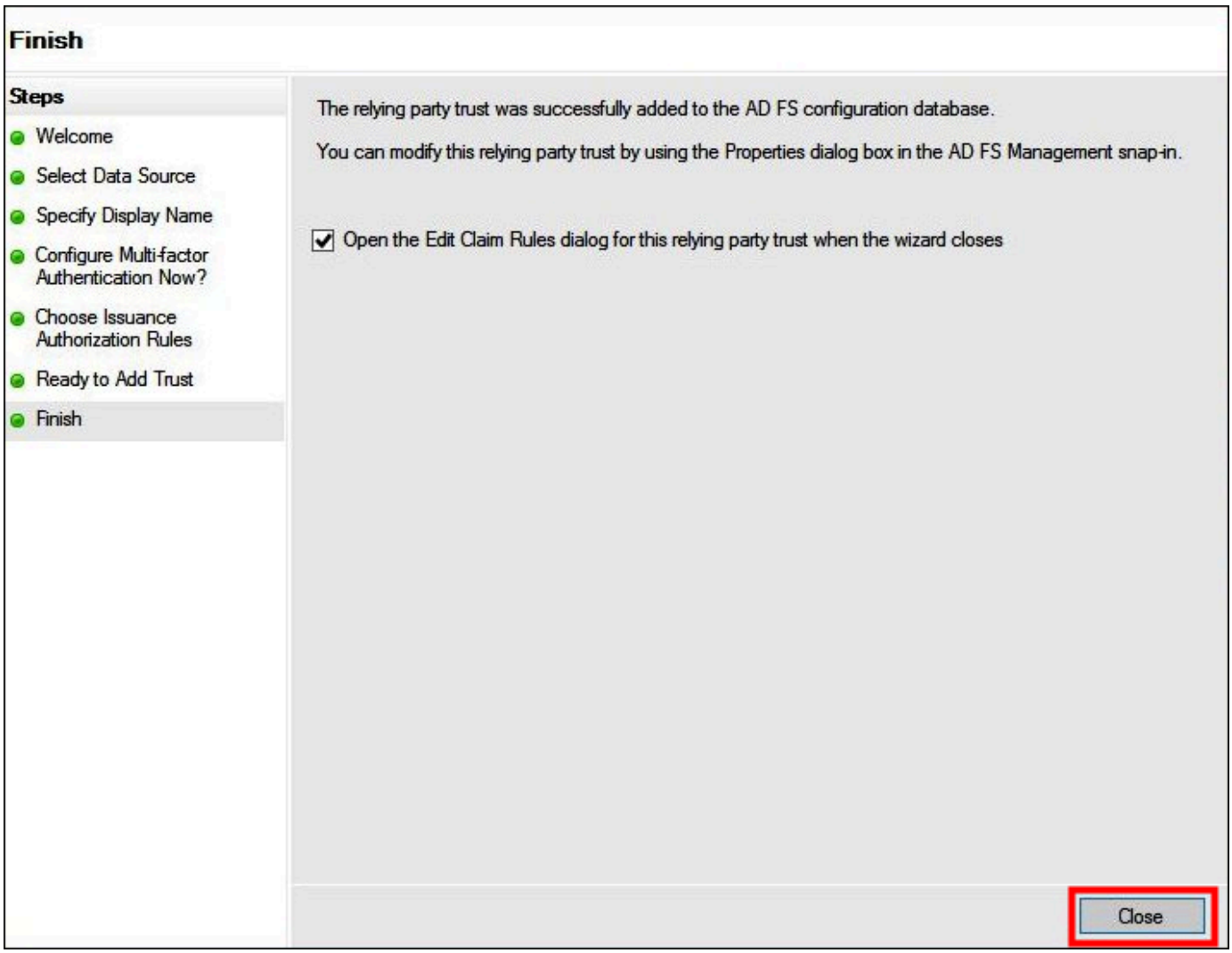
    Questa opzione consente a tutti gli utenti di Active Directory di utilizzare AD FS con AWS come relying party. Dovresti considerare i tuoi requisiti di sicurezza e adattare questa configurazione di conseguenza.



14. Seleziona Successivo.
15. Nella pagina Ready to Add Trust (Pronto per aggiungere la fiducia), scegli Next (Successivo) per aggiungere la fiducia nel relying party al database di configurazione AD FS.



16. Nella pagina Finish (Concludi), scegli Close (Chiudi).



Configurazione delle regole di attestazione SAML per il relying party

In questa attività, crei due set di regole di attestazione.

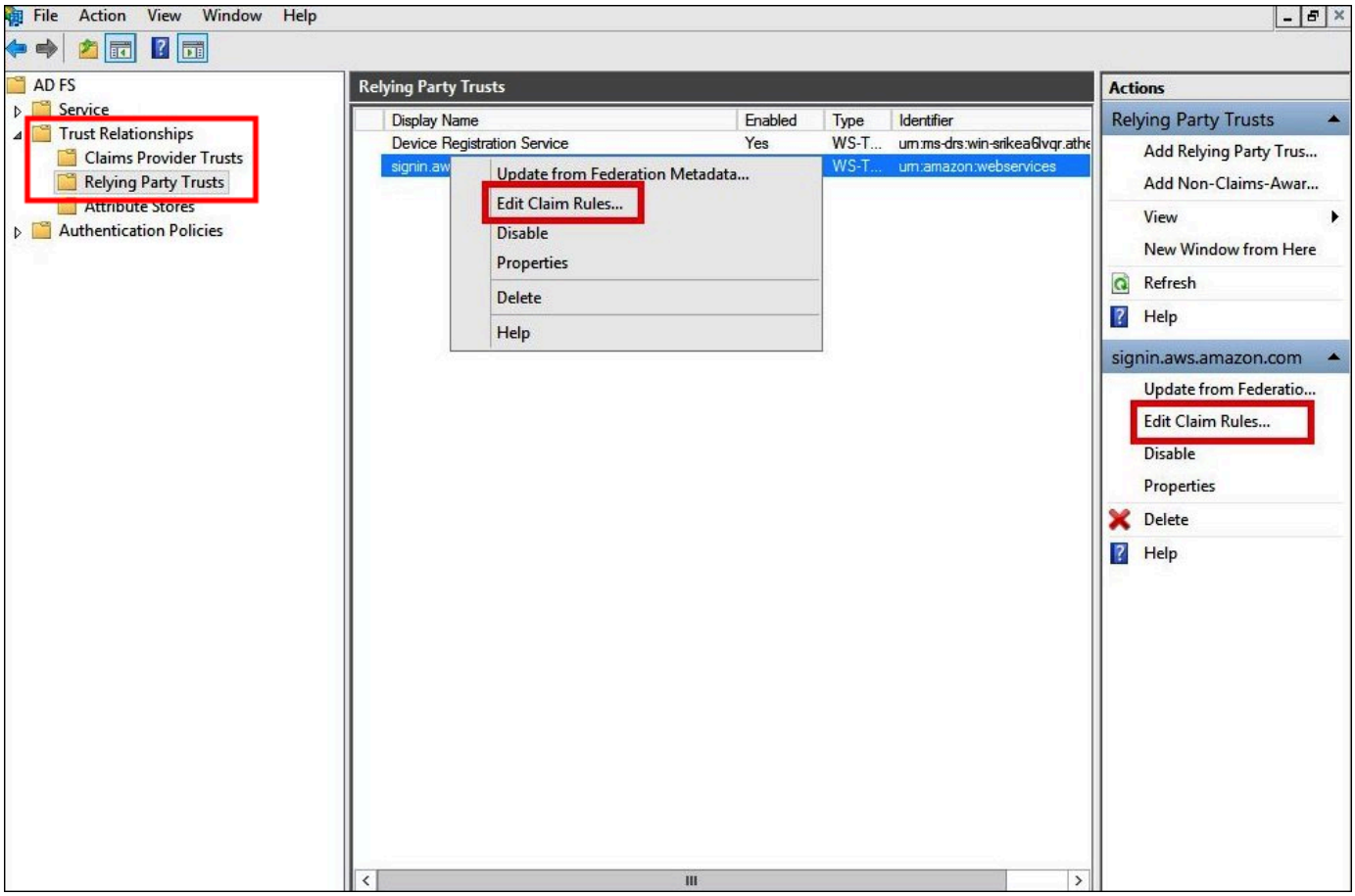
Il primo set, le regole da 1 a 4, contiene le regole di attestazione AD FS necessarie per assumere un ruolo IAM basato sull'appartenenza al gruppo AD. Queste sono le stesse regole che crei se desideri stabilire un accesso federato alla AWS Management Console.

Il secondo set, le regole 5 e 6, sono le regole di attestazione necessarie per il controllo degli accessi di Athena.

Creazione di regole di attestazione AD FS

1. Nel riquadro di navigazione della console di gestione AD FS, scegli Trust Relationships (Relazioni di fiducia), quindi scegli Relying Party Trusts (Fiducia nei relying party).

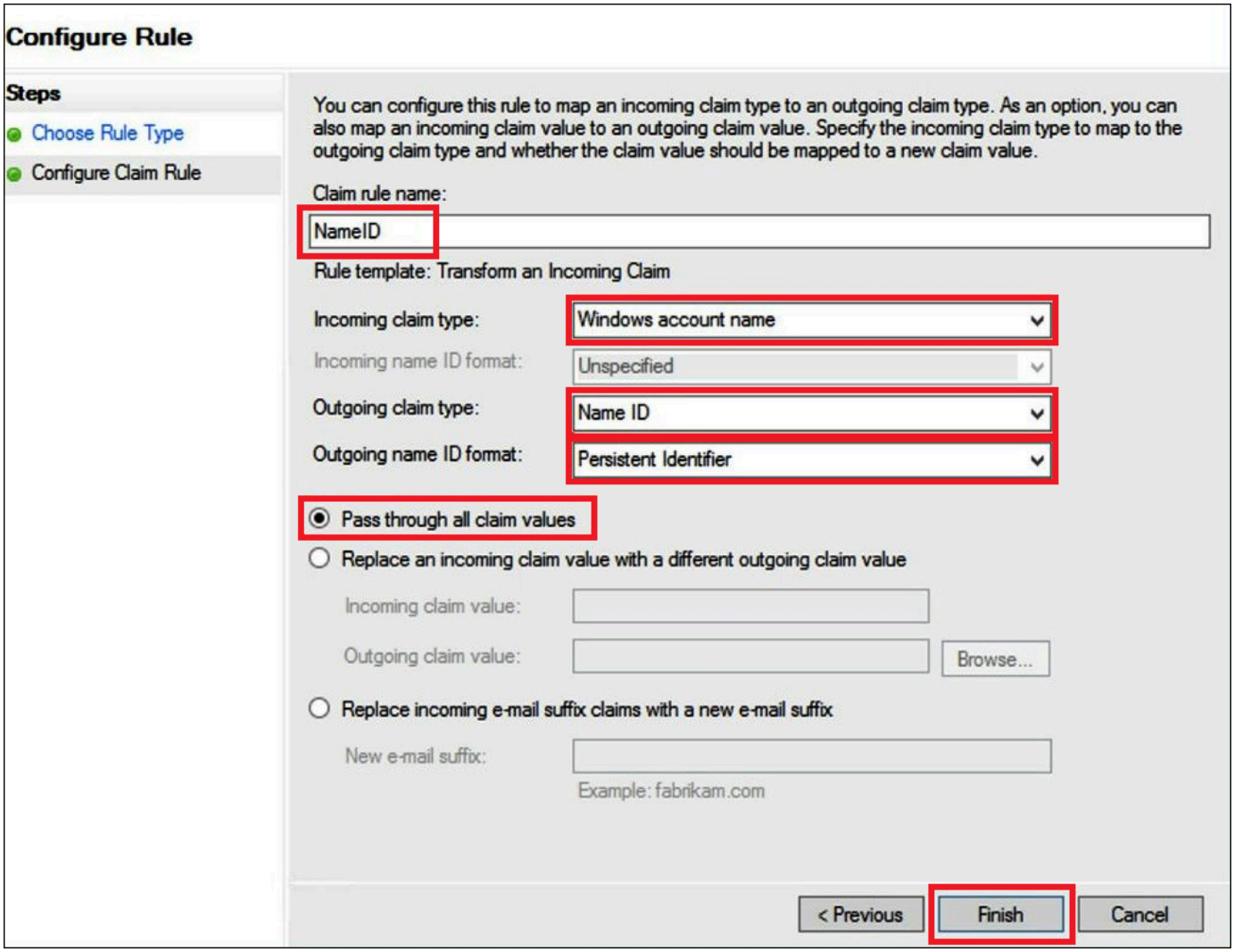
2. Trova il relying party che hai creato nella sezione precedente.
3. Fai clic con il pulsante destro del mouse sul relying party e scegli Edit Claim Rules (Modifica regole di attestazione) oppure scegli Edit Claim Rules (Modifica regole di attestazione) dal menu Actions (Operazioni).



4. Selezionare Add Rule (Aggiungi regola).
5. Nella pagina Configura regola della procedura guidata Add Transform Claim Rule (Aggiungi trasformazione regola di attestazione), inserisci le seguenti informazioni per creare la regola di attestazione 1, quindi scegli Finish (Concludi).

   - In Claim rule name (Nome regola di attestazione), inserisci **`NameID`**.
   - In Rule template (Modello di regola), scegli Transform an Incoming Claim (Trasforma un'attestazione in entrata).
   - In Incoming claim type (Tipo di attestazione in entrata), scegli Windows account name (Nome account Windows).
   - In Outgoing claim type (Tipo di attestazione in uscita), scegli Name ID (ID nome).

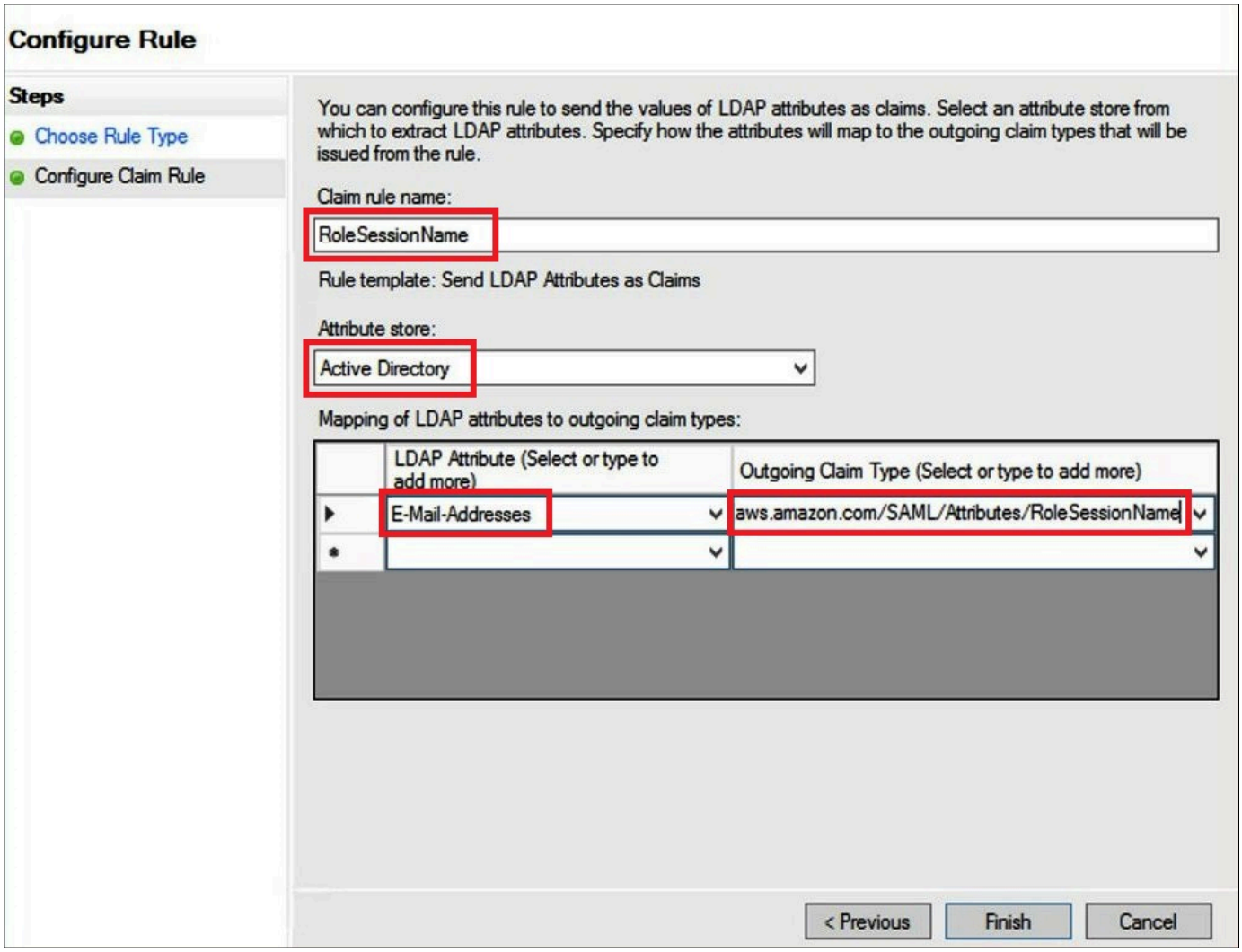
- In Outgoing name ID format (Formato ID nome in uscita), scegliere Persistent Identifier (Identificatore persistente).
- Scegli Pass through all claim values (Passa tutti i valori di attestazione).



6. Scegli Add Rule (Aggiungi regola), inserisci le seguenti informazioni per creare la regola di attestazione 2, quindi scegli Finish (Concludi).

- In Claim rule name (Nome regola di attestazione), inserisci **`RoleSessionName`**.
- In Rule template (Modello di regola), scegli Send LDAP Attribute as Claims (Invia attributo LDAP come attestazione).
- In Attribute store (Archivio attributi), scegliere Active Directory.
- In Mapping of LDAP attributes to outgoing claim types (Mappatura degli attributi LDAP ai tipi di attestazione in uscita), aggiungi l'attributo **`E-Mail-Addresses`**. In Outgoing Claim Type

(Tipo di attestazione in uscita), inserisci **`https://aws.amazon.com/SAML/Attributes/RoleSessionName`**.

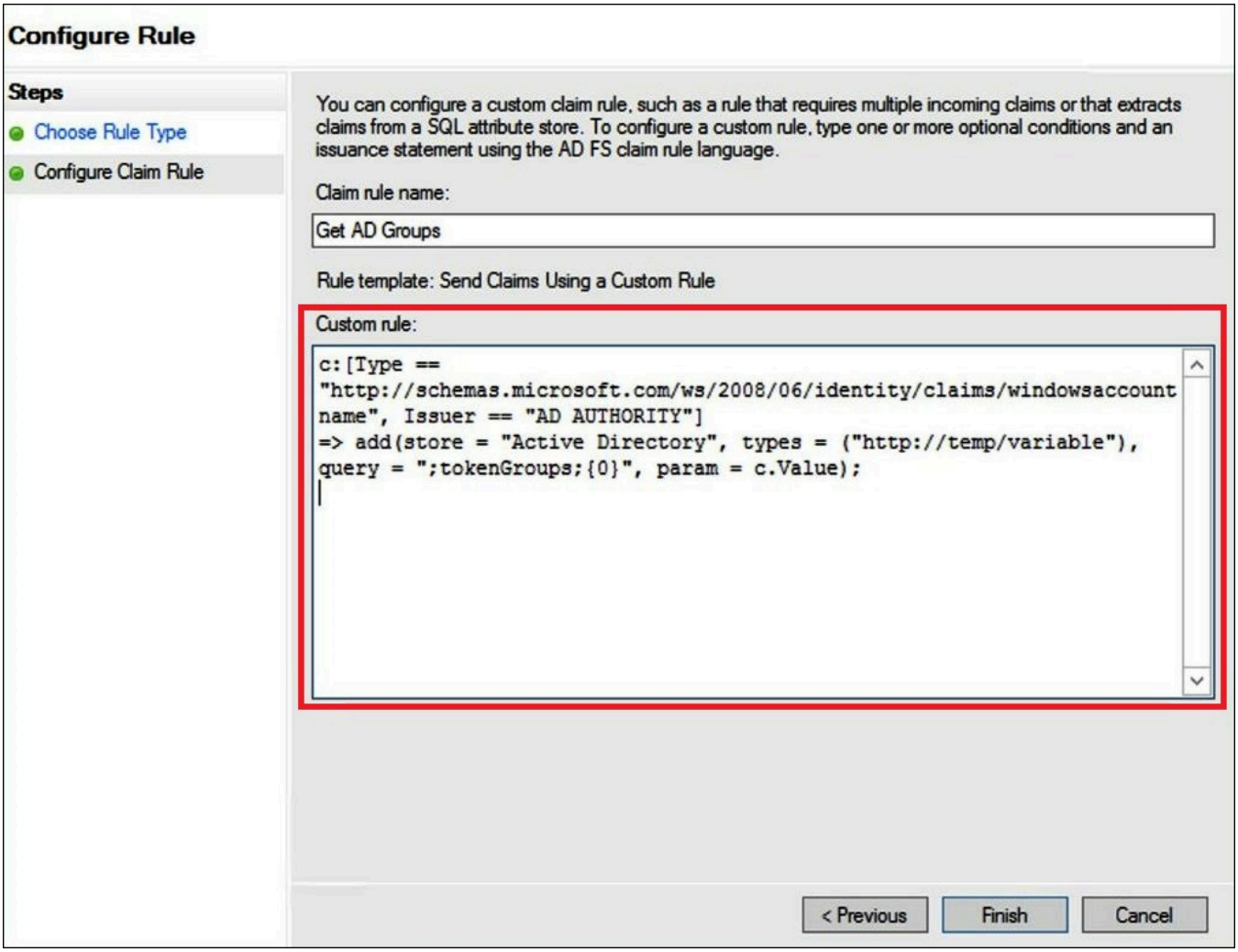


7. Scegli Add Rule (Aggiungi regola), inserisci le seguenti informazioni per creare la regola di attestazione 3, quindi scegli Finish (Concludi).

   - In Claim rule name (Nome regola di attestazione), inserisci **`Get AD Groups`**.
   - In Rule template (Modello di regola), scegli Send Claims Using a Custom Rule (Invia attestazioni utilizzando una regola personalizzata).
   - In Custom rule (Regola personalizzata), inserisci il codice seguente:

```
c:[Type == "http://schemas.microsoft.com/ws/2008/06/identity/claims/
windowsaccountname",
```

```
 Issuer == "AD AUTHORITY"]=> add(store = "Active Directory", types = ("http://
temp/variable"),
 query = ";tokenGroups;{0}", param = c.Value);
```

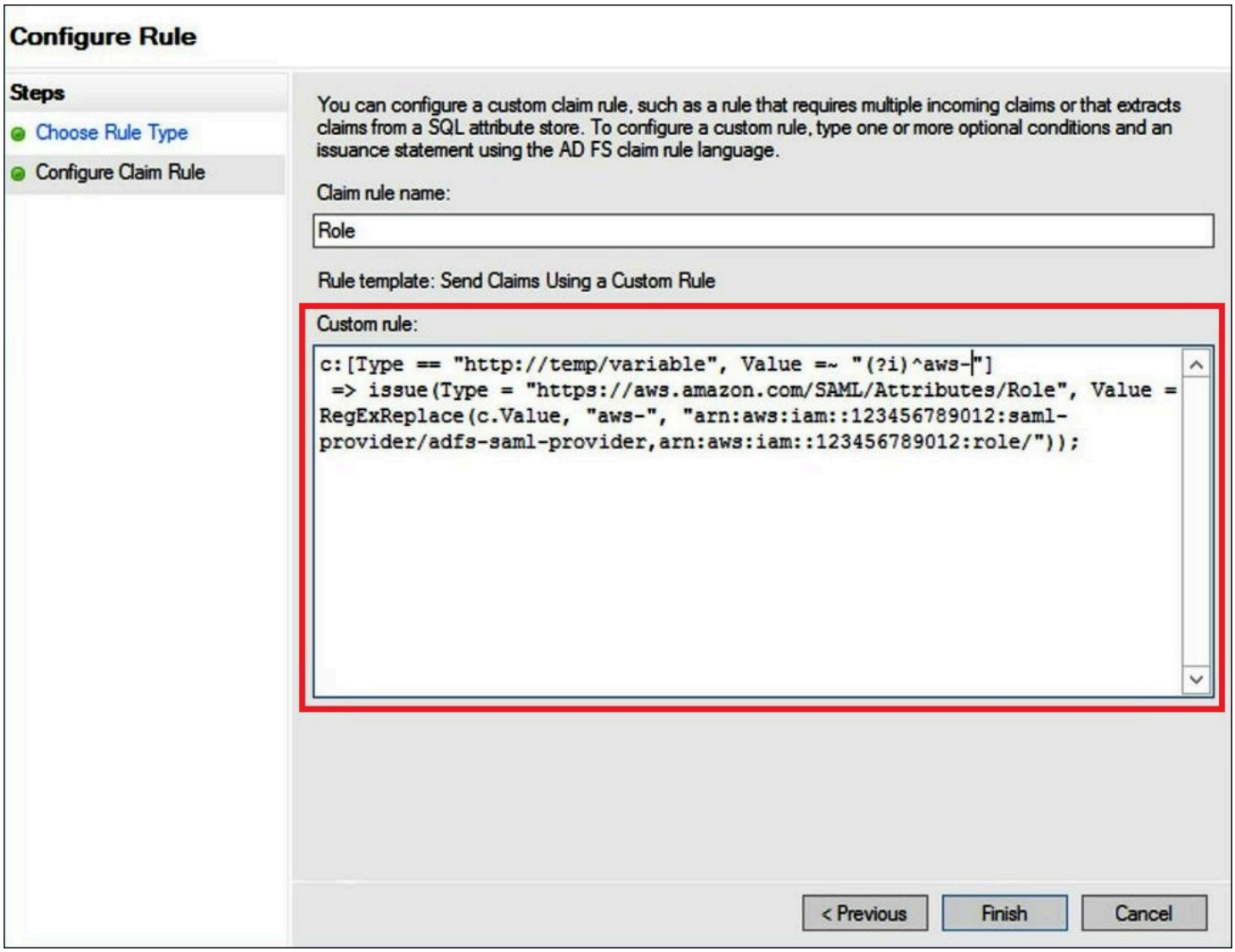


8. Selezionare Add Rule (Aggiungi regola). Inserisci le seguenti informazioni per creare la regola di attestazione 4, quindi scegli Finish (Concludi).

   - In Claim rule name (Nome regola di attestazione), inserisci **`Role`**.
   - In Rule template (Modello di regola), scegli Send Claims Using a Custom Rule (Invia attestazioni utilizzando una regola personalizzata).
   - In Custom rule (Regola personalizzata), inserisci il seguente codice con il numero di account e il nome del provider SAML che hai creato in precedenza:

```
c:[Type == "http://temp/variable", Value =~ "(?i)^aws-"]=> issue(Type = "https://
aws.amazon.com/SAML/Attributes/Role",
```

```
Value = RegExReplace(c.Value, "aws-", "arn:aws:iam::AWS_ACCOUNT_NUMBER:saml-
provider/adfs-saml-provider,arn:aws:iam:: AWS_ACCOUNT_NUMBER:role/"));
```

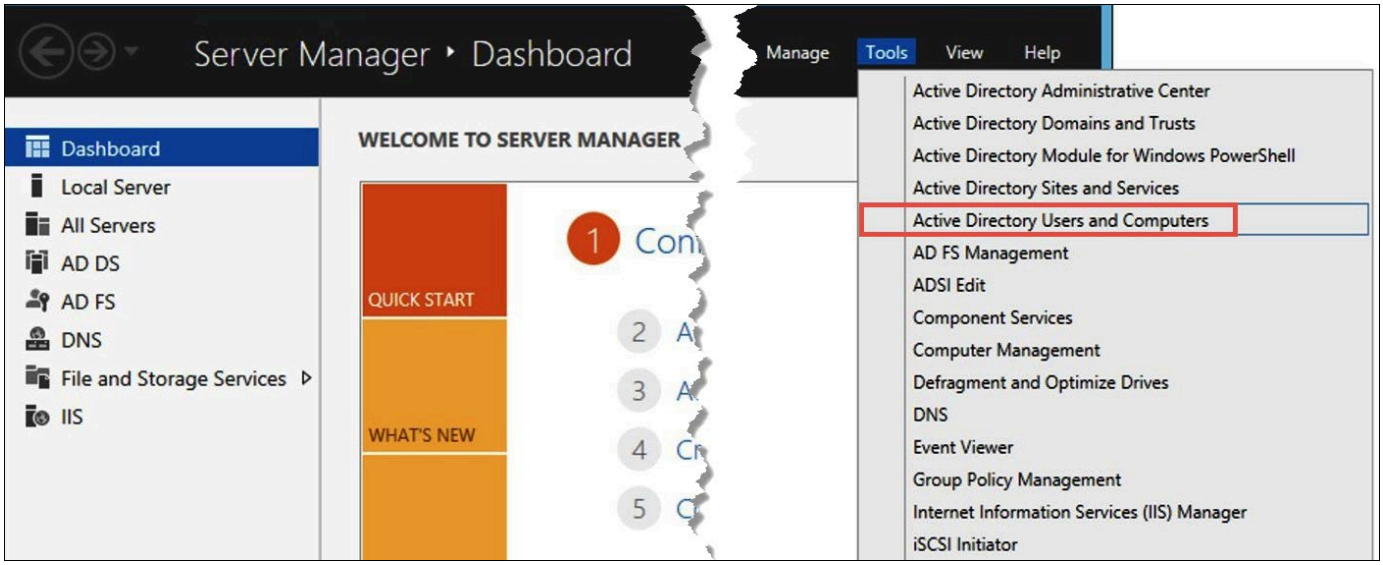


3. Creazione di utenti e gruppi Active Directory

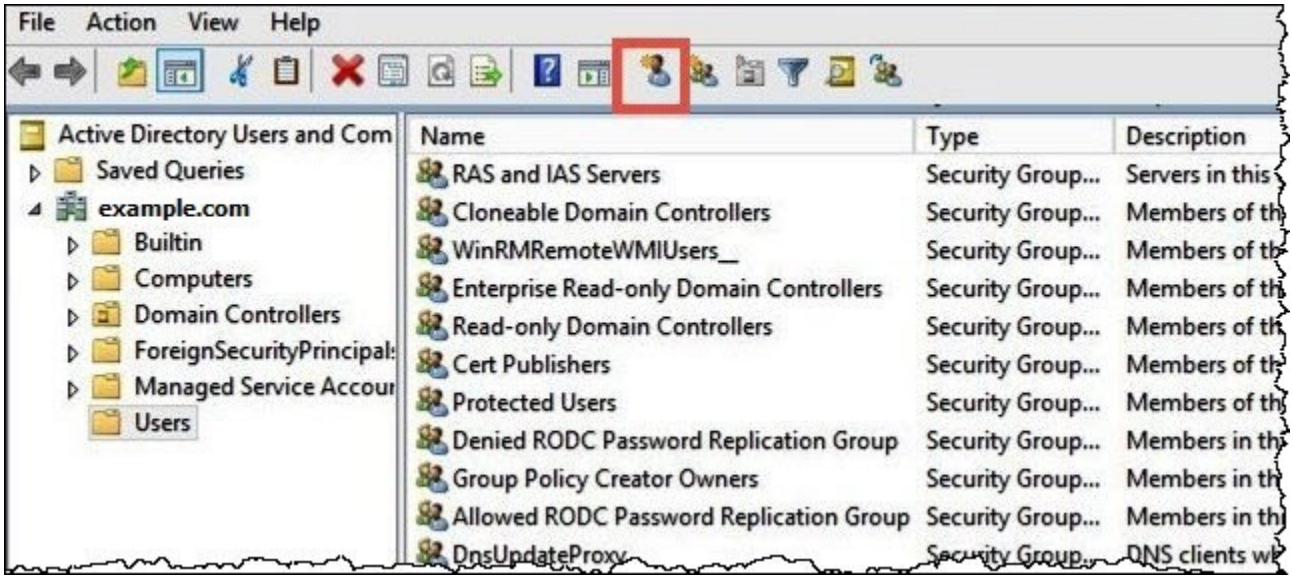
Ora è tutto pronto per creare utenti AD che accederanno ad Athena e gruppi AD in cui inserirli in modo da poter controllare i livelli di accesso per gruppo. Dopo avere creato gruppi AD che classificano i pattern di accesso ai dati, aggiungi gli utenti a tali gruppi.

Creazione di utenti AD per l'accesso ad Athena

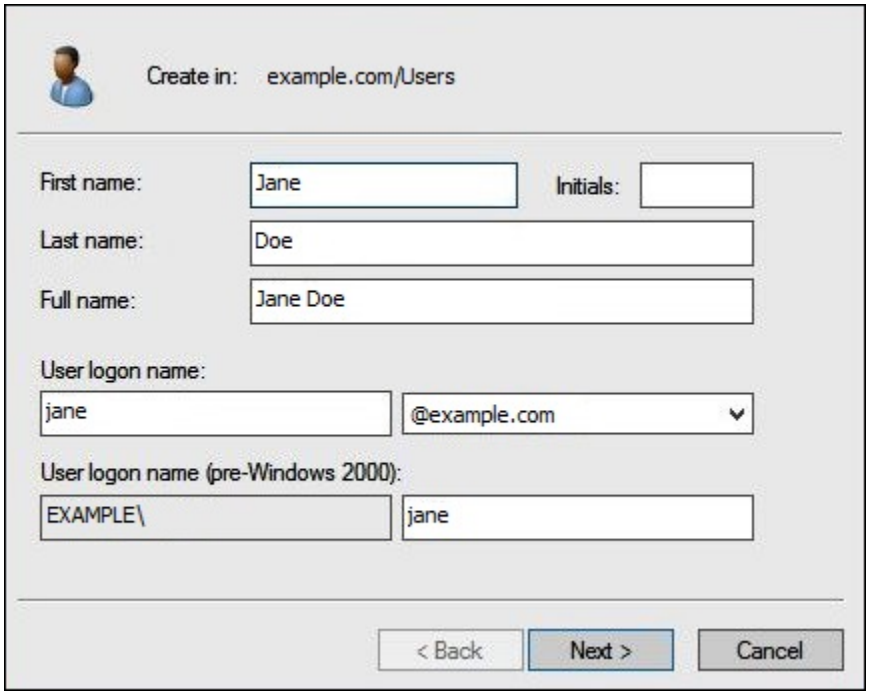
1. Nel pannello di controllo di Server Manager (Gestore server), scegli Tools (Strumenti), quindi scegli Active Directory Users and Computers (Strumento Utenti e computer di Active Directory).



2. Nel riquadro di navigazione, seleziona Users (Utenti).
3. Nella barra degli strumenti Active Directory Users and Computers (Utenti e computer di Active Directory), scegli l'opzione Create user (Crea utente).

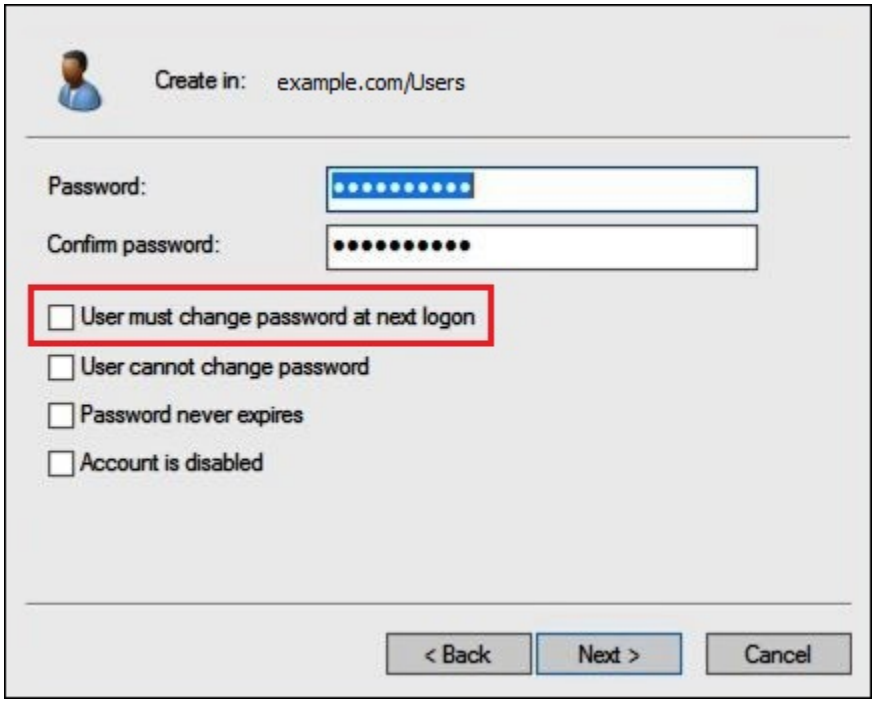


4. Nella finestra di dialogo New Object – User (Nuovo oggetto - Utente), in First name (Nome), Last name (Cognome) e Full name (Nome completo, inserisci un nome. In questo tutorial si utilizza **`Jane Doe`**.

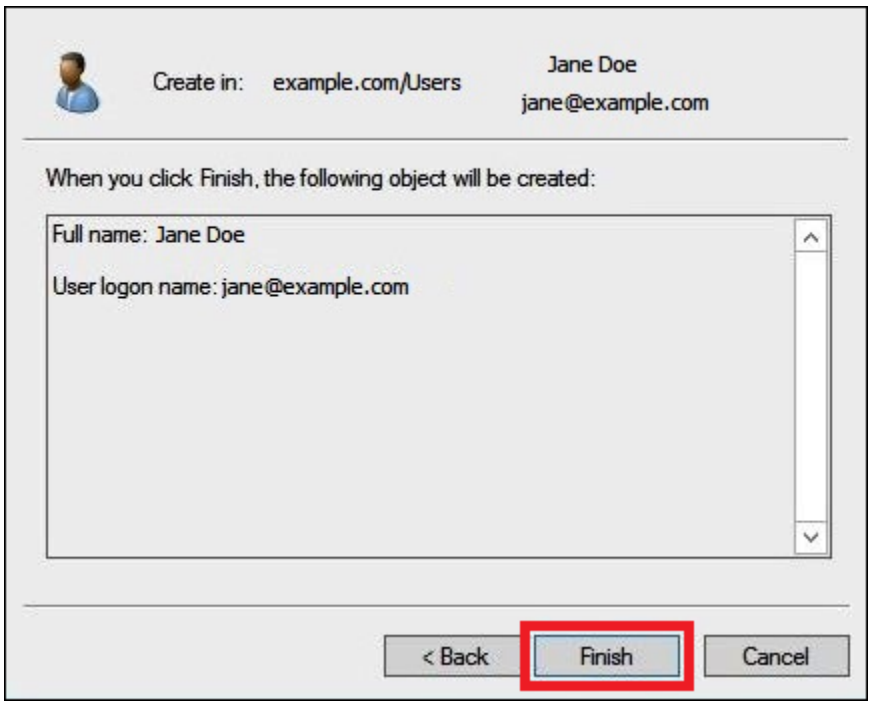


5. Seleziona Successivo.
6. In Password, inserisci una password, quindi digitala nuovamente per conferma.

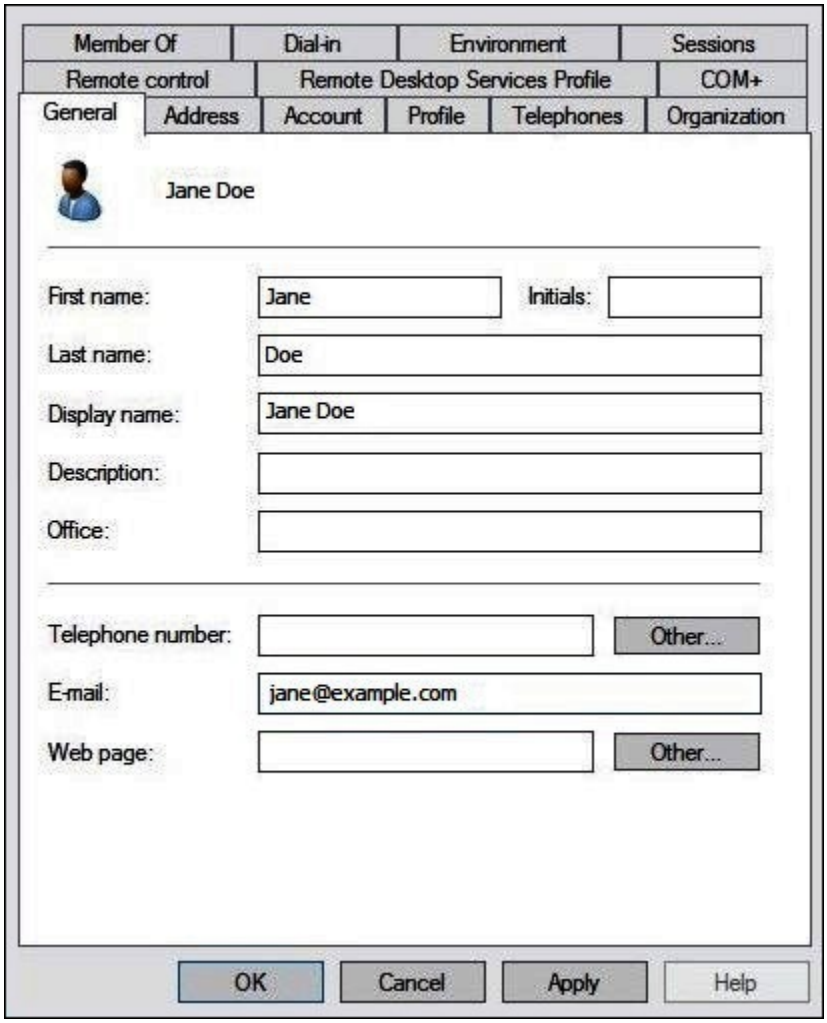
   Per semplicità, in questo tutorial viene deselezionato User must change password at next sign on (L'utente deve cambiare la password all'accesso successivo). Negli scenari reali, è necessario richiedere agli utenti appena creati di modificare la password.



7. Seleziona Successivo.
8. Scegli Finish (Fine).



9. In Active Directory Users and Computers (Strumento Utenti e computer di Active Directory), scegli il nome utente.
10. Nella finestra di dialogo Properties (Proprietà) per l'utente, in E-mail (Posta elettronica) inserisci un indirizzo e-mail. In questo tutorial si utilizza **`jane@example.com`**.



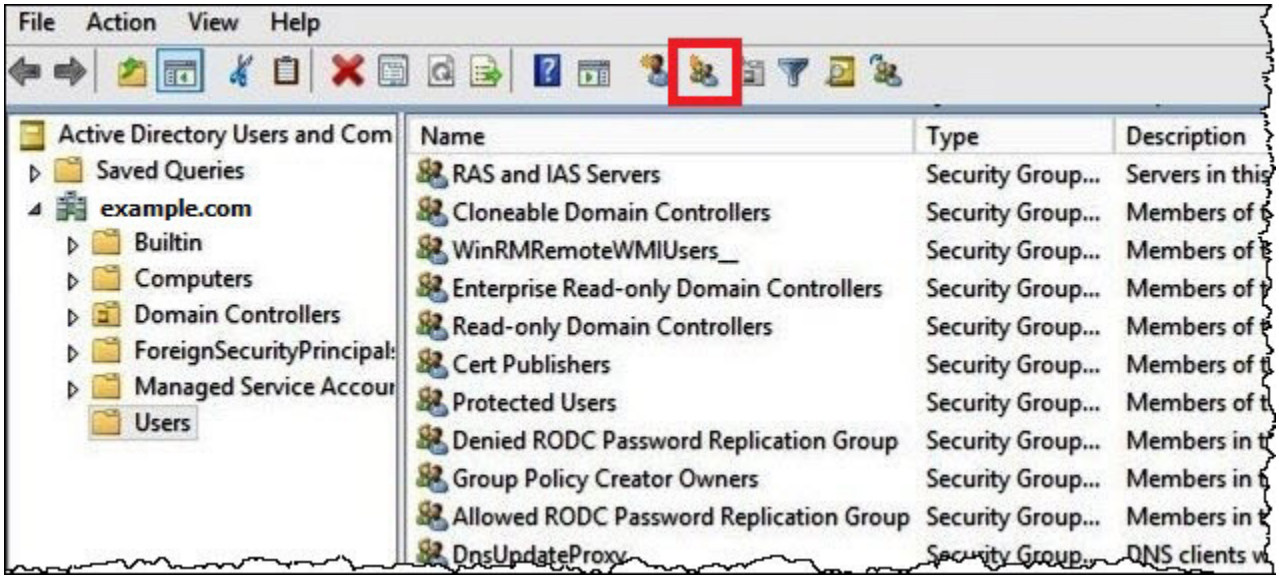
11. Scegli OK.

Creazione di gruppi AD per rappresentare i pattern di accesso ai dati

Puoi creare gruppi AD i cui membri assumono il ruolo IAM `adfs-data-access` al momento dell'accesso a AWS. L'esempio seguente crea un gruppo AD chiamato aws-adfs-data-access.

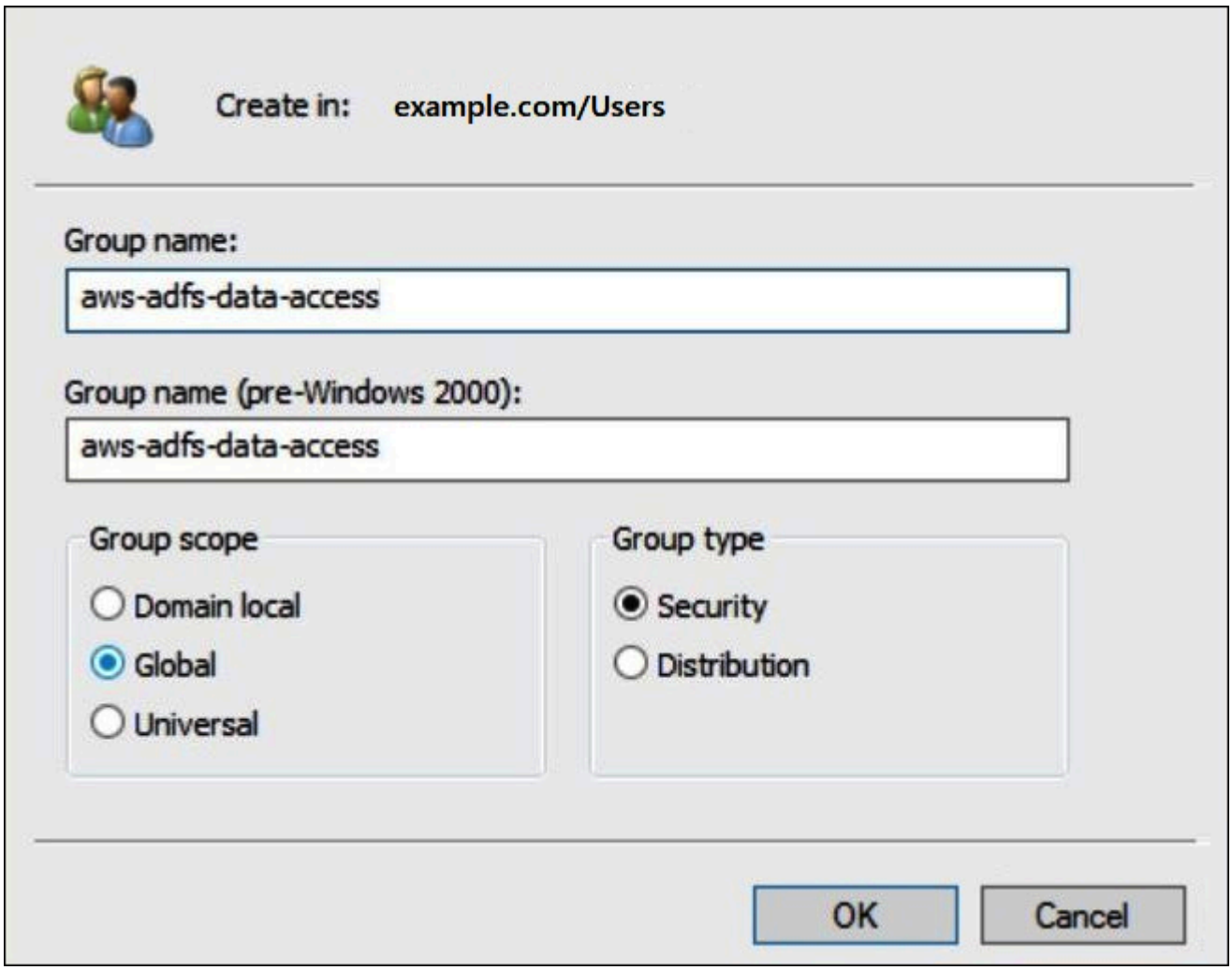
Creazione di un gruppo AD

1. Nel pannello di controllo di Server Manager (Gestore server), nel menu Tools (Strumenti), scegli Active Directory Users and Computers (Strumento Utenti e computer di Active Directory).
2. Sulla barra degli strumenti, scegli l'opzione Create new group (Crea nuovo gruppo).



3. Nella finestra di dialogo New Object - Group (Nuovo oggetto - Gruppo), inserisci le informazioni seguenti:

   - In Group Name (Nome del gruppo), inserisci **`aws-adfs-data-access`**.
   - In Group scope (Ambito del gruppo), seleziona Global (Globale).
   - In Group type (Tipo di gruppo), seleziona Security (Sicurezza).



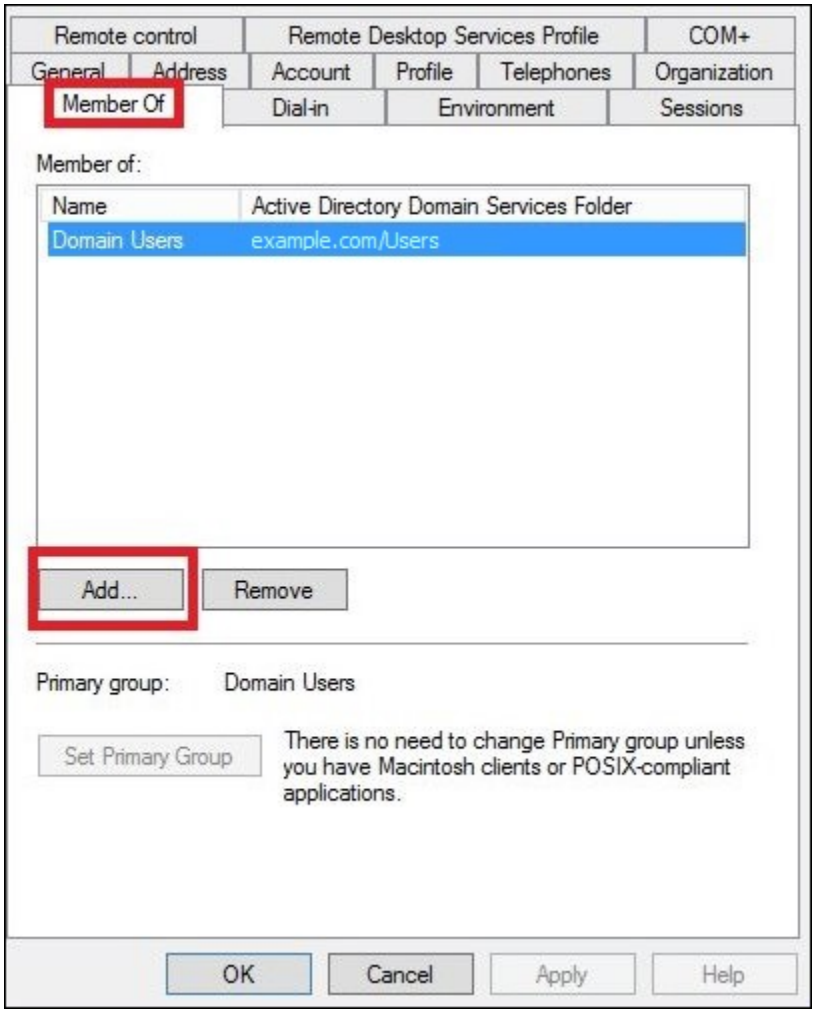
4. Scegli OK.

Aggiunta di utenti AD ai gruppi appropriati

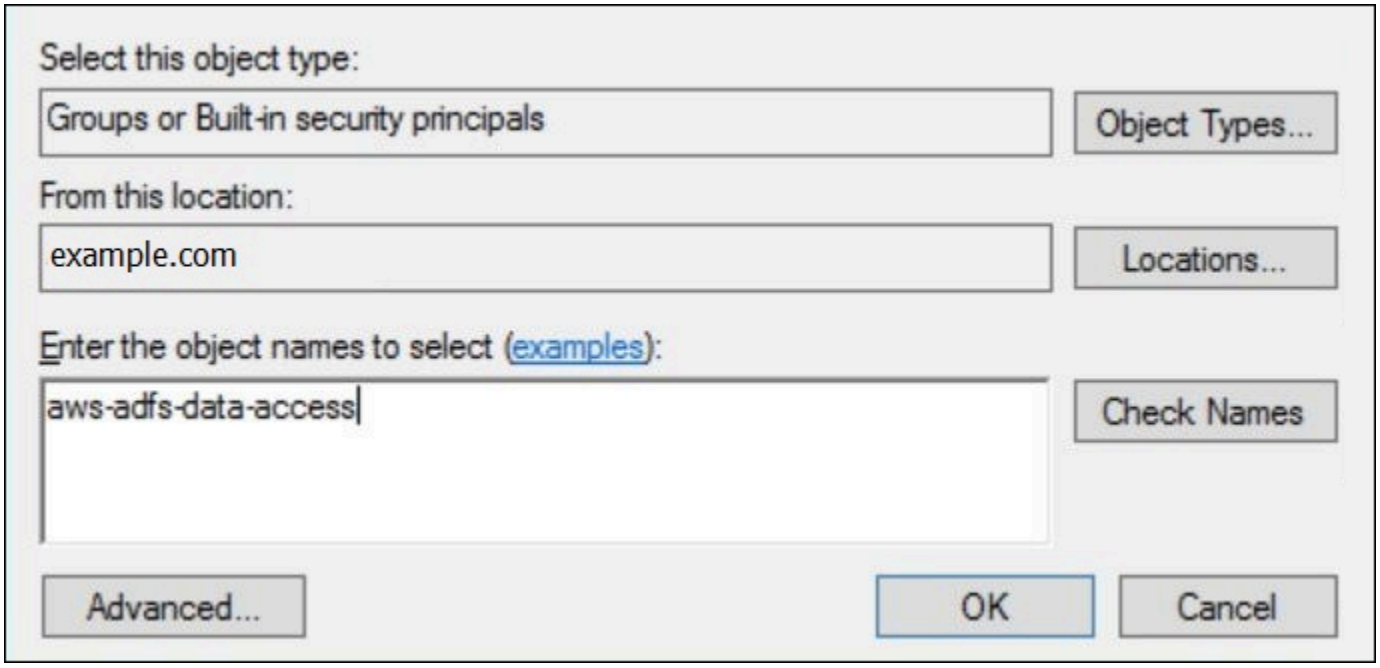
Ora che hai creato sia un utente AD sia un gruppo AD, puoi aggiungere l'utente al gruppo.

Aggiunta di un utente AD a un gruppo AD

1. Nel pannello di controllo di Server Manager (Gestore server), nel menu Tools (Strumenti), scegli Active Directory Users and Computers (Strumento Utenti e computer di Active Directory).
2. In First name (Nome) e Last name (Cognome), scegli un utente (ad esempio, Jane Doe).
3. Nella finestra di dialogo Properties (Proprietà) dell'utente, nella scheda Member Of (Membro di), scegli Add (Aggiungi).

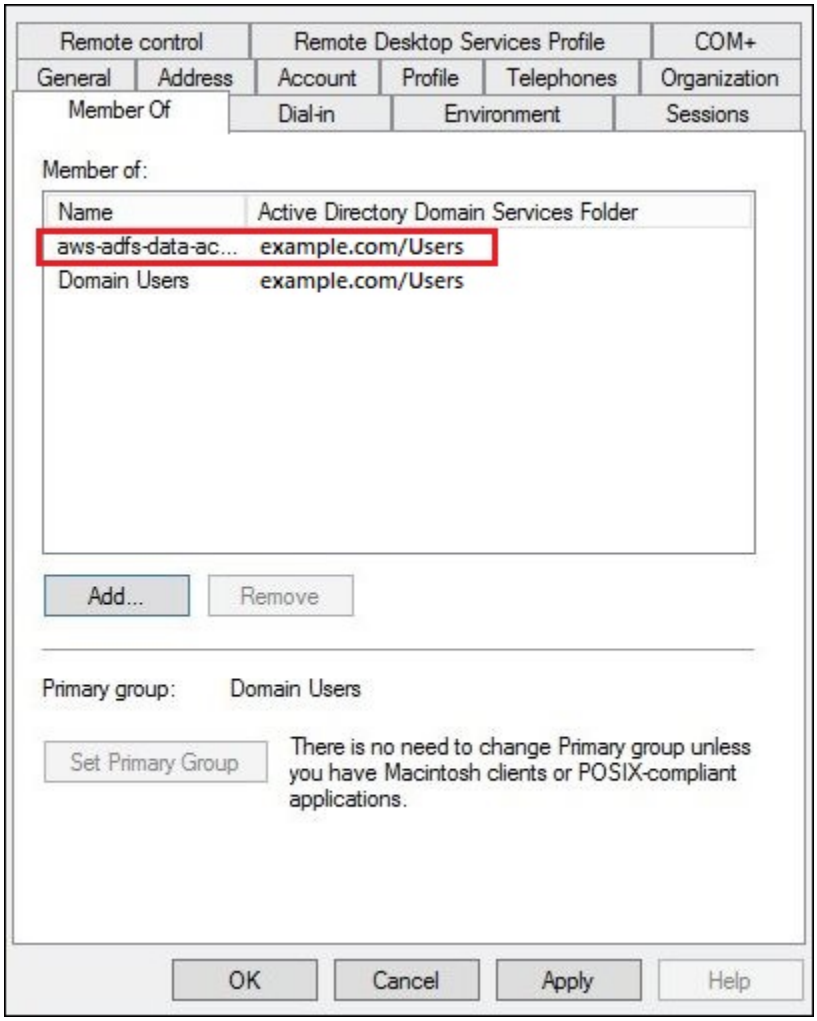


4. Aggiungi uno o più gruppi AD FS in base alle tue esigenze. Questo tutorial aggiunge il gruppo aws-adfs-data-access.
5. Nella finestra di dialogo Select Groups (Seleziona gruppi), in Enter the object names to select (Inserisci i nomi degli oggetti da selezionare), inserisci il nome del gruppo AD FS che hai creato (ad esempio **`aws-adfs-data-access`**), quindi scegli Check Names (Controlla nomi).



6. Scegli OK.

   Nella finestra di dialogo Properties (Proprietà) per l'utente, il nome del gruppo AD viene visualizzato nell'elenco Member of (Membro di).



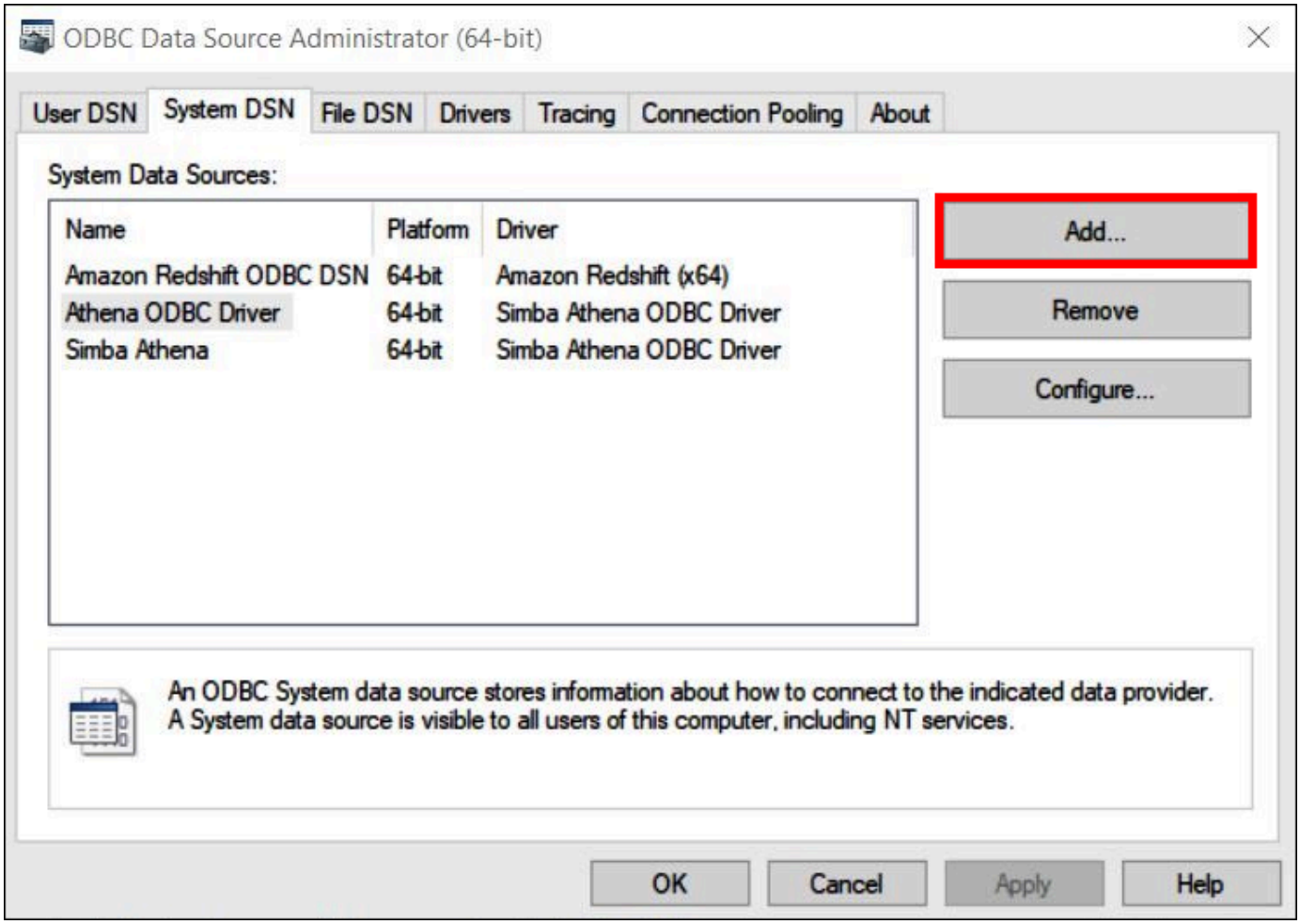
7. Scegli Apply (Applica), quindi scegli OK.

4. Configurazione della connessione ODBC di AD FS ad Athena

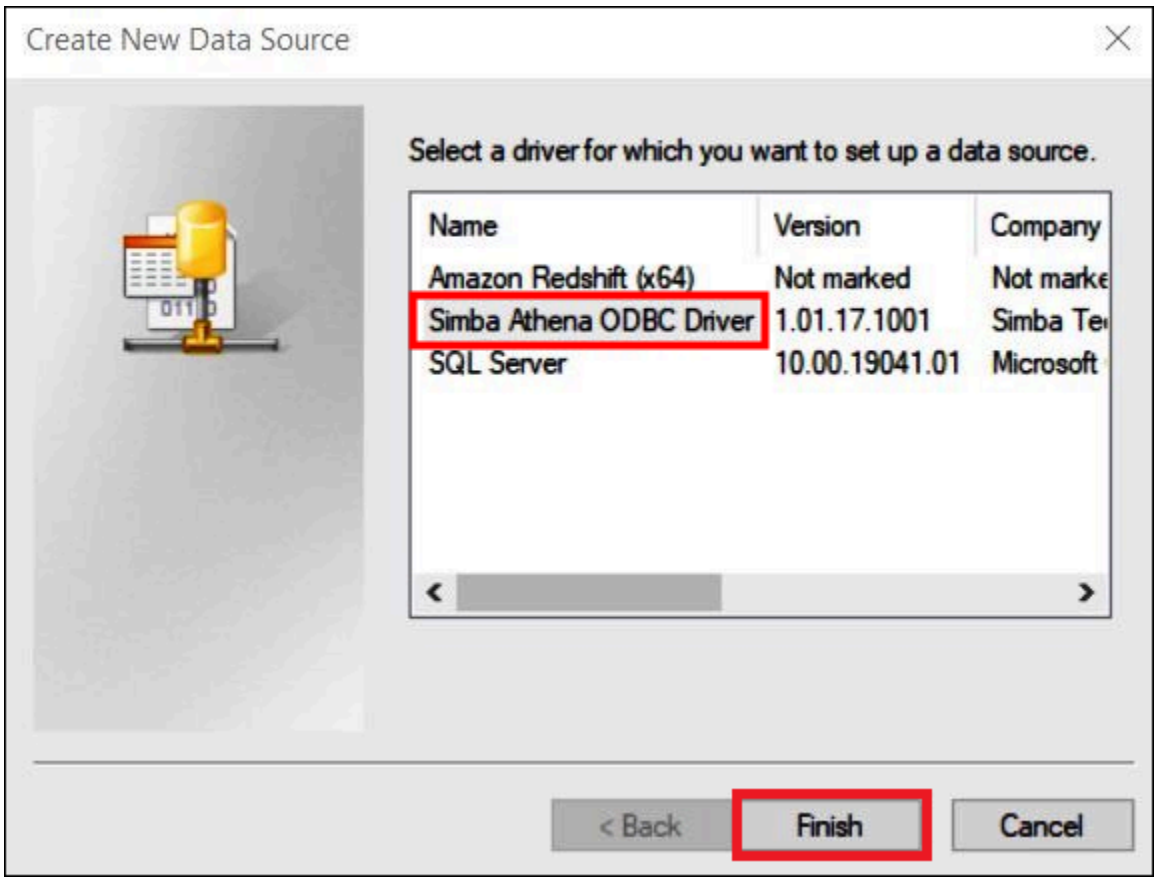
Dopo avere creato gli utenti e i gruppi AD, è possibile utilizzare il programma di origini dei dati ODBC in Windows per configurare la connessione ODBC di Athena per AD FS.

Configurazione della connessione ODBC di AD FS ad Athena

1. Installa il driver ODBC per Athena. Per i link di download, consulta la sezione Connessione ad Amazon Athena con ODBC.
2. In Windows, scegli Start, ODBC Data Sources (Origini dati ODBC).
3. Nel programma ODBC Data Source Administrator (Amministratore origine dati ODBC), scegli Add (Aggiungi).

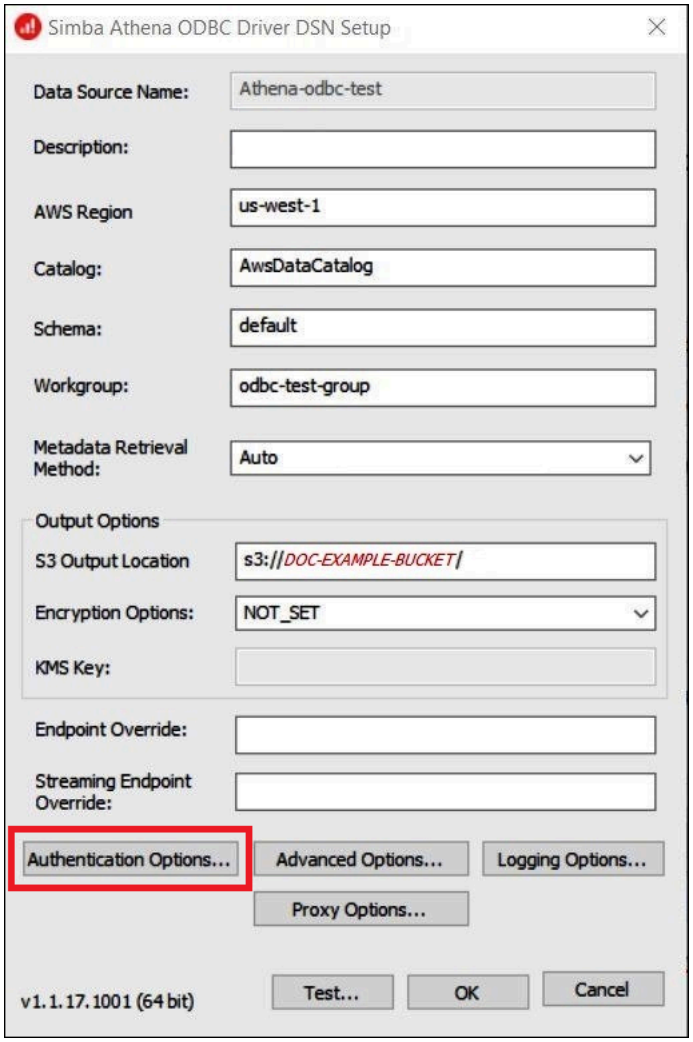


4. Nella finestra di dialogo Create New Data Source (Crea nuova origine dati), scegli Simba Athena ODBC Driver, quindi scegli Finish (Concludi).



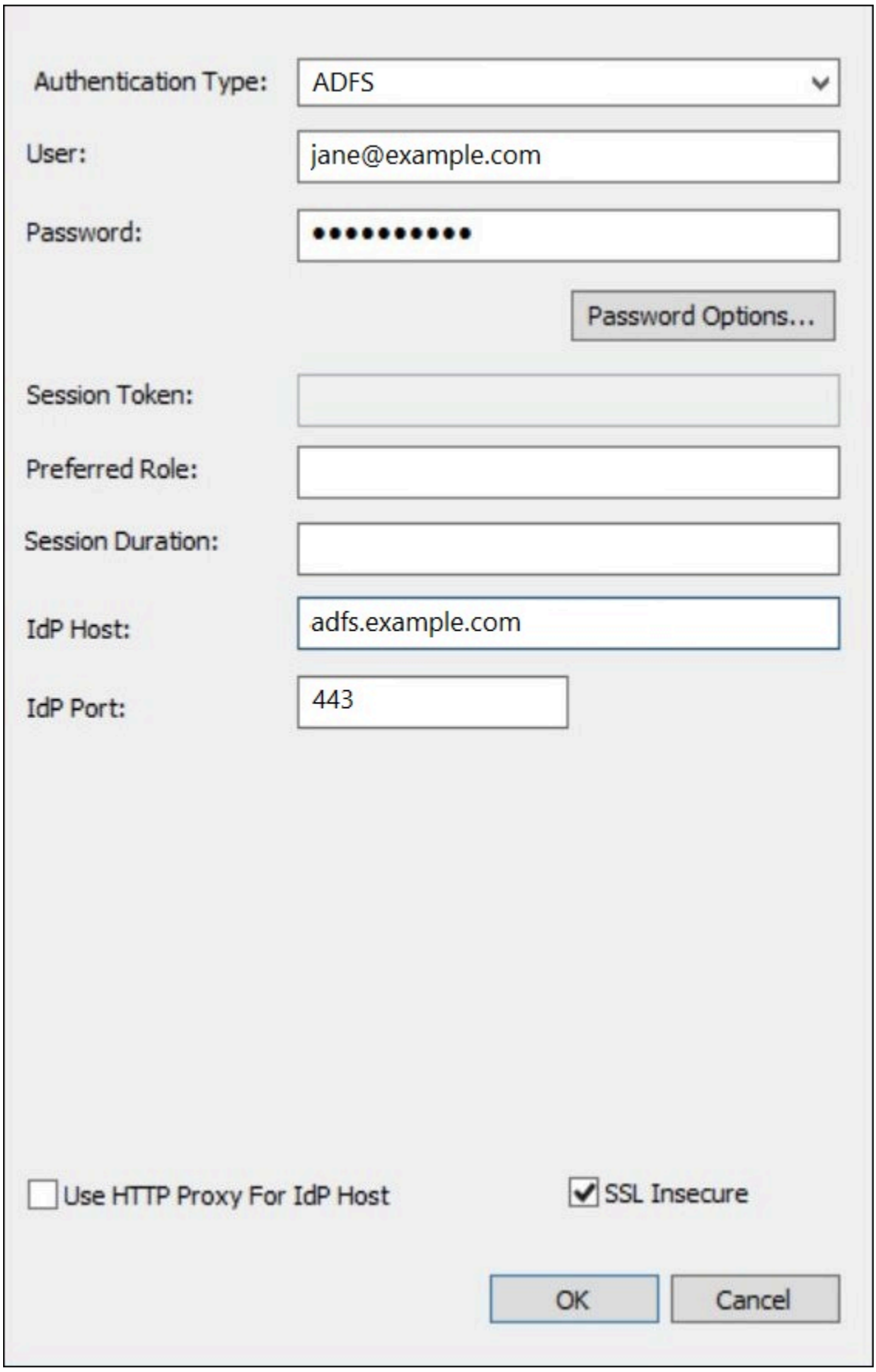
5. Nella finestra di dialogo Simba Athena ODBC Driver DSN Setup (Configurazione DSN driver ODBC Simba Athena), inserisci i valori seguenti:

   - In Data Source Name (Nome origine dei dati), inserisci un nome per l'origine dati (ad esempio, **`Athena-odbc-test`**).
   - In Description (Descrizione), inserisci una breve descrizione per l'origine dati.
   - Per Regione AWS inserisci la Regione AWS utilizzata (ad esempio, **`us-west-1`**).
   - Per S3 Output Location (Percorso di output S3), inserisci il percorso di Amazon S3 in cui archiviare l'output.



6. Scegli Authentication Options (Opzioni di autenticazione).
7. Nella finestra di dialogo Authentication Options (Opzioni di autenticazione), specifica i seguenti valori:

   - In Authentication Type (Tipo di autenticazione), scegli ADFS.
   - In User (Utente), inserisci l'indirizzo e-mail dell'utente (ad esempio, **`jane@example.com`**).
   - In Password, inserisci la password ADFS dell'utente.
   - In IdP Host (Host IdP), inserisci il nome del server AD FS (ad esempio, **`adfs.example.com`**).
   - In IdP Port (Porta IdP), utilizza il valore predefinito 443.
   - Seleziona l'opzione SSL Insecure.


Authentication Type:
ADFS
User:
jane@example.com
Password:
Password Options...
Session Token:
Preferred Role:
Session Duration:
IdP Host:
adfs.example.com
IdP Port:
443
Use HTTP Proxy For IdP Host
SSL Insecure
OK
Cancel

8. Scegli OK per chiudere la finestra Authentication Options (Opzioni di autenticazione).
9. Scegli Test per verificare la connessione oppure OK per terminare.

Configurazione di SSO per ODBC utilizzando il plug-in Okta e il gestore dell'identità digitale (IdP) Okta

Questa pagina descrive come configurare il driver ODBC di Amazon Athena e il plug-in Okta per aggiungere la funzionalità Single Sign-on (SSO) utilizzando il gestore dell'identità digitale (IdP) Okta.

Prerequisiti

Il completamento della procedura riportata in questo tutorial richiede quanto segue:

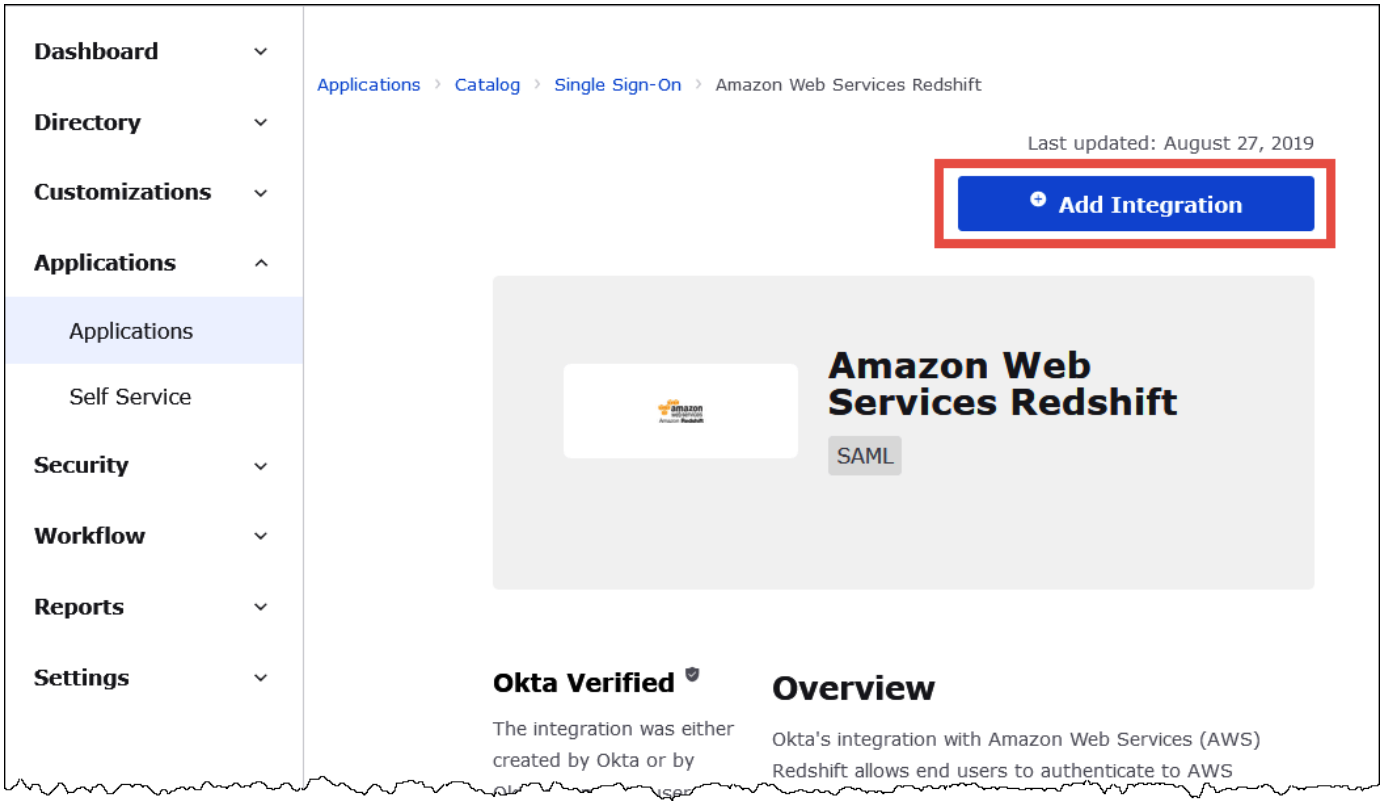
- Driver ODBC di Amazon Athena. Per i link di download, consulta la sezione Connessione ad Amazon Athena con ODBC.
- Un ruolo IAM che si desidera utilizzare con SAML. Per ulteriori informazioni, consulta Creazione di un ruolo per la federazione SAML 2.0 nella Guida per l'utente di IAM.
- Un account Okta. Per informazioni, visita il sito Okta.com.

Creazione di un'integrazione di app in Okta

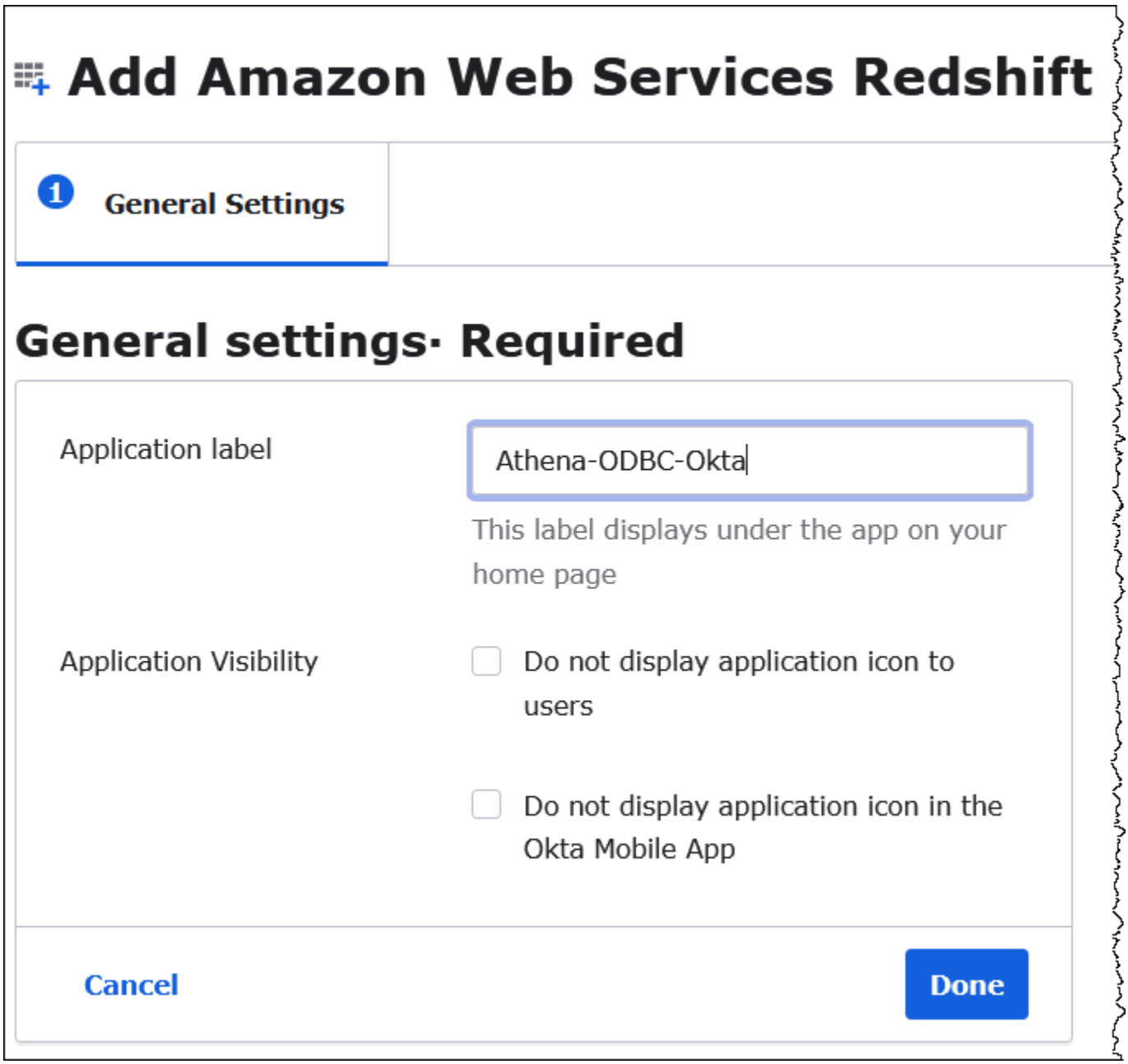
Innanzitutto, utilizza il pannello di controllo di Okta per creare e configurare un'app SAML 2.0 per il single sign-on ad Athena. Puoi utilizzare un'applicazione Redshift esistente in Okta per configurare l'accesso ad Athena.

Creazione di un'integrazione di app in Okta

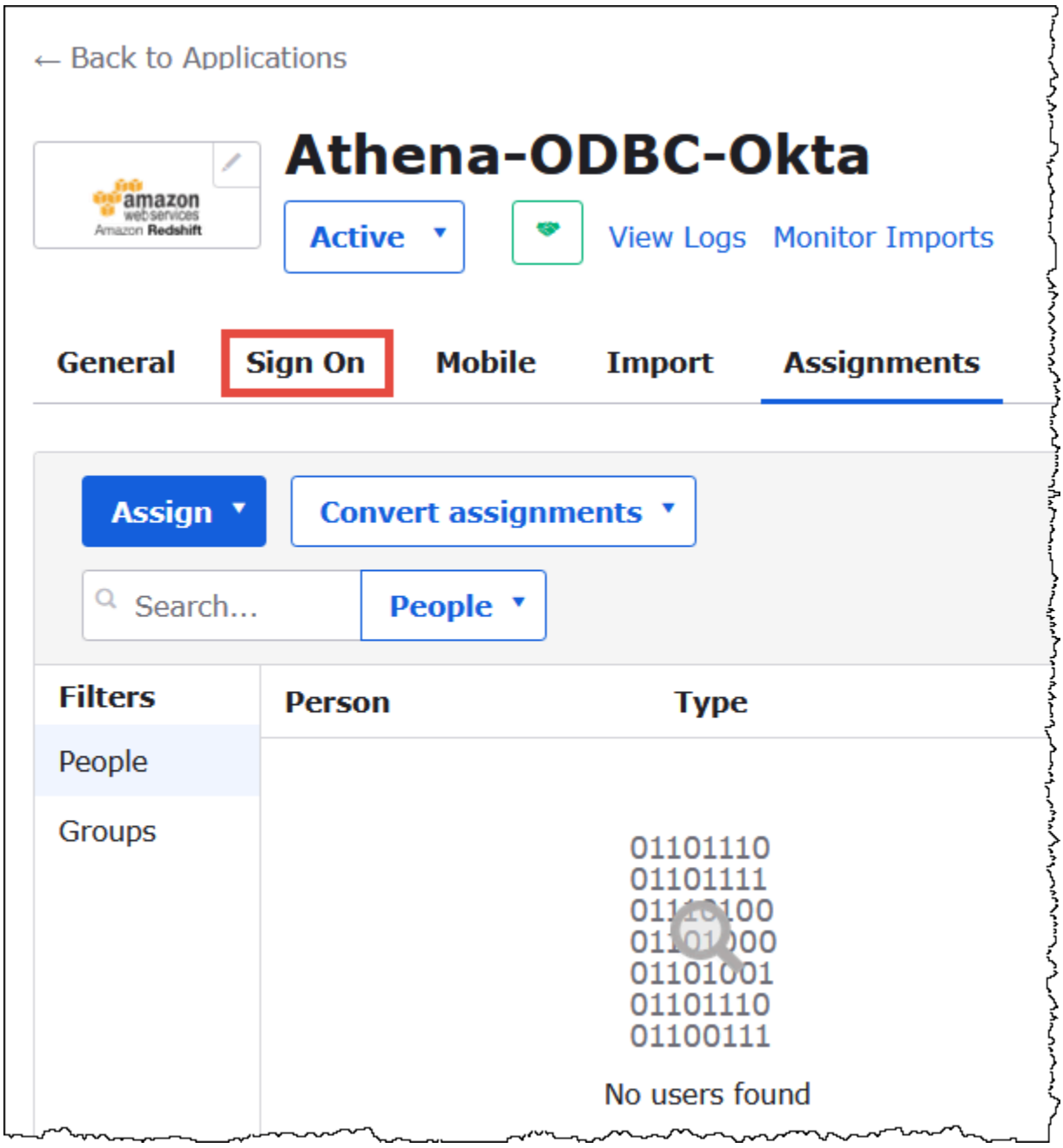
1. Accedi alla pagina di amministrazione del tuo account su Okta.com.
2. Nel riquadro di navigazione, scegli Applications (Applicazioni), quindi scegli Applications (Applicazioni).
3. Nella pagina Applications (Applicazioni), scegli Browse App Catalog (Sfoglia catalogo app).
4. Nella pagina Browse App Integration Catalog (Sfoglia catalogo di integrazione app), nella sezione Use Case (Caso d'uso), scegli All Integrations (Tutte le integrazioni).
5. Nella casella di ricerca, inserisci Amazon Web Services Redshift, quindi scegli Amazon Web Services Redshift SAML.
6. Scegli Add Integration (Aggiungi integrazione).



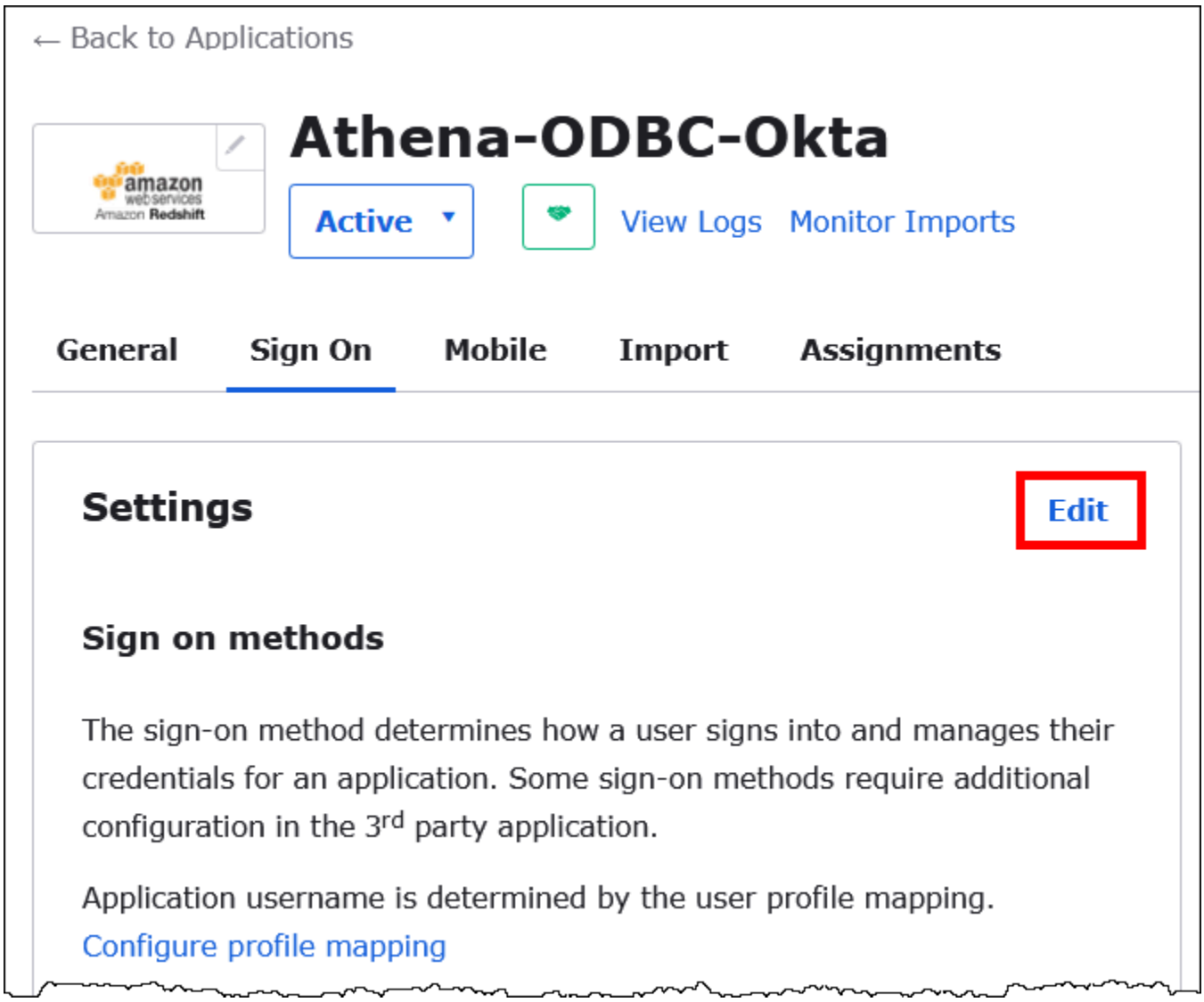
7. Nella sezione General Settings Required (Impostazioni generali obbligatorie), in Application label (Etichetta applicazione) inserisci un nome per l'applicazione. In questo tutorial si utilizza il nome Athena-ODBC-Okta.



8. Seleziona Fatto.
9. Nella pagina della tua applicazione Okta (ad esempio, Athena-ODBC-Okta), scegli Sign On (Accedi).



10. Nella sezione Settings (Impostazioni), scegli Edit (Modifica).



11. Nella sezione Advanced Sign-on Settings (Impostazioni di accesso avanzate), configura i valori seguenti.

    - Per IdP ARN e Role ARN, inserisci l'ARN IDP e l'ARN del ruolo come valori separati da AWS virgole. Per ulteriori informazioni sul formato del ruolo IAM, consulta Configurazione delle asserzioni SAML per la risposta di autenticazione nella Guida per l'utente di IAM.
    - In Session Duration (Durata della sessione), inserisci un valore compreso tra 900 e 43200 secondi. In questo tutorial si utilizza il valore predefinito 3600 (1 ora).

Advanced Sign-on Settings

These fields may be required for a Amazon Web Services Redshift proprietary sign-on option or general setting.

Idp ARN and Role ARN

arn:aws:iam::1234567890:saml-provid

Enter your AWS IDP ARN and Role ARN as comma separated values (e.g. "arn:aws:iam::1234567890:saml-provider/OKTA,arn:aws:iam::1234567890:role/SAML_ROLE").

Session Duration

3600

Set the user's session duration in seconds here.
Valid range is 900 to 43200.

DB User Format (Redshift)

${user.username}

EL expression to get DB User value (e.g. "${user.username}", "${user.firstName}${user.lastName}@acme.com")

Auto Create (Redshift)

AutoCreate Redshift property (Create a new database user if one does not exist)

Allowed DB Groups (Redshift)

Comma separated list of allowed user groups. Use "*" to allow all groups, "\," to escape comma in group name

Le impostazioni DbUser Format e Allowed DBGroups non vengono utilizzate da Athena. AutoCreate Non è necessario configurarle.

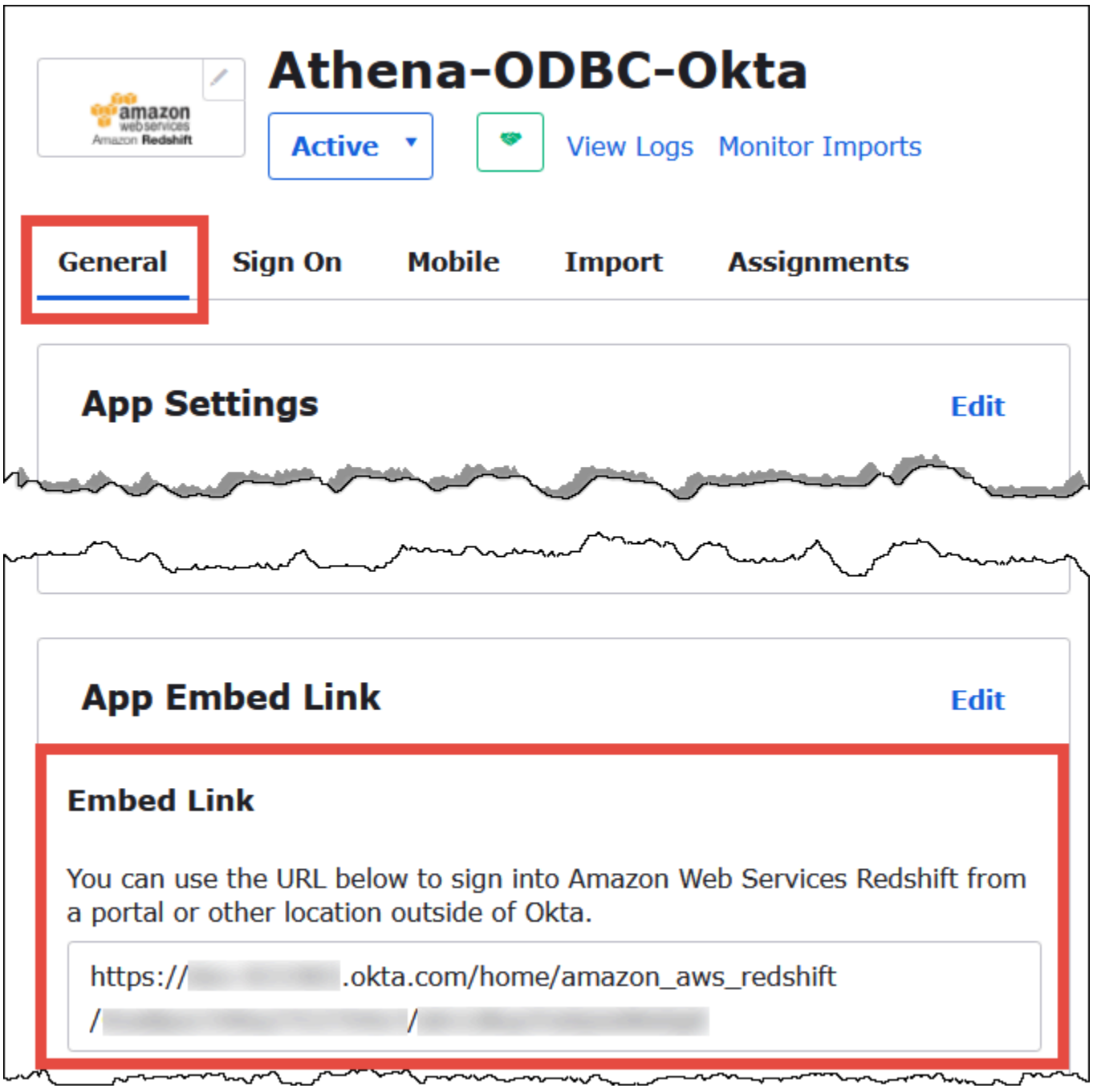
12. Selezionare Salva.

Recupero delle informazioni di configurazione ODBC da Okta

Ora che hai creato l'applicazione Okta, puoi recuperare l'ID dell'applicazione e l'URL dell'host IdP. Ti occorreranno in seguito per configurare ODBC per la connessione ad Athena.

Recupero delle informazioni di configurazione ODBC da Okta

1. Scegli la scheda General (Generali) dell'applicazione Okta, quindi scorri verso il basso fino alla sezione App Embed Link (Link di incorporamento app).



L'URL dell'Embed Link (Link di incorporamento) si presenta nel formato seguente:

```
https://trial-1234567.okta.com/home/amazon_aws_redshift/Abc1de2fghi3J45kL678/
abc1defghij2klmNo3p4
```

2. Dall'URL dell'Embed Link (Link di incorporamento), estrai e salva i seguenti elementi:

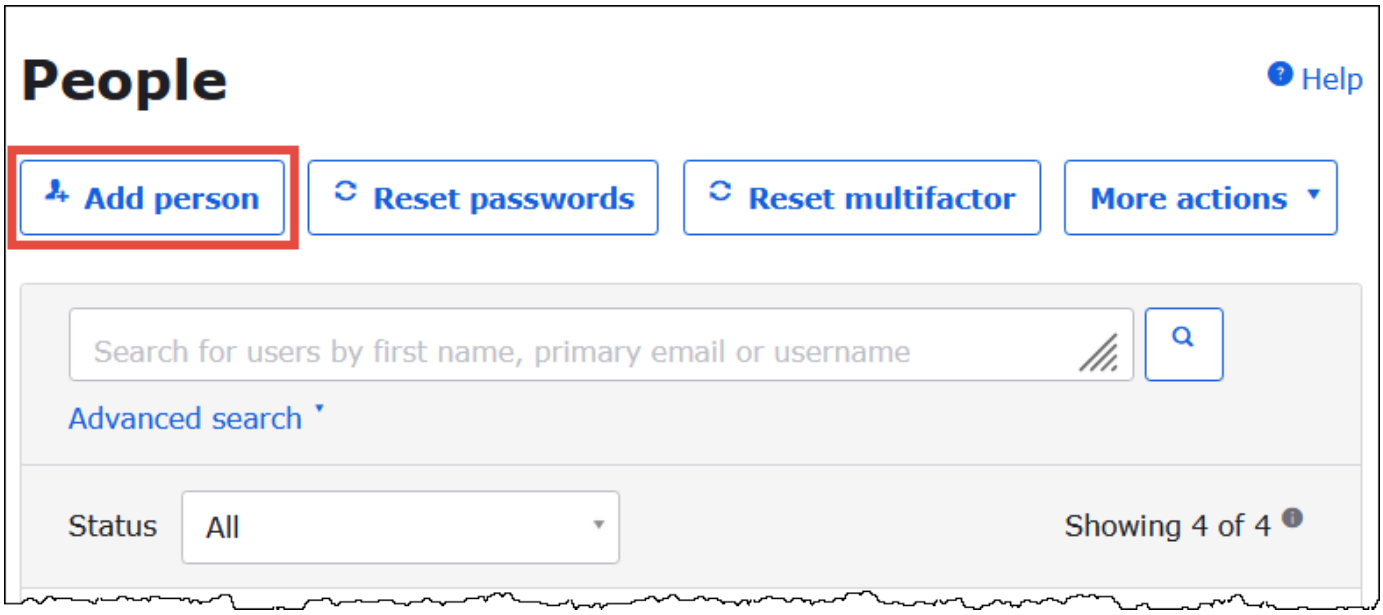
- Il primo segmento dopo `https://`, fino a `okta.com` incluso (ad esempio trial-1234567.okta.com). Questo è l'host del tuo IdP.
- Gli ultimi due segmenti dell'URL, inclusa la barra obliqua in mezzo. I segmenti sono due stringhe di 20 caratteri con un mix di numeri e lettere maiuscole e minuscole (ad esempio Abc1de2fghi3J45kL678/abc1defghij2klmNo3p4). Questo è l'ID della tua applicazione.

Aggiunta di un utente all'applicazione Okta

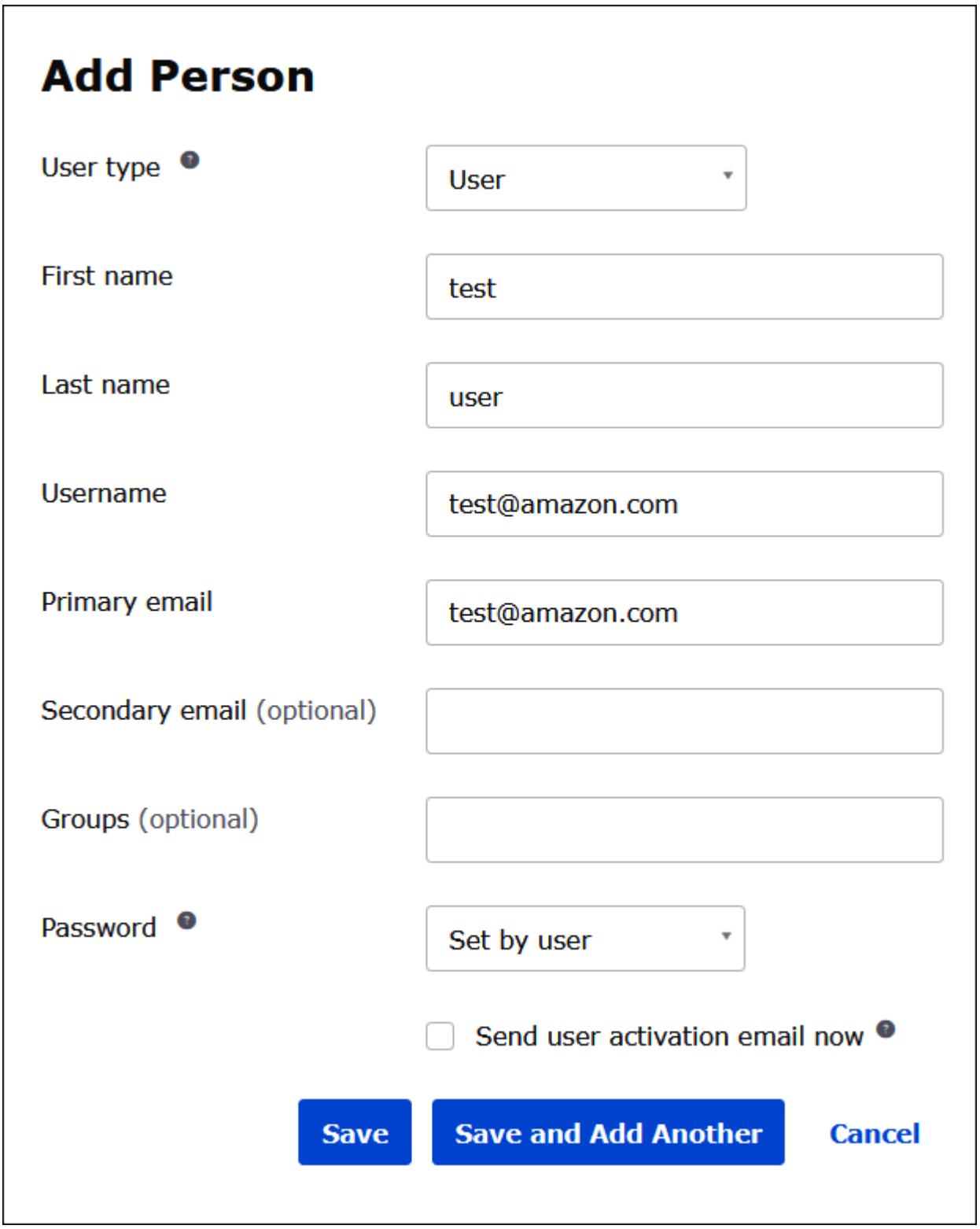
Ora puoi aggiungere un utente all'applicazione Okta.

Aggiunta di un utente all'applicazione Okta

1. Nel pannello di navigazione a sinistra selezionare Directory e poi Persone.
2. Scegli Add person (Aggiungi persona).



3. Nella finestra di dialogo Add Person (Aggiungi persona), inserisci le informazioni seguenti.
   - Inserisci i valori per Nome e Cognome. In questo tutorial si utilizza **`test user`**.
   - Specifica i valori Username (Nome utente) e Primary email (Indirizzo e-mail principale). In questo tutorial si utilizza **`test@amazon.com`** per entrambi. I requisiti di sicurezza per le password potrebbero variare.

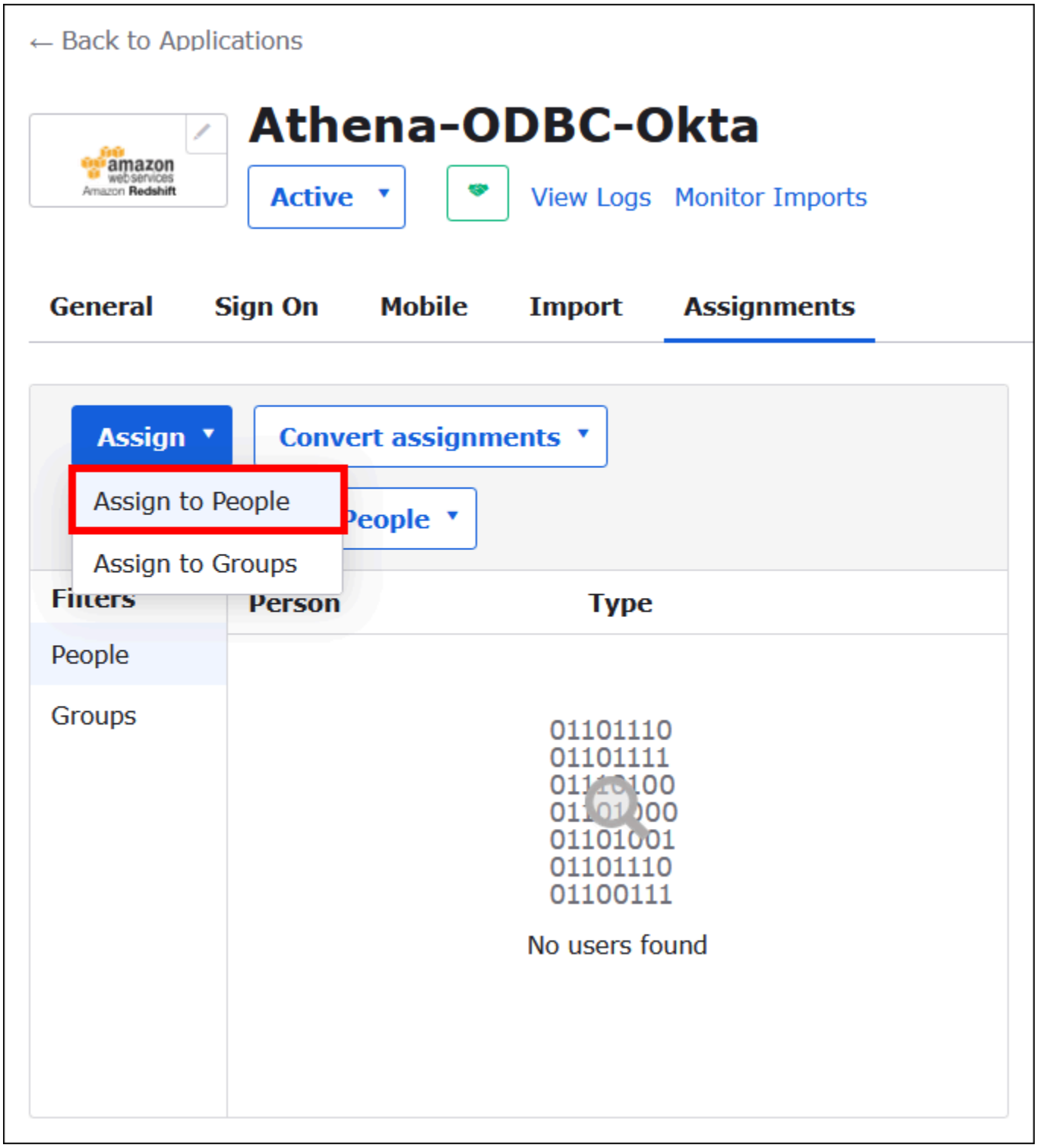


4. Selezionare Salva.

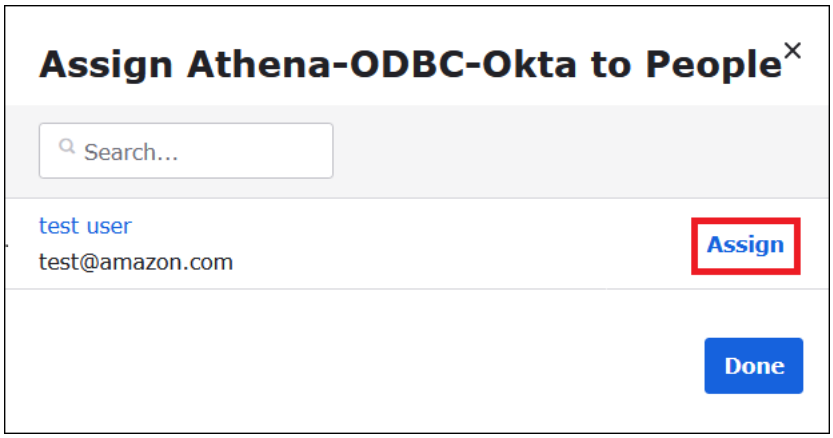
Ora puoi assegnare l'utente creato all'applicazione.

Per assegnare l'utente all'applicazione:

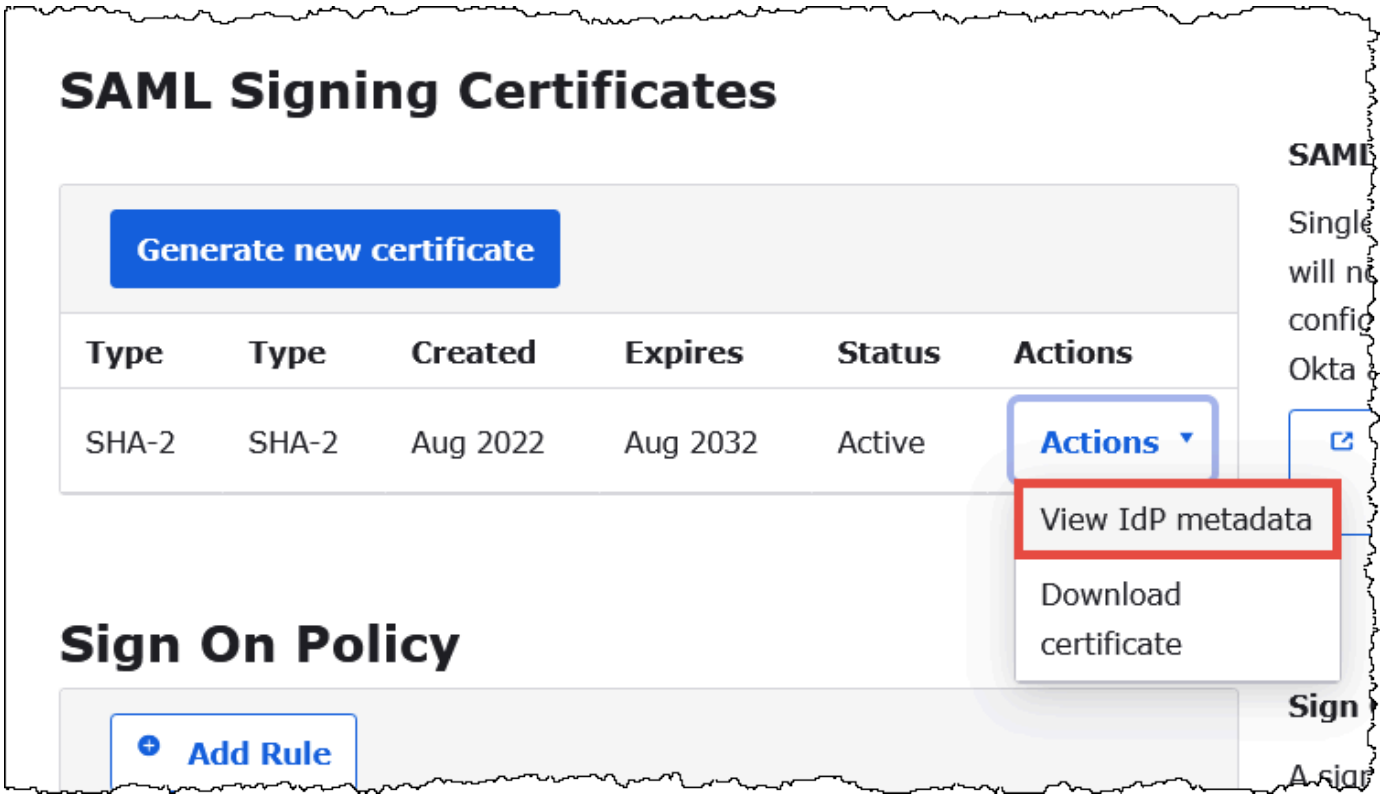
1. Nel riquadro di navigazione, scegli Applications (Applicazioni) e di nuovo Applications (Applicazioni), quindi scegli il nome dell'applicazione (ad esempio Athena-ODBC-Okta).
2. Scegli Assign (Assegna), quindi scegli Assign to People (Assegna a persone).



3. Scegli l'opzione Assign (Assegna) per il tuo utente, quindi scegli Done (Fatto).



4. Quando compare la richiesta, scegli Save and Go Back (Salva e torna indietro). La finestra di dialogo mostra lo stato dell'utente come Assigned (Assegnato).
5. Seleziona Fatto.
6. Scegli la scheda Sign On (Accedi).
7. Scorri fino alla sezione SAML Signing Certificates (Certificati di firma SAML).
8. Scegli Azioni.
9. Apri il menu contestuale (tasto destro del mouse), seleziona View IdP metadata (Visualizza metadati IdP), quindi scegli l'opzione browser per salvare il file.
10. Salva il file con l'estensione `.xml`.

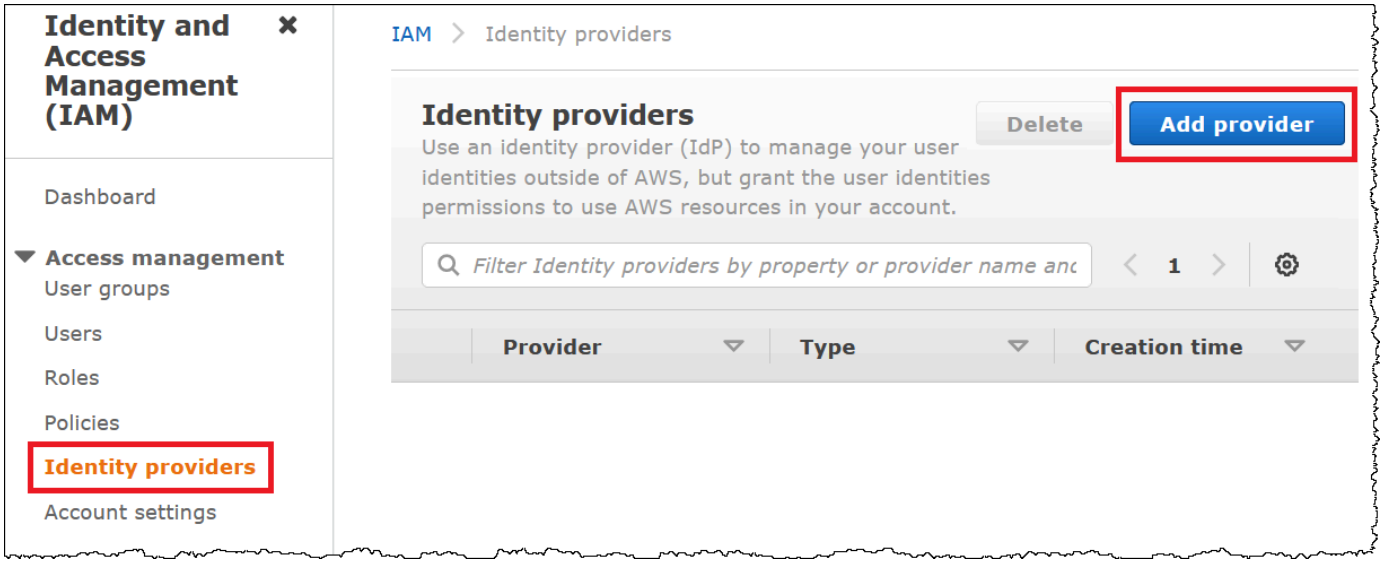


Crea un provider di identità e un AWS ruolo SAML

Ora sei pronto per caricare il file XML dei metadati sulla console IAM in. AWS Utilizzerai questo file per creare un provider di identità e un ruolo AWS SAML. Utilizza un account amministratore dei servizi AWS per eseguire questi passaggi.

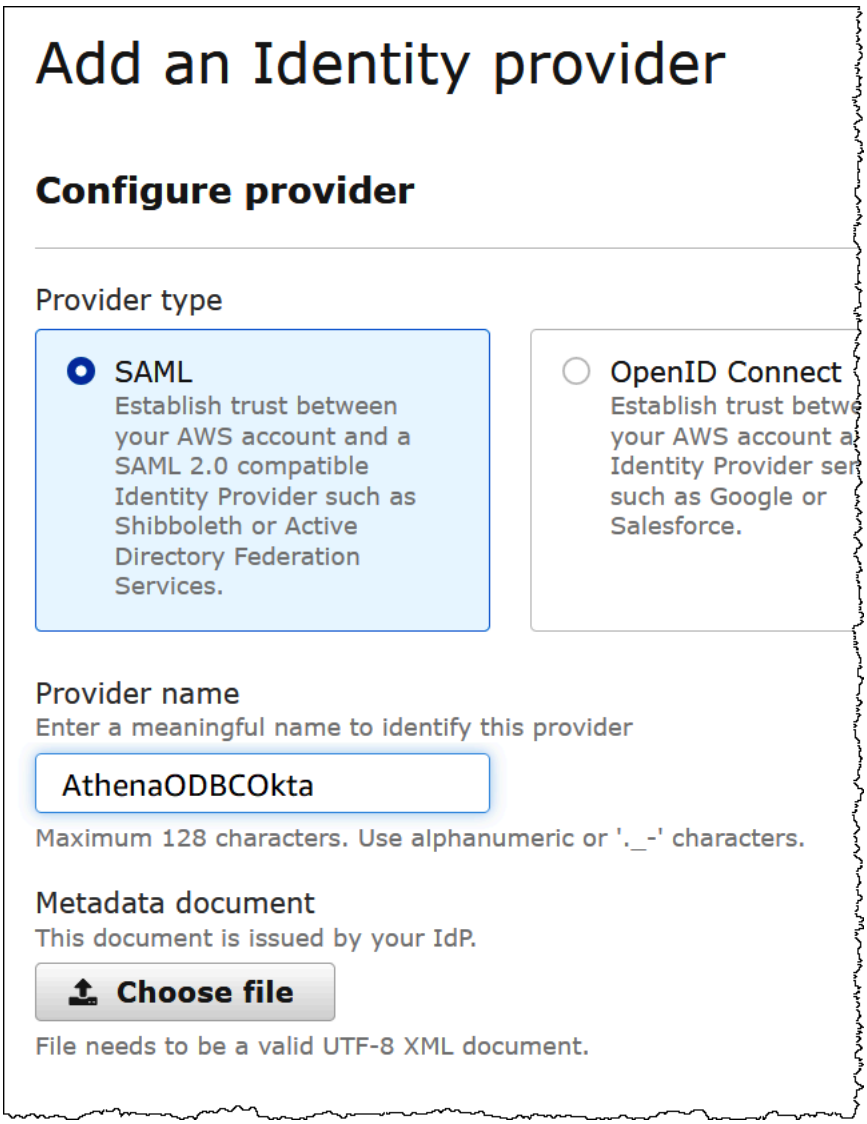
Per creare un provider di identità SAML e un ruolo in AWS

1. Accedi AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/IAM/.
2. Nel riquadro di navigazione, scegli Provider di identità, quindi seleziona Aggiungi provider.



3. Nella pagina Add an Identity provider (Aggiungi un gestore dell'identità digitale), in Configure provider (Configura gestore) inserisci le seguenti informazioni.

   - Per Tipo di provider, scegliere SAML.
   - Per Provider name (Nome del gestore), inserisci un nome per il gestore (ad esempio, **`AthenaODBCOkta`**).
   - Per Documento dei metadati, utilizzare l'opzione Scegli file per caricare il file XML dei metadati del provider di identità (IdP) scaricato.



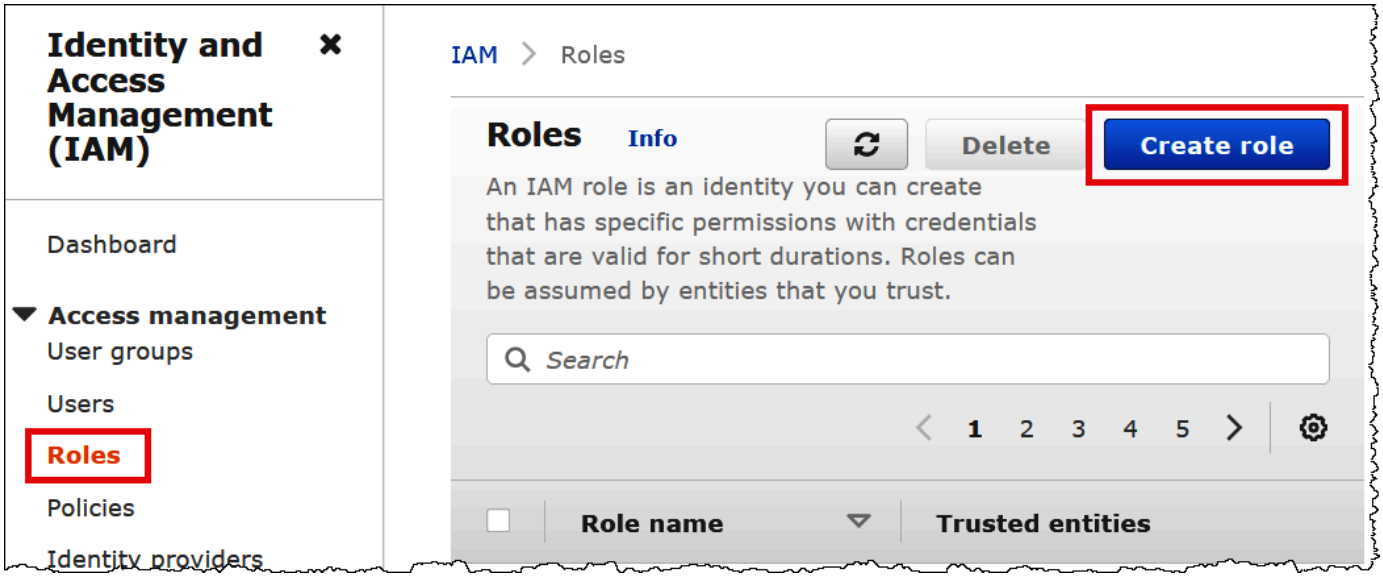
4. Scegli Aggiungi provider.

Creazione di un ruolo IAM per l'accesso ad Athena e Amazon S3

Ora puoi creare un ruolo IAM per l'accesso ad Athena e Amazon S3. Assegnerai questo ruolo al tuo utente. In questo modo, puoi fornire all'utente l'accesso Single Sign-On ad Athena.

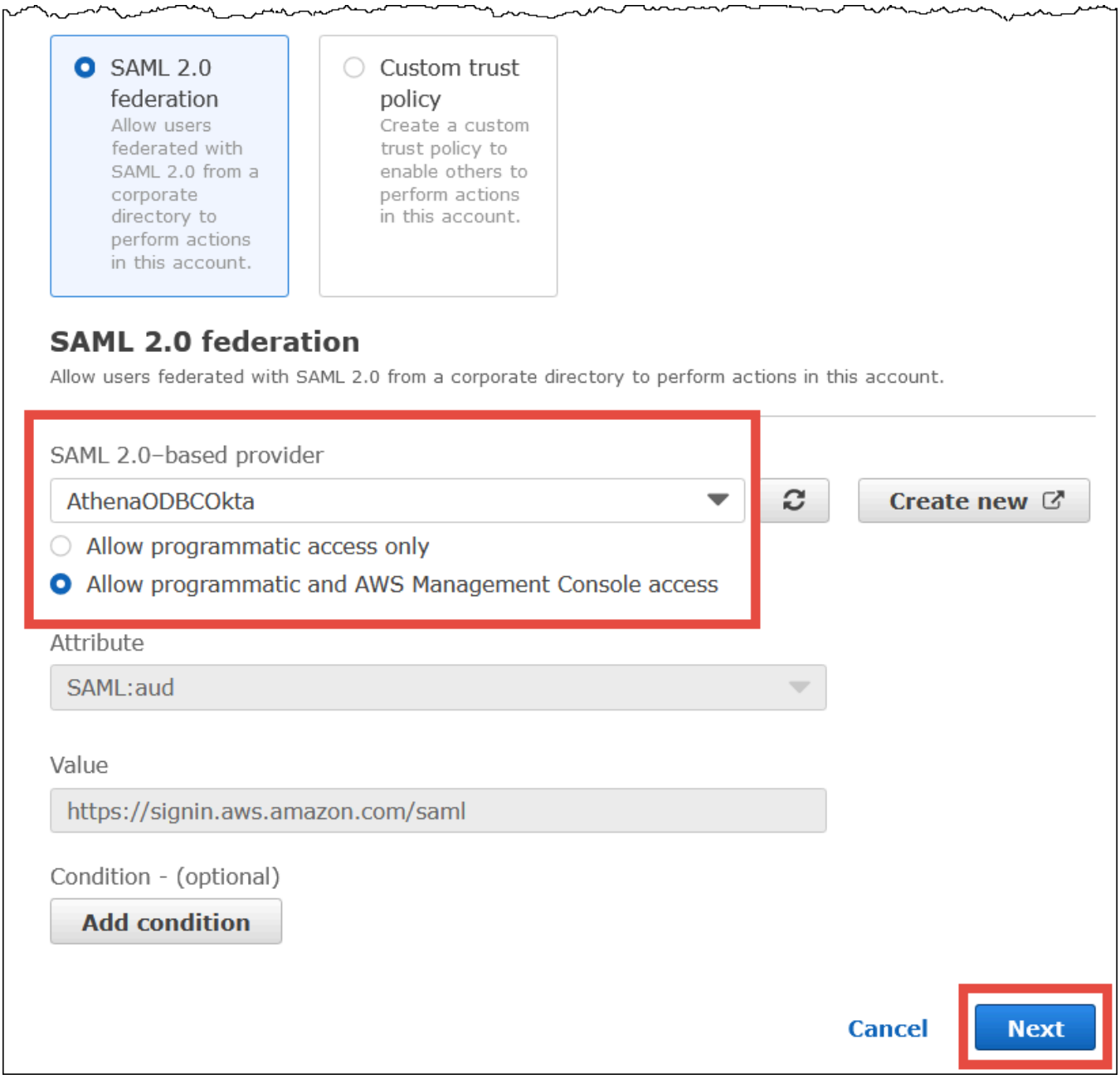
Creazione di un ruolo IAM per l'utente

1. Nel pannello di navigazione della console IAM, scegliere Ruoli e quindi Crea ruolo.

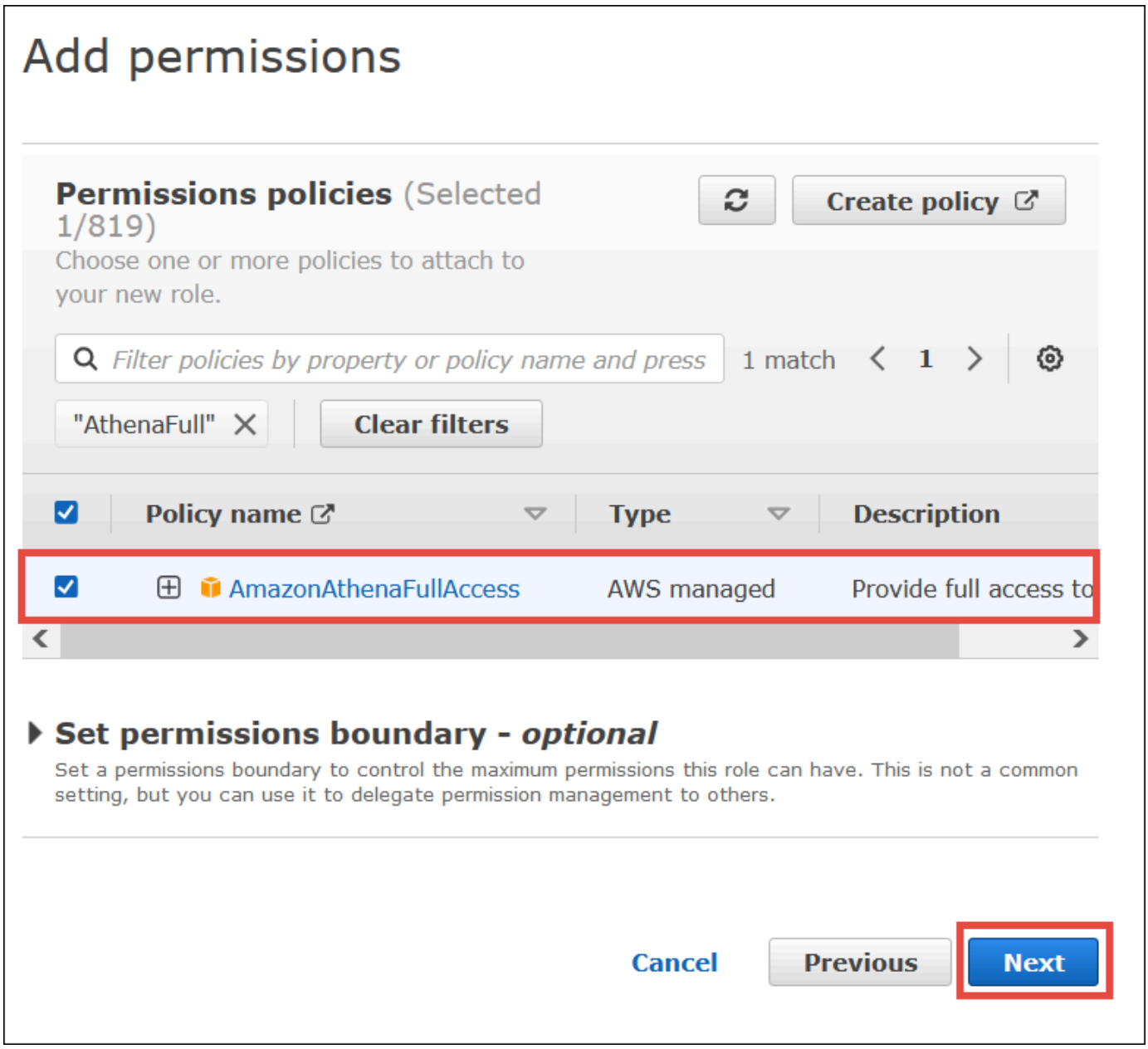


2. Nella pagina Create role (Crea ruolo), scegli le opzioni seguenti:

   - Per Seleziona tipo di entità attendibile, scegliere SAML 2.0 Federation.
   - Per SAML 2.0–based provider (Gestore basato su SAML 2.0), scegli il gestore dell'identità digitale SAML creato (ad esempio, AthenaODBCOkta).
   - Seleziona Allow programmatic and AWS Management Console access.



3. Seleziona Successivo.
4. Nella pagina Add Permissions (Aggiungi autorizzazioni), per Filter policies (Filtra le policy), inserisci **`AthenaFull`** e premi INVIO.
5. Seleziona la policy gestita `AmazonAthenaFullAccess`, quindi scegli Next (Successivo).



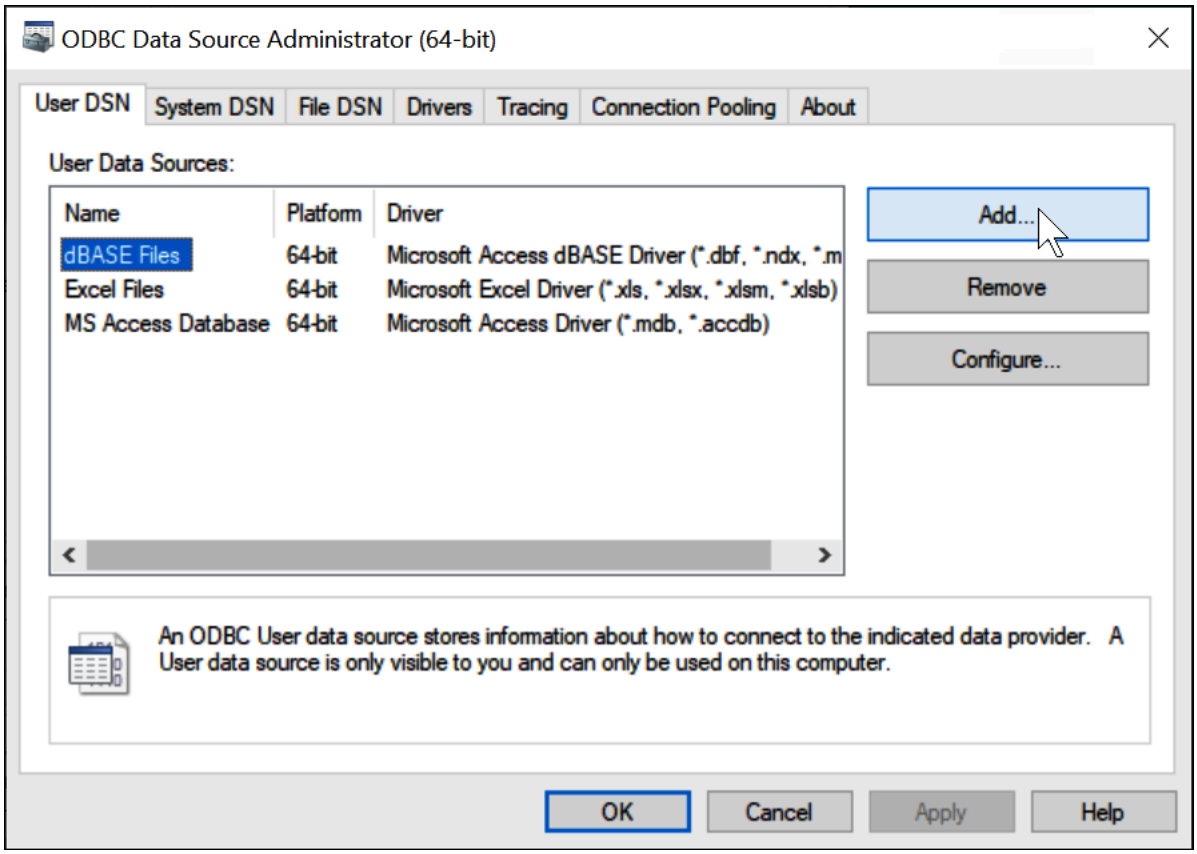
6. Nella pagina Name, review and create (Denomina, verifica e crea), in Role name (Nome ruolo) inserisci un nome per il ruolo (ad esempio, **`Athena-ODBC-OktaRole`**), quindi scegli Create role (Crea ruolo).

Configurazione della connessione ODBC di Okta ad Athena

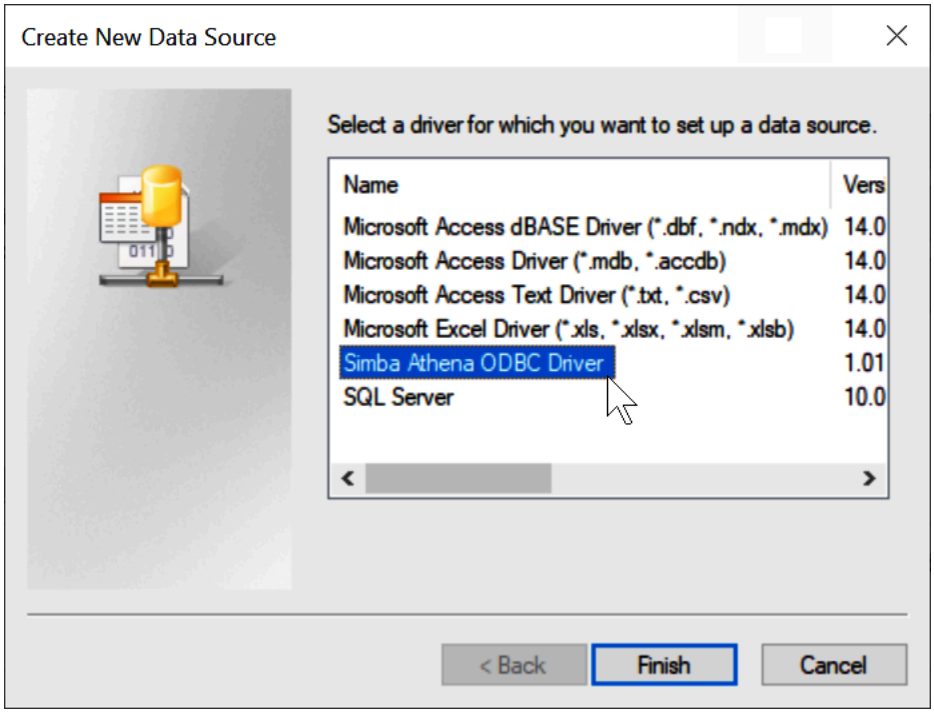
Ora puoi configurare la connessione ODBC di Okta ad Athena utilizzando il programma di origini dei dati ODBC in Windows.

Configurazione della connessione ODBC di Okta ad Athena

1. In Windows, avvia il programma ODBC Data Sources (Origini dati ODBC).
2. Nel programma ODBC Data Source Administrator (Amministratore origine dati ODBC), scegli Add (Aggiungi).

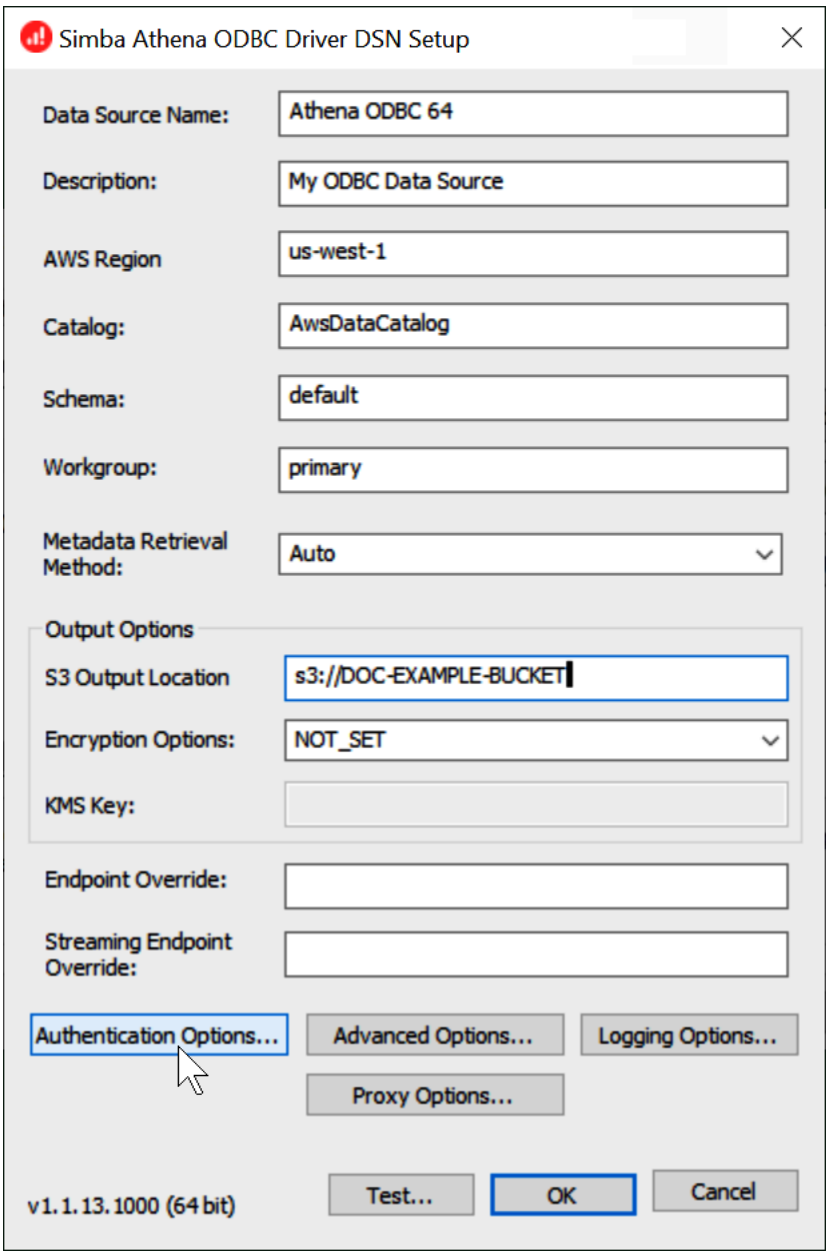


3. Scegli Simba Athena ODBC Driver (Driver ODBC Simba Athena), quindi scegli Finish (Fine).



4. Nella finestra di dialogo Simba Athena ODBC Driver DSN Setup (Configurazione DSN driver ODBC Simba Athena), inserisci i valori descritti.

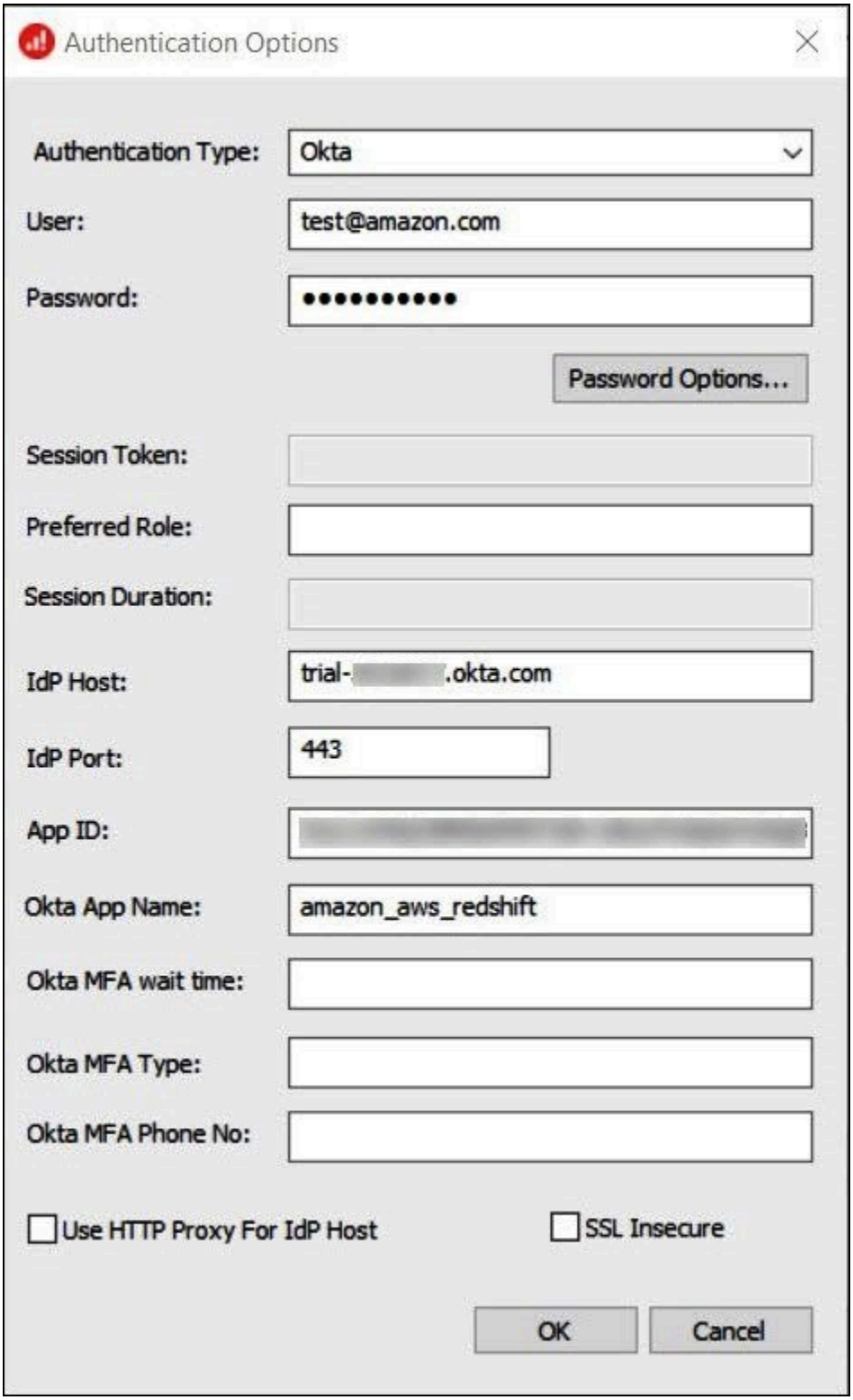
   - In Data Source Name (Nome origine dei dati), inserisci un nome per l'origine dati (ad esempio, **`Athena ODBC 64`**).
   - In Description (Descrizione), inserisci una breve descrizione per l'origine dati.
   - Per Regione AWS, inserisci quello Regione AWS che stai utilizzando (ad esempio,**`us-west-1`**).
   - Per S3 Output Location (Percorso di output S3), inserisci il percorso di Amazon S3 in cui archiviare l'output.



5. Scegli Authentication Options (Opzioni di autenticazione).
6. Nella finestra di dialogo Authentication Options (Opzioni di autenticazione), scegli o inserisci i seguenti valori.

   - Per Authentication Type (Tipo di autenticazione), scegli Okta.
   - Per User (Utente), immettere il nome utente Okta.
   - Per Password, immettere la password Okta.
   - In IdP Host (Host IdP), inserisci il valore registrato in precedenza (ad esempio, **`trial-1234567.okta.com`**).

- In IdP Port (Porta IdP), inserisci **`443`**.
- In App ID (ID app), inserisci il valore registrato in precedenza (gli ultimi due segmenti del link di incorporamento Okta).
- In Okta App Name (Nome app Okta), inserisci **`amazon_aws_redshift`**.



7. Scegli OK.
8. Scegli Test per verificare la connessione oppure OK per terminare.

Configurazione di Single Sign-On tramite ODBC, SAML 2.0 e il provider di identità Okta

Per connetterti alle fonti di dati, puoi utilizzare Amazon Athena con provider di identità (IdPs) come PingOne Okta e altri. OneLogin A partire dal driver Athena ODBC versione 1.1.13 e dal driver Athena JDBC versione 2.0.25, è incluso un plug-in SAML per browser che è possibile configurare in modo che funzioni con qualsiasi provider SAML 2.0. In questo argomento viene illustrato come configurare il driver ODBC di Amazon Athena e il plug-in SAML basato su browser per aggiungere la funzionalità Single Sign-on (SSO) utilizzando il provider di identità Okta.

Prerequisiti

Il completamento della procedura riportata in questo tutorial richiede quanto segue:

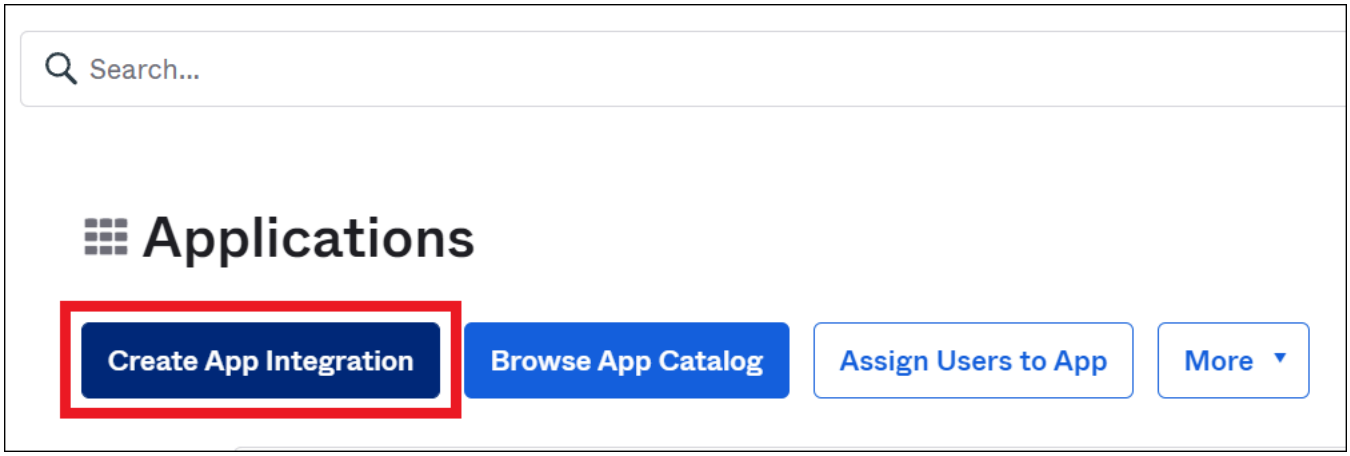
- Driver ODBC Athena versione 1.1.13 o successiva. Le versioni 1.1.13 e successive includono il supporto SAML del browser. Per i collegamenti per il download, consultare Connessione ad Amazon Athena con ODBC.
- Un ruolo IAM che si desidera utilizzare con SAML. Per ulteriori informazioni, consulta Creazione di un ruolo per la federazione SAML 2.0 nella Guida per l'utente di IAM.
- Un account Okta. Per informazioni, visita il sito okta.com.

Creazione di un'integrazione di app in Okta

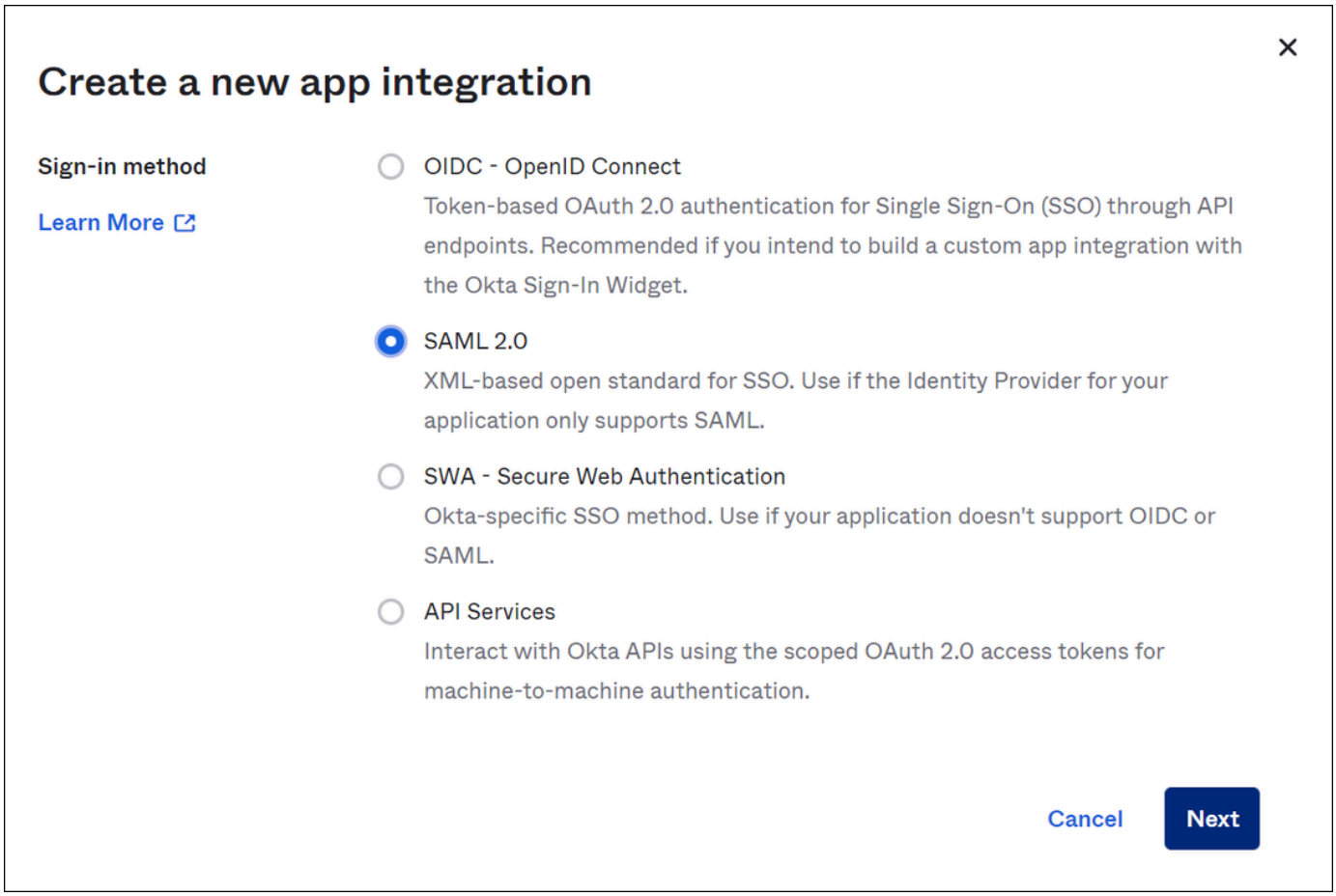
Innanzitutto, utilizza il pannello di controllo di Okta per creare e configurare un'app SAML 2.0 per il single sign-on ad Athena.

Come utilizzare il pannello di controllo di Okta per configurare il single sign-on per Athena

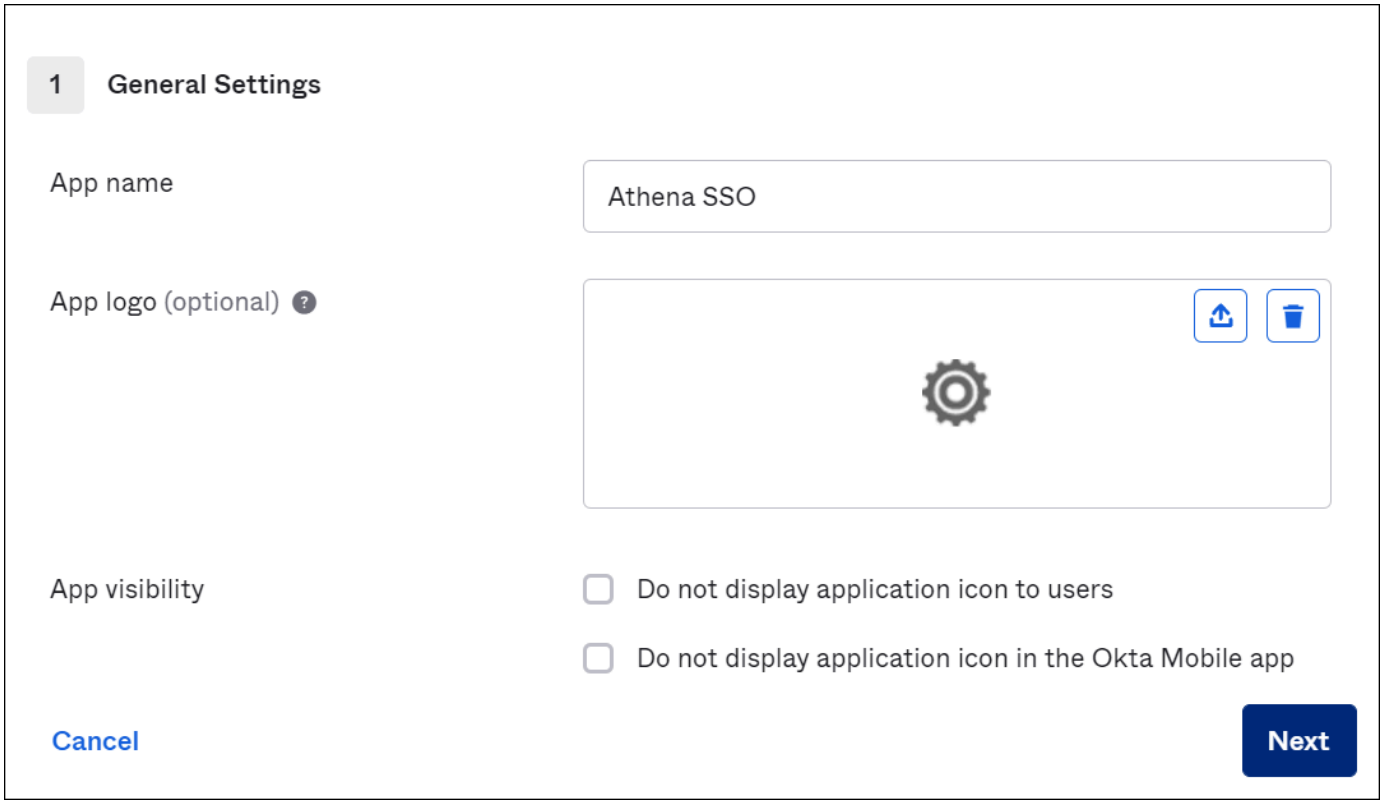
1. Accedi alla pagina di amministrazione di Okta su `okta.com`.
2. Nel pannello di navigazione, scegli Applications (Applicazioni), Applications (Applicazioni).
3. Nella pagina Applications (Applicazioni), scegli Create App Integration (Crea integrazione app).



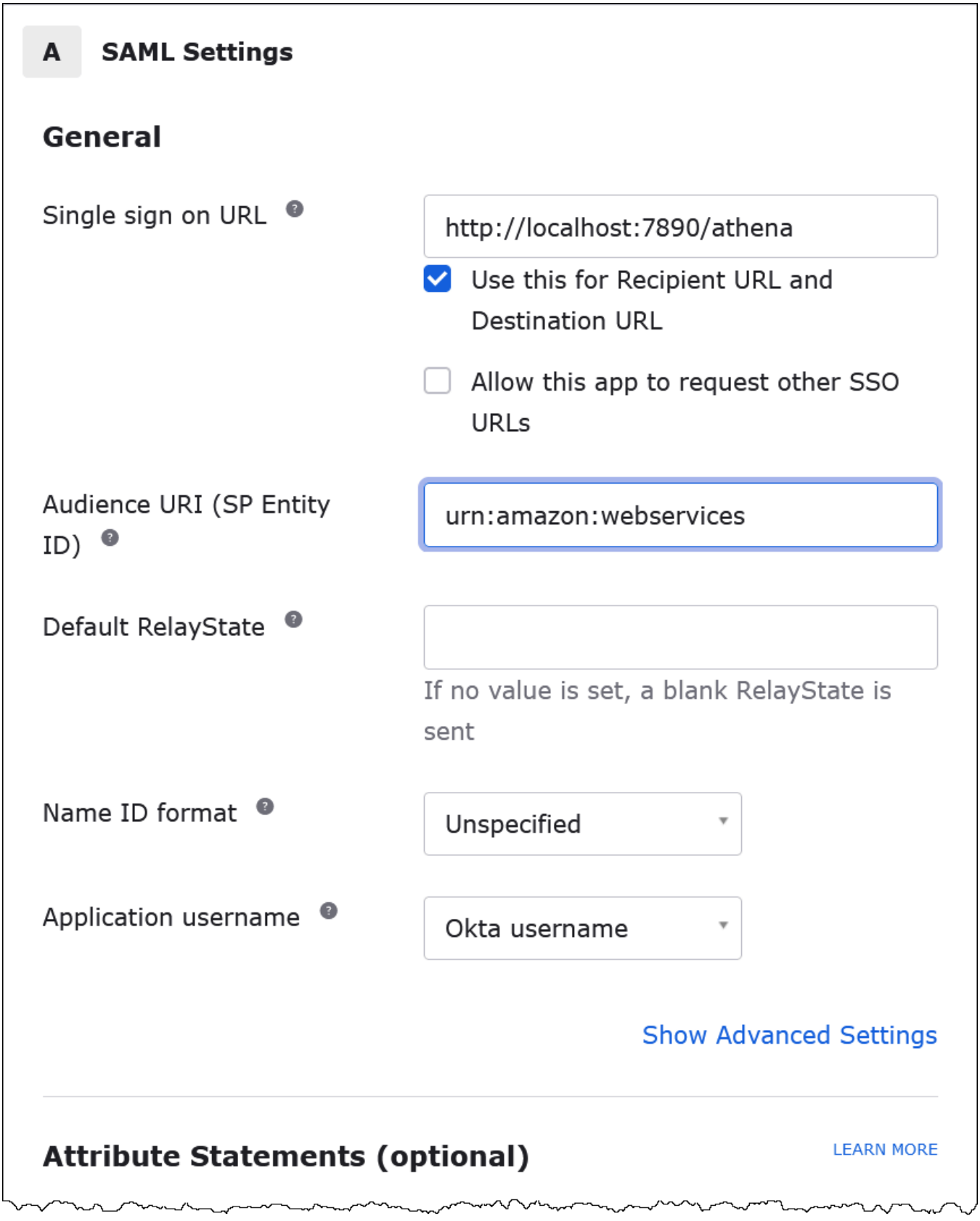
4. Nella finestra di dialogo Create a new app integration (Crea una nuova integrazione app), per Sign-in method (Metodo di accesso), seleziona SAML 2.0 e quindi scegli Next (Avanti).



5. Nella pagina Create SAML Integration (Crea integrazione SAML), nella sezione General Settings (Impostazioni generali), inserisci un nome per l'applicazione. In questo tutorial viene utilizzato il nome Athena SSO.



6. Seleziona Successivo.
7. Sulla pagina Configure SAML (Configura SAML), nella sezione SAML Settings (Impostazioni SAML), inserisci i seguenti valori:
   - Per Single sign on URL (URL Single Sign On), inserisci **`http://localhost:7890/athena`**
   - Per Audience URI (URI del pubblico), inserisci **`urn:amazon:webservices`**


A
SAML Settings
General
Single sign on URL
http://localhost:7890/athena
Use this for Recipient URL and Destination URL
Allow this app to request other SSO URLs
Audience URI (SP Entity ID)
urn:amazon:webservices
Default RelayState
If no value is set, a blank RelayState is sent
Name ID format
Unspecified
Application username
Okta username
Show Advanced Settings
Attribute Statements (optional)
LEARN MORE

8. Per Attribute Statements (optional) (Istruzioni attributi (opzionali)), inserisci le seguenti due coppie nome/valore. Questi sono attributi di mappatura obbligatori.

   - Per Name (Nome), inserisci il seguente URL:

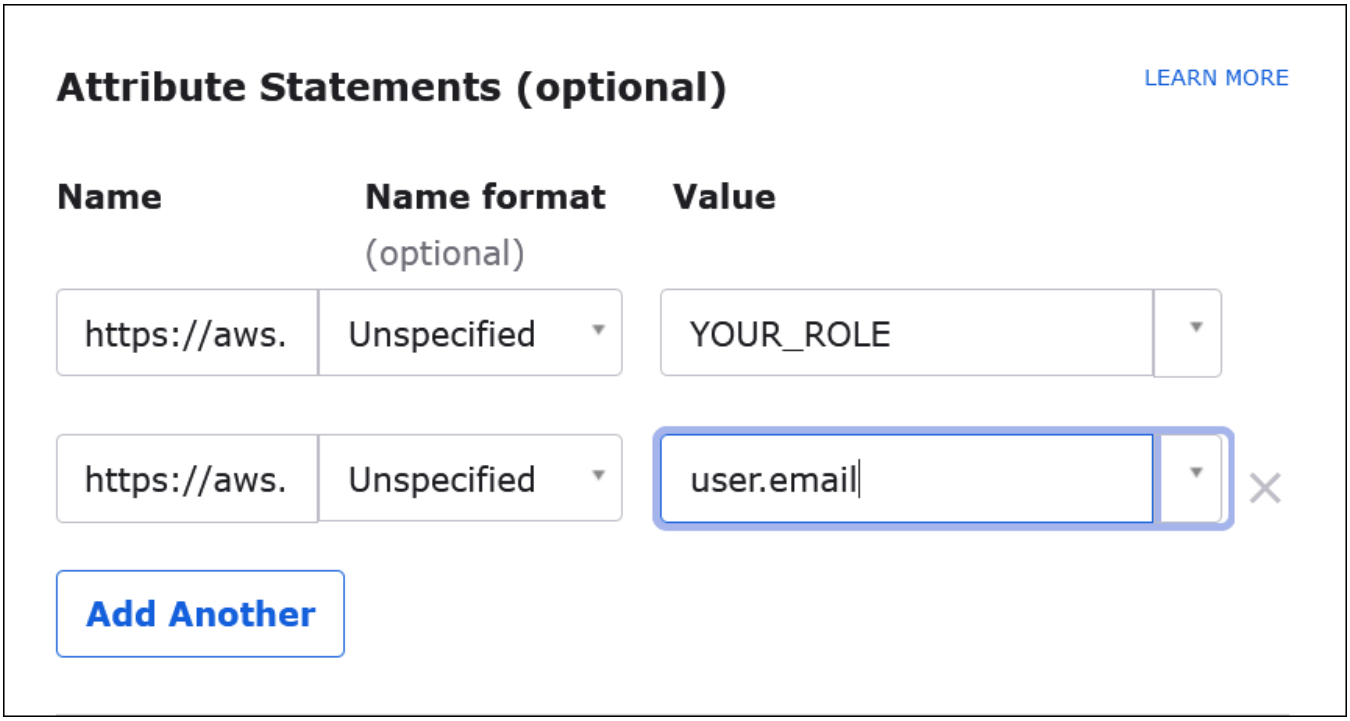
     `https://aws.amazon.com/SAML/Attributes/Role`

     Per Value (Valore), inserisci il nome del ruolo IAM. Per ulteriori informazioni sul formato del ruolo IAM, consulta Configurazione delle asserzioni SAML per la risposta di autenticazione nella Guida per l'utente di IAM.

   - Per Name (Nome), inserisci il seguente URL:

     `https://aws.amazon.com/SAML/Attributes/RoleSessionName`

     In Valore, specifica **`user.email`**.



9. Scegli Next (Avanti), quindi scegli Finish (Fine).

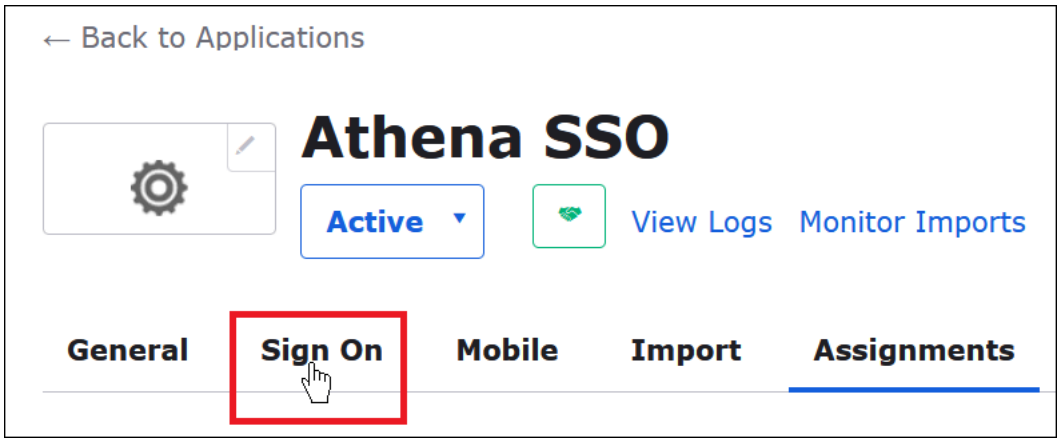
   Quando Okta crea l'applicazione, crea anche l'URL di accesso, che verrà recuperato successivamente.

Ottenimento dell'URL di accesso dal pannello di controllo di Okta

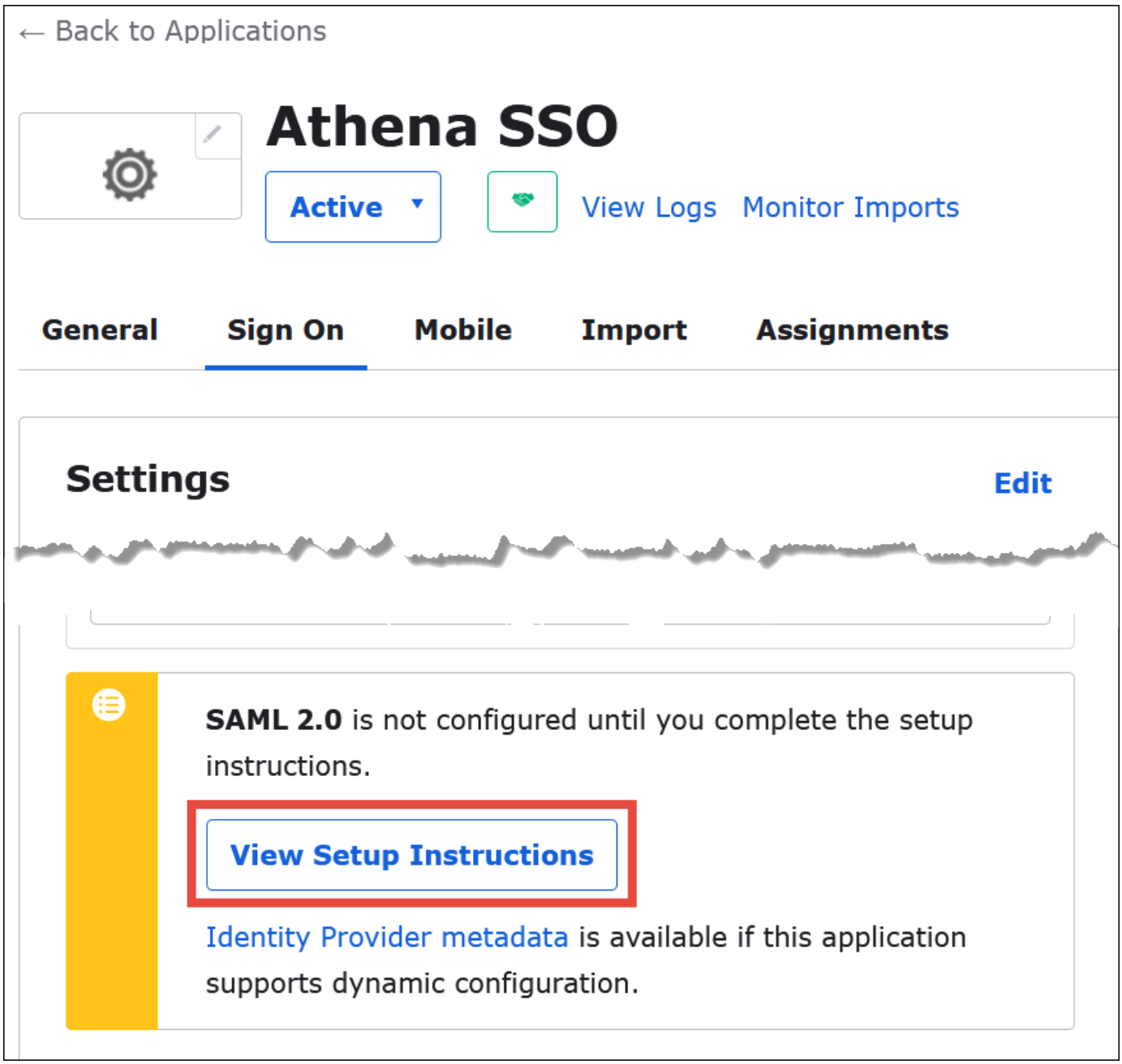
Ora che l'applicazione è stata creata, è possibile ottenere il suo URL di accesso e altri metadati dal pannello di controllo di Okta.

Come ottenere l'URL di accesso dal pannello di controllo di Okta

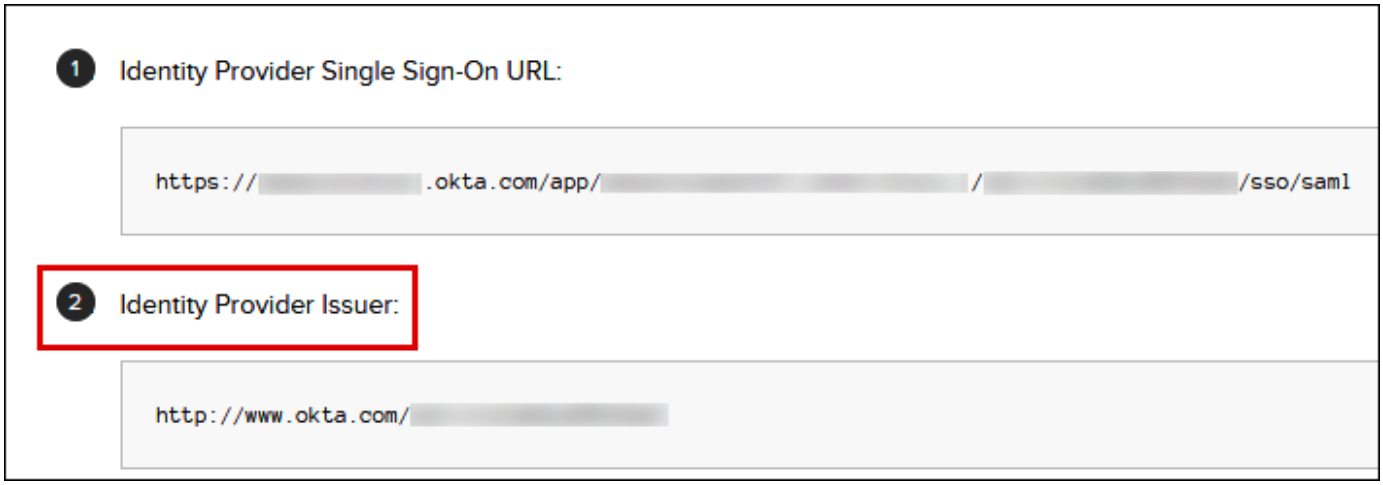
1. Nel pannello di navigazione di Okta, scegli Applications (Applicazioni), Applications (Applicazioni).
2. Scegli l'applicazione per cui trovare l'URL di accesso (ad esempio AthenaSSO).
3. Nella pagina della tua applicazione, scegli Sign On (Accedi).



4. Scegli View Setup Instructions (Visualizza istruzioni di configurazione).



5. Sulla pagina How to Configure SAML 2.0 for Athena SSO (Come configurare SAML 2.0 per SSO Athena), individua l'URL per Identity Provider Issuer (Emittente provider di identità). Alcuni punti nel pannello di controllo di Okta fanno riferimento a questo URL come ID dell'emittente SAML.



6. Copia o memorizza il valore per Identity Provider Single Sign-On URL (URL Single Sign-On del provider di identità).

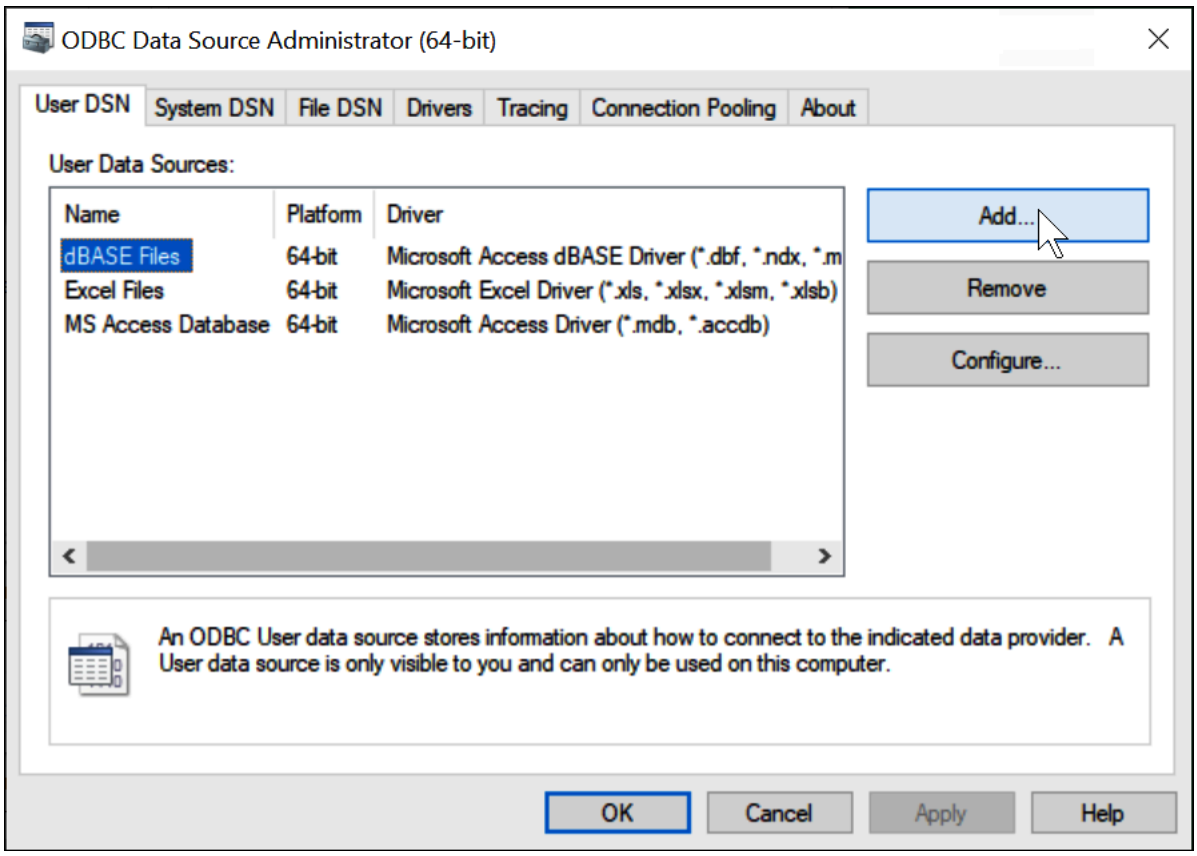
   Nella sezione successiva, quando si configura la connessione ODBC, questo valore verrà fornito come parametro di connessione URL di accesso per il plug-in SAML del browser.

Configurazione della connessione ODBC SAML del browser ad Athena

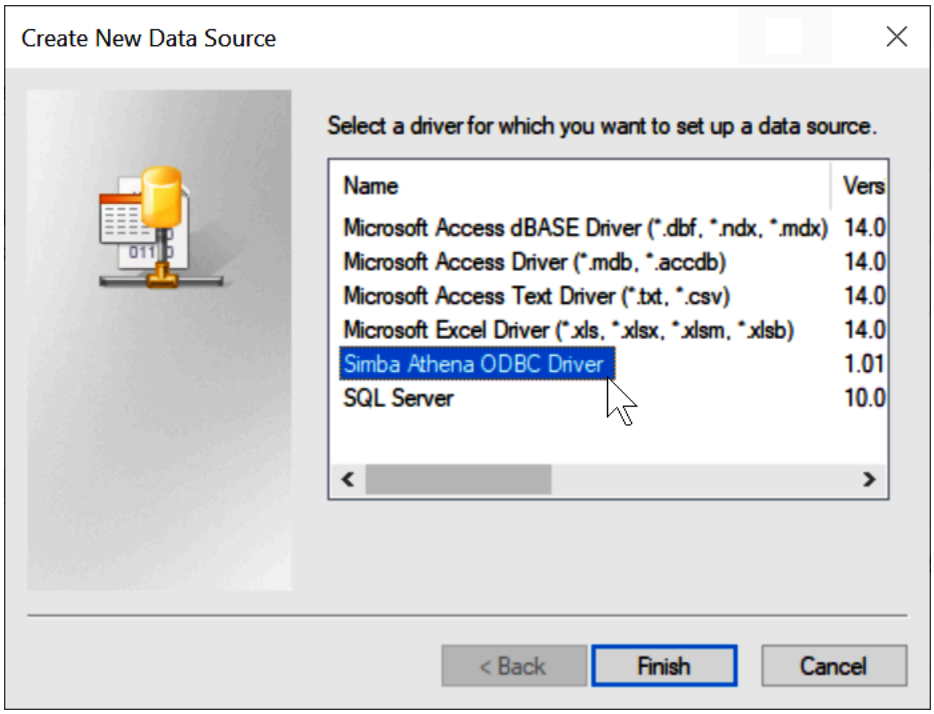
Ora è possibile configurare la connessione SAML del browser ad Athena utilizzando il programma di origini dati ODBC in Windows.

Come configurare la connessione ODBC SAML del browser ad Athena

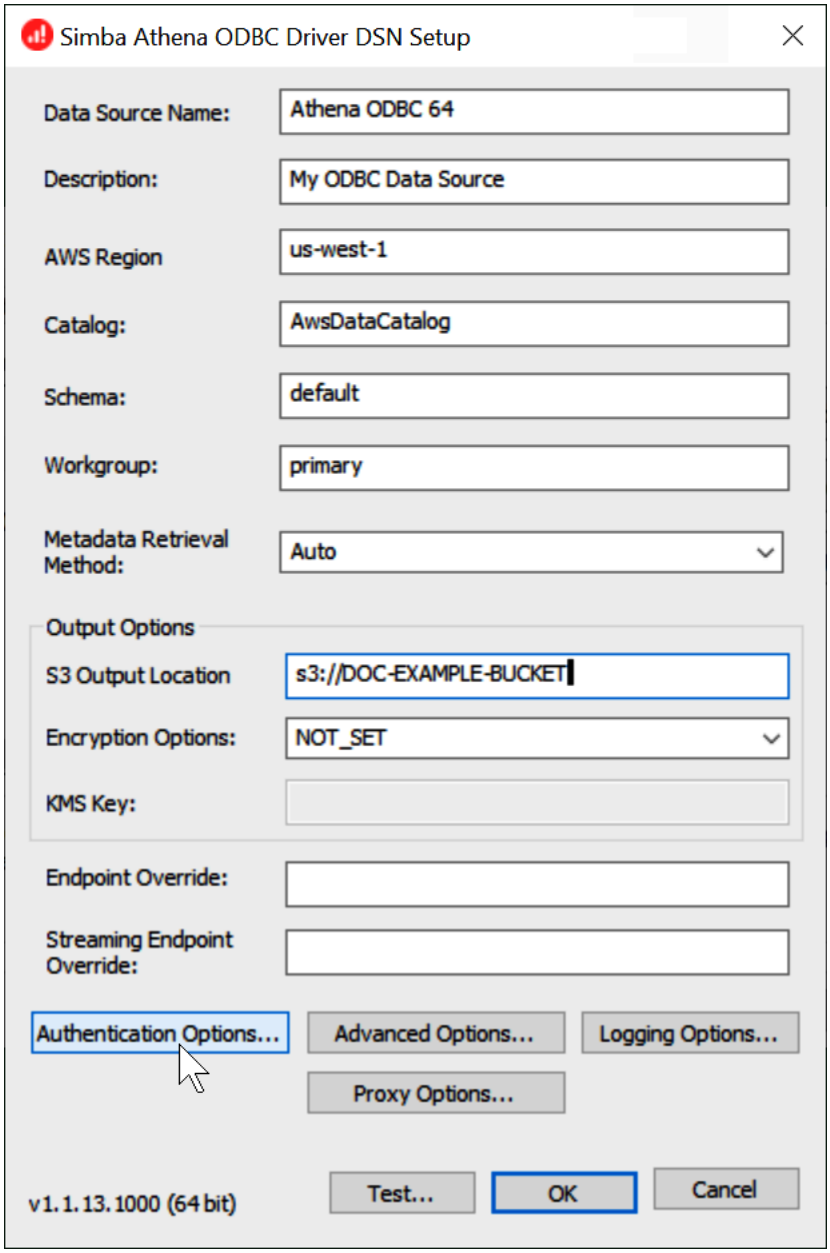
1. In Windows, avvia il programma ODBC Data Sources (Origini dati ODBC).
2. Nel programma ODBC Data Source Administrator (Amministratore origine dati ODBC), scegli Add (Aggiungi).



3. Scegli Simba Athena ODBC Driver (Driver ODBC Simba Athena), quindi scegli Finish (Fine).



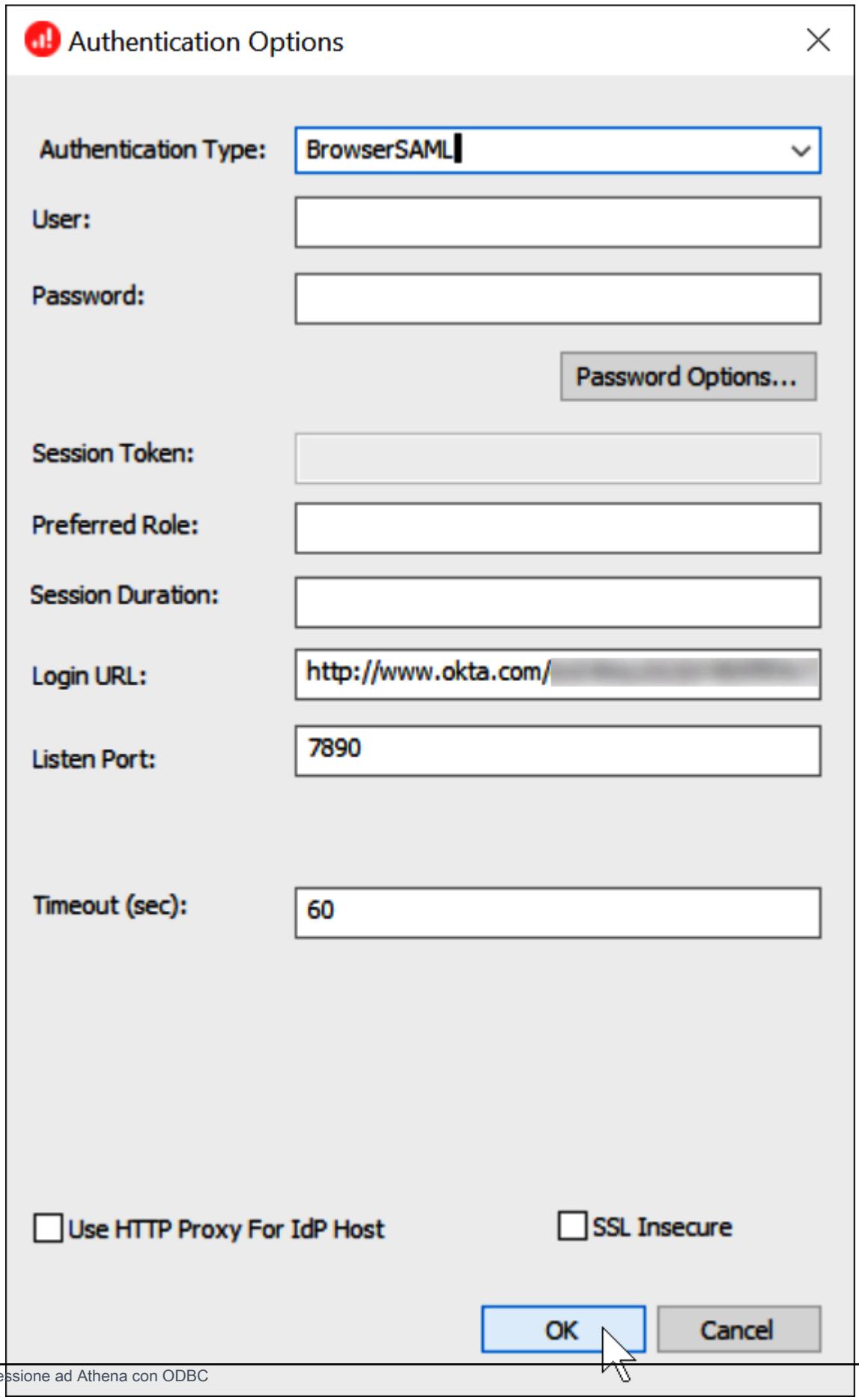
4. Nella finestra di dialogo Simba Athena ODBC Driver DSN Setup (Configurazione DSN driver ODBC Simba Athena), inserisci i valori descritti.



- Per Data Source Name (Nome origine dati), inserisci un nome per l'origine dati (ad esempio, Athena ODBC 64).
- In Description (Descrizione), inserisci una breve descrizione per l'origine dati.
- Per Regione AWS, inserisci quello Regione AWS che stai utilizzando (ad esempio,**`us-west-1`**).
- Per S3 Output Location (Percorso di output S3), inserisci il percorso di Amazon S3 in cui archiviare l'output.

5. Scegli Authentication Options (Opzioni di autenticazione).

6. Nella finestra di dialogo Authentication Options (Opzioni di autenticazione), scegli o inserisci i seguenti valori.


Authentication Options
Authentication Type:
BrowserSAML
User:
Password:
Password Options...
Session Token:
Preferred Role:
Session Duration:
Login URL:
http://www.okta.com/
Listen Port:
7890
Timeout (sec):
60
Use HTTP Proxy For IdP Host
SSL Insecure
OK
Cancel

- Per Authentication Type (Tipo di autenticazione), scegli BrowserSAML.
- Per Login URL (URL di accesso), inserisci l'URL Single Sign-On del provider di identità ottenuto dal pannello di controllo di Okta.
- Per Listen Port (Porta di ascolto), inserisci 7890.
- Per Timeout (sec), inserisci un valore di timeout di connessione in secondi.

7. Scegli OK per chiudere la finestra Authentication Options (Opzioni di autenticazione).
8. Scegli Test per verificare la connessione oppure OK per terminare.

Utilizzo del connettore Power BI di Amazon Athena

Nei sistemi operativi Windows, è possibile utilizzare il connettore Microsoft Power BI per Amazon Athena per analizzare i dati di Amazon Athena in Microsoft Power BI Desktop. Per informazioni su Power BI, consulta Microsoft Power BI. Dopo aver pubblicato un contenuto nel servizio Power BI, è possibile utilizzare la versione di luglio 2021 o successive di Gateway Power BI per mantenere aggiornato il contenuto tramite aggiornamenti on-demand o pianificati.

Prerequisiti

Prima di iniziare, verifica che il tuo ambiente soddisfi i requisiti seguenti. Il driver Amazon Athena ODBC è obbligatorio.

- Account AWS
- Autorizzazioni per usare Athena
- Driver Amazon Athena ODBC
- Power BI Desktop

Funzionalità supportate

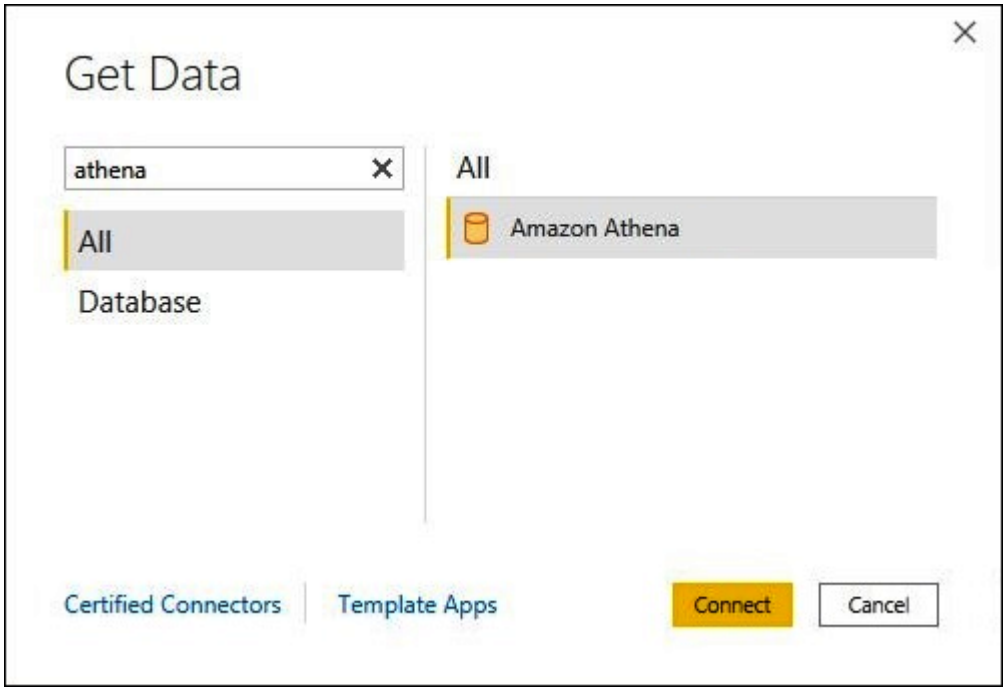
- Import: tabelle e colonne selezionate vengono importate in Power BI Desktop per l'esecuzione di query.
- DirectQuery— Nessun dato viene importato o copiato in Power BI Desktop. Power BI Desktop esegue una query direttamente sull'origine dati sottostante.
- Gateway Power BI: un gateway di dati locale Account AWS che funge da ponte tra il servizio Microsoft Power BI e Athena. Il gateway è necessario per visualizzare i dati nel servizio Microsoft Power BI.

Connettersi ad Amazon Athena

Per connettere Power BI desktop ai dati Amazon Athena, eseguire i seguenti passaggi:

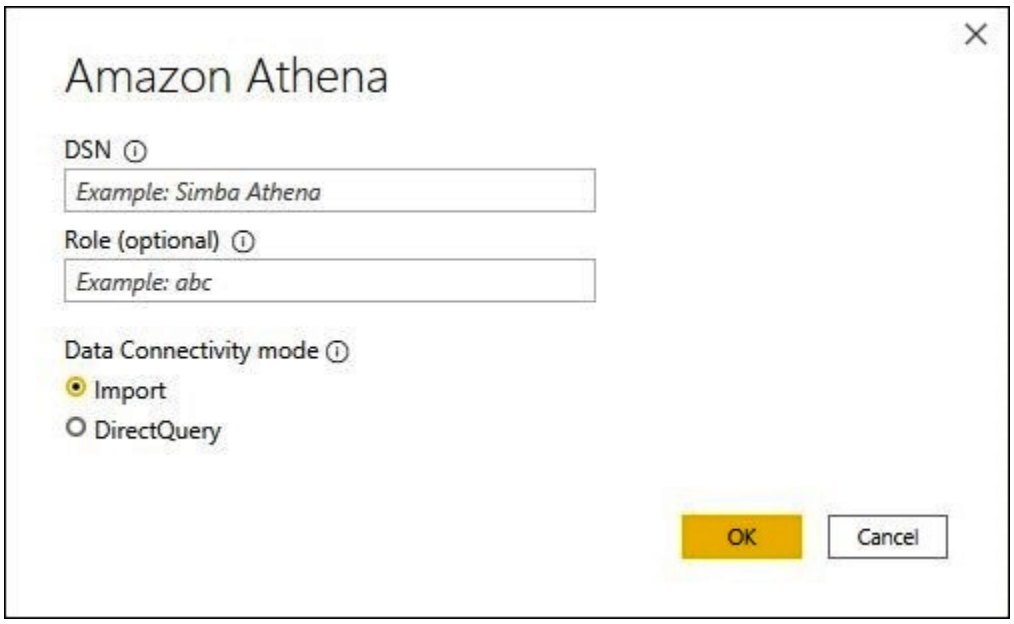
Per connettersi ai dati Athena da Power BI Desktop

1. Avvia Power BI Desktop.
2. Esegui una di queste operazioni:
   - Scegli File, Ottieni i dati
   - Dalla barra della Homepage, scegli Ottieni i dati.
3. Nella casella di ricerca inserisci Athena.
4. Seleziona Amazon Athena e quindi scegli Collegati.

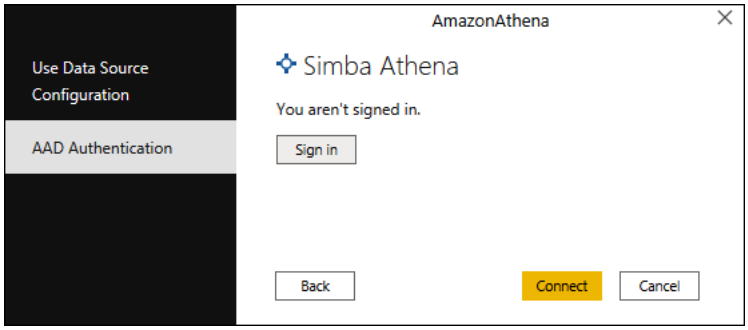


5. Nella pagina di connessione Amazon Athena, inserisci le seguenti informazioni.
   - Per DSN inserisci il nome del DSN ODBC da utilizzare. Per istruzioni sulla configurazione del DSN, consulta Documentazione del driver ODBC.
   - Per Modalità di connettività dei dati scegli una modalità appropriata per il tuo caso d'uso, seguendo queste linee guida generali:
     - Per set di dati più piccoli, scegli Importa. Quando si utilizza la modalità Importa, Power BI lavora con Athena per importare il contenuto dell'intero set di dati da utilizzare nelle visualizzazioni.

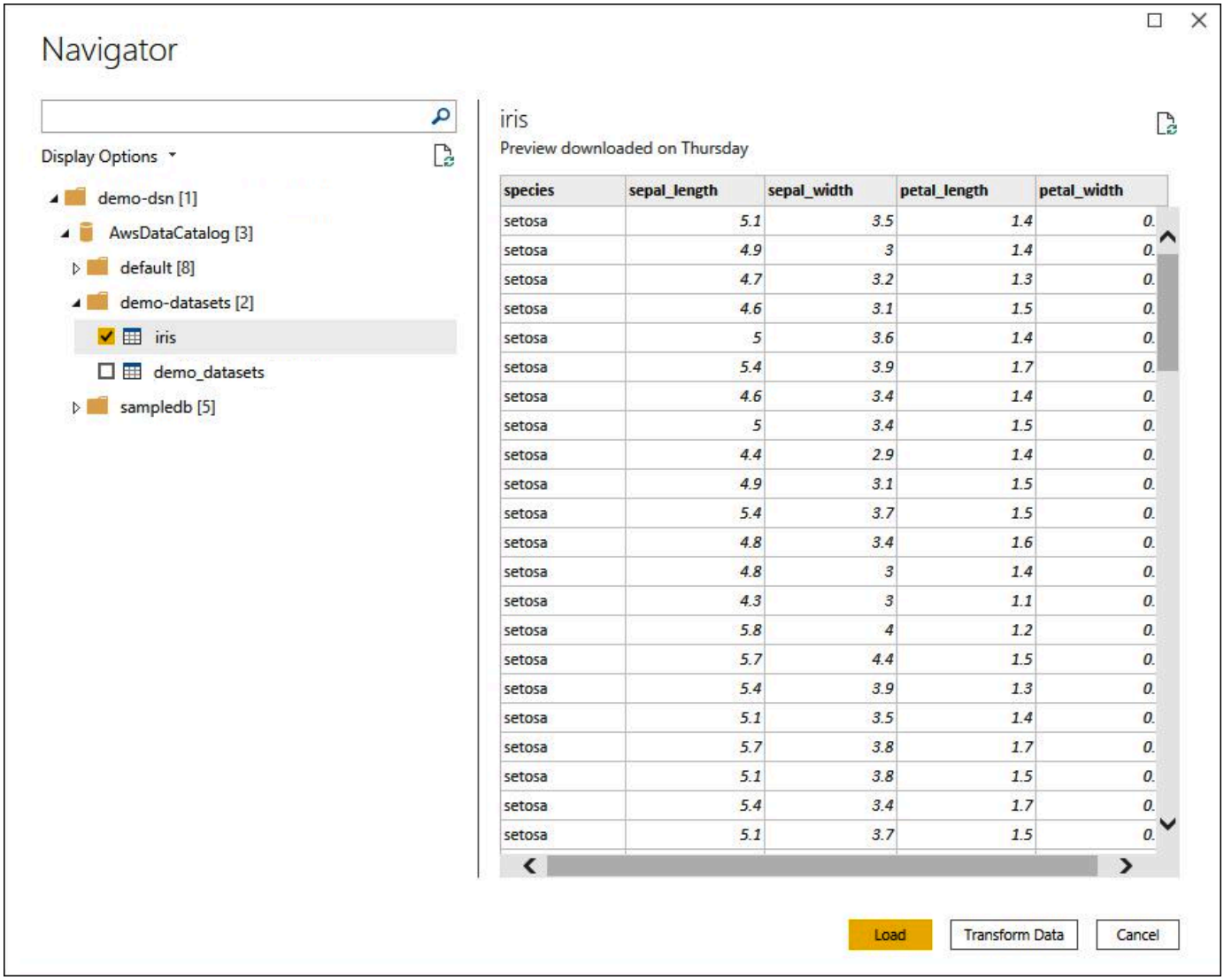
- Per set di dati più grandi, scegli. DirectQuery In DirectQuery modalità, nessun dato viene scaricato sulla workstation. Durante la creazione o l'interazione con una visualizzazione, Microsoft Power BI collabora con Athena per eseguire query dinamicamente sull'origine dati sottostante in modo da visualizzare sempre i dati correnti. Per ulteriori informazioni su DirectQuery, vedere Uso DirectQuery in Power BI Desktop nella documentazione Microsoft.



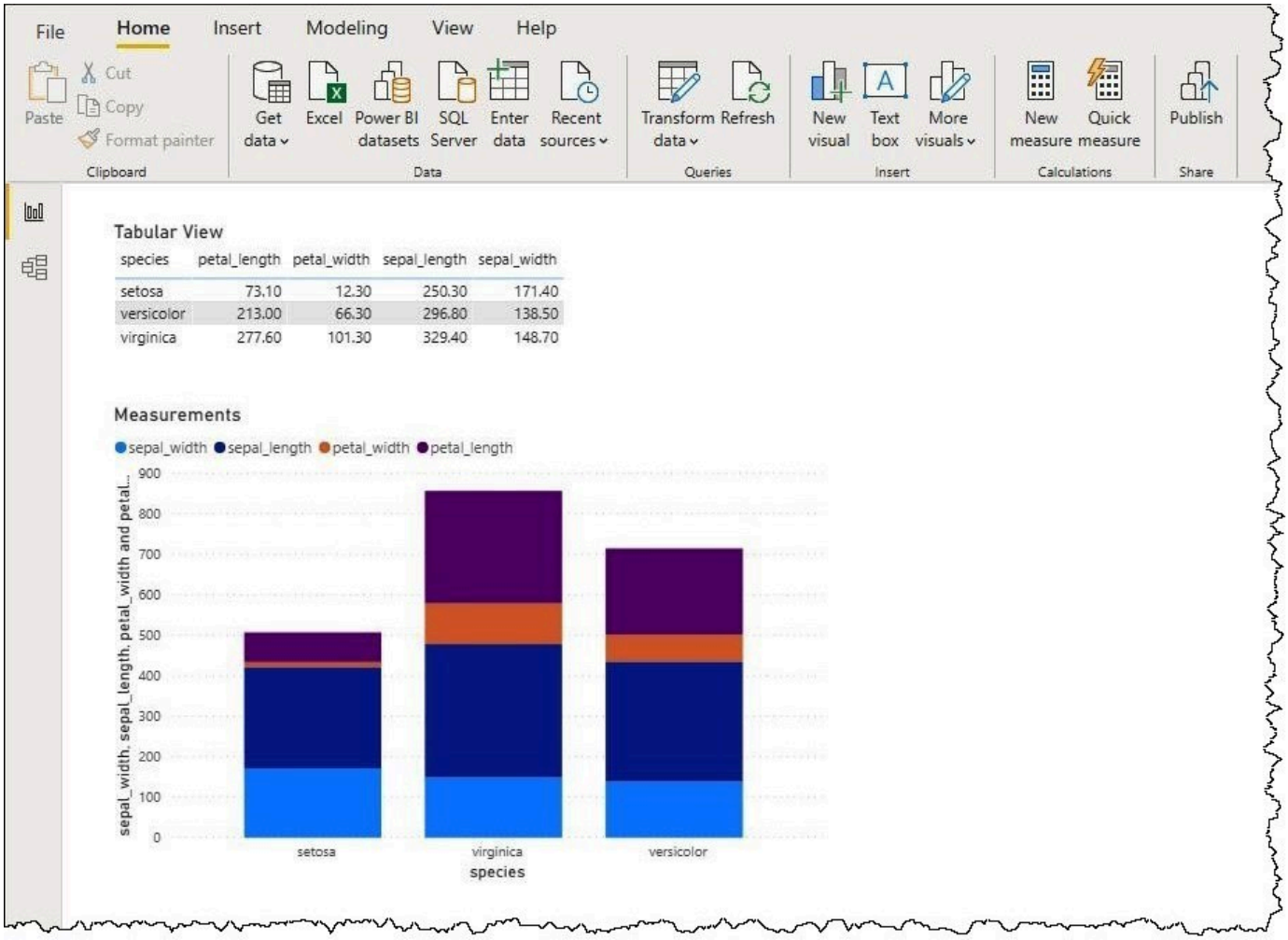
6. Scegli OK.
7. Al prompt per configurare l'autenticazione dell'origine dati, scegli Utilizzo della configurazione origine dati o Autenticazione AAD, quindi scegli Connessione.



Il catalogo dati, i database e le tabelle vengono visualizzati nella finestra di dialogo Navigatore.



8. Nel pannello Opzioni di visualizzazione Seleziona la casella di controllo del set di dati che desideri utilizzare.
9. Se desideri trasformare il set di dati prima di importarlo, vai in fondo alla finestra di dialogo e scegli Trasformazione dei dati. Si aprirà Power Query Editor dove potrai filtrare e perfezionare l'insieme di dati che desideri utilizzare.
10. Scegli Carica. Al termine del caricamento, puoi creare visualizzazioni come quella nell'immagine seguente. Se hai selezionato DirectQuerycome modalità di importazione, Power BI invia una query ad Athena per la visualizzazione richiesta.



Configurazione di un gateway on-premise

È possibile pubblicare dashboard e set di dati nel servizio Power BI in modo che altri utenti possano interagire con loro tramite app Web, per dispositivi mobili e incorporate. Per visualizzare i dati nel servizio Microsoft Power BI, installare il gateway dati locale di Microsoft Power BI in Account AWS. Il gateway funziona come un ponte tra il servizio Microsoft Power BI e Athena.

Per scaricare, installare e testare un gateway dati locale

1. Visita la pagina Download del gateway Microsoft Power BI e scegli la modalità personale o la modalità standard. La modalità personale è utile per testare localmente il connettore Athena. La modalità standard è appropriata in un'impostazione di produzione multiutente.
2. Per installare un gateway locale (in modalità personale o standard), consulta Installare un gateway dati on-premise nella documentazione di Microsoft.

3. Per testare il gateway, attenersi alla procedura descritta in Utilizzare connettori dati personalizzati con il gateway dati on-premise nella documentazione di Microsoft.

Per ulteriori informazioni sui gateway dati locali, consulta le seguenti risorse Microsoft.

- Che cos'è un gateway dati On-Premise?
- Indicazioni per la distribuzione di un gateway dati per Power BI

Per un esempio di configurazione di Power BI Gateway per l'uso con Athena, consulta AWS l'articolo del blog Big Data Creazione rapida di dashboard su Microsoft Power BI utilizzando Amazon Athena.

Creazione di database e tabelle

Amazon Athena supporta un sottoinsieme di istruzioni DDL (Data Definition Language) e di funzioni e operazioni ANSI SQL per definire e interrogare tabelle esterne in cui i dati risiedono in Amazon Simple Storage Service.

Quando si creano un database e una tabella in Athena, si descrivono lo schema e il percorso dei dati, rendendo i dati nella tabella pronti per l'interrogazione in tempo reale.

Per migliorare le prestazioni delle query e ridurre i costi, ti consigliamo di partizionare i dati e utilizzare i formati colonnari open source per l'archiviazione in Amazon S3, ad esempio Apache Parquet o ORC.

Argomenti

- Creazione di database in Athena
- Creazione di tabelle in Athena
- Nomi di tabelle, database e colonne
- Parole chiave riservate
- Posizione delle tabelle in Amazon S3
- Formati di archiviazione colonnare
- Conversione in formati colonnari
- Partizionamento dei dati in Athena
- Proiezione delle partizioni con Amazon Athena

Creazione di database in Athena

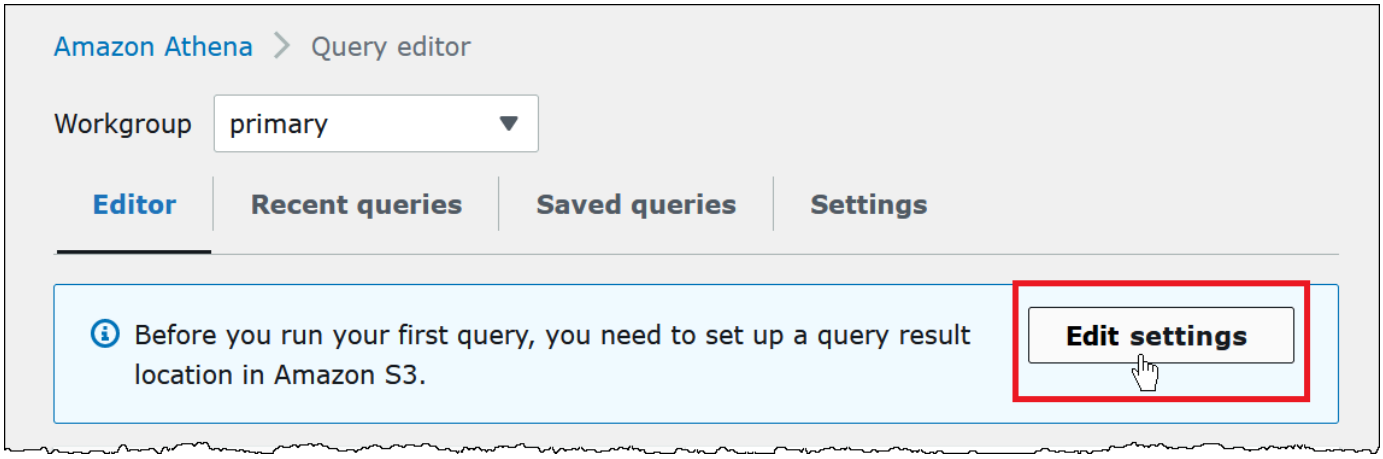
Un database in Athena è un raggruppamento logico delle tabelle in esso create.

Prerequisiti

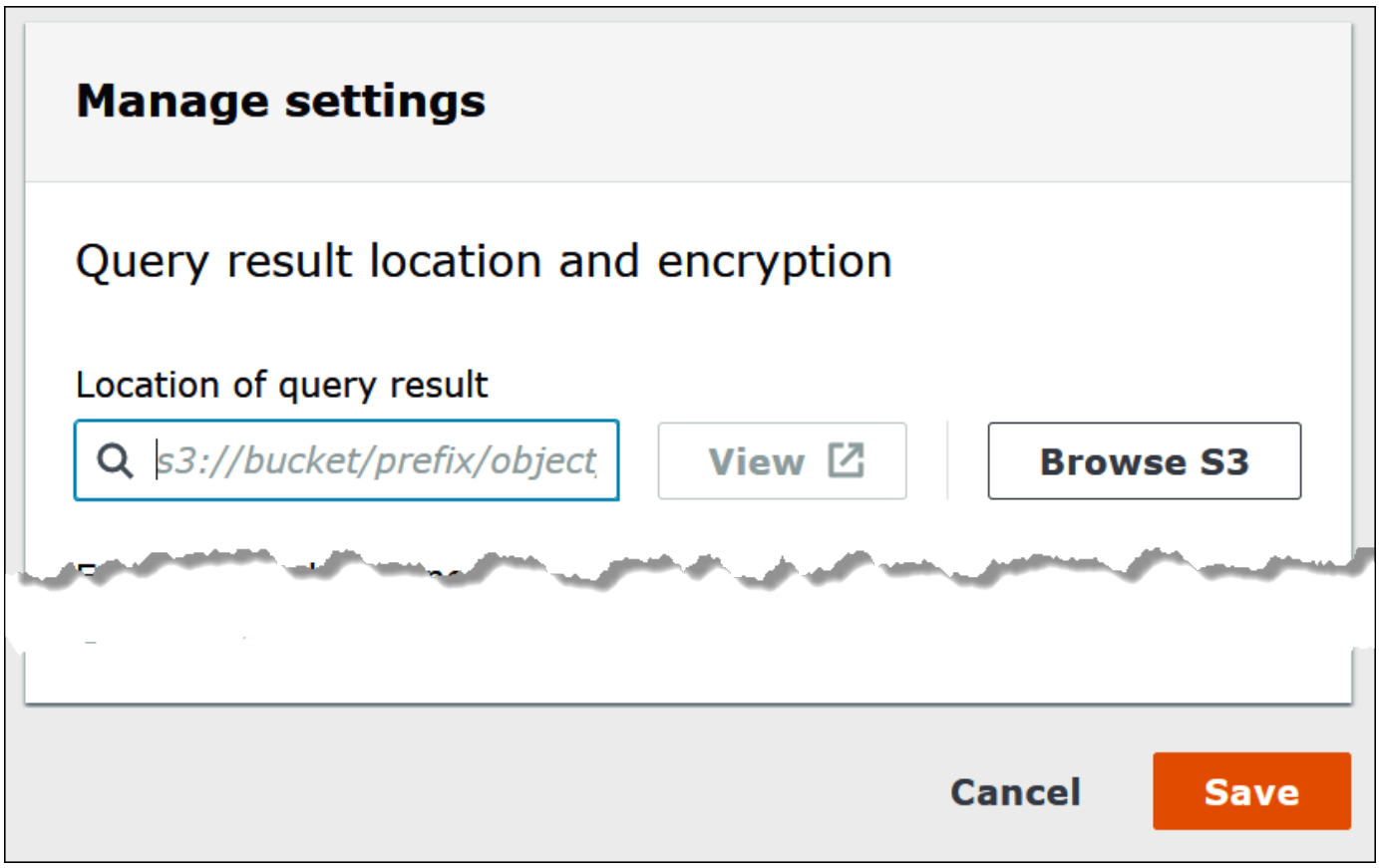
Se non hai già un percorso di output per le query configurato in Amazon S3, esegui i seguenti passaggi preliminari per tale fine.

Come creare un percorso di output per le query

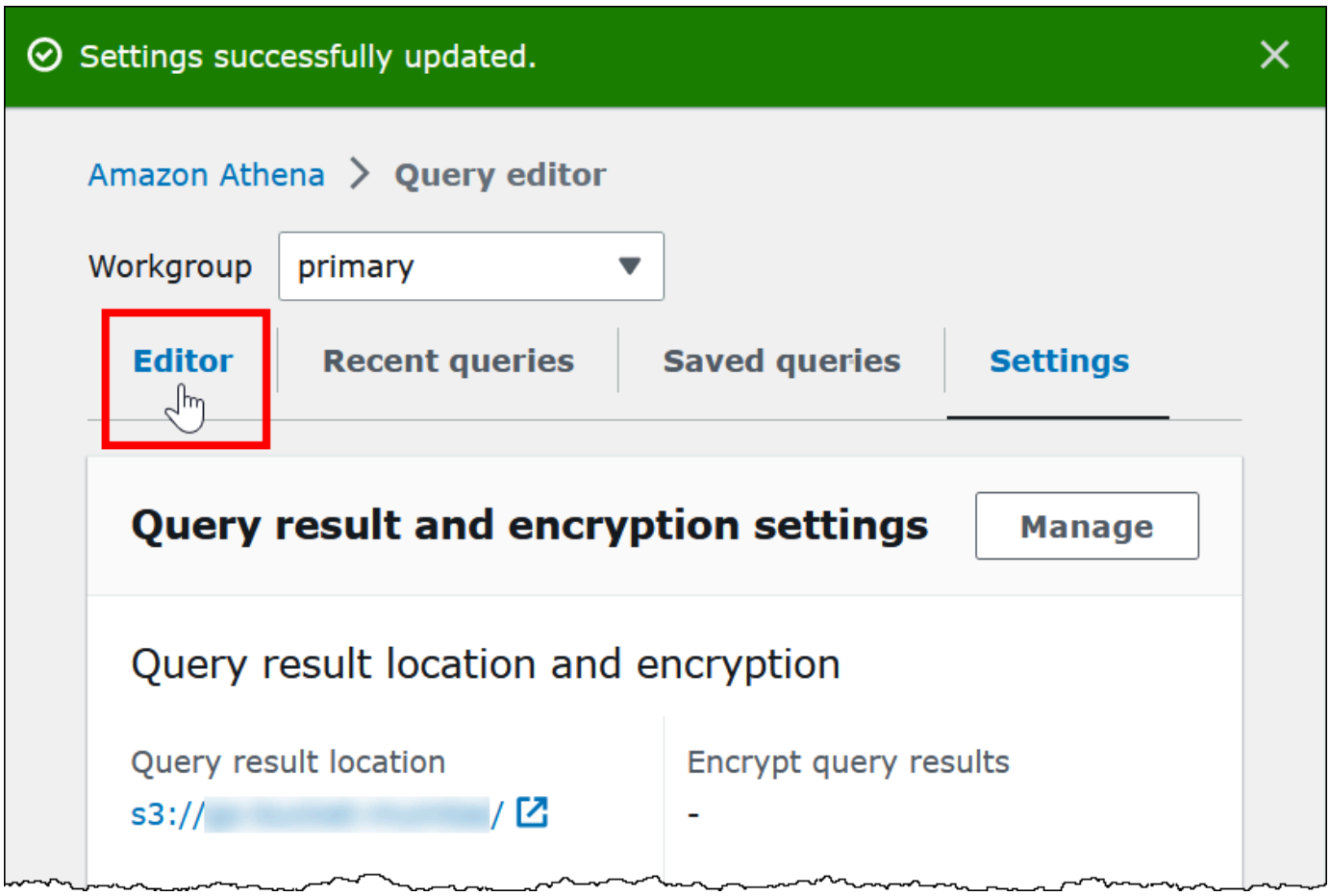
1. Usando la stessa Regione AWS e lo stesso account che utilizzi per Athena, segui la procedura (ad esempio creando una console Amazon S3) per creare un bucket in Amazon S3 in modo da mantenere i risultati delle query di Athena. Configurerai questo bucket come percorso di output per le tue query.
2. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
3. Se è la prima volta che visiti la console di Athena in questa Regione AWS, seleziona Esplora l'editor di query per aprire l'editor di query. Altrimenti, Athena apre la query nell'editor di query.
4. Scegli Edit Settings (Modifica impostazioni) per configurare una nuova posizione dei risultati di una query in Amazon S3.



5. Per Manage settings (Gestisci impostazioni), effettua una delle seguenti operazioni:
   - Nella casella Location of query result (Posizione dei risultati delle query) inserisci il percorso del bucket creato in Amazon S3 per i risultati delle query. Aggiungi al percorso il prefisso `s3://`.
   - Scegli Browse S3 (Sfoglia S3), scegli il bucket Amazon S3 creato per la tua regione corrente, quindi seleziona Choose (Scegli).



6. Seleziona Salva.
7. Scegli Editor per passare all'editor di query.



Creazione di un database

Dopo aver impostato il percorso dei risultati delle query, creare un database nell'editor di query della console Athena è semplice.

Per creare un database con l'editor di query Athena

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Nella scheda Editor, nell'editor di query, immetti il comando Hive DDL (Data Definition Language) `CREATE DATABASE myDataBase`. Sostituire *myDatabase* con il nome che si desidera utilizzare. Per restrizioni sui nomi di database, vedere Nomi di tabelle, database e colonne.
3. Scegli Run (Esegui) o premi **Ctrl+ENTER**.
4. Per rendere il database quello corrente, selezionalo dal menu Database a sinistra dell'editor di query.

Per informazioni sul controllo delle autorizzazioni per i database Athena, consulta la sezione Accesso granulare a database e tabelle in AWS Glue Data Catalog.

Creazione di tabelle in Athena

È possibile eseguire le istruzioni DDL nella console Athena tramite un driver JDBC oppure ODBC o tramite la procedura Create table form (Crea tabella da).

Quando si crea un nuovo schema della tabella in Athena, Athena archivia lo schema in un catalogo di dati e la utilizza quando si eseguono le query.

Athena utilizza un approccio noto come schema-on-read, il che significa che uno schema viene proiettato sui dati nel momento in cui si esegue una query. Non è richiesto quindi alcun caricamento o trasformazione dei dati.

Athena non modifica i dati in Amazon S3.

Athena utilizza Apache Hive per definire le tabelle e creare i database, che sono essenzialmente un namespace logico di tabelle.

Quando si crea un database e una tabella in Athena, si descrivono semplicemente lo schema e il percorso in cui si trovano i dati della tabella in Amazon S3 per l'interrogazione del tempo di lettura. Il database e la tabella, pertanto, hanno un significato leggermente diverso rispetto ai sistemi di database relazionali, i quanto i dati non sono archiviati con la definizione dello schema per il database e la tabella.

Quando si esegue la query, si esegue la query alla tabella tramite SQL standard e i dati vengono letti contestualmente. È possibile trovare indicazioni per la creazione di database e tabelle utilizzando la documentazione di Apache Hive, ma quanto segue fornisce indicazioni specifiche per Athena.

La lunghezza massima per la stringa di una query è di 256 KB.

Hive supporta diversi formati di dati tramite l'uso delle librerie serializer-deserializer (). SerDe È inoltre possibile definire schemi complessi utilizzando espressioni regolari. Per un elenco delle librerie supportate, vedere. SerDe Formati di SerDe e di dati supportati

Considerazioni e limitazioni

Di seguito sono riportate alcune importanti limitazioni e considerazioni per le tabelle in Athena.

Requisiti per tabelle in Athena e dati in Amazon S3

Quando si crea una tabella, è necessario specificare una posizione del bucket Amazon S3 per i dati sottostanti tramite la clausola `LOCATION`. Considera i seguenti aspetti:

- Athena è in grado di eseguire la query solamente alla versione più recente di dati su una versione di un bucket Amazon S3 e non è in grado di eseguire la query alle versioni precedenti dei dati.
- È necessario disporre delle autorizzazioni appropriate per lavorare con i dati nella posizione Amazon S3. Per ulteriori informazioni, consulta Accesso ad Amazon S3 da Athena.
- Athena supporta le query su oggetti archiviati con più classi di storage nello stesso bucket specificato dalla clausola `LOCATION`. Ad esempio, è possibile eseguire query sui dati in oggetti archiviati in diverse classi di storage (Standard, Standard-IA e Intelligent-Tiering) in Amazon S3.
- Athena supporta i bucket con Pagamento a carico del richiedente. Per informazioni su come abilitare il Pagamento a carico del richiedente per i bucket con dati di origine per i quali intendi eseguire query in Athena, consulta la sezione Creare un gruppo di lavoro.
- Athena non supporta l'esecuzione di query sui dati nelle classi di archiviazione S3 Glacier Flexible Retrieval o S3 Glacier Deep Archive. Gli oggetti nelle classi di archiviazione S3 Glacier Flexible Retrieval e S3 Glacier Deep Archive vengono ignorati. In alternativa, puoi utilizzare la classe di archiviazione Amazon S3 Glacier Instant Retrieval, che può essere sottoposta a query da Athena. Per ulteriori informazioni, consulta Classe di archiviazione Amazon S3 Glacier Instant Retrieval.

  Per informazioni sulle classi di archiviazione, consulta Classi di archiviazione, Modifica della classe di archiviazione di un oggetto in Amazon S3, Trasferimento alla classe di archiviazione GLACIER (archiviazione di oggetti) e Bucket con pagamento a carico del richiedente nella Guida per l'utente di Amazon Simple Storage Service.
- Se si esegue le query sui bucket Amazon S3 con un numero elevato di oggetti e i dati non sono partizionati, queste query possono influenzare i limiti del tasso di richieste Get in Amazon S3 e comportare eccezioni Amazon S3. Per evitare errori, partiziona i dati. Inoltre, considera di ottimizzare i tassi di richiesta Amazon S3. Per ulteriori informazioni, consulta Considerazioni sul tasso di richiesta e sulle prestazioni.
- Se si utilizza l'operazione AWS Glue CreateTableAPI o il AWS CloudFormation `AWS::Glue::Table`modello per creare una tabella da utilizzare in Athena senza specificare la `TableType` proprietà e quindi si esegue una query DDL come `SHOW CREATE TABLE` o`MSCK REPAIR TABLE`, è possibile ricevere il messaggio di errore FAILED: NullPointerException Name is null.

Per risolvere l'errore, specifica un valore per l'TableInput`TableType`attributo come parte della chiamata o del modello AWS Glue`CreateTable` API.AWS CloudFormation I valori possibili per `TableType` includono `EXTERNAL_TABLE` o `VIRTUAL_VIEW`.

Questo requisito si applica solo quando si crea una tabella utilizzando l'operazione AWS Glue `CreateTable` API o il `AWS::Glue::Table` modello. Se si crea una tabella per Athena utilizzando un'istruzione DDL o un crawler AWS Glue , la proprietà `TableType` viene definita automaticamente.

Funzioni supportate

Le funzioni supportate nelle query di Athena corrispondono a quelle di Trino e Presto. Per ulteriori informazioni sulle singole funzioni, consulta la sezione dedicata a funzioni e operatori nella documentazione di Trino o Presto.

Le trasformazioni dei dati transazionali non sono supportate

Athena non supporta le operazioni basate su transazioni (ad esempio, quelle che si trovano in Hive o Presto) sulla tabella dei dati. Per un elenco completo di parole chiave non supportate, consulta la sezione DDL non supportati.

Le operazioni che modificano gli stati della tabella sono ACID

Le operazioni di creazione, aggiornamento ed eliminazione delle tabelle sono conformi ad ACID. Ad esempio, se più utenti o client tentano di creare o modificare una tabella esistente nello stesso momento, solo uno di loro potrà eseguire l'operazione.

Le tabelle sono EXTERNAL

Tranne che durante la creazione di tabelle Iceberg, utilizza sempre la parola chiave `EXTERNAL`. Se utilizzi `CREATE TABLE` senza la parola chiave `EXTERNAL` per le tabelle non Iceberg, Athena genera un errore. Quando si elimina una tabella in Athena, solo i metadati della tabella vengono rimossi, mentre i dati rimangono in Amazon S3.

Creazione di tabelle utilizzando AWS Glue o la console Athena

È possibile creare tabelle in Athena utilizzando AWS Glue il modulo aggiungi tabella o eseguendo un'istruzione DDL nell'editor di query Athena.

Per creare una tabella utilizzando il crawler AWS Glue

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Nell'editor di query, accanto a Tabelle e visualizzazioni, scegli Crea e quindi scegli Crawler AWS Glue .
3. Nella pagina Add crawler (Aggiungi crawler) della console AWS Glue segui i passaggi per creare un crawler.

   Per ulteriori informazioni, consulta Utilizzo dei AWS Glue crawler.

Per creare una tabella utilizzando il modulo per la creazione di tabelle Athena

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Nell'editor di query, accanto a Tables and views (Tabelle e visualizzazioni), scegli Create (Crea) e quindi scegli S3 bucket data. (Dati del bucket S3).
3. Nel bucket Crea tabella dai dati del bucket S3, inserisci le informazioni per creare la tabella, quindi scegli Crea tabella. Per ulteriori informazioni sui campi del modulo, vedi Aggiunta di una tabella utilizzando un modulo.

Per creare una tabella tramite DDL Hive

1. Dal menu Database scegliere il database per il quale si desidera creare una tabella. Se non si specifica un database nell'istruzione `CREATE TABLE`, la tabella viene creata nel database attualmente selezionato nell'editor di query.
2. Immettere un'istruzione come la seguente nell'editor di query, quindi scegli Run (Esegui), o premi **`Ctrl+ENTER`**.

```
CREATE EXTERNAL TABLE IF NOT EXISTS cloudfront_logs (
    `Date` Date,
    Time STRING,
    Location STRING,
    Bytes INT,
    RequestIP STRING,
    Method STRING,
    Host STRING,
    Uri STRING,
    Status INT,
    Referrer STRING,
```

```
    OS String,
    Browser String,
    BrowserVersion String
) ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
"input.regex" = "^(?!#)([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s
+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+[^\(]+[\(]([^\;]+).*\%20([^
\/]+)[\/](.*)$"
) LOCATION 's3://athena-examples-MyRegion/cloudfront/plaintext/';
```

Visualizzazione delle informazioni sulla tabella

Dopo aver creato una tabella in Athena, il nome viene visualizzato nell'elenco Tables (Tabelle) a sinistra. Seleziona i tre puntini verticali accanto al nome della tabella nella console Athena per visualizzare le informazioni sulla tabella e gestirla.

- Anteprima tabella: mostra le prime 10 righe di tutte le colonne eseguendo l'istruzione `SELECT * FROM "database_name"."table_name" LIMIT 10` nell'editor di query Athena.
- Genera DDL tabella: genera un'istruzione DDL che è possibile utilizzare per ricreare la tabella eseguendo l'istruzione `SHOW CREATE TABLE table_name` nell'editor di query di Athena.
- Carica partizioni: esegue l'istruzione `MSCK REPAIR TABLE table_name` nell'editor di query Athena. Questa opzione è disponibile solo se la tabella dispone di partizioni.
- Inserisci nell'editor: inserisce il nome della tabella nell'editor di query nella posizione di modifica corrente.
- Elimina tabella: visualizza una finestra di dialogo di conferma che chiede se si desidera eliminare la tabella. Se sei d'accordo, esegue l'istruzione `DROP TABLE table_name` nell'editor di query di Athena.
- Proprietà tabella: mostra il nome della tabella, il nome del database, l'ora di creazione e se la tabella ha dati crittografati.

Nomi di tabelle, database e colonne

Usa questi suggerimenti per denominare gli oggetti del database in Athena.

Requisiti per i nomi di database, tabelle e colonne

- I caratteri accettabili per i nomi di database, i nomi delle tabelle e i nomi delle colonne AWS Glue devono essere una stringa UTF-8. La lunghezza della stringa non deve essere inferiore a 1 o superiore a 255 byte. Il superamento di questo limite genera un errore del tipo Impossibile soddisfare il vincolo per il valore in 'name': il member deve avere una lunghezza inferiore o uguale a 255. I caratteri che possono essere utilizzati includono spazi e sono definiti dal seguente schema di stringhe a riga singola:

  ```
  [\u0020-\uD7FF\uE000-\uFFFD\uD800\uDC00-\uDBFF\uDFFF\t]*
  ```

- Attualmente, il modello AWS Glue regex consente di aggiungere spazi iniziali all'inizio dei nomi. Poiché questi spazi iniziali possono essere difficili da rilevare e possono causare problemi di usabilità dopo la creazione, evitate di creare nomi di oggetti con spazi iniziali.
- Se utilizzi un AWS::Glue::Database AWS CloudFormation modello per creare un AWS Glue database e non specifichi un nome di database, genera AWS Glue automaticamente un nome di database nel formato *`resource_name–random_string`* che non è compatibile con Athena.
- È possibile utilizzare AWS Glue Catalog Manager per rinominare le colonne, ma non i nomi delle tabelle o dei database. Per ovviare a questa limitazione, è necessario utilizzare una definizione del vecchio database per creare un database con il nuovo nome. Quindi si utilizzano le definizioni delle tabelle del vecchio database per ricreare le tabelle nel nuovo database. Per fare ciò, puoi usare AWS CLI o AWS Glue SDK. Per le fasi, consulta Utilizzo di AWS CLI per ricreare un AWS Glue database e le relative tabelle.

Utilizzare lettere minuscole per i nomi delle tabelle e delle colonne in Athena

Athena accetta maiuscole e minuscole nelle query DDL e DML, ma scrive i nomi in minuscolo quando esegue la query. Per questo motivo, evitare di usare maiuscole e minuscole per i nomi delle tabelle o delle colonne e non fare affidamento sul solo contenitore di Athena per distinguere tali nomi. Ad esempio, se utilizzi un'istruzione DDL per creare una colonna denominata `Castle`, la colonna creata verrà scritta in minuscolo come `castle`. Se si specifica quindi il nome della colonna in una query DML come `Castle` o `CASTLE`, Athena scriverà in minuscolo il nome per l'esecuzione della query, ma mostrerà l'intestazione della colonna utilizzando la struttura scelta nella query.

I nomi di database, tabelle e colonne devono contenere un massimo di 255 caratteri.

Nomi che iniziano con un trattino basso

Quando crei tabelle, utilizza i backtick per racchiudere i nomi di tabelle, viste o colonne che iniziano con un trattino basso. Per esempio:

```
CREATE EXTERNAL TABLE IF NOT EXISTS `_myunderscoretable`(
  `_id` string, `_index` string)
LOCATION 's3://DOC-EXAMPLE-BUCKET/'
```

Nomi di tabelle, viste o colonne che iniziano con i numeri

Quando si eseguono le query `SELECT`, `CTAS` o `VIEW` inserire le virgolette attorno a identificatori come nomi di tabelle, viste o colonne che iniziano con una cifra. Per esempio:

```
CREATE OR REPLACE VIEW "123view" AS
SELECT "123columnone", "123columntwo"
FROM "234table"
```

Nomi di colonna e tipi complessi

Per i tipi complessi, nei nomi delle colonne sono consentiti solo caratteri alfanumerici, il trattino basso (_) e il punto (`.`). Per creare una tabella e una mappatura per le chiavi con caratteri limitati, è possibile utilizzare un'istruzione DDL personalizzata. Per ulteriori informazioni, consulta l'articolo Creare tabelle in Amazon Athena da JSON annidato e mappature usando SerDe JSON nel Big Data Blog.AWS

Parole riservate

Alcune parole riservate in Athena devono avere il carattere escape. Per impostare il carattere escape di parole chiave riservate in istruzioni DDL, occorre racchiuderle tra apici retroversi (`). Per impostare il carattere escape di parole chiave riservate in istruzioni SQL `SELECT` e nelle query su viste, occorre racchiuderle tra doppie virgolette (").

Per ulteriori informazioni, consulta Parole chiave riservate.

Altre risorse

Per la sintassi completa per la creazione di database e tabelle, vedere le pagine seguenti.

- CREATE DATABASE
- CREATE TABLE

Per ulteriori informazioni su database e tabelle in AWS Glue, consulta Databases and Tables nella Developer Guide.AWS Glue

Parole chiave riservate

Quando si esegue le query in Athena che includono parole chiave riservate, devi racchiuderli tra caratteri speciali. Utilizza gli elenchi in questo argomento per verificare quali parole chiave sono riservate in Athena.

Per impostare il carattere escape di parole chiave riservate in istruzioni DDL, occorre racchiuderle tra apici retroversi (`). Per impostare il carattere escape di parole chiave riservate in istruzioni SQL `SELECT` e nelle query su viste, occorre racchiuderle tra doppie virgolette (").

- Elenco delle parole chiave riservate nelle istruzioni DDL
- Elenco delle parole chiave riservate nelle istruzioni SQL SELECT
- Esempi di query con parole riservate

Elenco delle parole chiave riservate nelle istruzioni DDL

Athena utilizza il seguente elenco di parole chiave riservate nelle sue istruzioni DDL. Se le utilizzi senza importare il carattere escape, Athena emette un errore. Per impostarne il carattere escape, racchiudile tra apici retroversi (`).

Non è possibile utilizzare parole chiave riservate DDL come nomi di identificatori in istruzioni DDL senza racchiuderle tra apici retroversi (`).

```
ALL, ALTER, AND, ARRAY, AS, AUTHORIZATION, BETWEEN, BIGINT,
BINARY, BOOLEAN, BOTH, BY, CASE, CASHE, CAST, CHAR, COLUMN,
CONF, CONSTRAINT, COMMIT, CREATE, CROSS, CUBE, CURRENT,
CURRENT_DATE, CURRENT_TIMESTAMP, CURSOR, DATABASE, DATE,
DAYOFWEEK, DECIMAL, DELETE, DESCRIBE, DISTINCT, DOUBLE, DROP,
ELSE, END, EXCHANGE, EXISTS, EXTENDED, EXTERNAL, EXTRACT,
FALSE, FETCH, FLOAT, FLOOR, FOLLOWING, FOR, FOREIGN, FROM,
FULL, FUNCTION, GRANT, GROUP, GROUPING, HAVING, IF, IMPORT,
IN, INNER, INSERT, INT, INTEGER, INTERSECT, INTERVAL, INTO,
IS, JOIN, LATERAL, LEFT, LESS, LIKE, LOCAL, MACRO, MAP, MORE,
NONE, NOT, NULL, NUMERIC, OF, ON, ONLY, OR, ORDER, OUT,
OUTER, OVER, PARTIALSCAN, PARTITION, PERCENT, PRECEDING,
PRECISION, PRESERVE, PRIMARY, PROCEDURE, RANGE, READS,
```

```
REDUCE, REGEXP, REFERENCES, REVOKE, RIGHT, RLIKE, ROLLBACK,
ROLLUP, ROW, ROWS, SELECT, SET, SMALLINT, START,TABLE,
TABLESAMPLE, THEN, TIME, TIMESTAMP, TO, TRANSFORM, TRIGGER,
TRUE, TRUNCATE, UNBOUNDED,UNION, UNIQUEJOIN, UPDATE, USER,
USING, UTC_TIMESTAMP, VALUES, VARCHAR, VIEWS, WHEN, WHERE,
WINDOW, WITH
```

Elenco delle parole chiave riservate nelle istruzioni SQL SELECT

Athena utilizza il seguente elenco di parole chiave riservate nelle istruzioni SQL `SELECT` e nelle query su viste.

Se si utilizzano queste parole chiave come identificatori, è necessario racchiuderle tra doppie virgolette (") nella istruzioni di query.

```
ALTER, AND, AS, BETWEEN, BY, CASE, CAST, CONSTRAINT, CREATE,
CROSS, CUBE, CURRENT_CATALOG, CURRENT_DATE, CURRENT_PATH,
CURRENT_SCHEMA, CURRENT_TIME, CURRENT_TIMESTAMP, CURRENT_USER,
DEALLOCATE, DELETE, DESCRIBE, DISTINCT, DROP, ELSE, END, ESCAPE,
EXCEPT, EXECUTE, EXISTS, EXTRACT, FALSE, FIRST, FOR, FROM,
FULL, GROUP, GROUPING, HAVING, IN, INNER, INSERT, INTERSECT,
INTO, IS, JOIN, JSON_ARRAY, JSON_EXISTS, JSON_OBJECT,
JSON_QUERY, JSON_TABLE, JSON_VALUE, LAST, LEFT, LIKE,
LISTAGG, LOCALTIME, LOCALTIMESTAMP, NATURAL, NORMALIZE,
NOT, NULL, OF, ON, OR, ORDER, OUTER, PREPARE, RECURSIVE, RIGHT,
ROLLUP, SELECT, SKIP, TABLE, THEN, TRIM, TRUE, UESCAPE, UNION,
UNNEST, USING, VALUES, WHEN, WHERE, WITH
```

Esempi di query con parole riservate

La query nell'esempio riportato di seguito utilizza gli apici retroversi (`) come caratteri escape delle parole chiave riservate associate al DDL partizione e data, utilizzate per un nome di tabella e per uno dei nomi di colonna:

```
CREATE EXTERNAL TABLE `partition` (
`date` INT,
col2 STRING
)
PARTITIONED BY (year STRING)
STORED AS TEXTFILE
LOCATION 's3://DOC-EXAMPLE-BUCKET/test_examples/';
```

Le seguenti query di esempio includono un nome di colonna contenente le parole chiave riservate correlate al DDL nelle istruzioni `ALTER TABLE ADD PARTITION` e `ALTER TABLE DROP PARTITION`. Le parole chiave DDL riservate vengono racchiuse tra apici retroversi (`):

```
ALTER TABLE test_table
ADD PARTITION (`date` = '2018-05-14')
```

```
ALTER TABLE test_table
DROP PARTITION (`partition` = 'test_partition_value')
```

La seguente query di esempio include una parola chiave riservata (end) come identificatore in un'istruzione `SELECT`. Il carattere escape della parola chiave è impostato racchiudendola tra doppie virgolette:

```
SELECT *
FROM TestTable
WHERE "end" != nil;
```

La seguente query di esempio include una parola chiave riservata (first) in un'istruzione `SELECT`. Il carattere escape della parola chiave è impostato racchiudendola tra doppie virgolette:

```
SELECT "itemId"."first"
FROM testTable
LIMIT 10;
```

Posizione delle tabelle in Amazon S3

Quando esegui una `CREATE TABLE` query in Athena, Athena registra la tabella nel AWS Glue Data Catalog, dove Athena archivia i metadati.

Puoi specificare il percorso dei dati in Amazon S3, utilizzare la proprietà `LOCATION`, come illustrato nel seguente breve esempio:

```
CREATE EXTERNAL TABLE `test_table`(
...
)
ROW FORMAT ...
STORED AS INPUTFORMAT ...
OUTPUTFORMAT ...
```

```
LOCATION s3://DOC-EXAMPLE-BUCKET/folder/
```

- Per informazioni sulla denominazione dei bucket, consulta Restrizioni e limitazioni dei bucket nella Guida per l'utente di Amazon Simple Storage Service.
- Per informazioni sull'utilizzo delle cartelle in Amazon S3, consulta Utilizzo delle cartelle nella Guida per l'utente di Amazon Simple Storage Service.

La `LOCATION` in Amazon S3 specifica tutti i file che rappresentano la tabella.

> **Important**
>
> Athena legge tutti i dati archiviati nella cartella Amazon S3 specificata. Se alcuni dati non devono essere letti da Athena, non archiviarli nella stessa cartella Amazon S3 dei dati che vuoi che Athena legga. Se stai utilizzando il partizionamento, per garantire che Athena analizzi i dati all'interno di una partizione, il filtro `WHERE` deve includere la partizione. Per ulteriori informazioni, consulta Ubicazione della tabella e partizioni.

Quando si specifica `LOCATION` nell'istruzione `CREATE TABLE`, utilizza le seguenti linee guida:

- Utilizzare una barra finale.
- Puoi utilizzare un percorso di una cartella Amazon S3 o di un alias del punto di accesso Amazon S3. Per informazioni sugli alias del punto di accesso Amazon S3, consulta Utilizzo di un alias in stile bucket per il punto di accesso nella Guida per l'utente di Amazon S3.

Utilizza:

```
s3://DOC-EXAMPLE-BUCKET/folder/
```

```
s3://DOC-EXAMPLE-BUCKET-metadata-s3alias/folder/
```

Non utilizzare uno dei seguenti elementi per specificare `LOCATION` per i propri dati.

- Per specificare il percorso dei file, non utilizzare nomi di file, trattini bassi, caratteri jolly né modelli glob.
- Non aggiungere la notazione HTTP completa, ad esempio `s3.amazon.com`, al percorso del bucket Amazon S3.

- Non utilizzare cartelle vuote come `//` nel percorso, come segue: `S3://DOC-EXAMPLE-BUCKET/folder//folder/`. Pur essendo un percorso Amazon S3 valido, Athena non lo consente e lo cambia in `s3://DOC-EXAMPLE-BUCKET/folder/folder/`, rimuovendo il `/` extra.

  Non utilizzare:

```
s3://DOC-EXAMPLE-BUCKET
s3://DOC-EXAMPLE-BUCKET/*
s3://DOC-EXAMPLE-BUCKET/mySpecialFile.dat
s3://DOC-EXAMPLE-BUCKET/prefix/filename.csv
s3://DOC-EXAMPLE-BUCKET.s3.amazon.com
S3://DOC-EXAMPLE-BUCKET/prefix//prefix/
arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix
s3://arn:aws:s3:<region>:<account_id>:accesspoint/<accesspointname>
https://<accesspointname>-<number>.s3-accesspoint.<region>.amazonaws.com
```

Ubicazione della tabella e partizioni

I dati di origine possono essere raggruppati in cartelle Amazon S3 denominate partizioni basate su un insieme di colonne. Ad esempio, queste colonne potrebbero rappresentare l'anno, il mese e il giorno di creazione di un record specifico.

Quando crei una tabella, è possibile scegliere di partizionarla. Quando Athena esegue una query SQL su una tabella non partizionata, utilizza la proprietà `LOCATION` dalla definizione della tabella come percorso di base per elencare e quindi analizzare tutti i file disponibili. Tuttavia, prima di poter eseguire interrogazioni su una tabella partizionata, è necessario aggiornare il Data Catalog con le informazioni sulla partizione. AWS Glue Queste informazioni rappresentano lo schema di file all'interno della partizione specifica e la `LOCATION` dei file in Amazon S3 per la partizione.

- Per informazioni su come il AWS Glue crawler aggiunge partizioni, vedi Come fa un crawler a determinare quando creare le partizioni? nella AWS Glue Guida per gli sviluppatori.
- Per ulteriori informazioni su come configurare il crawler per creare le tabelle per i dati nelle partizioni esistenti, consulta la sezione Utilizzo di più origini dati con i crawler.
- Puoi anche creare partizioni in una tabella direttamente in Athena. Per ulteriori informazioni, consulta Partizionamento dei dati in Athena.

Quando Athena esegue una query su una tabella partizionata, controlla prima se sono state utilizzate eventuali colonne partizionate nella clausola `WHERE` della query. Se vengono utilizzate colonne

partizionate, Athena richiede al AWS Glue Data Catalog di restituire la specifica della partizione corrispondente alle colonne di partizione specificate. Le specifiche di partizione includono la proprietà `LOCATION` che indica ad Athena quale prefisso Amazon S3 utilizzare per leggere i dati. In questo caso, solo i dati archiviati in questo prefisso vengono analizzati. Se non si utilizzano colonne partizionate nella clausola `WHERE`, Athena esegue la scansione di tutti i file che appartengono alle partizioni della tabella.

Per esempi di utilizzo del partizionamento con Athena per migliorare le prestazioni delle query e ridurre i costi delle query, consulta i 10 migliori suggerimenti per l'ottimizzazione delle prestazioni per Amazon Athena.

Formati di archiviazione colonnare

Apache Parquet e ORC sono formati di archiviazione colonnari ottimizzati per il recupero rapido dei dati e utilizzati nelle applicazioni analitiche. AWS

I formati di archiviazione colonnare presentano le seguenti caratteristiche, che li rendono ideali per l'utilizzo con Athena:

- Compressione per colonna, con algoritmo selezionato per il tipo di dati della colonna per risparmiare spazio di archiviazione in Amazon S3 e ridurre lo spazio su disco e le operazioni di I/O durante l'elaborazione delle query.
- Pushdown dei predicati in Parquet e ORC, che consente alle query Athena di recuperare solo i blocchi necessari migliorandone così le prestazioni. Quando una query Athena ottiene i valori specifici della colonna dai dati, utilizza le statistiche dei predicati dei blocchi di dati, ad esempio i valori min/max, per determinare se leggere o saltare il blocco.
- Frazionamento dei dati in Parquet e ORC, che consente ad Athena di frazionare la lettura dei dati su più lettori e aumentare il parallelismo durante l'elaborazione delle query.

Per convertire i dati grezzi esistenti da altri formati di archiviazione in Parquet o ORC, puoi eseguire le query CREATE TABLE AS SELECT (CTAS) in Athena e specificare un formato di archiviazione dei dati come Parquet o ORC, oppure utilizzare il Crawler. AWS Glue

Come scegliere tra Parquet e ORC

La scelta tra ORC (Optimized Row Columnar) e Parquet dipende dai requisiti di uso specifico.

Apache Parquet offre efficienti schemi di compressione e codifica dei dati ed è ideale per eseguire query complesse ed elaborare grandi quantità di dati. Parquet è ottimizzato per l'uso con Apache Arrow, il che può essere vantaggioso se utilizzi strumenti correlati ad Arrow.

ORC offre un modo efficiente per archiviare i dati Hive. I file ORC sono spesso di dimensioni inferiori rispetto ai file Parquet e gli indici ORC possono velocizzare l'esecuzione di query. Inoltre, ORC supporta tipi complessi come strutture, mappe ed elenchi.

Quando scegli tra Parquet e ORC, valuta quanto segue:

Prestazioni delle query: poiché Parquet supporta una serie più ampia di tipi di query, Parquet potrebbe essere la scelta migliore se intendi eseguire query complesse.

Tipi di dati complessi: se utilizzi tipi di dati complessi, ORC potrebbe essere la scelta migliore in quanto supporta una serie più ampia di tipi di dati complessi.

Dimensioni dei file: se lo spazio su disco è un problema, ORC di solito produce file di dimensioni inferiori, il che può ridurre i costi di archiviazione.

Compressione: sia Parquet sia ORC offrono una buona compressione, ma il formato migliore per te può dipendere dal caso d'uso specifico.

Evoluzione: sia Parquet che ORC supportano l'evoluzione di schema, il che significa che puoi aggiungere, rimuovere o modificare colonne nel tempo.

Sia Parquet che ORC sono buone scelte per le applicazioni di big data, ma valuta i requisiti del tuo scenario prima di prendere una decisione. Puoi eseguire benchmark sui tuoi dati e sulle tue query per stabilire quale formato offre prestazioni migliori per il tuo caso d'uso.

Conversione in formati colonnari

Le prestazioni della query Amazon Athena migliorano se i dati vengono convertiti in formati colonnari open source, come Apache Parquet o ORC.

Le opzioni per convertire facilmente i dati di origine come JSON o CSV in un formato colonnare includono l'utilizzo di query CREATE TABLE AS o l'esecuzione di processi in AWS Glue.

- È possibile utilizzare query `CREATE TABLE AS` (CTAS) per convertire i dati in Parquet o ORC in un unico passaggio. Per un esempio, consulta Esempio: scrittura di risultati della query in un formato diverso nella pagina Esempi di query CTAS.

- Per informazioni sull'esecuzione di un AWS Glue processo per trasformare i dati CSV in Parquet, consulta la sezione «Trasformare i dati dal formato CSV al formato Parquet» nel post del blog AWS Big Data Build a data lake foundation with AWS Glue and Amazon S3. AWS Glue supporta l'utilizzo della stessa tecnica per convertire i dati CSV in ORC o i dati JSON in Parquet o ORC.

Partizionamento dei dati in Athena

Effettuando il partizionamento dei dati, è possibile limitare la quantità di dati scansionati da ogni query, migliorando così le prestazioni e riducendo i costi. in base a qualsiasi chiave di partizione. In genere è consigliabile partizionare i dati in base al tempo, spesso determinando uno schema di partizionamento multilivello. Ad esempio, un cliente che ha dati in entrata ogni ora potrebbe decidere di partizione per anno, mese, data e ora. Un altro cliente che ha dati provenienti da diverse origini ma che vengono caricati una volta al giorno, potrebbe eseguire la partizione in base all'identificatore dell'origine dati e alla data.

Athena può utilizzare partizioni in stile Apache Hive, i cui percorsi di dati contengono coppie chiave-valore collegate da simboli di uguale (ad esempio, `country=us/...` o `year=2021/month=01/day=26/...`). Pertanto, i percorsi includono sia i nomi delle chiavi di partizione che i valori rappresentati da ciascun percorso. Per caricare nuove partizioni Hive in una tabella partizionata, è possibile utilizzare il comando MSCK REPAIR TABLE, che funziona solo con partizioni in stile Hive.

Athena può utilizzare anche schemi di partizionamento in stile non Hive. Ad esempio, CloudTrail i log e i flussi di consegna Firehose utilizzano componenti di percorso separati per parti di data come. `data/2021/01/26/us/6fc7845e.json` Per le partizioni non in stile Hive, è possibile usare ALTER TABLE ADD PARTITION per aggiungere manualmente le partizioni.

Considerazioni e limitazioni

Quando si utilizza il partizionamento, tenere presente i seguenti punti:

- Se esegui una query su una tabella partizionata e specifichi la partizione nella clausola `WHERE`, Athena analizza i dati solo da quella partizione. Per ulteriori informazioni, consulta Ubicazione della tabella e partizioni.
- Se si esegue le query sui bucket Amazon S3 con un numero elevato di oggetti e i dati non sono partizionati, queste query possono influenzare i limiti del tasso di richieste `GET` in Amazon S3 e comportare eccezioni Amazon S3. Per evitare errori, partiziona i dati. Inoltre, considera di ottimizzare i tassi di richiesta Amazon S3. Per maggiori informazioni, consulta le Best practice per la creazione di modelli: ottimizzazione delle prestazioni di Amazon S3.

- Le posizioni delle partizioni da utilizzare con Athena devono utilizzare il protocollo `s3` (ad esempio, `s3://DOC-EXAMPLE-BUCKET/folder/`). In Athena, i percorsi che utilizzano altri protocolli (ad esempio, `s3a://DOC-EXAMPLE-BUCKET/folder/`) determineranno errori quando le query `MSCK REPAIR TABLE` vengono eseguite sulle tabelle contenenti.
- Assicurarsi che il percorso Amazon S3 sia in minuscolo invece che in camel case (ad esempio, `userid` invece di `userId`) Se il percorso S3 è in camel case, `MSCK REPAIR TABLE` non aggiunge le partizioni a AWS Glue Data Catalog. Per ulteriori informazioni, consulta MSCK REPAIR TABLE.
- Poiché `MSCK REPAIR TABLE` analizza sia una cartella che le relative sottocartelle per trovare uno schema di partizioni corrispondente, assicurarsi di conservare i dati per tabelle separate in gerarchie di cartelle separate. Ad esempio, supponiamo di avere dati per la tabella 1 in `s3://DOC-EXAMPLE-BUCKET1` e dati per la tabella 2 in. `s3://DOC-EXAMPLE-BUCKET1/table-2-data` Se entrambe le tabelle sono partizionate per stringa, `MSCK REPAIR TABLE` aggiungerà le partizioni per la tabella 2 alla tabella 1. Per evitare ciò, usa invece strutture di cartelle separate come `s3://DOC-EXAMPLE-BUCKET1` e. `s3://DOC-EXAMPLE-BUCKET2` Questo comportamento è coerente con Amazon EMR e Apache Hive.
- Se utilizzi il AWS Glue Data Catalog con Athena, consulta AWS Glue endpoint e quote per le quote di servizio sulle partizioni per account e per tabella.
  - Sebbene Athena supporti l'interrogazione di AWS Glue tabelle con 10 milioni di partizioni, Athena non è in grado di leggere più di 1 milione di partizioni in una singola scansione. In tali scenari, l'indicizzazione delle partizioni può essere utile. Per ulteriori informazioni, consulta l'articolo del blog AWS Big Data Migliora le prestazioni delle query di Amazon Athena utilizzando gli indici di AWS Glue Data Catalog partizione.
- Per richiedere un aumento della quota delle partizioni se utilizzi il AWS Glue Data Catalog, visita la console Service Quotas per. AWS Glue

Creazione e caricamento di una tabella con dati partizionati

Per creare una tabella con partizioni, utilizzare la clausola `PARTITIONED BY` nell'istruzione CREATE TABLE. La clausola `PARTITIONED BY` definisce le chiavi su cui partizionare i dati, come nell'esempio seguente. La clausola `LOCATION` specifica la posizione principale dei dati partizionati.

```
CREATE EXTERNAL TABLE users (
first string,
last string,
username string
```

```
)
PARTITIONED BY (id string)
STORED AS parquet
LOCATION 's3://DOC-EXAMPLE-BUCKET'
```

Dopo aver creato la tabella, è possibile caricare i dati nelle partizioni per l'interrogazione. Per le partizioni stile Hive, esegui MSCK REPAIR TABLE. Per le partizioni in stile non Hive, utilizza ALTER TABLE ADD PARTITION per aggiungere manualmente le partizioni.

Preparazione di dati in stile Hive e in stile non Hive per le query

Nelle sezioni seguenti viene illustrato come preparare i dati in stile Hive e non Hive per le query in Athena.

Scenario 1: dati archiviati su Amazon S3 in formato Hive

In questo scenario, le partizioni vengono archiviate in cartelle separate in Amazon S3. Ad esempio, ecco l'elenco parziale di campioni e impressioni prodotti dal comando `aws s3 ls`, che elenca gli oggetti S3 con un prefisso specificato:

```
aws s3 ls s3://elasticmapreduce/samples/hive-ads/tables/impressions/

    PRE dt=2009-04-12-13-00/
    PRE dt=2009-04-12-13-05/
    PRE dt=2009-04-12-13-10/
    PRE dt=2009-04-12-13-15/
    PRE dt=2009-04-12-13-20/
    PRE dt=2009-04-12-14-00/
    PRE dt=2009-04-12-14-05/
    PRE dt=2009-04-12-14-10/
    PRE dt=2009-04-12-14-15/
    PRE dt=2009-04-12-14-20/
    PRE dt=2009-04-12-15-00/
    PRE dt=2009-04-12-15-05/
```

Qui, i log vengono archiviati con il nome di colonna (dt) impostato in modo corrispondente a incrementi di giorno, ora e minuto. Quando si assegna un DDL con il percorso della cartella padre, lo schema e il nome della colonna partizionata, Athena è in grado di eseguire query sui dati in quelle sottocartelle.

Creazione della tabella

Per creare una tabella da questi dati, creare una partizione insieme a 'dt' come nella seguente istruzione DDL di Athena:

```
CREATE EXTERNAL TABLE impressions (
    requestBeginTime string,
    adId string,
    impressionId string,
    referrer string,
    userAgent string,
    userCookie string,
    ip string,
    number string,
    processId string,
    browserCookie string,
    requestEndTime string,
    timers struct<modelLookup:string, requestTime:string>,
    threadId string,
    hostname string,
    sessionId string)
PARTITIONED BY (dt string)
ROW FORMAT  serde 'org.apache.hive.hcatalog.data.JsonSerDe'
LOCATION 's3://elasticmapreduce/samples/hive-ads/tables/impressions/' ;
```

Questa tabella utilizza il serializzatore/deserializzatore JSON nativo di Hive per leggere i dati JSON archiviati in Amazon S3. Per ulteriori informazioni sui formati supportati, consulta Formati di SerDe e di dati supportati.

Esecuzione di MSCK REPAIR TABLE

Dopo aver eseguito la query `CREATE TABLE`, eseguire il comando `MSCK REPAIR TABLE` nell'editor di query Athena per caricare le partizioni, come nell'esempio seguente.

```
MSCK REPAIR TABLE impressions
```

Una volta eseguito il comando, i dati sono pronti per l'esecuzione della query.

Esecuzione di query sui dati

Eseguire la query sui dati dalla tabella di impressioni utilizzando la colonna di partizione. Ecco un esempio:

```
SELECT dt,impressionid FROM impressions WHERE dt<'2009-04-12-14-00' and
 dt>='2009-04-12-13-00' ORDER BY dt DESC LIMIT 100
```

Questa query dovrebbe restituire risultati simili ai seguenti:

```
2009-04-12-13-20    ap3HcVKAWfXtgIPu6WpuUfAfL0DQEc
2009-04-12-13-20    17uchtodoS9kdeQP1x0XThKl5IuRsV
2009-04-12-13-20    JOUf1SCtRwviGw8sVcghqE5h0nkgtp
2009-04-12-13-20    NQ2XP0J0dvVbCXJ0pb4XvqJ5A4QxxH
2009-04-12-13-20    fFAItiBMsgqro9kRdIwbeX60SROaxr
2009-04-12-13-20    V4og4R9W6G3QjHHwF7gI1cSqig5D1G
2009-04-12-13-20    hPEPtBwk45msmwWTxPVVo1kVu4v11b
2009-04-12-13-20    v0SkfxegheD90gp31UCr6FplnKpx6i
2009-04-12-13-20    1iD9odVgOIi4QWkwHMcOhmwTkWDKfj
2009-04-12-13-20    b31tJiIA25CK8eDHQrHnbcknfSndUk
```

Scenario 2: i dati non sono partizionati in formato Hive

Nell'esempio seguente, il comando `aws s3 ls` mostra i registri ELB archiviati in Amazon S3. Notare come il layout dei dati non utilizza coppie `key=value` e quindi non è in formato Hive. (L'opzione `--recursive` per il comando `aws s3 ls` specifica che siano elencati tutti i file o gli oggetti nella directory o nel prefisso specificati).

```
aws s3 ls s3://athena-examples-myregion/elb/plaintext/ --recursive

2016-11-23 17:54:46   11789573 elb/plaintext/2015/01/01/part-r-00000-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:46    8776899 elb/plaintext/2015/01/01/part-r-00001-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:46    9309800 elb/plaintext/2015/01/01/part-r-00002-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:47    9412570 elb/plaintext/2015/01/01/part-r-00003-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:47   10725938 elb/plaintext/2015/01/01/part-r-00004-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:46    9439710 elb/plaintext/2015/01/01/part-r-00005-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:47          0 elb/plaintext/2015/01/01_$folder$
2016-11-23 17:54:47    9012723 elb/plaintext/2015/01/02/part-r-00006-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:47    7571816 elb/plaintext/2015/01/02/part-r-00007-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
```

```
2016-11-23 17:54:47    9673393 elb/plaintext/2015/01/02/part-r-00008-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:48   11979218 elb/plaintext/2015/01/02/part-r-00009-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:48    9546833 elb/plaintext/2015/01/02/part-r-00010-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:48   10960865 elb/plaintext/2015/01/02/part-r-00011-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:48          0 elb/plaintext/2015/01/02_$folder$
2016-11-23 17:54:48   11360522 elb/plaintext/2015/01/03/part-r-00012-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:48   11211291 elb/plaintext/2015/01/03/part-r-00013-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:48    8633768 elb/plaintext/2015/01/03/part-r-00014-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:49   11891626 elb/plaintext/2015/01/03/part-r-00015-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:49    9173813 elb/plaintext/2015/01/03/part-r-00016-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:49   11899582 elb/plaintext/2015/01/03/part-r-00017-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:49          0 elb/plaintext/2015/01/03_$folder$
2016-11-23 17:54:50    8612843 elb/plaintext/2015/01/04/part-r-00018-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:50   10731284 elb/plaintext/2015/01/04/part-r-00019-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:50    9984735 elb/plaintext/2015/01/04/part-r-00020-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:50    9290089 elb/plaintext/2015/01/04/part-r-00021-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:50    7896339 elb/plaintext/2015/01/04/part-r-00022-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:51    8321364 elb/plaintext/2015/01/04/part-r-00023-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:51          0 elb/plaintext/2015/01/04_$folder$
2016-11-23 17:54:51    7641062 elb/plaintext/2015/01/05/part-r-00024-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:51   10253377 elb/plaintext/2015/01/05/part-r-00025-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:51    8502765 elb/plaintext/2015/01/05/part-r-00026-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:51   11518464 elb/plaintext/2015/01/05/part-r-00027-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
```

```
2016-11-23 17:54:51    7945189 elb/plaintext/2015/01/05/part-r-00028-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:51    7864475 elb/plaintext/2015/01/05/part-r-00029-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:51          0 elb/plaintext/2015/01/05_$folder$
2016-11-23 17:54:51   11342140 elb/plaintext/2015/01/06/part-r-00030-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:51    8063755 elb/plaintext/2015/01/06/part-r-00031-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:52    9387508 elb/plaintext/2015/01/06/part-r-00032-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:52    9732343 elb/plaintext/2015/01/06/part-r-00033-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:52   11510326 elb/plaintext/2015/01/06/part-r-00034-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:52    9148117 elb/plaintext/2015/01/06/part-r-00035-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:52          0 elb/plaintext/2015/01/06_$folder$
2016-11-23 17:54:52    8402024 elb/plaintext/2015/01/07/part-r-00036-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:52    8282860 elb/plaintext/2015/01/07/part-r-00037-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:52   11575283 elb/plaintext/2015/01/07/part-r-00038-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:53    8149059 elb/plaintext/2015/01/07/part-r-00039-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:53   10037269 elb/plaintext/2015/01/07/part-r-00040-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:53   10019678 elb/plaintext/2015/01/07/part-r-00041-ce65fca5-
d6c6-40e6-b1f9-190cc4f93814.txt
2016-11-23 17:54:53          0 elb/plaintext/2015/01/07_$folder$
2016-11-23 17:54:53          0 elb/plaintext/2015/01_$folder$
2016-11-23 17:54:53          0 elb/plaintext/2015_$folder$
```

Esecuzione di ALTER TABLE ADD PARTITION

Poiché i dati non sono in formato Hive, non è possibile utilizzare il comando `MSCK REPAIR TABLE` per aggiungere le partizioni alla tabella dopo averla creata. Invece, è possibile utilizzare il comando ALTER TABLE ADD PARTITION per aggiungere manualmente ogni partizione. Ad esempio, per caricare i dati in s3://athena-examples-*regione*/elb/plaintext/2015/01/01/, è possibile eseguire la query riportata. Si noti che non è richiesta una colonna di partizione separata per ogni cartella Amazon S3 e che il valore della chiave di partizione può essere diverso dalla chiave Amazon S3.

```
ALTER TABLE elb_logs_raw_native_part ADD PARTITION (dt='2015-01-01') location 's3://
athena-examples-us-west-1/elb/plaintext/2015/01/01/'
```

Se esiste già una partizione, viene visualizzato l'errore Partizione già esistente. Per evitare questo errore, è possibile utilizzare la clausola `IF NOT EXISTS`. Per ulteriori informazioni, consulta ALTER TABLE ADD PARTITION. Per rimuovere una partizione, è possibile utilizzare ALTER TABLE DROP PARTITION.

Proiezione delle partizioni

Per evitare di dover gestire le partizioni, è possibile utilizzare la proiezione delle partizioni. La proiezione delle partizioni è un'opzione per tabelle altamente partizionate la cui struttura è nota in anticipo. Nella proiezione delle partizioni, i valori e le posizioni delle partizioni vengono calcolati dalle proprietà della tabella configurate piuttosto che dalla lettura da un repository di metadati. Poiché i calcoli in memoria sono più veloci della ricerca remota, l'uso della proiezione delle partizioni può ridurre significativamente i tempi di esecuzione delle query.

Per ulteriori informazioni, consulta Proiezione delle partizioni con Amazon Athena.

Altre risorse

- Per informazioni sulle opzioni di partizionamento per i dati Firehose, vedere. Esempio di Amazon Data Firehose
- È inoltre possibile automatizzare l'aggiunta di partizioni utilizzando il driver JDBC.
- È possibile utilizzare CTAS e INSERT INTO per partizionare un set di dati. Per ulteriori informazioni, consulta Utilizzo di CTAS e INSERT INTO per ETL e analisi dei dati.

Proiezione delle partizioni con Amazon Athena

È possibile utilizzare la proiezione delle partizioni in Athena per velocizzare l'elaborazione delle query di tabelle altamente partizionate e automatizzare la gestione delle partizioni.

Durante la proiezione delle partizioni, Athena calcola i valori e le posizioni delle partizioni utilizzando le proprietà della tabella configurate direttamente sulla tabella in AWS Glue. Le proprietà della tabella consentono ad Athena di "proiettare", o determinare, le informazioni sulla partizione necessarie anziché eseguire in AWS Glue Data Catalog una ricerca di metadati più dispendiosa in termini di tempo. Poiché le operazioni in memoria sono spesso più veloci delle operazioni remote, la proiezione

delle partizioni può ridurre il runtime delle query su tabelle altamente partizionate. A seconda delle caratteristiche specifiche della query e dei dati sottostanti, la proiezione delle partizioni può ridurre significativamente il runtime delle query vincolate al recupero dei metadati delle partizioni.

Pruning e proiezione di tabelle fortemente partizionate

Il pruning delle partizioni raccoglie i metadati e li assegna solo alle partizioni che si applicano alla query. Questo spesso accelera le query. Athena utilizza il pruning delle partizioni per tutte le tabelle con colonne di partizione, incluse quelle configurate per la proiezione delle partizioni.

Normalmente, durante l'elaborazione delle interrogazioni, Athena effettua `GetPartitions` una chiamata a prima di eseguire AWS Glue Data Catalog l'eliminazione delle partizioni. Se una tabella ha un numero elevato di partizioni, l'utilizzo di `GetPartitions` può influire negativamente sulle prestazioni. Per evitare ciò, è possibile utilizzare la proiezione delle partizioni. La proiezione delle partizioni permette ad Athena di evitare di chiamare `GetPartitions` perché la configurazione della proiezione delle partizioni fornisce ad Athena tutte le informazioni necessarie per costruire le partizioni stesse.

Utilizzo della proiezione delle partizioni

Per utilizzare la proiezione delle partizioni, specificate gli intervalli di valori di partizione e i tipi di proiezione per ogni colonna di partizione nelle proprietà della tabella nel metastore Hive o nel metastore Hive esterno. AWS Glue Data Catalog Queste proprietà personalizzate nella tabella permettono ad Athena di sapere quali modelli di partizione aspettarsi quando viene eseguita una query sulla tabella. Durante l'esecuzione delle query, Athena utilizza queste informazioni per proiettare i valori delle partizioni invece di recuperarli dal metastore Hive o esterno. AWS Glue Data Catalog Questo non solo riduce i tempi di esecuzione delle query, ma automatizza anche la gestione delle partizioni perché elimina la necessità di creare manualmente partizioni in Athena, AWS Glue o nel metastore Hive esterno.

> **Important**
>
> L'abilitazione della proiezione delle partizioni su una tabella fa sì che Athena ignori tutti i metadati di partizione registrati nella tabella nel metastore o Hive. AWS Glue Data Catalog

Casi d'uso

Gli scenari in cui la proiezione delle partizioni è utile sono i seguenti:

- Le query su una tabella altamente partizionata non vengono completate rapidamente come si desidera.
- È possibile aggiungere regolarmente partizioni alle tabelle quando vengono create nuove partizioni di data o ora nei dati. Con la proiezione delle partizioni, è possibile configurare intervalli di date relativi che possono essere utilizzati all'arrivo di nuovi dati.
- Si dispone di dati altamente partizionati in Amazon S3. Non è pratico modellare i dati nel tuo AWS Glue Data Catalog metastore o in Hive e le tue query ne leggono solo piccole parti.

Strutture di partizioni proiettabili

La proiezione delle partizioni è configurata più facilmente quando le partizioni seguono un modello prevedibile come, a titolo esemplificativo, il seguente:

- Interi: qualsiasi sequenza continua di numeri interi come `[1, 2, 3, 4, ..., 1000]` o `[0500, 0550, 0600, ..., 2500]`.
- Date: qualsiasi sequenza continua di date o date e orari, ad esempio `[20200101, 20200102, ..., 20201231]` o `[1-1-2020 00:00:00, 1-1-2020 01:00:00, ..., 12-31-2020 23:00:00]`.
- Valori enumerati: un insieme finito di valori enumerati come codici aeroportuali o. Regioni AWS
- Servizio AWS logs: i Servizio AWS log hanno in genere una struttura nota il cui schema di partizione è possibile specificare e AWS Glue che Athena può quindi utilizzare per la proiezione delle partizioni.

Personalizzazione del modello di percorso della partizione

Per impostazione predefinita, Athena crea le posizioni delle partizioni utilizzando il modulo `s3://DOC-EXAMPLE-BUCKET/<table-root>/partition-col-1=<partition-col-1-val>/partition-col-2=<partition-col-2-val>/`, ma se i dati sono organizzati in modo diverso, Athena offre un meccanismo per personalizzare questo modello di percorso. Per le fasi, consulta Specifica dei percorsi di storage S3 personalizzati.

Considerazioni e limitazioni

Tieni presente le seguenti considerazioni:

- La proiezione delle partizioni elimina la necessità di specificare manualmente le partizioni AWS Glue o un metastore Hive esterno.

- Quando abilitate la proiezione delle partizioni su una tabella, Athena ignora tutti i metadati di partizione nel metastore Hive esterno AWS Glue Data Catalog o nel metastore Hive per quella tabella.
- Se una partizione proiettata non esiste in Amazon S3, Athena continuerà a proiettare la partizione. Athena non genera un errore, ma non viene restituito alcun dato. Tuttavia, se troppe partizioni sono vuote, le prestazioni possono essere inferiori rispetto alle partizioni tradizionali. AWS Glue Se più della metà delle partizioni proiettate sono vuote, si consiglia di utilizzare partizioni tradizionali.
- Le query per valori che superano i limiti di intervallo definiti per la proiezione della partizione non restituiscono un errore. La query invece viene eseguita, ma restituisce zero righe. Ad esempio, se si dispone di dati relativi al tempo che iniziano nel 2020 e sono definiti come `'projection.timestamp.range'='2020/01/01,NOW'`, una query come `SELECT * FROM table-name WHERE timestamp = '2019/02/02'` verrà completata correttamente, ma restituirà zero righe.
- La proiezione delle partizioni è utilizzabile solo quando la tabella viene interrogata tramite Athena. Se la stessa tabella viene letta attraverso un altro servizio, ad esempio Amazon Redshift Spectrum, Athena per Spark o Amazon EMR, vengono utilizzati i metadati della partizione standard.
- Poiché la proiezione delle partizioni è una funzionalità solo DML, `SHOW PARTITIONS` non elenca le partizioni proiettate da Athena ma non registrate nel catalogo o nel metastore Hive esterno. AWS Glue
- Athena non utilizza le proprietà della tabella delle viste come configurazione per la proiezione delle partizioni. Per aggirare questa limitazione, configurare e abilitare la proiezione della partizione nelle proprietà della tabella per le tabelle a cui fanno riferimento le viste.
- I filtri dati di Lake Formation non possono essere utilizzati con la proiezione delle partizioni nella versione 2 del motore Athena.

Video

Il video seguente mostra come utilizzare la proiezione delle partizioni per migliorare le prestazioni delle vostre query in Athena.

Proiezione delle partizioni con Amazon Athena

Argomenti

- Impostazione della proiezione delle partizioni
- Tipi supportati per la proiezione delle partizioni

- Partizionamento ID dinamico
- Esempio di Amazon Data Firehose

Impostazione della proiezione delle partizioni

L'impostazione della proiezione delle partizioni nelle proprietà di una tabella è un processo in due fasi:

1. Specificare gli intervalli di dati e i modelli pertinenti per ogni colonna di partizione oppure utilizzare un modello personalizzato.
2. Attivare la proiezione delle partizioni per la tabella.

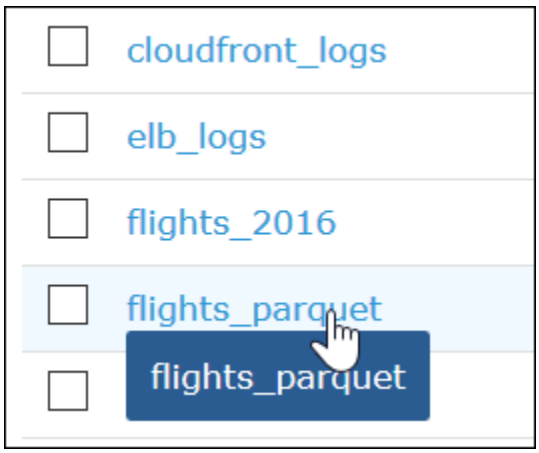
> **Note**
>
> Prima di aggiungere le proprietà di proiezione delle partizioni a una tabella esistente, la colonna di partizione per la quale stai impostando tali proprietà deve già esistere nello schema della tabella. Se la colonna di partizione non esiste ancora, è necessario aggiungere manualmente una colonna di partizione alla tabella esistente. AWS Glue non esegue questo passaggio automaticamente.

Questa sezione mostra come impostare le proprietà della tabella per AWS Glue. Per impostarli, puoi utilizzare la AWS Glue console, le CREATE TABLE query Athena o le operazioni. AWS Glue API La procedura seguente mostra come impostare le proprietà nella AWS Glue console.

Per configurare e abilitare la proiezione delle partizioni utilizzando la console AWS Glue

1. Accedere AWS Management Console e aprire la AWS Glue console all'indirizzo https://console.aws.amazon.com/glue/.
2. Scegliere la scheda Tabelle.

   Nella scheda Tabelle è possibile modificare le tabelle esistenti oppure scegliere Aggiungi tabelle per crearne di nuove. Per informazioni sull'aggiunta manuale di tabelle o con un crawler, consulta Utilizzo delle tabelle nella console AWS Glue nella Guida per gli sviluppatori di AWS Glue .
3. Nell'elenco delle tabelle scegliere il collegamento per la tabella che si desidera modificare.

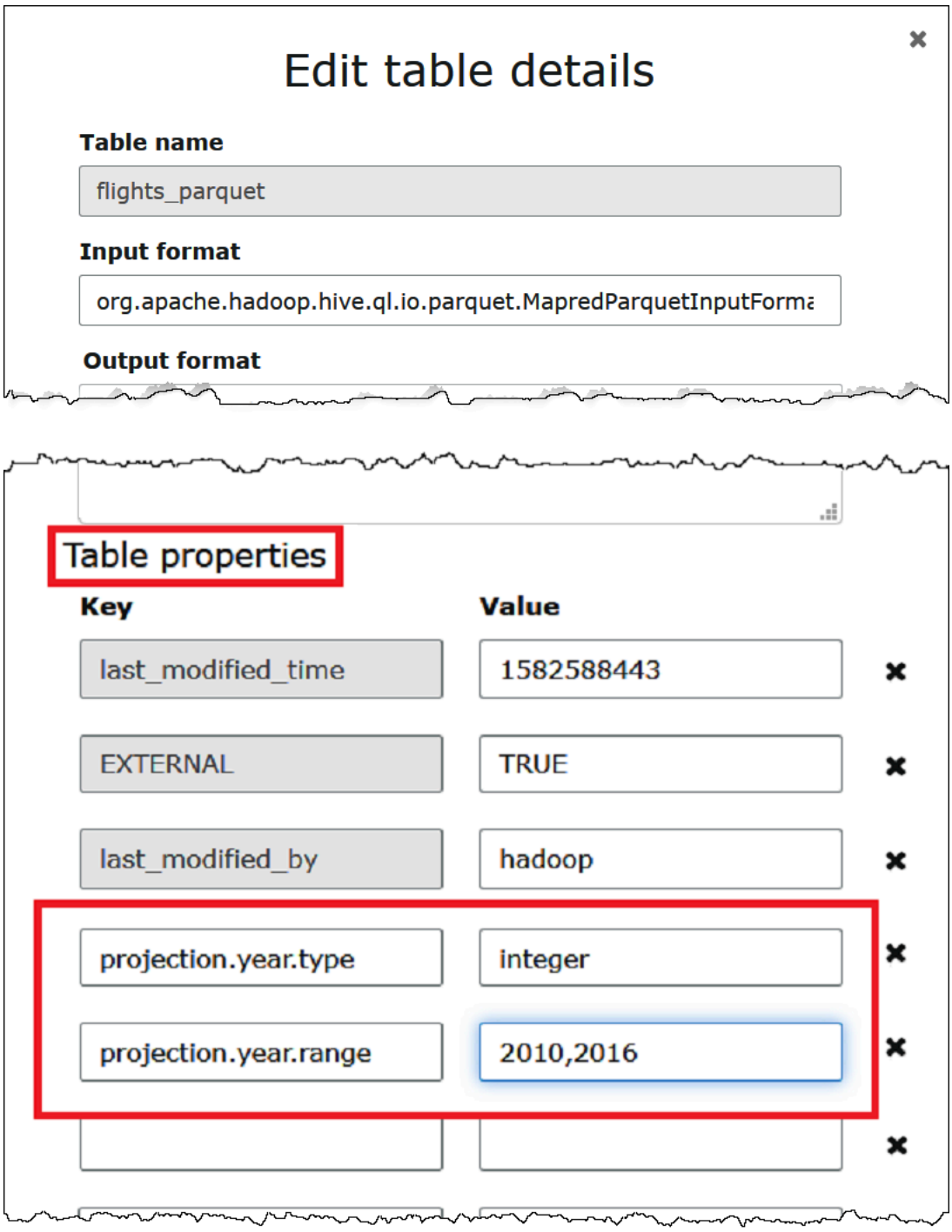


4. Scegli Actions (Operazioni), Edit (Modifica).
5. Nella pagina Edit table (Modifica tabella), nella sezione Table properties (Proprietà della tabella), per ogni colonna partizionata, aggiungi la seguente coppia chiave-valore:

   a. Per Chiave, aggiungere `projection.`*`columnName`*`.type`.

   b. Per Valore, aggiungere uno dei tipi supportati: `enum`, `integer`, `date` o `injected`. Per ulteriori informazioni, consulta Tipi supportati per la proiezione delle partizioni.
6. Seguendo le indicazioni fornite in Tipi supportati per la proiezione delle partizioni, aggiungere ulteriori coppie chiave-valore in base ai requisiti di configurazione.

   La configurazione della tabella di esempio seguente configura la colonna `year` per la proiezione delle partizioni, limitando i valori che possono essere restituiti a un intervallo compreso tra il 2010 e il 2016.

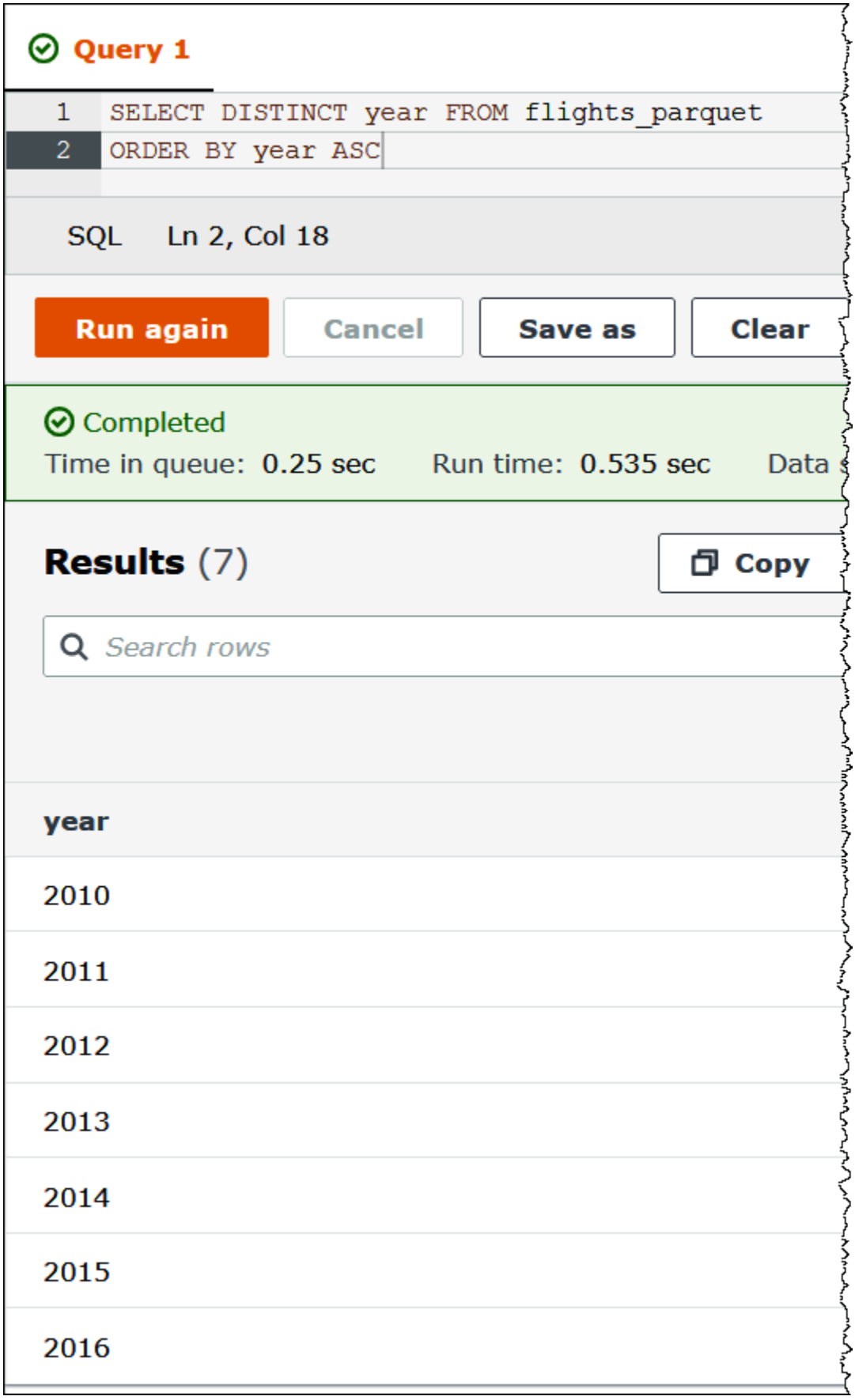


7. Aggiungere una coppia chiave-valore per abilitare la proiezione delle partizioni. Per Chiave, immettere `projection.enabled`, e per il relativo valore, immettere `true`.

> **Note**
>
> È possibile disattivare la proiezione delle partizioni su questa tabella in qualsiasi momento impostando `projection.enabled` su `false`.

8. Quando hai terminato, seleziona Save (Salva).
9. Nell'editor di query Athena, eseguire una query sulle colonne configurate per la tabella.

   La query di esempio seguente utilizza `SELECT DISTINCT` per restituire i valori univoci della colonna `year`. Il database contiene dati dal 1987 al 2016, ma la proprietà `projection.year.range` limita i valori restituiti agli anni dal 2010 al 2016.


Query 1
1 SELECT DISTINCT year FROM flights_parquet
2 ORDER BY year ASC
SQL Ln 2, Col 18
Run again
Cancel
Save as
Clear
Completed
Time in queue: 0.25 sec
Run time: 0.535 sec
Data
Results (7)
Copy
Search rows
year
2010
2011
2012
2013
2014
2015
2016

> **Note**
>
> Se si imposta su `projection.enabled` su `true` ma non si riesce a configurare una o più colonne di partizione, viene visualizzato un messaggio di errore analogo al seguente: `HIVE_METASTORE_ERROR: Table database_name.table_name is configured for partition projection, but the following partition columns are missing projection configuration: [column_name] (table database_name.table_name).`

Specifica dei percorsi di storage S3 personalizzati

Quando modifichi le proprietà della tabella in AWS Glue, puoi anche specificare un modello di percorso Amazon S3 personalizzato per le partizioni proiettate. Un modello personalizzato consente ad Athena di mappare correttamente i valori delle partizioni alle posizioni dei file Amazon S3 personalizzate che non seguono un modello `.../column=value/...` tipico.

L'utilizzo di un modello personalizzato è facoltativo. Tuttavia, se si utilizza un modello personalizzato, il modello deve contenere un segnaposto per ogni colonna di partizione. Le posizioni basate su modelli devono terminare con una barra in avanti in modo che i file di dati partizionati risiedano in una "cartella" per partizione.

Per specificare un modello di posizione della partizione personalizzato

1. Seguendo i passaggi per configurare e abilitare la proiezione delle partizioni utilizzando la AWS Glue console, aggiungi un'ulteriore coppia chiave-valore che specifica un modello personalizzato come segue:

   a. In Chiave, inserire `storage.location.template`.
   b. In Valore, specificare una posizione che includa un segnaposto per ogni colonna di partizione. Assicurarsi che ogni segnaposto (e il percorso S3 stesso) sia terminato da una singola barra in avanti.

      I valori del modello di esempio seguenti presuppongono una tabella con colonne di partizione `a`, `b` e `c`.

      ```
      s3://DOC-EXAMPLE-BUCKET/table_root/a=${a}/${b}/some_static_subdirectory/${c}/
      ```

```
s3://DOC-EXAMPLE-BUCKET/table_root/c=${c}/${b}/some_static_subdirectory/${a}/
${b}/${c}/${c}/
```

Per la stessa tabella, il valore del modello di esempio seguente non è valido perché non contiene segnaposto per la colonna `c`.

```
s3://DOC-EXAMPLE-BUCKET/table_root/a=${a}/${b}/some_static_subdirectory/
```

2. Scegli Applica.

Tipi supportati per la proiezione delle partizioni

Una tabella può avere qualsiasi combinazione di `enum`, `integer`, `date`, o tipi di colonna di partizione `injected`.

Tipi di enumerazione

Utilizzate il `enum` tipo per le colonne di partizione i cui valori sono membri di un set enumerato (ad esempio, codici aeroportuali o). Regioni AWS

Definire le proprietà della partizione nella tabella come segue:

Nome proprietà	Valori di esempio	Descrizione
`projection.` *`columnName`* `.type`	`enum`	Obbligatorio. Tipo di proiezione da utilizzare per la colonna *`columnName`* . Il valore deve essere `enum` (senza distinzione tra maiuscole e minuscole) per segnalare l'uso del tipo enum. È consentito lo spazio bianco iniziale e finale.
`projection.` *`columnName`* `.values`	`A,B,C,D,E,F,G,Unknown`	Obbligatorio. Elenco separato da virgole dei valori delle partizioni enumerate per la colonna *`columnName`* .

Nome proprietà	Valori di esempio	Descrizione
		Qualsiasi spazio bianco è considerato parte di un valore enum.

> **Note**
>
> Come best practice si consiglia di limitare l'uso delle proiezioni delle partizioni basate su `enum` a poche dozzine o meno. Sebbene non esista un limite specifico per `enum` le proiezioni, la dimensione totale dei metadati della tabella non può superare il AWS Glue limite di circa 1 MB quando viene compressa con gzip. Si noti che questo limite è condiviso tra le parti chiave della tabella, come i nomi delle colonne, la posizione, il formato di archiviazione e altri. Se utilizzi più di poche dozzine di ID univoci in `enum`, prendi in considerazione un approccio alternativo, ad esempio l'inserimento in un numero minore di valori univoci in un campo surrogato. Negoziando la cardinalità, puoi controllare il numero di valori univoci nel campo `enum`.

Tipo intero

Utilizzare il tipo intero per le colonne di partizione i cui valori possibili sono interpretabili come numeri interi all'interno di un intervallo definito. Le colonne intere proiettate sono attualmente limitate all'intervallo firmato Java (da -2^{63} a $2^{63}-1$ incluso).

Nome proprietà	Valori di esempio	Descrizione
`projection.` *`columnName`* `.type`	`integer`	Obbligatorio. Tipo di proiezione da utilizzare per la colonna *`columnName`* . Il valore deve essere `integer` (senza distinzione tra maiuscole e minuscole) per segnalare l'uso del tipo intero. È consentito lo spazio bianco iniziale e finale.

Nome proprietà	Valori di esempio	Descrizione

`projection.` *`columnName`* `.range`	`0,10`
	
	`-1,8675309`
	
	`0001,9999`
		Obbligatorio. Elenco separato da virgole a due elementi che fornisce i valori di intervallo minimo e massimo da restituire dalle query sulla colonna *`columnName`* . Tieni presente che i valori devono essere separati da virgole, non da trattini. Questi valori sono inclusivi, possono essere negativi e possono avere zeri iniziali. È consentito lo spazio bianco iniziale e finale.

`projection.` *`columnName`* `.interval`	`1`
	
	`5`
		Facoltativo. Un numero intero positivo che specifica l'intervallo tra i valori di partizione successivi per la colonna *`columnName`* . Ad esempio, un valore `range` di "1,3" con un valore `interval` di "1" produce i valori 1, 2 e 3. Lo stesso valore `range` con un valore `interval` di "2" produce i valori 1 e 3, saltando 2. È consentito lo spazio bianco iniziale e finale. Il valore di default è 1.

Nome proprietà	Valori di esempio	Descrizione

`projection.` *`columnName`* `.digits`	`1`
	
	`5`
		Facoltativo. Un numero intero positivo che specifica il numero di cifre da includere nella rappresentazione finale del valore della partizione per la colonna *`columnName`* . Ad esempio, un valore `range` di "1,3" che ha un valore `digits` di "1" produce i valori 1, 2 e 3. Lo stesso valore `range` con un valore `digits` di "2" produce i valori 01, 02 e 03. È consentito lo spazio bianco iniziale e finale. Il valore predefinito non è un numero statico di cifre e nessuno zero iniziale.

Tipo di data

Utilizzare il tipo di data per le colonne di partizione i cui valori sono interpretabili come date (con orari facoltativi) all'interno di un intervallo definito.

> **Important**
>
> Le colonne della data prevista vengono generate in ora UTC (Coordinated Universal Time) al momento dell'esecuzione della query.

Nome proprietà	Valori di esempio	Descrizione
`projection.` *`columnName`* `.type`	`date`	Obbligatorio. Tipo di proiezione da utilizzare per la colonna *`columnName`* . Il valore deve

Nome proprietà	Valori di esempio	Descrizione
		essere `date` (senza distinzione tra maiuscole e minuscole) per segnalare l'utilizzo del tipo di data. È consentito lo spazio bianco iniziale e finale.

`projection.` *`columnName`* `.range`	`201701,201812`
	
	`01-01-2010,12-31-2018`
	
	`NOW-3YEARS,NOW`
	
	`201801,NOW+1MONTH`
		Obbligatorio. Elenco separato da virgole a due elementi che fornisce i valori `range` minimo e massimo per la colonna *`columnName`* . Questi valori sono inclusivi e possono utilizzare qualsiasi formato compatibile con i tipi di data `java.time.*` Java. Entrambi i valori minimo e massimo devono utilizzare lo stesso formato. Il formato specificato nella proprietà `.format` deve corrispondere al formato utilizzato per questi valori.
		
		Questa colonna può anche contenere stringhe di data relative, formattate in questo modello di espressione regolare:
		
		`\s*NOW\s*(([\+\-])\s*([0-9]+)\s*(YEARS?|MONTHS?|WEEKS?|DAYS?|HOURS?|MINUTES?|SECONDS?)\s*)?`
		
		Gli spazi bianchi sono consentiti, ma nei valori letterali della data sono considerati parte delle stringhe di data stesse.

Nome proprietà	Valori di esempio	Descrizione

`projection.` *`columnName`* `.format`	`yyyyMM`
	
	`dd-MM-yyyy`
	
	`dd-MM-yyyy-HH-mm-ss`
		Obbligatorio. Una stringa di formato data basata sul formato di data Java. DateTimeFormatter Può essere qualsiasi tipo `Java.time.*` supportato.

`projection.` *`columnName`* `.interval`	`1`
	
	`5`
		Un numero intero positivo che specifica l'intervallo tra i valori di partizione successivi per la colonna *`columnName`* . Ad esempio, un valore `range` di `2017-01,2018-12` con un valore `interval` di `1` e un valore `interval.unit` di `MONTHS` produce i valori 2017-01, 2017-02, 2017-03 e così via. Lo stesso valore `range` con un valore `interval` di 2 e un valore `interval.unit` di `MONTHS` produce i valori 2017-01, 2017-03, 2017-05 e così via. È consentito lo spazio bianco iniziale e finale.
		
		Quando le date sono fornite con la precisione di un singolo giorno o di un mese, `interval` è facoltativo e il valore predefinito è 1 giorno o 1 mese, rispettivamente. In caso contrario, `interval` è richiesto.

Nome proprietà	Valori di esempio	Descrizione

`projection.` *`columnName`* `.interval.unit`	`YEARS`
	`MONTHS`
	`WEEKS`
	`DAYS`
	`HOURS`
	`MINUTES`
	`SECONDS`
	`MILLIS`
		Una parola unitaria di tempo che rappresenta la forma serializzata di a ChronoUnit. I valori possibili sono `YEARS`, `MONTHS`, `WEEKS`, `DAYS`, `HOURS`, `MINUTES`, `SECONDS` o `MILLIS`. I valori non rispettano la distinzione tra maiuscole e minuscole.
		
		Quando le date sono fornite con la precisione di un singolo giorno o di un mese, `interval.unit` è facoltativo e il valore predefinito è 1 giorno o 1 mese, rispettivamente. In caso contrario, `interval.unit` è richiesto.

Tipo iniettato

Utilizzare il tipo iniettato per le colonne di partizione con valori possibili che non possono essere generati proceduralmente all'interno di un intervallo logico, ma che sono forniti nella clausola `WHERE` di una query come valore singolo.

È importante tenere a mente i seguenti punti:

- Le query sulle colonne iniettate hanno esito negativo se non viene fornita un'espressione di filtro per ogni colonna iniettata.
- Le query con più valori per un'espressione filtro su una colonna iniettata hanno esito positivo solo se i valori sono disgiunti.
- Solo le colonne di tipo `string` sono supportate.

Nome proprietà	Valore	Descrizione
`projection.` *`columnName`* `.type`	`inje`	Obbligatorio. Tipo di proiezione da utilizzare per la colonna *`columnName`* . È supportato solo il tipo `string`. Il valore specificato deve essere `injected` (senza

Nome proprietà	Valore	Descrizione
		distinzione tra maiuscole e minuscole). È consentito lo spazio bianco iniziale e finale.

Per ulteriori informazioni, consulta Utilizzo del tipo di proiezione `injected`.

Partizionamento ID dinamico

Quando i dati sono partizionati in base a una proprietà con cardinalità elevata o quando i valori non possono essere conosciuti in anticipo, è possibile utilizzare il tipo di proiezione `injected`. Esempi di tali proprietà sono i nomi utente e gli ID di dispositivi o prodotti. Quando si utilizza il tipo di proiezione `injected` per configurare una chiave di partizione, Athena utilizza i valori della query stessa per calcolare l'insieme di partizioni che verranno lette.

Affinché Athena sia in grado di eseguire una query su una tabella con una chiave di partizione configurata con il tipo di proiezione `injected`, deve essere vero quanto segue:

- La query deve includere almeno un valore per la chiave di partizione.
- I valori devono essere letterali o espressioni che possono essere valutate senza leggere alcun dato.

Se uno di questi criteri non viene soddisfatto, la query ha esito negativo e viene visualizzato il seguente errore:

CONSTRAINT_VIOLATION: per la colonna di partizione proiettata iniettata *`column_name`*, la clausola WHERE deve contenere solo condizioni di uguaglianza statiche e deve essere presente almeno una di queste condizioni.

Utilizzo del tipo di proiezione **`injected`**

Immagina di avere un set di dati composto da eventi provenienti da dispositivi IoT partizionati sugli ID dei dispositivi. Questo esempio presenta le caratteristiche seguenti:

- Gli ID dei dispositivi vengono generati in modo casuale.
- Nuovi dispositivi vengono forniti frequentemente.
- Attualmente i dispositivi ammontano a centinaia di migliaia e in futuro saranno milioni.

Questo set di dati è difficile da gestire utilizzando i metastore tradizionali. È difficile mantenere sincronizzate le partizioni tra l'archiviazione di dati e il metastore e il filtraggio delle partizioni può essere lento durante la pianificazione delle query. Tuttavia, se si configura una tabella per utilizzare la proiezione delle partizioni e si utilizza il tipo di proiezione `injected`, si ottengono due vantaggi: non è necessario gestire le partizioni nel metastore e le query non devono cercare i metadati delle partizioni.

L'esempio di `CREATE TABLE` seguente crea una tabella per il set di dati degli eventi del dispositivo appena descritto. La tabella utilizza il tipo di proiezione iniettata.

```
CREATE EXTERNAL TABLE device_events (
   event_time TIMESTAMP,
  data STRING,
  battery_level INT
)
PARTITIONED BY (
  device_id STRING
)
LOCATION "s3://DOC-EXAMPLE-BUCKET/prefix/"
TBLPROPERTIES (
  "projection.enabled" = "true",
  "projection.device_id.type" = "injected",
  "storage.location.template" = "s3://DOC-EXAMPLE-BUCKET/prefix/${device_id}"
)
```

La seguente query di esempio cerca il numero di eventi ricevuti da tre dispositivi specifici nell'arco di 12 ore.

```
SELECT device_id, COUNT(*) AS events
FROM device_events
WHERE device_id IN (
  '4a770164-0392-4a41-8565-40ed8cec737e',
  'f71d12cf-f01f-4877-875d-128c23cbde17',
  '763421d8-b005-47c3-ba32-cc747ab32f9a'
)
AND event_time BETWEEN TIMESTAMP '2023-11-01 20:00' AND TIMESTAMP '2023-11-02 08:00'
GROUP BY device_id
```

Quando si esegue questa query, Athena vede i tre valori per la chiave di partizione `device_id` e li usa per calcolare le posizioni delle partizioni. Athena utilizza il valore della proprietà `storage.location.template` per generare le seguenti posizioni:

- s3://DOC-EXAMPLE-BUCKET/*prefix*/4a770164-0392-4a41-8565-40ed8cec737e
- s3://DOC-EXAMPLE-BUCKET/*prefix*/f71d12cf-f01f-4877-875d-128c23cbde17
- s3://DOC-EXAMPLE-BUCKET/*prefix*/763421d8-b005-47c3-ba32-cc747ab32f9a

Se si omette la proprietà `storage.location.template` dalla configurazione di proiezione della partizione, Athena utilizza il partizionamento in stile Hive per proiettare le posizioni delle partizioni in base al valore in `LOCATION` (ad esempio, `s3://DOC-EXAMPLE-BUCKET/prefix/device_id=4a770164-0392-4a41-8565-40ed8cec737e`).

Esempio di Amazon Data Firehose

Quando usi Firehose per fornire dati ad Amazon S3, la configurazione predefinita scrive oggetti con chiavi simili al seguente esempio:

```
s3://DOC-EXAMPLE-BUCKET/prefix/yyyy/MM/dd/HH/file.extension
```

Per creare una tabella Athena che trovi automaticamente le partizioni al momento della query, invece di doverle aggiungere all'arrivo di nuovi dati, puoi utilizzare la AWS Glue Data Catalog proiezione delle partizioni.

L'`CREATE TABLE`esempio seguente utilizza la configurazione Firehose predefinita.

```
CREATE EXTERNAL TABLE my_ingested_data (
 ...
)
...
PARTITIONED BY (
 datehour STRING
)
LOCATION "s3://DOC-EXAMPLE-BUCKET/prefix/"
TBLPROPERTIES (
 "projection.enabled" = "true",
 "projection.datehour.type" = "date",
 "projection.datehour.format" = "yyyy/MM/dd/HH",
 "projection.datehour.range" = "2021/01/01/00,NOW",
 "projection.datehour.interval" = "1",
 "projection.datehour.interval.unit" = "HOURS",
 "storage.location.template" = "s3://DOC-EXAMPLE-BUCKET/prefix/${datehour}/"
)
```

La clausola `TBLPROPERTIES` nell'istruzione `CREATE TABLE` indica ad Athena quanto segue:

- Usa la proiezione delle partizioni durante la query sulla tabella
- La chiave di partizione `datehour` è di tipo `date` (che include un orario facoltativo)
- Come vengono formattate le date
- L'intervallo di date e orari Tieni presente che i valori devono essere separati da virgole, non da trattini.
- Dove trovare i dati su Amazon S3.

Quando si esegue una query sulla tabella, Athena calcola i valori per `datehour` e utilizza il modello di posizione di archiviazione per generare un elenco di posizioni di partizione.

Utilizzo del tipo **`date`**

Quando si utilizza il tipo `date` per una chiave di partizione proiettata, è necessario specificare un intervallo. Poiché non disponi di dati relativi alle date precedenti alla creazione del flusso di distribuzione di Firehose, puoi utilizzare la data di creazione come inizio. E poiché non si dispone di dati per le date in futuro, è possibile utilizzare il token speciale `NOW` come la fine.

Nell'esempio `CREATE TABLE`, la data di inizio è specificata come 1 gennaio 2021 a mezzanotte UTC.

> **Note**
>
> Configurare un intervallo che corrisponda il più possibile ai propri dati in modo che Athena cerchi solo le partizioni esistenti.

Quando viene eseguita una query sulla tabella di esempio, Athena utilizza le condizioni sulla chiave di partizione `datehour` insieme all'intervallo per generare i valori. Considera la query seguente:

```
SELECT *
FROM my_ingested_data
WHERE datehour >= '2020/12/15/00'
AND datehour < '2021/02/03/15'
```

La prima condizione nella query `SELECT` utilizza una data precedente all'inizio dell'intervallo di date specificato dall'istruzione `CREATE TABLE`. Poiché la configurazione di proiezione delle partizioni non

specifica partizioni per le date precedenti al 1° gennaio 2021, Athena cerca i dati solo nelle posizioni seguenti e ignora le date precedenti nella query.

```
s3://DOC-EXAMPLE-BUCKET/prefix/2021/01/01/00/
s3://DOC-EXAMPLE-BUCKET/prefix/2021/01/01/01/
s3://DOC-EXAMPLE-BUCKET/prefix/2021/01/01/02/
...
s3://DOC-EXAMPLE-BUCKET/prefix/2021/02/03/12/
s3://DOC-EXAMPLE-BUCKET/prefix/2021/02/03/13/
s3://DOC-EXAMPLE-BUCKET/prefix/2021/02/03/14/
```

Analogamente, se la query viene eseguita in una data e ora precedenti al 3 febbraio 2021 alle 15:00, l'ultima partizione rifletterebbe la data e l'ora correnti, non la data e l'ora nella condizione della query.

Se si desidera eseguire una query per i dati più recenti, è possibile sfruttare il fatto che Athena non genera date future e specificare solo un inizio `datehour`, come nel seguente esempio.

```
SELECT *
FROM my_ingested_data
WHERE datehour >= '2021/11/09/00'
```

Scelta delle chiavi di partizione

È possibile specificare in che modo la proiezione delle partizioni mappa le posizioni delle partizioni alle chiavi di partizione. Nell'esempio `CREATE TABLE` nella sezione precedente, la data e l'ora sono state combinate in una chiave di partizione chiamata datehour, ma sono possibili altri schemi. Ad esempio, è possibile configurare una tabella con chiavi di partizione separate per l'anno, il mese, il giorno e l'ora.

Tuttavia, la suddivisione delle date in anno, mese e giorno significa che il tipo di proiezione della partizione `date` non può essere utilizzato. Un'alternativa consiste nel separare la data dall'ora per sfruttare comunque il tipo di proiezione della partizione `date`, ma rendere più facili da leggere le query che specificano gli intervalli di ore.

Ricordando questo, nell'esempio `CREATE TABLE` seguente la data viene separata dall'ora. Poiché `date` è una parola riservata in SQL, nell'esempio viene utilizzata `day` come nome la chiave di partizione che rappresenta la data.

```
CREATE EXTERNAL TABLE my_ingested_data2 (
 ...
```

```
)
...
PARTITIONED BY (
 day STRING,
 hour INT
)
LOCATION "s3://DOC-EXAMPLE-BUCKET/prefix/"
TBLPROPERTIES (
 "projection.enabled" = "true",
 "projection.day.type" = "date",
 "projection.day.format" = "yyyy/MM/dd",
 "projection.day.range" = "2021/01/01,NOW",
 "projection.day.interval" = "1",
 "projection.day.interval.unit" = "DAYS",
 "projection.hour.type" = "integer",
 "projection.hour.range" = "0,23",
 "projection.hour.digits" = "2",
 "storage.location.template" = "s3://DOC-EXAMPLE-BUCKET/prefix/${day}/${hour}/"
)
```

Nell'istruzione `CREATE TABLE` di esempio, l'ora è una chiave di partizione separata configurata come numero intero. La configurazione per la chiave di partizione dell'ora specifica l'intervallo da 0 a 23 e che l'ora deve essere formattata con due cifre quando Athena genera le posizioni della partizione.

Una query per la tabella `my_ingested_data2` potrebbe essere simile alla seguente:

```
SELECT *
FROM my_ingested_data2
WHERE day = '2021/11/09'
AND hour > 3
```

Tipi di chiavi di partizione e tipi di proiezione delle partizioni

Tieni presente che la chiave `datehour` nel primo esempio `CREATE TABLE` è configurata come `date` nella configurazione di proiezione della partizione, ma il tipo di chiave di partizione è `string`. Lo stesso vale per `day` nel secondo esempio. I tipi nella configurazione di proiezione delle partizioni indicano ad Athena soltanto come formattare i valori quando vengono generate le posizioni della partizione. I tipi specificati non modificano il tipo di chiave di partizione, nelle query, `datehour` e `day` sono di tipo `string`.

Quando una query include una condizione come `day = '2021/11/09'`, Athena analizza la stringa sul lato destro dell'espressione utilizzando il formato data specificato nella configurazione

di proiezione della partizione. Una volta che Athena ha verificato che la data rientra nell'intervallo configurato, utilizza nuovamente il formato della data per inserire la data come stringa nel modello di posizione di archiviazione.

Allo stesso modo, per una condizione di query come `day > '2021/11/09'`, Athena analizza il lato destro e genera un elenco di tutte le date corrispondenti all'interno dell'intervallo configurato. Viene quindi utilizzato il formato della data per inserire ogni data nel modello di posizione di archiviazione per creare l'elenco delle posizioni delle partizioni.

Scrivi la stessa condizione come `day > '2021-11-09'` altrimenti `day > DATE '2021-11-09'` non funzionerà. Nel primo caso, il formato della data non corrisponde (notare i trattini anziché le barre) mentre nel secondo caso i tipi di dati non corrispondono.

Utilizzo di prefissi personalizzati e partizionamento dinamico

Firehose può essere configurato con prefissi personalizzati e partizionamento dinamico. Grazie a queste funzionalità, è possibile configurare le chiavi Amazon S3 e impostare schemi di partizionamento che supportano meglio il proprio caso d'uso. È inoltre possibile utilizzare la proiezione delle partizioni con questi schemi di partizionamento e configurarli di conseguenza.

Ad esempio, è possibile utilizzare la funzione di prefisso personalizzato per ottenere chiavi Amazon S3 con date formattate in ISO anziché lo schema `yyyy/MM/dd/HH` di default.

È inoltre possibile combinare prefissi personalizzati con il partizionamento dinamico per estrarre una proprietà come dai messaggi `customer_id` Firehose, come nell'esempio seguente.

```
prefix/!{timestamp:yyyy}-!{timestamp:MM}-!{timestamp:dd}/!
{partitionKeyFromQuery:customer_id}/
```

Con quel prefisso Amazon S3, il flusso di distribuzione di Firehose scriverebbe oggetti su chiavi come. `s3://DOC-EXAMPLE-BUCKET/prefix/2021-11-01/customer-1234/file.extension` Per una proprietà come `customer_id` dove i valori potrebbero non essere noti in anticipo, è possibile utilizzare il tipo di proiezione della partizione `injected` e usare un'istruzione `CREATE TABLE` come la seguente:

```
CREATE EXTERNAL TABLE my_ingested_data3 (
 ...
)
...
```

```
PARTITIONED BY (
 day STRING,
 customer_id STRING
)
LOCATION "s3://DOC-EXAMPLE-BUCKET/prefix/"
TBLPROPERTIES (
 "projection.enabled" = "true",
 "projection.day.type" = "date",
 "projection.day.format" = "yyyy-MM-dd",
 "projection.day.range" = "2021-01-01,NOW",
 "projection.day.interval" = "1",
 "projection.day.interval.unit" = "DAYS",
 "projection.customer_id.type" = "injected",
 "storage.location.template" = "s3://DOC-EXAMPLE-BUCKET/prefix/${day}/${customer_id}/"
)
```

Quando si esegue una query su una tabella con una chiave di partizione di tipo `injected`, la query deve includere un valore per quella chiave di partizione. Una query per la tabella `my_ingested_data3` potrebbe essere simile alla seguente:

```
SELECT *
FROM my_ingested_data3
WHERE day BETWEEN '2021-11-01' AND '2021-11-30'
AND customer_id = 'customer-1234'
```

Date formattate in ISO

Perché i valori per la chiave di partizione `day` sono formattati in ISO, è anche possibile utilizzare il tipo `DATE` per la chiave di partizione del giorno invece di `STRING`, come nel seguente esempio:

```
PARTITIONED BY (day DATE, customer_id STRING)
```

Quando si esegue una query, questa strategia consente di utilizzare le funzioni data sulla chiave di partizione senza analizzare o eseguire il casting, come nell'esempio seguente:

```
SELECT *
FROM my_ingested_data3
WHERE day > CURRENT_DATE - INTERVAL '7' DAY
AND customer_id = 'customer-1234'
```

> **Note**
>
> La specificazione di una chiave di partizione di tipo `DATE` presuppone che per creare chiavi Amazon S3 con date in formato ISO sia stata utilizzata la funzionalità di prefisso personalizzato. Se si utilizza il formato Firehose predefinito di`yyyy/MM/dd/HH`, è necessario specificare la chiave di partizione come type `string` anche se la proprietà della tabella corrispondente è di tipo`date`, come nell'esempio seguente:
>
> ```
> PARTITIONED BY (
>   `mydate` string)
> TBLPROPERTIES (
>   'projection.enabled'='true',
>    ...
>   'projection.mydate.type'='date',
>   'storage.location.template'='s3://DOC-EXAMPLE-BUCKET/prefix/${mydate}')
> ```

Creazione di una tabella dai risultati delle query (CTAS)

Una query `CREATE TABLE AS SELECT` (CTAS) crea una nuova tabella in Athena dai risultati di un'istruzione `SELECT` da un'altra query. Athena di dati creati dall'istruzione CTAS in una percorso specifico in Amazon S3. Per la sintassi, consulta CREATE TABLE AS.

`CREATE TABLE AS` combina una dichiarazione DDL `CREATE TABLE` con una dichiarazione DML `SELECT`, quindi tecnicamente contiene sia DDL sia DML. Tuttavia, tieni presente che ai fini delle Service Quotas, le query CTAS in Athena vengono trattate come DML. Per informazioni sulle Service Quotas in Athena, consulta Service Quotas (Quote di Servizio).

Utilizza le query CTAS per:

- Creare tabelle dai risultati della query in un'unica operazione, senza eseguire ripetutamente la query del set di dati non elaborati. In questo modo è più semplice lavorare con set di dati non elaborati.
- Trasforma i risultati delle query e migra le tabelle in altri formati di tabella come Apache Iceberg. Ciò consente di migliorare le prestazioni delle query e ridurre i costi in Athena. Per informazioni, consulta Creazione di tabelle Iceberg.

- Trasforma i risultati delle query in formati di archiviazione come Parquet e ORC. Ciò consente di migliorare le prestazioni delle query e ridurre i costi in Athena. Per informazioni, consulta Formati di archiviazione colonnare.
- Creazione di copie di tabelle esistenti che contengono solo i dati necessari.

Argomenti

Considerazioni e restrizioni per le query CTAS

Le sezioni seguenti descrivono le considerazioni e le limitazioni da tenere a mente quando si utilizzano le query `CREATE TABLE AS SELECT` (CTAS) in Athena.

Sintassi delle query CTAS

La sintassi delle query CTAS differisce dalla sintassi di `CREATE [EXTERNAL] TABLE` utilizzato per la creazione di tabelle. Per informazioni, consulta CREATE TABLE AS.

Query CTAS e visualizzazioni

Le query CTAS scrivono nuovi dati per un percorso specifico in Amazon S3, mentre le visualizzazioni non scrivono alcun dato.

Percorso delle query CTAS

Se il gruppo di lavoro sovrascrive l'impostazione lato client per la posizione dei risultati della query, Athena crea la tabella nella posizione `s3://DOC-EXAMPLE-BUCKET/tables/<query-id>/`. Per visualizzare la posizione dei risultati della query specificata per il gruppo di lavoro, visualizza i dettagli del gruppo di lavoro.

Se il gruppo di lavoro non sovrascrive la posizione dei risultati della query, puoi utilizzare la sintassi `WITH (external_location ='s3://DOC-EXAMPLE-BUCKET/')` nella query CTAS per specificare dove sono memorizzati i risultati della query CTAS.

> **Note**
>
> La proprietà `external_location` deve specificare una posizione vuota. Una query CTAS verifica che il percorso (prefisso) nel bucket sia vuoto e non sovrascrive mai i dati se il percorso dispone già di dati al suo interno. Per utilizzare nuovamente la stessa posizione, elimina i dati nella posizione del prefisso della chiave nel bucket.

Se ometti la sintassi `external_location` e non utilizzi l'impostazione del gruppo di lavoro, Athena usa l'impostazione lato client per la posizione dei risultati della query e crea la tabella nella posizione `s3://DOC-EXAMPLE-BUCKET/<Unsaved-or-query-name>/<year>/<month/<date>/tables/<query-id>/`.

Individuazione di file orfani

Se un'istruzione `CTAS` o `INSERT INTO` non riesce, è possibile che i dati orfani vengano lasciati nel percorso dati. Poiché Athena in alcuni casi non elimina alcun dato (anche parziale) dal bucket, potresti essere in grado di leggere questi dati parziali nelle query successive. Per individuare i file orfani per l'ispezione o l'eliminazione, è possibile utilizzare il file manifesto dati fornito da Athena per tenere traccia dell'elenco dei file da scrivere. Per ulteriori informazioni, consulta Identificazione dei file di output delle query e DataManifestLocation.

Clausola ORDER BY ignorata

In una query CTAS, Athena ignora le clausole `ORDER BY` nella parte `SELECT` della query.

Secondo le specifiche SQL (ISO 9075, parte 2), l'ordine delle righe di una tabella specificata da un'espressione di query è garantito solo per l'espressione di query che contiene immediatamente la clausola `ORDER BY`. Le tabelle in SQL sono in ogni caso intrinsecamente non ordinate e l'implementazione delle clausole `ORDER BY` in sottoquery comporterebbe un rendimento scadente della query e non produrrebbe un output ordinato. Pertanto, nelle query Athena CTAS, non vi è alcuna garanzia che l'ordine specificato dalla clausola `ORDER BY` venga preservato al momento della scrittura dei dati.

Formati per l'archiviazione dei risultati della query

I risultati della query CTAS sono archiviati in Parquet per impostazione predefinita, se non si specifica un formato di storage dei dati. È possibile archiviare i risultati CTAS in `PARQUET`, `ORC`, `AVRO`, `JSON` e `TEXTFILE`. I delimitatori multi-carattere non sono supportati per il formato CTAS `TEXTFILE`. Le interrogazioni CTAS non richiedono la specifica di SerDe a per interpretare le trasformazioni di formato. Per informazioni, consulta Example: Writing query results to a different format.

Formati di compressione

La compressione `GZIP` viene utilizzata per i risultati delle query CTAS nei formati JSON e TEXTFILE. Per Parquet, puoi usare `GZIP` o `SNAPPY` e il valore predefinito è `GZIP`. Per Parquet, puoi usare `LZ4`, `SNAPPY`, `ZLIB` o `ZSTD` e il valore predefinito è `ZLIB`. Per gli esempi CTAS che specificano la compressione, consulta Example: Specifying data storage and compression formats. Per ulteriori informazioni sulla compressione in Athena, consulta Supporto della compressione in Athena.

Limiti di partizione e di bucket

È possibile partizionare ed eseguire il bucket dei dati dei risultati di una query CTAS. Per ulteriori informazioni, consulta Partizionamento e arrotolamento in Athena. Quando si crea una tabella partizionata utilizzando CTAS, Athena ha un limite di scrittura di 100 partizioni.

Includi i predicati di partizionamento e di bucketing alla fine della clausola `WITH` che specifica le proprietà della tabella di destinazione. Per ulteriori informazioni, consulta Example: Creating bucketed and partitioned tables e Partizionamento e arrotolamento in Athena.

Per informazioni su una soluzione alternativa per la limitazione di 100 partizioni, consulta Utilizzo di CTAS e INSERT INTO per aggirare il limite di 100 partizioni.

Crittografia

È possibile crittografare i risultati delle query CTAS in Amazon S3, analogamente al modo di crittografare gli altri risultati della query in Athena. Per ulteriori informazioni, consulta Crittografia dei risultati di query Athena archiviati in Amazon S3.

Proprietario previsto del bucket

Per le istruzioni CTAS, l'impostazione prevista per il proprietario del bucket non si applica alla posizione della tabella di destinazione in Amazon S3. L'impostazione prevista per il proprietario

del bucket si applica solo al percorso di output di Amazon S3 specificato per i risultati delle query di Athena. Per ulteriori informazioni, consulta Specificare una posizione dei risultati delle query utilizzando la console Athena.

Tipi di dati

I tipi di dati di colonna per una query CTAS sono gli stessi specificati per la query originale.

Esecuzione delle query CTAS nella console

Nella console Athena è possibile creare una query CTAS da un'altra query.

Per creare una query CTAS da un'altra query

1. Eseguire la query nell'editor di query della console Athena.
2. Nella parte inferiore dell'editor di query, scegli l'opzione Create (Crea), quindi scegli Table from query (Tabella dalla query).
3. Nel modulo Create table as select (Crea tabella come da selezione), completa i campi come segue:

   a. Per Table name (Nome tabella), immetti il nome per la nuova tabella. Utilizza solo i caratteri minuscoli e i trattini bassi, ad esempio `my_select_query_parquet`.

   b. Per Database configuration (Configurazione database), utilizza le opzioni per scegliere un database esistente o creane uno nuovo.

   c. (Facoltativo) In Result configuration (Configurazione risultati), per Location of CTAS query results (Posizione dei risultati della query CTAS), se l'impostazione relativa alla posizione dei risultati della query per il gruppo di lavoro non sostituisce questa opzione, effettua una delle seguenti operazioni:

      - Inserisci il percorso di una posizione S3 esistente nella casella di ricerca o scegli Browse S3 (Sfoglia S3) per selezionare una posizione da un elenco.
      - Scegli View (Visualizza) per aprire la pagina Buckets (Bucket) della console Amazon S3, dove puoi visualizzare ulteriori informazioni sui bucket esistenti oltre a scegliere un bucket o crearne uno nuovo con impostazioni personalizzate.

   Specifica una posizione vuota in Amazon S3 in cui i dati verranno emessi. Se i dati esistono già nella posizione specificata, la query ha esito negativo con un errore.

Se l'impostazione relativa alla posizione dei risultati della query per il gruppo di lavoro sostituisce questa impostazione, Athena crea la tabella nella posizione `s3://DOC-EXAMPLE-BUCKET/tables/query_id/`

d. Per Data format (Formato dei dati), specifica il formato in cui si trovano i dati.

- Table type (Tipo di tabella): il tipo di tabella predefinito in Athena è Apache Hive.
- File format (Formato file): scegli tra opzioni come CSV, TSV, JSON, Parquet o ORC. Per informazioni sui formati Parquet e ORC, consultare Formati di archiviazione colonnare.
- Write compression (Compressione per la scrittura): (facoltativo) scegli un formato di compressione. Athena supporta diversi formati di compressione per la lettura e la scrittura di dati, inclusa la lettura da una tabella che utilizza più formati di compressione. Ad esempio, Athena può leggere correttamente i dati in una tabella che utilizza il formato file Parquet quando alcuni file Parquet vengono compressi con Snappy e altri file Parquet vengono compressi con GZIP. Lo stesso principio vale per i formati di archiviazione ORC, file di testo e JSON. Per ulteriori informazioni, consulta Supporto della compressione in Athena.
- Partitions (Partizioni): (facoltativo) seleziona le colonne da partizionare. Effettuando il partizionamento dei dati, è possibile limitare la quantità di dati scansionati da ogni query, migliorando così le prestazioni e riducendo i costi. in base a qualsiasi chiave di partizione. Per ulteriori informazioni, consulta Partizionamento dei dati in Athena.
- Buckets (Bucket): (facoltativo) seleziona le colonne che desideri inserire nel bucket. Il bucketing è una tecnica di raggruppamento dei dati in base a colonne specifiche all'interno di un'unica partizione. Queste colonne sono note come chiavi bucket. Raggruppando i dati correlati in un unico bucket (un file all'interno di una partizione), si riduce notevolmente la quantità di dati scansionati da Athena, migliorando così le prestazioni delle query e riducendo i costi. Per ulteriori informazioni, consulta Partizionamento e arrotolamento in Athena.

e. Per Preview table query (Query di anteprima della tabella), esamina la query. Per la sintassi della query, consulta CREATE TABLE AS.

f. Scegliere Create table (Crea tabella).

Per creare una query CTAS utilizzando un modello SQL

Utilizzare il modello `CREATE TABLE AS SELECT` per creare una query CTAS nell'editor di query.

1. Nella console Athena, accanto a Tables and views (Tabelle e visualizzazioni), scegli Create table (Crea tabella) e quindi scegli CREATE TABLE AS SELECT (CREA TABELLA COME SELEZIONE). Nell'editor di query viene inserita una query CTAS con valori di segnaposto.
2. Nell'Editor di query, modificare la query se necessario. Per la sintassi della query, consulta CREATE TABLE AS.
3. Seleziona Esegui.

Per alcuni esempi, consulta Esempi di query CTAS.

Partizionamento e arrotolamento in Athena

Il partizionamento e il raggruppamento in bucket sono due modi per ridurre la quantità di dati che Athena deve scansionare quando si esegue una query. Il partizionamento e il raggruppamento in bucket sono complementari e possono essere utilizzati insieme. La riduzione della quantità di dati scansionati comporta un miglioramento delle prestazioni e una riduzione dei costi. Per le linee guida generali sulle prestazioni delle query Athena, consulta Suggerimenti di ottimizzazione delle prestazioni in Amazon Athena.

Cos'è il partizionamento?

Il partizionamento significa organizzare i dati in directory (o «prefissi») su Amazon S3 in base a una particolare proprietà dei dati. Tali proprietà sono chiamate chiavi di partizione. Una chiave di partizione comune è la data o un'altra unità di tempo come l'anno o il mese. Tuttavia, un set di dati può essere partizionato in base a più di una chiave. Ad esempio, i dati sulle vendite di prodotti potrebbero essere suddivisi per data, categoria di prodotto e mercato.

Decidere come partizionare

Elementi adatti a essere utilizzati come chiavi di partizione sono le proprietà che vengono utilizzate sempre o frequentemente nelle query e hanno una cardinalità bassa. Esiste un compromesso tra l'avere troppe partizioni e l'averne troppo poche. Con troppe partizioni, l'aumento del numero di file crea un sovraccarico. Inoltre, il filtraggio delle partizioni stesse comporta un certo sovraccarico. Con un numero insufficiente di partizioni, le query spesso devono scansionare più dati.

Creazione di una tabella partizionata

Quando un set di dati viene partizionato, puoi creare una tabella partizionata in Athena. Una tabella partizionata è una tabella con chiavi di partizione. Quando utilizzi `CREATE TABLE`, aggiungi delle

partizioni alla tabella. Quando utilizzi `CREATE TABLE AS`, le partizioni create su Amazon S3 vengono aggiunte automaticamente alla tabella.

In una dichiarazione `CREATE TABLE`, specifica le chiavi di partizione nella clausola `PARTITIONED BY (column_name data_type)`. In un'istruzione `CREATE TABLE AS`, si specificano le chiavi di partizione in una clausola `WITH (partitioned_by = ARRAY['partition_key'])` o `WITH (partitioning = ARRAY['partition_key'])` per le tabelle Iceberg. Per motivi di prestazioni, le chiavi di partizione devono essere sempre di tipo `STRING`. Per ulteriori informazioni, consulta Usa string come tipo di dati per le chiavi di partizione.

Per ulteriori dettagli sulla sintassi di `CREATE TABLE` e `CREATE TABLE AS`, consulta CREATE TABLE e Proprietà tabella CTAS.

Esecuzione di query su tabelle partizionate

Quando esegui una query su una tabella partizionata, Athena utilizza i predicati della query per filtrare l'elenco delle partizioni. Quindi utilizza le posizioni delle partizioni corrispondenti per elaborare i file trovati. Athena può ridurre in modo efficiente la quantità di dati analizzati semplicemente non leggendo i dati nelle partizioni che non corrispondono ai predicati delle query.

Esempi

Supponi di avere una tabella partizionata per `sales_date` e `product_category` e di voler conoscere le entrate totali nell'arco di una settimana in una categoria specifica. Includi i predicati nelle colonne `product_category` e `sales_date` per assicurarti che Athena analizzi solo la quantità minima di dati, come nell'esempio seguente.

```
SELECT SUM(amount) AS total_revenue
FROM sales
WHERE sales_date BETWEEN '2023-02-27' AND '2023-03-05'
AND product_category = 'Toys'
```

Supponi di avere un set di dati partizionato per data ma che abbia anche un timestamp dettagliato.

Con le tabelle Iceberg, puoi dichiarare che una chiave di partizione ha una relazione con una colonna, ma con le tabelle Hive il motore di query non conosce le relazioni tra colonne e chiavi di partizione. Per questo motivo, è necessario includere un predicato sia sulla colonna che sulla chiave di partizione nella query per assicurarsi che la query non analizzi più dati del necessario.

Ad esempio, supponi che la tabella `sales` dell'esempio precedente contenga anche una colonna `sold_at` del tipo di dati `TIMESTAMP`. Se desideri le entrate solo per un intervallo di tempo specifico, devi scrivere la query in questo modo:

```
SELECT SUM(amount) AS total_revenue
FROM sales
WHERE sales_date = '2023-02-28'
AND sold_at BETWEEN TIMESTAMP '2023-02-28 10:00:00' AND TIMESTAMP '2023-02-28
 12:00:00'
AND product_category = 'Toys'
```

Per ulteriori informazioni su questa differenza tra le query delle tabelle Hive e Iceberg, consulta Come scrivere query per campi timestamp che sono anche partizionati in base all'ora.

Cos'è il raggruppamento in bucket?

Il raggruppamento in bucket è un modo per organizzare i record di un set di dati in categorie chiamate bucket.

Questo significato di bucket and raggruppamento in bucket è diverso e non deve essere confuso con i bucket Amazon S3. Nel bucket di dati, i record che hanno lo stesso valore per una proprietà vengono inseriti nello stesso bucket. I record vengono distribuiti nel modo più uniforme possibile tra i bucket in modo che ogni bucket contenga all'incirca la stessa quantità di dati.

In pratica, i bucket sono file e una funzione hash determina il bucket in cui va inserito un record. Un set di dati con bucket avrà uno o più file per bucket per partizione. Il bucket a cui appartiene un file è codificato nel nome del file.

Vantaggi del raggruppamento in bucket

Il raggruppamento in bucket è utile quando un set di dati è inserito in un bucket in base a una determinata proprietà e si desidera recuperare i record in cui tale proprietà ha un determinato valore. Poiché i dati sono raggruppati in un bucket, Athena può utilizzare il valore per determinare quali file guardare. Ad esempio, supponi che un set di dati sia suddiviso in bucket in base a `customer_id` e che tu voglia trovare tutti i record di un cliente specifico. Athena determina il bucket che contiene tali record e legge solo i file in quel bucket.

I buoni candidati per il raggruppamento in bucket si presentano quando si hanno colonne con cardinalità elevata (ovvero molti valori distinti), sono distribuite uniformemente e vengono spesso eseguite interrogazioni per valori specifici.

> **Note**
>
> Athena non supporta l'utilizzo di `INSERT INTO` per aggiungere nuovi record alle tabelle bloccate.

Tipi di dati supportati per il filtro su colonne con bucket

Puoi aggiungere filtri su colonne separate con determinati tipi di dati. Athena supporta il filtraggio solo su colonne raggruppate per bucket con i seguenti tipi di dati:

- BOOLEAN
- BYTE
- DATE
- DOUBLE
- FLOAT
- INT
- LONG
- SHORT
- STRING
- VARCHAR

Supporto per Hive e Spark

La versione 2 del motore Athena supporta i set di dati raggruppati utilizzando l'algoritmo bucket Hive e la versione 3 del motore Athena supporta anche l'algoritmo di raggruppamento in bucket Apache Spark. L'impostazione predefinita è il raggruppamento in bucket Hive. Se il tuo set di dati è raggruppato in bucket tramite l'algoritmo Spark, utilizza la clausola `TBLPROPERTIES` per impostare il valore della proprietà `bucketing_format` su `spark`.

> **Note**
>
> Athena ha un limite di 100 partizioni per query `CREATE TABLE AS SELECT` (CTAS). Allo stesso modo, puoi aggiungere un massimo di 100 partizioni a una tabella di destinazione con una dichiarazione INSERT INTO. Questo limite di 100 si applica solo quando la tabella è sia suddivisa in bucket che in partizioni.

Se superi questa limitazione potresti ricevere il messaggio di errore HIVE_TOO_MANY_OPEN_PARTITIONS: Exceeded limit of 100 open writers for partitions/buckets (HIVE_TOO_MANY_OPEN_PARTITIONS: limite di 100 scrittori aperti per partizioni/bucket superato). Per ovviare questa limitazione, puoi utilizzare un'istruzione CTAS e una serie di istruzioni `INSERT INTO` che creano o inseriscono fino a 100 partizioni ciascuna. Per ulteriori informazioni, consulta Utilizzo di CTAS e INSERT INTO per aggirare il limite di 100 partizioni.

Esempio di raggruppamento in bucket di CREATE TABLE

Per creare una tabella per un set di dati raggruppato in bucket esistenti, utilizza la clausola `CLUSTERED BY (`*`column`*`)` seguita dalla clausola `INTO` *`N`* `BUCKETS`. La clausola `INTO` *`N`* `BUCKETS` specifica il numero di bucket in cui sono inseriti i dati.

Nell'esempio `CREATE TABLE` seguente, il set di dati `sales` viene suddiviso in 8 bucket in base a `customer_id` attraverso l'algoritmo Spark. L'istruzione `CREATE TABLE` utilizza le clausole `CLUSTERED BY` e `TBLPROPERTIES` per impostare le proprietà di conseguenza.

```
CREATE EXTERNAL TABLE sales (...)
...
CLUSTERED BY (`customer_id`) INTO 8 BUCKETS
...
TBLPROPERTIES (
  'bucketing_format' = 'spark'
)
```

Esempio di raggruppamento in bucket di CREATE TABLE AS (CTAS)

Per specificare il raggruppamento in bucket with`CREATE TABLE AS`, utilizzate i `bucket_count` parametri `bucketed_by` and, come nell'esempio seguente.

```
CREATE TABLE sales
WITH (
  ...
  bucketed_by = ARRAY['customer_id'],
  bucket_count = 8
)
AS SELECT ...
```

Esempio di query bucketing

La seguente query di esempio cerca i nomi dei prodotti che un cliente specifico ha acquistato nel corso di una settimana.

```
SELECT DISTINCT product_name
FROM sales
WHERE sales_date BETWEEN '2023-02-27' AND '2023-03-05'
AND customer_id = 'c123'
```

Se questa tabella è partizionata in base a `sales_date` e raggruppata in bucket in base a `customer_id`, Athena può calcolare il bucket in cui si trovano i record dei clienti. Al massimo, Athena legge un file per partizione.

Risorse aggiuntive

- Per un esempio di `CREATE TABLE AS` che crea tabelle sia partizionate sia raggruppate nei bucket, consulta Esempio: creazione di tabelle raggruppate in bucket e partizionate.
- Per informazioni sull'implementazione del bucketing sui AWS data lake, incluso l'utilizzo di un'istruzione Athena CTAS AWS Glue , per Apache Spark e del bucketing per le tabelle Apache Iceberg, consulta AWS il post sul blog Big Data Optimize data layout by bucketing with Amazon Athena and to accelerate downstream. AWS Glue

Esempi di query CTAS

Utilizzare i seguenti esempi per creare query CTAS. Per ulteriori informazioni sulla sintassi CTAS, consulta CREATE TABLE AS.

In questa sezione:

- Example: Creating partitioned tables
- Example: Creating bucketed and partitioned tables
- Example: Creating an Iceberg table with Parquet data
- Example: Creating an Iceberg table with Avro data

Example Esempio: duplicazione di una tabella selezionando tutte le colonne

L'esempio seguente crea una tabella copiando tutte le colonne di una tabella:

```
CREATE TABLE new_table AS
SELECT *
FROM old_table;
```

Nella seguente variazione dello stesso esempio, l'istruzione `SELECT` comprende anche una clausola `WHERE`. In questo caso, la query seleziona solo le righe dalla tabella che soddisfano la clausola `WHERE`:

```
CREATE TABLE new_table AS
SELECT *
FROM old_table
WHERE condition;
```

Example Esempio: selezione di colonne specifiche da una o più tabelle

L'esempio seguente crea una nuova query che viene eseguita su un set di colonne da un'altra tabella:

```
CREATE TABLE new_table AS
SELECT column_1, column_2, ... column_n
FROM old_table;
```

Questa variazione dello stesso esempio crea una nuova tabella in base a colonne specifiche di più tabelle:

```
CREATE TABLE new_table AS
SELECT column_1, column_2, ... column_n
FROM old_table_1, old_table_2, ... old_table_n;
```

Example Esempio: creazione di una copia vuota di una tabella esistente

L'esempio seguente utilizza `WITH NO DATA` per creare una nuova tabella vuota e ha lo stesso schema della tabella originale:

```
CREATE TABLE new_table
AS SELECT *
FROM old_table
WITH NO DATA;
```

Example Esempio: specifica dei formati di compressione e dell'archiviazione dei dati

Con CTAS, è possibile utilizzare una tabella di origine in un formato di archiviazione per creare un'altra tabella in un formato di archiviazione diverso.

Utilizza la proprietà `format` per specificare `ORC`, `PARQUET`, `AVRO`, `JSON` oppure `TEXTFILE` come formato di archiviazione per la nuova tabella.

Per i formati di archiviazione `PARQUET`, `ORC`, `TEXTFILE` e `JSON`, utilizzare la proprietà `write_compression` per specificare il formato di compressione per i dati della nuova tabella. Per informazioni sui formati di compressione supportati da ciascun formato di file, consultare Supporto della compressione in Athena.

L'esempio seguente specifica che i dati nella tabella `new_table` devono essere archiviati in formato Parquet e devono utilizzare la compressione Snappy. La compressione di default per Parquet è `GZIP`.

```
CREATE TABLE new_table
WITH (
      format = 'Parquet',
      write_compression = 'SNAPPY')
AS SELECT *
FROM old_table;
```

L'esempio seguente specifica che i dati nella tabella `new_table` devono essere archiviati in formato ORC tramite la compressione Snappy. La compressione di default per ORC è ZLIB.

```
CREATE TABLE new_table
WITH (format = 'ORC',
      write_compression = 'SNAPPY')
AS SELECT *
```

```
FROM old_table ;
```

L'esempio seguente specifica che i dati nella tabella `new_table` devono essere archiviati in formato file di testo tramite la compressione Snappy. La compressione di default sia per il file di testo che per i formati JSON è GZIP.

```
CREATE TABLE new_table
WITH (format = 'TEXTFILE',
      write_compression = 'SNAPPY')
AS SELECT *
FROM old_table ;
```

Example Esempio: scrittura dei risultati della query in un formato diverso

La query CTAS seguente seleziona tutti i record da `old_table`, che possono essere archiviati in CSV o in un altro formato, e crea una nuova tabella con i dati sottostanti salvati in Amazon S3 in formato ORC:

```
CREATE TABLE my_orc_ctas_table
WITH (
      external_location = 's3://DOC-EXAMPLE-BUCKET/my_orc_stas_table/',
      format = 'ORC')
AS SELECT *
FROM old_table;
```

Example Esempio: creazione di tabelle non partizionate

I seguenti esempi creano tabelle non partizionate. I dati della tabella vengono memorizzati in diversi formati. Alcuni di questi esempi specificano il percorso esterno.

L'esempio seguente crea una query CTAS che archivia i risultati in un file di testo:

```
CREATE TABLE ctas_csv_unpartitioned
WITH (
     format = 'TEXTFILE',
     external_location = 's3://DOC-EXAMPLE-BUCKET/ctas_csv_unpartitioned/')
AS SELECT key1, name1, address1, comment1
FROM table1;
```

Nel seguente esempio, i risultati vengono archiviati in Parquet e viene utilizzato il percorso dei risultati predefiniti.

```
CREATE TABLE ctas_parquet_unpartitioned
WITH (format = 'PARQUET')
AS SELECT key1, name1, comment1
FROM table1;
```

Nella seguente query, la tabella viene archiviata in formato JSON e vengono selezionate colonne specifiche dai risultati della tabella originale:

```
CREATE TABLE ctas_json_unpartitioned
WITH (
     format = 'JSON',
     external_location = 's3://DOC-EXAMPLE-BUCKET/ctas_json_unpartitioned/')
AS SELECT key1, name1, address1, comment1
FROM table1;
```

Nell'esempio seguente, il formato è ORC:

```
CREATE TABLE ctas_orc_unpartitioned
WITH (
     format = 'ORC')
AS SELECT key1, name1, comment1
FROM table1;
```

Nell'esempio seguente, il formato è Avro:

```
CREATE TABLE ctas_avro_unpartitioned
WITH (
     format = 'AVRO',
     external_location = 's3://DOC-EXAMPLE-BUCKET/ctas_avro_unpartitioned/')
AS SELECT key1, name1, comment1
FROM table1;
```

Example Esempio: creazione di tabelle partizionate

I seguenti esempi mostrano le query `CREATE TABLE AS SELECT` per tabelle partizionate in diversi formati di storage, che utilizzano `partitioned_by` e altre proprietà nella clausola `WITH`. Per la sintassi, consulta Proprietà tabella CTAS. Per ulteriori informazioni sulla scelta delle colonne per il partizionamento, consulta Partizionamento e arrotolamento in Athena.

> **Note**
>
> Elencare le colonne di partizione alla fine dell'elenco di colonne nell'istruzione SELECT. È possibile partizionare per più di una colonna e ottenere fino a 100 combinazioni univoche di partizioni e bucket. Ad esempio, è possibile avere 100 partizioni se non vengono specificati bucket.

```
CREATE TABLE ctas_csv_partitioned
WITH (
     format = 'TEXTFILE',
     external_location = 's3://DOC-EXAMPLE-BUCKET/ctas_csv_partitioned/',
     partitioned_by = ARRAY['key1'])
AS SELECT name1, address1, comment1, key1
FROM tables1;
```

```
CREATE TABLE ctas_json_partitioned
WITH (
     format = 'JSON',
     external_location = 's3://DOC-EXAMPLE-BUCKET/ctas_json_partitioned/',
     partitioned_by = ARRAY['key1'])
AS select name1, address1, comment1, key1
FROM table1;
```

Example Esempio: creazione di tabelle con bucket e partizionate

L'esempio seguente mostra una query CREATE TABLE AS SELECT che utilizza sia il partizionamento sia il bucketing per archiviare i risultati della query in Amazon S3. I risultati vengono partizionati e viene eseguito il bucketing in base a diverse colonne. Athena supporta un massimo di 100 combinazioni uniche di bucket e partizioni. Ad esempio, se si crea una tabella con cinque bucket, sono supportate 20 partizioni con cinque bucket ciascuno. Per la sintassi, consulta Proprietà tabella CTAS.

Per ulteriori informazioni sulla scelta delle colonne per il bucketing, consulta Partizionamento e arrotolamento in Athena.

```
CREATE TABLE ctas_avro_bucketed
WITH (
      format = 'AVRO',
      external_location = 's3://DOC-EXAMPLE-BUCKET/ctas_avro_bucketed/',
```

```
      partitioned_by = ARRAY['nationkey'],
      bucketed_by = ARRAY['mktsegment'],
      bucket_count = 3)
AS SELECT key1, name1, address1, phone1, acctbal, mktsegment, comment1, nationkey
FROM table1;
```

Example Esempio: creazione di una tabella Iceberg con dati Parquet

Nell'esempio seguente viene creata una tabella Iceberg con file di dati Parquet. I file vengono partizionati per mese nella colonna `dt` nella `table1`. Nell'esempio vengono aggiornate le proprietà di mantenimento della tabella in modo che per impostazione predefinita vengano conservati 10 snapshot su ogni ramo della tabella. Vengono mantenuti anche gli snapshot degli ultimi 7 giorni. Per ulteriori informazioni sulle proprietà delle tabelle Iceberg in Athena, consulta Proprietà tabella.

```
CREATE TABLE ctas_iceberg_parquet
WITH (table_type = 'ICEBERG',
      format = 'PARQUET',
      location = 's3://DOC-EXAMPLE-BUCKET/ctas_iceberg_parquet/',
      is_external = false,
      partitioning = ARRAY['month(dt)'],
      vacuum_min_snapshots_to_keep = 10,
      vacuum_max_snapshot_age_seconds = 604800
   )
AS SELECT key1, name1, dt FROM table1;
```

Example Esempio: creazione di una tabella Iceberg con dati Avro

Nell'esempio seguente viene creata una tabella Iceberg con file di dati Avro partizionati da `key1`.

```
CREATE TABLE ctas_iceberg_avro
WITH ( format = 'AVRO',
       location = 's3://DOC-EXAMPLE-BUCKET/ctas_iceberg_avro/',
       is_external = false,
       table_type = 'ICEBERG',
       partitioning = ARRAY['key1'])
AS SELECT key1, name1, date FROM table1;
```

Utilizzo di CTAS e INSERT INTO per ETL e analisi dei dati

Utilizza le istruzioni Create Table as Select (CTAS) e INSERT INTO in Athena per estrarre, trasformare e caricare (ETL) i dati in Amazon S3 per l'elaborazione dei dati. In questo argomento

viene illustrato come utilizzare queste istruzioni per partizionare e convertire un set di dati nel formato di dati in colonna per ottimizzarlo per l'analisi dei dati.

Le istruzioni CTAS utilizzano le query SELECT standard per creare le nuove tabelle. Puoi utilizzare un'istruzione CTAS per creare un sottoinsieme di dati per l'analisi. In un'istruzione CTAS, puoi partizionare i dati, specificare la compressione e convertire i dati in un formato a colonne come Apache Parquet o Apache ORC. Quando esegui la query CTAS, le tabelle e le partizioni che crea vengono automaticamente aggiunte al AWS Glue Data Catalog. In questo modo le nuove tabelle e partizioni create sono immediatamente disponibili per le query successive.

Le istruzioni INSERT INTO inseriscono nuove righe in una tabella di destinazione in base a un'istruzione di query SELECT che viene eseguita per una tabella di origine. Puoi utilizzare le istruzioni INSERT INTO per trasformare e caricare i dati della tabella di origine in formato CSV in dati della tabella di destinazione utilizzando tutte le trasformazioni supportate da CTAS.

Panoramica

In Athena, utilizza un'istruzione CTAS per eseguire una conversione batch iniziale dei dati. Quindi utilizza più istruzioni INSERT INTO per apportare aggiornamenti incrementali alla tabella creata dall'istruzione CTAS.

Fasi

- Fase 1: creare una tabella basata sul set di dati originale
- Fase 2: utilizzare CTAS per partizionare, convertire e comprimere i dati
- Fase 3: utilizzare INSERT INTO per aggiungere dati
- Fase 4: misurare le prestazioni e le differenze dei costi

Fase 1: creare una tabella basata sul set di dati originale

Nell'esempio riportato in questo argomento viene utilizzato un sottoinsieme Amazon S3 leggibile del set di dati NOAA Global Historical Climatology Network Daily (GHCN-D) disponibile al pubblico. I dati in Amazon S3 hanno le seguenti caratteristiche.

```
Location: s3://aws-bigdata-blog/artifacts/athena-ctas-insert-into-blog/
Total objects: 41727
Size of CSV dataset: 11.3 GB
Region: us-east-1
```

I dati originali vengono archiviati in Amazon S3 senza partizioni. I dati sono in formato CSV in file come il seguente.

```
2019-10-31 13:06:57  413.1 KiB artifacts/athena-ctas-insert-into-blog/2010.csv0000
2019-10-31 13:06:57  412.0 KiB artifacts/athena-ctas-insert-into-blog/2010.csv0001
2019-10-31 13:06:57   34.4 KiB artifacts/athena-ctas-insert-into-blog/2010.csv0002
2019-10-31 13:06:57  412.2 KiB artifacts/athena-ctas-insert-into-blog/2010.csv0100
2019-10-31 13:06:57  412.7 KiB artifacts/athena-ctas-insert-into-blog/2010.csv0101
```

Le dimensioni dei file in questo esempio sono relativamente piccole. Unendoli in file più grandi, è possibile ridurre il numero totale di file, consentendo prestazioni migliori delle query. Puoi utilizzare le istruzioni CTAS e INSERT INTO per migliorare le prestazioni delle query.

Per creare un database e una tabella in base al set di dati di esempio

1. Nella console Athena, scegli Stati Uniti orientali (Virginia settentrionale). Regione AWS Assicurarsi di eseguire tutte le query in questo tutorial in `us-east-1`.
2. Nell'editor di query Athena, eseguire il comando CREATE DATABASE per creare un database.

   ```
   CREATE DATABASE blogdb
   ```

3. Eseguire l'istruzione seguente per creare una tabella.

   ```
   CREATE EXTERNAL TABLE `blogdb`.`original_csv` (
     `id` string,
     `date` string,
     `element` string,
     `datavalue` bigint,
     `mflag` string,
     `qflag` string,
     `sflag` string,
     `obstime` bigint)
   ROW FORMAT DELIMITED
     FIELDS TERMINATED BY ','
   STORED AS INPUTFORMAT
     'org.apache.hadoop.mapred.TextInputFormat'
   OUTPUTFORMAT
     'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
   LOCATION
     's3://aws-bigdata-blog/artifacts/athena-ctas-insert-into-blog/'
   ```

Fase 2: utilizzare CTAS per partizionare, convertire e comprimere i dati

Dopo aver creato una tabella, puoi utilizzare una singola istruzione CTAS per convertire i dati in formato Parquet con compressione Snappy e per partizionare i dati per anno.

La tabella creata nella fase 1 ha un campo `date` con la data nel formato YYYYMMDD (ad esempio, `20100104`). Poiché la nuova tabella verrà partizionata per `year`, l'istruzione di esempio nella procedura seguente utilizza la funzione Presto `substr("date",1,4)` per estrarre il valore `year` dal campo `date`.

Per convertire i dati in formato Parquet con compressione Snappy e partizionamento per anno

- Eseguire la seguente istruzione CTAS, sostituendo *`your-bucket`* con la posizione del tuo bucket Amazon S3.

```
CREATE table new_parquet
WITH (format='PARQUET',
parquet_compression='SNAPPY',
partitioned_by=array['year'],
external_location = 's3://DOC-EXAMPLE-BUCKET/optimized-data/')
AS
SELECT id,
         date,
         element,
         datavalue,
         mflag,
         qflag,
         sflag,
         obstime,
         substr("date",1,4) AS year
FROM original_csv
WHERE cast(substr("date",1,4) AS bigint) >= 2015
        AND cast(substr("date",1,4) AS bigint) <= 2019
```

> **Note**
>
> In questo esempio, la tabella creata include solo i dati dal 2015 al 2019. Nella fase 3, aggiungi nuovi dati a questa tabella utilizzando il comando INSERT INTO.

Al termine della query, utilizza la procedura seguente per verificare l'output nella posizione Amazon S3 specificata nell'istruzione CTAS.

Per visualizzare le partizioni e i file parquet creati dall'istruzione CTAS

1. Per mostrare le partizioni create, esegui il seguente comando. AWS CLI Assicurarsi di includere la barra finale (/).

```
aws s3 ls s3://DOC-EXAMPLE-BUCKET/optimized-data/
```

   L'output mostra le partizioni.

```
      PRE year=2015/
      PRE year=2016/
      PRE year=2017/
      PRE year=2018/
      PRE year=2019/
```

2. Per visualizzare i file Parquet, eseguire il comando seguente. L'opzione | head -5 che limita l'output ai primi cinque risultati, non è disponibile in Windows.

```
aws s3 ls s3://DOC-EXAMPLE-BUCKET/optimized-data/ --recursive --human-readable |
 head -5
```

   L'output è simile a quello riportato di seguito.

```
2019-10-31 14:51:05    7.3 MiB optimized-data/
year=2015/20191031_215021_00001_3f42d_1be48df2-3154-438b-b61d-8fb23809679d
2019-10-31 14:51:05    7.0 MiB optimized-data/
year=2015/20191031_215021_00001_3f42d_2a57f4e2-ffa0-4be3-9c3f-28b16d86ed5a
2019-10-31 14:51:05    9.9 MiB optimized-data/
year=2015/20191031_215021_00001_3f42d_34381db1-00ca-4092-bd65-ab04e06dc799
2019-10-31 14:51:05    7.5 MiB optimized-data/
year=2015/20191031_215021_00001_3f42d_354a2bc1-345f-4996-9073-096cb863308d
2019-10-31 14:51:05    6.9 MiB optimized-data/
year=2015/20191031_215021_00001_3f42d_42da4cfd-6e21-40a1-8152-0b902da385a1
```

Fase 3: utilizzare INSERT INTO per aggiungere dati

Nella fase 2, hai utilizzato CTAS per creare una tabella con partizioni per gli anni dal 2015 al 2019. Tuttavia, il set di dati originale contiene anche dati relativi agli anni dal 2010 al 2014. Ora aggiungi questi dati utilizzando un'istruzione INSERT INTO.

Per aggiungere dati alla tabella utilizzando una o più istruzioni INSERT INTO

1. Eseguire il seguente comando INSERT INTO, specificando gli anni prima del 2015 nella clausola WHERE.

```
INSERT INTO new_parquet
SELECT id,
         date,
         element,
         datavalue,
         mflag,
         qflag,
         sflag,
         obstime,
         substr("date",1,4) AS year
FROM original_csv
WHERE cast(substr("date",1,4) AS bigint) < 2015
```

2. Eseguire nuovamente il comando `aws s3 ls`, utilizzando la seguente sintassi.

```
aws s3 ls s3://DOC-EXAMPLE-BUCKET/optimized-data/
```

L'output mostra le nuove partizioni.

```
      PRE year=2010/
      PRE year=2011/
      PRE year=2012/
      PRE year=2013/
      PRE year=2014/
      PRE year=2015/
      PRE year=2016/
      PRE year=2017/
      PRE year=2018/
      PRE year=2019/
```

3. Per vedere la riduzione delle dimensioni del set di dati ottenuta utilizzando la compressione e lo storage a colonne in formato Parquet, eseguire il comando seguente.

```
aws s3 ls s3://DOC-EXAMPLE-BUCKET/optimized-data/ --recursive --human-readable --
summarize
```

I seguenti risultati mostrano che la dimensione del set di dati dopo Parquet con compressione Snappy è 1,2 GB.

```
...
2020-01-22 18:12:02 2.8 MiB optimized-data/
year=2019/20200122_181132_00003_nja5r_f0182e6c-38f4-4245-afa2-9f5bfa8d6d8f
2020-01-22 18:11:59 3.7 MiB optimized-data/
year=2019/20200122_181132_00003_nja5r_fd9906b7-06cf-4055-a05b-f050e139946e
Total Objects: 300
     Total Size: 1.2 GiB
```

4. Se più dati CSV vengono aggiunti alla tabella originale, è possibile aggiungere i dati alla tabella Parquet utilizzando le istruzioni INSERT INTO. Ad esempio, se sono disponibili nuovi dati per l'anno 2020, è possibile eseguire la seguente istruzione INSERT INTO. L'istruzione aggiunge i dati e la partizione pertinente alla tabella `new_parquet`.

```
INSERT INTO new_parquet
SELECT id,
         date,
         element,
         datavalue,
         mflag,
         qflag,
         sflag,
         obstime,
         substr("date",1,4) AS year
FROM original_csv
WHERE cast(substr("date",1,4) AS bigint) = 2020
```

> **Note**
>
> L'istruzione INSERT INTO supporta la scrittura di un massimo di 100 partizioni nella tabella di destinazione. Tuttavia, per aggiungere più di 100 partizioni, è possibile

eseguire più istruzioni INSERT INTO. Per ulteriori informazioni, consulta Utilizzo di CTAS e INSERT INTO per aggirare il limite di 100 partizioni.

Fase 4: misurare le prestazioni e le differenze dei costi

Dopo aver trasformato i dati, puoi misurare i miglioramenti delle prestazioni e i risparmi sui costi eseguendo le stesse query nelle tabelle nuove e precedenti e confrontando i risultati.

> **Note**
>
> Per informazioni sui costi per query di Athena, consulta Prezzi di Amazon Athena.

Per misurare i miglioramenti delle performance e le differenze dei costi

1. Eseguire la seguente query nella tabella originale. La query trova il numero di ID distinti per ogni valore dell'anno.

```
SELECT substr("date",1,4) as year,
       COUNT(DISTINCT id)
FROM original_csv
GROUP BY 1 ORDER BY 1 DESC
```

2. Annotare il tempo di esecuzione della query e la quantità di dati analizzati.
3. Eseguire la stessa query nella nuova tabella, annotando il tempo di esecuzione della query e la quantità di dati analizzati.

```
SELECT year,
  COUNT(DISTINCT id)
FROM new_parquet
GROUP BY 1 ORDER BY 1 DESC
```

4. Confrontare i risultati e calcolare la differenza delle prestazioni e dei costi. I risultati di esempio riportati di seguito mostrano che la query di test nella nuova tabella è stata più veloce e più economica rispetto alla query nella vecchia tabella.

Tabella	Runtime	Dati scansionati
Originale	16,88 secondi	11,35 GB
Novità	3,79 secondi	428,05 MB

5. Eseguire la seguente query di esempio nella tabella originale. La query calcola la temperatura massima media (Celsius), la temperatura minima media (Celsius) e la piovosità media (mm) per la Terra nel 2018.

```
SELECT element, round(avg(CAST(datavalue AS real)/10),2) AS value
FROM original_csv
WHERE element IN ('TMIN', 'TMAX', 'PRCP') AND substr("date",1,4) = '2018'
GROUP BY 1
```

6. Annotare il tempo di esecuzione della query e la quantità di dati analizzati.
7. Eseguire la stessa query nella nuova tabella, annotando il tempo di esecuzione della query e la quantità di dati analizzati.

```
SELECT element, round(avg(CAST(datavalue AS real)/10),2) AS value
FROM new_parquet
WHERE element IN ('TMIN', 'TMAX', 'PRCP') and year = '2018'
GROUP BY 1
```

8. Confrontare i risultati e calcolare la differenza delle prestazioni e dei costi. I risultati di esempio riportati di seguito mostrano che la query di test nella nuova tabella è stata più veloce e più economica rispetto alla query nella vecchia tabella.

Tabella	Runtime	Dati scansionati
Originale	18,65 secondi	11,35 GB
Novità	1,92 secondi	68 MB

Riepilogo

In questo argomento viene illustrato come eseguire operazioni ETL utilizzando istruzioni CTAS e INSERT INTO in Athena. È stata eseguita la prima serie di trasformazioni utilizzando un'istruzione

CTAS che ha convertito i dati nel formato Parquet con compressione Snappy. L'istruzione CTAS ha anche convertito il set di dati da non partizionato a partizionato. Ciò ha ridotto le sue dimensioni e i costi di esecuzione delle query. Quando nuovi dati diventano disponibili, puoi utilizzare un'istruzione INSERT INTO per trasformare e caricare i dati nella tabella creata con l'istruzione CTAS.

Utilizzo di CTAS e INSERT INTO per aggirare il limite di 100 partizioni

Athena ha un limite di 100 partizioni per query `CREATE TABLE AS SELECT` (CTAS). Allo stesso modo, è possibile aggiungere un massimo di 100 partizioni a una tabella di destinazione con un'istruzione INSERT INTO.

Se superi questa limitazione potresti ricevere il messaggio di errore HIVE_TOO_MANY_OPEN_PARTITIONS: Exceeded limit of 100 open writers for partitions/buckets (HIVE_TOO_MANY_OPEN_PARTITIONS: limite di 100 scrittori aperti per partizioni/bucket superato). Per ovviare questa limitazione, puoi utilizzare un'istruzione CTAS e una serie di istruzioni `INSERT INTO` che creano o inseriscono fino a 100 partizioni ciascuna.

L'esempio in questo argomento utilizza un database chiamato i `tpch100` cui dati risiedono nella posizione del bucket Amazon S3 s3://DOC-EXAMPLE-BUCKET/.

Per utilizzare CTAS e INSERT INTO per creare una tabella con più di 100 partizioni

1. Utilizzare un'istruzione `CREATE EXTERNAL TABLE` per creare una tabella partizionata nel campo desiderato.

   L'istruzione di esempio seguente partiziona i dati dalla colonna `l_shipdate`. La tabella ha 2525 partizioni.

```
CREATE EXTERNAL TABLE `tpch100.lineitem_parq_partitioned`(
  `l_orderkey` int,
  `l_partkey` int,
  `l_suppkey` int,
  `l_linenumber` int,
  `l_quantity` double,
  `l_extendedprice` double,
  `l_discount` double,
  `l_tax` double,
  `l_returnflag` string,
  `l_linestatus` string,
  `l_commitdate` string,
  `l_receiptdate` string,
```

```
  `l_shipinstruct` string,
  `l_comment` string)
PARTITIONED BY (
  `l_shipdate` string)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe' STORED AS
 INPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat' OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat' LOCATION
 's3://DOC-EXAMPLE-BUCKET/lineitem/'
```

2. Eseguire un comando `SHOW PARTITIONS <table_name>` come il seguente per elencare le partizioni.

```
SHOW PARTITIONS lineitem_parq_partitioned
```

Di seguito sono riportati i risultati parziali di esempio.

```
/*
l_shipdate=1992-01-02
l_shipdate=1992-01-03
l_shipdate=1992-01-04
l_shipdate=1992-01-05
l_shipdate=1992-01-06

...

l_shipdate=1998-11-24
l_shipdate=1998-11-25
l_shipdate=1998-11-26
l_shipdate=1998-11-27
l_shipdate=1998-11-28
l_shipdate=1998-11-29
l_shipdate=1998-11-30
l_shipdate=1998-12-01
*/
```

3. Eseguire una query CTAS per creare una tabella partizionata.

L'esempio seguente crea una tabella chiamata `my_lineitem_parq_partitioned` e utilizza la clausola `WHERE` in modo che `DATE` sia precedente a `1992-02-01`. Poiché il set di dati di esempio inizia con gennaio 1992, vengono create solo le partizioni per gennaio 1992.

```
CREATE table my_lineitem_parq_partitioned
WITH (partitioned_by = ARRAY['l_shipdate']) AS
SELECT l_orderkey,
         l_partkey,
         l_suppkey,
         l_linenumber,
         l_quantity,
         l_extendedprice,
         l_discount,
         l_tax,
         l_returnflag,
         l_linestatus,
         l_commitdate,
         l_receiptdate,
         l_shipinstruct,
         l_comment,
         l_shipdate
FROM tpch100.lineitem_parq_partitioned
WHERE cast(l_shipdate as timestamp) < DATE ('1992-02-01');
```

4. Eseguire il comando `SHOW PARTITIONS` per verificare che la tabella contenga le partizioni desiderate.

```
SHOW PARTITIONS my_lineitem_parq_partitioned;
```

Le partizioni nell'esempio sono per gennaio 1992.

```
/*
l_shipdate=1992-01-02
l_shipdate=1992-01-03
l_shipdate=1992-01-04
l_shipdate=1992-01-05
l_shipdate=1992-01-06
l_shipdate=1992-01-07
l_shipdate=1992-01-08
l_shipdate=1992-01-09
l_shipdate=1992-01-10
l_shipdate=1992-01-11
l_shipdate=1992-01-12
l_shipdate=1992-01-13
l_shipdate=1992-01-14
```

```
l_shipdate=1992-01-15
l_shipdate=1992-01-16
l_shipdate=1992-01-17
l_shipdate=1992-01-18
l_shipdate=1992-01-19
l_shipdate=1992-01-20
l_shipdate=1992-01-21
l_shipdate=1992-01-22
l_shipdate=1992-01-23
l_shipdate=1992-01-24
l_shipdate=1992-01-25
l_shipdate=1992-01-26
l_shipdate=1992-01-27
l_shipdate=1992-01-28
l_shipdate=1992-01-29
l_shipdate=1992-01-30
l_shipdate=1992-01-31
*/
```

5. Utilizzare un'istruzione `INSERT INTO` per aggiungere le partizioni alla tabella.

   Nell'esempio seguente vengono aggiunte le partizioni per le date del mese di febbraio 1992.

```
INSERT INTO my_lineitem_parq_partitioned
SELECT l_orderkey,
         l_partkey,
         l_suppkey,
         l_linenumber,
         l_quantity,
         l_extendedprice,
         l_discount,
         l_tax,
         l_returnflag,
         l_linestatus,
         l_commitdate,
         l_receiptdate,
         l_shipinstruct,
         l_comment,
         l_shipdate
FROM tpch100.lineitem_parq_partitioned
WHERE cast(l_shipdate as timestamp) >= DATE ('1992-02-01')
AND cast(l_shipdate as timestamp) < DATE ('1992-03-01');
```

6. Eseguire nuovamente `SHOW PARTITIONS`.

```
SHOW PARTITIONS my_lineitem_parq_partitioned;
```

La tabella di esempio ora ha partizioni sia da gennaio che da febbraio 1992.

```
/*
l_shipdate=1992-01-02
l_shipdate=1992-01-03
l_shipdate=1992-01-04
l_shipdate=1992-01-05
l_shipdate=1992-01-06

...

l_shipdate=1992-02-20
l_shipdate=1992-02-21
l_shipdate=1992-02-22
l_shipdate=1992-02-23
l_shipdate=1992-02-24
l_shipdate=1992-02-25
l_shipdate=1992-02-26
l_shipdate=1992-02-27
l_shipdate=1992-02-28
l_shipdate=1992-02-29
*/
```

7. Continuare a utilizzare le istruzioni `INSERT INTO` che leggono e aggiungono non più di 100 partizioni ciascuna. Continuare fino a raggiungere il numero di partizioni necessarie.

> **Important**
>
> Quando imposti la condizione `WHERE`, assicurati che le query non si sovrappongano. In caso contrario, alcune partizioni potrebbero avere dati duplicati.

Documentazione di riferimento su SerDe

Athena supporta varie librerie SerDe per l'analisi dei dati da diversi formati di dati, ad esempio CSV, JSON, Parquet e ORC. Athena non supporta il servizio di SerDes personalizzato.

Argomenti

Usando un SerDe

A SerDe (Serializzatore/Deserializzatore) è un modo in cui Athena interagisce con i dati in vari formati.

È lo schema specificato dall' SerDe utente, e non il DDL, che definisce lo schema della tabella. In altre parole, SerDe possono sovrascrivere la configurazione DDL specificata in Athena quando si crea la tabella.

Per utilizzare un nelle interrogazioni SerDe

Per utilizzare a SerDe durante la creazione di una tabella in Athena, utilizzate uno dei seguenti metodi:

- Specifica `ROW FORMAT DELIMITED` e quindi utilizza le istruzioni DDL per specificare i separatori di campo, come nell'esempio seguente. Quando si specifica`ROW FORMAT DELIMITED`, Athena utilizza per impostazione LazySimpleSerDe predefinita.

  ```
  ROW FORMAT DELIMITED
  FIELDS TERMINATED BY ','
  ESCAPED BY '\\'
  COLLECTION ITEMS TERMINATED BY '|'
  MAP KEYS TERMINATED BY ':'
  ```

  Per esempi di `ROW FORMAT DELIMITED`, consulta i seguenti argomenti:

  LazySimpleSerDe per CSV, TSV e file delimitati in modo personalizzato

  Interrogazione dei log di Amazon CloudFront

  Esecuzione di query sui log Amazon EMR

  Esecuzione di query sui log di flusso Amazon VPC

  Utilizzo di CTAS e INSERT INTO per ETL e analisi dei dati
- `ROW FORMAT SERDE`Da utilizzare per specificare in modo esplicito il tipo di dati SerDe che Athena deve utilizzare per leggere e scrivere dati nella tabella. L'esempio seguente specifica il.

LazySimpleSerDe Per specificare i separatori, utilizzare `WITH SERDEPROPERTIES`. Le proprietà specificate da `WITH SERDEPROPERTIES` corrispondono alle istruzioni separate (come `FIELDS TERMINATED BY`) nell'esempio `ROW FORMAT DELIMITED`.

```
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'
WITH SERDEPROPERTIES (
'serialization.format' = ',',
'field.delim' = ',',
'collection.delim' = '|',
'mapkey.delim' = ':',
'escape.delim' = '\\'
)
```

Per esempi di `ROW FORMAT SERDE`, consulta i seguenti argomenti:

Avro SerDe

- Grok SerDe

Librerie JSON SerDe

OpenCSV per l'elaborazione di file SerDe CSV

Regex SerDe

Formati di SerDe e di dati supportati

Athena supporta la creazione di tabelle e le query sui dati da formati CSV, TSV, con delimitatori personalizzati e JSON; dati formati correlati a Hadoop: ORC, Apache Avro e Parquet; log di Logstash, di AWS CloudTrail e del server Web Apache.

> **Note**
>
> I formati elencati in questa sezione vengono utilizzati da Athena per la lettura di dati. Per informazioni sui formati che Athena impiega per scrivere i dati quando esegue query CTAS, consulta Creazione di una tabella dai risultati delle query (CTAS).

Per creare tabelle e query sui dati in questi formati in Athena, specifica una classe di serializzatore-deserializzatore (SerDe) in modo che Athena sappia che formato viene impiegato e come analizzare i dati.

Questa tabella elenca i formati di dati supportati in Athena e le relative librerie SerDe.

Un SerDe è una libreria personalizzata che indica al catalogo di dati utilizzato da Athena come gestire i dati. Puoi specificare un tipo di SerDe elencandolo esplicitamente nel `ROW FORMAT` che fa parte dell'istruzione `CREATE TABLE` in Athena. In alcuni casi, è possibile omettere il nome del SerDe, perché Athena utilizza alcuni tipi di SerDe predefiniti per alcuni tipi di formati di dati.

Formati di dati e di SerDe supportati

Formato dei dati	Descrizione	Tipi di SerDe supportati in Athena
Amazon Ion	Amazon Ion è un formato di dati altamente tipizzato e autodescrittivo ed è un superset di JSON, sviluppato e reso disponibile in open source da Amazon.	Utilizza il Amazon Ion Hive SerDe.
Apache Avro	Un formato per archiviare i dati in Hadoop che impiega schemi basati su JSON per i valori dei record.	Utilizzo della Avro SerDe
Apache Parquet	Un formato per storage colonnare di dati in Hadoop.	Utilizzare la compressione SNAPPY e Parquet SerDe.
Log del server Web Apache	Formato per archiviare log nel server Web di Apache.	Utilizzare - Grok SerDe o Regex SerDe.
Log di CloudTrail	Formato per archiviare log in CloudTrail.	• Utilizzo della JSON Hive SerDe. Per ulteriori informazioni, consulta Interrogazione dei log AWS CloudTrail.

Formato dei dati	Descrizione	Tipi di SerDe supportati in Athena
CSV (valori separati da virgola)	Per i dati in CSV, ogni riga rappresenta un record di dati e ogni record è composto da uno o più campi, separati da virgole.	• Utilizzare LazySimpleSerDe per CSV, TSV e file delimitati in modo personalizzato se i dati non includono valori racchiusi tra virgolette o se usano il formato `java.sql.Timestamp` . • Utilizzare OpenCSV per l'elaborazione di file SerDe CSV quando i dati includono virgolette nei valori o usano il formato numerico UNIX per `TIMESTAMP` (ad esempio, `1564610311`).
Delimitatore personalizzato	Per i dati in questo formato, ogni riga rappresenta un record di dati e i record sono separati da delimitatori personalizzati a carattere singolo.	Utilizzare LazySimpleSerDe per CSV, TSV e file delimitati in modo personalizzato e specificare un delimitatore di carattere singolo personalizzato.
JSON (JavaScript Object Notation)	Per i dati in JSON, ogni riga rappresenta un registro di dati e ogni registro è composto da coppie attributo-valore e da matrici, separate da virgole.	• Utilizzo della JSON Hive SerDe • Utilizzo della OpenX JSON SerDe
Log di Logstash	Formato per archiviare log in Logstash.	Utilizzo della - Grok SerDe
ORC (Optimized Row Columnar)	Un formato per storage colonnare ottimizzato basato su dati Hive.	Utilizzare la compressione ZLIB e ORCO SerDe.

Formato dei dati	Descrizione	Tipi di SerDe supportati in Athena
TSV (valori separati da tabulazione)	Per i dati in TSV, ogni riga rappresenta un record di dati e ogni record è composto da uno o più campi, separati da tabulazioni.	Utilizzare LazySimpleSerDe per CSV, TSV e file delimitati in modo personalizzato e specificare il carattere separatore come `FIELDS TERMINATED BY '\t'` .

Argomenti

- Amazon Ion Hive SerDe
- Avro SerDe
- - Grok SerDe
- Librerie JSON SerDe
- LazySimpleSerDe per CSV, TSV e file delimitati in modo personalizzato
- OpenCSV per l'elaborazione di file SerDe CSV
- ORCO SerDe
- Parquet SerDe
- Regex SerDe

Amazon Ion Hive SerDe

Puoi utilizzare Amazon Ion Hive SerDe per eseguire query sui dati archiviati nel formato Amazon Ion. Amazon Ion è un formato di dati open source altamente tipizzato e autodescrittivo. Il formato Amazon Ion viene utilizzato da servizi come Amazon Quantum Ledger Database (Amazon QLDB) e nel linguaggio di query SQL open source PartiQL.

Amazon Ion ha formati binari e testuali intercambiabili. Questa funzione combina la facilità d'uso del testo con l'efficienza della codifica binaria.

Per interrogare i dati Amazon Ion da Athena, è possibile utilizzare Amazon Ion Hive SerDe, che serializza e deserializza i dati Amazon Ion. La deserializzazione ti consente di eseguire query sui dati Amazon Ion o di leggerli per scriverli in un formato diverso come Parquet o ORC. La serializzazione ti

consente di generare dati nel formato Amazon Ion utilizzando `CREATE TABLE AS SELECT` (CTAS) o query `INSERT INTO` per copiare dati da tabelle esistenti.

> **Note**
>
> Poiché Amazon Ion è un superset di JSON, puoi utilizzare Amazon Ion Hive SerDE per eseguire query su set di dati non Amazon Ion JSON. A differenza di altre Librerie SerDe JSON, Amazon Ion SerDE non prevede che ogni riga di dati si trovi su una singola riga. Questa funzione è utile se si desidera eseguire query su set di dati JSON in formato «pretty print» o se si desidera suddividere i campi di una riga con caratteri di nuova riga.

Per ulteriori informazioni ed esempi di esecuzione di query su Amazon Ion con Athena, consulta Analisi di set di dati Amazon Ion con Amazon Athena.

Nome SerDe

- com.amazon.ionhiveserde.IonHiveSerDe

Considerazioni e limitazioni

- Campi duplicati: le strutture Amazon Ion vengono ordinate e supportano i campi duplicati, mentre `STRUCT<>` e `MAP<>` di Hive no. Pertanto, quando deserializzi un campo duplicato da una struttura Amazon Ion, viene scelto un singolo valore in modo non deterministico e gli altri vengono ignorati.
- Tabelle dei simboli esterni non supportate: attualmente Athena non supporta le tabelle dei simboli esterni o le seguenti proprietà Amazon Ion Hive SerDE:
  - `ion.catalog.class`
  - `ion.catalog.file`
  - `ion.catalog.url`
  - `ion.symbol_table_imports`
- Estensioni di file: Amazon Ion utilizza le estensioni di file per determinare quale codec di compressione utilizzare per deserializzare i file Amazon Ion. Pertanto, i file compressi devono avere l'estensione del file corrispondente all'algoritmo di compressione utilizzato. Ad esempio, se viene utilizzato ZSTD, i file corrispondenti dovrebbero avere l'estensione `.zst`.
- Dati omogenei: Amazon Ion non ha restrizioni sui tipi di dati che possono essere utilizzati per valori in campi specifici. Ad esempio, due documenti Amazon Ion diversi potrebbero avere un campo con

lo stesso nome con tipi di dati diversi. Tuttavia, poiché Hive utilizza uno schema, tutti i valori estratti in una singola colonna Hive devono avere lo stesso tipo di dati.

- Restrizioni sul tipo di chiave: quando serializzi dati da un altro formato in Amazon Ion, assicurati che il tipo di chiave mappa sia uno tra `STRING`, `VARCHAR`, oppure `CHAR`. Sebbene Hive ti consenta di utilizzare qualsiasi tipo di dati primitivo come chiave mappa, Simboli Amazon Ion deve essere un tipo di stringa.
- Tipo Union: Athena non supporta attualmente l'Hive tipo Union.
- Doppio tipo di dati: Amazon Ion attualmente non supporta questo tipo di dati `double`.

Argomenti

- Utilizzo di CREATE TABLE per creare tabelle Amazon Ion
- Utilizzo di CTAS e INSERT INTO per creare tabelle Amazon Ion
- Utilizzo delle proprietà di Amazon Ion SerDE
- Utilizzo degli estrattori di percorso

Utilizzo di CREATE TABLE per creare tabelle Amazon Ion

Per creare una tabella in Athena dai dati archiviati in formato Amazon Ion, è possibile utilizzare una delle seguenti tecniche in un'istruzione CREATE TABLE:

- Specifica `STORED AS ION`. In questo utilizzo, non è necessario specificare Amazon Ion Hive in SerDe modo esplicito. Questa è l'opzione più semplice.
- Specifica i percorsi della classe Amazon Ion nei campi `ROW FORMAT SERDE`, `INPUTFORMAT` e `OUTPUTFORMAT`.

È possibile utilizzare anche l'istruzione `CREATE TABLE AS SELECT` (CTAS) per creare tabelle Amazon Ion in Athena. Per informazioni, consulta Utilizzo di CTAS e INSERT INTO per creare tabelle Amazon Ion.

Specifica di STORED AS ION

La seguente istruzione di esempio `CREATE TABLE` utilizza `STORED AS ION` prima della clausola `LOCATION` per creare una tabella basata sui dati di volo in formato Amazon Ion. La clausola `LOCATION` specifica il bucket o la cartella in cui si trovano i file di input in formato Ion. Tutti i file nella posizione specificata vengono scansionati.

```
CREATE EXTERNAL TABLE flights_ion (
    yr INT,
    quarter INT,
    month INT,
    dayofmonth INT,
    dayofweek INT,
    flightdate STRING,
    uniquecarrier STRING,
    airlineid INT,
)
STORED AS ION
LOCATION 's3://DOC-EXAMPLE-BUCKET/'
```

Specifica dei percorsi della classe Amazon Ion

Invece di usare la sintassi `STORED AS ION`, è possibile specificare esplicitamente i valori del percorso della classe Ion per le clausole `ROW FORMAT SERDE`, `INPUTFORMAT` e `OUTPUTFORMAT` come segue.

Parametro	Percorso della classe Ion
`ROW FORMAT SERDE`	`'com.amazon.ionhiveserde.IonHiveSerDe'`
`STORED AS INPUTFORMAT`	`'com.amazon.ionhiveserde.formats.IonInputFormat'`
`OUTPUTFORMAT`	`'com.amazon.ionhiveserde.formats.IonOutputFormat'`

La seguente query DDL utilizza questa tecnica per creare la stessa tabella esterna dell'esempio precedente.

```
CREATE EXTERNAL TABLE flights_ion (
    yr INT,
    quarter INT,
    month INT,
    dayofmonth INT,
    dayofweek INT,
    flightdate STRING,
    uniquecarrier STRING,
```

```
    airlineid INT,
)
ROW FORMAT SERDE
 'com.amazon.ionhiveserde.IonHiveSerDe'
STORED AS INPUTFORMAT
 'com.amazon.ionhiveserde.formats.IonInputFormat'
OUTPUTFORMAT
 'com.amazon.ionhiveserde.formats.IonOutputFormat'
LOCATION 's3://DOC-EXAMPLE-BUCKET/'
```

Per informazioni sulle SerDe proprietà delle `CREATE TABLE` istruzioni in Athena, vedere. Utilizzo delle proprietà di Amazon Ion SerDE

Utilizzo di CTAS e INSERT INTO per creare tabelle Amazon Ion

Puoi utilizzare le istruzioni `CREATE TABLE AS SELECT` (CTAS) e `INSERT INTO` per copiare o inserire dati da una tabella ad una nuova in formato Amazon Ion in Athena.

In una query CTAS, specifica `format='ION'` nella clausola `WITH`, come nell'esempio seguente.

```
CREATE TABLE new_table
WITH (format='ION')
AS SELECT * from existing_table
```

Athena serializza di default i risultati di Amazon Ion in Formato binario Ion, ma è possibile anche utilizzare il formato di testo. Per utilizzare il formato di testo, specifica `ion_encoding = 'TEXT'` nella clausola CTAS `WITH`, come nell'esempio seguente.

```
CREATE TABLE new_table
WITH (format='ION', ion_encoding = 'TEXT')
AS SELECT * from existing_table
```

Per maggiori informazioni sulle proprietà specifiche di Amazon Ion nella clausola CTAS `WITH`, consulta la seguente sezione.

Proprietà Amazon Ion della clausola CTAS WITH

In una query CTAS, è possibile utilizzare la clausola `WITH` per specificare il formato Amazon Ion e facoltativamente specificare l'algoritmo di codifica e/o compressione di scrittura Amazon Ion da utilizzare.

format

È possibile specificare la parola chiave `ION` come opzione di formato nella clausola `WITH` di una query CTAS. In tal caso, la tabella creata utilizza il formato specificato per `IonInputFormat` per le letture e serializza i dati nel formato specificato per `IonOutputFormat`.

L'esempio seguente specifica che la query CTAS utilizza il formato Amazon Ion.

```
WITH (format='ION')
```

ion_encoding

Facoltativo

Impostazione predefinita: `BINARY`

Valori: `BINARY`, `TEXT`

Specifica se i dati sono serializzati in formato binario Amazon Ion o in formato di testo Amazon Ion. L'esempio seguente specifica il formato di testo Amazon Ion.

```
WITH (format='ION', ion_encoding='TEXT')
```

write_compression

Facoltativo

Impostazione predefinita: `GZIP`

Valori: `GZIP`, `ZSTD`, `BZIP2`, `SNAPPY`, `NONE`

Specifica l'algoritmo di compressione da utilizzare per comprimere i file di output.

L'esempio seguente specifica che la query CTAS scrive il suo output in formato Amazon Ion utilizzando l'algoritmo di compressione Zstandard.

```
WITH (format='ION', write_compression = 'ZSTD')
```

Per ulteriori informazioni sulla compressione in Athena, consulta Supporto della compressione in Athena.

Per ulteriori informazioni su altre proprietà CTAS in Athena, consulta Proprietà tabella CTAS.

Utilizzo delle proprietà di Amazon Ion SerDE

In questo argomento sono contenute informazioni sulle proprietà SerDe per le istruzioni `CREATE TABLE` in Athena. Per ulteriori informazioni ed esempi di utilizzo della proprietà Amazon Ion SerDe, consulta Proprietà SerDe nella documentazione di Amazon Ion Hive SerDE su GitHub.

Specifica delle proprietà di Amazon Ion SerDE

Per specificare le proprietà per Amazon Ion Hive SerDE nell'istruzione `CREATE TABLE`, usa la clausola `WITH SERDEPROPERTIES`. Poiché `WITH SERDEPROPERTIES` è un sottocampo della calusola `ROW FORMAT SERDE`, devi specificare innanzitutto `ROW FORMAT SERDE` e il percorso della classe Amazon Ion Hive SerDE, come mostrato nella seguente sintassi.

```
...
ROW FORMAT SERDE
 'com.amazon.ionhiveserde.IonHiveSerDe'
WITH SERDEPROPERTIES (
 'property' = 'value',
 'property' = 'value',
...
)
```

Notare che, anche se la clausola `ROW FORMAT SERDE` è obbligatoria, se si desidera utilizzare `WITH SERDEPROPERTIES`, è possibile utilizzare sia `STORED AS ION` che la sintassi più lunga `INPUTFORMAT` e `OUTPUTFORMAT` per specificare il formato Amazon Ion.

Proprietà di Amazon Ion SerDE

Di seguito sono riportate le proprietà Amazon Ion SerDE che possono essere utilizzate nelle istruzioni `CREATE TABLE` in Athena.

ion.encoding

Facoltativo

Impostazione predefinita: `BINARY`

Valori: `BINARY`, `TEXT`

Questa proprietà dichiara se i nuovi valori aggiunti sono serializzati come Amazon Ion o in formato testo Amazon Ion.

L'esempio di proprietà SerDE seguente specifica il formato di testo Amazon Ion.

```
'ion.encoding' = 'TEXT'
```

ion.fail_on_overflow

Facoltativo

Impostazione predefinita: `true`

Valori: `true`, `false`

Al contrario di Hive, Amazon Ion consente di utilizzare tipi numerici arbitrariamente grandi. SerDE dà di default esito negativo se il valore Amazon Ion non corrisponde alla colonna Hive, ma è possibile utilizzare l'opzione di configurazione `fail_on_overflow` per consentire l'overflow del valore anziché ottenere un esito negativo.

Questa proprietà può essere impostata a livello di tabella o colonna. Per specificarla a livello di tabella, occorre specificare `ion.fail_on_overflow` come nell'esempio seguente. Questo imposta di default il comportamento per tutte le colonne.

```
'ion.fail_on_overflow' = 'true'
```

Per controllare una colonna specifica, specifica il nome della colonna tra `ion` e `fail_on_overflow`, delimitato da punti, come nell'esempio seguente.

```
'ion.<column>.fail_on_overflow' = 'false'
```

ion.path_extractor.case_sensitive

Facoltativo

Impostazione predefinita: `false`

Valori: `true`, `false`

Determina se trattare i nomi dei campi Amazon Ion come maiuscole e minuscole. Quando il valore è `false`, SerDE ignora l'analisi dei nomi dei campi Amazon Ion.

Ad esempio, supponiamo di avere uno schema di tabella Hive che definisce un campo `alias` in minuscolo e un documento Amazon Ion con entrambi i campi `alias` e `ALIAS`, come nell'esempio seguente.

```
-- Hive Table Schema
alias: STRING

-- Amazon Ion Document
{ 'ALIAS': 'value1'}
{ 'alias': 'value2'}
```

L'esempio seguente mostra le proprietà SerDE e la tabella estratta risultante quando la distinzione tra maiuscole e minuscole è impostata su `false`:

```
-- Serde properties
'ion.alias.path_extractor' = '(alias)'
'ion.path_extractor.case_sensitive' = 'false'

--Extracted Table
| alias    |
|----------|
| "value1" |
| "value2" |
```

L'esempio seguente mostra le proprietà SerDE e la tabella estratta risultante quando la distinzione tra maiuscole e minuscole è impostata su `true`:

```
-- Serde properties
'ion.alias.path_extractor' = '(alias)'
'ion.path_extractor.case_sensitive' = 'true'

--Extracted Table
| alias    |
|----------|
| "value2" |
```

Nel secondo caso, il valore `value1` per il campo `ALIAS` viene ignorato quando la sensibilità tra maiuscole e minuscole è impostata su `true` e l'estrattore del percorso è specificato come `alias`.

Ion.*<column>*.path_extractor

Facoltativo

Valore predefinito: NA

Valori: stringa con percorso di ricerca

Crea un estrattore di percorso con il percorso di ricerca specificato per la colonna data. Gli estrattori di percorso mappano i campi Amazon Ion alle colonne Hive. Se non vengono specificati estrattori di percorso, Athena li crea dinamicamente in fase di esecuzione in base ai nomi delle colonne.

Il seguente esempio di estrattore di percorso mappa il `example_ion_field` al `example_hive_column`.

```
'ion.example_hive_column.path_extractor' = '(example_ion_field)'
```

Per ulteriori informazioni sugli estrattori di percorso e i percorsi di ricerca, consulta Utilizzo degli estrattori di percorso.

ion.timestamp.serialization_offset

Facoltativo

Impostazione predefinita: `'Z'`

Valori: `OFFSET`, dove `OFFSET` è rappresentato come *<signal>*`hh:mm`. Valori di esempio: `01:00`, `+01:00`, `-09:30`, `Z` (UTC 00:00)

A differenza dei timestamp di Apache Hive, che non hanno un fuso orario integrato e sono memorizzati come offset rispetto all'UNIX epoch, i timestamp Amazon Ion hanno un offset. Utilizza questa proprietà per specificare l'offset quando esegui la serializzazione su Amazon Ion.

L'esempio seguente aggiunge un offset di un'ora.

```
'ion.timestamp.serialization_offset' = '+01:00'
```

ion.serialize_null

Facoltativo

Impostazione predefinita: `OMIT`

Valori: `OMIT`, `UNTYPED`, `TYPED`

Amazon Ion SerDE può essere configurato per serializzare o omettere colonne con valori nulli. È possibile scegliere di scrivere null fortemente tipizzati (TYPED) o null non tipizzati (UNTYPED). I null fortemente tipizzati vengono determinati in base alla mappatura predefinita di tipo Amazon Ion to Hive.

L'esempio seguente specifica i valori null fortemente tipizzati.

```
'ion.serialize_null'='TYPED'
```

ion.ignore_malformed

Facoltativo

Impostazione predefinita: `false`

Valori: `true`, `false`

Quando è impostato `true`, ignora le voci malformate o l'intero file se SerDE non è in grado di leggerlo. Per ulteriori informazioni, consulta la documentazione relativa a ignora malformati su GitHub.

Ion.*<column>*.serialize_as

Facoltativo

Predefinito: tipo predefinito per la colonna.

Valori: stringa contenente il tipo Amazon Ion

Determina il tipo di dati Amazon Ion in cui viene serializzato un valore. Poiché i tipi Amazon Ion e Hive non hanno sempre una mappatura diretta, alcuni tipi di Hive hanno più tipi di dati validi per la serializzazione. Per serializzare i dati come tipo di dati non di default, utilizzare questa proprietà. Per ulteriori informazioni sulla mappatura dei tipi, consulta la pagina Tipo di mappatura di Amazon Ion su GitHub

Le colonne binarie Hive sono serializzate di default come BLOB Amazon Ion, ma possono anche essere serializzate come CLOB Amazon Ion (oggetto carattere grande). L'esempio seguente serializza la colonna `example_hive_binary_column` come un CLOB.

```
'ion.example_hive_binary_column.serialize_as' = 'clob'
```

Utilizzo degli estrattori di percorso

Amazon Ion ha un formato di file in stile documento, ma Apache Hive ha un formato a colonne semplice. Puoi utilizzare SerDe proprietà speciali di Amazon Ion chiamate `path extractors` per mappare tra i due formati. Gli estrattori di percorso appiattiscono il formato gerarchico di Amazon Ion, mappano i valori di Amazon Ion alle colonne di Hive e possono essere usati per rinominare i campi.

Athena può generare gli estrattori per conto tuo, ma è anche possibile definire estrattori personalizzati, se necessario.

Estrattori di percorso generati

Per impostazione predefinita, Athena cerca i valori Amazon Ion di primo livello che corrispondono ai nomi delle colonne Hive e crea estrattori di percorso in fase di runtime in base a questi valori. Se il formato dati Amazon Ion corrisponde allo schema della tabella Hive, Athena genera dinamicamente gli estrattori per conto tuo e non è necessario aggiungere ulteriori estrattori di percorso. Questi estrattori di percorso predefiniti non sono archiviati nei metadati della tabella.

Nell'esempio seguente viene illustrato come Athena genera estrattori in base al nome della colonna.

```
-- Example Amazon Ion Document
{
    identification: {
        name: "John Smith",
        driver_license: "XXXX"
    },

    alias: "Johnny"
}

-- Example DDL
CREATE EXTERNAL TABLE example_schema2 (
    identification MAP<STRING, STRING>,
    alias STRING
)
STORED AS ION
LOCATION 's3://DOC-EXAMPLE-BUCKET/path_extraction1/'
```

I seguenti estrattori di esempio sono generati da Athena. Il primo estrae il campo `identification` alla colonna `identification` e il secondo estrae il campo `alias` alla colonna `alias`.

```
'ion.identification.path_extractor' = '(identification)'
```

```
'ion.alias.path_extractor' = '(alias)'
```

La seguente query di esempio mostra una tabella estratta.

```
|                  identification                    |  alias   |
|----------------------------------------------------|----------|
|{["name", "driver_license"],["John Smith", "XXXX"]} | "Johnny" |
```

Definizione di estrattori di percorso personalizzati

Se i campi Amazon Ion non vengono mappati in modo ordinato alle colonne Hive, puoi specificare estrattori di percorso personalizzati. Nella clausola `WITH SERDEPROPERTIES` dell'istruzione `CREATE TABLE`, utilizza la seguente sintassi.

```
WITH SERDEPROPERTIES (
   "ion.path_extractor.case_sensitive" = "<Boolean>",
   "ion.<column_name>.path_extractor" = "<path_extractor_expression>"
)
```

> **Note**
>
> Per impostazione predefinita, gli estrattori di percorso non distinguono tra maiuscole e minuscole. Per ignorare questa impostazione, imposta la ion.path_extractor.case_sensitive SerDe proprietà `sutrue`.

Utilizzo di percorsi di ricerca negli estrattori di percorso

<path_extractor_expression>La sintassi della SerDe proprietà per l'estrattore di percorsi contiene:

```
"ion.<column_name>.path_extractor" = "<path_extractor_expression>"
```

Puoi utilizzare l'espressione *<path_extractor_expression>* per specificare un percorso di ricerca che analizza il documento Amazon Ion e trova i dati corrispondenti. Il percorso di ricerca è racchiuso tra parentesi e può contenere uno o più dei seguenti componenti separati da spazi.

- Carattere jolly: corrisponde a tutti i valori.
- Index (Indice): corrisponde al valore dell'indice numerico specificato. Gli indici sono a base zero.
- Testo: corrisponde a tutti i valori i cui nomi di campo corrispondono al testo specificato.

- Annotazioni: corrisponde ai valori specificati da un componente di percorso avvolto con le annotazioni specificate.

L'esempio seguente mostra un documento Amazon Ion e alcuni esempi di percorsi di ricerca.

```
-- Amazon Ion document
{
    foo: ["foo1", "foo2"] ,
    bar: "myBarValue",
    bar: A::"annotatedValue"
}

-- Example search paths
(foo 0)       # matches "foo1"
(1)           # matches "myBarValue"
(*)           # matches ["foo1", "foo2"], "myBarValue" and A::"annotatedValue"
()            # matches {foo: ["foo1", "foo2"] , bar: "myBarValue", bar:
 A::"annotatedValue"}
(bar)         # matches "myBarValue" and A::"annotatedValue"
(A::bar)      # matches A::"annotatedValue"
```

Esempi di estrattori

Appiattimento e ridenominazione dei campi

L'esempio seguente mostra una serie di percorsi di ricerca che appiattiscono e rinominano i campi. Nell'esempio vengono utilizzati i percorsi di ricerca per effettuare le operazioni seguenti:

- Mappatura della colonna `nickname` per il campo `alias`
- Mappatura della colonna `name` per il sottocampo `name` situato nella struttura `identification`.

Di seguito è riportato l'esempio di documento Amazon Ion.

```
-- Example Amazon Ion Document
{
    identification: {
        name: "John Smith",
        driver_license: "XXXX"
    },

    alias: "Johnny"
```

```
}
```

Di seguito è riportato l'esempio dell'istruzione `CREATE TABLE` che definisce gli estrattori di percorso.

```
-- Example DDL Query
CREATE EXTERNAL TABLE example_schema2 (
    name STRING,
    nickname STRING
)
ROW FORMAT SERDE
 'com.amazon.ionhiveserde.IonHiveSerDe'
WITH SERDEPROPERTIES (
 'ion.nickname.path_extractor' = '(alias)',
 'ion.name.path_extractor' = '(identification name)'
 )
STORED AS ION
LOCATION 's3://DOC-EXAMPLE-BUCKET/path_extraction2/'
```

L'esempio seguente mostra i dati estratti.

```
-- Extracted Table
| name         |   nickname   |
|--------------|--------------|
| "John Smith" |  "Johnny"    |
```

Per ulteriori informazioni sui percorsi di ricerca e altri esempi di percorsi di ricerca, vedere la pagina Ion Java Path Extraction su. GitHub

Estrazione dei dati di volo in formato testo

La seguente query di esempio `CREATE TABLE` utilizza `WITH SERDEPROPERTIES` per aggiungere estrattori di percorso per estrarre i dati di volo e specifica la codifica di output come testo Amazon Ion. Nell'esempio viene utilizzata la sintassi `STORED AS ION`.

```
CREATE EXTERNAL TABLE flights_ion (
    yr INT,
    quarter INT,
    month INT,
    dayofmonth INT,
    dayofweek INT,
    flightdate STRING,
```

```
    uniquecarrier STRING,
    airlineid INT,
)
ROW FORMAT SERDE
 'com.amazon.ionhiveserde.IonHiveSerDe'
WITH SERDEPROPERTIES (
 'ion.encoding' = 'TEXT',
 'ion.yr.path_extractor'='(year)',
 'ion.quarter.path_extractor'='(results quarter)',
 'ion.month.path_extractor'='(date month)')
STORED AS ION
LOCATION 's3://DOC-EXAMPLE-BUCKET/'
```

Avro SerDe

Nome SerDe

Avro SerDe

Nome della libreria

org.apache.hadoop.hive.serde2.avro.AvroSerDe

Esempi

Per motivi di sicurezza, Athena non supporta l'utilizzo di `avro.schema.url` per specificare lo schema di tabella. Utilizza `avro.schema.literal`. Per estrarre lo schema dai dati nel formato Avro, utilizza `avro-tools-<version>.jar` di Apache con il parametro `getschema`. Viene così restituito uno schema che è possibile utilizzare nell'istruzione `WITH SERDEPROPERTIES`. Ad esempio:

```
java -jar avro-tools-1.8.2.jar getschema my_data.avro
```

Il file `avro-tools-<version>.jar` file si trova nella sottodirectory `java` della tua release Avro installata. Per scaricare Avro, consulta la pagina relativa alle release di Apache Avro. Per scaricare direttamente gli strumenti Apache Avro, accedi all'apposita pagina del repository Maven Apache Avro.

Dopo aver ottenuto lo schema, utilizza un'istruzione `CREATE TABLE` per creare una tabella Athena basata sui dati Avro sottostanti archiviati in Amazon S3. Per specificare il SerDE Avro, utilizzare `ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.avro.AvroSerDe'`. Come

dimostrato nell'esempio seguente, è necessario specificare lo schema utilizzando la clausola `WITH SERDEPROPERTIES` oltre a specificare i nomi delle colonne e i tipi di dati corrispondenti per la tabella.

> **Note**
>
> Sostituire *myregion* in `s3://athena-examples-myregion/path/to/data/` con l'identificativo della Regione in cui si esegue Athena, ad esempio `s3://athena-examples-us-west-1/path/to/data/`.

```
CREATE EXTERNAL TABLE flights_avro_example (
   yr INT,
   flightdate STRING,
   uniquecarrier STRING,
   airlineid INT,
   carrier STRING,
   flightnum STRING,
   origin STRING,
   dest STRING,
   depdelay INT,
   carrierdelay INT,
   weatherdelay INT
)
PARTITIONED BY (year STRING)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.avro.AvroSerDe'
WITH SERDEPROPERTIES ('avro.schema.literal'='
{
   "type" : "record",
   "name" : "flights_avro_subset",
   "namespace" : "default",
   "fields" : [ {
      "name" : "yr",
      "type" : [ "null", "int" ],
      "default" : null
   }, {
      "name" : "flightdate",
      "type" : [ "null", "string" ],
      "default" : null
   }, {
      "name" : "uniquecarrier",
      "type" : [ "null", "string" ],
      "default" : null
```

```
   }, {
      "name" : "airlineid",
      "type" : [ "null", "int" ],
      "default" : null
   }, {
      "name" : "carrier",
      "type" : [ "null", "string" ],
      "default" : null
   }, {
      "name" : "flightnum",
      "type" : [ "null", "string" ],
      "default" : null
   }, {
      "name" : "origin",
      "type" : [ "null", "string" ],
      "default" : null
   }, {
      "name" : "dest",
      "type" : [ "null", "string" ],
      "default" : null
   }, {
      "name" : "depdelay",
      "type" : [ "null", "int" ],
      "default" : null
   }, {
      "name" : "carrierdelay",
      "type" : [ "null", "int" ],
      "default" : null
   }, {
      "name" : "weatherdelay",
      "type" : [ "null", "int" ],
      "default" : null
    } ]
}
')
STORED AS AVRO
LOCATION 's3://athena-examples-myregion/flight/avro/';
```

Esegui l'istruzione `MSCK REPAIR TABLE` sulla tabella per aggiornare i metadati della partizione.

```
MSCK REPAIR TABLE flights_avro_example;
```

Esegui una query per trovare le prime 10 città in base al numero totale di partenze.

```
SELECT origin, count(*) AS total_departures
FROM flights_avro_example
WHERE year >= '2000'
GROUP BY origin
ORDER BY total_departures DESC
LIMIT 10;
```

> **Note**
>
> I dati della tabella di volo provengono da Flights e sono forniti dal Dipartimento dei Trasporti degli Stati Uniti, Ufficio delle statistiche sui trasporti. Desaturati dall'originale.

- Grok SerDe

Logstash Grok SerDe è una libreria con una serie di modelli specializzati per la deserializzazione di dati di testo non strutturati, solitamente log. Ogni modello Grok è un'espressione regolare con nome. Puoi identificare e riutilizzare questi modelli di deserializzazione in base alle esigenze. Ciò rende più semplice utilizzare Grok anziché le espressioni regolari. Grok fornisce una serie di modelli predefiniti, tuttavia è possibile anche creare dei modelli personalizzati.

Per specificare Grok SerDe durante la creazione di una tabella in Athena, usa `ROW FORMAT SERDE 'com.amazonaws.glue.serde.GrokSerDe'` la clausola, seguita dalla clausola che specifica `WITH SERDEPROPERTIES` i modelli da abbinare nei tuoi dati, dove:

- L' espressione `input.format` definisce i modelli che devono corrispondere ai dati. È obbligatorio.
- L' espressione `input.grokCustomPatterns` definisce un modello personalizzato con nome, che potrà essere utilizzato successivamente all'interno dell'espressione `input.format`. È facoltativo. Per includere più voci di modello nell'espressione `input.grokCustomPatterns`, utilizza la carattere escape nuova riga (\n) per separarle, come segue: `'input.grokCustomPatterns'='INSIDE_QS ([^\"]*)\nINSIDE_BRACKETS ([^\\]]*)'`.
- Le clausole `STORED AS INPUTFORMAT` e `OUTPUTFORMAT` sono obbligatorie.
- La clausola `LOCATION` specifica un bucket Amazon S3 che può contenere più oggetti dati. Tutti gli oggetti dati presenti nel bucket vengono deserializzati per creare la tabella.

Esempi

Questi esempi si basano sull'elenco di modelli Grok predefiniti. A tal proposito, consulta la sezione relativa ai i modelli predefiniti.

Esempio 1

Questo esempio utilizza l'origine dati dalle voci del log mail Postfix salvate in `s3://DOC-EXAMPLE-BUCKET/groksample/`.

```
Feb  9 07:15:00 m4eastmail postfix/smtpd[19305]: B88C4120838: connect from
 unknown[192.168.55.4]
Feb  9 07:15:00 m4eastmail postfix/smtpd[20444]: B58C4330038:
 client=unknown[192.168.55.4]
Feb  9 07:15:03 m4eastmail postfix/cleanup[22835]: BDC22A77854: message-
id=<31221401257553.5004389LCBF@m4eastmail.example.com>
```

La seguente istruzione crea una tabella in Athena denominata `mygroktable` dall'origine dati, utilizzando un modello personalizzato e i modelli predefiniti da te specificati:

```
CREATE EXTERNAL TABLE `mygroktable`(
   syslogbase string,
   queue_id string,
   syslog_message string
   )
ROW FORMAT SERDE
   'com.amazonaws.glue.serde.GrokSerDe'
WITH SERDEPROPERTIES (
   'input.grokCustomPatterns' = 'POSTFIX_QUEUEID [0-9A-F]{7,12}',
   'input.format'='%{SYSLOGBASE} %{POSTFIX_QUEUEID:queue_id}:
 %{GREEDYDATA:syslog_message}'
   )
STORED AS INPUTFORMAT
   'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
   'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
   's3://DOC-EXAMPLE-BUCKET/groksample/';
```

Inizia dapprima con un modello semplice, ad esempio `%{NOTSPACE:column}`, per ottenere le colonne mappate; in seguito potrai specializzare le colonne, se necessario.

Esempio 2

In questo esempio, è necessario creare una query per i log di Log4j. Le voci dei log di esempio sono in questo formato:

```
2017-09-12 12:10:34,972 INFO  - processType=AZ, processId=ABCDEFG614B6F5E49,
 status=RUN,
threadId=123:amqListenerContainerPool23P:AJ|ABCDE9614B6F5E49||
2017-09-12T12:10:11.172-0700],
executionTime=7290, tenantId=12456, userId=123123f8535f8d76015374e7a1d87c3c,
 shard=testapp1,
jobId=12312345e5e7df0015e777fb2e03f3c, messageType=REAL_TIME_SYNC,
action=receive, hostname=1.abc.def.com
```

Per eseguire la query sui dati di questo log:

- Aggiungi il modello Grok per `input.format` per ogni colonna. Ad esempio, per `timestamp`, aggiungi `%{TIMESTAMP_ISO8601:timestamp}`. Per `loglevel`, aggiungi `%{LOGLEVEL:loglevel}`.
- Assicurati che il modello in `input.format` corrisponda esattamente al formato del log mappando i trattini (`-`) e le virgole che separano le voci nel formato di log.

```
CREATE EXTERNAL TABLE bltest (
 timestamp STRING,
 loglevel STRING,
 processtype STRING,
 processid STRING,
 status STRING,
 threadid STRING,
 executiontime INT,
 tenantid INT,
 userid STRING,
 shard STRING,
 jobid STRING,
 messagetype STRING,
 action STRING,
 hostname STRING
 )
ROW FORMAT SERDE 'com.amazonaws.glue.serde.GrokSerDe'
WITH SERDEPROPERTIES (
"input.grokCustomPatterns" = 'C_ACTION receive|send',
```

```
"input.format" = "%{TIMESTAMP_ISO8601:timestamp} %{LOGLEVEL:loglevel} - processType=
%{NOTSPACE:processtype}, processId=%{NOTSPACE:processid}, status=%{NOTSPACE:status},
 threadId=%{NOTSPACE:threadid}, executionTime=%{POSINT:executiontime}, tenantId=
%{POSINT:tenantid}, userId=%{NOTSPACE:userid}, shard=%{NOTSPACE:shard}, jobId=
%{NOTSPACE:jobid}, messageType=%{NOTSPACE:messagetype}, action=%{C_ACTION:action},
 hostname=%{HOST:hostname}"
) STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://DOC-EXAMPLE-BUCKET/samples/';
```

Esempio 3

L'esempio seguente di query dei log di Amazon S3 mostra l'espressione `'input.grokCustomPatterns'` che contiene due voci di modello separate dal carattere escape nuova riga (\n), come illustrato in questo frammento della query di esempio: `'input.grokCustomPatterns'='INSIDE_QS ([^\"]*)\nINSIDE_BRACKETS ([^\ \]]*)')`.

```
CREATE EXTERNAL TABLE `s3_access_auto_raw_02`(
  `bucket_owner` string COMMENT 'from deserializer',
  `bucket` string COMMENT 'from deserializer',
  `time` string COMMENT 'from deserializer',
  `remote_ip` string COMMENT 'from deserializer',
  `requester` string COMMENT 'from deserializer',
  `request_id` string COMMENT 'from deserializer',
  `operation` string COMMENT 'from deserializer',
  `key` string COMMENT 'from deserializer',
  `request_uri` string COMMENT 'from deserializer',
  `http_status` string COMMENT 'from deserializer',
  `error_code` string COMMENT 'from deserializer',
  `bytes_sent` string COMMENT 'from deserializer',
  `object_size` string COMMENT 'from deserializer',
  `total_time` string COMMENT 'from deserializer',
  `turnaround_time` string COMMENT 'from deserializer',
  `referrer` string COMMENT 'from deserializer',
  `user_agent` string COMMENT 'from deserializer',
  `version_id` string COMMENT 'from deserializer')
ROW FORMAT SERDE
  'com.amazonaws.glue.serde.GrokSerDe'
WITH SERDEPROPERTIES (
  'input.format'='%{NOTSPACE:bucket_owner} %{NOTSPACE:bucket} \
\[%{INSIDE_BRACKETS:time}\\] %{NOTSPACE:remote_ip} %{NOTSPACE:requester}
```

```
 %{NOTSPACE:request_id} %{NOTSPACE:operation} %{NOTSPACE:key} \"?
%{INSIDE_QS:request_uri}\"? %{NOTSPACE:http_status} %{NOTSPACE:error_code}
 %{NOTSPACE:bytes_sent} %{NOTSPACE:object_size} %{NOTSPACE:total_time}
 %{NOTSPACE:turnaround_time} \"?%{INSIDE_QS:referrer}\"? \"?%{INSIDE_QS:user_agent}\"?
 %{NOTSPACE:version_id}',
  'input.grokCustomPatterns'='INSIDE_QS ([^\"]*)\nINSIDE_BRACKETS ([^\\]]*)')
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET'
```

Librerie JSON SerDe

In Athena, puoi usare le SerDe librerie per deserializzare i dati JSON. La deserializzazione converte i dati JSON in modo che possano essere serializzati (scritti) in un formato diverso come Parquet o ORC.

- Il nativo JSON Hive SerDe
- Il OpenX JSON SerDe
- Il Amazon Ion Hive SerDe

> **Note**
>
> Le librerie Hive e OpenX prevedono che i dati JSON siano su una singola riga (non formattati), con registri separati da un carattere di nuova riga. Amazon Ion Hive SerDe non ha questo requisito e può essere usato come alternativa perché il formato dati Ion è un superset di JSON.

Nomi delle librerie

Utilizzare una delle seguenti operazioni:

org.apache.hive.hcatalog.data. JsonSerDe

org.openx.data.jsonserde. JsonSerDe

com.amazon.ionhiveserde. IonHiveSerDe

JSON Hive SerDe

Hive JSON SerDe è comunemente usato per elaborare dati JSON come eventi. Questi eventi sono rappresentati come stringhe su una sola riga di testo con codifica JSON separati da una nuova riga. Hive JSON SerDe non consente la duplicazione di chiavi o nomi di chiavi. `map` `struct`

> **Note**
>
> SerDe Si aspetta che ogni documento JSON si trovi su una singola riga di testo senza caratteri di terminazione di riga che separano i campi del record. Se il testo JSON è in un bel formato di stampa, potresti ricevere un messaggio di errore come HIVE_CURSOR_ERROR: Row is not a valid JSON Object o HIVE_CURSOR_ERROR:: Unexpected end-of-input: expected: expected close marker for OBJECT quando tenti di interrogare la tabella dopo averla JsonParseException creata. Per ulteriori informazioni, consulta JSON Data Files nella documentazione di SerDe OpenX su. GitHub

L'istruzione DDL di esempio seguente utilizza Hive JSON SerDe per creare una tabella basata su esempi di dati pubblicitari online. Nella clausola `LOCATION`, sostituire *`myregion`* in `s3://DOC-EXAMPLE-BUCKET.elasticmapreduce/samples/hive-ads/tables/impressions` con l'identificativo della Regione in cui si esegue Athena (ad esempio, `s3://us-west-2.elasticmapreduce/samples/hive-ads/tables/impressions`).

```
CREATE EXTERNAL TABLE impressions (
    requestbegintime string,
    adid string,
    impressionid string,
    referrer string,
    useragent string,
    usercookie string,
    ip string,
    number string,
    processid string,
    browsercookie string,
    requestendtime string,
    timers struct
                <
                 modellookup:string,
                 requesttime:string
                >,
    threadid string,
```

```
    hostname string,
    sessionid string
)
PARTITIONED BY (dt string)
ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe'
LOCATION 's3://DOC-EXAMPLE-BUCKET.elasticmapreduce/samples/hive-ads/tables/
impressions';
```

Specificazione dei formati di timestamp con Hive JSON SerDe

Per analizzare i valori del timestamp dalla stringa, è possibile aggiungere il sottocampo `WITH SERDEPROPERTIES` per la clausola `ROW FORMAT SERDE` e usarla per specificare il parametro `timestamp.formats`. Nel parametro specifica un elenco separato da virgole di uno o più modelli di timestamp, come nell'esempio seguente:

```
...
ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe'
WITH SERDEPROPERTIES ("timestamp.formats"="yyyy-MM-dd'T'HH:mm:ss.SSS'Z',yyyy-MM-
dd'T'HH:mm:ss")
...
```

Per ulteriori informazioni, consulta Marche temporali nella documentazione di Apache Hive.

Caricamento della tabella per l'esecuzione di query

Dopo aver creato la tabella, eseguire MSCK REPAIR TABLE per caricare la tabella e renderla eseguibile da Athena:

```
MSCK REPAIR TABLE impressions
```

CloudTrail Interrogazione dei log

Puoi usare Hive JSON SerDe per interrogare i log. CloudTrail Per ulteriori informazioni ed esempi delle istruzioni `CREATE TABLE`, consulta la pagina Interrogazione dei log AWS CloudTrail.

OpenX JSON SerDe

Come Hive JSON SerDe, puoi usare OpenX JSON per elaborare dati JSON. Questi dati sono rappresentati come stringhe su una sola riga di testo con codifica JSON separati da una nuova riga. Come Hive JSON SerDe, OpenX JSON SerDe non consente chiavi o nomi di chiavi duplicati. `map` `struct`

> **Note**
>
> SerDe Si aspetta che ogni documento JSON si trovi su una singola riga di testo senza caratteri di terminazione di riga che separano i campi del record. Se il testo JSON è in un bel formato di stampa, potresti ricevere un messaggio di errore come HIVE_CURSOR_ERROR: Row is not a valid JSON Object o HIVE_CURSOR_ERROR:: Unexpected end-of-input: expected: expected close marker for OBJECT quando tenti di interrogare la tabella dopo averla JsonParseException creata. Per ulteriori informazioni, consulta JSON Data Files nella documentazione di SerDe OpenX su. GitHub

Proprietà facoltative

A differenza di Hive JSON SerDe, OpenX JSON ha SerDe anche le seguenti SerDe proprietà opzionali che possono essere utili per risolvere le incongruenze nei dati.

ignore.malformed.json

Facoltativo. Quando è impostata su `TRUE`, consente di saltare la sintassi JSON errata. Il valore predefinito è `FALSE`.

dots.in.keys

Facoltativo. Il valore predefinito è `FALSE`. Se impostato su`TRUE`, consente di sostituire i punti nei nomi delle SerDe chiavi con caratteri di sottolineatura. Ad esempio, se il set di dati JSON contiene una chiave denominata `"a.b"`, è possibile usare questa proprietà per definire il nome della colonna come `"a_b"` in Athena. Per impostazione predefinita (senza questa opzione SerDe), Athena non consente l'uso di punti nei nomi delle colonne.

case.insensitive

Facoltativo. Il valore predefinito è `TRUE`. Se impostato su`TRUE`, SerDe converte tutte le colonne maiuscole in minuscole.

Per utilizzare i nomi di chiavi con distinzione tra maiuscole e minuscole nei dati, utilizzare `WITH SERDEPROPERTIES ("case.insensitive"= FALSE;)`. Quindi, per ogni chiave che non è già composta da sole minuscole, fornire un mapping dal nome della colonna al nome della proprietà utilizzando la sintassi seguente:

```
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
```

```
WITH SERDEPROPERTIES ("case.insensitive" = "FALSE", "mapping.userid" = "userId")
```

Se hai due chiavi tipo URL e `Url` che sono uguali quando sono in minuscolo, può verificarsi un errore come il seguente:

HIVE_CURSOR_ERROR: la riga non è un oggetto JSON valido - JSONException: "url" chiave duplicato

Per risolvere questo problema, impostare la proprietà `case.insensitive` su `FALSE` e mappare le chiavi a nomi diversi, come nell'esempio seguente:

```
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
WITH SERDEPROPERTIES ("case.insensitive" = "FALSE", "mapping.url1" = "URL",
 "mapping.url2" = "Url")
```

mappatura

Facoltativo. Mappa i nomi di colonna a chiavi JSON che non sono identiche ai nomi di colonna. Il parametro `mapping` è utile quando i dati JSON contengono chiavi che sono parole chiave. Ad esempio, se si dispone di una chiave JSON denominata `timestamp`, utilizzare la sintassi seguente per mappare la chiave a una colonna denominata `ts`:

```
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
WITH SERDEPROPERTIES ("mapping.ts" = "timestamp")
```

Mappatura dei nomi dei campi annidati con due punti a nomi compatibili con Hive

Se hai il nome di un campo con due punti all'interno di uno `struct`, puoi utilizzare la proprietà `mapping` per mappare il campo a un nome compatibile con Hive. Ad esempio, se le definizioni dei tipi di colonna contengono `my:struct:field:string`, puoi mappare la definizione a `my_struct_field:string` includendo la seguente voce in `WITH SERDEPROPERTIES`:

```
("mapping.my_struct_field" = "my:struct:field")
```

L'esempio seguente mostra l'istruzione `CREATE TABLE` corrispondente.

```
CREATE EXTERNAL TABLE colon_nested_field (
item struct<my_struct_field:string>)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
WITH SERDEPROPERTIES ("mapping.my_struct_field" = "my:struct:field")
```

Esempio: dati pubblicitari

L'istruzione DDL di esempio seguente utilizza OpenX SerDe JSON per creare una tabella basata sugli stessi dati pubblicitari online di esempio utilizzati nell'esempio per Hive JSON. SerDe Nella clausola `LOCATION` sostituire *`myregion`* con l'identificatore Regione in cui si esegue Athena.

```
CREATE EXTERNAL TABLE impressions (
    requestbegintime string,
    adid string,
    impressionId string,
    referrer string,
    useragent string,
    usercookie string,
    ip string,
    number string,
    processid string,
    browsercokie string,
    requestendtime string,
    timers struct<
       modellookup:string,
       requesttime:string>,
    threadid string,
    hostname string,
    sessionid string
)   PARTITIONED BY (dt string)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
LOCATION 's3://DOC-EXAMPLE-BUCKET.elasticmapreduce/samples/hive-ads/tables/
impressions';
```

Esempio di deserializzazione di un JSON nidificato

È possibile utilizzare JSON per analizzare dati con codifica JSON SerDes più complessi. Ciò richiede l'utilizzo di istruzioni `CREATE TABLE` che utilizzino elementi `struct` e `array` per rappresentare strutture nidificate.

Nell'esempio seguente viene creata una tabella Athena dai dati JSON con strutture nidificate. L'esempio ha la seguente struttura:

```
{
"DocId": "AWS",
"User": {
        "Id": 1234,
        "Username": "carlos_salazar",
```

```
        "Name": "Carlos",
"ShippingAddress": {
"Address1": "123 Main St.",
"Address2": null,
"City": "Anytown",
"State": "CA"
   },
"Orders": [
   {
     "ItemId": 6789,
     "OrderDate": "11/11/2022"
   },
   {
     "ItemId": 4352,
     "OrderDate": "12/12/2022"
   }
  ]
 }
}
```

Ricorda che OpenX SerDe si aspetta che ogni record JSON sia su una singola riga di testo. Se archiviati in Amazon S3, tutti i dati dell'esempio precedente devono trovarsi su un'unica riga, in questo modo:

```
{"DocId":"AWS","User":
{"Id":1234,"Username":"carlos_salazar","Name":"Carlos","ShippingAddress" ...
```

L'`CREATE TABLE`istruzione seguente utilizza i tipi di dati Openx- JsonSerDe with `struct` and `array` collection per stabilire gruppi di oggetti per i dati di esempio.

```
CREATE external TABLE complex_json (
   docid string,
   `user` struct<
               id:INT,
               username:string,
               name:string,
               shippingaddress:struct<
                                  address1:string,
                                  address2:string,
                                  city:string,
                                  state:string
                                  >,
```

```
               orders:array<
                           struct<
                                itemid:INT,
                                 orderdate:string
                                 >
                             >
               >
   )
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
LOCATION 's3://DOC-EXAMPLE-BUCKET/myjsondata/';
```

Per interrogare la tabella, utilizzate un'`SELECT`istruzione come la seguente.

```
SELECT
 user.name as Name,
 user.shippingaddress.address1 as Address,
 user.shippingaddress.city as City,
 o.itemid as Item_ID, o.orderdate as Order_date
FROM complex_json, UNNEST(user.orders) as temp_table (o)
```

Per accedere ai campi di dati all'interno delle strutture, la query di esempio utilizza la notazione a punti (ad esempio,`user.name`). Per accedere ai dati all'interno di una matrice di strutture (come nel `orders` campo), puoi usare la funzione. `UNNEST` La `UNNEST` funzione appiattisce l'array in una tabella temporanea (in questo caso chiamata). o Ciò consente di utilizzare la notazione a punti come si fa con le strutture per accedere agli elementi dell'array non annidati (ad esempio,). `o.itemid` Il nome`temp_table`, usato nell'esempio a scopo illustrativo, è spesso abbreviato in. `t`

La tabella seguente mostra i risultati dell'interrogazione.

#	Nome	Indirizzo	City	Item_ID	Data_ordine
1	Carlos	123 Main St.	Qualsiasi città	6789	11/11/2022
2	Carlos	123 Main St.	Qualsiasi città	4352	12/12/2022

Risorse aggiuntive

Per ulteriori informazioni sull'utilizzo di JSON e JSON nidificato in Athena, vedere le risorse seguenti:

- Crea tabelle in Amazon Athena da JSON annidato e mappature utilizzando JSON (Big Data Blog) SerDe AWS
- Ricevo errori quando cerco di leggere dati JSON in Amazon AthenaAWS (articolo del Knowledge Center)
- hive-json-schema(GitHub) — Strumento scritto in Java che genera `CREATE TABLE` istruzioni a partire da documenti JSON di esempio. Le istruzioni `CREATE TABLE` generate utilizzano SerDe JSON OpenX.

LazySimpleSerDe per CSV, TSV e file delimitati in modo personalizzato

La specificazione di questo valore SerDe è facoltativa. Si tratta dei SerDe dati in formato CSV, TSV e dei formati delimitati personalizzati utilizzati da Athena per impostazione predefinita. SerDe Viene utilizzato se non ne specifichi nessuno e lo specifichi solo. SerDe `ROW FORMAT DELIMITED` Usalo SerDe se i tuoi dati non hanno valori racchiusi tra virgolette.

Per la documentazione di riferimento su LazySimpleSerDe, consulta la SerDe sezione Hive della Apache Hive Developer Guide.

Nome della libreria

Il nome della libreria di classi per è. LazySimpleSerDe `org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe` Per informazioni sulla LazySimpleSerDe classe, vedete LazySimpleSerDe.java su GitHub .com.

Ignorare intestazioni

Per ignorare le intestazioni nei dati quando si definisce una tabella, è possibile utilizzare la proprietà `skip.header.line.count`, come nell'esempio seguente.

```
TBLPROPERTIES ("skip.header.line.count"="1")
```

Per alcuni esempi, consulta le istruzioni `CREATE TABLE` in Esecuzione di query sui log di flusso Amazon VPC e Interrogazione dei log di Amazon CloudFront .

Esempio di CSV

Il seguente esempio mostra come utilizzare `LazySimpleSerDe` per creare una tabella in Athena dai dati CSV. Per deserializzare i file delimitati in modo personalizzato utilizzando questa opzione SerDe, seguite lo schema riportato negli esempi ma utilizzate la `FIELDS TERMINATED BY` clausola per

specificare un delimitatore a carattere singolo diverso. LazySimpleSerDe non supporta delimitatori a più caratteri.

> **Note**
>
> Sostituire *myregion* in `s3://athena-examples-myregion/path/to/data/` con l'identificativo della Regione in cui si esegue Athena, ad esempio `s3://athena-examples-us-west-1/path/to/data/`.

Utilizza l'istruzione `CREATE TABLE` per creare una tabella Athena dai dati sottostanti in formato CSV archiviati in Amazon S3.

```
CREATE EXTERNAL TABLE flight_delays_csv (
    yr INT,
    quarter INT,
    month INT,
    dayofmonth INT,
    dayofweek INT,
    flightdate STRING,
    uniquecarrier STRING,
    airlineid INT,
    carrier STRING,
    tailnum STRING,
    flightnum STRING,
    originairportid INT,
    originairportseqid INT,
    origincitymarketid INT,
    origin STRING,
    origincityname STRING,
    originstate STRING,
    originstatefips STRING,
    originstatename STRING,
    originwac INT,
    destairportid INT,
    destairportseqid INT,
    destcitymarketid INT,
    dest STRING,
    destcityname STRING,
    deststate STRING,
    deststatefips STRING,
    deststatename STRING,
```

```
    destwac INT,
    crsdeptime STRING,
    deptime STRING,
    depdelay INT,
    depdelayminutes INT,
    depdel15 INT,
    departuredelaygroups INT,
    deptimeblk STRING,
    taxiout INT,
    wheelsoff STRING,
    wheelson STRING,
    taxiin INT,
    crsarrtime INT,
    arrtime STRING,
    arrdelay INT,
    arrdelayminutes INT,
    arrdel15 INT,
    arrivaldelaygroups INT,
    arrtimeblk STRING,
    cancelled INT,
    cancellationcode STRING,
    diverted INT,
    crselapsedtime INT,
    actualelapsedtime INT,
    airtime INT,
    flights INT,
    distance INT,
    distancegroup INT,
    carrierdelay INT,
    weatherdelay INT,
    nasdelay INT,
    securitydelay INT,
    lateaircraftdelay INT,
    firstdeptime STRING,
    totaladdgtime INT,
    longestaddgtime INT,
    divairportlandings INT,
    divreacheddest INT,
    divactualelapsedtime INT,
    divarrdelay INT,
    divdistance INT,
    div1airport STRING,
    div1airportid INT,
    div1airportseqid INT,
```

```
    div1wheelson STRING,
    div1totalgtime INT,
    div1longestgtime INT,
    div1wheelsoff STRING,
    div1tailnum STRING,
    div2airport STRING,
    div2airportid INT,
    div2airportseqid INT,
    div2wheelson STRING,
    div2totalgtime INT,
    div2longestgtime INT,
    div2wheelsoff STRING,
    div2tailnum STRING,
    div3airport STRING,
    div3airportid INT,
    div3airportseqid INT,
    div3wheelson STRING,
    div3totalgtime INT,
    div3longestgtime INT,
    div3wheelsoff STRING,
    div3tailnum STRING,
    div4airport STRING,
    div4airportid INT,
    div4airportseqid INT,
    div4wheelson STRING,
    div4totalgtime INT,
    div4longestgtime INT,
    div4wheelsoff STRING,
    div4tailnum STRING,
    div5airport STRING,
    div5airportid INT,
    div5airportseqid INT,
    div5wheelson STRING,
    div5totalgtime INT,
    div5longestgtime INT,
    div5wheelsoff STRING,
    div5tailnum STRING
)
    PARTITIONED BY (year STRING)
    ROW FORMAT DELIMITED
      FIELDS TERMINATED BY ','
      ESCAPED BY '\\'
      LINES TERMINATED BY '\n'
```

```
    LOCATION 's3://athena-examples-myregion/flight/csv/';
```

Esegui `MSCK REPAIR TABLE` per aggiornare i metadati della partizione ogni volta che una nuova partizione viene aggiunto a questa tabella:

```
MSCK REPAIR TABLE flight_delays_csv;
```

Esegui una query sui primi 10 voli con ritardo superiore a 1 ora:

```
SELECT origin, dest, count(*) as delays
FROM flight_delays_csv
WHERE depdelayminutes > 60
GROUP BY origin, dest
ORDER BY 3 DESC
LIMIT 10;
```

> **Note**
>
> I dati della tabella di volo provengono da Flights e sono forniti dal Dipartimento dei Trasporti degli Stati Uniti, Ufficio delle statistiche sui trasporti. Desaturati dall'originale.

Esempio di TSV

Per creare una tabella Athena dai dati TSV archiviati in Amazon S3, utilizza `ROW FORMAT DELIMITED` e specifica il `\t` come delimitatore del campo di tabulazione, `\n` come separatore di riga e `\` come carattere di escape. L'estratto seguente mostra questa sintassi. Nessun dato di volo TSV di esempio è disponibile nel percorso `athena-examples`, ma come per la tabella CSV, dovresti eseguire `MSCK REPAIR TABLE` per aggiornare i metadati di partizione ogni volta che viene aggiunta una nuova partizione.

```
...
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
ESCAPED BY '\\'
LINES TERMINATED BY '\n'
...
```

OpenCSV per l'elaborazione di file SerDe CSV

Quando crei una tabella Athena per dati CSV, determina la tabella SerDe da utilizzare in base ai tipi di valori contenuti nei dati:

- Se i tuoi dati contengono valori racchiusi tra virgolette doppie ('), puoi usare SerDeOpenCSV per deserializzare i valori in Athena. Se i tuoi dati non contengono valori racchiusi tra virgolette doppie ()', puoi omettere di specificarne uno. SerDe In questo caso, Athena impiega il `LazySimpleSerDe` predefinito. Per informazioni, consulta LazySimpleSerDe per CSV, TSV e file delimitati in modo personalizzato.
- Se i tuoi dati hanno `TIMESTAMP` valori numerici UNIX (ad esempio,`1579059880000`), usa OpenCSV. SerDe Se i tuoi dati utilizzano il formato, usa il. `java.sql.Timestamp` LazySimpleSerDe

CSV SerDe (SerDeOpenCSV)

SerDeOpenCSV ha le seguenti caratteristiche per i dati di tipo stringa:

- Utilizza doppie virgolette (') come virgolette predefinite e consente di specificare separatore, virgolette e caratteri di escape, ad esempio:

```
WITH SERDEPROPERTIES ("separatorChar" = ",", "quoteChar" = "`", "escapeChar" = "\\" )
```

- Impossibile utilizzare direttamente il carattere di escape `\t` o `\n`. Per utilizzarli come caratteri di escape, utilizza `"escapeChar" = "\\"`. Consulta l'esempio riportato in questo argomento.
- Non supporta le interruzioni di linea incorporate nei file CSV.

Per tipi di dati diversi da`STRING`, OpenCSV si comporta come SerDe segue:

- Riconosce i tipi di dati `BOOLEAN`, `BIGINT`, `INT` e `DOUBLE`.
- Non riconosce valori vuoti o nulli nelle colonne definite come un tipo di dati numerici, lasciandoli come `string`. Una soluzione alternativa è creare la colonna con i valori nulli come`string` e quindi utilizzare `CAST` per convertire il campo in una query per un tipo di dati numerico, fornendo un valore predefinito `0` per i valori nulli. Per ulteriori informazioni, consulta Quando eseguo una query sui dati CSV in Athena, ricevo l'errore HIVE_BAD_DATA: errore nell'analisi del valore del campo nel Knowledge Center. AWS

- Per le colonne specificate con il tipo di dati `timestamp` nell'istruzione `CREATE TABLE`, riconosce i dati `TIMESTAMP` se sono specificati nel formato numerico UNIX in millisecondi, ad esempio `1579059880000`.
  - OpenCSV non `TIMESTAMP` supporta il SerDe formato `java.sql.Timestamp` compatibile con JDBC, ad esempio (precisione a 9 cifre decimali). `"YYYY-MM-DD HH:MM:SS.fffffffff"`
- Per le colonne specificate con il tipo di dati `DATE` nell'istruzione `CREATE TABLE`, riconosce i valori come date se i valori rappresentano il numero di giorni trascorsi dal 1° gennaio 1970. Ad esempio, il valore `18276` in una colonna con il tipo di dati `date` viene eseguito come `2020-01-15` quando viene interrogato. In questo formato UNIX, si considera che ogni giorno abbia 86.400 secondi.
  - OpenCSV non `DATE` supporta SerDe direttamente nessun altro formato. Per elaborare i dati della marca temporale in altri formati, è possibile definire la colonna come `string` e quindi utilizzare le funzioni di conversione del tempo per restituire i risultati desiderati nella query `SELECT`. Per ulteriori informazioni, consulta l'articolo Quando eseguo una query su una tabella in Amazon Athena, il risultato TIMESTAMP è vuoto nel Portale del sapere di AWS.
- Per convertire ulteriormente le colonne nel tipo desiderato in una tabella, è possibile creare una vista della tabella e utilizzare `CAST` per convertirle nel tipo desiderato.

Example Esempio: utilizzo del tipo TIMESTAMP e del tipo DATE specificati nel formato numerico UNIX.

Considerare le tre seguenti colonne di dati separati da virgole. I valori in ciascuna colonna sono racchiusi tra virgolette doppie.

```
"unixvalue creationdate 18276 creationdatetime 1579059880000","18276","1579059880000"
```

L'istruzione seguente crea una tabella in Athena dal percorso del bucket Amazon S3.

```
CREATE EXTERNAL TABLE IF NOT EXISTS testtimestamp1(
 `profile_id` string,
 `creationdate` date,
 `creationdatetime` timestamp
 )
 ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
 LOCATION 's3://DOC-EXAMPLE-BUCKET'
```

Esegui quindi la seguente query:

```
SELECT * FROM testtimestamp1
```

La query restituisce il seguente risultato, mostrando i dati di data e ora:

```
profile_id                                                          creationdate
 creationdatetime
unixvalue creationdate 18276 creationdatetime 1579146280000        2020-01-15
 2020-01-15 03:44:40.000
```

Example Esempio: utilizzo di **\t** o **\n** come caratteri escape

Tieni in considerazione i seguenti dati di verifica:

```
" \\t\\t\\n 123 \\t\\t\\n ",abc
" 456 ",xyz
```

La seguente istruzione crea una tabella in Athena, specificando `"escapeChar" = "\\"`.

```
CREATE EXTERNAL TABLE test1 (
f1 string,
s2 string)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES ("separatorChar" = ",", "escapeChar" = "\\")
LOCATION 's3://DOC-EXAMPLE-BUCKET/dataset/test1/'
```

Esegui quindi la seguente query:

```
SELECT * FROM test1;
```

Restituisce questo risultato, utilizzando correttamente `\t` o `\n` come caratteri di escape:

```
f1            s2
\t\t\n 123 \t\t\n            abc
456                          xyz
```

SerDe nome

CSV SerDe

Nome della libreria

Per utilizzarlo SerDe, specifica il nome completo della classe dopo`ROW FORMAT SERDE`. Specificare anche i delimitatori all'interno di `SERDEPROPERTIES` nel modo seguente:

```
...
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
  "separatorChar" = ",",
  "quoteChar"     = "`",
  "escapeChar"    = "\\"
)
```

Ignorare intestazioni

Per ignorare le intestazioni nei dati quando si definisce una tabella, è possibile utilizzare la proprietà `skip.header.line.count`, come nell'esempio seguente.

```
TBLPROPERTIES ("skip.header.line.count"="1")
```

Per alcuni esempi, consulta le istruzioni `CREATE TABLE` in Esecuzione di query sui log di flusso Amazon VPC e Interrogazione dei log di Amazon CloudFront .

Esempio

Questo esempio presuppone la presenza di dati in CSV salvati in `s3://DOC-EXAMPLE-BUCKET/mycsv/` con il seguente contenuto:

```
"a1","a2","a3","a4"
"1","2","abc","def"
"a","a1","abc3","ab4"
```

Utilizza un'istruzione `CREATE TABLE` per creare una tabella Athena basata sui dati. Fate riferimento alla classe OpenCSV SerDe `ROW FORMAT SERDE` dopo e specificate il separatore di caratteri, il carattere di virgoletta e il carattere di escape in, come `WITH SERDEPROPERTIES` nell'esempio seguente.

```
CREATE EXTERNAL TABLE myopencsvtable (
   col1 string,
   col2 string,
   col3 string,
```

```
   col4 string
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
   'separatorChar' = ',',
   'quoteChar' = '"',
   'escapeChar' = '\\'
   )
STORED AS TEXTFILE
LOCATION 's3://DOC-EXAMPLE-BUCKET/mycsv/';
```

Crea una query di tutti i valori nella tabella:

```
SELECT * FROM myopencsvtable;
```

La query restituisce i seguenti valori:

```
col1     col2    col3    col4
-----------------------------
a1       a2      a3      a4
1        2       abc     def
a        a1      abc3    ab4
```

ORCO SerDe

SerDe nome

OrcSerDe

Nome della libreria

Questa libreria utilizza la classe Apache Hive OrcSerde.java per i dati in formato ORC. Questa classe passa l'oggetto da ORC al lettore e da ORC al writer: OrcSerDe

Esempi

> **Note**
>
> Sostituire *myregion* in `s3://athena-examples-myregion/path/to/data/` con l'identificativo della Regione in cui si esegue Athena, ad esempio `s3://athena-examples-us-west-1/path/to/data/`.

Nell'esempio seguente viene creata una tabella per i dati in formato ORC relativi ai ritardi dei voli. La tabella include partizioni:

```
DROP TABLE flight_delays_orc;
CREATE EXTERNAL TABLE flight_delays_orc (
    yr INT,
    quarter INT,
    month INT,
    dayofmonth INT,
    dayofweek INT,
    flightdate STRING,
    uniquecarrier STRING,
    airlineid INT,
    carrier STRING,
    tailnum STRING,
    flightnum STRING,
    originairportid INT,
    originairportseqid INT,
    origincitymarketid INT,
    origin STRING,
    origincityname STRING,
    originstate STRING,
    originstatefips STRING,
    originstatename STRING,
    originwac INT,
    destairportid INT,
    destairportseqid INT,
    destcitymarketid INT,
    dest STRING,
    destcityname STRING,
    deststate STRING,
    deststatefips STRING,
    deststatename STRING,
    destwac INT,
    crsdeptime STRING,
    deptime STRING,
    depdelay INT,
    depdelayminutes INT,
    depdel15 INT,
    departuredelaygroups INT,
    deptimeblk STRING,
    taxiout INT,
    wheelsoff STRING,
```

```
    wheelson STRING,
    taxiin INT,
    crsarrtime INT,
    arrtime STRING,
    arrdelay INT,
    arrdelayminutes INT,
    arrdel15 INT,
    arrivaldelaygroups INT,
    arrtimeblk STRING,
    cancelled INT,
    cancellationcode STRING,
    diverted INT,
    crselapsedtime INT,
    actualelapsedtime INT,
    airtime INT,
    flights INT,
    distance INT,
    distancegroup INT,
    carrierdelay INT,
    weatherdelay INT,
    nasdelay INT,
    securitydelay INT,
    lateaircraftdelay INT,
    firstdeptime STRING,
    totaladdgtime INT,
    longestaddgtime INT,
    divairportlandings INT,
    divreacheddest INT,
    divactualelapsedtime INT,
    divarrdelay INT,
    divdistance INT,
    div1airport STRING,
    div1airportid INT,
    div1airportseqid INT,
    div1wheelson STRING,
    div1totalgtime INT,
    div1longestgtime INT,
    div1wheelsoff STRING,
    div1tailnum STRING,
    div2airport STRING,
    div2airportid INT,
    div2airportseqid INT,
    div2wheelson STRING,
    div2totalgtime INT,
```

```
    div2longestgtime INT,
    div2wheelsoff STRING,
    div2tailnum STRING,
    div3airport STRING,
    div3airportid INT,
    div3airportseqid INT,
    div3wheelson STRING,
    div3totalgtime INT,
    div3longestgtime INT,
    div3wheelsoff STRING,
    div3tailnum STRING,
    div4airport STRING,
    div4airportid INT,
    div4airportseqid INT,
    div4wheelson STRING,
    div4totalgtime INT,
    div4longestgtime INT,
    div4wheelsoff STRING,
    div4tailnum STRING,
    div5airport STRING,
    div5airportid INT,
    div5airportseqid INT,
    div5wheelson STRING,
    div5totalgtime INT,
    div5longestgtime INT,
    div5wheelsoff STRING,
    div5tailnum STRING
)
PARTITIONED BY (year String)
STORED AS ORC
LOCATION 's3://athena-examples-myregion/flight/orc/'
tblproperties ("orc.compress"="ZLIB");
```

Esegui l'istruzione `MSCK REPAIR TABLE` sulla tabella per aggiornare i metadati della partizione:

```
MSCK REPAIR TABLE flight_delays_orc;
```

Utilizza questa query per ottenere i primi 10 voli in ritardo da più di 1 ora:

```
SELECT origin, dest, count(*) as delays
FROM flight_delays_orc
WHERE depdelayminutes > 60
GROUP BY origin, dest
```

```
ORDER BY 3 DESC
LIMIT 10;
```

Parquet SerDe

SerDe nome

ParquetHiveSerDe viene utilizzato per i dati memorizzati in formato Parquet.

> **Note**
>
> Per convertire i dati in formato Parquet, è possibile utilizzare le query CREATE TABLE AS SELECT (CTAS) . Per ulteriori informazioni, consultare Creazione di una tabella dai risultati delle query (CTAS), Esempi di query CTAS e Utilizzo di CTAS e INSERT INTO per ETL e analisi dei dati.

Nome della libreria

Athena utilizza la seguente classe quando deve deserializzare dati archiviati in Parquet: `org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe`

Esempio di esecuzione di query su un file archiviato in Parquet

> **Note**
>
> Sostituire *myregion* in `s3://athena-examples-myregion/path/to/data/` con l'identificativo della Regione in cui si esegue Athena, ad esempio `s3://athena-examples-us-west-1/path/to/data/`.

Utilizza la seguente `CREATE TABLE` istruzione per creare una tabella Athena dai dati sottostanti archiviati in formato Parquet in Amazon S3:

```
CREATE EXTERNAL TABLE flight_delays_pq (
    yr INT,
    quarter INT,
    month INT,
    dayofmonth INT,
    dayofweek INT,
    flightdate STRING,
```

```
    uniquecarrier STRING,
    airlineid INT,
    carrier STRING,
    tailnum STRING,
    flightnum STRING,
    originairportid INT,
    originairportseqid INT,
    origincitymarketid INT,
    origin STRING,
    origincityname STRING,
    originstate STRING,
    originstatefips STRING,
    originstatename STRING,
    originwac INT,
    destairportid INT,
    destairportseqid INT,
    destcitymarketid INT,
    dest STRING,
    destcityname STRING,
    deststate STRING,
    deststatefips STRING,
    deststatename STRING,
    destwac INT,
    crsdeptime STRING,
    deptime STRING,
    depdelay INT,
    depdelayminutes INT,
    depdel15 INT,
    departuredelaygroups INT,
    deptimeblk STRING,
    taxiout INT,
    wheelsoff STRING,
    wheelson STRING,
    taxiin INT,
    crsarrtime INT,
    arrtime STRING,
    arrdelay INT,
    arrdelayminutes INT,
    arrdel15 INT,
    arrivaldelaygroups INT,
    arrtimeblk STRING,
    cancelled INT,
    cancellationcode STRING,
    diverted INT,
```

```
    crselapsedtime INT,
    actualelapsedtime INT,
    airtime INT,
    flights INT,
    distance INT,
    distancegroup INT,
    carrierdelay INT,
    weatherdelay INT,
    nasdelay INT,
    securitydelay INT,
    lateaircraftdelay INT,
    firstdeptime STRING,
    totaladdgtime INT,
    longestaddgtime INT,
    divairportlandings INT,
    divreacheddest INT,
    divactualelapsedtime INT,
    divarrdelay INT,
    divdistance INT,
    div1airport STRING,
    div1airportid INT,
    div1airportseqid INT,
    div1wheelson STRING,
    div1totalgtime INT,
    div1longestgtime INT,
    div1wheelsoff STRING,
    div1tailnum STRING,
    div2airport STRING,
    div2airportid INT,
    div2airportseqid INT,
    div2wheelson STRING,
    div2totalgtime INT,
    div2longestgtime INT,
    div2wheelsoff STRING,
    div2tailnum STRING,
    div3airport STRING,
    div3airportid INT,
    div3airportseqid INT,
    div3wheelson STRING,
    div3totalgtime INT,
    div3longestgtime INT,
    div3wheelsoff STRING,
    div3tailnum STRING,
    div4airport STRING,
```

```
    div4airportid INT,
    div4airportseqid INT,
    div4wheelson STRING,
    div4totalgtime INT,
    div4longestgtime INT,
    div4wheelsoff STRING,
    div4tailnum STRING,
    div5airport STRING,
    div5airportid INT,
    div5airportseqid INT,
    div5wheelson STRING,
    div5totalgtime INT,
    div5longestgtime INT,
    div5wheelsoff STRING,
    div5tailnum STRING
)
PARTITIONED BY (year STRING)
STORED AS PARQUET
LOCATION 's3://athena-examples-myregion/flight/parquet/'
tblproperties ("parquet.compression"="SNAPPY");
```

Esegui l'istruzione `MSCK REPAIR TABLE` sulla tabella per aggiornare i metadati della partizione:

```
MSCK REPAIR TABLE flight_delays_pq;
```

Esegui una query sui primi 10 voli con ritardo superiore a 1 ora:

```
SELECT origin, dest, count(*) as delays
FROM flight_delays_pq
WHERE depdelayminutes > 60
GROUP BY origin, dest
ORDER BY 3 DESC
LIMIT 10;
```

> **Note**
>
> I dati della tabella di volo provengono da Flights e sono forniti dal Dipartimento dei Trasporti degli Stati Uniti, Ufficio delle statistiche sui trasporti. Desaturati dall'originale.

Come ignorare le statistiche Parquet

Quando leggi i dati di Parquet, potresti ricevere messaggi di errore simili a quelli che seguono:

```
HIVE_CANNOT_OPEN_SPLIT: Index x out of bounds for length y
HIVE_CURSOR_ERROR: Failed to read x bytes
HIVE_CURSOR_ERROR: FailureException at Malformed input: offset=x
HIVE_CURSOR_ERROR: FailureException at java.io.IOException:
can not read class org.apache.parquet.format.PageHeader: Socket is closed by peer.
```

Per risolvere questo problema, usa l'ALTER TABLE SET TBLPROPERTIESistruzione CREATE TABLE or per impostare la SerDe `parquet.ignore.statistics` proprietà Parquet su`true`, come negli esempi seguenti.

Esempio di CREATE TABLE

```
...
ROW FORMAT SERDE
'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
WITH SERDEPROPERTIES (
'parquet.ignore.statistics'='true')
STORED AS PARQUET
...
```

Esempio di ALTER TABLE

```
ALTER TABLE ... SET TBLPROPERTIES ('parquet.ignore.statistics'='true')
```

Regex SerDe

Regex SerDe utilizza un'espressione regolare (regex) per deserializzare i dati estraendo i gruppi regex nelle colonne della tabella.

Se una riga nei dati non corrisponde a regex, tutte le colonne nella riga vengono restituite come `NULL`. Se una riga corrisponde a regex ma ha meno gruppi del previsto, i gruppi mancanti sono `NULL`. Se una riga nei dati corrisponde a regex ma contiene più colonne rispetto ai gruppi in regex, le colonne aggiuntive vengono ignorate.

Per ulteriori informazioni, consulta Class RegexSerDe nella documentazione di Apache Hive.

SerDe nome

RegexSerDe

Nome della libreria

RegexSerDe

Esempi

L'esempio seguente crea una tabella dai CloudFront log utilizzando. RegExSerDe Sostituire *myregion* in `s3://athena-examples-myregion/cloudfront/plaintext/` con l'identificativo della Regione in cui si esegue Athena (ad esempio `s3://athena-examples-us-west-1/cloudfront/plaintext/`).

```
CREATE EXTERNAL TABLE IF NOT EXISTS cloudfront_logs (
  `Date` DATE,
  Time STRING,
  Location STRING,
  Bytes INT,
  RequestIP STRING,
  Method STRING,
  Host STRING,
  Uri STRING,
  Status INT,
  Referrer STRING,
  os STRING,
  Browser STRING,
  BrowserVersion STRING
 )
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
 WITH SERDEPROPERTIES (
 "input.regex" = "^(?!#)([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\
\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+([^ ]+)\\s+[^\(]+[\(]([^\;]+).*\%20([^\/]+)[\/]
(.*)$"
 )
LOCATION 's3://athena-examples-myregion/cloudfront/plaintext/';
```

Esecuzione di query SQL con Amazon Athena

Puoi eseguire query SQL mediante Amazon Athena su origini dati registrate con AWS Glue Data Catalog e le origini dati, come i metastore Hive e le istanze Amazon DocumentDB cui ti connetti

usando la funzione Athena Federated Query. Per ulteriori informazioni sull'utilizzo delle origini dati, consulta Connessione alle origini dati. Quando esegui una query DDL (Data Definition Language) che modifica lo schema, Athena scrive i metadati nel metastore associato all'origine dati. Inoltre, alcune query, come `CREATE TABLE AS` e `INSERT INTO`, possono scrivere record nel set di dati, ad esempio aggiungendo un record CSV a una posizione Amazon S3. Quando esegui una query, Athena salva i risultati di una query in una posizione dei risultati delle query specificata. Ciò consente di visualizzare la cronologia delle query e di scaricare e visualizzare i set di risultati delle query.

In questa sezione vengono fornite indicazioni per l'esecuzione di query Athena su origini dati comuni e tipi di dati utilizzando diverse istruzioni SQL. Vengono fornite indicazioni generali per lavorare con strutture e operatori comuni, ad esempio per lavorare con array, concatenare, filtrare, appiattire e ordinare. Altri esempi includono query per dati in tabelle con strutture e mappe annidate, tabelle basate su set di dati con codifica JSON e set di dati associati a log e log di Amazon EMR. Servizi AWS AWS CloudTrail La copertura completa dell'utilizzo di SQL standard non viene riportata in questa documentazione. Per ulteriori informazioni su SQL, consulta i riferimenti ai linguaggi Trino e Presto.

Argomenti

- Esecuzione di query tra regioni
- Esecuzione di query AWS Glue Data Catalog
- Interrogazione dei log Servizio AWS
- Esecuzione di query sui log del server Web archiviati in Amazon S3

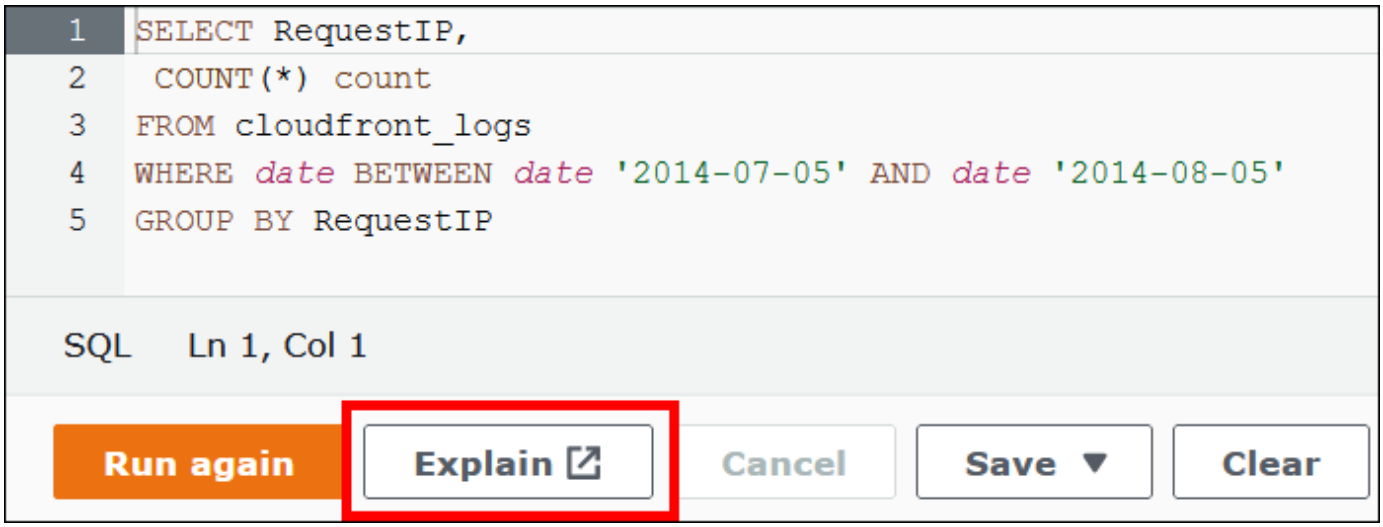
Per considerazioni e limitazioni, vedere Considerazioni e restrizioni per le query SQL in Amazon Athena.

Visualizzazione dei piani di esecuzione per query SQL

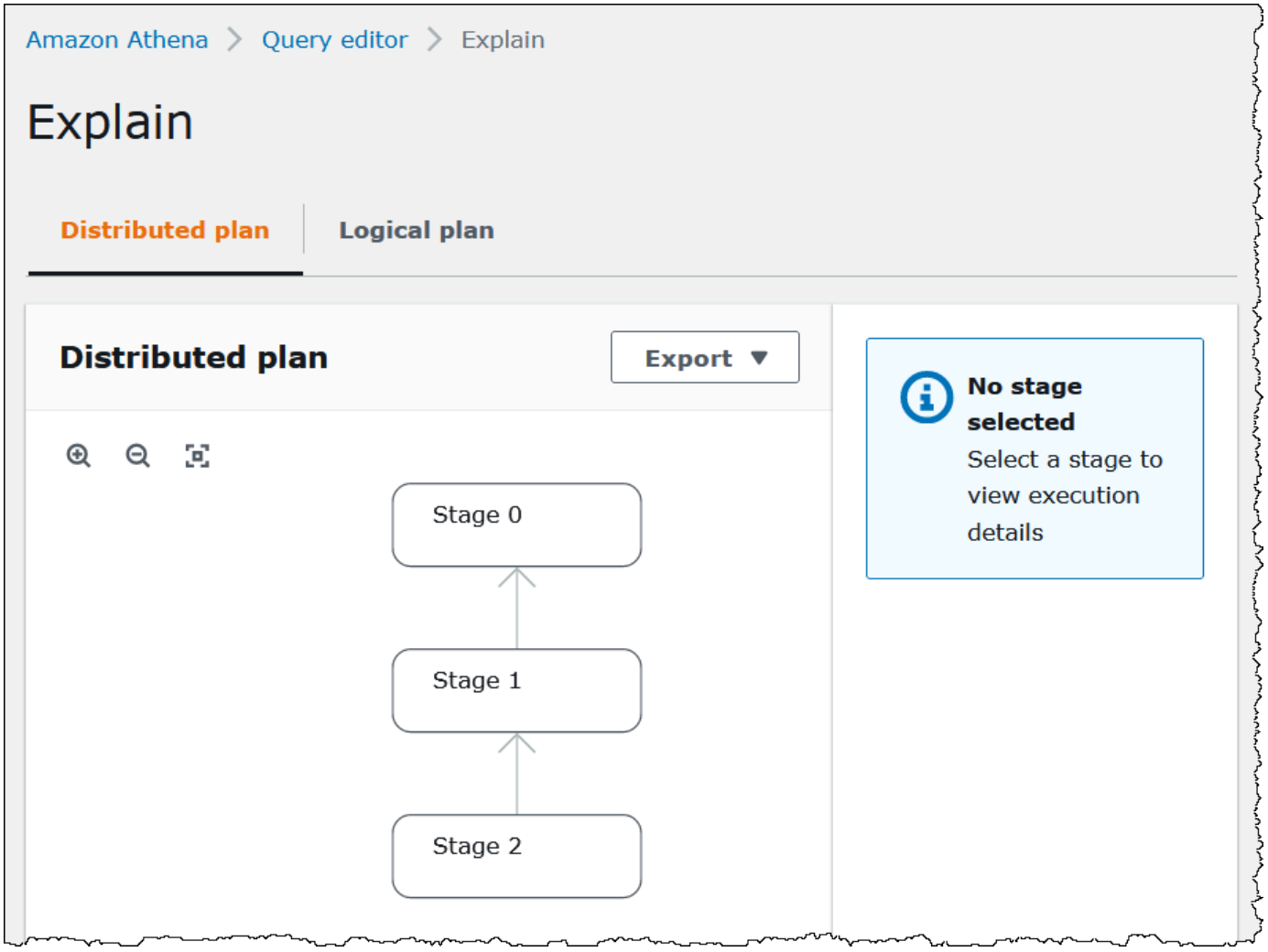
È possibile utilizzare l'editor di query Athena per visualizzare rappresentazioni grafiche di come verrà eseguita la query. Quando si immette una query nell'editor e si seleziona l'opzione Explain, Athena usa un'istruzione EXPLAIN di SQL sulla query per creare due grafici corrispondenti: un piano di esecuzione distribuito e un piano di esecuzione logico. Puoi utilizzare questi grafici per analizzare, risolvere i problemi e migliorare l'efficienza delle tue query.

Visualizzare i piani di esecuzione di una query

1. Inserisci la query nell'editor, quindi seleziona Run (esegui).



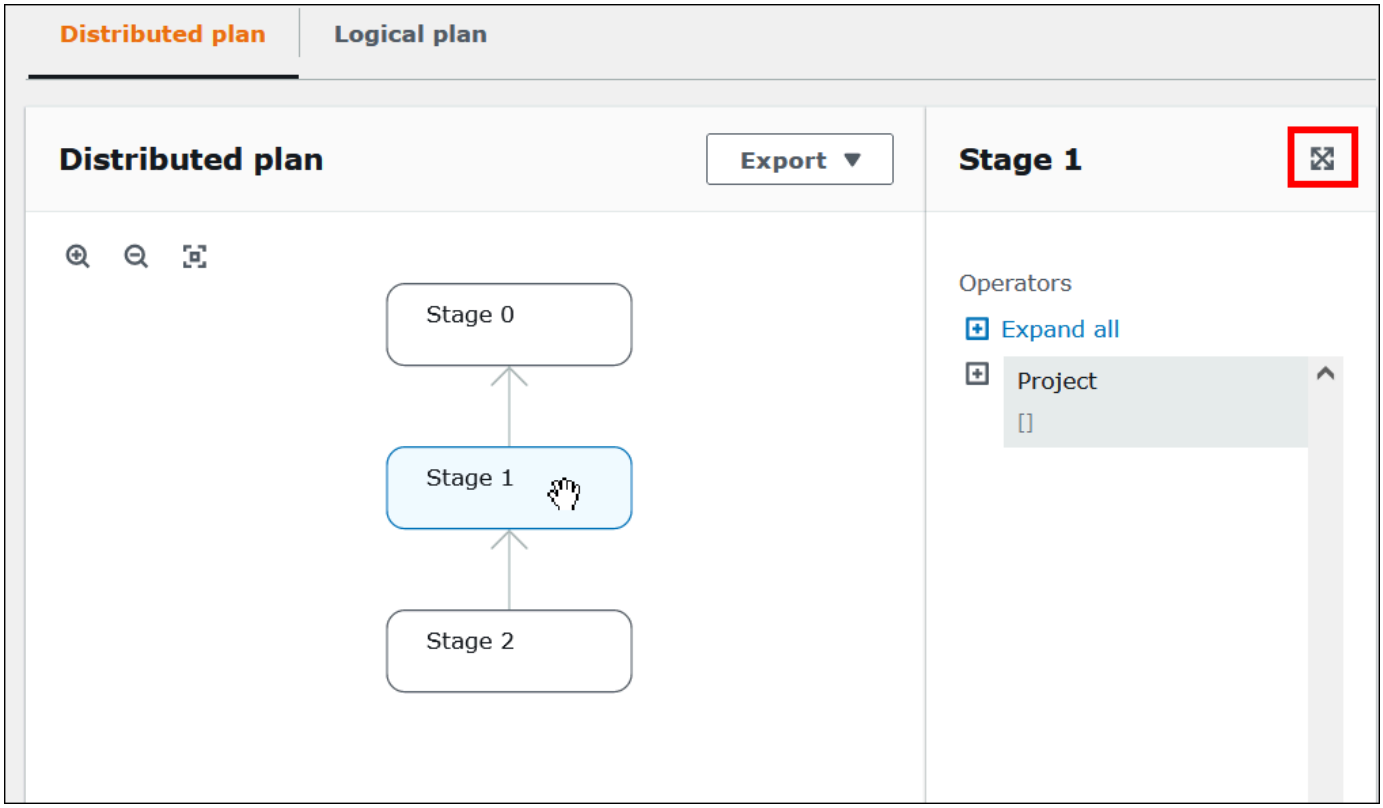
   La scheda Distributed plan (piano distribuito) mostra il piano di esecuzione della query in un ambiente distribuito. Un piano distribuito contiene frammenti di elaborazione o fasi. Ogni fase ha un numero di indice a base zero ed è elaborata da uno o più nodi. I dati possono essere scambiati tra i nodi.



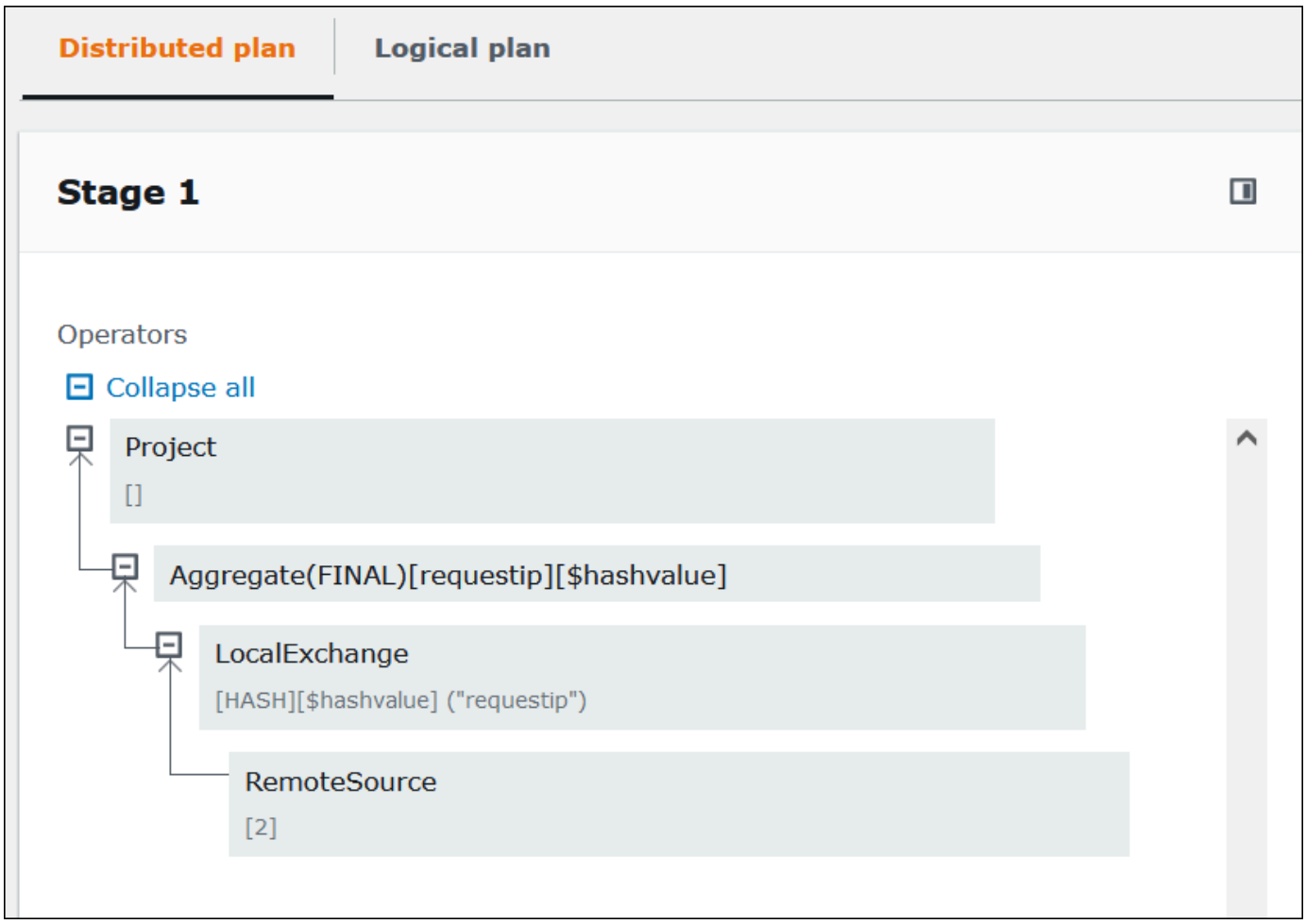
2. Per navigare nel grafico, utilizza le seguenti opzioni:

   - Per ingrandire o ridurre l'immagine, fai scorrere il mouse o utilizza le icone di ingrandimento.
   - Per regolare il grafico in base alle dimensioni della schermata, seleziona l'icona Zoom to fit (usa lo zoom per adattare l'immagine).
   - Per spostare il grafico, trascina il puntatore del mouse.

3. Per visualizzare maggiori dettagli di una fase, selezionala.

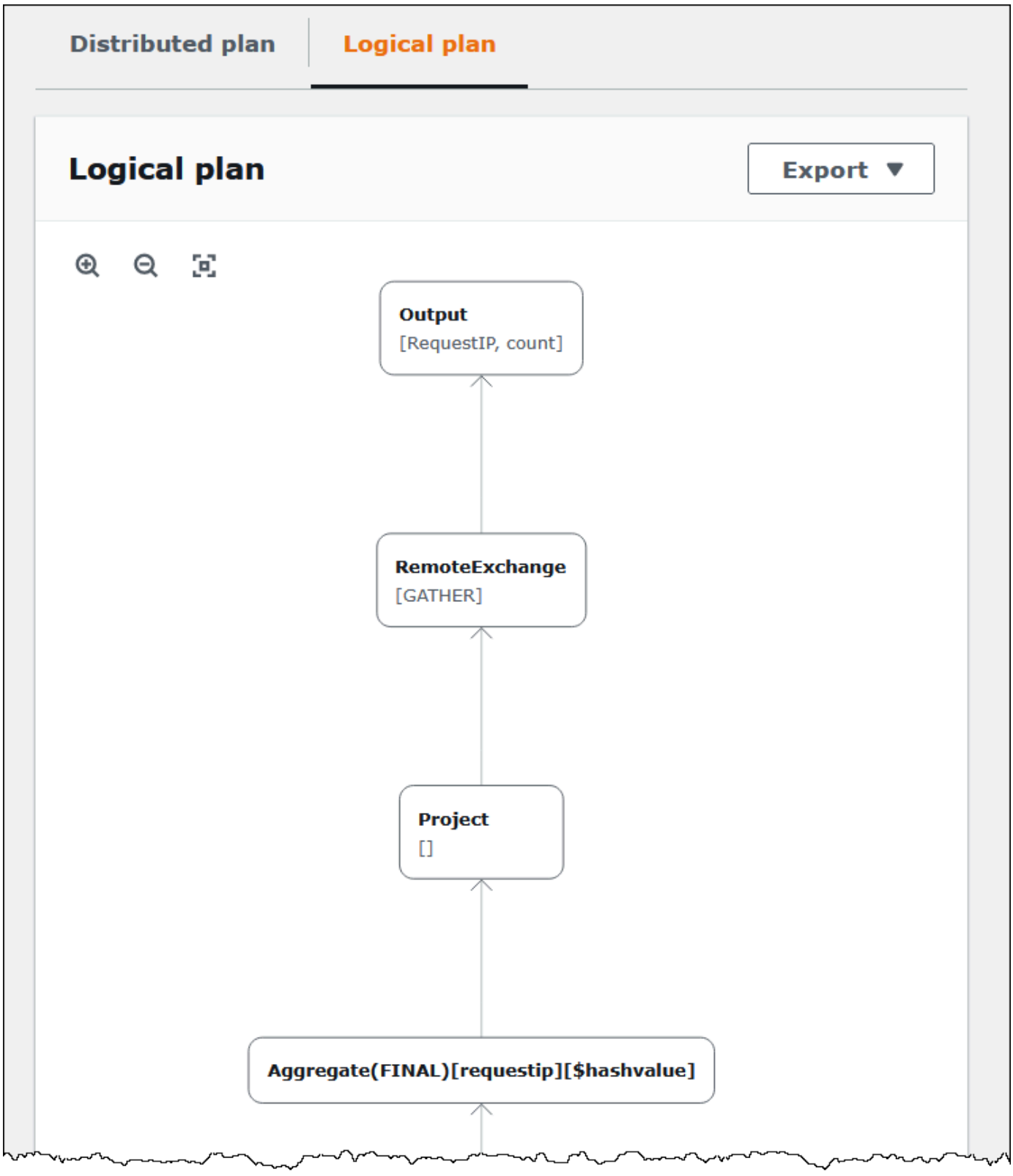


4. Per visualizzare i dettagli della fase a tutto schermo, seleziona l'icona di espansione in alto a destra nel pannello dei dettagli.
5. Per visualizzare più dettagli, espandi uno o più elementi nell'albero operatore. Per informazioni sui frammenti di piani distribuiti, consulta Tipi di output dell'istruzione EXPLAIN.



> **Important**
>
> Attualmente, alcuni filtri di partizione potrebbero non essere visibili nel grafico ad albero degli operatori annidato anche se Athena li applica alla tua query. Per verificare l'effetto di tali filtri, esegui EXPLAIN o EXPLAIN ANALYZE sulla query e visualizza i risultati.

6. Seleziona la scheda Logical plan (piano logico). Il grafico mostra il piano logico per l'esecuzione della query. Per ulteriori informazioni sull'operazione, consulta Capire i risultati dell'istruzione EXPLAIN di Athena.



7. Per esportare un piano come immagine SVG o PNG o come testo JSON, scegli Export (esporta).

Risorse aggiuntive

Per ulteriori informazioni, consulta le risorse seguenti.

Utilizzo di EXPLAIN e EXPLAIN ANALYZE in Athena

Capire i risultati dell'istruzione EXPLAIN di Athena

Visualizzazione di statistiche e dettagli di esecuzione per le query completate

Utilizzo dei risultati delle query, delle query recenti e dei file di output

Amazon Athena archivia automaticamente i risultati delle query e le informazioni sui metadati per ogni query eseguita in una posizione dei risultati delle query che puoi specificare in Amazon S3. Se necessario, puoi accedere ai file in questa posizione per utilizzarli. Puoi anche scaricare i file dei risultati delle query direttamente dalla console Athena.

Per impostare una posizione per i risultati delle query di Amazon S3 per la prima volta, consulta Specificare una posizione dei risultati delle query utilizzando la console Athena.

I file di output vengono salvati automaticamente per ogni query eseguita. Per accedere e visualizzare i file di output delle query utilizzando la console Athena, i responsabili IAM (utenti e ruoli) necessitano dell'autorizzazione all'GetObjectazione Amazon S3 per la posizione dei risultati della query, nonché dell'autorizzazione per l'azione Athena. GetQueryResults La posizione dei risultati delle query può essere crittografata. Se la posizione è crittografata, gli utenti devono disporre delle autorizzazioni chiave appropriate per crittografare e decrittografare la posizione dei risultati delle query.

> **Important**
>
> I principali IAM con l'autorizzazione per l'operazione Amazon S3 `GetObject` per la posizione dei risultati delle query sono in grado di recuperare i risultati delle query da Amazon S3 anche se l'autorizzazione per l'operazione Athena `GetQueryResults` viene negata.

Specificare una posizione dei risultati delle query

La posizione dei risultati delle query utilizzata da Athena è determinata da una combinazione di impostazioni del gruppo di lavoro e impostazioni lato client. Le impostazioni lato client sono basate sulla modalità di esecuzione della query.

- Se esegui la query utilizzando la console Athena, la posizione dei risultati delle query immessa in Impostazioni nella barra di navigazione determina l'impostazione lato client.
- Se esegui la query utilizzando l'API Athena, il `OutputLocation` parametro dell'StartQueryExecutionazione determina l'impostazione lato client.
- Se utilizzi i driver ODBC o JDBC per eseguire le query, la proprietà `S3OutputLocation` specificata nell'URL di connessione determina l'impostazione lato client.

> **Important**
>
> Quando esegui una query utilizzando l'API o il driver ODBC o JDBC, l'impostazione della console non viene applicata.

Ogni configurazione del gruppo di lavoro ha un'opzione Override client-side settings (Ignora impostazioni lato client) che può essere abilitata. Quando questa opzione è abilitata, le impostazioni del gruppo di lavoro hanno la precedenza sulle impostazioni lato client applicabili quando un principale IAM associato a tale gruppo di lavoro esegue la query.

Specificare una posizione dei risultati delle query utilizzando la console Athena

Prima di poter eseguire una query, è necessario specificare una posizione del bucket dei risultati delle query in Amazon S3, o utilizzare un gruppo di lavoro che ha specificato un bucket e la cui configurazione sostituisce le impostazioni del client.

Per specificare un percorso dei risultati della query di impostazione lato client utilizzando la console Athena

1. Passa al gruppo di lavoro per il quale desideri specificare una posizione per i risultati delle query. Il nome del gruppo di lavoro predefinito è primario.
2. Dalla barra di navigazione scegliere Impostazioni.
3. Scegli Manage (Gestisci) sulla barra di navigazione.
4. Per Manage settings (Gestisci impostazioni), effettua una delle seguenti operazioni:
   - Nella casella Location of query result (Posizione dei risultati delle query) inserisci il percorso del bucket creato in Amazon S3 per i risultati delle query. Aggiungi al percorso il prefisso `s3://`.

- Scegli Browse S3 (Sfoglia S3), scegli il bucket Amazon S3 creato per la tua regione corrente, quindi seleziona Choose (Scegli).

> **Note**
>
> Se utilizzi un gruppo di lavoro che specifica una posizione dei risultati delle query per tutti gli utenti del gruppo di lavoro, l'opzione per modificare la posizione dei risultati delle query non è disponibile.

5. (Facoltativo) Scegli View lifecycle configuration (Visualizza configurazione del ciclo di vita) per visualizzare e configurare le regole del ciclo di vita di Amazon S3 nel bucket dei risultati delle query. Le regole del ciclo di vita di Amazon S3 che crei possono essere regole di scadenza o regole di transizione. Le regole di scadenza eliminano automaticamente i risultati delle query dopo un certo periodo di tempo. Le regole di transizione li spostano su un altro livello di archiviazione di Amazon S3. Per ulteriori informazioni, consulta Configurazione del ciclo di vita per un bucket nella Guida per l'utente di Amazon Simple Storage Service.
6. (Facoltativo) In Expected bucket owner, inserisci l'ID del bucket di posizione di Account AWS output che pensi sia il proprietario del bucket di posizione di output. Si tratta di una misura di sicurezza aggiuntiva. Se l'ID account del proprietario del bucket non corrisponde all'ID specificato, i tentativi di output nel bucket avranno esito negativo. Per informazioni dettagliate, consulta Verifica della proprietà del bucket con condizione del proprietario del bucket nella Guida per l'utente di Amazon S3.

> **Note**
>
> L'impostazione prevista per il proprietario del bucket si applica solo al percorso di output di Amazon S3 specificato per i risultati delle query di Athena. Non si applica ad altri percorsi Amazon S3 come i percorsi dell'origine dati nei bucket Amazon S3 esterni, i percorsi relativi alle tabelle di destinazione con istruzioni `CTAS` e `INSERT INTO`, i percorsi di output delle istruzioni `UNLOAD`, le operazioni del bucket spill per le query federate o le query `SELECT` eseguite su una tabella in un altro account.

7. (Facoltativo) Scegli Encrypt query results (Esegui crittografia dei risultati delle query) se desideri crittografare i risultati delle query archiviati in Amazon S3. Per ulteriori informazioni sulla crittografia in Athena, consulta Crittografia a riposo.

8. (Facoltativo) Scegli Assign bucket owner full control over query results (Assegna al proprietario del bucket il controllo di accesso completo ai risultati delle query) per garantire il controllo di accesso completo ai risultati delle query al proprietario del bucket quando le ACL sono abilitate per il bucket dei risultati della query. Ad esempio, se la posizione dei risultati della query è di proprietà di un altro account, puoi concedere la proprietà e il controllo completo dei risultati della query all'altro account. Per ulteriori informazioni, consulta Controlling ownership of objects and disabling ACLs for your bucket (Controllo della proprietà degli oggetti e disabilitazione delle ACL per il bucket) nella Guida per l'utente di Amazon S3.
9. Selezionare Salva.

Posizioni di default create in precedenza

In precedenza in Athena, se si eseguiva una query senza specificare un valore per Query result location (Posizione dei risultati delle query) e l'impostazione della posizione dei risultati della query non è stata sovrascritta da un gruppo di lavoro, Athena creava una posizione predefinita per l'utente. La posizione predefinita eraaws-athena-query-results-*MyAcctID*-*MyRegion*, dove *MyAcctID* era l'ID dell'account Amazon Web Services del principale IAM che eseguiva la query ed *MyRegion*era la regione in cui veniva eseguita la query (ad esempio,us-west-1.)

Ora, prima di poter eseguire una query Athena in una Regione in cui l'account non ha utilizzato in Athena precedenza, è necessario specificare una posizione dei risultati delle query o utilizzare un gruppo di lavoro che sostituisce l'impostazione della posizione dei risultati delle query Sebbene Athena non crei più una posizione predefinita dei risultati delle query, le posizioni predefinite aws-athena-query-results-*MyAcctID*-*MyRegion* create in precedenza rimangono valide ed è possibile continuare a utilizzarle.

Specificare una posizione dei risultati delle query utilizzando un gruppo di lavoro

Puoi specificare la posizione dei risultati delle query in una configurazione del gruppo di lavoro utilizzando AWS CLI, AWS Management Console, o l'API Athena.

Quando utilizzi AWS CLI, specifica la posizione del risultato della query utilizzando il OutputLocation parametro dell'--configurationopzione quando esegui il aws athena update-work-groupcomando aws athena create-work-groupor.

Per specificare la posizione dei risultati delle query per un gruppo di lavoro utilizzando la console Athena

1. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

2. Nel pannello di navigazione, seleziona Workgroups (Gruppi di lavoro).
3. Nell'elenco dei gruppi di lavoro, scegli il collegamento del gruppo di lavoro che desideri configurare.
4. Scegli Modifica.
5. Per la posizione e la crittografia dei risultati delle query) esegui una delle seguenti operazioni:
   - Nella casella Location of query result (Posizione dei risultati delle query) inserisci il percorso del bucket in Amazon S3 per i risultati delle query. Aggiungi al percorso il prefisso `s3://`.
   - Scegli Browse S3 (Sfoglia S3), scegli il bucket Amazon S3 per la tua regione corrente che desideri utilizzare, quindi seleziona Choose (Scegli).
6. (Facoltativo) In Expected bucket owner, inserite l'ID del bucket di posizione di output Account AWS che ritenete sia il proprietario del bucket di posizione di output. Si tratta di una misura di sicurezza aggiuntiva. Se l'ID account del proprietario del bucket non corrisponde all'ID specificato, i tentativi di output nel bucket avranno esito negativo. Per informazioni dettagliate, consulta Verifica della proprietà del bucket con condizione del proprietario del bucket nella Guida per l'utente di Amazon S3.

> **Note**
>
> L'impostazione prevista per il proprietario del bucket si applica solo al percorso di output di Amazon S3 specificato per i risultati delle query di Athena. Non si applica ad altri percorsi Amazon S3 come i percorsi dell'origine dati nei bucket Amazon S3 esterni, i percorsi relativi alle tabelle di destinazione con istruzioni `CTAS` e `INSERT INTO`, i

percorsi di output delle istruzioni `UNLOAD`, le operazioni del bucket spill per le query federate o le query `SELECT` eseguite su una tabella in un altro account.

7. (Facoltativo) Scegli Encrypt query results (Esegui crittografia dei risultati delle query) se desideri crittografare i risultati delle query archiviati in Amazon S3. Per ulteriori informazioni sulla crittografia in Athena, consulta Crittografia a riposo.
8. (Facoltativo) Scegli Assign bucket owner full control over query results (Assegna al proprietario del bucket il controllo di accesso completo ai risultati delle query) per garantire il controllo di accesso completo ai risultati delle query al proprietario del bucket quando le ACL sono abilitate per il bucket dei risultati della query. Ad esempio, se la posizione dei risultati della query è di proprietà di un altro account, puoi concedere la proprietà e il controllo completo dei risultati della query all'altro account.

   Se l'impostazione di S3 Object Ownership del bucket è Bucket owner preferred (Preferita dal proprietario del bucket), il proprietario del bucket possiede anche tutti gli oggetti dei risultati della query scritti da questo gruppo di lavoro. Ad esempio, se il gruppo di lavoro di un account esterno abilita questa opzione e imposta la posizione dei risultati della query sul bucket Amazon S3 del tuo account con S3 Object Ownership impostato su Bucket owner preferred (Preferita dal proprietario del bucket), possiedi e hai il pieno controllo di accesso ai risultati delle query del gruppo di lavoro esterno.

   Selezionando questa opzione, quando l'impostazione di S3 Object Ownership del bucket dei risultati della query è Bucket owner enforced (Applicata dal proprietario del bucket), non si avrà alcun effetto. Per ulteriori informazioni, consulta Controlling ownership of objects and disabling ACLs for your bucket (Controllo della proprietà degli oggetti e disabilitazione delle ACL per il bucket) nella Guida per l'utente di Amazon S3.
9. Se desideri richiedere a tutti gli utenti del gruppo di lavoro di utilizzare la posizione dei risultati della query specificata, scorri verso il basso fino alla sezione Settings (Impostazioni) e seleziona Override client-side settings (Ignora impostazioni lato client).
10. Seleziona Salvataggio delle modifiche.

Download dei file dei risultati delle query mediante la console Athena

Puoi scaricare il file CSV dei risultati delle query dal riquadro della query subito dopo averla eseguita. È possibile anche scaricare i risultati delle query dalle query recenti nella scheda Recent queries (Query recenti).

> **Note**
>
> I file dei risultati delle query Athena sono file di dati che contengono informazioni che possono essere configurate dai singoli utenti. Alcuni programmi che leggono e analizzano questi dati possono interpretare potenzialmente alcuni dei dati come comandi (CSV Injection). Per questo motivo, quando importi i risultati delle query dei dati CSV in un programma di calcolo, il programma potrebbe mostrare avvisi su problemi di sicurezza. Per mantenere il sistema sicuro, è consigliabile scegliere sempre di disabilitare i link o le macro dai risultati delle query scaricati.

Come eseguire una query e scaricare i risultati delle query

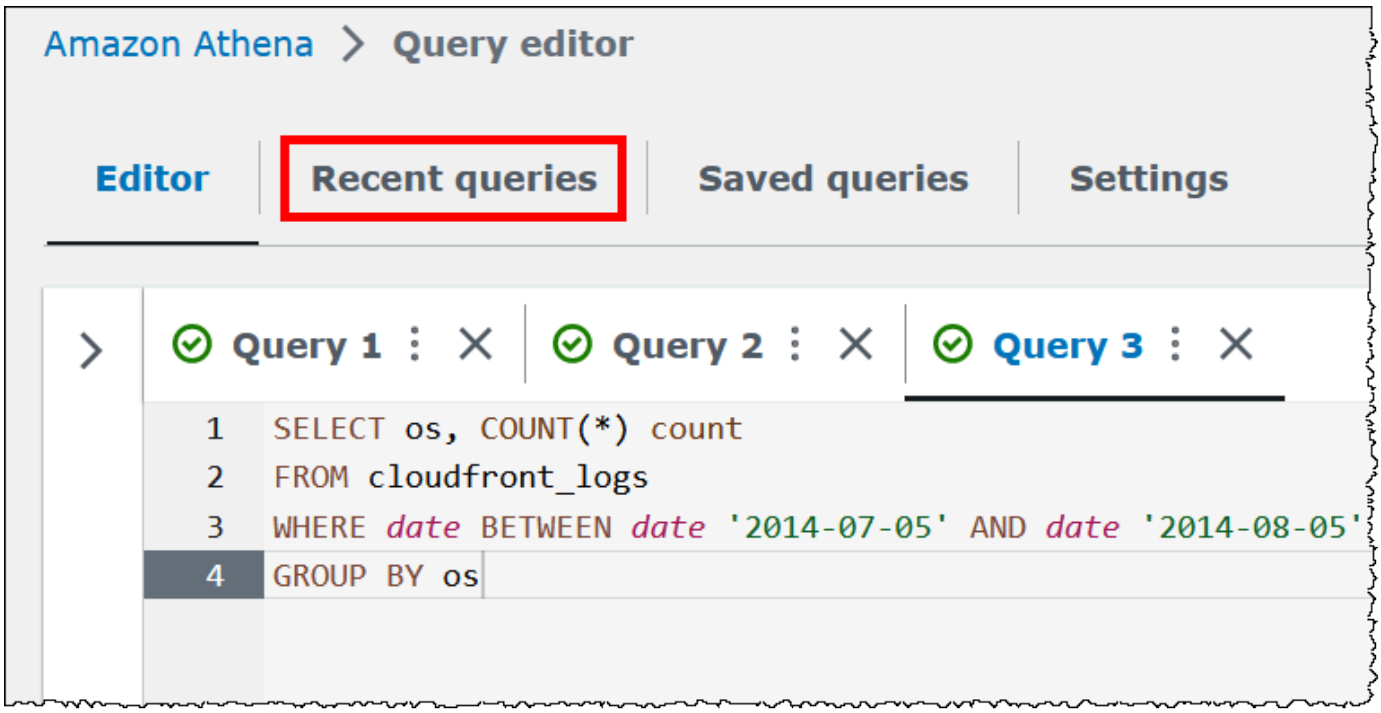
1. Inserisci la query nell'editor di query, quindi scegli Run (Esegui).

   Al termine dell'esecuzione della query, il riquadro Results (Risultati) mostra i risultati della query.

2. Per scaricare un file CSV dei risultati delle query, scegli Download results (Scarica risultati) sopra il riquadro dei risultati della query. A seconda del browser e della sua configurazione, potrebbe essere necessario confermare il download.

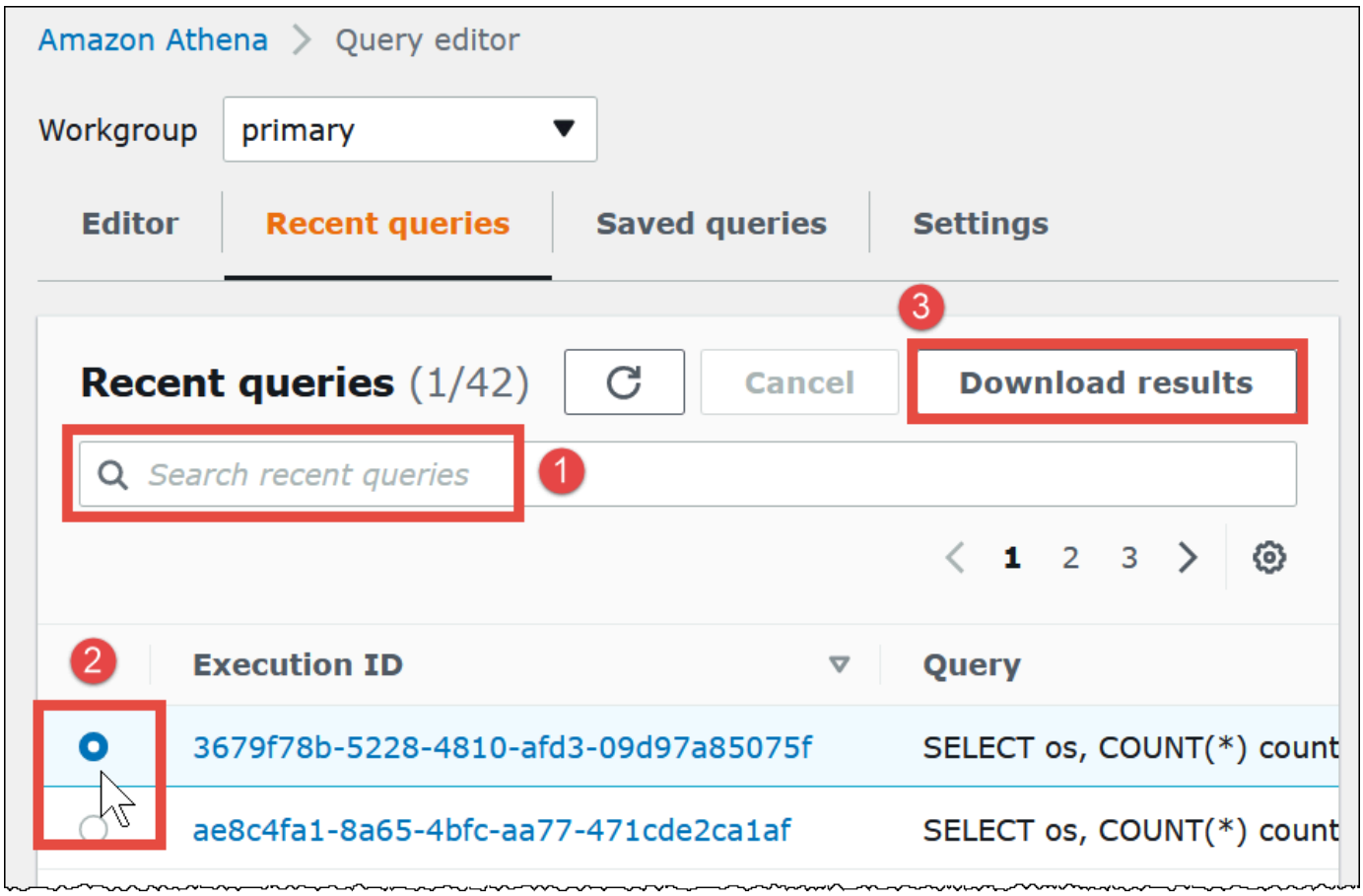
Per scaricare il file dei risultati di una query precedente

1. Scegli Recent queries (Query recenti).



2. Utilizza la casella di ricerca per trovare la query, selezionala e scegli quindi Download results (Scarica risultati).

> **Note**
>
> Non è possibile utilizzare l'opzione Download results (Scarica risultati) per recuperare i risultati delle query che sono stati eliminati manualmente o recuperare i risultati delle query che sono stati eliminati o spostati in un'altra posizione in base alle regole del ciclo di vita di Amazon S3.

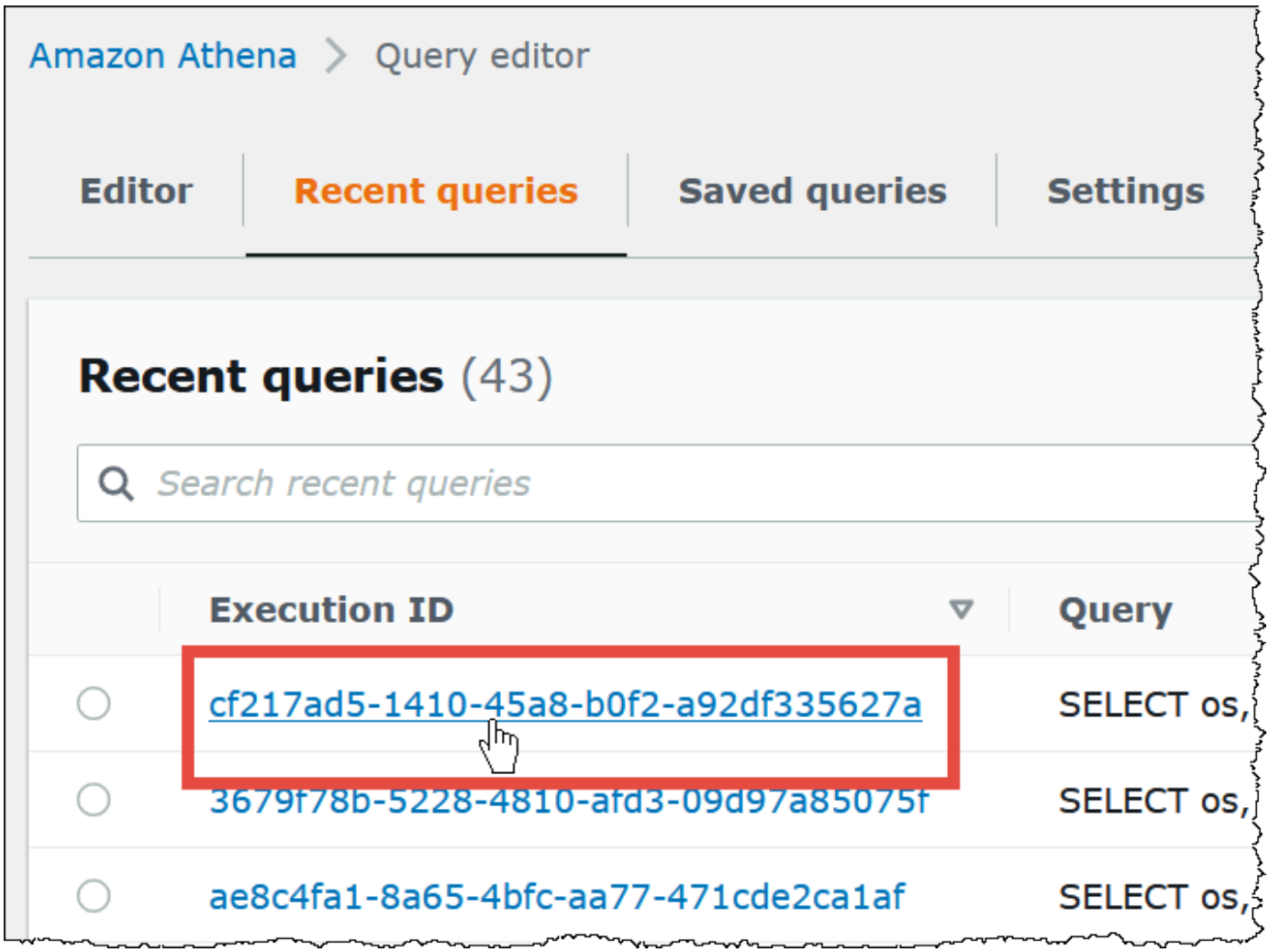


Visualizzazione delle query recenti

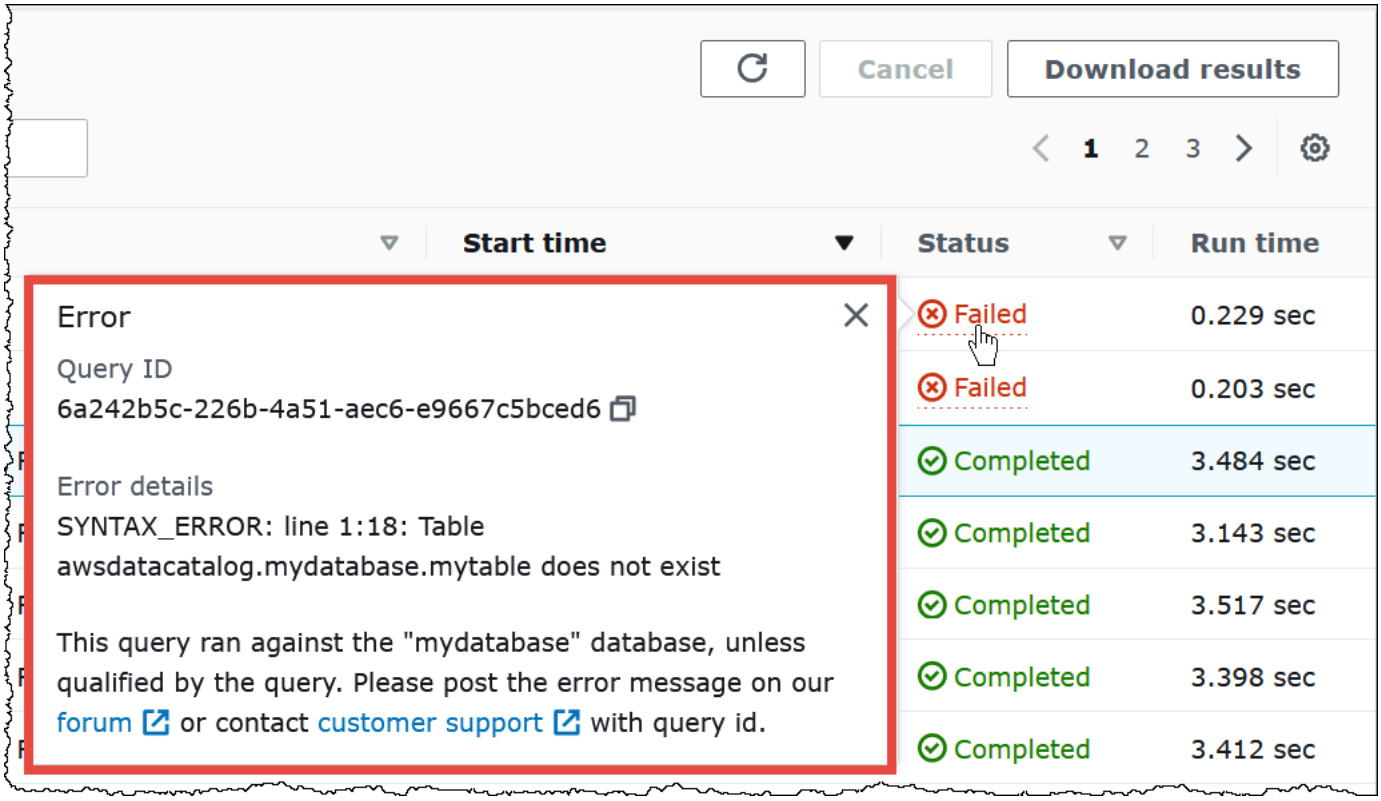
È possibile utilizzare la console Athena per vedere quali query sono riuscite o non riuscite e visualizzare i dettagli di errore per quelle non riuscite. Athena mantiene la cronologia delle query per 45 giorni.

Come visualizzare le query recenti nella console Athena

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Scegli Recent queries (Query recenti). La scheda Recent queries (Query recenti) mostra informazioni su ogni query eseguita.
3. Per aprire un'istruzione di query nell'editor di query, scegli l'ID di esecuzione della query.



4. Per visualizzare i dettagli di una query non riuscita, scegli il collegamento Failed (Non riuscito) per la query.

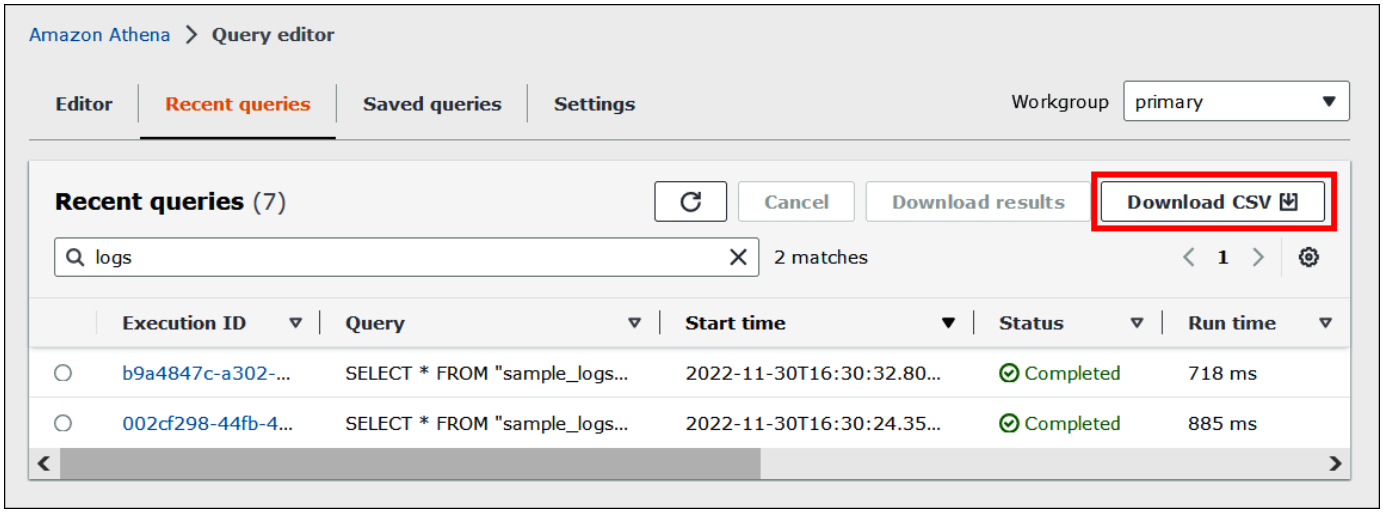


Scaricamento di più query recenti in un file CSV

Puoi utilizzare la scheda Recent queries (Query recenti) della console Athena per esportare una o più query recenti in un file CSV per visualizzarle in formato tabellare. Il file scaricato non contiene i risultati della query, ma la stringa della query SQL stessa e altre informazioni sulla query. I campi esportati includono l'ID di esecuzione, il contenuto della stringa di query, l'ora di inizio della query, lo stato, il tempo di esecuzione, la quantità di dati scansionati, la versione del motore di query utilizzato e il metodo di crittografia. Puoi esportare un massimo di 500 query recenti o un massimo di 500 query filtrate utilizzando i criteri immessi nella casella di ricerca.

Esportazione di una o più query recenti in un file CSV

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Scegli Recent queries (Query recenti).
3. (Facoltativo) Utilizza la casella di ricerca per filtrare le query recenti che desideri scaricare.
4. Scegli Download CSV (Scarica CSV).



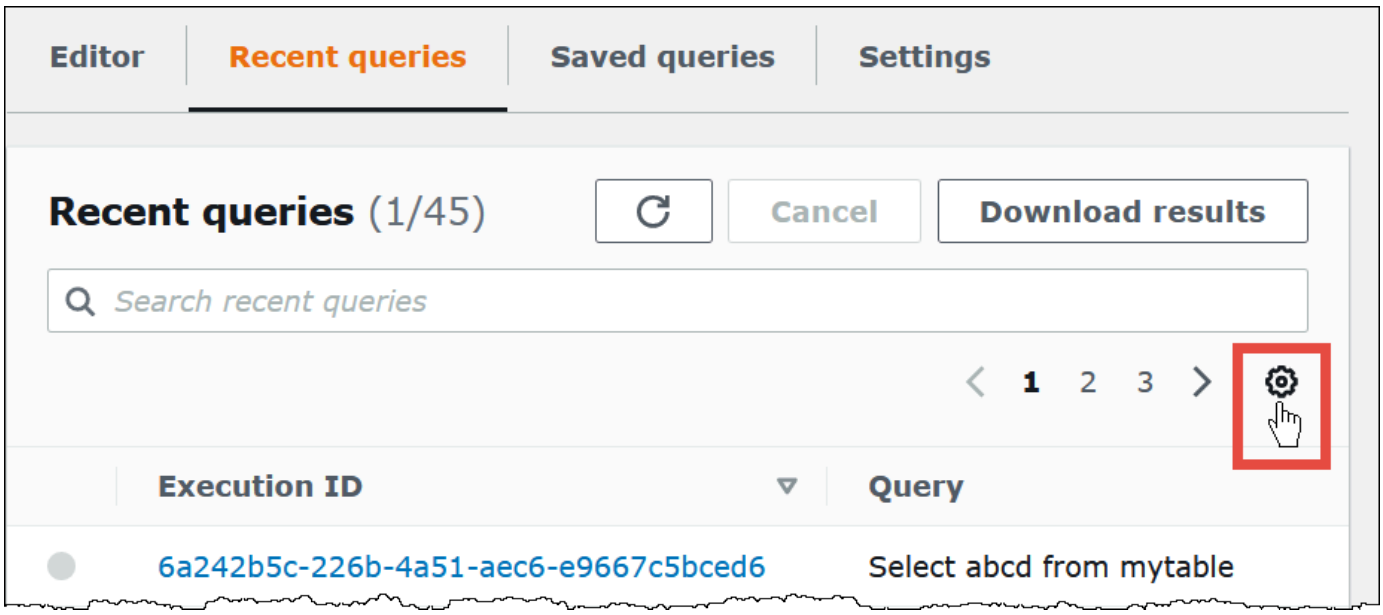
5. Alla richiesta di salvataggio del file, scegli Save (Salva). Il nome del file predefinito è `Recent Queries` seguito da un timestamp (ad esempio, `Recent Queries 2022-12-05T16 04 27.352-08 00.csv`)

Configurazione delle opzioni di visualizzazione delle query recenti

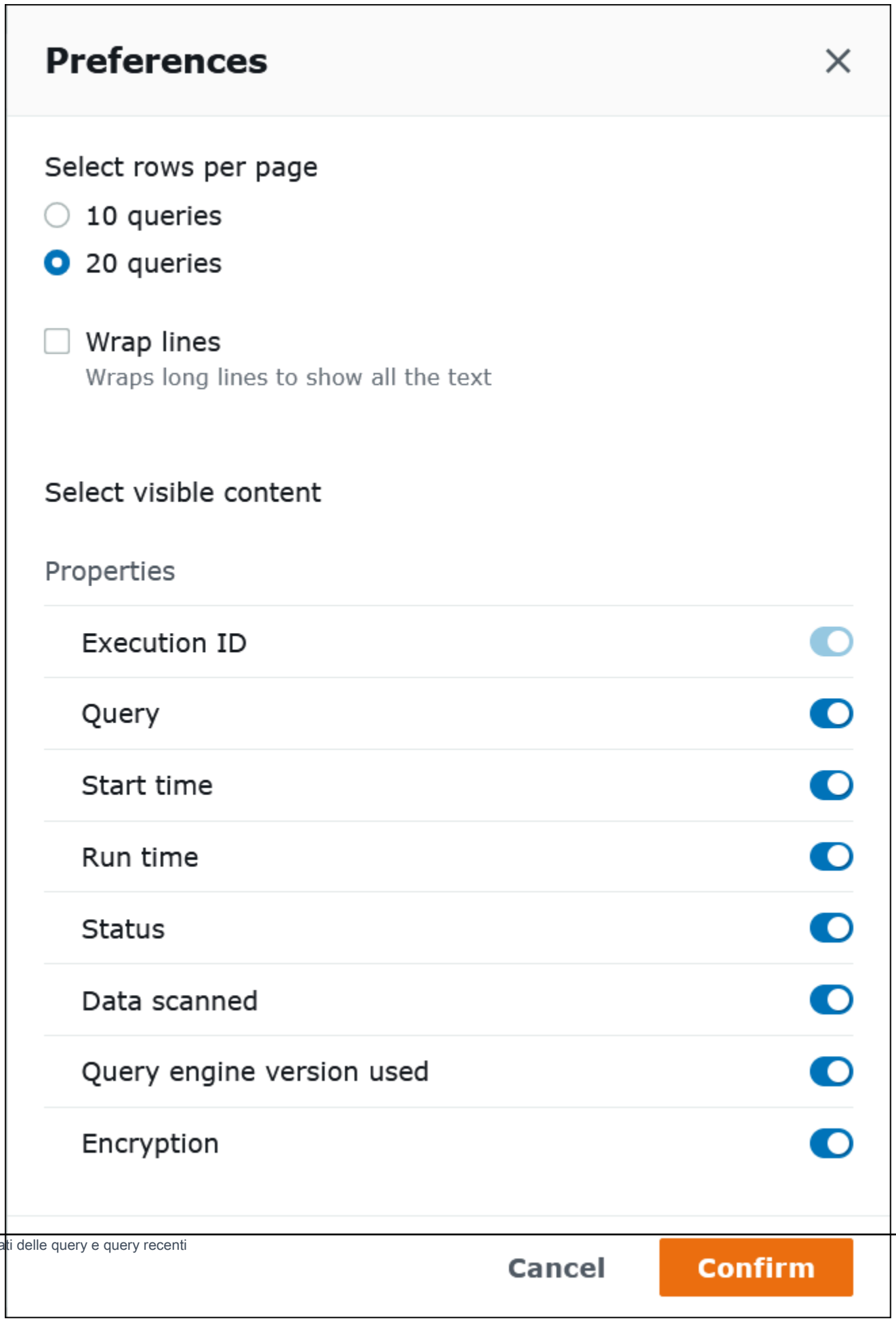
Puoi configurare le opzioni per la scheda Recent queries (Query recenti) come colonne da visualizzare con testo a capo.

Configurazione delle opzioni per la scheda Recent queries (Query recenti)

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Scegli Recent queries (Query recenti).
3. Scegli il pulsante delle opzioni (icona a forma di ingranaggio).



4. Nella finestra di dialogo Preferences (Preferenze), scegli il numero di righe per pagina, il comportamento del ritorno a capo e le colonne da visualizzare.


Preferences
Select rows per page
10 queries
20 queries
Wrap lines
Wraps long lines to show all the text
Select visible content
Properties
Execution ID
Query
Start time
Run time
Status
Data scanned
Query engine version used
Encryption
Cancel
Confirm

5. Scegli Conferma.

Conservazione della cronologia delle query per più di 45 giorni

Se si desidera mantenere la cronologia delle query più di 45 giorni, è possibile recuperare la cronologia delle query e salvarla in un archivio dati, ad esempio Amazon S3. Per automatizzare questo processo, è possibile utilizzare le operazioni dell'API Amazon S3 e Athena e i comandi CLI. Nella procedura seguente vengono riepilogati questi passaggi.

Per recuperare e salvare la cronologia delle query a livello di codice

1. Utilizza l'azione ListQueryExecutionsAPI Athena o il comando list-query-executionsCLI per recuperare gli ID delle query.
2. Utilizza l'azione GetQueryExecutionAPI Athena o il comando get-query-executionCLI per recuperare informazioni su ogni query in base al relativo ID.
3. Utilizza l'azione dell'PutObjectAPI Amazon S3 o il comando CLI put-object per salvare le informazioni in Amazon S3.

Ricerca dei file di output delle query in Amazon S3

I file di output delle query vengono archiviati in sottocartelle in Amazon S3 nel seguente schema di percorso, a meno che la query non si verifichi in un gruppo di lavoro la cui configurazione sostituisce le impostazioni lato client. Quando la configurazione del gruppo di lavoro sostituisce le impostazioni lato client, la query utilizza il percorso dei risultati specificato dal gruppo di lavoro.

```
QueryResultsLocationInS3/[QueryName|Unsaved/yyyy/mm/dd/]
```

- *QueryResultsLocationInS3* è la posizione dei risultati della query specificata dalle impostazioni del gruppo di lavoro o dalle impostazioni lato client. Per ulteriori informazioni, consultare the section called “Specificare una posizione dei risultati delle query” riportata di seguito in questo documento.
- Le seguenti sottocartelle vengono create solo per le query eseguite dalla console il cui percorso dei risultati non è stato sostituito dalla configurazione del gruppo di lavoro. *Le query eseguite da AWS CLI o utilizzando l'API Athena vengono salvate direttamente in S3. QueryResultsLocationIn*
  - *QueryName*è il nome della query per la quale vengono salvati i risultati. Se la query è stata eseguita ma non è stata salvata, viene utilizzato `Unsaved`.

- *yyy/mm/dd* è la data di esecuzione della query.

I file associati a una query `CREATE TABLE AS SELECT` vengono archiviati in una sottocartella `tables` del modello precedente.

Identificazione dei file di output delle query

I file vengono salvati nella posizione dei risultati delle query in Amazon S3 in base al nome, all'ID e alla data di esecuzione della query. I file per ogni query vengono denominati utilizzando il *QueryID*, ovvero un identificatore univoco che Athena assegna a ogni query quando viene eseguita.

Vengono salvati i seguenti tipi di file:

Tipo di file	Modelll di denominazione dei file	Descrizione
File dei risultati delle query	*QueryID*.csv *QueryID*.txt	I file dei risultati delle query DML vengono salvati in formato CSV (valori separati da virgola). I risultati delle query DDL vengono salvati come file di testo normale. Puoi scaricare i file dei risultati dalla console dal riquadro Risultati quando utilizzi la console o dalla Cronologia della query. Per ulteriori informazioni, consulta Download dei file dei risultati delle query mediante la console Athena.
File di metadati delle query	*QueryID*.csv.metadata *QueryID*.txt.metadata	I file di metadati delle query DML e DDL vengono salvati in formato binario e non sono

Tipo di file	Modelll di denominazione dei file	Descrizione
		leggibili dall'uomo. L'estensi one del file corrisponde al file dei risultati della query. Athena utilizza i metadati durante la lettura dei risultati delle query utilizzando l'operazione `GetQueryResults` . Anche se questi file possono essere eliminati, è sconsigliato perché informazioni importanti sulla query andrebbero perse.
File manifest di dati	*`QueryID`*`-manifest.csv`	I file manifest di dati vengono generati per tenere traccia dei file Athena creati da nelle posizioni dell'origine dati Amazon S3 quando viene eseguita una query INSERT INTO. Se una query ha esito negativo, il manifest tiene traccia anche dei file che la query intendeva scrivere. Il manifest è utile per identific are i file orfani risultanti da una query non riuscita.

Viene utilizzato AWS CLI per identificare la posizione e i file di output delle interrogazioni

Per utilizzare il AWS CLI per identificare la posizione di output della query e i file dei risultati, eseguite il `aws athena get-query-execution` comando, come illustrato nell'esempio seguente. Sostituisci *`abc1234d-5efg-67hi-jklm-89n0op12qr34`* con l'ID query.

```
aws athena get-query-execution --query-execution-id abc1234d-5efg-67hi-
jklm-89n0op12qr34
```

Il comando restituisce un output simile al seguente: Per le descrizioni di ogni parametro di output, vedete get-query-executionnel AWS CLI Command Reference.

```
{
    "QueryExecution": {
        "Status": {
            "SubmissionDateTime": 1565649050.175,
            "State": "SUCCEEDED",
            "CompletionDateTime": 1565649056.6229999
        },
        "Statistics": {
            "DataScannedInBytes": 5944497,
            "DataManifestLocation": "s3://DOC-EXAMPLE-BUCKET/athena-query-
results-123456789012-us-west-1/MyInsertQuery/2019/08/12/abc1234d-5efg-67hi-
jklm-89n0op12qr34-manifest.csv",
            "EngineExecutionTimeInMillis": 5209
        },
        "ResultConfiguration": {
            "EncryptionConfiguration": {
                "EncryptionOption": "SSE_S3"
            },
            "OutputLocation": "s3://DOC-EXAMPLE-BUCKET/athena-query-
results-123456789012-us-west-1/MyInsertQuery/2019/08/12/abc1234d-5efg-67hi-
jklm-89n0op12qr34"
        },
        "QueryExecutionId": "abc1234d-5efg-67hi-jklm-89n0op12qr34",
        "QueryExecutionContext": {},
        "Query": "INSERT INTO mydb.elb_log_backup SELECT * FROM mydb.elb_logs LIMIT
 100",
        "StatementType": "DML",
        "WorkGroup": "primary"
    }
}
```

Riutilizzo dei risultati delle query

Quando esegui nuovamente una query in Athena, puoi scegliere facoltativamente di riutilizzare l'ultimo risultato della query memorizzato. Questa opzione può aumentare le prestazioni e ridurre i costi in termini di numero di byte scansionati. Il riutilizzo dei risultati delle query è utile se, ad esempio, si sa che i risultati non cambieranno entro un determinato periodo di tempo. Puoi specificare un'età massima per il riutilizzo dei risultati delle query. Athena utilizza il risultato

memorizzato purché non sia più vecchio dell'età specificata. Per ulteriori informazioni, consulta Ridurre i costi e migliorare le prestazioni delle query con Amazon Athena nel blog sui AWS Big Data.

> **Note**
>
> La funzionalità di riutilizzo dei risultati delle query richiede la versione 3 del motore Athena. Per ulteriori informazioni sulla modifica delle versioni del motore, consulta Modifica delle versioni del motore Athena.

Funzionalità principali

- Il riutilizzo dei risultati delle query è una funzionalità opzionale per ogni singola query. Puoi abilitare il riutilizzo dei risultati delle query per ogni singola query.
- L'età massima per riutilizzare i risultati della query può essere specificata in minuti, ore o giorni. L'età massima specificabile è l'equivalente di 7 giorni, indipendentemente dall'unità di tempo utilizzata. Il valore predefinito è 60 minuti.
- Quando abiliti il riutilizzo dei risultati per una query, Athena cerca un'esecuzione precedente della query all'interno dello stesso gruppo di lavoro. Se Athena trova i risultati della query memorizzati corrispondenti, non esegue nuovamente la query, ma rimanda alla posizione del risultato precedente o recupera i dati da essa.

- Per ogni query per la quale è attivata l'opzione di riutilizzo dei risultati, Athena riutilizza l'ultimo risultato della query salvato nella cartella del gruppo di lavoro solo quando tutte le seguenti condizioni sono vere:
  - La stringa di query corrisponde esattamente.
  - Il nome del database e il nome del catalogo corrispondono.
  - Il risultato precedente non è più vecchio dell'età massima specificata o non supera i 60 minuti se non è stata specificata un'età massima.
  - Athena riutilizza solo un'esecuzione con la stessa configurazione dei risultati dell'esecuzione corrente.
  - Hai accesso a tutte le tabelle a cui si fa riferimento nella query.
  - Hai accesso alla posizione del file S3 in cui è archiviato il risultato precedente.

Se una di queste condizioni non è soddisfatta, Athena esegue la query senza utilizzare i risultati memorizzati nella cache.

Considerazioni e limitazioni

Quando utilizzi la funzionalità di riutilizzo dei risultati della query, tieni presenti i punti seguenti:

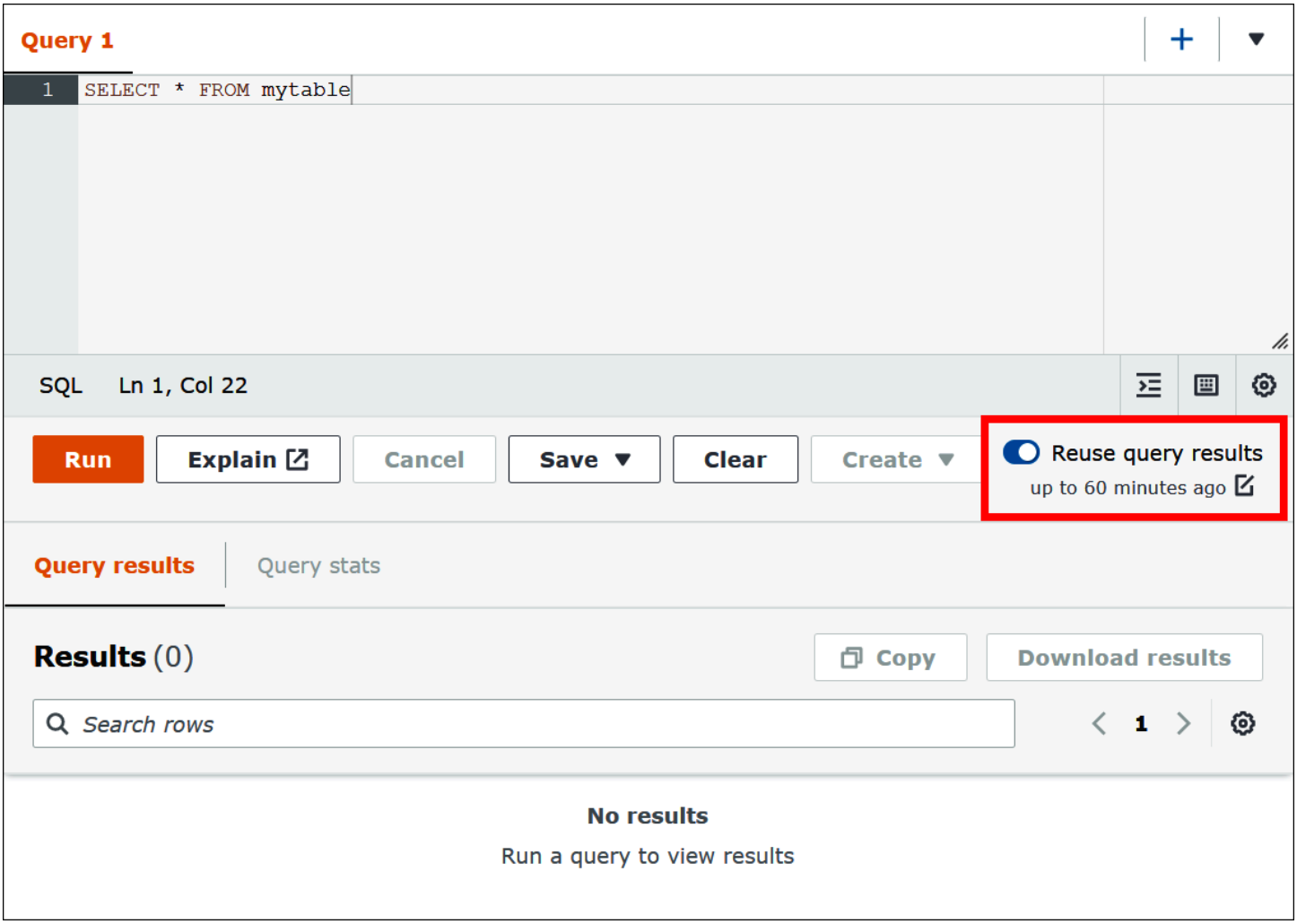
- Athena riutilizza i risultati delle query solo all'interno dello stesso gruppo di lavoro.
- La funzionalità di riutilizzo dei risultati delle query rispetta le configurazioni dei gruppi di lavoro. Se sovrascrivi la configurazione dei risultati per una query, la funzionalità viene disabilitata.
- Sono supportate le tabelle Apache Hive, Apache Hudi, Apache Iceberg e Linux Foundation Delta Lake registrate con. AWS Glue I metastore Hive esterni non sono supportati.
- Le query che fanno riferimento a cataloghi federati o a un metastore Hive esterno non sono supportate.
- Il riutilizzo dei risultati delle query non è supportato per le tabelle regolate da Lake Formation.
- Il riutilizzo dei risultati delle query non è supportato quando la posizione Amazon S3 dell'origine della tabella è registrata come posizione dati in Lake Formation.
- Le tabelle con autorizzazioni per righe e colonne non sono supportate.
- Le tabelle con un controllo degli accessi granulare (ad esempio, il filtraggio di colonne o righe) non sono supportate.
- Qualsiasi query che fa riferimento a una tabella non supportata non è idonea per il riutilizzo dei risultati della query.
- Athena richiede che tu disponga delle autorizzazioni di lettura di Amazon S3 per riutilizzare il file di output generato in precedenza.
- La funzionalità di riutilizzo dei risultati delle query presuppone che il contenuto del risultato precedente non sia stato modificato. Athena non verifica l'integrità di un risultato precedente prima di utilizzarlo.
- Se i risultati della query eseguita precedente sono stati eliminati o spostati in una posizione diversa in Amazon S3, l'esecuzione successiva della stessa query non riutilizzerà i risultati della query.
- È possibile che vengano restituiti risultati potenzialmente obsoleti. Athena non verifica le modifiche nei dati di origine fino al raggiungimento dell'età massima di riutilizzo specificata.
- Se sono disponibili più risultati riutilizzabili, Athena utilizza il risultato più recente.
- Query che utilizzano operatori o funzioni non deterministici, come `rand()` o `shuffle()`, non utilizzano risultati memorizzati nella cache. Ad esempio, `LIMIT` senza `ORDER BY` è non

deterministico e non viene memorizzato nella cache, ma `LIMIT` con `ORDER BY` è deterministico e viene memorizzato nella cache.

- Il riutilizzo dei risultati delle query è supportato nella console Athena, nell'API Athena e nel driver JDBC. Attualmente, il supporto dei driver ODBC per il riutilizzo dei risultati delle query è disponibile solo per Windows.
- Per utilizzare la funzione di riutilizzo dei risultati delle query con JDBC, la versione minima richiesta del driver è 2.0.34.1000. Per ODBC, la versione minima del driver richiesta è 1.1.19.1002. Per informazioni sul download dei driver, consulta Connessione ad Amazon Athena con i driver ODBC e JDBC.
- Il riutilizzo dei risultati delle query non è supportato per le query che utilizzano più di un catalogo di dati.
- Il riutilizzo dei risultati delle query non è supportato per le query che includono più di 20 tabelle.

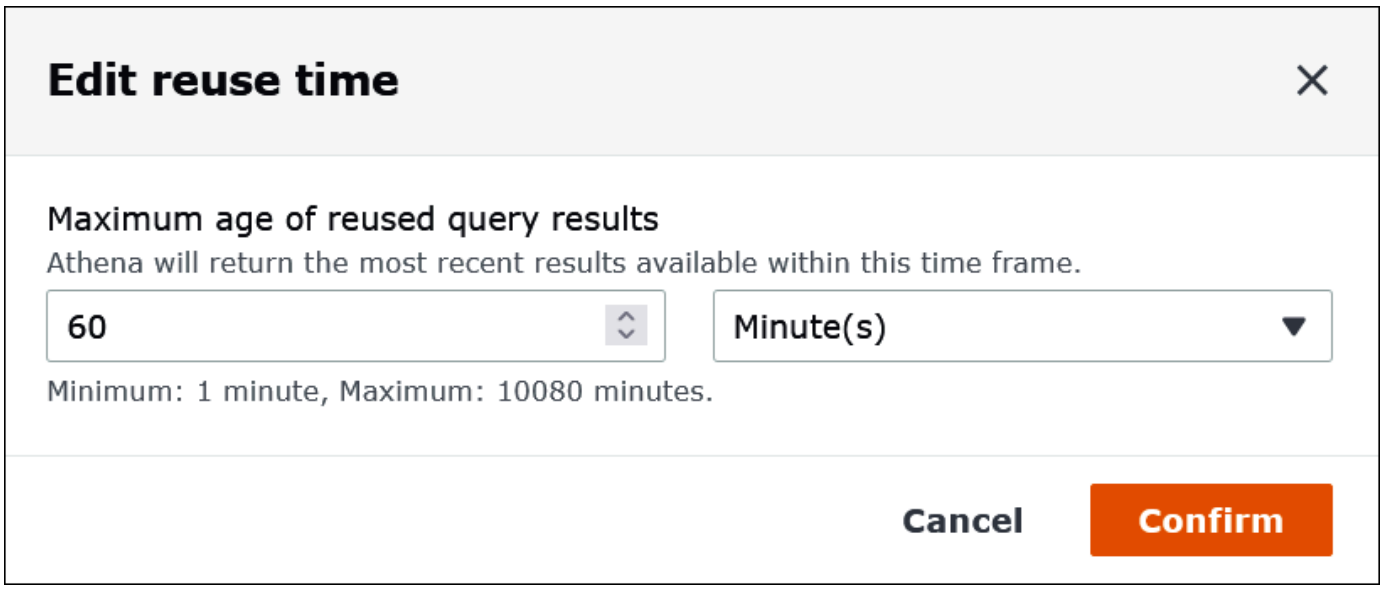
Riutilizzo dei risultati delle query nella console Athena

Per utilizzare la funzionalità, abilita l'opzione Reuse query results (Riutilizza i risultati della query) nell'editor di query di Athena.

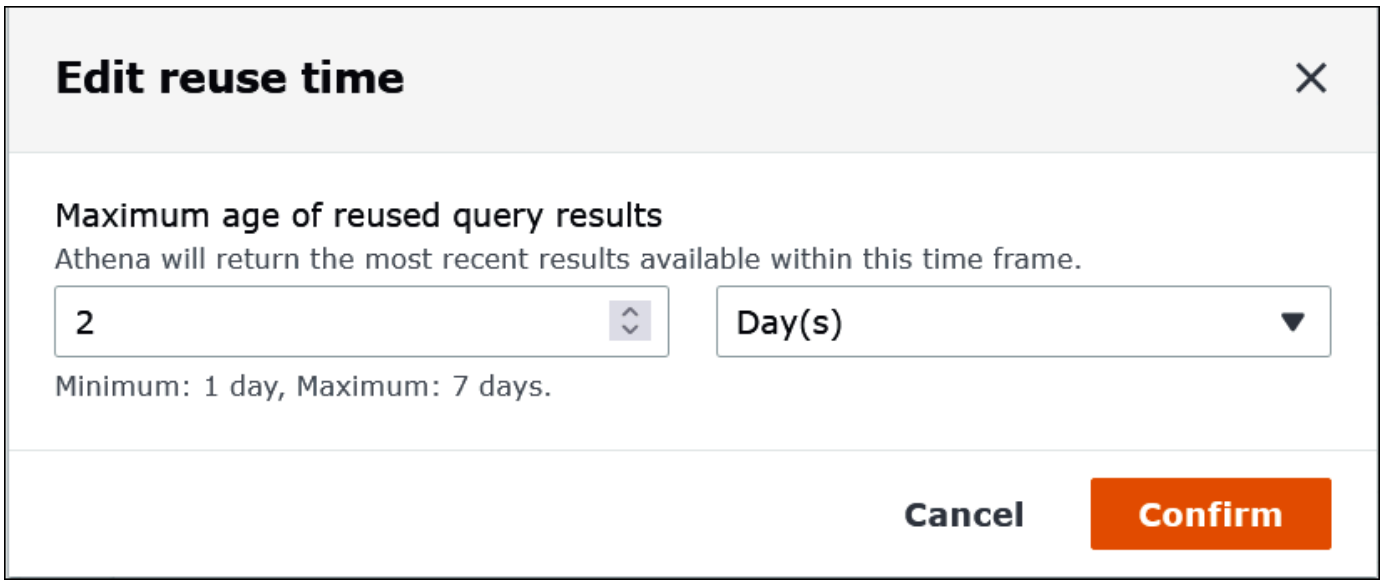


Configurazione della funzionalità di riutilizzo dei risultati delle query

1. Nell'editor di query di Athena, sotto l'opzione Reuse query results (Riutilizza i risultati della query), scegli l'icona di modifica accanto a Up to 60 minutes ago (Massimo 60 minuti fa).
2. Nella finestra di dialogo Edit reuse time (Modifica tempo di riutilizzo), nella casella a destra scegli un'unità di tempo (minuti, ore o giorni).
3. Nella casella a sinistra, inserisci o scegli il numero di unità di tempo che desideri specificare. Il tempo massimo che puoi inserire è l'equivalente di sette giorni, indipendentemente dall'unità di tempo scelta.

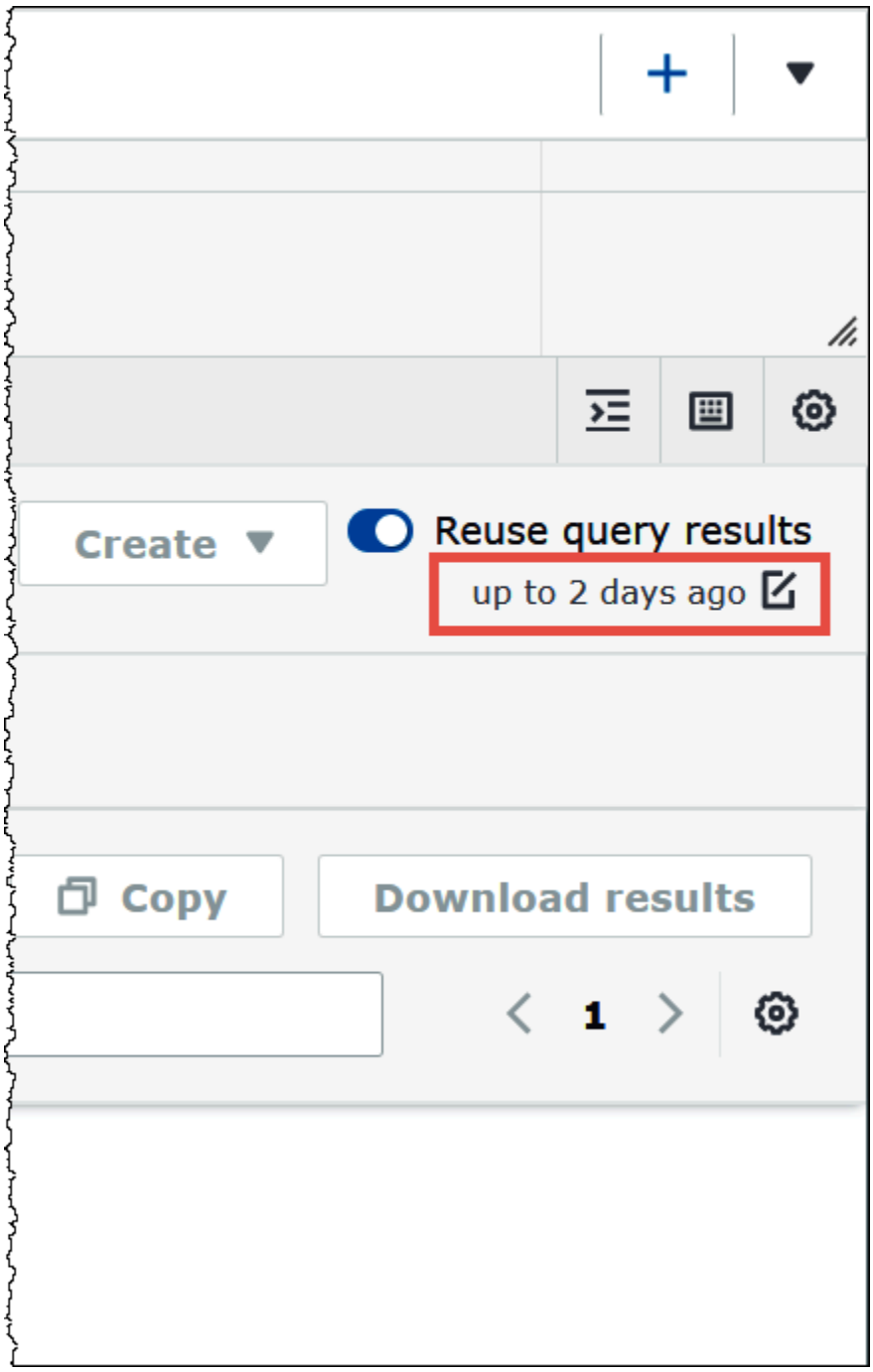


L'esempio seguente specifica un tempo di riutilizzo massimo di due giorni.



4. Scegli Conferma.

   Un banner conferma la modifica alla configurazione e l'opzione Reuse query results (Riutilizza i risultati della query) mostra la nuova impostazione.

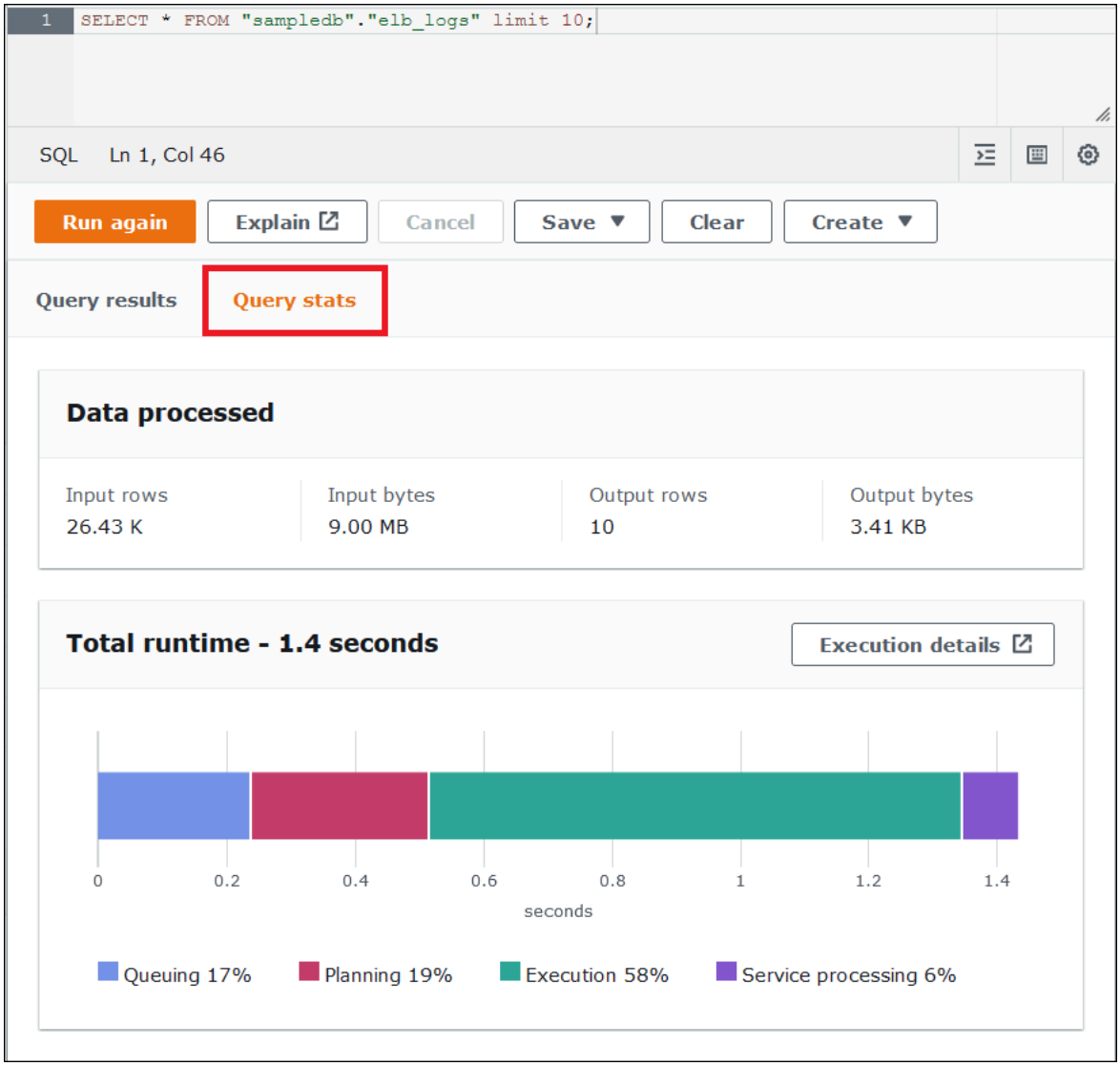


Visualizzazione di statistiche e dettagli di esecuzione per le query completate

Dopo aver eseguito una query, è possibile ottenere statistiche sui dati di input e output elaborati, visualizzare una rappresentazione grafica del tempo impiegato per ogni fase della query ed esplorare in modo interattivo i dettagli di esecuzione.

Visualizzare le statistiche di una query completata

1. Dopo aver eseguito una query nell'editor di Athena, seleziona la scheda Query stats (statistiche delle query).



La scheda Query stats (statistiche delle query) fornisce le seguenti informazioni:

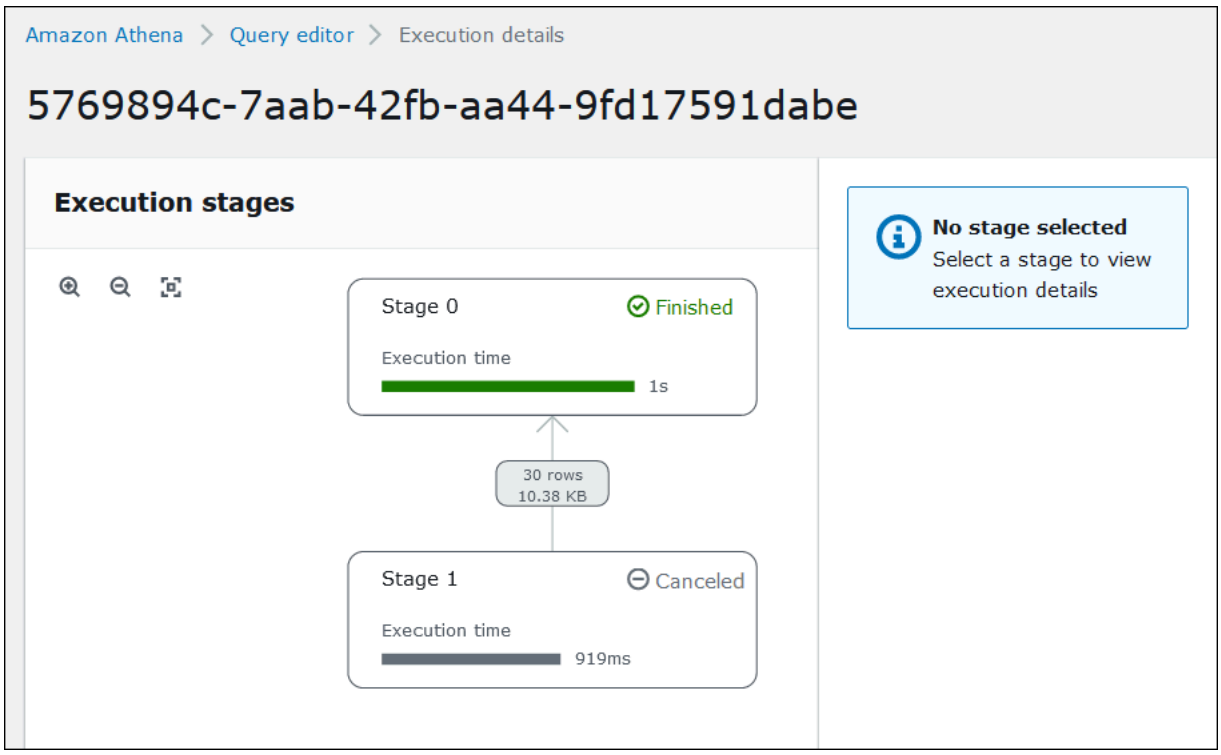
- Data processed (dati elaborati): mostra il numero di righe di input e byte elaborati e il numero di righe e byte di output.
- The Total runtime (runtime totale): mostra il tempo totale impiegato dalla query per l'esecuzione e una rappresentazione grafica di quanto tempo è stato impiegato per la creazione della coda, la pianificazione, l'esecuzione e l'elaborazione del servizio.

> **Note**
>
> Il conteggio delle righe di input e output a livello di fase e le informazioni sulla dimensione dei dati non vengono visualizzati quando una query ha filtri a livello di riga definiti in Lake Formation.

2. Per esplorare in modo interattivo le informazioni sulla modalità di esecuzione della query, seleziona Execution details (dettagli dell'esecuzione).

La pagina Execution details (dettagli dell'esecuzione) mostra l'ID di esecuzione e un grafico delle fasi a base zero nella query. Le fasi sono ordinate dall'inizio alla fine, dal basso verso l'alto. L'etichetta di ogni fase mostra il tempo impiegato per l'esecuzione della fase.

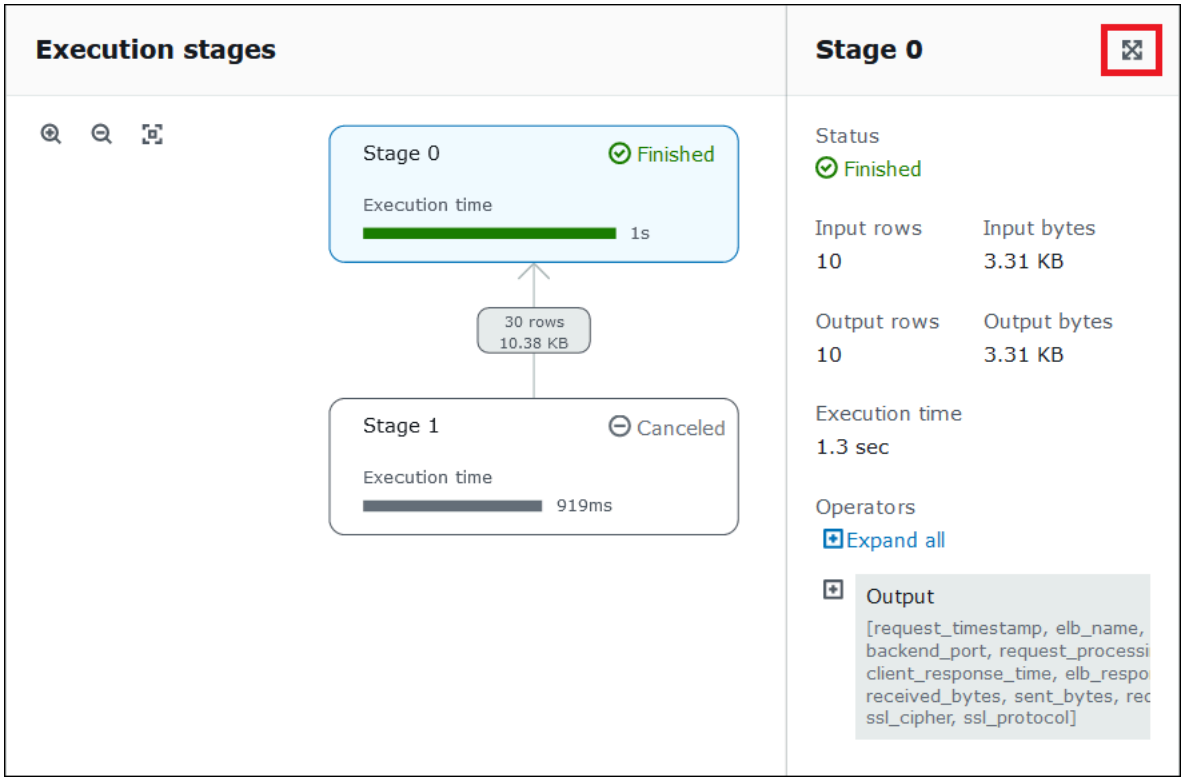
> **Note**
>
> Il tempo totale di esecuzione e la durata della fase di esecuzione di una query spesso differiscono in modo significativo. Ad esempio, una query con una durata totale in minuti può mostrare il tempo di esecuzione per una fase in ore. Poiché una fase è un'unità logica di calcolo eseguita in parallelo su molte attività, il tempo di esecuzione di una fase è il tempo di esecuzione aggregato di tutte le relative attività. Nonostante questa discrepanza, il tempo di esecuzione dello stage può essere utile come indicatore relativo di quale fase ha richiesto la maggiore intensità di calcolo in una query.



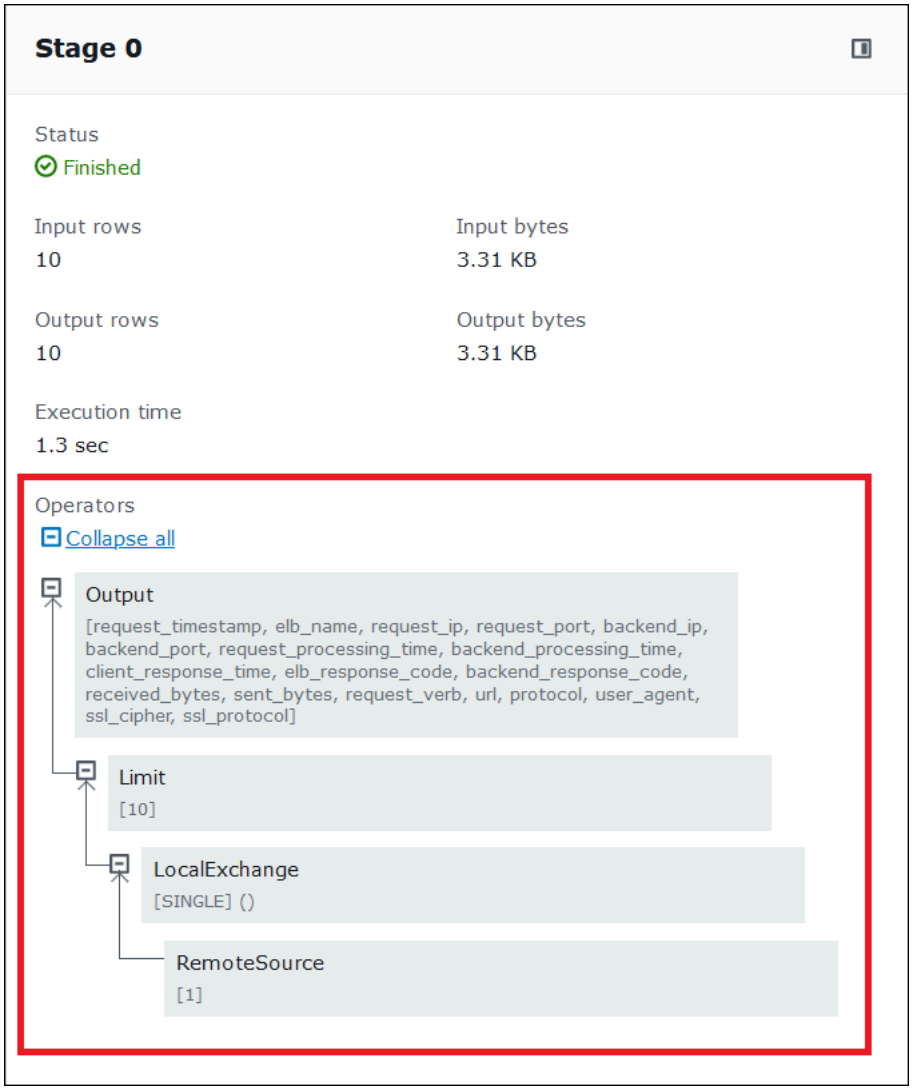
Per navigare nel grafico, utilizza le seguenti opzioni:

- Per ingrandire o ridurre l'immagine, fai scorrere il mouse o utilizza le icone di ingrandimento.
- Per regolare il grafico in base alle dimensioni della schermata, seleziona l'icona Zoom to fit (usa lo zoom per adattare l'immagine).
- Per spostare il grafico, trascina il puntatore del mouse.

3. Per visualizzare maggiori dettagli di una fase, selezionala. Il riquadro dei dettagli della fase sulla destra mostra il numero di righe e byte di input e output e un albero dell'operatore.



4. Per visualizzare i dettagli della fase a schermo intero, seleziona l'icona di espansione in alto a destra nel pannello dei dettagli.
5. Per ottenere informazioni sulle parti della fase, espandi uno o più elementi nell'albero dell'operatore.



Per ulteriori informazioni sui dettagli dell'esecuzione, consulta Capire i risultati dell'istruzione EXPLAIN di Athena.

Risorse aggiuntive

Per ulteriori informazioni, consulta le risorse seguenti.

Visualizzazione dei piani di esecuzione per query SQL

Utilizzo di EXPLAIN e EXPLAIN ANALYZE in Athena

Utilizzo delle visualizzazioni

Una visualizzazione in Amazon Athena è una tabella logica, non fisica. La query che definisce una visualizzazione viene eseguita ogni volta che si fa riferimento alla visualizzazione in una query.

È possibile creare una visualizzazione da una query `SELECT` e quindi fare riferimento a questa visualizzazione nelle query future. Per ulteriori informazioni, consulta CREATE VIEW.

Argomenti

Quando usare le visualizzazioni?

È possibile creare visualizzazioni per:

- Eseguire query su un sottoinsieme di dati. Ad esempio, è possibile creare una visualizzazione con un sottoinsieme di colonne della tabella originale per semplificare l'esecuzione delle query sui dati.
- Combinare più tabelle in un'unica query. Quando si dispone di più tabelle e si desidera combinarle con `UNION ALL`, è possibile creare una visualizzazione con quell'espressione per semplificare le query sulle tabelle combinate.
- Nascondere la complessità delle query di base esistenti e semplificare l'esecuzione delle query da parte degli utenti. Le query di base spesso includono join tra tabelle, espressioni nell'elenco delle colonne e altre sintassi SQL che rendono difficile la comprensione e l'esecuzione del debug. È possibile creare una visualizzazione che nasconda la complessità e semplifichi le query.
- Provare tecniche di ottimizzazione e creare query ottimizzate. Ad esempio, se si trova una combinazione di condizioni `WHERE`, ordine `JOIN` o altre espressioni che dimostrano le prestazioni migliori, è possibile creare una visualizzazione con queste clausole ed espressioni. Le applicazioni possono quindi rendere relativamente semplici le query su questa visualizzazione. Se

successivamente si trova un modo migliore per ottimizzare la query originale, quando si ricrea la visualizzazione, tutte le applicazioni sfruttano immediatamente la query di base ottimizzata.

- Nascondere i nomi delle tabelle e delle colonne sottostanti e ridurre al minimo i problemi di manutenzione in caso di modifica di tali nomi. In questo caso, è necessario ricreare la visualizzazione utilizzando i nuovi nomi. Tutte le query che utilizzano la visualizzazione anziché le tabelle sottostanti continuano a essere eseguite senza modifiche.

Operazioni supportate per le visualizzazioni in Athena

Athena supporta le seguenti operazioni per le visualizzazioni. È possibile eseguire questi comandi nell'Editor delle query.

Dichiarazione	Descrizione
CREATE VIEW	Crea una nuova visualizzazione da una query `SELECT` specificata. Per ulteriori informazioni, consulta Creazione di visualizzazioni. La clausola opzionale `OR REPLACE` consente di aggiornare la visualizz azione esistente sostituendola.
DESCRIBE VIEW	Visualizza l'elenco delle colonne per la visualizzazione specificata. In questo modo è possibile esaminare gli attributi di una visualizzazione complessa.
DROP VIEW	Elimina una visualizzazione esistente. La clausola facoltativa `IF EXISTS` sopprime l'errore se la visualizzazione non esiste.
SHOW CREATE VIEW	Mostra l'istruzione SQL che crea la vista specificata.
SHOW VIEWS	Elenca le viste nel database specificato o, se si omette il nome del database, quelle nel database corrente. Utilizza la clausola `LIKE` facoltativa con un'espressione regolare per limitare l'elenco dei nomi di vista. È possibile visualizzare l'elenco delle visualizzazioni anche nel riquadro a sinistra della console.
SHOW COLUMNS	Elenca le colonne nello schema per una visualizzazione.

Considerazioni per le visualizzazioni

Le seguenti considerazioni valgono per la creazione e l'utilizzo di visualizzazioni in Athena:

- In Athena, puoi visualizzare in anteprima e lavorare con le viste create nella console Athena, in AWS Glue Data Catalog, se hai effettuato la migrazione per utilizzarla, o con Presto in esecuzione sul cluster Amazon EMR connesso allo stesso catalogo. Non è possibile visualizzare in anteprima o aggiungere ad Athena visualizzazioni create in altri modi.
- Se hai creato visualizzazioni Athena nel Catalogo dati, le visualizzazioni vengono trattate come tabelle. È possibile utilizzare il controllo degli accessi granulare a livello di tabella nel catalogo dati per limitare l'accesso a queste visualizzazioni.
- Athena impedisce di eseguire visualizzazioni ricorsive e, in questi casi, viene visualizzato un messaggio di errore. Una visualizzazione ricorsiva è una query di visualizzazione che fa riferimento a se stessa.
- Athena visualizza un messaggio di errore quando rileva viste non aggiornate. Una vista obsoleta viene segnalata quando si verifica una delle seguenti operazioni:
  - La vista fa riferimento a tabelle o database che non esistono.
  - Una modifica dello schema o dei metadati viene effettuata in una tabella di riferimento.
  - Una tabella di riferimento viene eliminata e ricreata con uno schema o una configurazione diversa.
- È possibile creare ed eseguire visualizzazioni nidificate a condizione che la query alla base della visualizzazione nidificata sia valida e le tabelle e i database esistano.

Limitazioni per le visualizzazioni

- I nomi delle visualizzazioni Athena non possono contenere caratteri speciali, diversi dal trattino basso (`_`). Per ulteriori informazioni, consulta Nomi di tabelle, database e colonne.
- Occorre evitare di utilizzare parole chiave per nominare le visualizzazioni. Se si utilizzano parole chiave riservate, occorre utilizzare le virgolette doppie per racchiudere le parole chiave riservate nella query sulle visualizzazioni. Per informazioni, consulta Parole chiave riservate.
- Non puoi utilizzare le visualizzazioni create in Athena con metastore Hive esterni o UDF. Per informazioni sull'utilizzo di visualizzazioni create esternamente in Hive, consulta Utilizzo delle visualizzazioni Hive.
- Non è possibile utilizzare visualizzazioni con funzioni geospaziali.

- Non è possibile utilizzare le visualizzazioni per gestire il controllo degli accessi sui dati in Amazon S3. Per eseguire query su una visualizzazione, è necessario disporre delle autorizzazioni per accedere ai dati archiviati in Amazon S3. Per ulteriori informazioni, consulta Accesso ad Amazon S3 da Athena.
- Sebbene l'esecuzione di query sulle visualizzazioni tra gli account sia supportata sia nella versione 2 sia nella versione 3 del motore Athena, non puoi creare una visualizzazione che AWS Glue Data Catalog su più account. Per ulteriori informazioni sull'accesso tra account ai cataloghi dati, consulta Accesso tra account ai cataloghi dati AWS Glue.
- Le colonne di metadati nascosti Hive o Iceberg `$bucket`, `$file_modified_time`, `$file_size` e `$partition` non sono supportate per le visualizzazioni in Athena. Per informazioni sull'utilizzo della colonna dei metadati `$path` in Athena, consulta Ottenere le posizioni dei file per i dati di origine in Amazon S3 .

Utilizzo delle visualizzazioni nella console

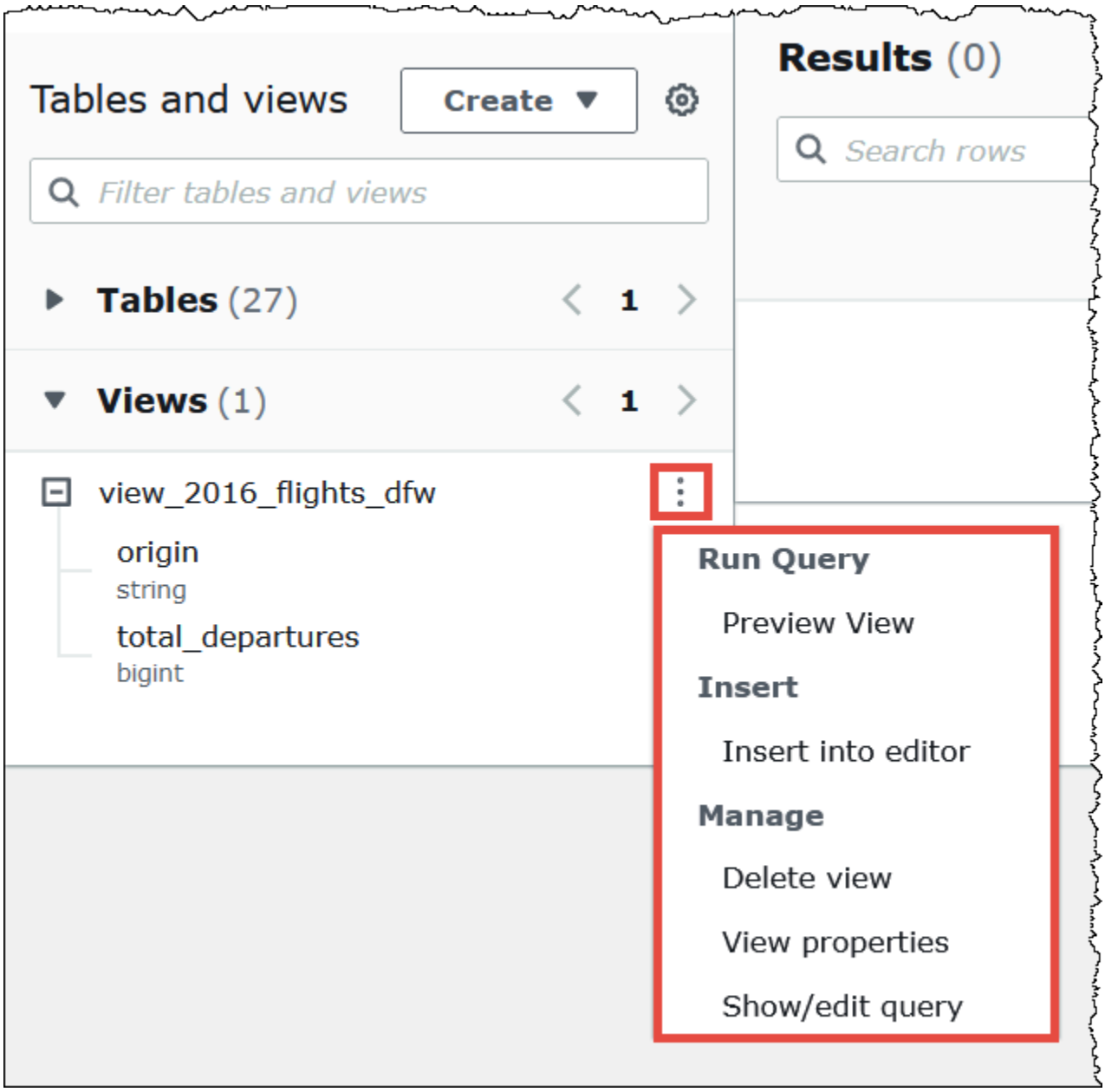
Nella console Athena, puoi:

- Individuare tutte le visualizzazioni nel riquadro a sinistra, in cui sono elencate le tabelle.
- Filtrare le visualizzazioni.
- Eseguire l'anteprima di una visualizzazione, visualizzarne le proprietà, modificarla o eliminarla.

Per mostrare le operazioni per una visualizzazione

Una visualizzazione viene mostrata nella console solo se è già stata creata.

1. Nella console Athena, seleziona Views (Visualizzazioni) e quindi scegli una visualizzazione per espanderla e mostrare le colonne al suo interno.
2. Scegli i tre punti verticali accanto alla visualizzazione per visualizzare un elenco di operazioni.



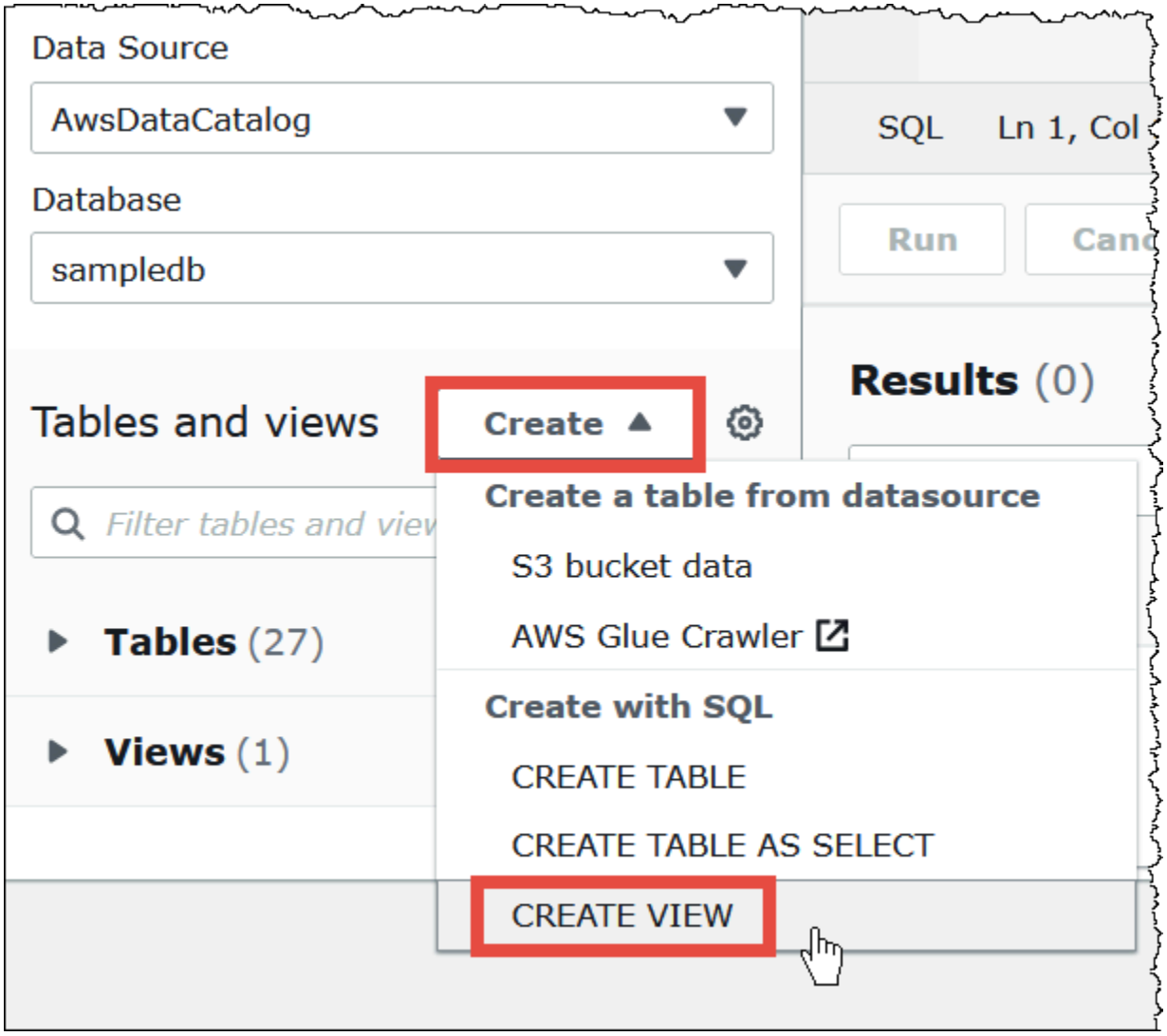
3. Scegli le operazioni per visualizzare in anteprima la visualizzazione, inserire il nome della visualizzazione nell'editor di query, eliminare la visualizzazione, visualizzare le proprietà della visualizzazione o visualizzare e modificare la visualizzazione nell'editor di query.

Creazione di visualizzazioni

È possibile creare una visualizzazione nella console di Athena utilizzando un modello o eseguendo una query esistente.

Per utilizzare un modello per creare una visualizzazione

1. Nella console Athena, accanto a Tables and views (Tabelle e visualizzazioni), scegli Create (Crea) e quindi scegli Create view (Crea visualizzazione).



   Questa operazione inserisce un modello di visualizzazione modificabile nell'editor di query.

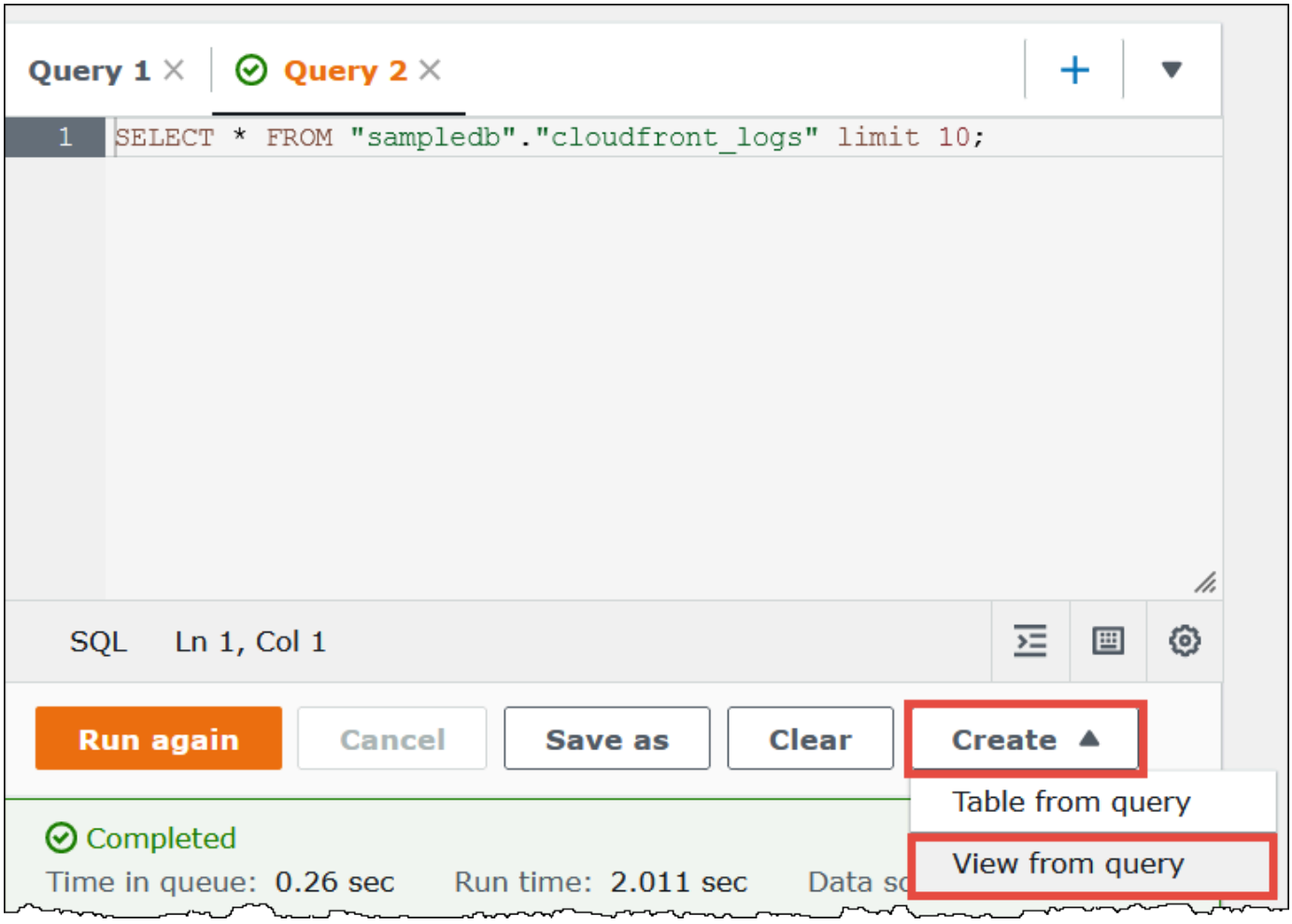
2. Modifica il modello di visualizzazione in base alle tue esigenze. Quando inserisci un nome per la visualizzazione nell'istruzione, ricorda che i nomi delle visualizzazioni non possono contenere caratteri speciali diversi dal carattere di sottolineatura (`_`). Per informazioni, consulta Nomi di tabelle, database e colonne. Evitare di utilizzare Parole chiave riservate per nominare le visualizzazioni.

   Per ulteriori informazioni sulla creazione di visualizzazioni, consulta CREATE VIEW e Esempi di visualizzazioni.

3. Scegli Run (Esegui) per creare la visualizzazione. La visualizzazione viene mostrata nell'elenco delle visualizzazioni nella console Athena.

Per creare una visualizzazione da una query esistente

1. Utilizza l'editor di query Athena per eseguire una query esistente.
2. Nella finestra dell'editor di query, scegli Create (Crea) e quindi scegli View from query (Visualizzazione da query).



3. Nella finestra di dialogo Create view (Crea visualizzazione), inserisci un nome per la visualizzazione, quindi scegli Create (Crea). I nomi delle visualizzazioni non possono contenere caratteri speciali, diversi dal carattere di sottolineatura (_). Per informazioni, consulta Nomi di tabelle, database e colonne. Evitare di utilizzare Parole chiave riservate per nominare le visualizzazioni.

Athena aggiunge la visualizzazione all'elenco delle visualizzazioni nella console e mostra l'istruzione `CREATE VIEW` per la visualizzazione nell'editor di query.

Note

- Se si elimina una tabella su cui è basata una tabella e quindi si cerca di eseguire la visualizzazione, Athena mostra un messaggio di errore.
- È possibile creare una visualizzazione nidificata, ovvero una visualizzazione sopra una visualizzazione esistente. Athena ti impedisce di eseguire una visualizzazione ricorsiva che fa riferimento a se stessa.

Esempi di visualizzazioni

Per visualizzare la sintassi della query di visualizzazione, utilizza SHOW CREATE VIEW.

Example Esempio 1

Considera le due tabelle seguenti: una tabella `employees` con due colonne, `id` e `name`, e una tabella `salaries` con due colonne, `id` e `salary`.

In questo esempio, creiamo una visualizzazione denominata `name_salary` come query `SELECT` che ottiene un elenco di ID mappati agli stipendi delle tabelle `employees` e `salaries`:

```
CREATE VIEW name_salary AS
SELECT
 employees.name,
 salaries.salary
FROM employees, salaries
WHERE employees.id = salaries.id
```

Example Esempio 2

Nell'esempio seguente, creiamo una visualizzazione denominata `view1` che consente di nascondere la sintassi delle query più complesse.

Questa visualizzazione viene eseguita su due tabelle, `table1` e `table2`, ognuna delle quali è una query `SELECT` diversa. La visualizzazione seleziona le colonne di `table1` e unisce i risultati con `table2`. Il join si basa sulla colonna a presente in entrambe le tabelle.

```
CREATE VIEW view1 AS
WITH
  table1 AS (
         SELECT a,
         MAX(b) AS the_max
         FROM x
         GROUP BY a
         ),
  table2 AS (
         SELECT a,
         AVG(d) AS the_avg
         FROM y
         GROUP BY a)
SELECT table1.a, table1.the_max, table2.the_avg
FROM table1
JOIN table2
ON table1.a = table2.a;
```

Per informazioni relative all'esecuzione di query su visualizzazioni federate, consulta Esecuzione di query su visualizzazioni federate.

Usare le viste AWS Glue Data Catalog

Questa caratteristica è in versione di anteprima ed è soggetta a modifica. Per ulteriori informazioni, consulta la sezione Beta e anteprime nel documento Termini del servizio AWS.

Usa le AWS Glue Data Catalog viste quando desideri un'unica vista comune su Servizi AWS Amazon Athena e Amazon Redshift. Nelle viste di Catalogo dati, le autorizzazioni di accesso sono definite dall'utente che ha creato la vista anziché dall'utente che la interroga. Questo metodo di concessione delle autorizzazioni è chiamato semantica del definitore.

Di seguito sono riportati casi d'uso che mostrano come è possibile utilizzare le viste di Catalogo dati.

- Maggiore controllo degli accessi: crea una vista che limita l'accesso ai dati in base al livello di autorizzazioni richiesto dall'utente. Ad esempio, è possibile utilizzare le viste di Catalogo dati per impedire ai dipendenti che non lavorano nel reparto delle risorse umane di visualizzare informazioni di identificazione personale.
- Completezza dei record garantita: applicando determinati filtri alla vista di Catalogo dati, puoi assicurarti che i record di dati in una vista di Catalogo dati siano sempre completi.

- Sicurezza avanzata: nelle viste di Catalogo dati, la definizione della query che crea la vista deve essere intatta per assicurare la corretta creazione della vista. Ciò rende le visualizzazioni di Catalogo dati meno suscettibili ai comandi SQL da parte di soggetti malintenzionati.
- Accesso alle tabelle sottostanti interdetto: la semantica del definitore consente agli utenti di accedere a una vista senza rendere loro disponibile la tabella sottostante. Solo l'utente che definisce la vista richiede l'accesso alle tabelle.

Le definizioni delle viste di Catalogo dati sono archiviate in AWS Glue Data Catalog. Ciò significa che è possibile utilizzare AWS Lake Formation per concedere l'accesso tramite concessioni di risorse, concessioni di colonne o controlli di accesso basati su tag. Per ulteriori informazioni sulla concessione e la revoca dell'accesso a Lake Formation, consulta la pagina Granting and revoking permissions on Data Catalog resources nella Guida per gli sviluppatori di AWS Lake Formation .

Autorizzazioni

Le viste di Catalogo Dati richiedono tre ruoli: `Lake Formation Admin`, `Definer` e `Invoker`.

- **`Lake Formation Admin`**: ha accesso alla configurazione di tutte le autorizzazioni di Lake Formation.
- **`Definer`**: crea la vista di Catalogo Dati. Il ruolo `Definer` deve disporre di autorizzazioni `SELECT` complete per tutte le tabelle sottostanti a cui la definizione della vista fa riferimento.
- **`Invoker`**: può interrogare la vista di Catalogo Dati o controllarne i metadati. Per mostrare l'invocatore di una query, puoi usare la funzione DML. `invoker_principal()` Per ulteriori informazioni, consulta invoker_principal ().

Le relazioni di fiducia del `Definer` ruolo devono consentire l'`sts:AssumeRole`azione dei responsabili del servizio AWS Glue e di Lake Formation, come nell'esempio seguente.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "glue.amazonaws.com",
                    "lakeformation.amazonaws.com"
```

```
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Sono inoltre necessarie le autorizzazioni IAM per l'accesso ad Athena. Per ulteriori informazioni, consulta AWS politiche gestite per Amazon Athena.

Limitazioni

- Le viste di Catalogo Dati non possono fare riferimento ad altre viste.
- È possibile fare riferimento a un massimo di 10 tabelle nella definizione della vista.
- Le tabelle sottostanti devono essere registrate con Lake Formation.
- Il principale `DEFINER` può essere solo un ruolo IAM.
- Il ruolo `DEFINER` deve disporre di autorizzazioni `SELECT` (concedibili) complete per tutte le tabelle sottostanti.
- Le viste di Catalogo Dati di `UNPROTECTED` non sono supportate.
- Le funzioni definite dall'utente (UFD) non sono supportate nella definizione della vista.
- Le origini dati federate Athena non possono essere utilizzate nelle viste di Catalogo Dati.
- Le viste di Catalogo Dati non sono supportate per i metastore Hive esterni.
- Athena visualizza un messaggio di errore quando rileva viste non aggiornate. Una vista obsoleta viene segnalata quando si verifica una delle seguenti operazioni:
  - La vista fa riferimento a tabelle o database che non esistono.
  - Una modifica dello schema o dei metadati viene effettuata in una tabella di riferimento.
  - Una tabella di riferimento viene eliminata e ricreata con uno schema o una configurazione diversa.

Creazione di una vista di Catalogo Dati

La sintassi di esempio seguente mostra come un utente del ruolo `Definer` crea la vista di Catalogo Dati di `orders_by_date`. L'esempio presuppone che il ruolo `Definer` disponga delle autorizzazioni `SELECT` complete sulla tabella `orders` del database `default`.

```
CREATE PROTECTED MULTI DIALECT VIEW orders_by_date
```

```
SECURITY DEFINER
AS
SELECT orderdate, sum(totalprice) AS price
FROM orders
WHERE order_city = 'SEATTLE'
GROUP BY orderdate
```

Interrogazione di una vista di Catalogo Dati

Dopo la creazione della vista, l'`Admin` di `Lake Formation` può concedere le autorizzazioni `SELECT` per la vista di Catalogo Dati ai principali `Invoker`. I principali `Invoker` possono quindi interrogare la vista senza avere accesso alle tabelle di base sottostanti a cui la vista fa riferimento. Di seguito è riportato un esempio di query `Invoker`.

```
SELECT * from orders_by_date where price > 5000
```

Aggiornamento di una vista di Catalogo Dati

L'`Lake Formation Admin` oppure il `Definer` possono utilizzare la sintassi `ALTER VIEW UPDATE DIALECT` per aggiornare la definizione della vista. L'esempio seguente modifica la definizione della vista per selezionare le colonne dalla tabella `returns` anziché dalla tabella `orders`.

```
ALTER VIEW orders_by_date UPDATE DIALECT
AS
SELECT return_date, sum(totalprice) AS price
FROM returns
WHERE order_city = 'SEATTLE'
GROUP BY orderdate
```

Per ulteriori informazioni sulla sintassi per la creazione e la gestione delle viste di Catalogo Dati, consulta la pagina Sintassi delle viste di Catalogo Dati di Glue.

Sintassi delle viste di Catalogo Dati di Glue

> Questa caratteristica è in versione di anteprima ed è soggetta a modifica. Per ulteriori informazioni, consulta la sezione Beta e anteprime nel documento Termini del servizio AWS.

Questa sezione descrive i comandi DDL (Data Definition Language) per la creazione e la gestione delle AWS Glue Data Catalog viste.

ALTER VIEW DIALECT

È possibile aggiornare le viste di Catalogo Dati aggiungendo un dialetto del motore o aggiornando o eliminando un dialetto del motore esistente. Solo l'`Admin` di `Lake Formation` e `Definer` (l'utente che ha creato la vista) sono autorizzati a utilizzare l'istruzione `ALTER VIEW DIALECT` in una vista di Catalogo Dati.

Sintassi

```
ALTER VIEW name [ FORCE ] [ ADD|UPDATE ] DIALECT AS query
```

```
ALTER VIEW name [ DROP ] DIALECT
```

FORCE

La parola chiave `FORCE` causa la sovrascrittura delle informazioni in dialetto del motore di una vista in conflitto con la nuova definizione. La parola chiave `FORCE` è utile quando un aggiornamento di una vista di Catalogo Dati genera definizioni delle viste in conflitto tra i dialetti del motore esistenti. Supponiamo che una vista di Catalogo Dati contenga entrambi i dialetti Athena e Amazon Redshift e che l'aggiornamento generi un conflitto con Amazon Redshift nella definizione della vista. In questo caso, puoi utilizzare la parola chiave `FORCE` per consentire il completamento dell'aggiornamento e contrassegnare il dialetto di Amazon Redshift come obsoleto. Quando i motori contrassegnati come obsoleti interrogano la vista, la query ha esito negativo. I motori generano un'eccezione per non consentire risultati obsoleti. Per correggere questo problema, aggiorna i dialetti obsoleti nella vista.

ADD

Aggiunge un nuovo dialetto del motore alla vista di Catalogo Dati. Il motore specificato non può esistere già nella vista di Catalogo Dati.

UPDATE

Aggiorna un dialetto del motore esistente nella vista di Catalogo Dati.

DROP

Elimina un dialetto del motore esistente da una vista di Catalogo Dati. Dopo aver eliminato un motore da una vista di Catalogo Dati, tale vista non può essere interrogata dal motore che è stato disattivato. Gli altri dialetti del motore nella vista possono comunque interrogare la vista.

DIALECT AS

Introduce una query SQL specifica per il motore.

Esempi

```
ALTER VIEW orders_by_date FORCE ADD DIALECT
AS
SELECT orderdate, sum(totalprice) AS price
FROM orders
GROUP BY orderdate
```

```
ALTER VIEW orders_by_date FORCE UPDATE DIALECT
AS
SELECT orderdate, sum(totalprice) AS price
FROM orders
GROUP BY orderdate
```

```
ALTER VIEW orders_by_date DROP DIALECT
```

CREATE PROTECTED MULTI DIALECT VIEW

Crea una vista del catalogo dati in. AWS Glue Data Catalog Una vista di Catalogo Dati è uno schema di visualizzazione unico che funziona perfettamente su Athena e altri motori SQL come Amazon Redshift e Amazon EMR.

Sintassi

```
CREATE [ OR REPLACE ] PROTECTED MULTI DIALECT VIEW view_name
 [ SECURITY DEFINER ]
AS query
```

PROTECTED

La parola chiave è obbligatoria. Specifica che la vista è protetta contro le fughe di dati. Le viste di Catalogo Dati possono essere create solo come vista `PROTECTED`.

MULTI DIALECT

Specifica che la vista supporta i dialetti SQL di diversi motori di query e può quindi essere letta da tali motori.

SECURITY DEFINER

Specifica che la semantica dei definitori è in vigore per questa vista. La semantica del definitore indica che i permessi di lettura effettivi per le tabelle sottostanti appartengono al principale o al ruolo che ha definito la vista anziché al principale che esegue la lettura effettiva.

OR REPLACE

Una vista di Catalogo Dati non può essere sostituita se nella vista sono presenti dialetti SQL di altri motori. Se il motore chiamante ha l'unico dialetto SQL presente nella vista, la vista può essere sostituita.

Esempio

L'esempio seguente crea la vista di Catalogo Dati `orders_by_date` in base a una query sulla tabella `orders`.

```
CREATE PROTECTED MULTI DIALECT VIEW orders_by_date
SECURITY DEFINER
AS
SELECT orderdate, sum(totalprice) AS price
FROM orders
WHERE order_city = 'SEATTLE'
GROUP BY orderdate
```

DESCRIBE

Mostra l'elenco delle colonne per la vista di Catalogo Dati specificata. L'istruzione `DESCRIBE` è simile all'istruzione `DESCRIBE` per le viste di Athena. A differenza delle viste di Athena, l'output dell'istruzione è controllato tramite il controllo degli accessi di Lake Formation. Pertanto, l'output di questa query non è costituito da tutte le colonne della vista, ma soltanto dalle colonne a cui il chiamante ha accesso.

Sintassi

```
DESCRIBE [db_name.]view_name
```

Esempi

```
DESCRIBE orders
```

DROP VIEW

Elimina una vista di Catalogo Dati solo se il dialetto del motore di chiamata è presente nella vista di Catalogo Dati. Ad esempio, se un utente chiama `DROP VIEW` da Athena, la vista viene eliminata solo se al suo interno è presente il dialetto di Athena. In caso contrario, l'operazione non va a buon fine. Solo l'amministratore di Lake Formation e il definitore della vista sono autorizzati a utilizzare l'istruzione `DROP VIEW` in una vista di Catalogo Dati.

Sintassi

```
DROP VIEW [ IF EXISTS ] view_name
```

Esempi

```
DROP VIEW orders_by_date
```

```
DROP FORCE VIEW IF EXISTS orders_by_date
```

La clausola facoltativa `IF EXISTS` provoca l'errore da sopprimere se la vista non esiste.

SHOW COLUMNS

Mostra solo i nomi delle colonne per una singola vista di Catalogo Dati specificata. L'istruzione `SHOW COLUMNS` è simile all'istruzione `SHOW COLUMNS` per le viste di Athena. A differenza delle viste di Athena, l'output dell'istruzione è controllato tramite il controllo degli accessi di Lake Formation. Pertanto, l'output di questa query non è costituito da tutte le colonne della vista, ma soltanto dalle colonne a cui il chiamante ha accesso.

Sintassi

```
SHOW COLUMNS {FROM|IN} database_name.view_name
```

```
SHOW COLUMNS {FROM|IN} view_name [{FROM|IN} database_name]
```

SHOW CREATE VIEW

Mostra la sintassi SQL che ha creato la vista di Catalogo Dati. L'istruzione SQL restituita mostra la sintassi di creazione della vista utilizzata in Athena. Solo l'amministratore di Lake Formation e

i principali definitori della vista sono autorizzati a chiamare `SHOW CREATE VIEW` su una vista di Catalogo Dati.

Sintassi

```
SHOW CREATE VIEW view_name
```

Esempi

```
SHOW CREATE VIEW orders_by_date
```

SHOW VIEWS

Elenca i nomi di tutte le viste nel database. Sono elencate tutte le viste di Catalogo Dati nel database che utilizzano il dialetto SQL del motore Athena. Le altre viste di Catalogo Dati che non dispongono del dialetto del motore Athena nella vista sono escluse.

Sintassi

```
SHOW VIEWS [IN database_name] [LIKE 'regular_expression']
```

Esempi

```
SHOW VIEWS
```

```
SHOW VIEWS IN marketing_analytics LIKE 'orders*'
```

Utilizzo di query salvate

Puoi utilizzare la console Athena per salvare, modificare, eseguire, rinominare ed eliminare le query create nell'editor di query.

Considerazioni e limitazioni

- Puoi aggiornare il nome, la descrizione e il testo delle query salvate.
- Puoi aggiornare le query solo nel tuo account.
- Non puoi modificare il gruppo di lavoro o il database a cui appartiene la query.

- Athena non conserva una cronologia delle modifiche alle query. Se desideri mantenere una versione particolare di una query, salvala con un nome diverso.

Utilizzo delle query salvate nella console Athena

Salvare e denominare una query

1. Nell'editor di query della console Athena, inserisci o esegui una query.
2. Sopra la finestra dell'editor di query, nella scheda della query, seleziona i tre punti verticali, quindi scegli Save as (Salva con nome).
3. Nella finestra di dialogo Save query (Salva query), inserisci un nome per la query e una descrizione facoltativa. Puoi utilizzare la finestra espandibile Preview SQL query (Anteprima query SQL) per verificare il contenuto della query prima di salvarla.
4. Scegli Save query (Salva query).

   Nell'editor di query, la scheda della query mostra il nome specificato.

Per eseguire una query salvata

1. Nella console Athena, scegli la scheda Query salvate.
2. Nell'elenco Saved queries (Query salvate), scegli il nome della query da eseguire.

   L'editor di query visualizza la query scelta.
3. Scegli Run (Esegui).

Modificare una query salvata

1. Nella console Athena, scegli la scheda Query salvate.
2. Nell'elenco Saved queries (Query salvate), scegli il nome della query da modificare.
3. Modifica la query nell'editor di query.
4. Eseguire una delle seguenti fasi:

   - Per eseguire la query, scegli Run (Esegui).
   - Per salvare la query, seleziona i tre punti verticali nella scheda della query, quindi scegli Save (Salva).

- Per salvare la query con un nome diverso, seleziona i tre punti verticali nella scheda della query, quindi scegli Save as (Salva con nome).

Ridenominazione o eliminazione di una query salvata già visualizzata nell'editor di query

1. Seleziona i tre punti verticali nella scheda per della query, quindi scegli Rename (Rinomina) o Delete (Elimina).
2. Segui le istruzioni per rinominare o eliminare la query.

Ridenominazione di una query salvata non visualizzata nell'editor di query

1. Nella console Athena, scegli la scheda Query salvate.
2. Seleziona la casella di controllo per la query che desideri rinominare.
3. Scegliere Rinomina.
4. Nella finestra di dialogo Rename query (Rinomina query), modifica il nome e la descrizione della query. Puoi utilizzare la finestra espandibile Preview SQL query (Anteprima query SQL) per verificare il contenuto della query prima di rinominarla.
5. Scegli Rename query (Rinomina query).

   La query rinominata viene visualizzata nell'elenco Query salvate.

Eliminazione di una query salvata non visualizzata nell'editor di query

1. Nella console Athena, scegli la scheda Query salvate.
2. Seleziona una o più caselle di controllo per le query da eliminare.
3. Scegliere Delete (Elimina).
4. Alla richiesta di conferma, scegli Delete (Elimina).

   Le query vengono rimosse dall'elenco Saved queries (Query salvate).

Utilizzo dell'API Athena per aggiornare le query salvate

Per informazioni sull'utilizzo dell'API Athena per aggiornare una query salvata, consulta l'operazione UpdateNamedQuery nella documentazione di riferimento dell'API Athena.

Utilizzo di query parametrizzate

È possibile utilizzare le query con parametri Athena per rieseguire la stessa query con valori di parametri diversi in fase di esecuzione e contribuire a prevenire gli attacchi di iniezione SQL. In Athena, le query con parametri possono assumere la forma di parametri di esecuzione in qualsiasi query DML o istruzioni preparate in SQL.

- Le query con parametri di esecuzione possono essere eseguite in un unico passaggio e non sono specifiche del gruppo di lavoro. È possibile inserire punti interrogativi in qualsiasi query DML per i valori che si desidera parametrizzare. Quando si esegue la query, si dichiarano i valori dei parametri di esecuzione in sequenza. La dichiarazione dei parametri e l'assegnazione dei valori per i parametri possono essere eseguite nella stessa query, ma in modo disaccoppiato. A differenza delle istruzioni preparate, è possibile selezionare il gruppo di lavoro quando si invia una query con parametri di esecuzione.
- Le istruzioni preparate richiedono due istruzioni SQL separate: `PREPARE` e `EXECUTE`. In primo luogo, è necessario definire i parametri nel campo dell'istruzione `PREPARE`. Quindi, viene eseguita un'istruzione `EXECUTE` che fornisce i valori dei parametri definiti. Le istruzioni preparate sono specifiche del gruppo di lavoro; non è possibile eseguirle al di fuori del contesto del gruppo di lavoro a cui appartengono.

Considerazioni e limitazioni

- Le query parametrizzate sono supportate nella versione 2 del motore Athena e versioni successive. Per ulteriori informazioni sulle versioni del motore Athena, consulta Controllo delle versioni del motore di Athena.
- Attualmente, le query con parametri sono supportate solo per le istruzioni `SELECT`, `INSERT INTO`, `CTAS` e `UNLOAD`.
- Nelle query con parametri, i parametri sono posizionali e sono indicati da `?`. Ai parametri vengono assegnati valori in base al loro ordine nella query. I parametri nominati non sono supportati.
- Attualmente, i parametri `?` possono essere inseriti solo nella clausola `WHERE`. Sintassi come `SELECT ? FROM table` non è supportata.
- I parametri del punto interrogativo non possono essere inseriti tra virgolette doppie o singole (ovvero, `'?'` e `"?"` non sono una sintassi valida).
- I parametri di esecuzione SQL, affinché vengano trattati come stringhe, devono essere racchiusi tra virgolette singole anziché tra virgolette doppie.

- Se necessario, puoi utilizzare la funzione `CAST` quando immetti un valore per un termine parametrizzato. Ad esempio, se hai una colonna del tipo `date` che è stata parametrizzata in una query e vuoi eseguire una query per la data `2014-07-05`, l'immissione `CAST('2014-07-05' AS DATE)` del valore del parametro restituirà il risultato.
- Le istruzioni preparate sono specifiche del gruppo di lavoro e i nomi delle istruzioni preparate devono essere univoci all'interno del gruppo di lavoro.
- Sono necessarie autorizzazioni IAM per le istruzioni preparate. Per ulteriori informazioni, consulta Consenti l'accesso alle istruzioni preparate.
- Le query con parametri di esecuzione nella console Athena sono limitate a un massimo di 25 punti interrogativi.

Esecuzione di query tramite parametri di esecuzione

È possibile utilizzare i placeholder del punto interrogativo in qualsiasi query DML per creare una query con parametri senza creare prima un'istruzione preparata. Per eseguire queste query, puoi utilizzare la console Athena oppure utilizzare AWS CLI l'SDK AWS e dichiarare le variabili nell'argomento. `execution-parameters`

Esecuzione di query con parametri di esecuzione nella console Athena

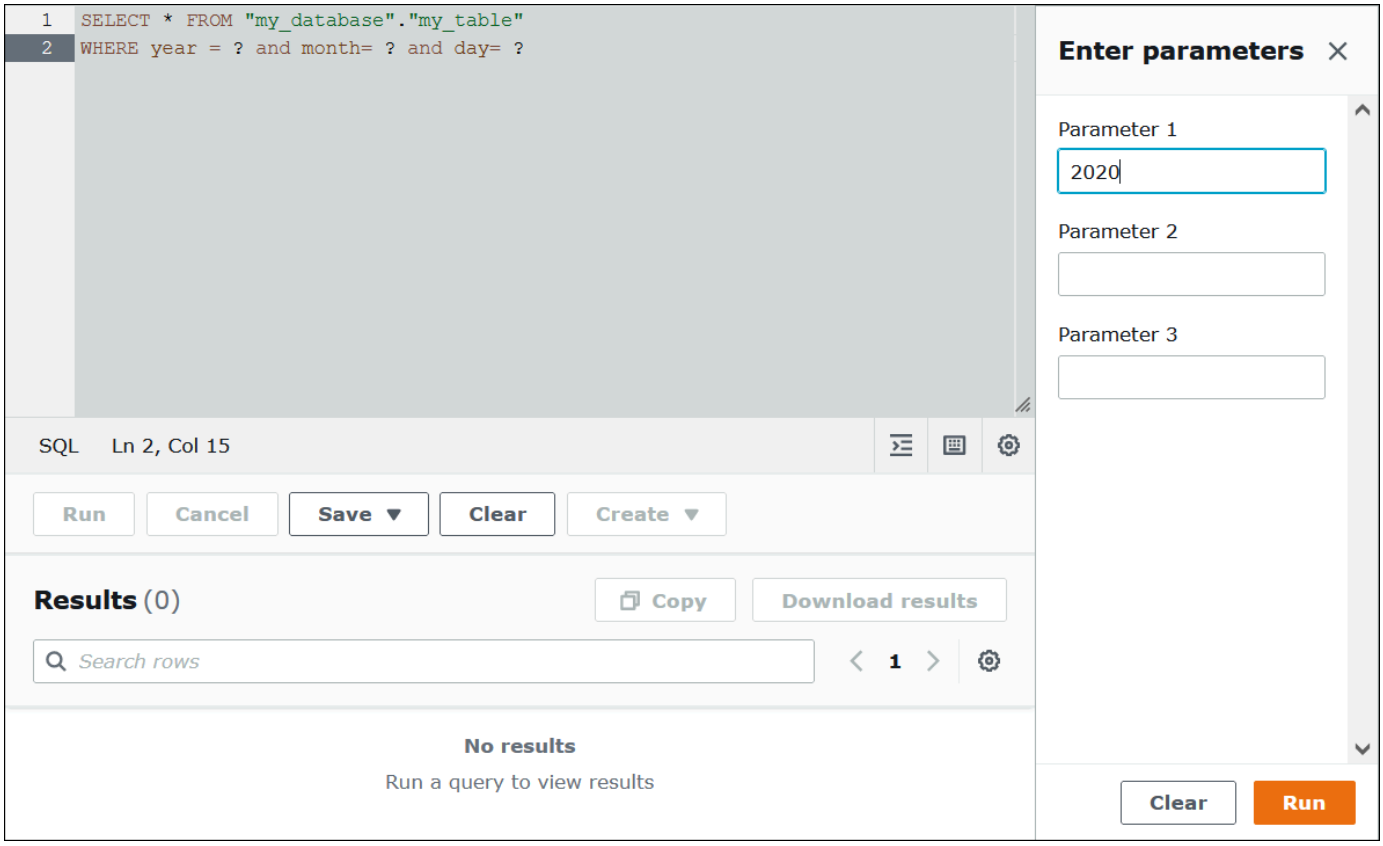
Quando si esegue una query con parametri che include parametri di esecuzione (punti interrogativi) nella console Athena, vengono richiesti i valori nell'ordine in cui si trovano i punti interrogativi nella query.

Per eseguire una query con parametri di esecuzione

1. Inserisci una query con placeholder con punti interrogativi nell'editor di Athena, come nell'esempio seguente.

   ```
   SELECT * FROM "my_database"."my_table"
   WHERE year = ? and month= ? and day= ?
   ```

2. Seleziona Esegui.
3. Nella finestra di dialogo Enter parameters (inserisci parametri), inserisci un valore in ordine per ciascuno dei punti interrogativi nella query.



4. Una volta inseriti i parametri, seleziona Run (esegui). L'editor mostra i risultati della query per i valori dei parametri inseriti.

In questo caso, puoi:

- Inserire valori di parametro diversi per la stessa query, quindi selezionare Run again (esegui nuovamente).
- Per cancellare tutti i valori inseriti contemporaneamente, seleziona Clear (annulla).
- Per modificare direttamente la query (ad esempio, per aggiungere o rimuovere punti interrogativi), chiudi innanzitutto la finestra di dialogo Enter parameters (inserisci parametri).
- Per salvare la query con parametri per un uso successivo, seleziona Save (salva) o Save as (salva con nome) e assegna un nome alla query. Per ulteriori informazioni sull'utilizzo di query salvate, consulta Utilizzo di query salvate.

Per comodità, la finestra di dialogo Enter parameters (inserisci parametri) memorizza i valori inseriti in precedenza per la query purché venga utilizzata la stessa scheda nell'editor di query.

Esecuzione di query con parametri di esecuzione utilizzando AWS CLI

AWS CLI Per eseguire query con parametri di esecuzione, utilizzate il `start-query-execution` comando e fornite una query con parametri nell'argomento. `query-string` Poi, nell'argomento `execution-parameters` fornisci i valori per i parametri di esecuzione. Nell'esempio seguente viene descritta tale tecnica.

```
aws athena start-query-execution
--query-string "SELECT * FROM table WHERE x = ? AND y = ?"
--query-execution-context "Database"="default"
--result-configuration "OutputLocation"="s3://DOC-EXAMPLE-BUCKET;/..."
--execution-parameters "1" "2"
```

Esecuzione di query con istruzioni preparate

È possibile utilizzare le istruzioni preparate per eseguire ripetutamente una stessa query con parametri diversi. Un'istruzione preparata contiene segnaposto di parametro i cui valori vengono forniti al momento dell'esecuzione.

> **Note**
>
> Il numero massimo di istruzioni preparate in un gruppo di lavoro è 1.000.

Istruzioni SQL

Puoi utilizzare le istruzioni SQL `PREPARE`, `EXECUTE` e `DEALLOCATE PREPARE` per eseguire query parametrizzate nell'editor di query della console Athena.

- Per specificare i parametri in cui normalmente si utilizzano valori letterali, utilizzare i punti interrogativi nell'istruzione `PREPARE`.
- Per sostituire i parametri con i valori quando si esegue la query, utilizzare la clausola `USING` nell'istruzione `EXECUTE`.
- Per rimuovere un'istruzione preparata dall'elenco delle istruzioni preparate in un gruppo di lavoro, utilizzare l'istruzione `DEALLOCATE PREPARE`.

Nelle sezioni seguenti vengono forniti ulteriori dettagli su ciascuna di queste istruzioni.

PREPARE

Prepara un'istruzione da eseguire in un secondo momento. Le istruzioni preparate vengono salvate nel gruppo di lavoro corrente con il nome specificato. L'istruzione può includere parametri al posto dei valori letterali da sostituire quando viene eseguita la query. I parametri da sostituire con valori sono denotati da punti interrogativi.

Sintassi

```
PREPARE statement_name FROM statement
```

Nella tabella seguente vengono descritti questi parametri.

Parametro	Descrizione
nome_istruzione	Il nome dell'istruzione da preparare. Il nome deve essere univoco all'interno del gruppo di lavoro.
Istruzione	Una query `SELECT`, `CTAS` o `INSERT INTO`.

Esempi PREPARE

I seguenti esempi mostrano l'uso dell'istruzione `PREPARE`. I punti interrogativi indicano i valori che devono essere forniti dall'istruzione `EXECUTE` quando viene eseguita la query.

```
PREPARE my_select1 FROM
SELECT * FROM nation
```

```
PREPARE my_select2 FROM
SELECT * FROM "my_database"."my_table" WHERE year = ?
```

```
PREPARE my_select3 FROM
SELECT order FROM orders WHERE productid = ? and quantity < ?
```

```
PREPARE my_insert FROM
INSERT INTO cities_usa (city, state)
```

```
SELECT city, state
FROM cities_world
WHERE country = ?
```

```
PREPARE my_unload FROM
UNLOAD (SELECT * FROM table1 WHERE productid < ?)
TO 's3://DOC-EXAMPLE-BUCKET/'
WITH (format='PARQUET')
```

EXECUTE

Esegue un'istruzione preparata. I valori per i parametri sono specificati nella clausola `USING`.

Sintassi

```
EXECUTE statement_name [USING value1 [ ,value2, ... ] ]
```

`statement_name` è il nome dell'istruzione preparata. *`value1`* e *`value2`* sono i valori da specificare per i parametri nell'istruzione.

Esempi EXECUTE

L'esempio seguente esegue l'esecuzione dell'istruzione preparata `my_select1`, che non contiene parametri.

```
EXECUTE my_select1
```

L'esempio seguente esegue l'esecuzione dell'istruzione preparata `my_select2`, che contiene un parametro singolo.

```
EXECUTE my_select2 USING 2012
```

L'esempio seguente esegue l'esecuzione dell'istruzione preparata `my_select3`, che contiene due parametri.

```
EXECUTE my_select3 USING 346078, 12
```

Nell'esempio seguente viene fornito un valore stringa per un parametro nell'istruzione preparata `my_insert`.

```
EXECUTE my_insert USING 'usa'
```

Nell'esempio seguente viene fornito un valore stringa per un parametro `productid` nell'istruzione preparata `my_unload`.

```
EXECUTE my_unload USING 12
```

DEALLOCATE PREPARE

Rimuove l'istruzione preparata con il nome specificato dall'elenco delle istruzioni preparate nel gruppo di lavoro corrente.

Sintassi

```
DEALLOCATE PREPARE statement_name
```

`statement_name` è il nome dell'istruzione preparata da rimuovere.

Esempio

L'esempio seguente rimuove l'istruzione preparata `my_select1` dal gruppo di lavoro corrente.

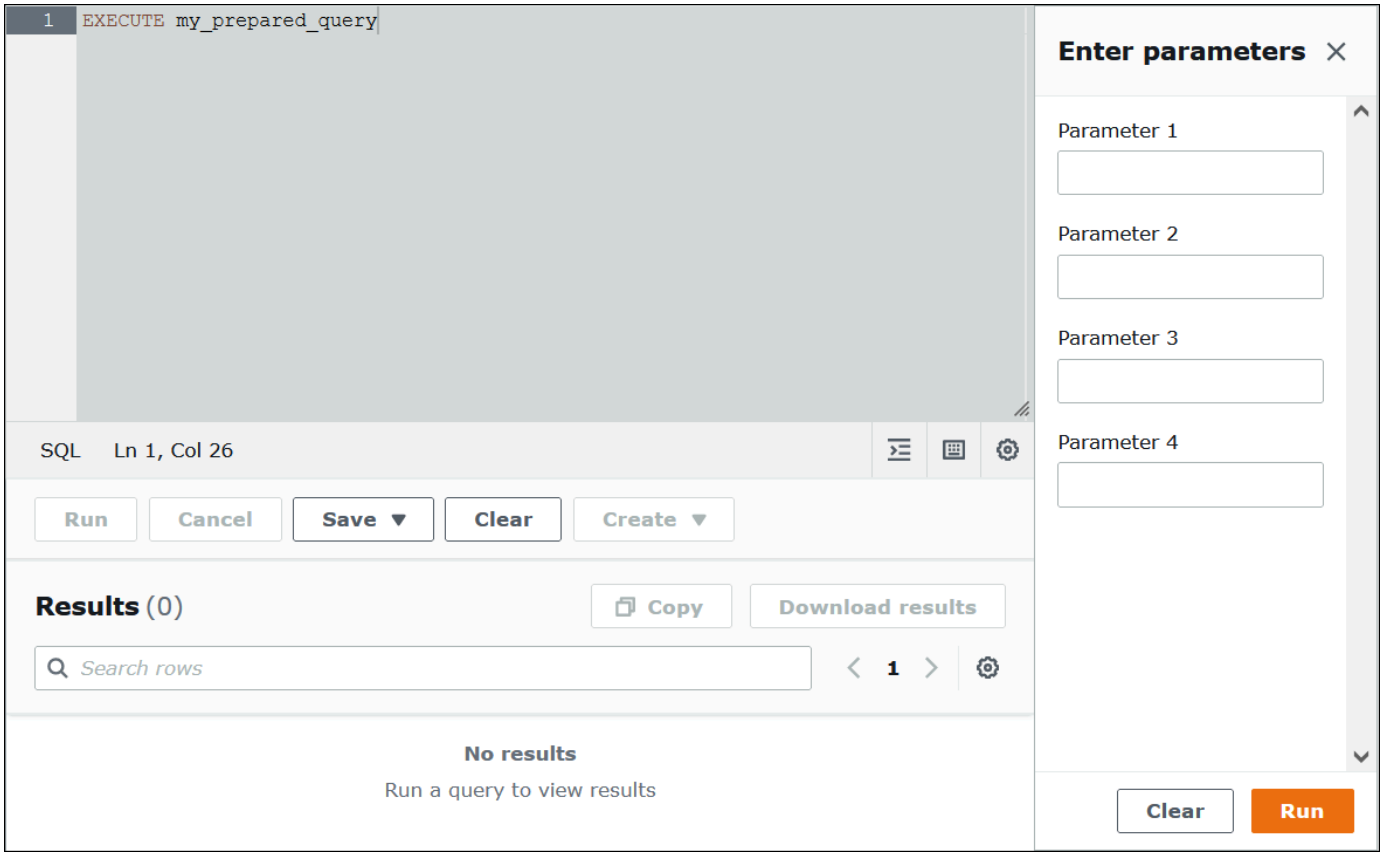
```
DEALLOCATE PREPARE my_select1
```

Esecuzione di istruzioni preparate senza la clausola USING nella console Athena

Se si esegue un'istruzione preparata esistente con la sintassi `EXECUTE` *`prepared_statement`*nell'editor di query, Athena apre la finestra di dialogo Enter parameters (inserisci parametri)per permetterti di inserire i valori che normalmente andrebbero nella clausola `USING` dell'istruzione `EXECUTE ... USING`.

Eseguire un'istruzione preparata utilizzando la finestra di dialogo Enter parameters (inserisci parametri)

1. Nell'editor di query, invece di usare la sintassi `EXECUTE prepared_statement USING` *`value1`*`,` *`value2`* `...`, usa la sintassi `EXECUTE` *`prepared_statement`*.
2. Seleziona Esegui. Apparirà la finestra di dialogo Enter parameters (inserisci parametri).



3. Inserisci i valori in ordine nella finestra di dialogo Execution parameters (parametri di esecuzione). Poiché il testo originale della query non è visibile, è necessario ricordare il significato di ciascun parametro posizionale o avere l'istruzione preparata disponibile come riferimento.
4. Seleziona Esegui.

Creazione di istruzioni preparate utilizzando AWS CLI

Per utilizzare la AWS CLI per creare un'istruzione preparata, è possibile utilizzare uno dei seguenti `athena` comandi:

- Utilizza il comando `create-prepared-statement` e fornisci un'istruzione di query con parametri di esecuzione.
- Utilizza il comando `start-query-execution` e fornisci una stringa di query che utilizzi la sintassi `PREPARE`.

Usando create-prepared-statement

In un comando `create-prepared-statement`, definisci il testo della query nell'argomento `query-statement`, come nell'esempio seguente.

```
aws athena create-prepared-statement
--statement-name PreparedStatement1
--query-statement "SELECT * FROM table WHERE x = ?"
--work-group athena-engine-v2
```

Utilizzo start-query-execution e sintassi PREPARE

Utilizza il comando `start-query-execution`. Inserisci l'istruzione `PREPARE` nell'argomento `query-string`, come nell'esempio seguente:

```
aws athena start-query-execution
--query-string "PREPARE PreparedStatement1 FROM SELECT * FROM table WHERE x = ?"
--query-execution-context '{"Database": "default"}'
--result-configuration '{"OutputLocation": "s3://DOC-EXAMPLE-BUCKET/..."}'
```

Esecuzione di istruzioni preparate utilizzando il AWS CLI

Per eseguire un'istruzione preparata con AWS CLI, è possibile fornire valori per i parametri utilizzando uno dei seguenti metodi:

- Utilizzo dell'argomento `execution-parameters`.
- Utilizzo della sintassi SQL `EXECUTE ... USING` nell'argomento `query-string`.

Utilizzo dell'argomento execution-parameters

In questo caso, è necessario fornire il comando `start-query-execution` e fornire il nome di un'istruzione preparata esistente nell'argomento `query-string`. Poi, nell'argomento `execution-parameters` fornisci i valori per i parametri di esecuzione. Di seguito viene illustrato un esempio di policy che mostra questo approccio:

```
aws athena start-query-execution
--query-string "Execute PreparedStatement1"
--query-execution-context "Database"="default"
--result-configuration "OutputLocation"="s3://DOC-EXAMPLE-BUCKET/..."
--execution-parameters "1" "2"
```

Utilizzo di EXECUTE ... Utilizzo della sintassi SQL

Per eseguire un'istruzione preparata esistente utilizzando la sintassi `EXECUTE ... USING`, utilizza il comando `start-query-execution` e inserisci il nome dell'istruzione preparata e i valori dei parametri nell'argomento `query-string`, come nell'esempio:

```
aws athena start-query-execution
--query-string "EXECUTE PreparedStatement1 USING 1"
--query-execution-context '{"Database": "default"}'
--result-configuration '{"OutputLocation": "s3://DOC-EXAMPLE-BUCKET/..."}'
```

Come elencare istruzioni preparate

Per elencare le istruzioni preparate per un gruppo di lavoro specifico, puoi utilizzare il comando list-prepared-statements AWS CLI Athena o l'azione API ListPreparedStatementsAthena. Il parametro `--work-group` è obbligatorio.

```
aws athena list-prepared-statements --work-group primary
```

Risorse aggiuntive

Vedi i seguenti post correlati nel AWS Big Data Blog.

- Migliora la riusabilità e la sicurezza utilizzando le query parametrizzate di Amazon Athena
- Utilizzare le query parametrizzate di Amazon Athena per fornire dati come servizio

Utilizzo dell'ottimizzatore basato sui costi

Per ottimizzare le tue query, puoi utilizzare la funzionalità di ottimizzazione basata sui costi (CBO) di Athena SQL. Facoltativamente, puoi richiedere che Athena raccolga statistiche a livello di tabella o colonna per una delle tue tabelle in AWS Glue. Se tutte le tabelle della query contengono statistiche, Athena utilizza le statistiche per creare un piano di esecuzione che ritiene essere il più performante. L'ottimizzatore di query calcola i piani alternativi sulla base di un modello statistico e quindi seleziona quello che sarà probabilmente il più veloce per eseguire la query.

Le statistiche sulle AWS Glue tabelle vengono raccolte e archiviate in AWS Glue Data Catalog e rese disponibili ad Athena per migliorare la pianificazione e l'esecuzione delle query. Queste statistiche sono statistiche a livello di colonna, ad esempio il numero di valori distinti, il numero di valori nulli, massimi e minimi su tipi di file come Parquet, ORC, JSON, ION, CSV e XML. Amazon Athena

utilizza queste statistiche per ottimizzare le query applicando i filtri più restrittivi il prima possibile nell'elaborazione delle query. Questo filtro limita l'utilizzo della memoria e il numero di record da leggere per fornire i risultati delle query.

Oltre a CBO, Athena utilizza una funzionalità chiamata ottimizzatore basato su regole (RBO). L'RBO applica meccanicamente delle regole che dovrebbero migliorare le prestazioni delle query. L'RBO di solito è utile perché le sue trasformazioni mirano a semplificare il piano di interrogazione. Tuttavia, poiché l'RBO non esegue calcoli dei costi o confronti tra piani, le query più complicate rendono difficile la creazione di un piano ottimale.

Per questo motivo, per ottimizzare le query Athena utilizza sia RBO che CBO. Dopo aver identificato le opportunità per migliorare l'esecuzione delle query, Athena crea un piano ottimale. Per ulteriori informazioni sui dettagli del piano di esecuzione, consulta Visualizzazione dei piani di esecuzione per query SQL. Per una discussione dettagliata su come funziona CBO, consulta Accelerare le query con l'ottimizzatore basato sui costi di Amazon Athena nel blog Big Data. AWS

Per generare statistiche per le tabelle AWS Glue del catalogo, puoi utilizzare la console Athena, la AWS Glue Console o AWS Glue le API. Poiché Athena è integrato con AWS Glue Catalog, ottieni automaticamente i corrispondenti miglioramenti delle prestazioni delle query quando esegui query da Amazon Athena.

Considerazioni e limitazioni

- Tipi di tabelle: al momento, la funzionalità CBO di Athena supporta solo le tabelle Hive presenti nel AWS Glue Data Catalog.
- Athena per Spark: la funzionalità CBO non è disponibile in Athena per Spark.
- Prezzi: per informazioni sui prezzi, consulta la pagina dei prezzi di AWS Glue.

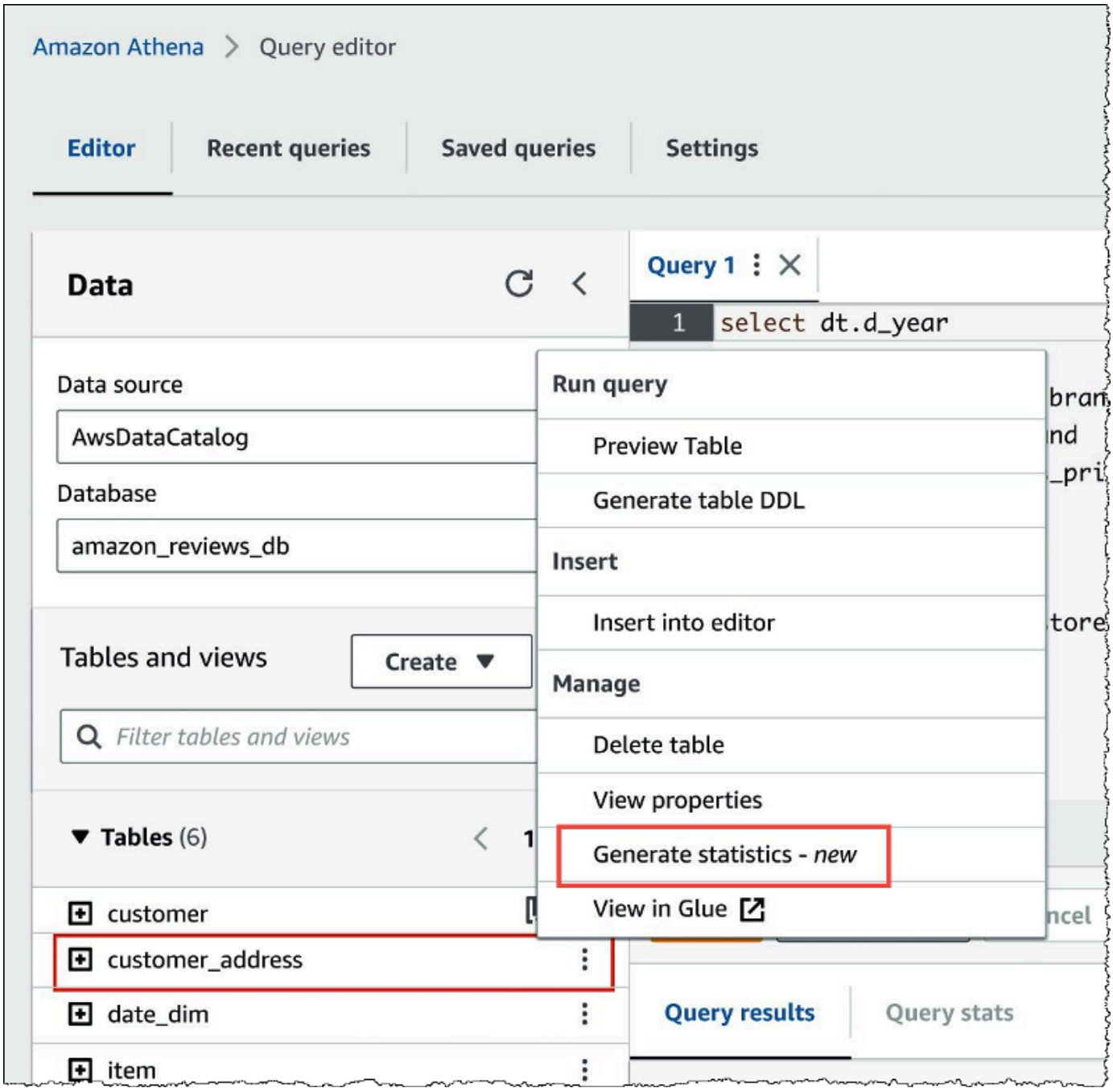
Generazione di statistiche sulle tabelle tramite la console Athena

In questa sezione viene descritto come usare la console Athena per generare statistiche a livello di tabella o colonna per una tabella in AWS Glue. Per informazioni sull'utilizzo per AWS Glue generare statistiche sulle tabelle, consulta Lavorare con le statistiche delle colonne nella Guida per gli AWS Glue sviluppatori.

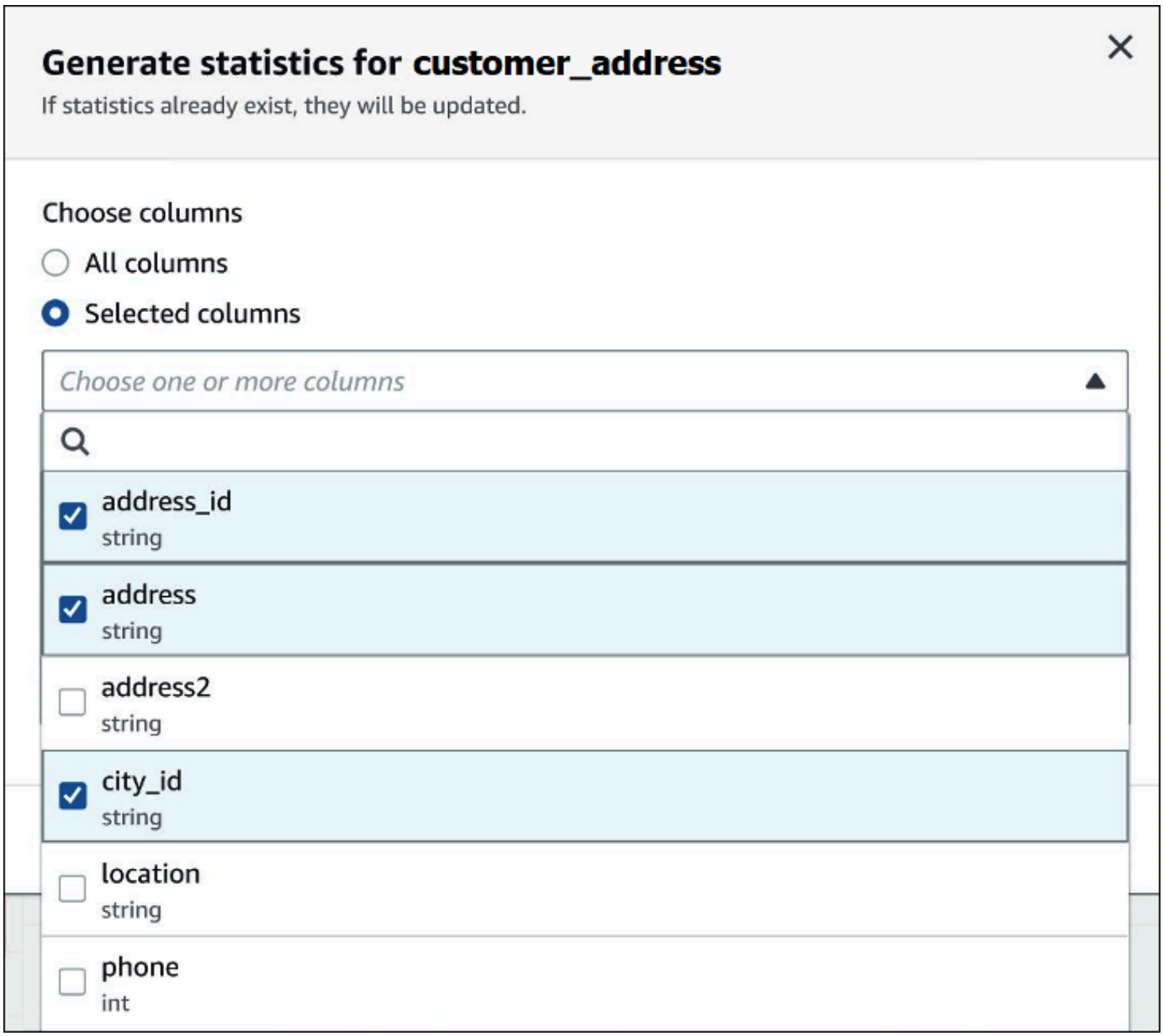
Generazione di statistiche per una tabella tramite la console Athena

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.

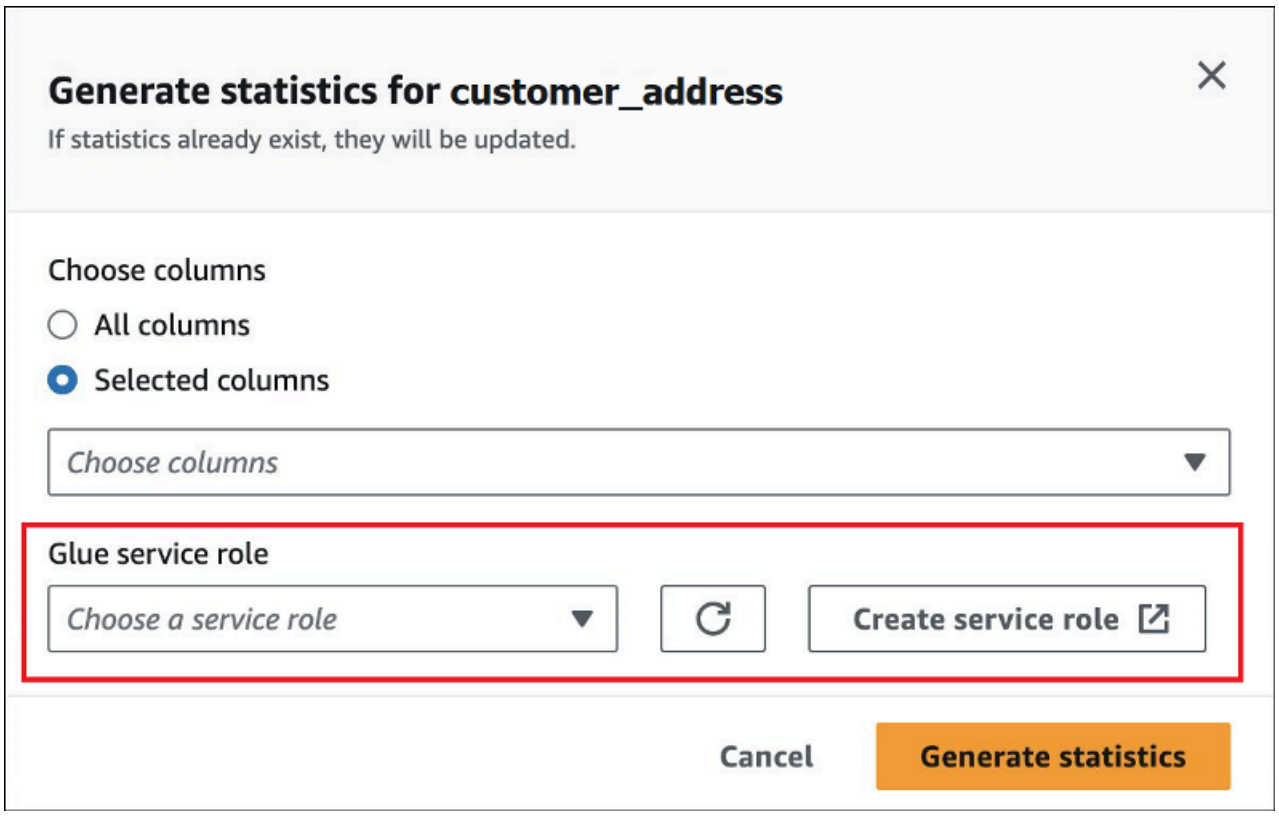
2. Nell'elenco Tabelle dell'editor di query Athena, scegli i tre punti verticali per la tabella desiderata, quindi scegli Genera statistiche.



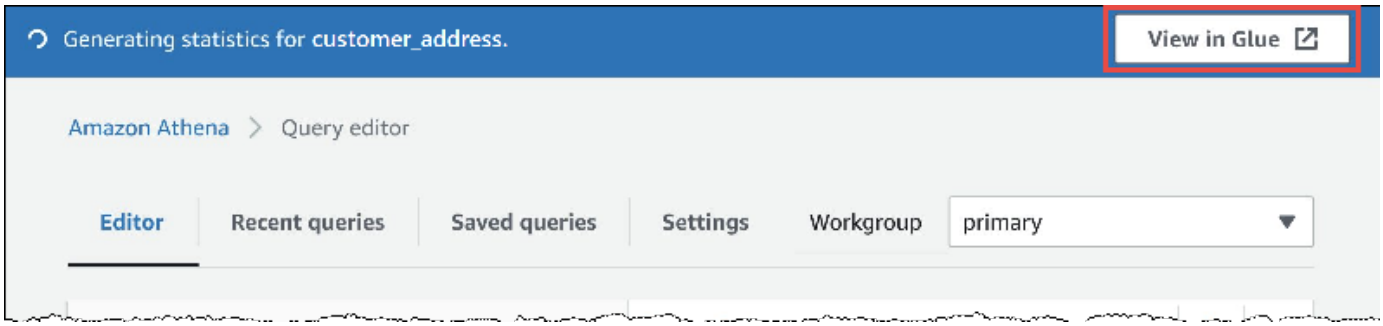
3. Nella finestra di dialogo Genera statistiche, scegli Tutte le colonne per generare le statistiche per tutte le colonne della tabella oppure Colonne selezionate per selezionare colonne specifiche. L'impostazione predefinita è Tutte le colonne.



4. Per il ruolo di AWS Glue servizio, crea o seleziona un ruolo di servizio esistente per AWS Glue autorizzare la generazione di statistiche. Il ruolo di servizio AWS Glue richiede anche le autorizzazioni `S3:GetObject` per il bucket Amazon S3 che contiene i dati della tabella.



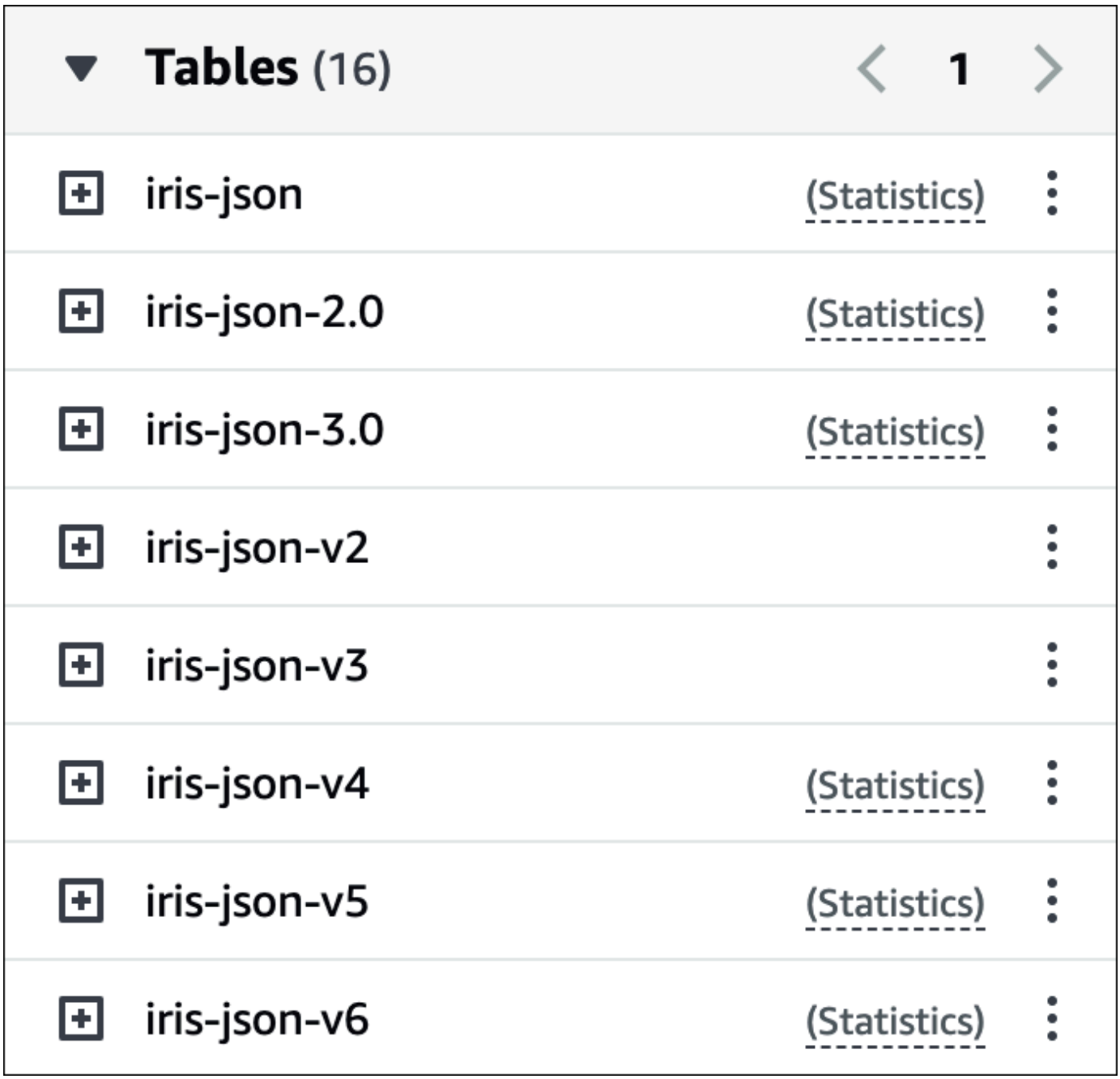
5. Scegli Genera statistiche. Un banner di notifica Generazione di statistiche per ***nome_tabella*** mostra lo stato dell'attività.



6. Per visualizzare i dettagli nella AWS Glue console, scegli Visualizza in Glue.

   Per informazioni sulla visualizzazione delle statistiche nella AWS Glue console, consulta Visualizzazione delle statistiche delle colonne nella Guida per gli AWS Glue sviluppatori.

7. Dopo la generazione delle statistiche, le tabelle e le colonne che contengono le statistiche mostrano la parola Statistiche tra parentesi, come nell'immagine seguente.



Ora, quando esegui le tue query, Athena eseguirà l'ottimizzazione in base ai costi delle tabelle e delle colonne per le quali sono state generate le statistiche.

Risorse aggiuntive

Per ulteriori informazioni, consulta la seguente risorsa.

Interrogazione dei dati di S3 Express One Zone

La classe di archiviazione Amazon S3 Express One Zone è una classe di archiviazione di Amazon S3 ad alte prestazioni che fornisce tempi di risposta nell'ordine dei millisecondi. Pertanto, è utile per le applicazioni che accedono frequentemente ai dati con centinaia di migliaia di richieste al secondo.

S3 Express One Zone replica e archivia i dati all'interno della stessa zona di disponibilità per ottimizzare velocità e costi. Ciò differisce dalle classi di storage regionali di Amazon S3, che replicano automaticamente i dati su almeno AWS tre zone di disponibilità all'interno di una. Regione AWS

Per ulteriori informazioni, consulta la pagina What is S3 Express One Zone? nella Guida per l'utente di Amazon S3.

Prerequisiti

Prima di iniziare, verifica che siano soddisfatte le seguenti condizioni:

- Versione 3 del motore Athena: per utilizzare S3 Express One Zone con Athena SQL, il gruppo di lavoro deve essere configurato per l'utilizzo della versione 3 del motore Athena.
- Autorizzazioni S3 Express One Zone: quando S3 Express One Zone richiama un'operazione come `GET`, `LIST` o `PUT` su un oggetto Amazon S3, la classe di archiviazione chiama `CreateSession` per tuo conto. Per questo motivo, la tua policy IAM deve consentire l'operazione `s3express:CreateSession`, che consente ad Athena di richiamare l'operazione dell'API corrispondente.

Considerazioni e limitazioni

Durante l'interrogazione di S3 Express One Zone con Athena, considera i seguenti punti.

- I bucket S3 Express One Zone supportano soltanto la crittografia `SSE_S3`. I risultati delle query Athena vengono scritti utilizzando la crittografia `SSE_S3` indipendentemente dall'opzione di crittografia dei risultati delle query selezionata nelle impostazioni del gruppo di lavoro. Questa limitazione include tutti gli scenari in cui Athena scrive dati su bucket S3 Express One Zone, incluse le istruzioni `CREATE TABLE AS` (CTAS) e `INSERT INTO`.
- Il AWS Glue crawler non è supportato per la creazione di tabelle sui dati di S3 Express One Zone.
- L'istruzione `MSCK REPAIR TABLE` non è supportata. Come soluzione alternativa, utilizza ALTER TABLE ADD PARTITION.

- `ALTER TABLE ADD PARTITIONALTER TABLE DROP PARTITION`, e non `ALTER TABLE RENAME PARTITION` sono supportati per le tabelle Iceberg in S3 Express One Zone.
- I seguenti formati di file e tabelle non sono supportati o hanno un supporto limitato. Se i formati, pur non essendo elencati, sono supportati per Athena (come Parquet, ORC e JSON), sono supportati anche per l'uso con l'archiviazione S3 Express One Zone.

Formato di file o tabella	Limitazione
Apache Avro	Non supportato
CloudTrail registri	Non supportato
Apache Hudi	Non supportato
Amazon Ion	Non supportato
Log di Logstash	Non supportato
registri di Apache WebServer	Non supportato
Delta Lake	DDL non supportato. Per informazioni sulla creazione di una tabella Delta Lake utilizzando uno schema fittizio, consulta la pagina Sincronizzazione dei metadati Delta Lake. Le query `SELECT` sulla tabella sono supportate.

Nozioni di base

L'interrogazione dei dati di S3 Express One Zone con Athena è semplice. Per iniziare, attieniti alla procedura seguente.

Utilizzo di Athena SQL per interrogare i dati di S3 Express One Zone

1. Trasferisci i tuoi dati all'archiviazione S3 Express One Zone. Per ulteriori informazioni, consulta la pagina Setting the storage class of an object nella Guida per l'utente di Amazon S3.

2. Utilizza un'istruzione CREATE TABLE in Athena per catalogare i tuoi dati in AWS Glue Data Catalog. Per ulteriori informazioni sulla creazione delle tabelle in Athena, consulta la pagina Creazione di tabelle in Athena e l'istruzione CREATE TABLE.
3. (Facoltativo) Configura la posizione dei risultati della query del tuo gruppo di lavoro Athena per utilizzare un bucket di directory Amazon S3. I bucket di directory Amazon S3 sono più performanti dei bucket generici e sono progettati per carichi di lavoro o applicazioni critiche per le prestazioni che richiedono una latenza costante nell'ordine dei millisecondi. Per ulteriori informazioni, consulta la pagina Directory buckets overview nella Guida per l'utente di Amazon S3.

Esecuzione di query su oggetti Amazon S3 Glacier ripristinati

Puoi usare Athena per eseguire query sugli oggetti ripristinati dalle classi di archiviazione di Amazon S3, S3 Glacier Flexible Retrieval (precedentemente Glacier) e S3 Glacier Deep Archive. È necessario abilitare questa funzionalità per ogni tabella. Se non abiliti la funzionalità su una tabella prima di eseguire una query, Athena salta tutti gli oggetti S3 Glacier Flexible Retrieval e S3 Glacier Deep Archive della tabella.

Considerazioni e limitazioni

- L'esecuzione di query di oggetti Amazon S3 Glacier ripristinati è supportata solo sulla versione 3 del motore Athena.
- La funzionalità è supportata solo per le tabelle Apache Hive.
- È necessario ripristinare gli oggetti prima di eseguire query sui dati; Athena non ripristina gli oggetti per conto dell'utente.

Configurazione di una tabella per utilizzare oggetti ripristinati

Per configurare la tabella Athena in modo che includa gli oggetti ripristinati nelle query, è necessario impostare la relativa proprietà di tabella `read_restored_glacier_objects` su `true`. Per fare ciò, puoi usare l'editor di query Athena o la AWS Glue console. Inoltre, puoi utilizzare la CLI di AWS Glue, le API di AWS Glue o l'SDK di AWS Glue.

Utilizzo dell'editor della query Athena

In Athena puoi utilizzare il comando ALTER TABLE SET TBLPROPERTIES per impostare la proprietà di tabella, come nell'esempio seguente.

```
ALTER TABLE table_name SET TBLPROPERTIES ('read_restored_glacier_objects' = 'true')
```

Utilizzo della console di AWS Glue

Nella AWS Glue console, effettuate le seguenti operazioni per aggiungere la proprietà della `read_restored_glacier_objects` tabella.

Per configurare le proprietà della tabella nella AWS Glue console

1. Accedere AWS Management Console e aprire la AWS Glue console all'indirizzo https://console.aws.amazon.com/glue/.
2. Esegui una di queste operazioni:
   - Seleziona Vai al catalogo dati.
   - Nel pannello di navigazione, seleziona Tabelle catalogo dati.
3. Nella pagine delle Tabelle, nell'elenco delle tabelle, seleziona il link della tabella che intendi modificare.
4. Scegli Actions (Operazioni), Edit (Modifica).
5. Nella pagina Modifica tabella, nella sezione Proprietà tabella, aggiungi la seguente coppia chiave-valore:
   - Per Chiave, aggiungere `read_restored_glacier_objects`.
   - In Valore, specifica `true`.
6. Selezionare Salva.

Utilizzando il AWS CLI

In AWS CLI, è possibile utilizzare il comando AWS Glue update-table e il relativo `--table-input` argomento per ridefinire la tabella e quindi aggiungere la proprietà. `read_restored_glacier_objects` Nell'argomento `--table-input` utilizza la struttura `Parameters` per specificare la proprietà `read_restored_glacier_objects` e il valore di `true`. Tieni presente che l'argomento per `--table-input` non deve avere spazi e deve utilizzare barre rovesciate per evitare le doppie virgolette. Nell'esempio seguente, sostituite *`my_database` e `my_table`con il nome del database e della tabella*.

```
aws glue update-table \
   --database-name my_database \
```

```
   --table-input={\"Name\":\"my_table\",\"Parameters\":{\"read_restored_glacier_objects
\":\"true\"}}
```

> **Important**
>
> Il AWS Glue `update-table` comando funziona in modalità sovrascrittura, il che significa che sostituisce la definizione della tabella esistente con la nuova definizione specificata dal parametro. `table-input` Per questo motivo, assicuratevi di specificare nel `table-input` parametro anche tutti i campi che desiderate inserire nella tabella quando aggiungete la `read_restored_glacier_objects` proprietà.

Gestione degli aggiornamenti degli schemi

Questa sezione fornisce delle linee guida su come gestire gli aggiornamenti degli schemi per vari formati di dati. Athena è un motore di schema-on-read interrogazione. Ciò significa che quando si crea una tabella in Athena, applica schemi durante la lettura dei dati. Non modifica né riscrive i dati sottostanti.

Se si prevedono delle modifiche agli schemi tabella, è consigliabile crearli in un formato di dati idoneo alle proprie esigenze. Il tuo obiettivo è riutilizzare query Athena esistenti anziché modificare gli schemi, evitando così errori di mancata corrispondenza tra gli schemi durante l'esecuzione di query su tabelle con partizioni.

Per raggiungere questi obiettivi, scegli un formato di dati della tabella in base alla tabella nell'argomento seguente.

Argomenti

Riepilogo: Aggiornamenti e formati di dati in Athena

La tabella seguente riepiloga i formati di storage dei dati e le relative manipolazioni supportate per gli schemi. Utilizza questa tabella per scegliere più facilmente il formato che ti consenta di continuare a utilizzare le query Athena anche man mano che i tuoi schemi cambiano nel corso del tempo.

In questa tabella, osserva che Parquet e ORC sono formati di colonna con diversi metodi di accesso predefiniti alle colonne. Per impostazione predefinita, Parquet accederà alle colonne in base al nome, mentre ORC in base all'indice (valore ordinale). Pertanto, Athena fornisce una SerDe proprietà definita durante la creazione di una tabella per attivare il metodo di accesso alle colonne predefinito che consente una maggiore flessibilità con l'evoluzione dello schema.

Per Parquet, la proprietà `parquet.column.index.access` può essere configurata su `true`, impostando così il metodo di accesso alle colonne in base al numero ordinale della colonna. Impostando questa proprietà su `false`, il metodo di accesso alle colonne cambierà, impiegando il nome della colonna. Analogamente, per ORC utilizza la proprietà `orc.column.index.access` per controllare il metodo di accesso alle colonne. Per ulteriori informazioni, consulta Accesso per indice in ORC e Parquet.

CSV e TSV consentono di eseguire qualsiasi manipolazione degli schemi, eccetto il riordinamento di colonne o l'aggiunta di colonne all'inizio della tabella. Ad esempio, se l'evoluzione del tuo schema richiede solo la ridenominazione delle colonne ma non la loro rimozione, è puoi decidere di creare le tabelle in CSV o TSV. Se è invece necessaria la rimozione di colonne, non utilizzare CSV o TSV, ma impiega invece uno qualsiasi degli altri formati supportati, preferibilmente un formato di colonna, ad esempio ORC o Parquet.

Aggiornamenti di schemi e formati di dati in Athena

Aggiornamento tipo di schema atteso	Riepilogo	CSV (con e senza intestazioni) e TSV	JSO	AVI	PARQUE lettura in base al nome (predefinito)	PARQUE lettura in base a indice	ORC: lettura in base a indice (predefinito)	ORC: lettura in base al nome
Rinomina colonne	Archivia i tuoi dati in CSV e TSV oppure in	Y	N	N	N	Y	Y	N

Aggiornamento tipo di schema atteso	Riepilogo	CSV (con e senza intestazioni) e TSV	JSO	AVF	PARQUE lettura in base al nome (predefinito)	PARQUE lettura in base a indice	ORC: lettura in base a indice (predefinito)	ORC: lettura in base al nome
	ORC e Parquet se vengono letti in base all'indice.							
Aggiungi colonne all'inizio o al centro della tabella	Archivia i tuoi dati in JSON e AVRO oppure in ORC e Parquet se vengono letti in base al nome. Non utilizzare CSV e TSV.	N	Y	Y	Y	N	N	Y
Aggiungi colonne alla fine di una tabella	Archivia i tuoi dati in CSV o TSV, JSON, AVRO, ORC o Parquet.	Y	Y	Y	Y	Y	Y	Y
Rimuovi colonne	Archivia i tuoi dati in JSON e AVRO oppure in Parquet e ORC se vengono letti in base al nome. Non utilizzare CSV e TSV.	N	Y	Y	Y	N	N	Y

Aggiornamento tipo di schema atteso	Riepilogo	CSV (con e senza intestazioni) e TSV	JSO	AVI	PARQUE lettura in base al nome (predefinito)	PARQUE lettura in base a indice	ORC: lettura in base a indice (predefinito)	ORC: lettura in base al nome
Riordina colonne	Archivia i tuoi dati in AVRO, JSON oppure in ORC e Parquet se vengono letti in base al nome.	N	Y	Y	Y	N	N	Y
Modifica il tipo di dati di una colonna	Archivia i dati in qualunque formato, ma testa la query in Athena per verificare che i tipi di dati siano compatibili. Per Parquet e ORC, la modifica di un tipo di dati funziona solo per tabelle partizionate.	Y	Y	Y	Y	Y	Y	Y

Accesso per indice in ORC e Parquet

PARQUET e ORC sono formati colonnari di storage dei dati che possono essere letti per indice o per nome. Archiviando i dati in uno di questi formati è possibile eseguire tutte le operazioni sugli schemi ed eseguire le query Athena senza errori di mancata corrispondenza di schemi.

- Athena legge ORC per indice per impostazione predefinita, come definito in `SERDEPROPERTIES ( 'orc.column.index.access'='true')`. Per ulteriori informazioni, consulta ORC: lettura in base all'indice.

- Athena legge Parquet per nome per impostazione predefinita, come definito in `SERDEPROPERTIES ( 'parquet.column.index.access'='false')`. Per ulteriori informazioni, consulta Parquet: lettura in base al nome.

Poiché si tratta di impostazioni predefinite, la specificazione di queste SerDe proprietà nelle `CREATE TABLE` query è facoltativa, in quanto vengono utilizzate implicitamente. Quando sono utilizzate, consentono di eseguire alcune operazioni di aggiornamento dello schema, impedendone altre. Per abilitare tali operazioni, esegui un'altra `CREATE TABLE` query e modifica le impostazioni. SerDe

> **Note**
>
> Le SerDe proprietà non vengono propagate automaticamente a ciascuna partizione. `ALTER TABLE ADD PARTITIONUtilizzate` le istruzioni per impostare le SerDe proprietà per ogni partizione. Per automatizzare il processo, scrivi uno script che esegue le istruzioni `ALTER TABLE ADD PARTITION`.

Le seguenti sezioni descrivono in dettaglio questi casi.

ORC: lettura in base all'indice

Una tabella in ORC viene letta in base all'indice per impostazione predefinita. Questo è definito dalla seguente sintassi:

```
WITH SERDEPROPERTIES (
  'orc.column.index.access'='true')
```

La lettura per indice consente l'assegnazione di un nuovo nome alle colonne. Tuttavia, in tal caso non sarà più possibile rimuovere colonne o aggiungerle al centro della tabella.

Per fare in modo che ORC venga letto per nome, il che ti consentirà di aggiungere colonne al centro della tabella o rimuovere colonne in ORC, imposta la SerDe proprietà su `false` nell'`orc.column.index.access`istruzione. `CREATE TABLE` In questa configurazione, perderai la possibilità di rinominare le colonne.

> **Note**
>
> Nel motore di Athena versione 2, quando le tabelle ORC sono impostate per la lettura per nome, Athena richiede che tutti i nomi delle colonne nei file ORC siano minuscoli. Poiché

Apache Spark non utilizza nomi di campo minuscoli quando genera file ORC, Athena potrebbe non essere in grado di leggere i dati così generati. La soluzione alternativa consiste nel rinominare le colonne in modo che siano in minuscolo, oppure nell'utilizzare la versione 3 del motore Athena.

L'esempio seguente spiega come modificare la lettura ORC in modo che avvenga in base al nome:

```
CREATE EXTERNAL TABLE orders_orc_read_by_name (
   `o_comment` string,
   `o_orderkey` int,
   `o_custkey` int,
   `o_orderpriority` string,
   `o_orderstatus` string,
   `o_clerk` string,
   `o_shippriority` int,
   `o_orderdate` string
)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.ql.io.orc.OrcSerde'
WITH SERDEPROPERTIES (
  'orc.column.index.access'='false')
STORED AS INPUTFORMAT
  'org.apache.hadoop.hive.ql.io.orc.OrcInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.orc.OrcOutputFormat'
LOCATION 's3://DOC-EXAMPLE-BUCKET/orders_orc/';
```

Parquet: lettura in base al nome

Una tabella in Parquet viene letta in base al nome per impostazione predefinita. Questo è definito dalla seguente sintassi:

```
WITH SERDEPROPERTIES (
  'parquet.column.index.access'='false')
```

La lettura per nome consente di aggiungere colonne al centro della tabella e di rimuovere colonne. Tuttavia, in tal caso si non sarà più possibile rinominare le colonne.

Per fare in modo che Parquet venga letto per indice, che consentirà di rinominare le colonne, è necessario creare una tabella con la `parquet.column.index.access` SerDe proprietà impostata su. `true`

Tipi di aggiornamenti

Questo argomento descrive alcune delle modifiche che è possibile apportare allo schema nelle istruzioni `CREATE TABLE` senza alterare effettivamente i dati. Verifichiamo ogni tipo di aggiornamento degli schemi e specifichiamo quali formati di dati consentono di averli in Athena. Per aggiornare uno schema, in alcuni casi puoi utilizzare un comando `ALTER TABLE`, ma in altri casi non è possibile modificare effettivamente una tabella esistente. Crea invece una tabella con un nuovo nome che modifica lo schema utilizzato nell'istruzione originale `CREATE TABLE`.

- Aggiunta di colonne all'inizio o al centro della tabella
- Aggiunta di colonne alla fine della tabella
- Rimozione di colonne
- Assegnazione di un nuovo nome alle colonne
- Riordinamento di colonne
- Modifica del tipo di dati di una colonna

A seconda dell'evoluzione prevista degli schemi, per continuare a utilizzare le query Athena, è necessario scegliere un formato di dati compatibile.

Considera un'applicazione che legge le informazioni sugli ordini da una tabella `orders` esistente in due formati: CSV e Parquet.

L'esempio seguente crea una tabella in formato Parquet:

```
CREATE EXTERNAL TABLE orders_parquet (
   `orderkey` int,
   `orderstatus` string,
   `totalprice` double,
   `orderdate` string,
   `orderpriority` string,
   `clerk` string,
   `shippriority` int
) STORED AS PARQUET
LOCATION 's3://DOC-EXAMPLE-BUCKET/orders_ parquet/';
```

L'esempio seguente crea la stessa tabella in formato CSV:

```
CREATE EXTERNAL TABLE orders_csv (
   `orderkey` int,
   `orderstatus` string,
   `totalprice` double,
   `orderdate` string,
   `orderpriority` string,
   `clerk` string,
   `shippriority` int
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION 's3://DOC-EXAMPLE-BUCKET/orders_csv/';
```

Nelle sezioni seguenti, analizziamo il modo in cui gli aggiornamenti di queste tabelle influenzano le query Athena.

Aggiunta di colonne all'inizio o al centro della tabella

L'aggiunta di colonne è una delle modifiche più frequenti di uno schema. Ad esempio, è possibile aggiungere una nuova colonna per arricchire la tabella con nuovi dati. In alternativa, è possibile aggiungere una nuova colonna se è cambiata la fonte di una colonna esistente e mantenere la versione precedente di questa colonna per regolare le applicazioni che dipendono da essa.

Per aggiungere colonne all'inizio o al centro della tabella e continuare a eseguire query su tabelle esistenti, utilizzate AVRO, JSON e Parquet e ORC se la loro SerDe proprietà è impostata su «Leggi per nome». Per informazioni, consulta Accesso per indice in ORC e Parquet.

Non aggiungere colonne all'inizio o al centro della tabella in CSV e TSV, poiché questi formati dipendono dall'ordinamento. L'aggiunta di una colonna in questi casi porterebbe a errori di mancata corrispondenza tra schemi quando lo schema di partizioni viene modificato.

L'esempio seguente crea una nuova tabella che aggiunge una colonna `o_comment` al centro di una tabella basata su dati JSON.

```
CREATE EXTERNAL TABLE orders_json_column_addition (
   `o_orderkey` int,
   `o_custkey` int,
   `o_orderstatus` string,
   `o_comment` string,
   `o_totalprice` double,
```

```
    `o_orderdate` string,
    `o_orderpriority` string,
    `o_clerk` string,
    `o_shippriority` int,
)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
LOCATION 's3://DOC-EXAMPLE-BUCKET/orders_json/';
```

Aggiunta di colonne alla fine della tabella

Se si creano tabelle in uno dei formati supportati da Athena, ad esempio Parquet, ORC, Avro, JSON, CSV e TSV, è possibile utilizzare l'istruzione `ALTER TABLE ADD COLUMNS` per aggiungere colonne dopo le colonne esistenti ma prima delle colonne di partizione.

Nell'esempio seguente viene aggiunta una colonna `comment` alla fine della tabella `orders_parquet` prima di qualsiasi colonna di partizione:

```
ALTER TABLE orders_parquet ADD COLUMNS (comment string)
```

> **Note**
>
> Per visualizzare una nuova colonna di tabella nell'editor di query Athena dopo l'esecuzione di `ALTER TABLE ADD COLUMNS`, aggiornare manualmente l'elenco di tabelle nell'editor e quindi espandere nuovamente la tabella.

Rimozione di colonne

Potrebbe essere necessario rimuovere colonne dalle tabelle se queste non contengono più dati o per limitare l'accesso ai dati in esse contenuti.

- È possibile rimuovere colonne da tabelle in formato JSON, Avro e Parquet e ORC se la lettura è per nome. Per informazioni, consulta Accesso per indice in ORC e Parquet.
- Sconsigliamo di rimuovere colonne delle tabelle in CSV e TSV se si desidera conservare le tabelle già create in Athena. La rimozione di una colonna altera lo schema e richiede di ricreare la tabella senza la colonna rimossa.

In questo esempio, si rimuove una colonna `` `totalprice` `` da una tabella in Parquet e si esegue una query. In Athena, il formato Parquet è letto per nome per impostazione predefinita e questo è il

motivo per cui omettiamo la configurazione SERDEPROPERTIES che specifica la lettura per nome. Tieni presente che la seguente query va a buon fine anche se si modifica lo schema:

```
CREATE EXTERNAL TABLE orders_parquet_column_removed (
   `o_orderkey` int,
   `o_custkey` int,
   `o_orderstatus` string,
   `o_orderdate` string,
   `o_orderpriority` string,
   `o_clerk` string,
   `o_shippriority` int,
   `o_comment` string
)
STORED AS PARQUET
LOCATION 's3://DOC-EXAMPLE-BUCKET/orders_parquet/';
```

Assegnazione di un nuovo nome alle colonne

È possibile rinominare le colonne nelle tabelle per correggere l'ortografia, rendere i nomi delle colonne più descrittivi o riutilizzare una colonna esistente per evitare il riordinamento delle colonne.

È possibile rinominare le colonne se si archiviano i dati in CSV e TSV o in Parquet e ORC con configurazione per la lettura per indice. Per informazioni, consulta Accesso per indice in ORC e Parquet.

Athena legge i dati in CSV e TSV nell'ordine delle colonne nello schema e li restituisce nello stesso ordine. Non utilizza i nomi delle colonne per la mappatura dei dati a una colonna e questo è il motivo per cui è possibile rinominare le colonne in CSV o TSV senza interrompere le query Athena.

Una strategia per rinominare le colonne consiste nel creare una nuova tabella basata sugli stessi dati sottostanti, ma utilizzando nuovi nomi di colonna. Il seguente esempio crea una nuova tabella `orders_parquet` chiamata `orders_parquet_column_renamed`. L'esempio modifica il nome `` `o_totalprice` `` della colonna in `` `o_total_price` `` e quindi esegue una query in Athena:

```
CREATE EXTERNAL TABLE orders_parquet_column_renamed (
   `o_orderkey` int,
   `o_custkey` int,
   `o_orderstatus` string,
   `o_total_price` double,
   `o_orderdate` string,
   `o_orderpriority` string,
   `o_clerk` string,
```

```
   `o_shippriority` int,
   `o_comment` string
)
STORED AS PARQUET
LOCATION 's3://DOC-EXAMPLE-BUCKET/orders_parquet/';
```

Nel caso della tabella Parquet, la query seguente viene eseguita, ma la colonna rinominata non mostra i dati poiché l'accesso alla stessa è avvenuto per nome (impostazione predefinita in Parquet) anziché per indice:

```
SELECT *
FROM orders_parquet_column_renamed;
```

Di seguito un esempio di query con una tabella in CSV:

```
CREATE EXTERNAL TABLE orders_csv_column_renamed (
   `o_orderkey` int,
   `o_custkey` int,
   `o_orderstatus` string,
   `o_total_price` double,
   `o_orderdate` string,
   `o_orderpriority` string,
   `o_clerk` string,
   `o_shippriority` int,
   `o_comment` string
)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
LOCATION 's3://DOC-EXAMPLE-BUCKET/orders_csv/';
```

Nel caso della tabella CSV, la query seguente viene eseguita e i dati vengono visualizzati in tutte le colonne, compresa quella che è stata rinominata:

```
SELECT *
FROM orders_csv_column_renamed;
```

Riordinamento di colonne

È possibile riordinare le colonne solo per tabelle con dati in formati che leggono per nome, ad esempio JSON o Parquet, i quali leggono per nome per impostazione predefinita. Se necessario, è possibile fare in modo che anche ORC legga per nome. Per informazioni, consulta Accesso per indice in ORC e Parquet.

L'esempio seguente crea una nuova tabella con le colonne in un ordine diverso:

```
CREATE EXTERNAL TABLE orders_parquet_columns_reordered (
   `o_comment` string,
   `o_orderkey` int,
   `o_custkey` int,
   `o_orderpriority` string,
   `o_orderstatus` string,
   `o_clerk` string,
   `o_shippriority` int,
   `o_orderdate` string
)
STORED AS PARQUET
LOCATION 's3://DOC-EXAMPLE-BUCKET/orders_parquet/';
```

Modifica del tipo di dati di una colonna

Potresti voler utilizzare un tipo di colonna diverso quando il tipo esistente non può più contenere la quantità di informazioni richieste. Ad esempio, i valori di una colonna ID potrebbero superare la dimensione del tipo di dati `INT` e richiedere l'uso del tipo di dati `BIGINT`.

Quando si prevede di utilizzare un tipo di dati diverso per una colonna, considerare i seguenti punti:

- Nella maggior parte dei casi, non è possibile modificare direttamente il tipo di dati di una colonna. Al contrario, si ricrea la tabella Athena e si definisce la colonna con il nuovo tipo di dati.
- Solo alcuni tipi di dati possono essere letti come altri tipi di dati. Consulta la tabella in questa sezione per i tipi di dati che possono essere trattati in tal modo.
- Per i dati in Parquet e ORC, non puoi utilizzare un tipo di dati diverso per una colonna se la tabella non è partizionata.
- Per le tabelle partizionate in Parquet e ORC, il tipo di colonna di una partizione può essere diverso dal tipo di colonna di un'altra partizione e, se possibile, Athena eseguirà `CAST` per il tipo desiderato. Per informazioni, consulta Come evitare errori di mancata corrispondenza tra schemi per tabelle con partizioni.
- Per le tabelle create utilizzando l'LazySimpleSerDeunico, è possibile utilizzare l'`ALTER TABLE REPLACE COLUMNS`istruzione per sostituire le colonne esistenti con un tipo di dati diverso, ma tutte le colonne esistenti che si desidera conservare devono essere ridefinite nell'istruzione, altrimenti verranno eliminate. Per ulteriori informazioni, consulta ALTER TABLE REPLACE COLUMNS.
- Solo per le tabelle Apache Iceberg, puoi utilizzare l'istruzione ALTER TABLE CHANGE COLUMN per modificare il tipo di dati di una colonna. `ALTER TABLE REPLACE COLUMNS` non è supportato

per le tabelle Iceberg. Per ulteriori informazioni, consulta Schema della tabella Iceberg in evoluzione.

> **Important**
>
> Consigliamo vivamente di testare e verificare le query prima di modificare il tipo di dati. Se Athena non è in grado di utilizzare il tipo di dati di destinazione, la query `CREATE TABLE` potrebbe avere esito negativo.

La tabella seguente elenca i tipi di dati che devono essere trattati come altri tipi di dati:

Tipi di dati compatibili

Tipo di dati originale	Tipi di dati di destinazione disponibili
`STRING`	`BYTE`, `TINYINT`, `SMALLINT`, `INT`, `BIGINT`
`BYTE`	`TINYINT`, `SMALLINT`, `INT`, `BIGINT`
`TINYINT`	`SMALLINT`, `INT`, `BIGINT`
`SMALLINT`	`INT`, `BIGINT`
`INT`	`BIGINT`
`FLOAT`	`DOUBLE`

L'esempio seguente mostra l'utilizzo dell'istruzione `CREATE TABLE` per la tabella `orders_json` originale per creare una nuova tabella denominata `orders_json_bigint`. La nuova tabella utilizza `BIGINT` anziché `INT` come tipo di dati per la colonna `` `o_shippriority` ``.

```
CREATE EXTERNAL TABLE orders_json_bigint (
   `o_orderkey` int,
   `o_custkey` int,
   `o_orderstatus` string,
   `o_totalprice` double,
   `o_orderdate` string,
   `o_orderpriority` string,
   `o_clerk` string,
```

```
   `o_shippriority` BIGINT
)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
LOCATION 's3://DOC-EXAMPLE-BUCKET/orders_json';
```

La seguente query viene eseguita correttamente, in modo analogo alla query SELECT originale, prima della modifica del tipo di dati:

```
Select * from orders_json
LIMIT 10;
```

Aggiornamenti in tabelle con partizioni

In Athena, una tabella e le relative partizioni devono utilizzare gli stessi formati di dati, tuttavia i loro schemi possono differire. Quando si crea una nuova partizione, questa in genere eredita lo schema della tabella. Nel corso del tempo, gli schemi possono iniziare a differire. I motivi includono:

- Se lo schema della tua tabella cambia, gli schemi delle partizioni non vengono aggiornati per mantenere la sincronizzazione con lo schema della tabella.
- Il AWS Glue Crawler consente di scoprire dati in partizioni con schemi diversi. Ciò significa che se si crea una tabella in Athena con AWS Glue, dopo che il crawler ha terminato l'elaborazione, gli schemi per la tabella e le relative partizioni potrebbero essere diversi.
- Se aggiungi partizioni direttamente utilizzando un'API. AWS

Athena elabora correttamente le tabelle con partizioni se soddisfano i seguenti vincoli. Se tali requisiti non vengono soddisfatti, Athena emette un errore HIVE_PARTITION_SCHEMA_MISMATCH.

- Ciascuno schema di partizione è compatibile con lo schema della tabella.
- Il formato di dati della tabella consente il tipo di aggiornamento che si desidera eseguire: aggiunta, eliminazione, riordinamento o modifica del tipo di dati di una colonna.

  Ad esempio, per i formati CSV e TSV è possibile rinominare colonne, aggiungere nuove colonne alla fine della tabella e modificare il tipo di dati di una colonna se i tipi sono compatibili, ma non è possibile rimuovere colonne. Per altri formati, è possibile aggiungere o rimuovere colonne oppure modificare il tipo di dati di una colonna in un altro tipo se i tipi sono compatibili. Per ulteriori informazioni, consulta il Riepilogo: aggiornamenti e formati di dati in Athena.

Come evitare errori di mancata corrispondenza tra schemi per tabelle con partizioni

All'inizio dell'esecuzione di una query, Athena verifica lo schema della tabella controllando che ogni tipo di dati colonna sia compatibile tra la tabella e la partizione.

- Per i tipi di storage dei dati Parquet e ORC, Athena si avvale dei nomi delle colonne e li utilizza per la verifica del proprio schema basato sul nome di colonna. Ciò consente di eliminare gli errori `HIVE_PARTITION_SCHEMA_MISMATCH` per le tabelle con partizioni in Parquet e ORC. (Questo vale per ORC se la SerDe proprietà è impostata per accedere all'indice per nome:. `orc.column.index.access=FALSE` Per impostazione predefinita, Parquet legge l'indice per nome).
- Per CSV, JSON e Avro, Athena usa una verifica dello schema basata sull'indice. Questo significa che, se si verifica un errore di mancata corrispondenza di schema, è necessario eliminare la partizione che provoca tale differenza e ricrearla, in modo che Athena possa eseguire le query senza errori.

Athena confronta lo schema della tabella con gli schemi della partizione. Se crei una tabella in CSV, JSON e AVRO in Athena con AWS Glue Crawler, dopo che il Crawler ha terminato l'elaborazione, gli schemi per la tabella e le sue partizioni potrebbero essere diversi. Se non c'è corrispondenza tra lo schema della tabella e gli schemi delle partizioni, le query falliranno in Athena a causa dell'errore di verifica dello schema simile a questo: 'crawler_test.click_avro' è dichiarato come tipo "string", ma la partizione "partition_0 = 2017-01-17" ha dichiarato la colonna "col68" come tipo "doppio"'.

Una tipica soluzione per questi errori è eliminare la partizione che provoca l'errore e ricrearla. Per ulteriori informazioni, consultare ALTER TABLE DROP PARTITION e ALTER TABLE ADD PARTITION.

Esecuzione di query sulle matrici

Amazon Athena consente di creare matrici, concatenarle, convertirle in diversi tipi di dati e in seguito filtrarle, appiattirle e ordinarle.

Argomenti

- Creazione di matrici
- Concatenazione di stringhe e matrici
- Conversione di tipi di dati della matrice
- Individuazione di lunghezze

- Accesso agli elementi della matrice
- Appiattimento di matrici nidificate
- Creazione di matrici da sottoquery
- Filtraggio delle matrici
- Ordinamento matrici
- Utilizzo delle funzioni di aggregazione con le matrici
- Conversione delle matrici in stringhe
- Utilizzo delle matrici per creare mappe
- Esecuzione di query su matrici con tipi complessi e strutture nidificate

Creazione di matrici

Per creare una matrice letterale in Athena, utilizzare la parola chiave ARRAY, seguita da parentesi `[ ]` e includere gli elementi della matrice separati da virgole.

Esempi

Questa query crea una matrice con quattro elementi.

```
SELECT ARRAY [1,2,3,4] AS items
```

Restituisce:

```
+-----------+
| items     |
+-----------+
| [1,2,3,4] |
+-----------+
```

Questa query crea due matrici.

```
SELECT ARRAY[ ARRAY[1,2], ARRAY[3,4] ] AS items
```

Restituisce:

```
+--------------------+
```

```
| items              |
+--------------------+
| [[1, 2], [3, 4]]   |
+--------------------+
```

Per creare una matrice da colonne selezionate di tipi compatibili, utilizzare una query, come in questo esempio:

```
WITH
dataset AS (
  SELECT 1 AS x, 2 AS y, 3 AS z
)
SELECT ARRAY [x,y,z] AS items FROM dataset
```

Questa query restituisce:

```
+-----------+
| items     |
+-----------+
| [1,2,3]   |
+-----------+
```

In questo esempio, due matrici vengono selezionate e restituite come un messaggio di benvenuto.

```
WITH
dataset AS (
  SELECT
    ARRAY ['hello', 'amazon', 'athena'] AS words,
    ARRAY ['hi', 'alexa'] AS alexa
)
SELECT ARRAY[words, alexa] AS welcome_msg
FROM dataset
```

Questa query restituisce:

```
+----------------------------------------+
| welcome_msg                            |
+----------------------------------------+
| [[hello, amazon, athena], [hi, alexa]] |
+----------------------------------------+
```

Per creare una matrice di coppie chiave-valore, utilizzare l'operatore MAP che richiede una matrice di chiavi seguite da una matrice di valori, come in questo esempio:

```
SELECT ARRAY[
   MAP(ARRAY['first', 'last', 'age'],ARRAY['Bob', 'Smith', '40']),
   MAP(ARRAY['first', 'last', 'age'],ARRAY['Jane', 'Doe', '30']),
   MAP(ARRAY['first', 'last', 'age'],ARRAY['Billy', 'Smith', '8'])
] AS people
```

Questa query restituisce:

```
+-----------------------------------------------------------------------------------------------
+
| people
               |
+-----------------------------------------------------------------------------------------------
+
| [{last=Smith, first=Bob, age=40}, {last=Doe, first=Jane, age=30}, {last=Smith,
 first=Billy, age=8}] |
+-----------------------------------------------------------------------------------------------
+
```

Concatenazione di stringhe e matrici

Concatenazione di stringhe

Per concatenare due stringhe, è possibile utilizzare l'operatore doppio pipe `||`, come nell'esempio seguente.

```
SELECT 'This' || ' is' || ' a' || ' test.' AS Concatenated_String
```

Questa query restituisce:

#	Concatenated_String
1	This is a test.

Puoi utilizzare la funzione `concat()` per ottenere lo stesso risultato.

```
SELECT concat('This', ' is', ' a', ' test.') AS Concatenated_String
```

Questa query restituisce:

#	Concatenated_String
1	This is a test.

È possibile utilizzare la funzione `concat_ws()` per concatenare stringhe con il separatore specificato nel primo argomento.

```
SELECT concat_ws(' ', 'This', 'is', 'a', 'test.') as Concatenated_String
```

Questa query restituisce:

#	Concatenated_String
1	This is a test.

Per concatenare due colonne del tipo di dati stringa utilizzando un punto, fai riferimento alle due colonne utilizzando virgolette doppie e racchiudi il punto tra virgolette singole come stringa codificata. Se una colonna non è del tipo di dati stringa, puoi utilizzare prima `CAST("`*`column_name`*`" as VARCHAR)` per eseguire il cast della colonna.

```
SELECT "col1" || '.' || "col2" as Concatenated_String
FROM my_table
```

Questa query restituisce:

#	Concatenated_String
1	*col1_string_value* .*col2_string_value*

Concatenazione di matrici

È possibile utilizzare le stesse tecniche per concatenare le matrici.

Per concatenare più matrici, utilizza l'operatore doppio pipe `||`.

```
SELECT ARRAY [4,5] || ARRAY[ ARRAY[1,2], ARRAY[3,4] ] AS items
```

Questa query restituisce:

#	items
1	[[4, 5], [1, 2], [3, 4]]

Per combinare più array in un solo array, utilizza l'operatore doppio pipe o la funzione `concat()`.

```
WITH
dataset AS (
  SELECT
    ARRAY ['Hello', 'Amazon', 'Athena'] AS words,
    ARRAY ['Hi', 'Alexa'] AS alexa
)
SELECT concat(words, alexa) AS welcome_msg
FROM dataset
```

Questa query restituisce:

#	welcome_msg
1	[Hello, Amazon, Athena, Hi, Alexa]

Per ulteriori informazioni sulle funzioni `concat()` di altre stringhe, consulta Funzioni e operatori per le stringhe nella documentazione di Trino.

Conversione di tipi di dati della matrice

Per convertire i dati delle matrici in tipi di dati supportati, utilizza l'operatore `CAST`, come `CAST(value AS type)`. Athena supporta tutti i tipi di dati nativi di Presto.

```
SELECT
   ARRAY [CAST(4 AS VARCHAR), CAST(5 AS VARCHAR)]
```

```
AS items
```

Questa query restituisce:

```
+-------+
| items |
+-------+
| [4,5] |
+-------+
```

Creare due matrici con elementi della coppia chiave-valore, convertirli in formato JSON e concatenarli, come nel seguente esempio:

```
SELECT
   ARRAY[CAST(MAP(ARRAY['a1', 'a2', 'a3'], ARRAY[1, 2, 3]) AS JSON)] ||
   ARRAY[CAST(MAP(ARRAY['b1', 'b2', 'b3'], ARRAY[4, 5, 6]) AS JSON)]
AS items
```

Questa query restituisce:

```
+-------------------------------------------------+
| items                                           |
+-------------------------------------------------+
| [{"a1":1,"a2":2,"a3":3}, {"b1":4,"b2":5,"b3":6}] |
+-------------------------------------------------+
```

Individuazione di lunghezze

La funzione `cardinality` restituisce la lunghezza di un array, come in questo esempio:

```
SELECT cardinality(ARRAY[1,2,3,4]) AS item_count
```

Questa query restituisce:

```
+------------+
| item_count |
+------------+
| 4          |
+------------+
```

Accesso agli elementi della matrice

Per accedere agli elementi della matrice, utilizza l'operatore `[]`, dove 1 specifica il primo elemento, 2 specifica il secondo e così via, come in questo esempio:

```
WITH dataset AS (
SELECT
   ARRAY[CAST(MAP(ARRAY['a1', 'a2', 'a3'], ARRAY[1, 2, 3]) AS JSON)] ||
   ARRAY[CAST(MAP(ARRAY['b1', 'b2', 'b3'], ARRAY[4, 5, 6]) AS JSON)]
AS items )
SELECT items[1] AS item FROM dataset
```

Questa query restituisce:

```
+------------------------+
| item                   |
+------------------------+
| {"a1":1,"a2":2,"a3":3} |
+------------------------+
```

Per accedere agli elementi di una matrice in una determinata posizione (nota come posizione indice), utilizza la funzione `element_at()` e specifica il nome della matrice e la posizione indice:

- Se l'indice è maggiore di 0, `element_at()` restituisce l'elemento specificato, contando dall'inizio alla fine della matrice. Si comporta come l'operatore `[]`.
- Se l'indice è minore di 0, `element_at()` restituisce l'elemento, contando dalla fine all'inizio della matrice.

La seguente query crea una matrice `words` e seleziona il primo elemento `hello` come `first_word`, il secondo elemento `amazon` (contando dalla fine della matrice) come `middle_word` e il terzo elemento `athena` come `last_word`.

```
WITH dataset AS (
  SELECT ARRAY ['hello', 'amazon', 'athena'] AS words
)
SELECT
  element_at(words, 1) AS first_word,
  element_at(words, -2) AS middle_word,
  element_at(words, cardinality(words)) AS last_word
```

```
FROM dataset
```

Questa query restituisce:

```
+---------------------------------------+
| first_word  | middle_word | last_word  |
+---------------------------------------+
| hello       | amazon      | athena     |
+---------------------------------------+
```

Appiattimento di matrici nidificate

Quando si utilizzano matrici nidificate, spesso è necessario espanderne gli elementi in una singola matrice oppure espandere la matrice stessa in più righe.

Esempi

Per appiattire gli elementi di una matrice nidificata in una singola matrice di valori, utilizza la funzione `flatten`. Questa query restituisce una riga per ciascun elemento della matrice.

```
SELECT flatten(ARRAY[ ARRAY[1,2], ARRAY[3,4] ]) AS items
```

Questa query restituisce:

```
+-----------+
| items     |
+-----------+
| [1,2,3,4] |
+-----------+
```

Per appiattire una matrice in più righe, utilizza `CROSS JOIN` in combinazione con l'operatore `UNNEST`, come in questo esempio:

```
WITH dataset AS (
  SELECT
    'engineering' as department,
    ARRAY['Sharon', 'John', 'Bob', 'Sally'] as users
)
SELECT department, names FROM dataset
```

```
CROSS JOIN UNNEST(users) as t(names)
```

Questa query restituisce:

```
+----------------------+
| department  | names  |
+----------------------+
| engineering | Sharon |
+----------------------|
| engineering | John   |
+----------------------|
| engineering | Bob    |
+----------------------|
| engineering | Sally  |
+----------------------+
```

Per appiattire una matrice di coppie chiave-valore, ridisponi le chiavi selezionate in colonne, come in questo esempio:

```
WITH
dataset AS (
  SELECT
    'engineering' as department,
     ARRAY[
      MAP(ARRAY['first', 'last', 'age'],ARRAY['Bob', 'Smith', '40']),
      MAP(ARRAY['first', 'last', 'age'],ARRAY['Jane', 'Doe', '30']),
      MAP(ARRAY['first', 'last', 'age'],ARRAY['Billy', 'Smith', '8'])
     ] AS people
  )
SELECT names['first'] AS
 first_name,
 names['last'] AS last_name,
 department FROM dataset
CROSS JOIN UNNEST(people) AS t(names)
```

Questa query restituisce:

```
+--------------------------------------+
| first_name | last_name | department  |
+--------------------------------------+
| Bob        | Smith     | engineering |
| Jane       | Doe       | engineering |
```

```
| Billy      | Smith     | engineering |
+-------------------------------------+
```

Da un elenco di dipendenti, seleziona quello con il miglior punteggio combinato. UNNEST può essere utilizzato nella clausola FROM senza essere preceduto da CROSS JOIN, poiché è l'operatore join predefinito e pertanto è implicito.

```
WITH
dataset AS (
  SELECT ARRAY[
    CAST(ROW('Sally', 'engineering', ARRAY[1,2,3,4]) AS ROW(name VARCHAR, department
 VARCHAR, scores ARRAY(INTEGER))),
    CAST(ROW('John', 'finance', ARRAY[7,8,9]) AS ROW(name VARCHAR, department VARCHAR,
 scores ARRAY(INTEGER))),
    CAST(ROW('Amy', 'devops', ARRAY[12,13,14,15]) AS ROW(name VARCHAR, department
 VARCHAR, scores ARRAY(INTEGER)))
  ] AS users
),
users AS (
 SELECT person, score
 FROM
   dataset,
   UNNEST(dataset.users) AS t(person),
   UNNEST(person.scores) AS t(score)
)
SELECT person.name, person.department, SUM(score) AS total_score FROM users
GROUP BY (person.name, person.department)
ORDER BY (total_score) DESC
LIMIT 1
```

Questa query restituisce:

```
+---------------------------------+
| name | department | total_score |
+---------------------------------+
| Amy  | devops     | 54          |
+---------------------------------+
```

Da un elenco di dipendenti, seleziona quello con il miglior punteggio individuale.

```
WITH
```

```
dataset AS (
 SELECT ARRAY[
   CAST(ROW('Sally', 'engineering', ARRAY[1,2,3,4]) AS ROW(name VARCHAR, department
 VARCHAR, scores ARRAY(INTEGER))),
   CAST(ROW('John', 'finance', ARRAY[7,8,9]) AS ROW(name VARCHAR, department VARCHAR,
 scores ARRAY(INTEGER))),
   CAST(ROW('Amy', 'devops', ARRAY[12,13,14,15]) AS ROW(name VARCHAR, department
 VARCHAR, scores ARRAY(INTEGER)))
 ] AS users
),
users AS (
 SELECT person, score
 FROM
   dataset,
   UNNEST(dataset.users) AS t(person),
   UNNEST(person.scores) AS t(score)
)
SELECT person.name, score FROM users
ORDER BY (score) DESC
LIMIT 1
```

Questa query restituisce:

```
+--------------+
| name | score |
+--------------+
| Amy  | 15    |
+--------------+
```

Considerazioni e limitazioni

Se UNNEST viene utilizzato su uno o più array nella query e uno degli array è NULL, la query non restituisce righe. Se UNNEST viene utilizzato su un array che è una stringa vuota, viene restituita la stringa vuota.

Ad esempio, nella query seguente, poiché il secondo array è nullo, la query non restituisce alcuna riga.

```
SELECT
    col1,
    col2
FROM UNNEST (ARRAY ['apples','oranges','lemons']) AS t(col1)
```

```
CROSS JOIN UNNEST (ARRAY []) AS t(col2)
```

Nell'esempio successivo, il secondo array viene modificato per contenere una stringa vuota. Per ogni riga, la query restituisce il valore in `col1` e una stringa vuota per il valore in `col2`. La stringa vuota nel secondo array è necessaria per restituire i valori del primo array.

```
SELECT
    col1,
    col2
FROM UNNEST (ARRAY ['apples','oranges','lemons']) AS t(col1)
CROSS JOIN UNNEST (ARRAY ['']) AS t(col2)
```

Creazione di matrici da sottoquery

Creare una matrice da una raccolta di righe.

```
WITH
dataset AS (
  SELECT ARRAY[1,2,3,4,5] AS items
)
SELECT array_agg(i) AS array_items
FROM dataset
CROSS JOIN UNNEST(items) AS t(i)
```

Questa query restituisce:

```
+-----------------+
| array_items     |
+-----------------+
| [1, 2, 3, 4, 5] |
+-----------------+
```

Per creare una matrice di valori univoci da un set di righe, utilizzare la parola chiave `distinct`.

```
WITH
dataset AS (
  SELECT ARRAY [1,2,2,3,3,4,5] AS items
)
SELECT array_agg(distinct i) AS array_items
FROM dataset
```

```
CROSS JOIN UNNEST(items) AS t(i)
```

Questa query restituisce il seguente risultato. Si noti che l'ordine non è garantito.

```
+-----------------+
| array_items     |
+-----------------+
| [1, 2, 3, 4, 5] |
+-----------------+
```

Per ulteriori informazioni sull'utilizzo della funzione `array_agg`, consulta la sezione Aggregate functions (Funzioni aggregate) nella documentazione di Trino.

Filtraggio delle matrici

Creare una matrice da una raccolta di righe nel caso in cui soddisfino i criteri di filtro.

```
WITH
dataset AS (
  SELECT ARRAY[1,2,3,4,5] AS items
)
SELECT array_agg(i) AS array_items
FROM dataset
CROSS JOIN UNNEST(items) AS t(i)
WHERE i > 3
```

Questa query restituisce:

```
+-------------+
| array_items |
+-------------+
| [4, 5]      |
+-------------+
```

Filtrare una matrice in base al fatto che uno dei suoi elementi contenga un valore specifico, ad esempio 2, come nel seguente esempio:

```
WITH
dataset AS (
  SELECT ARRAY
```

```
  [
    ARRAY[1,2,3,4],
    ARRAY[5,6,7,8],
    ARRAY[9,0]
  ] AS items
)
SELECT i AS array_items FROM dataset
CROSS JOIN UNNEST(items) AS t(i)
WHERE contains(i, 2)
```

Questa query restituisce:

```
+--------------+
| array_items  |
+--------------+
| [1, 2, 3, 4] |
+--------------+
```

La funzione `filter`

```
 filter(ARRAY [list_of_values], boolean_function)
```

Puoi utilizzare la funzione `filter` su un'espressione ARRAY per creare una nuova matrice che è il sottoinsieme degli elementi in *`list_of_values`* per cui*`boolean_function`* è "true". La funzione `filter` può essere utile nei casi in cui non è possibile utilizzare la funzione *`UNNEST`*.

Nell'esempio seguente vengono filtrati i valori maggiori di zero nella matrice `[1,0,5,-1]`.

```
SELECT filter(ARRAY [1,0,5,-1], x -> x>0)
```

Risultati

```
[1,5]
```

Nell'esempio seguente vengono filtrati i valori non-null nella matrice `[-1, NULL, 10, NULL]`.

```
SELECT filter(ARRAY [-1, NULL, 10, NULL], q -> q IS NOT NULL)
```

Risultati

```
[-1,10]
```

Ordinamento matrici

Per creare un array ordinato di valori univoci da un set di righe, puoi utilizzare la funzione array_sort come nell'esempio seguente.

```
WITH
dataset AS (
  SELECT ARRAY[3,1,2,5,2,3,6,3,4,5] AS items
)
SELECT array_sort(array_agg(distinct i)) AS array_items
FROM dataset
CROSS JOIN UNNEST(items) AS t(i)
```

Questa query restituisce:

```
+--------------------+
| array_items        |
+--------------------+
| [1, 2, 3, 4, 5, 6] |
+--------------------+
```

Per informazioni sull'espansione di un array in più righe, consulta la sezione Appiattimento di matrici nidificate.

Utilizzo delle funzioni di aggregazione con le matrici

- Per aggiungere valori all'interno di un array, utilizza SUM, come nell'esempio seguente.
- Per aggregare più righe all'interno di un array, utilizza `array_agg`. Per informazioni, consultare Creazione di matrici da sottoquery.

> **Note**
>
> `ORDER BY` è supportato per le funzioni di aggregazione a partire dalla versione 2 del motore Athena.

```
WITH
```

```
dataset AS (
  SELECT ARRAY
  [
    ARRAY[1,2,3,4],
    ARRAY[5,6,7,8],
    ARRAY[9,0]
  ] AS items
),
item AS (
  SELECT i AS array_items
  FROM dataset, UNNEST(items) AS t(i)
)
SELECT array_items, sum(val) AS total
FROM item, UNNEST(array_items) AS t(val)
GROUP BY array_items;
```

Nell'ultima istruzione SELECT, invece di utilizzare `sum()` e UNNEST, è possibile utilizzare `reduce()` per ridurre il tempo di elaborazione e il trasferimento dei dati, come nell'esempio seguente.

```
WITH
dataset AS (
  SELECT ARRAY
  [
    ARRAY[1,2,3,4],
    ARRAY[5,6,7,8],
    ARRAY[9,0]
  ] AS items
),
item AS (
  SELECT i AS array_items
  FROM dataset, UNNEST(items) AS t(i)
)
SELECT array_items, reduce(array_items, 0 , (s, x) -> s + x, s -> s) AS total
FROM item;
```

Qualsiasi query restituisce i seguenti risultati. L'ordine dei risultati restituiti non è garantito.

```
+----------------------+
| array_items  | total |
+----------------------+
| [1, 2, 3, 4] | 10    |
| [5, 6, 7, 8] | 26    |
| [9, 0]       | 9     |
```

```
+----------------------+
```

Conversione delle matrici in stringhe

Per convertire una matrice in una stringa singola, utilizza la funzione `array_join`. Il seguente esempio standalone crea una tabella denominata `dataset` che contiene una matrice alias chiamata `words`. La query utilizza `array_join` per unire gli elementi della matrice in `words`, separarli con spazi e restituire la stringa risultante in una colonna alias chiamata `welcome_msg`.

```
WITH
dataset AS (
  SELECT ARRAY ['hello', 'amazon', 'athena'] AS words
)
SELECT array_join(words, ' ') AS welcome_msg
FROM dataset
```

Questa query restituisce:

```
+----------------------+
| welcome_msg          |
+----------------------+
| hello amazon athena |
+----------------------+
```

Utilizzo delle matrici per creare mappe

Le mappe sono coppie chiave-valore formate da tipi di dati disponibili in Athena. Per creare mappe, utilizza l'operatore `MAP` e passalo in due matrici: la prima è costituita dai nomi delle colonne (chiave) e la seconda dai valori. Tutti i valori nelle matrici devono essere dello stesso tipo. Se uno qualsiasi degli elementi nelle matrici di valori deve essere di tipo diverso, è possibile convertirlo più tardi.

Esempi

In quest'esempio viene selezionato un utente da un set di dati. e si utilizza l'operatore `MAP`, facendolo poi passare da due matrici. La prima matrice include i valori per i nomi di colonna, ad esempio "primo", "ultimo" ed "età". La seconda matrice è formata da valori per ciascuna colonna, ad esempio "Bob", "Smith", "35".

```
WITH dataset AS (
```

```
  SELECT MAP(
    ARRAY['first', 'last', 'age'],
    ARRAY['Bob', 'Smith', '35']
  ) AS user
)
SELECT user FROM dataset
```

Questa query restituisce:

```
+--------------------------------+
| user                           |
+--------------------------------+
| {last=Smith, first=Bob, age=35} |
+--------------------------------+
```

È possibile recuperare i valori `Map` selezionando il nome di campo seguito da `[key_name]`, come in questo esempio:

```
WITH dataset AS (
 SELECT MAP(
   ARRAY['first', 'last', 'age'],
   ARRAY['Bob', 'Smith', '35']
 ) AS user
)
SELECT user['first'] AS first_name FROM dataset
```

Questa query restituisce:

```
+------------+
| first_name |
+------------+
| Bob        |
+------------+
```

Esecuzione di query su matrici con tipi complessi e strutture nidificate

I dati di origine contengono spesso matrici con tipi di dati complessi e strutture nidificate. Alcuni esempi di questa sezione mostrano come modificare il tipo di dati degli elementi, individuare elementi all'interno di matrici e trovare parole chiave utilizzando le query Athena.

- Creazione di una `ROW`

- Modifica dei nomi di campo nelle matrici utilizzando `CAST`
- Filtraggio delle matrici utilizzando la notazione `.`.
- Filtraggio delle matrici con valori nidificati
- Filtraggio delle matrici utilizzando `UNNEST`
- Ricerca di parole chiave nelle matrici utilizzando `regexp_like`

Creazione di una `ROW`

> **Note**
>
> Negli esempi di questa sezione viene utilizzato `ROW` come strumento per creare i dati di esempio con cui lavorare. Quando si eseguono le query di tabelle all'interno di Athena, non è necessario creare tipi di dati `ROW`, poiché sono già creati dall'origine dati. Quando si utilizza `CREATE_TABLE`, Athena definisce un `STRUCT` al suo interno, lo popola con i dati e crea il tipo di dati `ROW` per l'utente, per ogni riga nel set di dati. Il tipo di dati `ROW` sottostante è composto da campi designati di qualsiasi tipo di dati SQL supportato.

```
WITH dataset AS (
 SELECT
   ROW('Bob', 38) AS users
 )
SELECT * FROM dataset
```

Questa query restituisce:

```
+-------------------------+
| users                   |
+-------------------------+
| {field0=Bob, field1=38} |
+-------------------------+
```

Modifica dei nomi di campo nelle matrici utilizzando `CAST`

Per modificare il nome di campo in una matrice che contiene valori `ROW`, è possibile utilizzare `CAST` per l'istruzione `ROW`:

```
WITH dataset AS (
```

```
  SELECT
    CAST(
      ROW('Bob', 38) AS ROW(name VARCHAR, age INTEGER)
    ) AS users
)
SELECT * FROM dataset
```

Questa query restituisce:

```
+--------------------+
| users              |
+--------------------+
| {NAME=Bob, AGE=38} |
+--------------------+
```

> **Note**
>
> Nell'esempio sopra riportato, si dichiara `name` come `VARCHAR`, perché è il suo tipo in Presto. Se dichiari questo `STRUCT` all'interno di un'istruzione `CREATE TABLE`, utilizza il tipo `String`, perché Hive definisce questo tipo di dati come `String`.

Filtraggio delle matrici utilizzando la notazione `.`.

In questo esempio, seleziona il campo `accountId` dalla colonna `userIdentity` di una tabella di log di AWS CloudTrail utilizzando la notazione `.`. Per ulteriori informazioni, consulta Esecuzione di query sui log di AWS CloudTrail.

```
SELECT
  CAST(useridentity.accountid AS bigint) as newid
FROM cloudtrail_logs
LIMIT 2;
```

Questa query restituisce:

```
+--------------+
| newid        |
+--------------+
| 112233445566 |
+--------------+
```

```
| 998877665544 |
+--------------+
```

Per eseguire query di una matrice di valori, invia questa query:

```
WITH dataset AS (
  SELECT ARRAY[
    CAST(ROW('Bob', 38) AS ROW(name VARCHAR, age INTEGER)),
    CAST(ROW('Alice', 35) AS ROW(name VARCHAR, age INTEGER)),
    CAST(ROW('Jane', 27) AS ROW(name VARCHAR, age INTEGER))
  ] AS users
)
SELECT * FROM dataset
```

Tale operazione restituisce questo risultato:

```
+-----------------------------------------------------------------+
| users                                                           |
+-----------------------------------------------------------------+
| [{NAME=Bob, AGE=38}, {NAME=Alice, AGE=35}, {NAME=Jane, AGE=27}] |
+-----------------------------------------------------------------+
```

Filtraggio delle matrici con valori nidificati

Le matrici di grandi dimensioni spesso contengono strutture nidificate ed è necessario essere in grado di filtrare o cercare i valori all'interno di tali matrici.

Per definire un set di dati per una matrice di valori che include un valore `BOOLEAN` nidificato, emetti la query seguente:

```
WITH dataset AS (
  SELECT
    CAST(
      ROW('aws.amazon.com', ROW(true)) AS ROW(hostname VARCHAR, flaggedActivity
 ROW(isNew BOOLEAN))
    ) AS sites
)
SELECT * FROM dataset
```

Tale operazione restituisce questo risultato:

```
+-----------------------------------------------------------+
| sites                                                     |
+-----------------------------------------------------------+
| {HOSTNAME=aws.amazon.com, FLAGGEDACTIVITY={ISNEW=true}}   |
+-----------------------------------------------------------+
```

Quindi, per filtrare e individuare il valore BOOLEAN di tale elemento, continua a utilizzare la notazione `.`.

```
WITH dataset AS (
  SELECT
    CAST(
      ROW('aws.amazon.com', ROW(true)) AS ROW(hostname VARCHAR, flaggedActivity
 ROW(isNew BOOLEAN))
    ) AS sites
)
SELECT sites.hostname, sites.flaggedactivity.isnew
FROM dataset
```

Questa query seleziona i campi nidificati e restituisce questo risultato:

```
+------------------------+
| hostname       | isnew |
+------------------------+
| aws.amazon.com | true  |
+------------------------+
```

Filtraggio delle matrici utilizzando **UNNEST**

Per filtrare una matrice che include una struttura nidificata in base a uno dei suoi elementi figlio, invia una query con un operatore UNNEST. Per ulteriori informazioni su UNNEST, consulta la sezione relativa all'appiattimento delle matrici nidificate.

Ad esempio, questa query individua nomi host dei siti nel set di dati.

```
WITH dataset AS (
  SELECT ARRAY[
    CAST(
      ROW('aws.amazon.com', ROW(true)) AS ROW(hostname VARCHAR, flaggedActivity
 ROW(isNew BOOLEAN))
    ),
```

```
    CAST(
      ROW('news.cnn.com', ROW(false)) AS ROW(hostname VARCHAR, flaggedActivity
 ROW(isNew BOOLEAN))
    ),
    CAST(
      ROW('netflix.com', ROW(false)) AS ROW(hostname VARCHAR, flaggedActivity ROW(isNew
 BOOLEAN))
    )
  ] as items
)
SELECT sites.hostname, sites.flaggedActivity.isNew
FROM dataset, UNNEST(items) t(sites)
WHERE sites.flaggedActivity.isNew = true
```

Restituisce:

```
+------------------------+
| hostname       | isnew |
+------------------------+
| aws.amazon.com | true  |
+------------------------+
```

Ricerca di parole chiave nelle matrici utilizzando **`regexp_like`**

I seguenti esempi illustrano come cercare una parola chiave in un set di dati in un elemento all'interno di una matrice, utilizzando la funzione regexp_like. Come input prende in considerazione un modello di espressione regolare da analizzare o un elenco di termini separati da una barra verticale (|), valuta i modelli e determina se la stringa specificata li contiene.

L'espressione regolare deve essere contenuta all'interno della stringa e non deve corrispondere con essa. Affinché corrisponda all'intera stringa, racchiudi il motivo con ^ all'inizio e $ al termine, ad esempio `'^pattern$'`.

Considera una matrice di siti contenenti il loro nome host e un elemento `flaggedActivity`. Tale elemento include una `ARRAY` che a sua volta contiene diversi elementi `MAP`, ciascuno dei quali elenca diverse parole chiave comuni e il relativo conteggio popolarità. Supponi di voler trovare una determinata parola chiave all'interno di una `MAP` in questa matrice.

Per cercare questo set di dati per i siti con una parola chiave specifica, utilizziamo `regexp_like` anziché l'operatore `LIKE` SQL simile, perché la ricerca di un numero elevato di parole chiave è più efficiente con `regexp_like`.

Example Esempio 1: utilizzo di **`regexp_like`**

La query in questo esempio utilizza la funzione `regexp_like` per cercare termini `'politics|bigdata'`, ubicati nei valori all'interno delle matrici:

```
WITH dataset AS (
  SELECT ARRAY[
    CAST(
      ROW('aws.amazon.com', ROW(ARRAY[
          MAP(ARRAY['term', 'count'], ARRAY['bigdata', '10']),
          MAP(ARRAY['term', 'count'], ARRAY['serverless', '50']),
          MAP(ARRAY['term', 'count'], ARRAY['analytics', '82']),
          MAP(ARRAY['term', 'count'], ARRAY['iot', '74'])
      ])
      ) AS ROW(hostname VARCHAR, flaggedActivity ROW(flags ARRAY(MAP(VARCHAR,
 VARCHAR)) ))
    ),
    CAST(
      ROW('news.cnn.com', ROW(ARRAY[
        MAP(ARRAY['term', 'count'], ARRAY['politics', '241']),
        MAP(ARRAY['term', 'count'], ARRAY['technology', '211']),
        MAP(ARRAY['term', 'count'], ARRAY['serverless', '25']),
        MAP(ARRAY['term', 'count'], ARRAY['iot', '170'])
      ])
      ) AS ROW(hostname VARCHAR, flaggedActivity ROW(flags ARRAY(MAP(VARCHAR,
 VARCHAR)) ))
    ),
    CAST(
      ROW('netflix.com', ROW(ARRAY[
        MAP(ARRAY['term', 'count'], ARRAY['cartoons', '1020']),
        MAP(ARRAY['term', 'count'], ARRAY['house of cards', '112042']),
        MAP(ARRAY['term', 'count'], ARRAY['orange is the new black', '342']),
        MAP(ARRAY['term', 'count'], ARRAY['iot', '4'])
      ])
      ) AS ROW(hostname VARCHAR, flaggedActivity ROW(flags ARRAY(MAP(VARCHAR,
 VARCHAR)) ))
    )
 ] AS items
),
sites AS (
  SELECT sites.hostname, sites.flaggedactivity
  FROM dataset, UNNEST(items) t(sites)
)
SELECT hostname
```

```
FROM sites, UNNEST(sites.flaggedActivity.flags) t(flags)
WHERE regexp_like(flags['term'], 'politics|bigdata')
GROUP BY (hostname)
```

Questa query restituisce due siti:

```
+----------------+
| hostname       |
+----------------+
| aws.amazon.com |
+----------------+
| news.cnn.com   |
+----------------+
```

Example Esempio 2: utilizzo di **`regexp_like`**

La query nell'esempio seguente aggiunge il punteggio totale di popolarità per i siti che soddisfano i termini di ricerca con la funzione `regexp_like`, quindi li ordina dal più alto al più basso.

```
WITH dataset AS (
  SELECT ARRAY[
    CAST(
      ROW('aws.amazon.com', ROW(ARRAY[
          MAP(ARRAY['term', 'count'], ARRAY['bigdata', '10']),
          MAP(ARRAY['term', 'count'], ARRAY['serverless', '50']),
          MAP(ARRAY['term', 'count'], ARRAY['analytics', '82']),
          MAP(ARRAY['term', 'count'], ARRAY['iot', '74'])
      ])
      ) AS ROW(hostname VARCHAR, flaggedActivity ROW(flags ARRAY(MAP(VARCHAR,
 VARCHAR)) ))
    ),
    CAST(
      ROW('news.cnn.com', ROW(ARRAY[
        MAP(ARRAY['term', 'count'], ARRAY['politics', '241']),
        MAP(ARRAY['term', 'count'], ARRAY['technology', '211']),
        MAP(ARRAY['term', 'count'], ARRAY['serverless', '25']),
        MAP(ARRAY['term', 'count'], ARRAY['iot', '170'])
      ])
      ) AS ROW(hostname VARCHAR, flaggedActivity ROW(flags ARRAY(MAP(VARCHAR,
 VARCHAR)) ))
    ),
    CAST(
```

```
      ROW('netflix.com', ROW(ARRAY[
        MAP(ARRAY['term', 'count'], ARRAY['cartoons', '1020']),
        MAP(ARRAY['term', 'count'], ARRAY['house of cards', '112042']),
        MAP(ARRAY['term', 'count'], ARRAY['orange is the new black', '342']),
        MAP(ARRAY['term', 'count'], ARRAY['iot', '4'])
      ])
      ) AS ROW(hostname VARCHAR, flaggedActivity ROW(flags ARRAY(MAP(VARCHAR,
 VARCHAR)) ))
    )
  ] AS items
),
sites AS (
  SELECT sites.hostname, sites.flaggedactivity
  FROM dataset, UNNEST(items) t(sites)
)
SELECT hostname, array_agg(flags['term']) AS terms, SUM(CAST(flags['count'] AS
 INTEGER)) AS total
FROM sites, UNNEST(sites.flaggedActivity.flags) t(flags)
WHERE regexp_like(flags['term'], 'politics|bigdata')
GROUP BY (hostname)
ORDER BY total DESC
```

Questa query restituisce due siti:

```
+-------------------------------------+
| hostname       | terms    | total  |
+----------------+------------------+
| news.cnn.com   | politics |  241   |
+----------------+------------------+
| aws.amazon.com | bigdata |  10    |
+----------------+------------------+
```

Esecuzione di query su dati geospaziali

I dati geospaziali contengono identificatori che specificano una posizione geografica per un oggetto. Esempi di questo tipo di dati includono i bollettini meteorologici, le indicazioni stradali sulle mappe, i tweet con posizioni geografiche, le ubicazioni dei negozi e le tratte delle compagnie aeree. I dati geospaziali svolgono un ruolo importante per le attività di analisi aziendale, creazione di report e di previsione.

Gli identificatori geospaziali quali latitudine e longitudine consentono di convertire qualsiasi indirizzo postale in un set di coordinate geografiche.

Argomenti

Cos'è una query geospaziale?

Le query geospaziali sono tipi speciali di query SQL supportati in Athena. Si differiscono dalle query SQL non spaziali nei seguenti modi:

- Utilizzano i seguenti tipi di dati geometrici specializzati: `point`, `line`, `multiline`, `polygon` e `multipolygon`.
- Esprimono relazioni tra tipi di dati geometrici, ad esempio `distance`, `equals`, `crosses`, `touches`, `overlaps`, `disjoint`, tra le altre.

Se si utilizzano query geospaziali in Athena, è possibile eseguire queste e altre operazioni simili:

- Individuare la distanza tra due punti;
- Controllare se un'area (poligono) ne contiene un'altra;
- Controllare se una linea attraversa o tocca un'altra linea o un poligono.

Ad esempio, per ottenere un tipo di dati geometria `point` da valori di tipo `double` per le coordinate geografiche di Monte Rainier in Athena, utilizzare la funzione geospaziale `ST_Point (longitude, latitude)`, come nell'esempio seguente.

```
ST_Point(-121.7602, 46.8527)
```

Formati dei dati di input e tipi di dati di geometria

Per utilizzare le funzioni geospaziali in Athena, inserisci i dati nel formato WKT o usa Hive JSON. SerDe È inoltre possibile utilizzare i tipi di dati di geometria supportati in Athena.

Formati dei dati di input

Per gestire le query geospaziali, Athena supporta i dati di input in questi formati di dati:

- WKT (Well-known Text). In Athena, WKT è rappresentato da un tipo di dati `varchar(x)` o `string`.
- Dati geospaziali codificati JSON. Per analizzare i file JSON con dati geospaziali e creare tabelle per essi, Athena utilizza Hive JSON. SerDe Per ulteriori informazioni sull'utilizzo di questa funzionalità SerDe in Athena, vedere. Librerie JSON SerDe

Tipi di dati di geometria

Per gestire le query geospaziali, Athena supporta questi tipi di dati di geometria specializzati:

- `point`
- `line`
- `polygon`
- `multiline`
- `multipolygon`

Funzioni geospaziali supportate

Le funzioni geospaziali disponibili in Athena dipendono dalla versione del motore utilizzata.

- Per informazioni sulle funzioni geospaziali della versione 3 del motore Athena, consulta Funzioni geospaziali nella documentazione di Trino.
- Per un elenco delle modifiche apportate ai nomi delle funzioni e per le nuove funzioni disponibili nella versione 2 del motore Athena, consulta Modifiche ai nomi delle funzioni geospaziali e nuove funzioni nella versione 2 del motore Athena.

Per ulteriori informazioni sulle versioni del motore Athena, consulta Controllo delle versioni del motore di Athena.

Argomenti

- Funzioni geospaziali nella versione 3 del motore Athena
- Funzioni geospaziali nella versione 2 del motore Athena

Funzioni geospaziali nella versione 3 del motore Athena

Per informazioni sulle funzioni geospaziali della versione 3 del motore Athena, consulta Funzioni geospaziali nella documentazione di Trino.

Funzioni geospaziali nella versione 2 del motore Athena

In questo argomento vengono elencate solo le funzioni geospaziali ESRI supportate a partire dalla versione 2 del motore Athena. Per ulteriori informazioni sulle versioni del motore Athena, consulta Controllo delle versioni del motore di Athena.

Cambiamenti nella versione 2 del motore Athena

- I tipi di input e output per alcune funzioni sono cambiati. In particolare, il tipo `VARBINARY` non è più direttamente supportato per l'input. Per ulteriori informazioni, consulta Modifiche alle funzioni geospaziali.
- I nomi di alcune funzioni geospaziali sono cambiati. Per ulteriori informazioni, consulta Modifiche ai nomi delle funzioni geospaziali nella versione 2 del motore Athena.
- Sono state aggiunte nuove funzioni. Per ulteriori informazioni, consulta Nuove funzioni geospaziali nella versione 2 del motore Athena.

Athena supporta i seguenti tipi di funzioni geospaziali:

- Funzioni costruttore
- Funzioni di relazioni geospaziali
- Funzioni operazione
- Funzioni di accesso
- Funzioni di aggregazione
- Funzioni del riquadro Bing

Funzioni costruttore

Le funzioni costruttore consentono di ottenere rappresentazioni binarie dei tipi di dati di geometria `point`, `line` o `polygon`. È anche possibile utilizzare queste funzioni per convertire dati binari in testo e ottenere valori binari per i dati di geometria espressi in formato WKT (Well-Known Text).

ST_AsBinary(geometry)

Restituisce un tipo di dati varbinary che contiene la rappresentazione WKB della geometria specificata. Esempio:

```
SELECT ST_AsBinary(ST_Point(-158.54, 61.56))
```

ST_AsText(geometry)

Converte ciascuno dei tipi di dati di geometria specificati in testo. Restituisce un valore nel tipo di dati varchar, che è una rappresentazione WKT del tipo di dati di geometria. Esempio:

```
SELECT ST_AsText(ST_Point(-158.54, 61.56))
```

ST_GeomAsLegacyBinary(geometry)

Restituisce un varbinary legacy dalla geometria specificata. Esempio:

```
SELECT ST_GeomAsLegacyBinary(ST_Point(-158.54, 61.56)
```

ST_GeometryFromText(varchar)

Converte il testo in formato WKT in un tipo di dati di geometria. Restituisce un valore in un tipo di dati di geometria. Esempio:

```
SELECT ST_GeometryFromText(ST_AsText(ST_Point(1, 2)))
```

ST_GeomFromBinary(varbinary)

Restituisce un oggetto di tipo geometria da una rappresentazione WKB. Esempio:

```
SELECT ST_GeomFromBinary(ST_AsBinary(ST_Point(-158.54, 61.56)))
```

ST_GeomFromLegacyBinary(varbinary)

Restituisce un oggetto del tipo geometria da un tipo varbinary legacy. Esempio:

```
SELECT ST_GeomFromLegacyBinary(ST_GeomAsLegacyBinary(ST_Point(-158.54, 61.56)))
```

ST_LineFromText(varchar)

Restituisce un valore nel tipo di dati di geometria `line`. Esempio:

```
SELECT ST_Line('linestring(1 1, 2 2, 3 3)')
```

ST_LineString(array(point))

Restituisce un tipo di geometria `LineString` formato da un array di tipi di geometria puntiforme. Se nell'array specificato sono presenti meno di due punti non vuoti, viene restituita una `LineString` vuota. Restituisce un'eccezione se qualsiasi elemento nell'array è nullo o vuoto o identico a quello precedente. La geometria restituita potrebbe non essere semplice. A seconda dell'input specificato, la geometria restituita può auto-intersecarsi o contenere vertici duplicati. Esempio:

```
SELECT ST_LineString(ARRAY[ST_Point(-158.54, 61.56), ST_Point(-158.55, 61.56)])
```

ST_MultiPoint(array(point))

Restituisce un oggetto geometria `MultiPoint` formato dai punti specificati. Restituisce null se l'array specificato è vuoto. Restituisce un'eccezione se qualsiasi elemento nell'array è nullo o vuoto. La geometria restituita potrebbe non essere semplice e può contenere punti duplicati se l'array specificato ha duplicati. Esempio:

```
SELECT ST_MultiPoint(ARRAY[ST_Point(-158.54, 61.56), ST_Point(-158.55, 61.56)])
```

ST_Point(double, double)

Restituisce un oggetto di tipo di geometria `point`. Per i valori dei dati di input per questa funzione, è possibile usare valori geometrici, come i valori del sistema di coordinate cartesiane UTM (proiezione trasversa di Mercatore), o coordinate geografiche (longitudine e latitudine) in gradi decimali. I valori di latitudine e longitudine usano lo standard World Geodetic System, noto anche come WGS 1984 o EPSG:4326. WGS 1984 è il sistema di coordinate utilizzato dal sistema di posizionamento globale (GPS).

Ad esempio, nella notazione seguente le coordinate geografiche sono specificate in longitudine e latitudine e il valore `.072284`, che è la distanza buffer, è specificato in unità angolari come gradi decimali:

```
SELECT ST_Buffer(ST_Point(-74.006801, 40.705220), .072284)
```

Sintassi:

```
SELECT ST_Point(longitude, latitude) FROM earthquakes LIMIT 1
```

L'esempio esempio usa specifiche coordinate di longitudine e latitudine:

```
SELECT ST_Point(-158.54, 61.56)
FROM earthquakes
LIMIT 1
```

Il prossimo esempio usa specifiche coordinate di longitudine e latitudine:

```
SELECT ST_Point(-74.006801, 40.705220)
```

L'esempio seguente utilizza la funzione `ST_AsText` per ottenere la geometria da WKT:

```
SELECT ST_AsText(ST_Point(-74.006801, 40.705220)) AS WKT
```

ST_Polygon(varchar)

Utilizzando la sequenza delle ordinate fornite in senso orario, da sinistra a destra, restituisce un tipo di dati di geometria `polygon`. A partire dalla versione 2 del motore Athena, solo i poligoni sono accettati come input. Esempio:

```
SELECT ST_Polygon('polygon ((1 1, 1 4, 4 4, 4 1))')
```

to_geometry(sphericalGeography)

Restituisce un oggetto geometria dall'oggetto geografico sferico specificato. Esempio:

```
SELECT to_geometry(to_spherical_geography(ST_Point(-158.54, 61.56)))
```

to_spherical_geography(geometry)

Restituisce un oggetto geografico sferico dalla geometria specificata. Utilizza questa funzione per convertire un oggetto geometrico in un oggetto geografico sferico sulla sfera del raggio terrestre. Questa funzione può essere utilizzata solo sulle geometrie `POINT`, `MULTIPOINT`, `LINESTRING`, `MULTILINESTRING`, `POLYGON` e `MULTIPOLYGON` definite nello spazio 2D o su

GEOMETRYCOLLECTION di tali geometrie. Per ogni punto della geometria specificata, la funzione verifica che point.x si trovi all'interno di [-180.0, 180.0] e che point.y si trovi all'interno di [-90.0, 90.0]. La funzione utilizza questi punti come gradi di longitudine e latitudine per costruire la forma del risultato sphericalGeography.

Esempio:

```
SELECT to_spherical_geography(ST_Point(-158.54, 61.56))
```

Funzioni di relazioni geospaziali

Le seguenti funzioni esprimono le relazioni tra due diverse geometrie specificate come input e restituiscono risultati di tipo boolean. L'ordine in cui viene specificata la coppia di geometrie conta: il primo valore di geometria viene denominato geometria a sinistra, il secondo valore di geometria viene denominato geometria a destra.

Queste funzioni restituiscono:

- TRUE se e solo se la relazione descritta dalla funzione viene soddisfatta.
- FALSE se e solo se la relazione descritta dalla funzione non viene soddisfatta.

ST_Contains(geometry, geometry)

Restituisce TRUE se e solo se la geometria a sinistra contiene geometria a destra. Esempi:

```
SELECT ST_Contains('POLYGON((0 2,1 1,0 -1,0 2))', 'POLYGON((-1 3,2 1,0 -3,-1 3))')
```

```
SELECT ST_Contains('POLYGON((0 2,1 1,0 -1,0 2))', ST_Point(0, 0))
```

```
SELECT ST_Contains(ST_GeometryFromText('POLYGON((0 2,1 1,0 -1,0 2))'),
 ST_GeometryFromText('POLYGON((-1 3,2 1,0 -3,-1 3))'))
```

ST_Crosses(geometry, geometry)

Restituisce TRUE se e solo se la geometria a sinistra incrocia la geometria a destra. Esempio:

```
SELECT ST_Crosses(ST_Line('linestring(1 1, 2 2 )'), ST_Line('linestring(0 1, 2 2)'))
```

ST_Disjoint(geometry, geometry)

Restituisce TRUE se e solo se l'intersezione di geometrie a sinistra e geometrie a destra è vuota. Esempio:

```
SELECT ST_Disjoint(ST_Line('linestring(0 0, 0 1)'), ST_Line('linestring(1 1, 1 0)'))
```

ST_Equals(geometry, geometry)

Restituisce TRUE se e solo se la geometria a sinistra è uguale alla geometria a destra. Esempio:

```
SELECT ST_Equals(ST_Line('linestring( 0 0, 1 1)'), ST_Line('linestring(1 3, 2 2)'))
```

ST_Intersects(geometry, geometry)

Restituisce TRUE se e solo se la geometria a sinistra interseca la geometria a destra. Esempio:

```
SELECT ST_Intersects(ST_Line('linestring(8 7, 7 8)'), ST_Polygon('polygon((1 1, 4 1, 4
 4, 1 4))'))
```

ST_Overlaps(geometry, geometry)

Restituisce TRUE se e solo se la geometria a sinistra si sovrappone alla geometria a destra. Esempio:

```
SELECT ST_Overlaps(ST_Polygon('polygon((2 0, 2 1, 3 1))'), ST_Polygon('polygon((1 1, 1
 4, 4 4, 4 1))'))
```

ST_Relate(geometry, geometry, varchar)

Restituisce TRUE se e solo se la geometria a sinistra dispone della relazione Dimensionally Extended nine-Intersection Model (DE-9IM) specificata con la geometria a destra. Il terzo input (`varchar`) prende la relazione. Esempio:

```
SELECT ST_Relate(ST_Line('linestring(0 0, 3 3)'), ST_Line('linestring(1 1, 4 4)'),
 'T********')
```

ST_Touches(geometry, geometry)

Restituisce TRUE se e solo se la geometria a sinistra tocca la geometria a destra.

Esempio:

```
SELECT ST_Touches(ST_Point(8, 8), ST_Polygon('polygon((1  1, 1  4, 4  4, 4 1))'))
```

ST_Within(geometry, geometry)

Restituisce TRUE se e solo se la geometria a sinistra si trova all'interno della geometria a destra.

Esempio:

```
SELECT ST_Within(ST_Point(8, 8), ST_Polygon('polygon((1  1, 1  4, 4  4, 4 1))'))
```

Funzioni operazione

Utilizzare le funzioni di operazione per eseguire operazioni su valori dei tipi di dati di geometria. Ad esempio, è possibile ottenere i limiti di un singolo tipo di dati di geometria; intersezioni tra due tipi di dati di geometria; differenza tra geometria a sinistra e a destra, dove ogni geometria è dello stesso tipo di dati; oppure un buffer esterno o anello intorno a un determinato tipo di dati di geometria.

geometry_union(array(geometry))

Restituisce una geometria che rappresenta l'unione degli insiemi di punti delle geometrie specificate. Esempio:

```
SELECT geometry_union(ARRAY[ST_Point(-158.54, 61.56), ST_Point(-158.55, 61.56)])
```

ST_Boundary(geometry)

Richiede come input uno dei tipi di dati di geometria e restituisce il tipo di dati di geometria `boundary`.

Esempi:

```
SELECT ST_Boundary(ST_Line('linestring(0 1, 1 0)')))
```

```
SELECT ST_Boundary(ST_Polygon('polygon((1  1, 1  4, 4  4, 4 1))'))
```

ST_Buffer(geometry, double)

Richiede come input uno dei tipi di dati di geometria, ad esempio punto, linea, poligono, multilinea o multipoligono e una distanza come tipo `double`). Restituisce il tipo di dati di geometria eseguiti in buffer dalla distanza specificata (o raggio). Esempio:

```
SELECT ST_Buffer(ST_Point(1, 2), 2.0)
```

Nell'esempio seguente le coordinate geografiche sono specificate in longitudine e latitudine e il valore `.072284`, che è la distanza buffer, è specificato in unità angolari come gradi decimali:

```
SELECT ST_Buffer(ST_Point(-74.006801, 40.705220), .072284)
```

ST_Difference(geometry, geometry)

Restituisce una rappresentazione della differenza tra geometria a sinistra e geometria a destra. Esempio:

```
SELECT ST_AsText(ST_Difference(ST_Polygon('polygon((0 0, 0 10, 10 10, 10 0))'),
 ST_Polygon('polygon((0 0, 0 5, 5 5, 5 0))')))
```

ST_Envelope(geometry)

Accetta tipi di dati di geometria `line`, `polygon`, `multiline` e `multipolygon` come input. Non supporta il tipo di dati di geometria `point`. Restituisce l'envelope come geometria, dove l'envelope è un rettangolo attorno al tipo di dati di geometria specificato. Esempi:

```
SELECT ST_Envelope(ST_Line('linestring(0 1, 1 0)'))
```

```
SELECT ST_Envelope(ST_Polygon('polygon((1  1, 1  4, 4  4, 4 1))'))
```

ST_EnvelopeAsPts(geometry)

Restituisce un array di due punti che rappresentano gli angoli inferiore sinistro e superiore destro del poligono rettangolare che delimita la geometria. Restituisce null se la geometria specificata è vuoto. Esempio:

```
SELECT ST_EnvelopeAsPts(ST_Point(-158.54, 61.56))
```

ST_ExteriorRing(geometry)

Restituisce la geometria dell'anello esterno del tipo di input `polygon`. A partire dalla versione 2 del motore Athena, i poligoni sono le uniche geometrie accettate come input. Esempi:

```
SELECT ST_ExteriorRing(ST_Polygon(1,1, 1,4, 4,1))
```

```
SELECT ST_ExteriorRing(ST_Polygon('polygon ((0 0, 8 0, 0 8, 0 0), (1 1, 1 5, 5 1, 1
 1))'))
```

ST_Intersection(geometry, geometry)

Restituisce la geometria dell'intersezione della geometria a sinistra e a destra. Esempi:

```
SELECT ST_Intersection(ST_Point(1,1), ST_Point(1,1))
```

```
SELECT ST_Intersection(ST_Line('linestring(0 1, 1 0)'), ST_Polygon('polygon((1  1, 1
 4, 4  4, 4 1))'))
```

```
SELECT ST_AsText(ST_Intersection(ST_Polygon('polygon((2 0, 2 3, 3 0))'),
 ST_Polygon('polygon((1 1, 4 1, 4 4, 1 4))')))
```

ST_SymDifference(geometry, geometry)

Restituisce la geometria della differenza geometricamente simmetrica tra la geometria a sinistra e quella a destra. Esempio:

```
SELECT ST_AsText(ST_SymDifference(ST_Line('linestring(0 2, 2 2)'),
 ST_Line('linestring(1 2, 3 2)')))
```

ST_Union(geometry, geometry)

Restituisce un tipo di dati della geometria che rappresenta l'unione di set di punti delle geometrie specificate. Esempio:

```
SELECT ST_Union(ST_Point(-158.54, 61.56),ST_LineString(array[ST_Point(1,2),
 ST_Point(3,4)]))
```

Funzioni di accesso

Le funzioni di accesso sono utili per ottenere valori nei tipi `varchar`, `bigint` o `double` da diversi tipi di dati `geometry`, dove `geometry` è uno dei tipi di dati di geometria supportati in Athena: `point`, `line`, `polygon`, `multiline` e `multipolygon`. Ad esempio, è possibile ottenere un'area di un tipo di dati di geometria `polygon`, valori massimo e minimo di X e Y per un determinato tipo di dati di

geometria, ottenere la lunghezza di `line` o ricevere il numero di punti in un determinato tipo di dati di geometria.

`geometry_invalid_reason(geometry)`

Restituisce, in un tipo di dati varchar, il motivo per cui la geometria specificata non è valida o non semplice. Se la geometria specificata non è né valida né semplice, restituisce il motivo per cui non è valida. Se la geometria specificata è valida e semplice, restituisce null. Esempio:

```
SELECT geometry_invalid_reason(ST_Point(-158.54, 61.56))
```

`great_circle_distance(latitude1, longitude1, latitude2, longitude2)`

Restituisce, come doppio, la distanza ortodromica tra due punti sulla superficie terrestre in chilometri. Esempio:

```
SELECT great_circle_distance(36.12, -86.67, 33.94, -118.40)
```

`line_locate_point(lineString, point)`

Restituisce un doppio compreso tra 0 e 1 che rappresenta la posizione del punto più vicino sulla stringa di riga specificata al punto specificato come frazione della lunghezza totale della linea 2d.

Restituisce null se la stringa di riga o il punto specificato è vuota o nullo. Esempio:

```
SELECT line_locate_point(ST_GeometryFromText('LINESTRING (0 0, 0 1)'), ST_Point(0,
 0.2))
```

`simplify_geometry(geometry, double)`

Utilizza l'amer-douglas-peucker algoritmo R per restituire un tipo di dati geometrico che è una versione semplificata della geometria specificata. Evita la creazione di geometrie derivate (in particolare poligoni) non valide. Esempio:

```
SELECT simplify_geometry(ST_GeometryFromText('POLYGON ((1 0, 2 1, 3 1, 3 1, 4 1, 1
 0))'), 1.5)
```

`ST_Area(geometry)`

Richiede come input un tipo di dati di geometria e restituisce un'area nel tipo `double`. Esempio:

```
SELECT ST_Area(ST_Polygon('polygon((1 1, 4 1, 4 4, 1 4))'))
```

ST_Centroid(geometry)

Richiede come input un tipo di dati di geometria `polygon` e restituisce un tipo di dati di geometria `point`, che è il centro dell'envelope del poligono. Esempi:

```
SELECT ST_Centroid(ST_GeometryFromText('polygon ((0 0, 3 6, 6 0, 0 0))'))
```

```
SELECT ST_AsText(ST_Centroid(ST_Envelope(ST_GeometryFromText('POINT (53 27)'))))
```

ST_ConvexHull(geometry)

Restituisce un tipo di dati della geometria che è la geometria convessa più piccola che racchiude tutte le geometrie nell'input specificato. Esempio:

```
SELECT ST_ConvexHull(ST_Point(-158.54, 61.56))
```

ST_CoordDim(geometry)

Richiede come input uno dei tipi di dati di geometria supportati e restituisce il numero di componenti di coordinata nel tipo `tinyint`. Esempio:

```
SELECT ST_CoordDim(ST_Point(1.5,2.5))
```

ST_Dimension(geometry)

Richiede come input uno dei tipi di dati di geometria supportati e restituisce la dimensione spaziale di una geometria nel tipo `tinyint`. Esempio:

```
SELECT ST_Dimension(ST_Polygon('polygon((1 1, 4 1, 4 4, 1 4))'))
```

ST_Distance(geometry, geometry)

Restituisce, in base al ref spaziale, un doppio contenente la distanza cartesiana minima bidimensionale tra due geometrie in unità proiettate. A partire dalla versione 2 del motore Athena, restituisce null se uno degli input è una geometria vuota. Esempio:

```
SELECT ST_Distance(ST_Point(0.0,0.0), ST_Point(3.0,4.0))
```

ST_Distance(sphericalGeography, sphericalGeography)

Restituisce, come doppio, la distanza grande cerchio tra due punti geografici sferici, espressa in metri. Esempio:

```
SELECT ST_Distance(to_spherical_geography(ST_Point(61.56,
 -86.67)),to_spherical_geography(ST_Point(61.56, -86.68)))
```

ST_EndPoint(geometry)

Restituisce l'ultimo punto di un tipo di dati di geometria `line` nel tipo di dati della geometria `point`. Esempio:

```
SELECT ST_EndPoint(ST_Line('linestring(0 2, 2 2)'))
```

ST_Geometries(geometry)

Restituisce un array di oggetti geometrici nell'insieme specificato. Se la geometria specificata non è una geometria multigeometria, restituisce un array a un elemento. Se la geometria specificata è vuota, restituisce il valore null.

Ad esempio, dato un oggetto `MultiLineString`, `ST_Geometries` crea un array di oggetti `LineString`. Dato un oggetto `GeometryCollection`, `ST_Geometries` restituisce un array non appiattito dei suoi costituenti. Esempio:

```
SELECT ST_Geometries(GEOMETRYCOLLECTION(MULTIPOINT(0 0, 1 1),
 GEOMETRYCOLLECTION(MULTILINESTRING((2 2, 3 3)))))
```

Risultato:

```
array[MULTIPOINT(0 0, 1 1),GEOMETRYCOLLECTION(MULTILINESTRING((2 2, 3 3)))]
```

ST_GeometryN(geometry, index)

Restituisce, come tipo di dati della geometria, l'elemento della geometria a un indice intero specificato. Gli indici iniziano da 1. Se la geometria specificata è un insieme di geometrie (ad

esempio, un oggetto GEOMETRYCOLLECTION o MULTI*), restituisce la geometria all'indice specificato. Se l'indice specificato è minore di 1 o maggiore del numero totale di elementi nell'insieme, restituisce null. Per trovare il numero totale di elementi, usare ST_NumGeometries. Le geometrie singolari (ad esempio, POINT, LINESTRING o POLYGON), vengono trattate come insiemi di un elemento. Le geometrie vuote vengono trattate come raccolte vuote. Esempio:

```
SELECT ST_GeometryN(ST_Point(-158.54, 61.56),1)
```

ST_GeometryType(geometry)

Restituisce, come varchar, il tipo di geometria. Esempio:

```
SELECT ST_GeometryType(ST_Point(-158.54, 61.56))
```

ST_InteriorRingN(geometry, index)

Restituisce l'elemento anello interno all'indice specificato (gli indici iniziano da 1). Se l'indice dato è minore di 1 o maggiore del numero totale di elementi degli anelli interni nella geometria specificata, restituisce null. Genera un errore se la geometria specificata non è un poligono. Per trovare il numero totale di elementi, usare ST_NumInteriorRing. Esempio:

```
SELECT ST_InteriorRingN(st_polygon('polygon ((0 0, 1 0, 1 1, 0 1, 0 0))'),1)
```

ST_InteriorRings(geometry)

Restituisce un array geometrico di tutti gli anelli interni trovati nella geometria specificata o un array vuoto se il poligono non ha anelli interni. Se la geometria specificata è vuota, restituisce il valore null. Se la geometria specificata non è un poligono, genera un errore. Esempio:

```
SELECT ST_InteriorRings(st_polygon('polygon ((0 0, 1 0, 1 1, 0 1, 0 0))'))
```

ST_IsClosed(geometry)

Accetta come input solo line e i tipi di dati di geometria multiline. Restituisce TRUE (tipo boolean) se e solo se la linea è chiusa. Esempio:

```
SELECT ST_IsClosed(ST_Line('linestring(0 2, 2 2)'))
```

ST_IsEmpty(geometry)

Accetta come input solo `line` e i tipi di dati di geometria `multiline`. Restituisce `TRUE` (tipo `boolean`) se e solo se la geometria specificata è vuota, in altre parole, quando i valori di inizio e fine `line` coincidono. Esempio:

```
SELECT ST_IsEmpty(ST_Point(1.5, 2.5))
```

ST_IsRing(geometry)

Restituisce `TRUE` (tipo `boolean`) se e solo se il tipo `line` è chiuso e semplice. Esempio:

```
SELECT ST_IsRing(ST_Line('linestring(0 2, 2 2)'))
```

ST_IsSimple(geometry)

Restituisce true se la geometria specificata non ha punti geometrici anomali (ad esempio, auto-intersezione o auto-tangenza). Per determinare il motivo per cui la geometria non è semplice, utilizzare `geometry_invalid_reason()`. Esempio:

```
SELECT ST_IsSimple(ST_LineString(array[ST_Point(1,2), ST_Point(3,4)]))
```

ST_IsValid(geometry)

Restituisce true se e solo se la geometria specificata è ben formata. Per determinare il motivo per cui la geometria non è ben formata, utilizzare `geometry_invalid_reason()`. Esempio:

```
SELECT ST_IsValid(ST_Point(61.56, -86.68))
```

ST_Length(geometry)

Restituisce la lunghezza di `line` nel tipo `double`. Esempio:

```
SELECT ST_Length(ST_Line('linestring(0 2, 2 2)'))
```

ST_NumGeometries(geometry)

Restituisce, come numero intero intero, il numero di oggetti geometrici presenti nell'insieme. Se la geometria è un insieme di geometrie (ad esempio, un oggetto `GEOMETRYCOLLECTION` o

MULTI*), restituisce il numero di geometrie. Le geometrie singole restituiscono 1; le geometrie vuote restituiscono 0. Una geometria vuota in un oggetto GEOMETRYCOLLECTION viene conteggiata come una geometria. Ad esempio, la seguente espressione di esempio restituisce 1:

```
ST_NumGeometries(ST_GeometryFromText('GEOMETRYCOLLECTION(MULTIPOINT EMPTY)'))
```

ST_NumInteriorRing(geometry)

Restituisce il numero di anelli interni nella geometria polygon nel tipo bigint. Esempio:

```
SELECT ST_NumInteriorRing(ST_Polygon('polygon ((0 0, 8 0, 0 8, 0 0), (1 1, 1 5, 5 1, 1
 1))'))
```

ST_NumPoints(geometry)

Restituisce il numero di punti nella geometria nel tipo bigint. Esempio:

```
SELECT ST_NumPoints(ST_Point(1.5, 2.5))
```

ST_PointN(lineString, index)

Restituisce, come tipo di dati della geometria puntuale, il vertice della stringa di linea specificata in corrispondenza dell'indice intero specificato. Gli indici iniziano da 1. Se l'indice specificato è minore di 1 o maggiore del numero totale di elementi nell'insieme, restituisce null. Per trovare il numero totale di elementi, usare ST_NumPoints. Esempio:

```
SELECT ST_PointN(ST_LineString(array[ST_Point(1,2), ST_Point(3,4)]),1)
```

ST_Points(geometry)

Restituisce una matrice di punti dall'oggetto geometria specificato della stringa di linea. Esempio:

```
SELECT ST_Points(ST_LineString(array[ST_Point(1,2), ST_Point(3,4)]))
```

ST_StartPoint(geometry)

Restituisce il primo punto di un tipo di dati di geometria line nel tipo di dati della geometria point. Esempio:

```
SELECT ST_StartPoint(ST_Line('linestring(0 2, 2 2)'))
```

ST_X(point)

Restituisce la coordinata X di un punto nel tipo `double`. Esempio:

```
SELECT ST_X(ST_Point(1.5, 2.5))
```

ST_XMax(geometry)

Restituisce la coordinata X massima di una geometria nel tipo `double`. Esempio:

```
SELECT ST_XMax(ST_Line('linestring(0 2, 2 2)'))
```

ST_XMin(geometry)

Restituisce la coordinata X minima di una geometria nel tipo `double`. Esempio:

```
SELECT ST_XMin(ST_Line('linestring(0 2, 2 2)'))
```

ST_Y(point)

Restituisce la coordinata Y di un punto nel tipo `double`. Esempio:

```
SELECT ST_Y(ST_Point(1.5, 2.5))
```

ST_YMax(geometry)

Restituisce la coordinata Y massima di una geometria nel tipo `double`. Esempio:

```
SELECT ST_YMax(ST_Line('linestring(0 2, 2 2)'))
```

ST_YMin(geometry)

Restituisce la coordinata Y minima di una geometria nel tipo `double`. Esempio:

```
SELECT ST_YMin(ST_Line('linestring(0 2, 2 2)'))
```

Funzioni di aggregazione

`convex_hull_agg(geometry)`

Restituisce la geometria convessa minima che racchiude tutte le geometrie trasmesse come input.

`geometry_union_agg(geometry)`

Restituisce una geometria che rappresenta l'unione degli insiemi di punti di tutte le geometrie trasmesse come input.

Funzioni del riquadro Bing

Le seguenti funzioni convertono geometrie e riquadri nel sistema di riquadri Bing Maps di Microsoft.

`bing_tile(x, y, zoom_level)`

Restituisce un oggetto riquadro Bing dalle coordinate intere `x` e `y` e il livello di zoom specificato. Il livello di zoom deve essere un numero intero compreso tra 1 e 23. Esempio:

```
SELECT bing_tile(10, 20, 12)
```

`bing_tile(quadKey)`

Restituisce un oggetto riquadro Bing da un quadkey. Esempio:

```
SELECT bing_tile(bing_tile_quadkey(bing_tile(10, 20, 12)))
```

`bing_tile_at(latitude, longitude, zoom_level)`

Restituisce un oggetto riquadro Bing al livello di latitudine, longitudine e zoom specificati. La latitudine deve essere compresa tra -85.05112878 e 85.05112878. La longitudine deve essere compresa tra -180 e 180. I valori `latitude` e `longitude` devono essere `double` e `zoom_level` deve essere un numero intero. Esempio:

```
SELECT bing_tile_at(37.431944, -122.166111, 12)
```

`bing_tiles_around(latitude, longitude, zoom_level)`

Restituisce un array di riquadri Bing che circondano il punto di latitudine e longitudine specificato al livello di zoom specificato. Esempio:

```
SELECT bing_tiles_around(47.265511, -122.465691, 14)
```

bing_tiles_around(latitude, longitude, zoom_level, radius_in_km)

Restituisce, al livello di zoom specificato, un array di riquadri Bing. L'array contiene l'insieme minimo di riquadri Bing che copre un cerchio del raggio specificato in chilometri attorno alla latitudine e alla longitudine specificate. I valori `latitude`, `longitude` `eradius_in_km` sono `double`; il livello di zoom è un `integer`. Esempio:

```
SELECT bing_tiles_around(37.8475, 112.596667, 10, .5)
```

bing_tile_coordinates(tile)

Restituisce le coordinate `x` e `y` del riquadro Bing specificato. Esempio:

```
SELECT bing_tile_coordinates(bing_tile_at(37.431944, -122.166111, 12))
```

bing_tile_polygon(tile)

Restituisce la rappresentazione del poligono del riquadro Bing specificato. Esempio:

```
SELECT bing_tile_polygon(bing_tile_at(47.265511, -122.465691, 4))
```

bing_tile_quadkey(tile)

Restituisce il quadkey del riquadro Bing specificato. Esempio:

```
SELECT bing_tile_quadkey(bing_tile(52, 143, 10))
```

bing_tile_zoom_level(tile)

Restituisce il livello di zoom del riquadro Bing specificato come numero intero. Esempio:

```
SELECT bing_tile_zoom_level(bing_tile(52, 143, 10))
```

geometry_to_bing_tiles(geometry, zoom_level)

Restituisce l'insieme minimo di riquadri Bing che copre completamente la geometria specificata al livello di zoom specificato. Sono supportati livelli di zoom da 1 a 23. Esempio:

```
SELECT geometry_to_bing_tiles(ST_Point(61.56, 58.54), 10)
```

Modifiche ai nomi delle funzioni geospaziali e nuove funzioni nella versione 2 del motore Athena

Questa sezione elenca le modifiche ai nomi delle funzioni geospaziali e le nuove funzioni geospaziali nella versione 2 del motore Athena. Per informazioni su altre modifiche apportate alla versione 2 del motore Athena, consulta Versione 2 del motore Athena.

Per ulteriori informazioni sulle versioni del motore Athena, consulta Controllo delle versioni del motore di Athena.

Modifiche ai nomi delle funzioni geospaziali nella versione 2 del motore Athena

I nomi delle seguenti funzioni sono cambiati. In alcuni casi, anche i tipi di input e output sono cambiati. Per maggiori informazioni, consulta i link corrispondenti.

Nome della funzione precedente	Nome della funzione a partire dalla versione 2 del motore Athena
st_coordinate_dimension	ST_ CoordDim
st_end_point	ST_ EndPoint
st_exterior_ring	ST_ ExteriorRing
st_interior_ring_number	ST_ NumInteriorRing
st_geometry_from_text	ST_ GeometryFromText
st_is_closed	ST_ IsClosed
st_is_empty	ST_ IsEmpty
st_is_ring	ST_ IsRing
st_max_x	ST_XMax
st_max_y	ST_YMax
st_min_x	ST_XMin

Nome della funzione precedente	Nome della funzione a partire dalla versione 2 del motore Athena
st_min_y	ST_YMin
st_point_number	ST_ NumPoints
st_start_point	ST_ StartPoint
st_symmetric_difference	ST_ SymDifference

Nuove funzioni geospaziali nella versione 2 del motore Athena

Le seguenti funzioni geospaziali sono nuove nella versione 2 del motore Athena. Per maggiori informazioni, consulta i link corrispondenti.

Funzioni costruttore

- ST_ AsBinary
- ST_ GeomAsLegacyBinary
- ST_ GeomFromBinary
- ST_ GeomFromLegacyBinary
- ST_ LineString
- ST_ MultiPoint
- to_geometry
- to_spherical_geography

Funzioni operazione

- eometry_union
- ST_ EnvelopeAsPts
- ST_Union

Funzioni di accesso

- geometry_invalid_reason

- great_circle_distance
- line_locate_point
- simplify_geometry
- ST_ ConvexHull
- ST_Distance (geografia sferica)
- ST_Geometries
- ST_GeometryN
- ST_ GeometryType
- ST_N InteriorRing
- ST_ InteriorRings
- ST_ IsSimple
- ST_ IsValid
- ST_ NumGeometries
- ST_PointN
- ST_Points

Funzioni di aggregazione

- convex_hull_agg
- geometry_union_agg

Funzioni del riquadro Bing

- bing_tile
- bing_tile (quadkey)
- bing_tile_at
- bing_tiles_around
- bing_tiles_around (radius)
- bing_tile_coordinates
- bing_tile_polygon
- bing_tile_quadkey

- bing_tile_zoom_level
- geometry_to_bing_tiles

Esempi: query geospaziali

Gli esempi in questo argomento creano due tabelle a partire dai dati di esempio disponibili GitHub e interrogano le tabelle in base ai dati. I dati di esempio, che sono solo a scopo illustrativo e non sono garantiti per essere accurati, si trovano nei seguenti file:

- **`earthquakes.csv`**: elenca i terremoti che si sono verificati in California. Nella tabella di esempio `earthquakes` vengono utilizzati i campi di questi dati.
- **`california-counties.json`**: elenca i dati della contea per lo stato della California in formato ESRI-compliant GeoJSON. I dati includono molti campi, ad esempio `AREA`, `PERIMETER`, `STATE`, `COUNTY`, e `NAME`, ma nella tabella `counties` di esempio ne vengono utilizzati solo due: `Name` (stringa) e `BoundaryShape` (binario).

> **Note**
>
> Athena utilizza `com.esri.json.hadoop.EnclosedEsriJsonInputFormat` per convertire i dati JSON in formato binario geospaziale.

Il seguente esempio di codice crea una tabella denominata `earthquakes`.

```
CREATE external TABLE earthquakes
(
 earthquake_date string,
 latitude double,
 longitude double,
 depth double,
 magnitude double,
 magtype string,
 mbstations string,
 gap string,
 distance string,
 rms string,
 source string,
 eventid string
)
```

```
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION 's3://DOC-EXAMPLE-BUCKET/my-query-log/csv/';
```

Il seguente esempio di codice crea una tabella denominata `counties`.

```
CREATE external TABLE IF NOT EXISTS counties
 (
 Name string,
 BoundaryShape binary
 )
ROW FORMAT SERDE 'com.esri.hadoop.hive.serde.EsriJsonSerDe'
STORED AS INPUTFORMAT 'com.esri.json.hadoop.EnclosedEsriJsonInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://DOC-EXAMPLE-BUCKET/my-query-log/json/';
```

La seguente query di esempio utilizza la funzione `CROSS JOIN` nelle tabelle `counties` e `earthquake`. L'esempio utilizza `ST_CONTAINS` per eseguire query per le contee i cui confini includono le località dei terremoti, specificate con `ST_POINT`. La query raggruppa quindi gruppi tali paesi per nome, li ordina per conteggio e li restituisce in modo decrescente.

> **Note**
>
> A partire dalla versione 2 del motore Athena, le funzioni come `ST_CONTAINS` non supportano più il tipo `VARBINARY` come input. Per questo motivo, l'esempio utilizza la funzione `ST_GeomFromLegacyBinary(varbinary)` per convertire il valore binario `boundaryshape` in una geometria. Per ulteriori informazioni, consulta Modifiche alle funzioni geospaziali nella referenza per Versione 2 del motore Athena.

```
SELECT counties.name,
        COUNT(*) cnt
FROM counties
CROSS JOIN earthquakes
WHERE ST_CONTAINS (ST_GeomFromLegacyBinary(counties.boundaryshape),
 ST_POINT(earthquakes.longitude, earthquakes.latitude))
GROUP BY  counties.name
ORDER BY  cnt DESC
```

Questa query restituisce:

```
+-----------------------+
| name             | cnt |
+-----------------------+
| Kern             | 36  |
+-----------------------+
| San Bernardino   | 35  |
+-----------------------+
| Imperial         | 28  |
+-----------------------+
| Inyo             | 20  |
+-----------------------+
| Los Angeles      | 18  |
+-----------------------+
| Riverside        | 14  |
+-----------------------+
| Monterey         | 14  |
+-----------------------+
| Santa Clara      | 12  |
+-----------------------+
| San Benito       | 11  |
+-----------------------+
| Fresno           | 11  |
+-----------------------+
| San Diego        | 7   |
+-----------------------+
| Santa Cruz       | 5   |
+-----------------------+
| Ventura          | 3   |
+-----------------------+
| San Luis Obispo  | 3   |
+-----------------------+
| Orange           | 2   |
+-----------------------+
| San Mateo        | 1   |
+-----------------------+
```

Risorse aggiuntive

Per altri esempi di query geospaziali, consulta i seguenti post di blog:

- Estendi le query geospaziali in Amazon Athena con UDF e AWS Lambda
- Visualizza oltre 200 anni di dati climatici globali utilizzando Amazon Athena e Amazon. QuickSight

- Interrogazioni OpenStreetMap con Amazon Athena

Esecuzione di query su JSON

Amazon Athena consente di interrogare dati con codifica JSON, estrarre dati da JSON annidato, cercare valori e trovare la lunghezza e la dimensione degli array JSON. Per apprendere le nozioni di base sull'interrogazione dei dati JSON in Athena, prendi in considerazione i seguenti dati planetari di esempio:

```
{name:"Mercury",distanceFromSun:0.39,orbitalPeriod:0.24,dayLength:58.65}
{name:"Venus",distanceFromSun:0.72,orbitalPeriod:0.62,dayLength:243.02}
{name:"Earth",distanceFromSun:1.00,orbitalPeriod:1.00,dayLength:1.00}
{name:"Mars",distanceFromSun:1.52,orbitalPeriod:1.88,dayLength:1.03}
```

Notate come ogni record (essenzialmente, ogni riga della tabella) si trovi su una riga distinta. Per interrogare questi dati JSON, puoi usare un'`CREATE TABLE`istruzione come la seguente:

```
CREATE EXTERNAL TABLE `planets_json`(
  `name` string,
  `distancefromsun` double,
  `orbitalperiod` double,
  `daylength` double)
ROW FORMAT SERDE
  'org.openx.data.jsonserde.JsonSerDe'
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET/json/'
```

Per interrogare i dati, utilizzate un'`SELECT`istruzione semplice come nell'esempio seguente.

```
SELECT * FROM planets_json
```

I risultati della query sono simili ai seguenti.

#	nome	distanza dal sole	periodo orbitale	durata del giorno
1	mercurio	0,39	0,24	58,65
2	Venere	0,72	0,62	243,02
3	Terra	1.0	1.0	1
4	Marte	1,52	1,88	1,03

Nota come l'`CREATE TABLE`istruzione utilizza ilOpenX JSON SerDe, che richiede che ogni record JSON si trovi su una riga separata. Se il JSON è in un bel formato di stampa o se tutti i record sono su una sola riga, i dati non verranno letti correttamente.

Per interrogare dati JSON in un bel formato di stampa, puoi usare Amazon Ion Hive SerDe invece di SerDe OpenX JSON. Considerate i dati precedenti memorizzati in un grazioso formato di stampa:

```
{
  name:"Mercury",
  distanceFromSun:0.39,
  orbitalPeriod:0.24,
  dayLength:58.65
}
{
  name:"Venus",
  distanceFromSun:0.72,
  orbitalPeriod:0.62,
  dayLength:243.02
}
{
  name:"Earth",
  distanceFromSun:1.00,
  orbitalPeriod:1.00,
  dayLength:1.00
}
{
  name:"Mars",
  distanceFromSun:1.52,
  orbitalPeriod:1.88,
  dayLength:1.03
```

```
}
```

Per interrogare questi dati senza riformattarli, puoi usare un'`CREATE TABLE`istruzione come la seguente. Nota che, invece di specificare OpenX SerDe JSON, l'istruzione specifica. `STORED AS ION`

```
CREATE EXTERNAL TABLE `planets_ion`(
  `name` string,
  `distancefromsun` DECIMAL(10, 2),
  `orbitalperiod` DECIMAL(10, 2),
  `daylength` DECIMAL(10, 2))
STORED AS ION
LOCATION
  's3://DOC-EXAMPLE-BUCKET/json-ion/'
```

La query `SELECT * FROM planets_ion` produce gli stessi risultati di prima. Per ulteriori informazioni sulla creazione di tabelle in questo modo utilizzando Amazon Ion Hive SerDe, consultaUtilizzo di CREATE TABLE per creare tabelle Amazon Ion.

L'esempio precedente: i dati JSON non contengono tipi di dati complessi come array o strutture annidate. Per ulteriori informazioni sull'interrogazione di dati JSON annidati, consulta. Esempio di deserializzazione di un JSON nidificato

Argomenti

- Best practice per la lettura di dati JSON
- Estrazione di dati JSON dalle stringhe
- Ricerca di valori nelle matrici JSON
- Come ottenere matrici JSON di lunghezza e dimensione
- Risoluzione dei problemi relativi alle query JSON

Best practice per la lettura di dati JSON

JavaScript Object Notation (JSON) è un metodo comune per codificare le strutture di dati come testo. Molte applicazioni e numerosi strumenti restituiscono dati con codifica JSON.

In Amazon Athena, è possibile creare tabelle da dati esterni e includervi i dati con codifica JSON. Per tali tipi di dati di origine, utilizza Athena insieme a Librerie JSON SerDe .

Segui i seguenti suggerimenti per leggere i dati con codifica JSON:

- Scegli quello giusto SerDe, un JSON nativo o un SerDe `org.openx.data.jsonserde.JsonSerDe` OpenX SerDe`org.apache.hive.hcatalog.data.JsonSerDe`,. Per ulteriori informazioni, consulta Librerie JSON SerDe .
- Assicurati che ogni record con codifica JSON sia rappresentato su una riga separata, non formattata.

> **Note**
>
> SerDe Si aspetta che ogni documento JSON si trovi su una singola riga di testo senza caratteri di terminazione di riga che separano i campi del record. Se il testo JSON è in un bel formato di stampa, potresti ricevere un messaggio di errore come HIVE_CURSOR_ERROR: Row is not a valid JSON Object o HIVE_CURSOR_ERROR:: Unexpected end-of-input: expected: expected close marker for OBJECT quando tenti di interrogare la tabella dopo averla JsonParseException creata. Per ulteriori informazioni, consulta JSON Data Files nella documentazione di SerDe OpenX su. GitHub

- Genera i dati con codifica JSON in colonne con distinzione tra maiuscole e minuscole.
- Fornisci un'opzione per ignorare i record in formato errato, come in questo esempio.

```
CREATE EXTERNAL TABLE json_table (
  column_a string,
  column_b int
 )
 ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
 WITH SERDEPROPERTIES ('ignore.malformed.json' = 'true')
 LOCATION 's3://DOC-EXAMPLE-BUCKET/path/';
```

- Converti i campi nei dati di origine che hanno uno schema indeterminato in stringhe con codifica JSON in Athena.

Quando Athena crea tabelle basate su dati con codifica JSON, analizza i dati in base allo schema esistente e predefinito. Tuttavia, è possibile che non tutti i dati presentino uno schema predefinito. In questi casi, per semplificare la gestione degli schemi, spesso è consigliabile convertire i campi in dati di origine che dispongano di uno schema indeterminato su stringhe JSON in Athena, per poi utilizzare Librerie JSON SerDe .

Prendiamo ad esempio in considerazione un'applicazione IoT che pubblica eventi con campi comuni da diversi sensori. Uno di tali campi devono archiviare un payload personalizzato univoco per il sensore che invia l'evento. In questo caso, poiché lo schema non è noto, è consigliabile archiviare le informazioni in forma di stringa con codifica JSON. A tale scopo, occorre convertire i dati nella tabella Athena in JSON, come nell'esempio seguente. È inoltre possibile convertire i dati con codifica JSON in tipi di dati Athena.

- Conversione di tipi di dati Athena in JSON
- Conversione di tipi di dati JSON in Athena

Conversione di tipi di dati Athena in JSON

Per convertire i tipi di dati Athena in JSON, utilizza `CAST`.

```
WITH dataset AS (
  SELECT
    CAST('HELLO ATHENA' AS JSON) AS hello_msg,
    CAST(12345 AS JSON) AS some_int,
    CAST(MAP(ARRAY['a', 'b'], ARRAY[1,2]) AS JSON) AS some_map
)
SELECT * FROM dataset
```

Questa query restituisce:

```
+-----------------------------------------+
| hello_msg      | some_int | some_map      |
+-----------------------------------------+
| "HELLO ATHENA" | 12345    | {"a":1,"b":2} |
+-----------------------------------------+
```

Conversione di tipi di dati JSON in Athena

Per convertire i dati JSON nei tipi di dati Athena, utilizza `CAST`.

> **Note**
>
> In questo esempio, per indicare le stringhe con codifica JSON, inizia con la parola chiave `JSON` e utilizza virgolette singole, ad esempio `JSON '12345'`

```
WITH dataset AS (
  SELECT
    CAST(JSON '"HELLO ATHENA"' AS VARCHAR) AS hello_msg,
    CAST(JSON '12345' AS INTEGER) AS some_int,
    CAST(JSON '{"a":1,"b":2}' AS MAP(VARCHAR, INTEGER)) AS some_map
)
SELECT * FROM dataset
```

Questa query restituisce:

```
+-------------------------------------+
| hello_msg    | some_int | some_map  |
+-------------------------------------+
| HELLO ATHENA | 12345    | {a:1,b:2} |
+-------------------------------------+
```

Estrazione di dati JSON dalle stringhe

È possibile che alcuni dei dati di origine che contengono stringhe codificate JSON non debbano essere necessariamente deserializzate in una tabella Athena. In tal caso, è comunque possibile eseguire operazioni SQL su questi dati utilizzando le funzioni JSON disponibili in Presto.

Considera questa stringa JSON come un set di dati di esempio.

```
{"name": "Susan Smith",
"org": "engineering",
"projects":
    [
     {"name":"project1", "completed":false},
     {"name":"project2", "completed":true}
    ]
}
```

Esempi: proprietà di estrazione

Per estrarre le proprietà `name` e `projects` dalla stringa JSON, utilizza la funzione `json_extract`, come nell'esempio seguente. La funzione `json_extract` analizza la colonna contenente la stringa JSON e vi esegue la ricerca impiegando un'espressione LIKE `JSONPath` con la notazione del punto `.`.

> **Note**
>
> `JSONPath` esegue un attraversamento semplice della struttura. utilizzando il simbolo $ per indicare la radice del documento JSON, seguita da un punto e da un elemento nidificato direttamente sotto il livello radice, ad esempio `$.name`.

```
WITH dataset AS (
  SELECT '{"name": "Susan Smith",
           "org": "engineering",
           "projects": [{"name":"project1", "completed":false},
           {"name":"project2", "completed":true}]}'
    AS myblob
)
SELECT
  json_extract(myblob, '$.name') AS name,
  json_extract(myblob, '$.projects') AS projects
FROM dataset
```

Il valore restituito è una stringa con codifica JSON e non un tipo di dati Athena nativo.

```
+-----------------------------------------------------------------------------------------------
+
| name           | projects
         |
+-----------------------------------------------------------------------------------------------
+
| "Susan Smith"  | [{"name":"project1","completed":false},
{"name":"project2","completed":true}] |
+-----------------------------------------------------------------------------------------------
+
```

Per estrarre il valore scalare dalla stringa JSON, utilizza la funzione `json_extract_scalar`. Quest'ultima è simile a `json_extract`, ma restituisce solo valori scalari (booleani, numeri o stringhe).

> **Note**
>
> Non utilizzare la funzione `json_extract_scalar` su matrici, mappe o strutture.

```
WITH dataset AS (
  SELECT '{"name": "Susan Smith",
           "org": "engineering",
           "projects": [{"name":"project1", "completed":false},{"name":"project2",
 "completed":true}]}'
    AS myblob
)
SELECT
  json_extract_scalar(myblob, '$.name') AS name,
  json_extract_scalar(myblob, '$.projects') AS projects
FROM dataset
```

Questa query restituisce:

```
+---------------------------+
| name           | projects |
+---------------------------+
| Susan Smith    |          |
+---------------------------+
```

Per ottenere il primo elemento della proprietà `projects` nella matrice di esempio, utilizza la funzione `json_array_get` e specifica la posizione indice.

```
WITH dataset AS (
  SELECT '{"name": "Bob Smith",
           "org": "engineering",
           "projects": [{"name":"project1", "completed":false},{"name":"project2",
 "completed":true}]}'
    AS myblob
)
SELECT json_array_get(json_extract(myblob, '$.projects'), 0) AS item
FROM dataset
```

Restituirà il valore nella posizione indice specificata nella matrice codificata JSON.

```
+---------------------------------------+
| item                                  |
+---------------------------------------+
| {"name":"project1","completed":false} |
+---------------------------------------+
```

Per restituire un tipo di stringa Athena, utilizza l'operatore `[]` all'interno di un'espressione `JSONPath`, quindi utilizza la funzione `json_extract_scalar`. Per ulteriori informazioni su `[]`, consulta Accesso agli elementi della matrice.

```
WITH dataset AS (
   SELECT '{"name": "Bob Smith",
             "org": "engineering",
             "projects": [{"name":"project1", "completed":false},{"name":"project2",
 "completed":true}]}'
     AS myblob
)
SELECT json_extract_scalar(myblob, '$.projects[0].name') AS project_name
FROM dataset
```

Tale operazione restituisce questo risultato:

```
+--------------+
| project_name |
+--------------+
| project1     |
+--------------+
```

Ricerca di valori nelle matrici JSON

Per determinare se un determinato valore esiste all'interno di una matrice con codifica JSON, utilizza la funzione `json_array_contains`.

La query seguente elenca i nomi degli utenti che partecipano a "project2".

```
WITH dataset AS (
  SELECT * FROM (VALUES
    (JSON '{"name": "Bob Smith", "org": "legal", "projects": ["project1"]}'),
    (JSON '{"name": "Susan Smith", "org": "engineering", "projects": ["project1",
 "project2", "project3"]}'),
    (JSON '{"name": "Jane Smith", "org": "finance", "projects": ["project1",
 "project2"]}')
  ) AS t (users)
)
SELECT json_extract_scalar(users, '$.name') AS user
FROM dataset
WHERE json_array_contains(json_extract(users, '$.projects'), 'project2')
```

Questa query restituisce un elenco di utenti.

```
+-------------+
| user        |
+-------------+
| Susan Smith |
+-------------+
| Jane Smith  |
+-------------+
```

La seguente query di esempio elenca i nomi degli utenti che hanno completato progetti con il numero totale di progetti completati. Esegue le seguenti operazioni:

- Utilizza istruzioni SELECT nidificate per chiarezza.
- Estrae la matrice di progetti.
- Converte la matrice in una matrice nativa di coppie chiave-valore utilizzando CAST.
- Estrae ogni singolo elemento di matrice utilizzando l'operatore UNNEST.
- Filtra i valori ottenuti in base ai progetti completati e li conta.

> **Note**
>
> Quando si utilizza CAST su MAP, è possibile specificare l'elemento chiave come VARCHAR (stringa nativa in Presto), lasciando però il valore come JSON, perché i valori in MAP sono di tipi diversi: String per la prima coppia chiave-valore e Boolean per il secondo.

```
WITH dataset AS (
  SELECT * FROM (VALUES
    (JSON '{"name": "Bob Smith",
            "org": "legal",
            "projects": [{"name":"project1", "completed":false}]}'),
    (JSON '{"name": "Susan Smith",
            "org": "engineering",
            "projects": [{"name":"project2", "completed":true},
                         {"name":"project3", "completed":true}]}'),
    (JSON '{"name": "Jane Smith",
            "org": "finance",
            "projects": [{"name":"project2", "completed":true}]}')
  ) AS t (users)
```

```
),
employees AS (
  SELECT users, CAST(json_extract(users, '$.projects') AS
    ARRAY(MAP(VARCHAR, JSON))) AS projects_array
  FROM dataset
),
names AS (
  SELECT json_extract_scalar(users, '$.name') AS name, projects
  FROM employees, UNNEST (projects_array) AS t(projects)
)
SELECT name, count(projects) AS completed_projects FROM names
WHERE cast(element_at(projects, 'completed') AS BOOLEAN) = true
GROUP BY name
```

Questa query restituisce il seguente risultato:

```
+----------------------------------+
| name        | completed_projects |
+----------------------------------+
| Susan Smith | 2                  |
+----------------------------------+
| Jane Smith  | 1                  |
+----------------------------------+
```

Come ottenere matrici JSON di lunghezza e dimensione

Esempio: **`json_array_length`**

Per ottenere la lunghezza di una matrice con codifica JSON, utilizza la funzione `json_array_length`.

```
WITH dataset AS (
  SELECT * FROM (VALUES
    (JSON '{"name":
            "Bob Smith",
            "org":
            "legal",
            "projects": [{"name":"project1", "completed":false}]}'),
    (JSON '{"name": "Susan Smith",
            "org": "engineering",
            "projects": [{"name":"project2", "completed":true},
                         {"name":"project3", "completed":true}]}'),
```

```
    (JSON '{"name": "Jane Smith",
             "org": "finance",
             "projects": [{"name":"project2", "completed":true}]}')
  ) AS t (users)
)
SELECT
  json_extract_scalar(users, '$.name') as name,
  json_array_length(json_extract(users, '$.projects')) as count
FROM dataset
ORDER BY count DESC
```

Questa query restituisce il risultato seguente:

```
+---------------------+
| name        | count |
+---------------------+
| Susan Smith | 2     |
+---------------------+
| Bob Smith   | 1     |
+---------------------+
| Jane Smith  | 1     |
+---------------------+
```

Esempio: **`json_size`**

Per ottenere le dimensioni di una matrice o di oggetto con codifica JSON, utilizza la funzione `json_size` e specifica la colonna contenente la stringa JSON e l'espressione `JSONPath` per la matrice o l'oggetto.

```
WITH dataset AS (
  SELECT * FROM (VALUES
    (JSON '{"name": "Bob Smith", "org": "legal", "projects": [{"name":"project1",
 "completed":false}]}'),
    (JSON '{"name": "Susan Smith", "org": "engineering", "projects":
 [{"name":"project2", "completed":true},{"name":"project3", "completed":true}]}'),
    (JSON '{"name": "Jane Smith", "org": "finance", "projects": [{"name":"project2",
 "completed":true}]}')
  ) AS t (users)
)
SELECT
  json_extract_scalar(users, '$.name') as name,
  json_size(users, '$.projects') as count
```

```
FROM dataset
ORDER BY count DESC
```

Questa query restituisce il risultato seguente:

```
+---------------------+
| name        | count |
+---------------------+
| Susan Smith | 2     |
+---------------------+
| Bob Smith   | 1     |
+---------------------+
| Jane Smith  | 1     |
+---------------------+
```

Risoluzione dei problemi relativi alle query JSON

Per assistenza sulla risoluzione dei problemi relativi alle query relative a JSON, consulta Errori correlati a JSON o le risorse seguenti:

- Ricevo messaggi di errore quando cerco di leggere dati JSON su Amazon Athena
- Come posso risolvere «HIVE_CURSOR_ERROR: Row is not a valid JSON object - JsonException: Duplicate key» durante la lettura di file da Athena? AWS Config
- La query SELECT COUNT in Amazon Athena restituisce solo un record, anche se il file di input JSON dispone di più record
- In che modo è possibile visualizzare il file sorgente Amazon S3 per cercare una riga di una tabella Athena?

Consulta anche Considerazioni e restrizioni per le query SQL in Amazon Athena.

Utilizzo di Machine Learning (ML) con Amazon Athena

Machine Learning (ML) con Amazon Athena ti consente di utilizzare Athena per scrivere istruzioni SQL che eseguono inferenze di Machine Learning (ML) utilizzando Amazon. SageMaker Questa caratteristica semplifica l'accesso ai modelli ML per l'analisi dei dati, eliminando la necessità di utilizzare metodi di programmazione complessi per eseguire l'inferenza.

Per usare ML con Athena, si definisce un ML con la funzione Athena con la clausola `USING EXTERNAL FUNCTION`. La funzione punta all'endpoint del SageMaker modello che desideri utilizzare

e specifica i nomi delle variabili e i tipi di dati da passare al modello. Le clausole successive nella query fanno riferimento alla funzione per passare valori al modello. Il modello esegue l'inferenza in base ai valori passati dalla query e quindi restituisce i risultati dell'inferenza. Per ulteriori informazioni SageMaker e su come funzionano gli SageMaker endpoint, consulta l'Amazon SageMaker Developer Guide.

Per un esempio che utilizza ML con Athena e SageMaker inferenza per rilevare un valore anomalo in un set di risultati, consulta l'articolo del AWS Big Data Blog Rilevare valori anomali richiamando la funzione di inferenza di apprendimento automatico di Amazon Athena.

Considerazioni e limitazioni

- Regioni disponibili: la funzionalità Athena ML è una funzionalità in cui è supportata la versione 2 o successiva del motore Regioni AWS Athena.
- SageMaker model endpoint must accept and return **`text/csv`** — Per ulteriori informazioni sui formati di dati, consulta Common data formats for inference nella Amazon SageMaker Developer Guide.
- Athena non invia intestazioni CSV: se l' SageMaker endpoint lo è`text/csv`, il gestore di input non deve presumere che la prima riga dell'input sia un'intestazione CSV. Poiché Athena non invia intestazioni CSV, l'output restituito ad Athena conterrà una riga in meno di quella prevista da Athena e causerà un errore.
- SageMaker scalabilità dell'endpoint: assicuratevi che l'endpoint del SageMaker modello di riferimento sia sufficientemente scalato per le chiamate Athena all'endpoint. Per ulteriori informazioni, consulta SageMaker i modelli di scalabilità automatica nell'Amazon SageMaker Developer Guide e CreateEndpointConfignell'Amazon SageMaker API Reference.
- Autorizzazioni IAM: per eseguire una query che specifica un ML con la funzione Athena, è necessario consentire al principale IAM che esegue la query di eseguire l'`sagemaker:InvokeEndpoint`azione per l'endpoint del modello di riferimento. SageMaker Per ulteriori informazioni, consulta Autorizzazione per l'accesso per ML con Athena.
- ML con funzioni Athena non può essere usato direttamente nelle clausole **`GROUP BY`**

ML con sintassi Athena

La clausola `USING EXTERNAL FUNCTION` specifica un ML con funzione o più funzioni Athena a cui è possibile fare riferimento da un'istruzione `SELECT` successiva nella query. Definire il nome della funzione, i nomi delle variabili e i tipi di dati per le variabili e i valori restituiti.

Riepilogo

La seguente sintassi mostra una clausola `USING EXTERNAL FUNCTION` che specifica un ML con funzione Athena.

```
USING EXTERNAL FUNCTION ml_function_name (variable1 data_type[, variable2 data_type]
[,...])
RETURNS data_type
SAGEMAKER 'sagemaker_endpoint'
SELECT ml_function_name()
```

Parametri

USING EXTERNAL FUNCTION ***ml_function_name*** (***variable1 data_type***[, ***variable2 data_type***][,...])

ml_function_name definisce il nome della funzione, che può essere utilizzato nelle successive clausole di query. Ogni *variabile data_type* specifica una variabile denominata e il tipo di dati corrispondente che il modello accetta come input. SageMaker Il tipo di dati specificato deve essere un tipo di dati Athena supportato.

RETURNS ***data_type***

data_type specifica il tipo di dati SQL che *ml_function_name* restituisce alla query come output del modello. SageMaker

SAGEMAKER '***sagemaker_endpoint***'

sagemaker_endpoint specifica l'endpoint del modello. SageMaker

SELEZIONA [...] ***ml_function_name***(***expression***) [...]

La query SELECT che passa i valori alle variabili di funzione e al modello per restituire un risultato. SageMaker *ml_function_name* specifica la funzione definita in precedenza nella query, seguita da un'*espressione* che viene valutata per passare valori. I valori passati e restituiti devono corrispondere ai tipi di dati corrispondenti specificati per la funzione nella clausola `USING EXTERNAL FUNCTION`.

Esempio

Nell'esempio seguente viene illustrata una query mediante ML con Athena.

Example

```
USING EXTERNAL FUNCTION predict_customer_registration(age INTEGER)
    RETURNS DOUBLE
    SAGEMAKER 'xgboost-2019-09-20-04-49-29-303'
SELECT predict_customer_registration(age) AS probability_of_enrolling, customer_id
     FROM "sampledb"."ml_test_dataset"
     WHERE predict_customer_registration(age) < 0.5;
```

Esempi di utilizzo per i clienti

I seguenti video, che utilizzano la versione di anteprima di Machine Learning (ML) con Amazon Athena, mostrano i modi in cui è possibile utilizzare Athena. SageMaker

Previsione dell'abbandono del cliente

Il video seguente mostra come combinare Athena con le funzionalità di machine learning di Amazon SageMaker per prevedere il tasso di abbandono dei clienti.

Prevedi il tasso di abbandono dei clienti utilizzando Amazon Athena e Amazon SageMaker

Rilevamento delle botnet

Il video seguente mostra come un'azienda utilizza Amazon Athena e Amazon SageMaker per rilevare le botnet.

Rileva le botnet utilizzando Amazon Athena e Amazon SageMaker

Esecuzione di query con funzioni definite dall'utente

Funzioni definite dall'utente (FDU) in Amazon Athena consentono di creare funzioni personalizzate per elaborare registri o gruppi di registri. Una funzione definita dall'utente accetta parametri, esegue il lavoro e quindi restituisce un risultato.

Per utilizzare una funzione definita dall'utente in Athena, scrivi una clausola `USING EXTERNAL FUNCTION` prima di un'istruzione `SELECT` in una query SQL. L'istruzione `SELECT` fa riferimento alla funzione definita dall'utente e definisce le variabili che vengono passate alla funzione definita dall'utente quando viene eseguita la query. La query SQL richiama una funzione Lambda utilizzando il runtime Java quando chiama la funzione definita dall'utente. Le funzioni definite dall'utente sono definite all'interno della funzione Lambda come metodi in un pacchetto di distribuzione Java. Più

funzioni definite dall'utente possono essere definite nello stesso pacchetto di distribuzione Java per una funzione Lambda. Puoi anche specificare il nome della funzione Lambda nella clausola `USING EXTERNAL FUNCTION`.

Sono disponibili due opzioni per la distribuzione di una funzione Lambda per le funzioni Athena definite dall'utente. Puoi distribuire la funzione direttamente utilizzando Lambda oppure puoi utilizzare AWS Serverless Application Repository. Per trovare le funzioni Lambda esistenti per le UDF, puoi cercare nell'archivio pubblico AWS Serverless Application Repository o privato e quindi distribuirle su Lambda. Puoi inoltre creare o modificare il codice sorgente Java, creare un pacchetto in un file JAR e distribuirlo utilizzando Lambda o il AWS Serverless Application Repository. Per esempi di codice origine e pacchetti Java con cui iniziare, consulta Creazione e distribuzione di una funzione definita dall'utente utilizzando Lambda. Per ulteriori informazioni su Lambda, consulta la Guida per gli sviluppatori di AWS Lambda. Per ulteriori informazioni in merito, consulta la Guida per gli AWS Serverless Application Repository sviluppatori.AWS Serverless Application Repository

Per un esempio che utilizza UDF con Athena per tradurre e analizzare il testo, consulta AWS l'articolo del Machine Learning Blog Tradurre e analizzare il testo utilizzando le funzioni SQL con Amazon Athena, Amazon Translate e Amazon Comprehend oppure guarda il. video

Per un esempio di utilizzo delle UDF per l'estensione delle query geospaziali in Amazon Athena, consulta Estensione delle query geospaziali in Amazon Athena con UDF e AWS Lambda nel Blog sui Big Data di AWS .

Considerazioni e limitazioni

- Funzioni Athena integrate: le funzioni integrate in Athena sono progettate per essere altamente performanti. Ti consigliamo di utilizzare funzioni incorporate anziché funzioni definite dall'utente quando possibile. Per ulteriori informazioni sulle funzioni incorporate, consulta Funzioni in Amazon Athena.
- Solo UDF scalari — Athena supporta solo le UDF scalari, che elaborano una riga alla volta e restituiscono un singolo valore di colonna. Athena passa un batch di righe, potenzialmente in parallelo, all'UDF ogni volta che invoca Lambda. Durante la progettazione di funzioni definite dall'utente e query, sii consapevole del potenziale impatto sul traffico di rete di questo processo.
- Le funzioni del gestore UDF utilizzano un formato abbreviato: usa il formato abbreviato (non il formato completo) per le funzioni UDF (ad esempio, `package.Class` anziché `package.Class::method`).
- I metodi UDF devono essere scritti in lettere minuscole: i metodi UDF devono essere scritti in lettere minuscole; la notazione a cammello non è consentita.

- I metodi UDF richiedono parametri: i metodi UDF devono avere almeno un parametro di input. Il tentativo di richiamare un file UDF definito senza parametri di input comporta un'eccezione di runtime. Le UDF sono concepite per eseguire funzioni su record di dati, ma una UDF senza argomenti non accetta dati, quindi si verifica un'eccezione.
- Supporto per Java Runtime— Attualmente, le UDF Athena supportano Java Runtime 8 e 11 per Lambda. Per ulteriori informazioni, consulta Creazione di funzioni Lambda con Java nella Guida per gli sviluppatori di AWS Lambda .
- Autorizzazioni IAM — Per eseguire e creare istruzioni di query UDF in Athena, l'entità principale IAM che esegue la query deve essere autorizzata a eseguire operazioni in aggiunta alle funzioni Athena. Per ulteriori informazioni, consulta Esempio di policy di autorizzazione IAM per consentire le funzioni definite dall'utente (UDF) di Amazon Athena.
- Quote di Lambda — Le quote lambda si applicano alle UDF. Per ulteriori informazioni, consulta la sezione Quote Lambda nella Guida per gli sviluppatori di AWS Lambda .
- Filtraggio a livello di riga: il filtro a livello di riga di Lake Formation non è supportato per le UDF.
- Visualizzazioni: non è possibile utilizzare le visualizzazioni con UDF.
- Problemi noti: per l' up-to-dateelenco più completo dei problemi noti, consulta Limitazioni e problemi nella sezione awslabs/ di. aws-athena-query-federation GitHub

Video

Guarda i video seguenti per sapere di più sull'utilizzo delle UDF in Athena.

Video: Introduzione delle funzioni definite dall'utente (UDF) in Amazon Athena

Il video seguente mostra come utilizzare le UDF in Amazon Athena per redigere informazioni sensibili.

> **Note**
>
> La sintassi in questo video è precedente al rilascio, ma i concetti sono gli stessi. Utilizza Athena senza il gruppo di lavoro `AmazonAthenaPreviewFunctionality`.

Introduzione delle funzioni definite dall'utente (UDF) in Amazon Athena

Video: Tradurre, analizzare e redigere campi di testo utilizzando query SQL in Amazon Athena

Il seguente video mostra come utilizzare le UDF in Amazon Athena insieme ad altri Servizi AWS per tradurre e analizzare il testo.

> **Note**
>
> La sintassi in questo video è precedente al rilascio, ma i concetti sono gli stessi. Per la sintassi corretta, vedere il post del blog correlato Tradurre, redigere e analizzare il testo utilizzando le funzioni SQL con Amazon Athena, Amazon Translate e Amazon Comprehend sulblog AWS Machine Learning.

Tradurre, analizzare e redigere campi di testo utilizzando query SQL in Amazon Athena

Sintassi query funzione definita dall'utente

La clausola `USING EXTERNAL FUNCTION` specifica una o più funzioni definite dall'utente a cui è possibile fare riferimento da un'istruzione `SELECT` successiva nella query. È necessario il nome del metodo per la funzione definita dall'utente e il nome della funzione Lambda che ospita la funzione definita dall'utente. Al posto del nome della funzione Lambda, puoi utilizzare l'ARN Lambda. Negli scenari con più account, è necessario l'utilizzo dell'ARN Lambda.

Riepilogo

```
USING EXTERNAL FUNCTION UDF_name(variable1 data_type[, variable2 data_type][,...])
RETURNS data_type
LAMBDA 'lambda_function_name_or_ARN'
[, EXTERNAL FUNCTION UDF_name2(variable1 data_type[, variable2 data_type][,...])
RETURNS data_type
LAMBDA 'lambda_function_name_or_ARN'[,...]]
SELECT  [...] UDF_name(expression) [, UDF_name2(expression)] [...]
```

Parametri

UTILIZZO DELLA FUNZIONE ESTERNA ***UDF_name***(***variable1 data_type***[, ***variable2 data_type***][,...])

UDF_name specifica il nome della funzione definita dall'utente, che deve corrispondere a un metodo Java all'interno della funzione Lambda di riferimento. Ogni *variabile data_type* specifica una variabile denominata e il tipo di dati corrispondente, che l'UDF accetta come input.

data_type deve essere uno dei tipi di dati Athena supportati nella seguente tabella e mappato al tipo di dati Java corrispondente.

Tipo di dati Athena	Tipo di dati Java
TIMESTAMP	java.time. LocalDateTime (UTC)
DATE	java.time. LocalDate (UTC)
TINYINT	java.lang.Byte
SMALLINT	java.lang.Short
REAL	java.lang.Float
DOUBLE	java.lang.Double
DECIMAL (vd. nota RETURNS)	java.matematica. BigDecimal
BIGINT	java.lang.Long
INTEGER	java.lang.Int
VARCHAR	java.lang.String
VARBINARY	byte[]
BOOLEAN	java.lang.Boolean
ARRAY	java.util.List
ROW	java.util.Map<String, Object>

RETURNS *data_type*

`data_type` specifica il tipo di dati SQL restituito dall'UDF come output. I tipi di dati Athena elencati nella tabella precedente sono supportati. Per il tipo di dati `DECIMAL`, utilizzare la sintassi `RETURNS DECIMAL(`*precision*`, `*scale*`)` dove *precision* e *scale* sono numeri interi.

LAMBDA '***lambda_function***'

lambda_function specifica il nome della funzione da richiamare durante l'esecuzione della funzione Lambda definita dall'utente.

SELEZIONA [...] ***UDF_name***(***expression***) [...]

La query `SELECT` che trasmette i valori all'UDF e restituisce un risultato. *UDF_name* specifica l'UDF da utilizzare, seguito da un'*espressione* che viene valutata per trasmettere i valori. I valori passati e restituiti devono corrispondere ai tipi di dati corrispondenti specificati per la funzione definita dall'utente nella clausola `USING EXTERNAL FUNCTION`.

Esempi

Per le query ad esempio basate sul codice AthenaUDFHandler.java su GitHub, consulta la pagina del connettore GitHub Amazon Athena UDF.

Creazione e distribuzione di una funzione definita dall'utente utilizzando Lambda

Per creare una funzione definita dall'utente personalizzata, crea una nuova classe Java estendendo la classe `UserDefinedFunctionHandler`. Il codice sorgente del UserDefinedFunctionHandlerfile.java nell'SDK è disponibile nel athena-federation-sdk repository awslabs/aws-athena-query-federation/, insieme GitHub a esempi di implementazioni UDF che puoi esaminare e modificare per creare una UDF personalizzata.

La procedura in questa sezione illustra la scrittura e la creazione di un file Jar della funzione definita dall'utente personalizzata utilizzando Apache Maven dalla riga di comando e una distribuzione.

Procedura per creare una funzione definita dall'utente personalizzata per Athena con Maven

- Duplicazione dell'SDK e preparazione dell'ambiente di sviluppo
- Creazione del progetto Maven
- Aggiunta di dipendenze e plugin al progetto Maven
- Scrittura di codice Java per le funzioni definite dall'utente
- Compilazione del file JAR
- Distribuisci il JAR su AWS Lambda

Duplicazione dell'SDK e preparazione dell'ambiente di sviluppo

Prima di iniziare, assicurati che git sia installato sul tuo sistema utilizzando `sudo yum install git -y`.

AWS Per installare l'SDK di federazione delle query

- Immettere quanto segue nella riga di comando per clonare il repository SDK. Questo repository include l'SDK, esempi e una suite di connettori di origine dati. Per ulteriori informazioni sui connettori di origine dati, consulta Utilizzo di Amazon Athena Federated Query.

```
git clone https://github.com/awslabs/aws-athena-query-federation.git
```

Per installare i prerequisiti per questa procedura

Se state lavorando su una macchina di sviluppo su cui sono già installati Apache Maven AWS CLI, lo strumento di AWS Serverless Application Model compilazione e sviluppo, potete saltare questo passaggio.

1. Dalla radice della directory `aws-athena-query-federation` creata durante la clonazione, esegui lo script prepare_dev_env.sh che prepara l'ambiente di sviluppo.
2. Aggiorna la shell per generare nuove variabili create dal processo di installazione o riavvia la sessione del terminale.

```
source ~/.profile
```

> **Important**
>
> Se salti questo passaggio, in seguito riceverai degli errori relativi all'impossibilità dello strumento AWS CLI o AWS SAM build di pubblicare la tua funzione Lambda.

Creazione del progetto Maven

Esegui il comando seguente per creare il progetto Maven. Sostituisci *`GroupiD`* con l'ID univoco della tua organizzazione e *`my-athena-udf`*sostituiscilo con il nome della tua applicazione Per ulteriori informazioni, vedi Come faccio a creare il mio primo progetto Maven? nella documentazione di Apache Maven.

```
mvn -B archetype:generate \
-DarchetypeGroupId=org.apache.maven.archetypes \
-DgroupId=groupId \
-DartifactId=my-athena-udfs
```

Aggiunta di dipendenze e plugin al progetto Maven

Aggiungi le seguenti configurazioni al file `pom.xml` del progetto Maven. Per un esempio, consultate il file pom.xml in. GitHub

```
<properties>
    <aws-athena-federation-sdk.version>2022.47.1</aws-athena-federation-sdk.version>
</properties>

<dependencies>
    <dependency>
        <groupId>com.amazonaws</groupId>
        <artifactId>aws-athena-federation-sdk</artifactId>
        <version>${aws-athena-federation-sdk.version}</version>
    </dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-shade-plugin</artifactId>
            <version>3.2.1</version>
            <configuration>
                <createDependencyReducedPom>false</createDependencyReducedPom>
                <filters>
                    <filter>
                        <artifact>*:*</artifact>
                        <excludes>
                            <exclude>META-INF/*.SF</exclude>
                            <exclude>META-INF/*.DSA</exclude>
                            <exclude>META-INF/*.RSA</exclude>
                        </excludes>
                    </filter>
                </filters>
            </configuration>
            <executions>
                <execution>
```

```
                    <phase>package</phase>
                    <goals>
                        <goal>shade</goal>
                    </goals>
                </execution>
            </executions>
        </plugin>
    </plugins>
</build>
```

Scrittura di codice Java per le funzioni definite dall'utente

Crea una nuova classe estendendo UserDefinedFunctionHandler.java. Scrivi le funzioni definite dall'utente all'interno della classe.

Nell'esempio seguente, due metodi Java per funzioni definite dall'utente, `compress()` e `decompress()`, vengono creati all'interno della classe `MyUserDefinedFunctions`.

```
*package *com.mycompany.athena.udfs;

public class MyUserDefinedFunctions
        extends UserDefinedFunctionHandler
{
    private static final String SOURCE_TYPE = "MyCompany";

    public MyUserDefinedFunctions()
    {
        super(SOURCE_TYPE);
    }

    /**
     * Compresses a valid UTF-8 String using the zlib compression library.
     * Encodes bytes with Base64 encoding scheme.
     *
     * @param input the String to be compressed
     * @return the compressed String
     */
    public String compress(String input)
    {
        byte[] inputBytes = input.getBytes(StandardCharsets.UTF_8);

        // create compressor
        Deflater compressor = new Deflater();
```

```
        compressor.setInput(inputBytes);
        compressor.finish();

        // compress bytes to output stream
        byte[] buffer = new byte[4096];
        ByteArrayOutputStream byteArrayOutputStream = new
ByteArrayOutputStream(inputBytes.length);
        while (!compressor.finished()) {
            int bytes = compressor.deflate(buffer);
            byteArrayOutputStream.write(buffer, 0, bytes);
        }

        try {
            byteArrayOutputStream.close();
        }
        catch (IOException e) {
            throw new RuntimeException("Failed to close ByteArrayOutputStream", e);
        }

        // return encoded string
        byte[] compressedBytes = byteArrayOutputStream.toByteArray();
        return Base64.getEncoder().encodeToString(compressedBytes);
    }

    /**
     * Decompresses a valid String that has been compressed using the zlib compression
library.
     * Decodes bytes with Base64 decoding scheme.
     *
     * @param input the String to be decompressed
     * @return the decompressed String
     */
    public String decompress(String input)
    {
        byte[] inputBytes = Base64.getDecoder().decode((input));

        // create decompressor
        Inflater decompressor = new Inflater();
        decompressor.setInput(inputBytes, 0, inputBytes.length);

        // decompress bytes to output stream
        byte[] buffer = new byte[4096];
        ByteArrayOutputStream byteArrayOutputStream = new
ByteArrayOutputStream(inputBytes.length);
```

```
        try {
            while (!decompressor.finished()) {
                int bytes = decompressor.inflate(buffer);
                if (bytes == 0 && decompressor.needsInput()) {
                    throw new DataFormatException("Input is truncated");
                }
                byteArrayOutputStream.write(buffer, 0, bytes);
            }
        }
        catch (DataFormatException e) {
            throw new RuntimeException("Failed to decompress string", e);
        }

        try {
            byteArrayOutputStream.close();
        }
        catch (IOException e) {
            throw new RuntimeException("Failed to close ByteArrayOutputStream", e);
        }

        // return decoded string
        byte[] decompressedBytes = byteArrayOutputStream.toByteArray();
        return new String(decompressedBytes, StandardCharsets.UTF_8);
    }
}
```

Compilazione del file JAR

Esegui `mvn clean install` per compilare il progetto. Al termine della compilazione, viene creato un file JAR nella cartella `target` del progetto denominato *`artifactId`*`-`*`version`*`.jar`, dove *`artifactId`* è il nome fornito nel progetto Maven, ad esempio `my-athena-udfs`.

Distribuisci il JAR su AWS Lambda

Hai a disposizione due opzioni per distribuire il codice in Lambda:

- Distribuisci utilizzando AWS Serverless Application Repository (consigliato)
- Creazione di una funzione Lambda dal file JAR

Opzione 1: distribuzione su AWS Serverless Application Repository

Quando si distribuisce il file JAR su AWS Serverless Application Repository, si crea un file YAML AWS SAM modello che rappresenta l'architettura dell'applicazione. Quindi devi specificare questo file YAML e un bucket Amazon S3 in cui gli artefatti dell'applicazione vengono caricati e resi disponibili ad AWS Serverless Application Repository. Nella procedura riportata di seguito viene utilizzato lo script publish.sh che si trova nella directory `athena-query-federation/tools` dell'SDK di Athena Query Federation clonato in precedenza.

Per ulteriori informazioni e requisiti, consulta Pubblicazione di applicazioni nella AWS Serverless Application Repository Developer Guide, concetti di AWS SAM template nella AWS Serverless Application Model Developer Guide e Pubblicazione di applicazioni serverless utilizzando la AWS SAM CLI.

Nell'esempio seguente vengono illustrati i parametri in un file YAML. Aggiungi parametri simili al file YAML e salvalo nella directory del progetto. Vedi athena-udf.yaml in per un esempio completo. GitHub

```
Transform: 'AWS::Serverless-2016-10-31'
Metadata:
  'AWS::ServerlessRepo::Application':
    Name: MyApplicationName
    Description: 'The description I write for my application'
    Author: 'Author Name'
    Labels:
      - athena-federation
    SemanticVersion: 1.0.0
Parameters:
  LambdaFunctionName:
    Description: 'The name of the Lambda function that will contain your UDFs.'
    Type: String
  LambdaTimeout:
    Description: 'Maximum Lambda invocation runtime in seconds. (min 1 - 900 max)'
    Default: 900
    Type: Number
  LambdaMemory:
    Description: 'Lambda memory in MB (min 128 - 3008 max).'
    Default: 3008
    Type: Number
Resources:
  ConnectorConfig:
    Type: 'AWS::Serverless::Function'
```

```
    Properties:
      FunctionName: !Ref LambdaFunctionName
      Handler: "full.path.to.your.handler. For example,
 com.amazonaws.athena.connectors.udfs.MyUDFHandler"
      CodeUri: "Relative path to your JAR file. For example, ./target/athena-
udfs-1.0.jar"
      Description: "My description of the UDFs that this Lambda function enables."
      Runtime: java8
      Timeout: !Ref LambdaTimeout
      MemorySize: !Ref LambdaMemory
```

Copia lo script `publish.sh` nella directory del progetto in cui è stato salvato il file YAML ed esegui il comando seguente:

```
./publish.sh MyS3Location MyYamlFile
```

Ad esempio, se la posizione del bucket è `s3://DOC-EXAMPLE-BUCKET/mysarapps/athenaudf` e il file YAML è stato salvato come `my-athena-udfs.yaml`:

```
./publish.sh DOC-EXAMPLE-BUCKET/mysarapps/athenaudf my-athena-udfs
```

Per creare una funzione Lambda

1. Apri la console Lambda all'indirizzo https://console.aws.amazon.com/lambda/, scegli Crea funzione, quindi scegliere Sfoglia il repository delle app serverless
2. Scegliere Private applications (Applicazioni private), trovare l'applicazione nell'elenco o cercarla usando le parole chiave e selezionarla.
3. Rivedere e fornire i dettagli dell'applicazione, quindi scegliere Deploy (Distribuisci).

   È ora possibile utilizzare i nomi dei metodi definiti nel file JAR della funzione Lambda come funzioni definite dall'utente in Athena.

Opzione 2: Creare direttamente una funzione Lambda

Puoi anche creare una funzione Lambda direttamente utilizzando la console o. AWS CLI L'esempio seguente illustra l'utilizzo del comando CLI della `create-function` Lambda.

```
aws lambda create-function \
 --function-name MyLambdaFunctionName \
```

```
--runtime java8 \
--role arn:aws:iam::1234567890123:role/my_lambda_role \
--handler com.mycompany.athena.udfs.MyUserDefinedFunctions \
--timeout 900 \
--zip-file fileb://./target/my-athena-udfs-1.0-SNAPSHOT.jar
```

Esecuzione di query tra regioni

Athena supporta la possibilità di interrogare i dati di Amazon S3 in Regione AWS una regione diversa dalla regione in cui utilizzi Athena. L'esecuzione di query tra regioni è utile quando lo spostamento dei dati non è pratico o consentito oppure quando si desidera eseguire query sui dati in più aree geografiche. Anche se Athena non è disponibile in una determinata regione, i dati di quest'ultima possono essere richiamati da un'altra regione in cui è disponibile Athena.

Per eseguire query sui dati in una regione, il tuo account deve essere abilitato nella regione in questione anche se i dati Amazon S3 non appartengono al tuo account. Per alcune regioni come Stati Uniti orientali (Ohio), l'accesso alla regione viene automaticamente abilitato al momento della creazione dell'account. Altre regioni richiedono che il tuo account sia «attivato» nella regione prima di poterlo utilizzare. Per un elenco delle regioni che richiedono l'attivazione, consulta la sezione Regioni disponibili nella Amazon EC2 User Guide. Per istruzioni specifiche sull'adesione a una regione, consulta Gestire AWS le regioni in. Riferimenti generali di Amazon Web Services

Considerazioni e limitazioni

- Autorizzazioni di accesso ai dati: per eseguire correttamente query sui dati di Amazon S3 da Athena tra regioni, il tuo account deve disporre delle autorizzazioni per leggere i dati. Se i dati su cui desidere esguire query appartengono a un altro account, questo deve concederti l'accesso alla posizione Amazon S3 che contiene i dati.
- Costi per il trasferimento dati: per le query tra regioni sono previsti costi di trasferimento dati Amazon S3. L'esecuzione di una query può comportare il trasferimento di più dati rispetto alla dimensione del set di dati. È consigliabile iniziare a testare le query su un sottoinsieme di dati e valutare i costi in AWS Cost Explorer.
- AWS Glue— È possibile utilizzare AWS Glue in più regioni. Potrebbero essere applicati costi aggiuntivi per il AWS Glue traffico interregionale. Per ulteriori informazioni, consulta Creare AWS Glue connessioni tra account e regioni nel AWS Big Data Blog.
- Opzioni di crittografia Amazon S3: le opzioni di crittografia SSE-S3 e SSE-KMS sono supportate per le query tra regioni; mentre CSE-KMS non lo è. Per ulteriori informazioni, consulta Opzioni di crittografia supportate da Amazon S3.

- Interrogazioni federate: l'utilizzo di query federate non è supportato. Regioni AWS
- Regioni della Cina: le query interregionali non sono supportate nelle regioni cinesi.

A condizione che siano soddisfatte le condizioni di cui sopra, puoi creare una tabella Athena che punti al valore `LOCATION` specificato e interrogare i dati in modo trasparente. Non è necessaria alcuna sintassi speciale. Per ulteriori informazioni sulla creazione delle tabelle Athena, consulta Creazione di tabelle in Athena.

Esecuzione di query AWS Glue Data Catalog

Poiché AWS Glue Data Catalog viene utilizzato da molti Servizi AWS come repository centrale di metadati, è possibile eseguire query sui metadati del catalogo dati. A tale scopo, è possibile utilizzare le query SQL in Athena. È possibile utilizzare Athena per eseguire query sui metadati del catalogo AWS Glue come database, tabelle, partizioni e colonne.

Per ottenere i metadati del catalogo AWS Glue, è possibile eseguire una query sul database `information_schema` sul back end di Athena. Le query di esempio in questo argomento mostrano come utilizzare Athena per eseguire query sui metadati del catalogo AWS Glue per i casi di utilizzo comuni.

Argomenti

Considerazioni e limitazioni

- Invece di interrogare il database `information_schema`, puoi utilizzare i singoli comandi DDLdi Apache Hive per estrarre informazioni sui metadati per database, tabelle, viste, partizioni e colonne specifici da Athena. Tuttavia, l'output è in formato non tabulare.

- L'interrogazione di `information_schema` è più performante se disponi di una quantità di metadati AWS Glue da piccola a moderata. Se disponi di una quantità di metadati elevata, possono verificarsi degli errori.
- Non è possibile utilizzare `CREATE VIEW` per creare una visualizzazione nel database `information_schema`.

Elencare database e ricercare un database specificato

Negli esempi in questa sezione viene illustrato come elencare i database nei metadati in base al nome dello schema.

Example Elencare i database

La query di esempio seguente elenca i database della tabella `information_schema.schemata`.

```
SELECT schema_name
FROM   information_schema.schemata
LIMIT  10;
```

Nella tabella seguente sono riportati i risultati di esempio.

6	alb-databas1
7	alb_original_cust
8	alblogsdatabase
9	athena_db_test
10	athena_ddl_db

Example Ricerca di un database specificato

Nella query di esempio seguente, `rdspostgresql` è un database di esempio.

```
SELECT schema_name
FROM   information_schema.schemata
WHERE  schema_name = 'rdspostgresql'
```

Nella tabella seguente sono riportati i risultati di esempio.

	schema_name
1	rdspostgresql

Elencare le tabelle in un database specificato e ricercare una tabella per nome

Per elencare i metadati per le tabelle, è possibile eseguire query in base allo schema di tabella o al nome della tabella.

Example Elencare le tabelle per schema

Nella query seguente sono elencate le tabelle che utilizzano lo schema di tabella `rdspostgresql`.

```
SELECT table_schema,
       table_name,
       table_type
FROM   information_schema.tables
WHERE  table_schema = 'rdspostgresql'
```

La tabella seguente mostra un risultato di esempio.

	table_schema	table_name	table_type
1	rdspostgresql	rdspostgresqldb1_public_account	BASE TABLE

Example Ricercare una tabella per nome

La query seguente ottiene informazioni sui metadati per la tabella `athena1`.

```
SELECT table_schema,
       table_name,
       table_type
FROM   information_schema.tables
WHERE  table_name = 'athena1'
```

La tabella seguente mostra un risultato di esempio.

	table_schema	table_name	table_type
1	default	athena1	BASE TABLE

Elencare le partizioni per una tabella specifica

Puoi utilizzare`SHOW PARTITIONS` *`table_name`* per elencare le partizioni di una tabella specificata, come nell'esempio seguente.

```
SHOW PARTITIONS cloudtrail_logs_test2
```

È possibile utilizzare una query di metadati `$partitions` per elencare i numeri e i valori di partizione per una tabella specifica.

Example - Esecuzione di query sulle partizioni di una tabella utilizzando la sintassi $partitions

La query di esempio seguente elenca le partizioni della tabella `cloudtrail_logs_test2` utilizzando la sintassi `$partitions`.

```
SELECT * FROM default."cloudtrail_logs_test2$partitions" ORDER BY partition_number
```

Nella tabella seguente sono riportati i risultati di esempio.

	table_catalog	table_schema	table_name	Anno	Mese	Day (Giorno)
1	awsdatacatalog	predefinito	cloudtrail_logs_test2	2020	08	10
2	awsdatacatalog	predefinito	cloudtrail_logs_test2	2020	08	11
3	awsdatacatalog	predefinito	cloudtrail_logs_test2	2020	08	12

Elencare tutte le colonne di ogni tabella

Puoi elencare tutte le colonne di ogni tabella in `AwsDataCatalog` o di tutte le tabelle in un database specifico in `AwsDataCatalog`.

- Per elencare tutte le colonne di ogni database in `AwsDataCatalog`, utilizza la query `SELECT * FROM information_schema.columns`.
- Per limitare i risultati a un database specifico, utilizza `table_schema='database_name'` nella clausola `WHERE`.

Example - Elencare tutte le colonne di ogni tabella di un database specifico

La query di esempio seguente elenca tutte le colonne di ogni tabella nel database `webdata`.

```
SELECT * FROM information_schema.columns WHERE table_schema = 'webdata'
```

Elencare le colonne che hanno in comune tabelle specifiche

Puoi elencare le colonne che hanno in comune tabelle specifiche di un database.

- Utilizzo della sintassi `SELECT column_name FROM information_schema.columns`.
- Per la clausola `WHERE`, utilizza la sintassi `WHERE table_name IN ('table1', 'table2')`.

Example - Elencare le colonne comuni per due tabelle nello stesso database

La seguente query di esempio elenca le colonne che le tabelle `table1` e `table2` hanno in comune.

```
SELECT column_name
FROM information_schema.columns
WHERE table_name IN ('table1', 'table2')
GROUP BY column_name
HAVING COUNT(*) > 1;
```

Elencare o ricercare colonne per una tabella o una vista specificata

È possibile elencare tutte le colonne per una tabella, tutte le colonne per una vista o ricercare una colonna per nome in un database e una tabella specificati.

Per elencare le colonne, utilizzare una query `SELECT *`. Nella clausola `FROM`, specificare `information_schema.columns`. Nella clausola `WHERE`, utilizzare `table_schema='database_name'` per specificare il database e `table_name = 'table_name'` per specificare la tabella o la visualizzazione con le colonne che si desidera elencare.

Example Elencare tutte le colonne di una tabella specificata

La query di esempio seguente elenca tutte le colonne della tabella `rdspostgresqldb1_public_account`.

```
SELECT *
FROM   information_schema.columns
WHERE  table_schema = 'rdspostgresql'
       AND table_name = 'rdspostgresqldb1_public_account'
```

Nella tabella seguente sono riportati i risultati di esempio.

	table_catalog	table_schema	table_name	column_name	ordinal_position	column_default	is_nullable	data_t	comm	extra_info
1	awsdatacatalog	rdspostgresql	rdspostgresqldb1_public_account	passwor	1		YES	varcha		
2	awsdatacatalog	rdspostgresql	rdspostgresqldb1_public_account	user_id	2		YES	intege		
3	awsdatacatalog	rdspostgresql	rdspostgresqldb1_public_account	created_n	3		YES	timest		
4	awsdatacatalog	rdspostgresql	rdspostgresqldb1_public_account	last_login	4		YES	timest		

	table_catalog	table_schema	table_name	column_name	ordinal_position	column_default	is_nullable	data_t	comm	extra_info
5	awsdatacatalog	rdspostgresql	rdspostgresqldb1_public_account	e-mail	5		YES	varcha		
6	awsdatacatalog	rdspostgresql	rdspostgresqldb1_public_account	usernam	6		YES	varcha		

Example Elencare le colonne di una visualizzazione specificata

La query di esempio seguente elenca tutte le colonne nel database `default` per la vista `arrayview`.

```
SELECT *
FROM   information_schema.columns
WHERE  table_schema = 'default'
       AND table_name = 'arrayview'
```

Nella tabella seguente sono riportati i risultati di esempio.

	table_catalog	table_schema	table_name	column_name	ordinal_position	column_default	is_nullable	data_typ	comm	extra_info
1	awsdatacatalog	predefinito	arrayvie	searchdate	1		YES	varchar		
2	awsdatacatalog	predefinito	arrayvie	sid	2		YES	varchar		
3	awsdatacatalog	predefinito	arrayvie	btid	3		YES	varchar		

	table_catalog	table_schema	table_name	column_name	ordinal_position	column_default	is_nullable	data_type	comm	extra_info
4	awsdatacatalog	predefinito	arrayvie	p	4		YES	varchar		
5	awsdatacatalog	predefinito	arrayvie	infantprice	5		YES	varchar		
6	awsdatacatalog	predefinito	arrayvie	sump	6		YES	varchar		
7	awsdatacatalog	predefinito	arrayvie	journeymaparray	7		YES	array(varchar)		

Example Ricercare una colonna per nome in un database e una tabella specificati

La query di esempio seguente cerca i metadati per la colonna `sid` nella vista `arrayview` del database `default`.

```
SELECT *
FROM   information_schema.columns
WHERE  table_schema = 'default'
       AND table_name = 'arrayview'
       AND column_name='sid'
```

La tabella seguente mostra un risultato di esempio.

	table_catalog	table_schema	table_name	column_name	ordinal_position	column_default	is_nullable	data_type	comm	extra_info
1	awsdatacatalog	predefinito	arrayview	sid	2		YES	varcha		

Interrogazione dei log Servizio AWS

Questa sezione include diverse procedure per utilizzare Amazon Athena per interrogare i set di dati più diffusi, come AWS CloudTrail log, log CloudFront Amazon, log Classic Load Balancer, log Application Load Balancer, log di flusso Amazon VPC e log Network Load Balancer.

Per le attività riportate in questa sezione viene impiegata la console Athena, tuttavia puoi usare anche altri strumenti come il driver JDBC Athena, la AWS CLI o la documentazione di riferimento all'API Amazon Athena.

Per informazioni sull'utilizzo per AWS CloudFormation creare automaticamente tabelle di Servizio AWS log, partizioni e query di esempio in Athena, consulta Automatizzare la creazione di tabelle di Servizio AWS log e interrogarle con Amazon Athena nel Big Data Blog. AWS Per informazioni sull'utilizzo di una libreria Python per creare un framework comune AWS Glue per l'elaborazione dei Servizio AWS log e la loro interrogazione in Athena, consulta Easly query logs using Amazon Athena. Servizio AWS

Gli argomenti di questa sezione presuppongono che tu abbia configurato le autorizzazioni adeguate per accedere ad Athena e al bucket Amazon S3 in cui devono risiedere i dati su cui eseguire la query. Per ulteriori informazioni, consulta Configurazione e Nozioni di base.

Argomenti

Esecuzione di query nei log di Application Load Balancer

Un Application Load Balancer è un'opzione di bilanciamento del carico in Elastic Load Balancing che consente di smistare il traffico in una distribuzione a microservizi con container. Eseguire query sui log dell'Application Load Balancer consente di individuare l'origine del traffico, la latenza e i byte trasferiti a e da istanze Elastic Load Balancing e applicazioni di back-end. Per ulteriori informazioni, consulta i log di accesso per l'Application Load Balancer e i log di connessione per l'Application Load Balancer nella User Guide for Application Load Balancer.

Argomenti

Prerequisiti

- Abilita la registrazione degli accessi o la registrazione delle connessioni in modo che i log di Application Load Balancer possano essere salvati nel tuo bucket Amazon S3.
- Un database per mantenere la tabella da creare per Athena. Per creare un database, puoi usare Athena o AWS Glue la console. Per ulteriori informazioni, consulta Creazione di database in Athena in questa guida o Lavorare con i database nella console AWS Glue nella Guida per gli sviluppatori di AWS Glue .

Creazione della tabella per i log di accesso ALB

1. Copia e incolla la seguente `CREATE TABLE` istruzione nell'editor di query nella console Athena, quindi modificala se necessario per i tuoi requisiti di immissione dei log. Per informazioni sull'utilizzo della console Athena, consulta Nozioni di base. Sostituisci il percorso nella `LOCATION` clausola con la posizione della cartella del log di accesso di Amazon S3. Per ulteriori informazioni

sulla posizione dei file di log di accesso, consulta Access log files nella User Guide for Application Load Balancers.

Per informazioni su ogni campo del file di registro, vedere Access log entry nella User Guide for Application Load Balancers.

> **Note**
>
> L'`CREATE TABLE`istruzione di esempio seguente include le colonne `classificationclassification_reason`, e `conn_trace_id` ('traceability ID' o TID) aggiunte di recente. Per creare una tabella per i log di accesso di Application Load Balancer che non contengono queste voci, rimuovete le colonne corrispondenti dall'`CREATE TABLE`istruzione e modificate l'espressione regolare di conseguenza.

```
CREATE EXTERNAL TABLE IF NOT EXISTS alb_access_logs (
             type string,
             time string,
             elb string,
             client_ip string,
             client_port int,
             target_ip string,
             target_port int,
             request_processing_time double,
             target_processing_time double,
             response_processing_time double,
             elb_status_code int,
             target_status_code string,
             received_bytes bigint,
             sent_bytes bigint,
             request_verb string,
             request_url string,
             request_proto string,
             user_agent string,
             ssl_cipher string,
             ssl_protocol string,
             target_group_arn string,
             trace_id string,
             domain_name string,
             chosen_cert_arn string,
             matched_rule_priority string,
```

```
            request_creation_time string,
            actions_executed string,
            redirect_url string,
            lambda_error_reason string,
            target_port_list string,
            target_status_code_list string,
            classification string,
            classification_reason string,
            conn_trace_id string
            )
            ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
            WITH SERDEPROPERTIES (
            'serialization.format' = '1',
            'input.regex' =
        '([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*):([0-9]*) ([^ ]*)[:-]([0-9]*) ([-.0-9]*)
 ([-.0-9]*) ([-.0-9]*) (|[-0-9]*) (-|[-0-9]*) ([-0-9]*) ([-0-9]*) \"([^ ]*) (.*) (-
 |[^ ]*)\" \"([^\"]*)\" ([A-Z0-9-_]+) ([A-Za-z0-9.-]*) ([^ ]*) \"([^\"]*)\" \"([^
\"]*)\" \"([^\"]*)\" ([-.0-9]*) ([^ ]*) \"([^\"]*)\" \"([^\"]*)\" \"([^ ]*)\" \"([^
\s]+?)\" \"([^\s]+)\" \"([^ ]*)\" \"([^ ]*)\" ?([^ ]*)?( .*)?')
            LOCATION 's3://DOC-EXAMPLE-BUCKET/access-log-folder-path/'
```

2. Eseguire la query nella console Athena. Una volta completata la query, Athena registra la tabella `alb_access_logs`, rendendo i dati in essa contenuti pronti per l'esecuzione di query.

Creazione della tabella per i log di accesso ALB in Athena utilizzando la proiezione delle partizioni

Poiché i log di accesso ALB hanno una struttura nota il cui schema di partizione è possibile specificare in anticipo, è possibile ridurre il tempo di esecuzione delle query e automatizzare la gestione delle partizioni utilizzando la funzionalità di proiezione delle partizioni Athena. La proiezione delle partizioni aggiunge automaticamente nuove partizioni man mano che vengono aggiunti nuovi dati. Ciò elimina la necessità di aggiungere manualmente le partizioni utilizzando `ALTER TABLE ADD PARTITION`.

L'`CREATE TABLE`istruzione di esempio seguente utilizza automaticamente la proiezione delle partizioni nei log di accesso ALB da una data specificata fino ad oggi per una singola regione. AWS L'istruzione si basa sull'esempio della sezione precedente, ma aggiunge le clausole `PARTITIONED BY` e `TBLPROPERTIES` per abilitare la proiezione delle partizioni. Nelle `storage.location.template` clausole `LOCATION` and, sostituisci i segnaposto con valori che identificano la posizione del bucket Amazon S3 dei log di accesso ALB. Per ulteriori informazioni sulla posizione dei file di log di accesso, consulta Access log files nella User Guide for Application Load Balancers. Su `projection.day.range`, sostituire *2022*/*01*/*01* con la data di inizio che si desidera

usare. Dopo aver eseguito la query con esito positivo, è possibile eseguire query sulla tabella. Non è necessario eseguire `ALTER TABLE ADD PARTITION` per caricare le partizioni. Per informazioni su ogni campo del file di registro, vedere Access log entry.

```
CREATE EXTERNAL TABLE IF NOT EXISTS alb_access_logs (
             type string,
             time string,
             elb string,
             client_ip string,
             client_port int,
             target_ip string,
             target_port int,
             request_processing_time double,
             target_processing_time double,
             response_processing_time double,
             elb_status_code int,
             target_status_code string,
             received_bytes bigint,
             sent_bytes bigint,
             request_verb string,
             request_url string,
             request_proto string,
             user_agent string,
             ssl_cipher string,
             ssl_protocol string,
             target_group_arn string,
             trace_id string,
             domain_name string,
             chosen_cert_arn string,
             matched_rule_priority string,
             request_creation_time string,
             actions_executed string,
             redirect_url string,
             lambda_error_reason string,
             target_port_list string,
             target_status_code_list string,
             classification string,
             classification_reason string,
             conn_trace_id string
             )
             PARTITIONED BY
             (
              day STRING
```

```
            )
            ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
            WITH SERDEPROPERTIES (
            'serialization.format' = '1',
            'input.regex' =
        '([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*):([0-9]*) ([^ ]*)[:-]([0-9]*) ([-.0-9]*)
 ([-.0-9]*) ([-.0-9]*) (|[-0-9]*) (-|[-0-9]*) ([-0-9]*) ([-0-9]*) \"([^ ]*) (.*) (- |
[^ ]*)\" \"([^\"]*)\" ([A-Z0-9-_]+) ([A-Za-z0-9.-]*) ([^ ]*) \"([^\"]*)\" \"([^\"]*)\"
 \"([^\"]*)\" ([-.0-9]*) ([^ ]*) \"([^\"]*)\" \"([^\"]*)\" \"([^ ]*)\" \"([^\s]+?)\"
 \"([^\s]+)\" \"([^ ]*)\" \"([^ ]*)\" ?([^ ]*)?( .*)?')
            LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/<ACCOUNT-NUMBER>/
elasticloadbalancing/<REGION>/'
            TBLPROPERTIES
            (
             "projection.enabled" = "true",
             "projection.day.type" = "date",
             "projection.day.range" = "2022/01/01,NOW",
             "projection.day.format" = "yyyy/MM/dd",
             "projection.day.interval" = "1",
             "projection.day.interval.unit" = "DAYS",
             "storage.location.template" = "s3://DOC-EXAMPLE-BUCKET/AWSLogs/<ACCOUNT-
NUMBER>/elasticloadbalancing/<REGION>/${day}"
            )
```

Per maggiori informazioni sulla proiezione delle partizioni, consulta Proiezione delle partizioni con Amazon Athena.

Query di esempio per i log di accesso ALB

La seguente query conteggia il numero di richieste HTTP GET ricevute dal load balancer, raggruppate in base all'indirizzo IP del client:

```
SELECT COUNT(request_verb) AS
 count,
 request_verb,
 client_ip
FROM alb_access_logs
GROUP BY request_verb, client_ip
LIMIT 100;
```

Un'altra query mostra gli URL visitati dagli utenti del browser Safari:

```
SELECT request_url
```

```
FROM alb_access_logs
WHERE user_agent LIKE '%Safari%'
LIMIT 10;
```

La seguente query mostra i record con valori del codice di stato ELB maggiori o uguali a 500.

```
SELECT * FROM alb_access_logs
WHERE elb_status_code >= 500
```

Il seguente esempio illustra come analizzare i log per `datetime`:

```
SELECT client_ip, sum(received_bytes)
FROM alb_access_logs
WHERE parse_datetime(time,'yyyy-MM-dd''T''HH:mm:ss.SSSSSS''Z')
     BETWEEN parse_datetime('2018-05-30-12:00:00','yyyy-MM-dd-HH:mm:ss')
     AND parse_datetime('2018-05-31-00:00:00','yyyy-MM-dd-HH:mm:ss')
GROUP BY client_ip;
```

La seguente query interroga la tabella che utilizza la proiezione delle partizioni per tutti i log di accesso ALB del giorno specificato.

```
SELECT *
FROM alb_access_logs
WHERE day = '2022/02/12'
```

Creazione della tabella per i log di connessione ALB

1. Copia e incolla la seguente `CREATE TABLE` istruzione di esempio nell'editor di query nella console Athena, quindi modificala se necessario per i tuoi requisiti di immissione del registro. Per informazioni sull'utilizzo della console Athena, consulta Nozioni di base. Sostituisci il percorso nella `LOCATION` clausola con la posizione della cartella del log di connessione Amazon S3. Per ulteriori informazioni sulla posizione dei file di registro delle connessioni, consulta File di registro delle connessioni nella Guida utente per Application Load Balancers. Per informazioni su ogni campo del file di registro, vedere Connection log entry.

   ```
   CREATE EXTERNAL TABLE IF NOT EXISTS alb_connection_logs (
            time string,
            client_ip string,
            client_port int,
            listener_port int,
   ```

```
        tls_protocol string,
        tls_cipher string,
        tls_handshake_latency double,
        leaf_client_cert_subject string,
        leaf_client_cert_validity string,
        leaf_client_cert_serial_number string,
        tls_verify_status string,
        conn_trace_id string
        )
        ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
        WITH SERDEPROPERTIES (
        'serialization.format' = '1',
        'input.regex' =
         '([^ ]*) ([^ ]*) ([0-9]*) ([0-9]*) ([A-Za-z0-9.-]*) ([^ ]*) ([-.0-9]*)
\"([^\"]*)\" ([^ ]*) ([^ ]*) ([^ ]*) ?([^ ]*)?( .*)?')
        LOCATION 's3://DOC-EXAMPLE-BUCKET/connection-log-folder-path/'
```

2. Eseguire la query nella console Athena. Una volta completata la query, Athena registra la tabella `alb_connection_logs`, rendendo i dati in essa contenuti pronti per l'esecuzione di query.

Creazione della tabella per i log di connessione ALB in Athena utilizzando la proiezione delle partizioni

Poiché i log di connessione ALB hanno una struttura nota il cui schema di partizione è possibile specificare in anticipo, è possibile ridurre il tempo di esecuzione delle query e automatizzare la gestione delle partizioni utilizzando la funzionalità di proiezione delle partizioni Athena. La proiezione delle partizioni aggiunge automaticamente nuove partizioni man mano che vengono aggiunti nuovi dati. Ciò elimina la necessità di aggiungere manualmente le partizioni utilizzando `ALTER TABLE ADD PARTITION`.

L'`CREATE TABLE`istruzione di esempio seguente utilizza automaticamente la proiezione delle partizioni nei log di connessione ALB da una data specificata fino ad oggi per una singola regione. AWS L'istruzione si basa sull'esempio della sezione precedente, ma aggiunge le clausole `PARTITIONED BY` e `TBLPROPERTIES` per abilitare la proiezione delle partizioni. Nelle `storage.location.template` clausole `LOCATION` and, sostituisci i segnaposto con valori che identificano la posizione del bucket Amazon S3 dei log di connessione ALB. Per ulteriori informazioni sulla posizione dei file di registro delle connessioni, consulta File di registro delle connessioni nella Guida utente per Application Load Balancers. Per`projection.day.range`, sostituisci *2023*/*01*/*01* con la data di inizio che desideri utilizzare. Dopo aver eseguito la query con esito positivo, è possibile eseguire query sulla tabella. Non è necessario eseguire `ALTER TABLE ADD PARTITION` per

caricare le partizioni. Per informazioni su ogni campo del file di registro, consulta Voci del registro di connessione.

```
CREATE EXTERNAL TABLE IF NOT EXISTS alb_connection_logs (
         time string,
         client_ip string,
         client_port int,
         listener_port int,
         tls_protocol string,
         tls_cipher string,
         tls_handshake_latency double,
         leaf_client_cert_subject string,
         leaf_client_cert_validity string,
         leaf_client_cert_serial_number string,
         tls_verify_status string,
         conn_trace_id string
         )
            PARTITIONED BY
            (
             day STRING
            )
            ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
            WITH SERDEPROPERTIES (
            'serialization.format' = '1',
            'input.regex' =
             '([^ ]*) ([^ ]*) ([0-9]*) ([0-9]*) ([A-Za-z0-9.-]*) ([^ ]*) ([-.0-9]*)
 \"([^\"]*)\" ([^ ]*) ([^ ]*) ([^ ]*) ?([^ ]*)?( .*)?')
            LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/<ACCOUNT-NUMBER>/
elasticloadbalancing/<REGION>/'
            TBLPROPERTIES
            (
             "projection.enabled" = "true",
             "projection.day.type" = "date",
             "projection.day.range" = "2023/01/01,NOW",
             "projection.day.format" = "yyyy/MM/dd",
             "projection.day.interval" = "1",
             "projection.day.interval.unit" = "DAYS",
             "storage.location.template" = "s3://DOC-EXAMPLE-BUCKET/AWSLogs/<ACCOUNT-
NUMBER>/elasticloadbalancing/<REGION>/${day}"
            )
```

Per maggiori informazioni sulla proiezione delle partizioni, consulta Proiezione delle partizioni con Amazon Athena.

Query di esempio per i log di connessione ALB

Le seguenti query contano le occorrenze in cui il valore per non `tls_verify_status` era `'Success'`, raggruppate per indirizzo IP del client:

```
SELECT DISTINCT client_ip, count() AS count FROM alb_connection_logs
WHERE tls_verify_status != 'Success'
GROUP BY client_ip
ORDER BY count() DESC;
```

La seguente query cerca le occorrenze in cui il valore di `tls_handshake_latency` è superiore a 2 secondi nell'intervallo di tempo specificato:

```
SELECT * FROM alb_connection_logs
WHERE
  (
    parse_datetime(time, 'yyyy-MM-dd''T''HH:mm:ss.SSSSSS''Z')
    BETWEEN
    parse_datetime('2024-01-01-00:00:00', 'yyyy-MM-dd-HH:mm:ss')
    AND
    parse_datetime('2024-03-20-00:00:00', 'yyyy-MM-dd-HH:mm:ss')
  )
  AND
    (tls_handshake_latency >= 2.0);
```

Risorse aggiuntive

- Come posso analizzare i log di accesso di Application Load Balancer utilizzando Amazon Athena? nel Centro conoscenze di AWS .
- Per informazioni sui codici di stato HTTP in Elastic Load Balancing, consulta Risoluzione dei problemi con Application Load Balancer nella Guida per l'utente di Application Load Balancer.
- Cataloga e analizza i log di Application Load Balancer in modo più efficiente con classificatori AWS Glue personalizzati e Amazon Athena nel Big Data Blog.AWS

Richiesta di log Classic Load Balancer

Usa i log di Classic Load Balancer per analizzare e comprendere i modelli di traffico da e verso le istanze di Elastic Load Balancing e le applicazioni di back-end. È possibile visualizzare sorgente di traffico, latenza e byte trasferiti.

Prima di analizzare i log di Elastic Load Balancing, configurali affinché siano salvati nel bucket Amazon S3 di destinazione. Per ulteriori informazioni, consulta la sezione relativa all'accesso ai log di Classic Load Balancer.

- Creazione della tabella per i log di Elastic Load Balancing
- Query di esempio di Elastic Load Balancing

Per creare la tabella per i log di Elastic Load Balancing

1. Copiare e incollare la seguente istruzione DDL nella console Athena. Controlla la sintassi dei record di log di Elastic Load Balancing. Potrebbe essere necessario aggiornare la seguente query per includere le colonne e la sintassi Regex per la versione più recente del record.

```
CREATE EXTERNAL TABLE IF NOT EXISTS elb_logs (

 timestamp string,
 elb_name string,
 request_ip string,
 request_port int,
 backend_ip string,
 backend_port int,
 request_processing_time double,
 backend_processing_time double,
 client_response_time double,
 elb_response_code string,
 backend_response_code string,
 received_bytes bigint,
 sent_bytes bigint,
 request_verb string,
 url string,
 protocol string,
 user_agent string,
 ssl_cipher string,
 ssl_protocol string
)
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
WITH SERDEPROPERTIES (
 'serialization.format' = '1',
 'input.regex' = '([^ ]*) ([^ ]*) ([^ ]*):([0-9]*) ([^ ]*)[:-]([0-9]*) ([-.0-9]*)
 ([-.0-9]*) ([-.0-9]*) (|[-0-9]*) (-|[-0-9]*) ([-0-9]*) ([-0-9]*) \\\"([^ ]*)
 ([^ ]*) (- |[^ ]*)\\\" (\"[^\"]*\") ([A-Z0-9-]+) ([A-Za-z0-9.-]*)$'
```

```
)
LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/AWS_account_ID/elasticloadbalancing/';
```

2. Modifica il bucket Amazon S3 `LOCATION` per specificare la destinazione dei tuoi log di Elastic Load Balancing.
3. Eseguire la query nella console Athena. Una volta completata la query, Athena registra la tabella `elb_logs`, rendendo i dati in essa contenuti pronti per le query. Per ulteriori informazioni, consulta Query di esempio di Elastic Load Balancing.

Query di esempio di Elastic Load Balancing

Eseguire una query simile all'esempio seguente. In tale query sono elencati i server di applicazioni di back-end che hanno restituito un codice di risposta di errore `4XX` o `5XX`. Utilizza l'operatore `LIMIT` per limitare il numero di log di cui eseguire la query alla volta.

```
SELECT
 timestamp,
 elb_name,
 backend_ip,
 backend_response_code
FROM elb_logs
WHERE backend_response_code LIKE '4%' OR
      backend_response_code LIKE '5%'
LIMIT 100;
```

Utilizza una successiva query per sommare il tempo di risposta di tutte le transazioni raggruppate in base all'indirizzo IP del back-end e al nome dell'istanza Elastic Load Balancing.

```
SELECT sum(backend_processing_time) AS
 total_ms,
 elb_name,
 backend_ip
FROM elb_logs WHERE backend_ip <> ''
GROUP BY backend_ip, elb_name
LIMIT 100;
```

Per ulteriori informazioni, consulta la sezione relativa all'analisi dei dati su S3 utilizzando Athena.

Interrogazione dei log di Amazon CloudFront

Puoi configurare Amazon CloudFront CDN per esportare i log di accesso alla distribuzione Web su Amazon Simple Storage Service. Usa questi log per esplorare i modelli di navigazione degli utenti tra le proprietà web servite da. CloudFront

Prima di iniziare a interrogare i log, abilita il registro di accesso alle distribuzioni Web sulla tua distribuzione preferita. CloudFront Per informazioni, consulta i log di accesso nella Amazon CloudFront Developer Guide. Prendi nota del bucket Amazon S3 in cui salvi questi log.

- Creazione di una tabella per i log standard CloudFront
- Creazione di una tabella per i log CloudFront in tempo reale
- Interrogazioni di esempio per log standard CloudFront

Creazione di una tabella per i log standard CloudFront

> **Note**
>
> Questa procedura funziona per i log di accesso alla distribuzione Web. CloudFront Non si applica ai log di streaming delle distribuzioni RTMP.

Per creare una tabella per i campi CloudFront standard dei file di log

1. Copia e incolla la seguente istruzione DDL di esempio nell'Editor di query della console Athena. L'istruzione di esempio utilizza i campi del file di registro documentati nella sezione Campi dei file di log standard della Amazon CloudFront Developer Guide. Modifica il parametro `LOCATION` per il bucket Amazon S3 in cui sono archiviati i log. Per ulteriori informazioni sull'utilizzo dell'editor di query, consulta Nozioni di base.

   Questa query specifica `ROW FORMAT DELIMITED` e indica che `FIELDS TERMINATED BY '\t'` i campi sono delimitati da caratteri di tabulazione. Per`ROW FORMAT DELIMITED`, Athena utilizza per impostazione LazySimpleSerDepredefinita. Nella colonna `date` viene inserito il carattere escape con l'apice inverso (`) perché è una parola riservata in Athena. Per informazioni, consulta Parole chiave riservate.

```
CREATE EXTERNAL TABLE IF NOT EXISTS cloudfront_standard_logs (
  `date` DATE,
```

```
  time STRING,
  x_edge_location STRING,
  sc_bytes BIGINT,
  c_ip STRING,
  cs_method STRING,
  cs_host STRING,
  cs_uri_stem STRING,
  sc_status INT,
  cs_referrer STRING,
  cs_user_agent STRING,
  cs_uri_query STRING,
  cs_cookie STRING,
  x_edge_result_type STRING,
  x_edge_request_id STRING,
  x_host_header STRING,
  cs_protocol STRING,
  cs_bytes BIGINT,
  time_taken FLOAT,
  x_forwarded_for STRING,
  ssl_protocol STRING,
  ssl_cipher STRING,
  x_edge_response_result_type STRING,
  cs_protocol_version STRING,
  fle_status STRING,
  fle_encrypted_fields INT,
  c_port INT,
  time_to_first_byte FLOAT,
  x_edge_detailed_result_type STRING,
  sc_content_type STRING,
  sc_content_len BIGINT,
  sc_range_start BIGINT,
  sc_range_end BIGINT
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
LOCATION 's3://DOC-EXAMPLE-BUCKET/'
TBLPROPERTIES ( 'skip.header.line.count'='2' )
```

2. Eseguire la query nella console Athena. Una volta completata la query, Athena registra la tabella `cloudfront_standard_logs`, rendendo i dati in essa contenuti pronti per l'esecuzione di query.

Creazione di una tabella per i log CloudFront in tempo reale

Per creare una tabella per i campi dei file di registro CloudFront in tempo reale

1. Copia e incolla la seguente istruzione DDL di esempio nell'Editor di query della console Athena. L'istruzione di esempio utilizza i campi del file di registro documentati nella sezione Real-time logs della Amazon CloudFront Developer Guide. Modifica il parametro `LOCATION` per il bucket Amazon S3 in cui sono archiviati i log. Per ulteriori informazioni sull'utilizzo dell'editor di query, consulta Nozioni di base.

   Questa query specifica `ROW FORMAT DELIMITED` e indica che `FIELDS TERMINATED BY '\t'` i campi sono delimitati da caratteri di tabulazione. Per`ROW FORMAT DELIMITED`, Athena utilizza per impostazione LazySimpleSerDepredefinita. Nella colonna `timestamp` viene inserito il carattere escape con l'apice inverso (`) perché è una parola riservata in Athena. Per informazioni, consulta Parole chiave riservate.

   L'esempio seguente contiene tutti i campi disponibili. Puoi aggiungere commenti o rimuovere campi che non ti servono.

```
CREATE EXTERNAL TABLE IF NOT EXISTS cloudfront_real_time_logs (
`timestamp` STRING,
c_ip STRING,
time_to_first_byte BIGINT,
sc_status BIGINT,
sc_bytes BIGINT,
cs_method STRING,
cs_protocol STRING,
cs_host STRING,
cs_uri_stem STRING,
cs_bytes BIGINT,
x_edge_location STRING,
x_edge_request_id STRING,
x_host_header STRING,
time_taken BIGINT,
cs_protocol_version STRING,
c_ip_version STRING,
cs_user_agent STRING,
cs_referer STRING,
cs_cookie STRING,
cs_uri_query STRING,
x_edge_response_result_type STRING,
x_forwarded_for STRING,
```

```
ssl_protocol STRING,
ssl_cipher STRING,
x_edge_result_type STRING,
fle_encrypted_fields STRING,
fle_status STRING,
sc_content_type STRING,
sc_content_len BIGINT,
sc_range_start STRING,
sc_range_end STRING,
c_port BIGINT,
x_edge_detailed_result_type STRING,
c_country STRING,
cs_accept_encoding STRING,
cs_accept STRING,
cache_behavior_path_pattern STRING,
cs_headers STRING,
cs_header_names STRING,
cs_headers_count BIGINT,
primary_distribution_id STRING,
primary_distribution_dns_name STRING,
origin_fbl STRING,
origin_lbl STRING,
asn STRING
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
LOCATION 's3://DOC-EXAMPLE-BUCKET/'
TBLPROPERTIES ( 'skip.header.line.count'='2' )
```

2. Eseguire la query nella console Athena. Una volta completata la query, Athena registra la tabella `cloudfront_real_time_logs`, rendendo i dati in essa contenuti pronti per l'esecuzione di query.

Interrogazioni di esempio per log standard CloudFront

La seguente query somma il numero di byte serviti CloudFront tra il 9 giugno e l'11 giugno 2018. Racchiudi fra virgolette doppie il nome della colonna della data perché è una parola riservata.

```
SELECT SUM(bytes) AS total_bytes
FROM cloudfront_standard_logs
WHERE "date" BETWEEN DATE '2018-06-09' AND DATE '2018-06-11'
LIMIT 100;
```

Per eliminare righe duplicate (ad esempio, righe vuote duplicate) dai risultati della query, è possibile utilizzare l'istruzione `SELECT DISTINCT`, come nell'esempio seguente.

```
SELECT DISTINCT *
FROM cloudfront_standard_logs
LIMIT 10;
```

Risorse aggiuntive

Per ulteriori informazioni sull'utilizzo di Athena per interrogare CloudFront i log, consulta i seguenti post del blog sui AWS big data.

Interroga facilmente Servizio AWS i log con Amazon Athena (29 maggio 2019).

Analizza i log di CloudFront accesso di Amazon su larga scala (21 dicembre 2018).

Crea un'architettura serverless per analizzare i log di CloudFront accesso di Amazon utilizzando AWS Lambda Amazon Athena e Amazon Managed Service for Apache Flink (26 maggio 2017).

Interrogazione dei log AWS CloudTrail

AWS CloudTrail è un servizio che registra chiamate ed eventi AWS API per gli account Amazon Web Services.

CloudTrail i log includono dettagli su tutte le chiamate API effettuate al tuo account Servizi AWS, inclusa la console. CloudTrail genera file di registro crittografati e li archivia in Amazon S3. Per ulteriori informazioni, consulta la Guida per l'utente AWS CloudTrail.

> **Note**
>
> Se desideri eseguire query SQL sulle informazioni sugli CloudTrail eventi tra account, regioni e date, prendi in considerazione l'utilizzo CloudTrail di Lake. CloudTrail Lake è un' AWS alternativa alla creazione di percorsi che aggregano le informazioni di un'azienda in un unico archivio di dati sugli eventi ricercabile. Invece di utilizzare lo storage bucket Amazon S3, memorizza gli eventi in un data lake, che consente di effettuare query più ricche e più rapide. È possibile utilizzarlo per creare query SQL che cercano eventi tra organizzazioni, regioni e intervalli di tempo personalizzati. Poiché esegui le query su CloudTrail Lake all'interno della CloudTrail console stessa, l'utilizzo di CloudTrail Lake non richiede Athena. Per ulteriori informazioni, consulta la documentazione di CloudTrail Lake.

L'utilizzo di Athena con CloudTrail i log è un modo efficace per migliorare l'analisi delle attività. Servizio AWS Ad esempio, puoi utilizzare le query per identificare le tendenze e isolare con maggiore precisione le attività in base ad attributi specifici, ad esempio l'indirizzo IP di origine o un utente.

Un'applicazione comune consiste nell'utilizzare CloudTrail i log per analizzare l'attività operativa per motivi di sicurezza e conformità. Per informazioni su un esempio dettagliato, consulta il post sul blog AWS Big Data, Analyze security, compliance and operations activity using AWS CloudTrail and Amazon Athena.

È possibile usare Athena per eseguire query su questi file di log direttamente da Amazon S3, specificando la voce `LOCATION` dei file di log. Ci sono due modi per farlo:

- Creando tabelle per i file di CloudTrail log direttamente dalla CloudTrail console.
- Creando manualmente tabelle per i file di CloudTrail registro nella console Athena.

Argomenti

Informazioni CloudTrail sui log e sulle tabelle Athena

Prima di iniziare a creare tabelle, è necessario acquisire maggiori informazioni su CloudTrail e su come vengono archiviati i dati. Questo può aiutarti a creare le tabelle di cui hai bisogno, indipendentemente dal fatto che tu le crei dalla CloudTrail console o da Athena.

CloudTrail salva i log come file di testo JSON in formato gzip compresso (*.json.gzip). La posizione dei file di registro dipende dalla modalità di configurazione dei percorsi, dalle regioni in cui si effettua la registrazione e da Regione AWS altri fattori.

Per ulteriori informazioni su dove vengono archiviati i log, la struttura JSON e i contenuti dei file di record, consulta i seguenti argomenti nella Guida per l'utente di AWS CloudTrail:

- Individuazione dei file di CloudTrail registro
- CloudTrail Esempi di file di registro
- CloudTrail contenuti dei record
- CloudTrail riferimento all'evento

Per raccogliere i log e salvarli su Amazon S3, CloudTrail abilita da. AWS Management Console Per ulteriori informazioni, consulta Creazione di un percorso nella Guida per l'utente di AWS CloudTrail .

Prendi nota del bucket Amazon S3 di destinazione in cui salvi i log. Sostituisci la `LOCATION` clausola con il percorso della posizione del CloudTrail registro e il set di oggetti con cui lavorare. L'esempio utilizza un `LOCATION` valore di log per un determinato account, ma è possibile utilizzare il grado di specificità più adatta alla tua applicazione.

Ad esempio:

- Per analizzare i dati provenienti da più account, puoi eseguire il rollback dell'identificatore `LOCATION` per selezionare tutti gli `AWSLogs` con `LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/'`.
- Per analizzare i dati provenienti da una data, account e regione specifici, utilizza `LOCATION 's3://DOC-EXAMPLE-BUCKET/123456789012/CloudTrail/us-east-1/2016/03/14/'`.

Indicando il livello più elevato nella gerarchia degli oggetti hai la massima flessibilità nelle query con Athena.

Utilizzo della CloudTrail console per creare una tabella Athena per i log CloudTrail

È possibile creare una tabella Athena non partizionata per CloudTrail interrogare i log direttamente dalla console. CloudTrail La creazione di una tabella Athena dalla CloudTrail console richiede l'accesso con un ruolo con autorizzazioni sufficienti per creare tabelle in Athena.

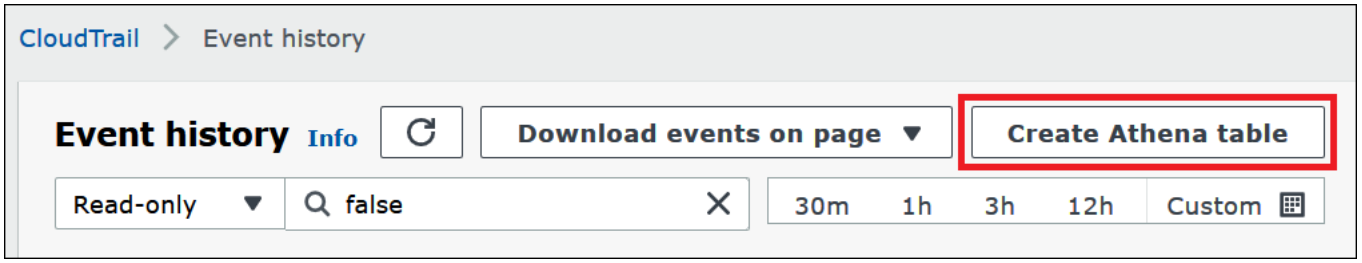
> **Note**
>
> Non è possibile utilizzare la CloudTrail console per creare una tabella Athena per i log degli itinerari organizzativi. Crea invece la tabella manualmente utilizzando la console Athena in

modo da poter specificare la posizione di archiviazione corretta. Per ulteriori informazioni sui percorsi dell'organizzazione, consulta Creazione di un percorso per un'organizzazione nella Guida per l'utente di AWS CloudTrail .

- Per informazioni sull'impostazione delle autorizzazioni di gruppo per Athena, consulta Configurazione.
- Per informazioni sulla creazione di una tabella con partizioni, consulta Creazione di una tabella per CloudTrail i log in Athena utilizzando il partizionamento manuale.

Per creare una tabella Athena per un CloudTrail percorso utilizzando la console CloudTrail

1. Apri la CloudTrail console all'indirizzo https://console.aws.amazon.com/cloudtrail/.
2. Nel riquadro di navigazione scegliere Event history (Cronologia eventi).
3. Scegli Crea la tabella Athena.



4. Per Storage location (Posizione di archiviazione), utilizza la freccia giù per selezionare il bucket Amazon S3 in cui sono archiviati i file di log per il trail sul quale eseguire le query.

> **Note**
>
> Per trovare il nome del bucket associato a un percorso, scegli Percorsi nel riquadro di CloudTrail navigazione e visualizza la colonna del bucket S3 del percorso. Per visualizzare la posizione del bucket in Amazon S3, scegliere il link per il bucket nella colonna Bucket S3. Questo apre la console Amazon S3 nella posizione del CloudTrail bucket.

5. Scegliere Create table (Crea tabella). La tabella viene creata con un nome di default che include il nome del bucket Amazon S3.

Creazione di una tabella per CloudTrail i log in Athena utilizzando il partizionamento manuale

È possibile creare manualmente tabelle per i file di CloudTrail registro nella console Athena e quindi eseguire query in Athena.

Per creare una tabella Athena per un CloudTrail percorso utilizzando la console Athena

1. Copia e incolla la seguente istruzione DDL nell'editor di query Athena.

```
CREATE EXTERNAL TABLE cloudtrail_logs (
eventversion STRING,
useridentity STRUCT<
               type:STRING,
               principalid:STRING,
               arn:STRING,
               accountid:STRING,
               invokedby:STRING,
               accesskeyid:STRING,
               userName:STRING,
  sessioncontext:STRUCT<
    attributes:STRUCT<
               mfaauthenticated:STRING,
               creationdate:STRING>,
    sessionissuer:STRUCT<
               type:STRING,
               principalId:STRING,
               arn:STRING,
               accountId:STRING,
               userName:STRING>,
    ec2RoleDelivery:string,
    webIdFederationData: STRUCT<
               federatedProvider: STRING,
               attributes: map<string,string>
    >
  >
>,
eventtime STRING,
eventsource STRING,
eventname STRING,
awsregion STRING,
sourceipaddress STRING,
useragent STRING,
errorcode STRING,
```

```
errormessage STRING,
requestparameters STRING,
responseelements STRING,
additionaleventdata STRING,
requestid STRING,
eventid STRING,
resources ARRAY<STRUCT<
               arn:STRING,
               accountid:STRING,
               type:STRING>>,
eventtype STRING,
apiversion STRING,
readonly STRING,
recipientaccountid STRING,
serviceeventdetails STRING,
sharedeventid STRING,
vpcendpointid STRING,
eventCategory STRING,
tlsDetails struct<
  tlsVersion:string,
  cipherSuite:string,
  clientProvidedHostHeader:string>
)
PARTITIONED BY (region string, year string, month string, day string)
ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe'
STORED AS INPUTFORMAT 'com.amazon.emr.cloudtrail.CloudTrailInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/Account_ID/CloudTrail/';
```

> **Note**
>
> Suggeriamo di utilizzare quanto `org.apache.hive.hcatalog.data.JsonSerDe` mostrato nell'esempio. Sebbene a `com.amazon.emr.hive.serde.CloudTrailSerde` esista, attualmente non gestisce alcuni dei CloudTrail campi più recenti.

2. (Facoltativo) Rimuovi tutti i campi non obbligatori per la tabella. Se è necessario leggere solo un determinato set di colonne, la definizione della tabella può escludere le altre colonne.
3. Modificare `s3://DOC-EXAMPLE-BUCKET/AWSLogs/Account_ID/CloudTrail/` affinché punti al bucket Amazon S3 che contiene i dati del log.

4. Verificare che i campi siano elencati correttamente. Per ulteriori informazioni sull'elenco completo dei campi in un CloudTrail record, vedere il contenuto del CloudTrail record.

   L'`CREATE TABLE`istruzione di esempio nel passaggio 1 utilizza ilJSON Hive SerDe. Nell'esempio, i campi `requestparametersresponseelements`, e `additionaleventdata` sono elencati come tipo `STRING` nella query, ma sono `STRUCT` i tipi di dati utilizzati in JSON. Pertanto, per estrarre i dati da questi campi, utilizza le funzioni `JSON_EXTRACT`. Per ulteriori informazioni, consulta the section called “Estrazione di dati JSON dalle stringhe”. Per migliorare le prestazioni, l'esempio partiziona i dati per anno Regione AWS, mese e giorno.

5. Esegui l'istruzione `CREATE TABLE` nella console Athena.

6. Utilizzare il comando ALTER TABLE ADD PARTITION per caricare le partizioni in modo da poterle interrogare, come nell'esempio seguente.

   ```
   ALTER TABLE table_name ADD
      PARTITION (region='us-east-1',
                 year='2019',
                 month='02',
                 day='01')
      LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/Account_ID/CloudTrail/us-
   east-1/2019/02/01/'
   ```

Creazione di una tabella per un percorso a livello di organizzazione utilizzando il partizionamento manuale

Per creare una tabella per i file di CloudTrail registro a livello di organizzazione in Athena, segui la procedura riportata di seguitoCreazione di una tabella per CloudTrail i log in Athena utilizzando il partizionamento manuale, ma apporta le modifiche indicate nella procedura seguente.

Per creare una tabella Athena per i registri a livello di organizzazione CloudTrail

1. Nell'istruzione `CREATE TABLE`, modifica la clausola `LOCATION` per includere l'ID dell'organizzazione, come nell'esempio seguente:

   ```
   LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/organization_id/Account_ID/CloudTrail/'
   ```

2. Nella clausola `PARTITIONED BY`, aggiungi una voce per l'ID account sotto forma di stringa, come nell'esempio seguente:

```
PARTITIONED BY (account string, region string, year string, month string, day
 string)
```

L'esempio seguente mostra solo il risultato combinato:

```
...

PARTITIONED BY (account string, region string, year string, month string, day
 string)
ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe'
STORED AS INPUTFORMAT 'com.amazon.emr.cloudtrail.CloudTrailInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/organization_id/Account_ID/CloudTrail/'
```

3. Nella clausola `ADD PARTITION` dell'istruzione `ALTER TABLE`, includi l'ID dell'account, come nell'esempio seguente:

```
ALTER TABLE table_name ADD
PARTITION (account='111122223333',
region='us-east-1',
year='2022',
month='08',
day='08')
```

4. Nella clausola `LOCATION` dell'istruzione `ALTER TABLE`, includi l'ID dell'organizzazione, l'ID dell'account e la partizione che desideri aggiungere, come nell'esempio seguente:

```
LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/organization_id/Account_ID/CloudTrail/us-
east-1/2022/08/08/'
```

L'istruzione di esempio `ALTER TABLE` seguente mostra solo il risultato combinato:

```
ALTER TABLE table_name ADD
PARTITION (account='111122223333',
region='us-east-1',
year='2022',
month='08',
day='08')
LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/organization_id/111122223333/CloudTrail/
us-east-1/2022/08/08/'
```

Creazione della tabella per CloudTrail i log in Athena utilizzando la proiezione delle partizioni

Poiché CloudTrail i log hanno una struttura nota il cui schema di partizione è possibile specificare in anticipo, è possibile ridurre il tempo di esecuzione delle query e automatizzare la gestione delle partizioni utilizzando la funzionalità di proiezione delle partizioni Athena. La proiezione delle partizioni aggiunge automaticamente nuove partizioni man mano che vengono aggiunti nuovi dati. Ciò elimina la necessità di aggiungere manualmente le partizioni utilizzando `ALTER TABLE ADD PARTITION`.

L'`CREATE TABLE`istruzione di esempio che segue utilizza automaticamente la proiezione delle partizioni sui CloudTrail registri da una data specificata a quella attuale per un singolo registro. Regione AWS Nelle clausole `LOCATION` e `storage.location.template`, sostituire i segnaposto *`bucket`*, *`account-id`* e *`aws-region`* con valori corrispondenti identici. Per `projection.timestamp.range`, sostituire *`2020`*/*`01`*/*`01`* con la data di inizio che si desidera usare. Dopo aver eseguito la query con esito positivo, è possibile eseguire query sulla tabella. Non è necessario eseguire `ALTER TABLE ADD PARTITION` per caricare le partizioni.

```
CREATE EXTERNAL TABLE cloudtrail_logs_pp(
    eventVersion STRING,
    userIdentity STRUCT<
        type: STRING,
        principalId: STRING,
        arn: STRING,
        accountId: STRING,
        invokedBy: STRING,
        accessKeyId: STRING,
        userName: STRING,
        sessionContext: STRUCT<
            attributes: STRUCT<
                mfaAuthenticated: STRING,
                creationDate: STRING>,
            sessionIssuer: STRUCT<
                type: STRING,
                principalId: STRING,
                arn: STRING,
                accountId: STRING,
                userName: STRING>,
            ec2RoleDelivery:string,
            webIdFederationData: STRUCT<
                federatedProvider: STRING,
                attributes: map<string,string>
            >
        >
```

```
    >,
    eventTime STRING,
    eventSource STRING,
    eventName STRING,
    awsRegion STRING,
    sourceIpAddress STRING,
    userAgent STRING,
    errorCode STRING,
    errorMessage STRING,
    requestparameters STRING,
    responseelements STRING,
    additionaleventdata STRING,
    requestId STRING,
    eventId STRING,
    readOnly STRING,
    resources ARRAY<STRUCT<
        arn: STRING,
        accountId: STRING,
        type: STRING>>,
    eventType STRING,
    apiVersion STRING,
    recipientAccountId STRING,
    serviceEventDetails STRING,
    sharedEventID STRING,
    vpcendpointid STRING,
    eventCategory STRING,
    tlsDetails struct<
        tlsVersion:string,
        cipherSuite:string,
        clientProvidedHostHeader:string>
  )
PARTITIONED BY (
   `timestamp` string)
ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe'
STORED AS INPUTFORMAT 'com.amazon.emr.cloudtrail.CloudTrailInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET/AWSLogs/account-id/CloudTrail/aws-region'
TBLPROPERTIES (
  'projection.enabled'='true',
  'projection.timestamp.format'='yyyy/MM/dd',
  'projection.timestamp.interval'='1',
  'projection.timestamp.interval.unit'='DAYS',
  'projection.timestamp.range'='2020/01/01,NOW',
```

```
  'projection.timestamp.type'='date',
  'storage.location.template'='s3://DOC-EXAMPLE-BUCKET/AWSLogs/account-id/
CloudTrail/aws-region/${timestamp}')
```

Per maggiori informazioni sulla proiezione delle partizioni, consulta Proiezione delle partizioni con Amazon Athena.

Esecuzione di query nei campi nidificati

Poiché i campi `userIdentity` e `resources` sono tipi di dati nidificati, l'esecuzioni di query richiede un trattamento speciale.

L'oggetto `userIdentity` è costituito da tipi nidificati `STRUCT`. È possibile eseguire query utilizzando un punto per separare i campi, come nell'esempio seguente:

```
SELECT
    eventsource,
    eventname,
    useridentity.sessioncontext.attributes.creationdate,
    useridentity.sessioncontext.sessionissuer.arn
FROM cloudtrail_logs
WHERE useridentity.sessioncontext.sessionissuer.arn IS NOT NULL
ORDER BY eventsource, eventname
LIMIT 10
```

Il campo `resources` è un array di oggetti `STRUCT`. Per questi array, utilizzare `CROSS JOIN UNNEST` per annullare l'array in modo da poter interrogare i suoi oggetti.

L'esempio seguente restituisce tutte le righe in cui la risorsa ARN termina in `example/datafile.txt`. Per la leggibilità, la funzione replace rimuove la sottostringa iniziale `arn:aws:s3:::` dall'ARN.

```
SELECT
    awsregion,
    replace(unnested.resources_entry.ARN,'arn:aws:s3:::') as s3_resource,
    eventname,
    eventtime,
    useragent
FROM cloudtrail_logs t
CROSS JOIN UNNEST(t.resources) unnested (resources_entry)
WHERE unnested.resources_entry.ARN LIKE '%example/datafile.txt'
ORDER BY eventtime
```

Di seguito sono illustrati alcuni esempi di query per gli eventi `DeleteBucket`. La query estrae il nome del bucket e l'ID account a cui appartiene il bucket dall'oggetto `resources`.

```
SELECT
    awsregion,
    replace(unnested.resources_entry.ARN,'arn:aws:s3:::') as deleted_bucket,
    eventtime AS time_deleted,
    useridentity.username,
    unnested.resources_entry.accountid as bucket_acct_id
FROM cloudtrail_logs t
CROSS JOIN UNNEST(t.resources) unnested (resources_entry)
WHERE eventname = 'DeleteBucket'
ORDER BY eventtime
```

Per ulteriori informazioni sull'annullamento della nidificazione, consulta Filtraggio delle matrici.

Query di esempio

L'esempio seguente mostra una parte di una query che restituisce tutte le richieste anonime (non firmate) dalla tabella creata per i registri degli eventi. CloudTrail Questa query seleziona le richieste in cui `useridentity.accountid` è anonimo e `useridentity.arn` non è specificato:

```
SELECT *
FROM cloudtrail_logs
WHERE
    eventsource = 's3.amazonaws.com' AND
    eventname in ('GetObject') AND
    useridentity.accountid = 'anonymous' AND
    useridentity.arn IS NULL AND
    requestparameters LIKE '%[your bucket name ]%';
```

Per ulteriori informazioni, consulta il post sul blog AWS Big Data Analyze security, compliance and operations activity using AWS CloudTrail and Amazon Athena.

Suggerimenti per l' CloudTrail interrogazione dei log

Per esplorare i dati dei CloudTrail log, utilizza questi suggerimenti:

- Prima di eseguire query sui log, verifica che la tabella di log sia uguale a quella definita in the section called “Creazione di una tabella per CloudTrail i log in Athena utilizzando il partizionamento manuale”. Se non è la prima tabella, elimina la tabella esistente utilizzando il comando: `DROP TABLE cloudtrail_logs`.

- Dopo aver eliminato la tabella esistente, ricreala. Per ulteriori informazioni, consulta Creazione di una tabella per CloudTrail i log in Athena utilizzando il partizionamento manuale.

  Verifica che i campi della query Athena siano elencati correttamente. Per informazioni sull'elenco completo dei campi in un CloudTrail record, consulta il contenuto del CloudTrail record.

  Se la query include campi nei formati JSON, ad esempio `STRUCT`, estrarre i dati da JSON. Per ulteriori informazioni, consulta Estrazione di dati JSON dalle stringhe.

  Alcuni suggerimenti per eseguire interrogazioni sulla tabella: CloudTrail
- Inizia osservando quali utenti hanno eseguito specifiche operazioni API e gli indirizzi IP di origine.
- Utilizza la seguente query SQL di base come modello. Incolla la query nella console Athena ed eseguila.

```
SELECT
 useridentity.arn,
 eventname,
 sourceipaddress,
 eventtime
FROM cloudtrail_logs
LIMIT 100;
```

- Modifica la query per esplorare ulteriormente i dati.
- Per migliorare le prestazioni, includi la clausola `LIMIT` per ottenere uno specifico sottoinsieme di righe.

Esecuzione di query sui log Amazon EMR

Amazon EMR e le applicazioni Big Data che vengono eseguite su Amazon EMR generano file di log. I file di log vengono scritti sul nodo primario e puoi anche configurare Amazon EMR per archiviare automaticamente i file di log su Amazon S3. È possibile utilizzare Amazon Athena per eseguire query su questi log per identificare eventi e tendenze per applicazioni e cluster. Per ulteriori informazioni sui tipi di file di log in Amazon EMR e sul loro salvataggio in Amazon S3, consulta Visualizzazione di file di log nella Guida alla gestione di Amazon EMR.

Creazione ed esecuzione di query su una tabella di base basata sui file di log di Amazon EMR

Nell'esempio seguente viene creata una tabella di base `myemrlogs`, basata sui file di log salvati in `s3://aws-logs-`*`123456789012`*`-`*`us-west-2`*`/elasticmapreduce/`*`j-2ABCDE34F5GH6`*`/`

`elasticmapreduce/`. La posizione Amazon S3 utilizzata negli esempi seguenti riflette il modello di posizione di log predefinito per un cluster EMR creato dall'account Amazon Web Services *123456789012* nella Regione *us-west-2*. Se si utilizza una posizione personalizzata, il modello è s3://DOC-EXAMPLE-BUCKET/ *ClusterId.*

Per informazioni sulla creazione di una tabella partizionata per migliorare potenzialmente le prestazioni delle query e ridurre il trasferimento dei dati, consulta Creazione ed esecuzione di query su una tabella partizionata in base ai log di Amazon EMR.

```
CREATE EXTERNAL TABLE `myemrlogs`(
  `data` string COMMENT 'from deserializer')
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|'
LINES TERMINATED BY '\n'
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  's3://aws-logs-123456789012-us-west-2/elasticmapreduce/j-2ABCDE34F5GH6'
```

Le query di esempio seguenti possono essere eseguite sulla tabella `myemrlogs` creata dall'esempio precedente.

Example Esecuzione di query su log di fase per occorrenze di ERROR, WARN, INFO, EXCEPTION, FATAL o DEBUG

```
SELECT data,
        "$PATH"
FROM "default"."myemrlogs"
WHERE regexp_like("$PATH",'s-86URH188Z6B1')
        AND regexp_like(data, 'ERROR|WARN|INFO|EXCEPTION|FATAL|DEBUG') limit 100;
```

Example Esecuzione di query su log di istanze specifiche, i-00b3c0a839ece0a9c, per ERROR, WARN, INFO, EXCEPTION, FATAL o DEBUG

```
SELECT "data",
        "$PATH" AS filepath
FROM "default"."myemrlogs"
WHERE regexp_like("$PATH",'i-00b3c0a839ece0a9c')
        AND regexp_like("$PATH",'state')
```

```
        AND regexp_like(data, 'ERROR|WARN|INFO|EXCEPTION|FATAL|DEBUG') limit 100;
```

Example Esecuzione di query su log di applicazioni Presto per ERROR, WARN, INFO, EXCEPTION, FATAL o DEBUG

```
SELECT "data",
        "$PATH" AS filepath
FROM "default"."myemrlogs"
WHERE regexp_like("$PATH",'presto')
        AND regexp_like(data, 'ERROR|WARN|INFO|EXCEPTION|FATAL|DEBUG') limit 100;
```

Example Esecuzione di query su log di applicazioni Namenode per ERROR, WARN, INFO, EXCEPTION, FATAL o DEBUG

```
SELECT "data",
        "$PATH" AS filepath
FROM "default"."myemrlogs"
WHERE regexp_like("$PATH",'namenode')
        AND regexp_like(data, 'ERROR|WARN|INFO|EXCEPTION|FATAL|DEBUG') limit 100;
```

Example Esecuzione di query su tutti i log per data e ora per ERROR, WARN, INFO, EXCEPTION, FATAL o DEBUG

```
SELECT distinct("$PATH") AS filepath
FROM "default"."myemrlogs"
WHERE regexp_like("$PATH",'2019-07-23-10')
        AND regexp_like(data, 'ERROR|WARN|INFO|EXCEPTION|FATAL|DEBUG') limit 100;
```

Creazione ed esecuzione di query su una tabella partizionata in base ai log di Amazon EMR

Questi esempi utilizzano la stessa posizione dei log per creare una tabella Athena, ma la tabella viene partizionata e viene quindi creata una partizione per ogni posizione. Per ulteriori informazioni, consulta Partizionamento dei dati in Athena.

La query seguente crea la tabella partizionata denominata `mypartitionedemrlogs`:

```
CREATE EXTERNAL TABLE `mypartitionedemrlogs`(
  `data` string COMMENT 'from deserializer')
 partitioned by (logtype string)
```

```
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '|'
LINES TERMINATED BY '\n'
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://aws-logs-123456789012-us-west-2/elasticmapreduce/j-2ABCDE34F5GH6'
```

Le istruzioni di query seguenti creano quindi partizioni di tabella basate su sottodirectory per diversi tipi di log che Amazon EMR crea in Amazon S3:

```
ALTER TABLE mypartitionedemrlogs ADD
     PARTITION (logtype='containers')
     LOCATION 's3://aws-logs-123456789012-us-west-2/elasticmapreduce/j-2ABCDE34F5GH6/
containers/'
```

```
ALTER TABLE mypartitionedemrlogs ADD
     PARTITION (logtype='hadoop-mapreduce')
     LOCATION 's3://aws-logs-123456789012-us-west-2/elasticmapreduce/j-2ABCDE34F5GH6/
hadoop-mapreduce/'
```

```
ALTER TABLE mypartitionedemrlogs ADD
     PARTITION (logtype='hadoop-state-pusher')
     LOCATION 's3://aws-logs-123456789012-us-west-2/elasticmapreduce/j-2ABCDE34F5GH6/
hadoop-state-pusher/'
```

```
ALTER TABLE mypartitionedemrlogs ADD
     PARTITION (logtype='node')
     LOCATION 's3://aws-logs-123456789012-us-west-2/elasticmapreduce/j-2ABCDE34F5GH6/
node/'
```

```
ALTER TABLE mypartitionedemrlogs ADD
     PARTITION (logtype='steps')
     LOCATION 's3://aws-logs-123456789012-us-west-2/elasticmapreduce/j-2ABCDE34F5GH6/
steps/'
```

Dopo aver creato le partizioni, puoi eseguire una query `SHOW PARTITIONS` nella tabella per confermare:

```
SHOW PARTITIONS mypartitionedemrlogs;
```

Negli esempi seguenti vengono illustrate le query per voci di log specifiche che utilizzano la tabella e le partizioni create dagli esempi precedenti.

Example Esecuzione di query sui log di applicazioni application_1561661818238_0002 nella partizione dei contenitori per ERROR o WARN

```
SELECT data,
        "$PATH"
FROM "default"."mypartitionedemrlogs"
WHERE logtype='containers'
        AND regexp_like("$PATH",'application_1561661818238_0002')
        AND regexp_like(data, 'ERROR|WARN') limit 100;
```

Example Esecuzione di query sulla partizione Hadoop-Mapreduce per l'attività job_1561661818238_0004 e Failed Reduces

```
SELECT data,
        "$PATH"
FROM "default"."mypartitionedemrlogs"
WHERE logtype='hadoop-mapreduce'
        AND regexp_like(data,'job_1561661818238_0004|Failed Reduces') limit 100;
```

Example Esecuzione di query sui log Hive nella partizione dei nodi per l'ID query 056e0609-33e1-4611-956c-7a31b42d2663

```
SELECT data,
        "$PATH"
FROM "default"."mypartitionedemrlogs"
WHERE logtype='node'
        AND regexp_like("$PATH",'hive')
        AND regexp_like(data,'056e0609-33e1-4611-956c-7a31b42d2663') limit 100;
```

Example Esecuzione di query sui log ResourceManager nella partizione dei nodi per l'applicazione 1567660019320_0001_01_000001

```
SELECT data,
        "$PATH"
```

```
FROM "default"."mypartitionedemrlogs"
WHERE logtype='node'
        AND regexp_like(data,'resourcemanager')
        AND regexp_like(data,'1567660019320_0001_01_000001') limit 100
```

Interrogazione dei log di AWS Global Accelerator flusso

Puoi utilizzarli AWS Global Accelerator per creare acceleratori che indirizzano il traffico di rete verso endpoint ottimali sulla rete globale. AWS Per ulteriori informazioni su Global Accelerator, consulta What is. AWS Global Accelerator

I log di flusso Global Accelerator ti consentono di acquisire informazioni sul traffico degli indirizzi IP in entrata e in uscita dalle interfacce di rete negli acceleratori. I dati dei log di flusso vengono pubblicati in Amazon S3, dove è possibile recuperare e visualizzare i dati. Per ulteriori informazioni, consulta Log di flusso in AWS Global Accelerator.

Puoi utilizzare Athena per eseguire query sui log di flusso Global Accelerator creando una tabella che ne specifica la posizione in Amazon S3.

Per creare la tabella per i log di flusso di Global Accelerator

1. Copiare e incollare la seguente istruzione DDL nella console Athena. Questa query specifica ROW FORMAT DELIMITED e omette di specificare a SerDe, il che significa che la query utilizza il. `LazySimpleSerDe` In questa query, i campi terminano con uno spazio.

   ```
   CREATE EXTERNAL TABLE IF NOT EXISTS aga_flow_logs (
     version string,
     account string,
     acceleratorid string,
     clientip string,
     clientport int,
     gip string,
     gipport int,
     endpointip string,
     endpointport int,
     protocol string,
     ipaddresstype string,
     numpackets bigint,
     numbytes int,
     starttime int,
     endtime int,
     action string,
   ```

```
  logstatus string,
  agasourceip string,
  agasourceport int,
  endpointregion string,
  agaregion string,
  direction string
)
PARTITIONED BY (dt string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ' '
LOCATION 's3://DOC-EXAMPLE-BUCKET/prefix/AWSLogs/account_id/globalaccelerator/
region/'
TBLPROPERTIES ("skip.header.line.count"="1");
```

2. Modificare il valore `LOCATION` in modo che punti al bucket Amazon S3 che contiene i dati di log.

```
's3://DOC-EXAMPLE-BUCKET/prefix/AWSLogs/account_id/globalaccelerator/region_code/'
```

3. Eseguire la query nella console Athena. Una volta completata la query, Athena registra la tabella `aga_flow_logs`, rendendo i dati in essa contenuti disponibili per l'esecuzione di query.
4. Creare partizioni per leggere i dati, come nella seguente query di esempio. La query crea una singola partizione per una data specificata. Sostituire i segnaposto per data e posizione.

```
ALTER TABLE aga_flow_logs
ADD PARTITION (dt='YYYY-MM-dd')
LOCATION 's3://DOC-EXAMPLE-BUCKET/prefix/AWSLogs/account_id/
globalaccelerator/region_code/YYYY/MM/dd';
```

Query di esempio per i log di flusso AWS Global Accelerator

Example Elencare le richieste che passano attraverso una edge location specifica

La query di esempio seguente elenca le richieste passate attraverso la edge location LHR. Utilizza l'operatore `LIMIT` per limitare il numero di log di cui eseguire la query alla volta.

```
SELECT
  clientip,
  agaregion,
  protocol,
  action
FROM
```

```
  aga_flow_logs
WHERE
  agaregion LIKE 'LHR%'
LIMIT
  100;
```

Example Elencare gli indirizzi IP dell'endpoint che ricevono la maggior parte delle richieste HTTPS

Utilizza la query seguente per verificare quali indirizzi IP dell'endpoint ricevono il numero più elevato di richieste HTTPS. La query conta il numero di pacchetti ricevuti sulla porta HTTPS 443, li raggruppa in base all'indirizzo IP di destinazione e restituisce i primi 10 indirizzi IP.

```
SELECT
  SUM(numpackets) AS packetcount,
  endpointip
FROM
  aga_flow_logs
WHERE
  endpointport = 443
GROUP BY
  endpointip
ORDER BY
  packetcount DESC
LIMIT
  10;
```

Interrogazione dei risultati di Amazon GuardDuty

Amazon GuardDuty è un servizio di monitoraggio della sicurezza che aiuta a identificare attività impreviste e potenzialmente non autorizzate o dannose nel tuo AWS ambiente. Quando rileva attività impreviste e potenzialmente dannose, GuardDuty genera risultati di sicurezza che puoi esportare in Amazon S3 per l'archiviazione e l'analisi. Dopo aver esportato i risultati in Amazon S3, è possibile utilizzare Athena per eseguire le query. Questo articolo mostra come creare una tabella in Athena per i GuardDuty risultati e interrogarli.

Per ulteriori informazioni su Amazon GuardDuty, consulta la Amazon GuardDuty User Guide.

Prerequisiti

- Abilita la GuardDuty funzionalità per esportare i risultati in Amazon S3. Per i passaggi, consulta Esportazione dei risultati nella Amazon GuardDuty User Guide.

Creazione di una tabella in Athena per i risultati GuardDuty

Per interrogare GuardDuty i tuoi risultati con Athena, devi creare una tabella per essi.

Per creare una tabella in Athena per i risultati GuardDuty

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Incollare la seguente istruzione DDL nella console Athena. Modifica i valori in `LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/account-id/GuardDuty/'` modo che rimandino ai tuoi GuardDuty risultati in Amazon S3.

```
CREATE EXTERNAL TABLE `gd_logs` (
  `schemaversion` string,
  `accountid` string,
  `region` string,
  `partition` string,
  `id` string,
  `arn` string,
  `type` string,
  `resource` string,
  `service` string,
  `severity` string,
  `createdat` string,
  `updatedat` string,
  `title` string,
  `description` string)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
 LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/account-id/GuardDuty/'
 TBLPROPERTIES ('has_encrypted_data'='true')
```

> **Note**
>
> SerDe Si aspetta che ogni documento JSON si trovi su una singola riga di testo senza caratteri di terminazione di riga che separano i campi del record. Se il testo JSON è in un bel formato di stampa, potresti ricevere un messaggio di errore come HIVE_CURSOR_ERROR: Row is not a valid JSON Object o HIVE_CURSOR_ERROR:: Unexpected end-of-input: expected: expected close marker for OBJECT quando tenti di interrogare la tabella dopo averla JsonParseException creata. Per ulteriori informazioni, consulta JSON Data Files nella documentazione di SerDe OpenX su. GitHub

3. Eseguire la query nella console Athena per registrare la tabella `gd_logs`. Al termine della query, i risultati sono pronti per le query da Athena.

Query di esempio

Gli esempi seguenti mostrano come interrogare GuardDuty i risultati di Athena.

Example Estrazione dei dati DNS

La query seguente restituisce informazioni relative alle istanze Amazon EC2 che potrebbero estrarre i dati tramite query DNS.

```
SELECT
    title,
    severity,
    type,
    id AS FindingID,
    accountid,
    region,
    createdat,
    updatedat,
    json_extract_scalar(service, '$.count') AS Count,
    json_extract_scalar(resource, '$.instancedetails.instanceid') AS InstanceID,
    json_extract_scalar(service, '$.action.actiontype') AS DNS_ActionType,
    json_extract_scalar(service, '$.action.dnsrequestaction.domain') AS DomainName,
    json_extract_scalar(service, '$.action.dnsrequestaction.protocol') AS protocol,
    json_extract_scalar(service, '$.action.dnsrequestaction.blocked') AS blocked
FROM gd_logs
WHERE type = 'Trojan:EC2/DNSDataExfiltration'
ORDER BY severity DESC
```

Example - Accesso utente IAM non autorizzato

La query seguente restituisce tutti i tipi di risultati `UnauthorizedAccess:IAMUser` per un'entità principale IAM da tutte le Regioni.

```
SELECT title,
         severity,
         type,
         id,
         accountid,
```

```
         region,
         createdat,
         updatedat,
         json_extract_scalar(service, '$.count') AS Count,
         json_extract_scalar(resource, '$.accesskeydetails.username') AS IAMPrincipal,
         json_extract_scalar(service,'$.action.awsapicallaction.api') AS
 APIActionCalled
FROM gd_logs
WHERE type LIKE '%UnauthorizedAccess:IAMUser%'
ORDER BY severity desc;
```

Suggerimenti per interrogare i risultati GuardDuty

Quando si crea la query, tenere a mente i seguenti punti.

- Per estrarre dati dai campi JSON nidificati, utilizzare le funzioni `json_extract` o `json_extract_scalar` di Presto. Per ulteriori informazioni, consulta Estrazione di dati JSON dalle stringhe.
- Assicurarsi che tutti i caratteri nei campi JSON siano in minuscolo.
- Per informazioni sul download dei risultati delle query, consulta Download dei file dei risultati delle query mediante la console Athena.

Interrogazione dei log AWS Network Firewall

AWS Network Firewall è un servizio gestito che puoi utilizzare per implementare protezioni di rete essenziali per le tue istanze di Amazon Virtual Private Cloud. AWS Network Firewall collabora con te AWS Firewall Manager in modo da poter creare politiche basate su AWS Network Firewall regole e poi applicarle centralmente ai tuoi VPC e account. Per ulteriori informazioni su AWS Network Firewall, consulta AWS Network Firewall.

È possibile configurare AWS Network Firewall la registrazione del traffico da inoltrare al motore stateful rules del firewall. I log forniscono informazioni dettagliate sul traffico di rete, tra cui l'ora in cui il motore con stato ha ricevuto un pacchetto, informazioni dettagliate sul pacchetto e qualsiasi operazione di regola con stato eseguita rispetto al pacchetto. I log vengono pubblicati nella destinazione di log configurata, dove è possibile recuperarli e visualizzarli. Per ulteriori informazioni, consulta Registrazione del traffico di rete da AWS Network Firewallnella Guida per gli sviluppatori di AWS Network Firewall .

Crea una tabella per i registri degli avvisi

1. Modificare la seguente istruzione DDL di esempio in modo che sia conforme alla struttura del registro degli avvisi. Potrebbe essere necessario aggiornare l'istruzione per includere le colonne per la versione più recente dei log. Per ulteriori informazioni, consulta Contenuti di un log firewall nella Guida per gli sviluppatori di AWS Network Firewall .

```
CREATE EXTERNAL TABLE network_firewall_alert_logs (
  firewall_name string,
  availability_zone string,
  event_timestamp string,
  event struct<
    timestamp:string,
    flow_id:bigint,
    event_type:string,
    src_ip:string,
    src_port:int,
    dest_ip:string,
    dest_port:int,
    proto:string,
    app_proto:string,
    tls_inspected:boolean,
    alert:struct<
      alert_id:string,
      alert_type:string,
      action:string,
      signature_id:int,
      rev:int,
      signature:string,
      category:string,
      severity:int,
      rule_name:string,
      alert_name:string,
      alert_severity:string,
      alert_description:string,
      file_name:string,
      file_hash:string,
      packet_capture:string,
      reference_links:array<string>
    >,
    src_country:string,
    dest_country:string,
    src_hostname:string,
```

```
    dest_hostname:string,
    user_agent:string,
    url:string
   >
)
 ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
 LOCATION 's3://DOC-EXAMPLE-BUCKET/path_to_alert_logs_folder/';
```

2. Modifica la `LOCATION` clausola per specificare la cartella per i log in Amazon S3.
3. Esegui la tua `CREATE TABLE` query nell'editor di query Athena. Una volta completata la query, Athena registra `network_firewall_alert_logs` la tabella, rendendo i dati a cui punta pronti per le query.

Esempio di interrogazione di esempio nel registro

L'esempio di query del registro degli avvisi in questa sezione filtra gli eventi in cui è stata eseguita l'ispezione TLS con avvisi con un livello di gravità pari o superiore a 2.

La query utilizza alias per creare intestazioni di colonna di output che mostrano a cosa appartiene la `struct` colonna. Ad esempio, l'intestazione di colonna del `event.alert.category` campo è `event_alert_category` invece di una semplice. `category` Per personalizzare ulteriormente i nomi delle colonne, puoi modificare gli alias in base alle tue preferenze. Ad esempio, è possibile utilizzare caratteri di sottolineatura o altri separatori per delimitare i nomi e `struct` i nomi dei campi.

Ricordatevi di modificare i nomi e i `struct` riferimenti delle colonne in base alla definizione della tabella e ai campi che desiderate inserire nel risultato della query.

```
SELECT
  firewall_name,
  availability_zone,
  event_timestamp,
  event.timestamp AS event_timestamp,
  event.flow_id AS event_flow_id,
  event.event_type AS event_type,
  event.src_ip AS event_src_ip,
  event.src_port AS event_src_port,
  event.dest_ip AS event_dest_ip,
  event.dest_port AS event_dest_port,
  event.proto AS event_protol,
  event.app_proto AS event_app_proto,
  event.tls_inspected AS event_tls_inspected,
```

```
  event.alert.alert_id AS event_alert_alert_id,
  event.alert.alert_type AS event_alert_alert_type,
  event.alert.action AS event_alert_action,
  event.alert.signature_id AS event_alert_signature_id,
  event.alert.rev AS event_alert_rev,
  event.alert.signature AS event_alert_signature,
  event.alert.category AS event_alert_category,
  event.alert.severity AS event_alert_severity,
  event.alert.rule_name AS event_alert_rule_name,
  event.alert.alert_name AS event_alert_alert_name,
  event.alert.alert_severity AS event_alert_alert_severity,
  event.alert.alert_description AS event_alert_alert_description,
  event.alert.file_name AS event_alert_file_name,
  event.alert.file_hash AS event_alert_file_hash,
  event.alert.packet_capture AS event_alert_packet_capture,
  event.alert.reference_links AS event_alert_reference_links,
  event.src_country AS event_src_country,
  event.dest_country AS event_dest_country,
  event.src_hostname AS event_src_hostname,
  event.dest_hostname AS event_dest_hostname,
  event.user_agent AS event_user_agent,
  event.url AS event_url
FROM
  network_firewall_alert_logs
WHERE
  event.alert.severity >= 2
  AND event.tls_inspected = true
LIMIT 10;
```

Crea una tabella per i log di netflow

1. Modifica la seguente istruzione DDL di esempio per renderla conforme alla struttura dei log di netflow. Potrebbe essere necessario aggiornare l'istruzione per includere le colonne per la versione più recente dei log. Per ulteriori informazioni, consulta Contenuti di un log firewall nella Guida per gli sviluppatori di AWS Network Firewall .

   ```
   CREATE EXTERNAL TABLE network_firewall_netflow_logs (
     firewall_name string,
     availability_zone string,
     event_timestamp string,
     event struct<
       timestamp:string,
       flow_id:bigint,
   ```

```
    event_type:string,
    src_ip:string,
    src_port:int,
    dest_ip:string,
    dest_port:int,
    proto:string,
    app_proto:string,
    netflow:struct<
      pkts:int,
      bytes:bigint,
      start:string,
      `end`:string,
      age:int,
      min_ttl:int,
      max_ttl:int,
      tcp_flags:struct<
        syn:boolean,
        fin:boolean,
        rst:boolean,
        psh:boolean,
        ack:boolean,
        urg:boolean
      >,
      tls_inspected:boolean
    >
  >
)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
LOCATION 's3://DOC-EXAMPLE-BUCKET/path_to_netflow_logs_folder/';
```

2. Modifica la `LOCATION` clausola per specificare la cartella per i log in Amazon S3.
3. Esegui la `CREATE TABLE` query nell'editor di query Athena. Una volta completata la query, Athena registra `network_firewall_netflow_logs` la tabella, rendendo i dati a cui punta pronti per le query.

Esempio di interrogazione del registro Netflow

La query di log netflow di esempio in questa sezione filtra gli eventi in cui è stata eseguita l'ispezione TLS.

La query utilizza alias per creare intestazioni di colonna di output che mostrano l'appartenenza della `struct` colonna. Ad esempio, l'intestazione di colonna del `event.netflow.bytes` campo

è `event_netflow_bytes` invece di una semplice. `bytes` Per personalizzare ulteriormente i nomi delle colonne, puoi modificare gli alias in base alle tue preferenze. Ad esempio, è possibile utilizzare caratteri di sottolineatura o altri separatori per delimitare i nomi e `struct` i nomi dei campi.

Ricordatevi di modificare i nomi e i `struct` riferimenti delle colonne in base alla definizione della tabella e ai campi che desiderate inserire nel risultato della query.

```
SELECT
  event.src_ip AS event_src_ip,
  event.dest_ip AS event_dest_ip,
  event.proto AS event_proto,
  event.app_proto AS event_app_proto,
  event.netflow.pkts AS event_netflow_pkts,
  event.netflow.bytes AS event_netflow_bytes,
  event.netflow.tcp_flags.syn AS event_netflow_tcp_flags_syn,
  event.netflow.tls_inspected AS event_netflow_tls_inspected
FROM network_firewall_netflow_logs
WHERE event.netflow.tls_inspected = true
```

Esecuzione di query sui log di Network Load Balancer

Utilizzare Athena per analizzare ed elaborare i log di Network Load Balancer. Questi log ricevono informazioni dettagliate sulle richieste Transport Layer Security (TLS) inviate al Network Load Balancer. Puoi utilizzare questi log per analizzare i modelli di traffico e risolvere i problemi che potresti incontrare.

Prima di analizzare i log di accesso di Network Load Balancer, abilitarli e configurali affinché siano salvati nel bucket Amazon S3 di destinazione. Per ulteriori informazioni e per informazioni su ogni voce del log di accesso a Network Load Balancer, consulta Log di accesso al Network Load Balancer.

- Creare la tabella per i log di Network Load Balancer
- Query di esempio di Network Load Balancer

Per creare la tabella per i log di Network Load Balancer

1. Copiare e incollare la seguente istruzione DDL nella console Athena. Controlla la sintassi dei record di log di Network Load Balancer. Potrebbe essere necessario aggiornare la seguente query per includere le colonne e la sintassi Regex per la versione più recente del record.

```
CREATE EXTERNAL TABLE IF NOT EXISTS nlb_tls_logs (
            type string,
            version string,
            time string,
            elb string,
            listener_id string,
            client_ip string,
            client_port int,
            target_ip string,
            target_port int,
            tcp_connection_time_ms double,
            tls_handshake_time_ms double,
            received_bytes bigint,
            sent_bytes bigint,
            incoming_tls_alert int,
            cert_arn string,
            certificate_serial string,
            tls_cipher_suite string,
            tls_protocol_version string,
            tls_named_group string,
            domain_name string,
            alpn_fe_protocol string,
            alpn_be_protocol string,
            alpn_client_preference_list string,
            tls_connection_creation_time string
            )
            ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.RegexSerDe'
            WITH SERDEPROPERTIES (
            'serialization.format' = '1',
            'input.regex' =
        '([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*):([0-9]*) ([^ ]*):([0-9]*)
 ([-.0-9]*) ([-.0-9]*) ([-0-9]*) ([-0-9]*) ([-0-9]*) ([^ ]*) ([^ ]*) ([^ ]*)
 ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*) ([^ ]*)$')
            LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/AWS_account_ID/
elasticloadbalancing/region';
```

2. Modifica il bucket Amazon S3 `LOCATION` per specificare la destinazione dei tuoi log di Network Load Balancer.
3. Eseguire la query nella console Athena. Una volta completata la query, Athena registra la tabella `nlb_tls_logs`, rendendo i dati in essa contenuti pronti per le query.

Query di esempio di Network Load Balancer

Per vedere quante volte viene utilizzato un certificato, utilizzare una query simile a questo esempio:

```
SELECT count(*) AS
         ct,
         cert_arn
FROM "nlb_tls_logs"
GROUP BY  cert_arn;
```

La query seguente mostra come molti utenti utilizzano una versione di TLS precedente alla 1.3:

```
SELECT tls_protocol_version,
         COUNT(tls_protocol_version) AS
         num_connections,
         client_ip
FROM "nlb_tls_logs"
WHERE tls_protocol_version < 'tlsv13'
GROUP BY tls_protocol_version, client_ip;
```

Utilizzare la seguente query per identificare le connessioni che richiedono un tempo di handshake TLS lungo:

```
SELECT *
FROM "nlb_tls_logs"
ORDER BY  tls_handshake_time_ms DESC
LIMIT 10;
```

Utilizzare la seguente query per identificare e contare quali versioni del protocollo TLS e suite di cifratura sono state negoziate negli ultimi 30 giorni.

```
SELECT tls_cipher_suite,
         tls_protocol_version,
         COUNT(*) AS ct
FROM "nlb_tls_logs"
WHERE from_iso8601_timestamp(time) > current_timestamp - interval '30' day
        AND NOT tls_protocol_version = '-'
GROUP BY tls_cipher_suite, tls_protocol_version
ORDER BY ct DESC;
```

Esecuzione di query sui log di Amazon Route 53 Resolver

Puoi creare tabelle Athena per i log delle query di Amazon Route 53 Resolver ed eseguire query da Athena.

I log delle query del Route 53 Resolver consentono la registrazione delle query DNS eseguite da risorse all'interno di risorse VPC on-premise che utilizzano endpoint Resolver in ingresso, query che utilizzano un endpoint Resolver in uscita per la risoluzione DNS ricorsiva e query che utilizzano regole del firewall DNS di Route 53 Resolver Route per bloccare, consentire o monitorare un elenco di domini. Per ulteriori informazioni, consulta Log delle query Resolver nella Guida per sviluppatori di Amazon Route 53. Per informazioni su ciascuno dei campi nei log, consulta Valori visualizzati nei log di query di Resolver nella Guida per sviluppatori di Amazon Route 53.

Creazione della tabella per i log di query di Resolver

È possibile utilizzare l'editor di query nella console Athena per creare e interrogare una tabella per i log delle query di Route 53 Resolver.

Per creare ed eseguire query su una tabella Athena per i log delle query di Route 53 Resolver

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Nell'editor di query Athena, inserire la seguente istruzione `CREATE TABLE`. Sostituire i valori della clausola `LOCATION` con quelli corrispondenti alla posizione dei log di Resolver nel bucket Amazon S3.

```
CREATE EXTERNAL TABLE r53_rlogs (
  version string,
  account_id string,
  region string,
  vpc_id string,
  query_timestamp string,
  query_name string,
  query_type string,
  query_class
    string,
  rcode string,
  answers array<
    struct<
      Rdata: string,
      Type: string,
      Class: string>
```

```
    >,
  srcaddr string,
  srcport int,
  transport string,
  srcids struct<
    instance: string,
    resolver_endpoint: string
    >,
  firewall_rule_action string,
  firewall_rule_group_id string,
  firewall_domain_list_id string
 )

ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
LOCATION 's3://DOC-EXAMPLE-BUCKET/AWSLogs/aws_account_id/vpcdnsquerylogs/{vpc-id}/'
```

Poiché i dati del registro delle query di Resolver sono in formato JSON, l'istruzione CREATE TABLE utilizza una SerDelibreria JSON per analizzare i dati.

> **Note**
>
> SerDe Si aspetta che ogni documento JSON si trovi su una singola riga di testo senza caratteri di terminazione di riga che separano i campi del record. Se il testo JSON è in un bel formato di stampa, potresti ricevere un messaggio di errore come HIVE_CURSOR_ERROR: Row is not a valid JSON Object o HIVE_CURSOR_ERROR:: Unexpected end-of-input: expected: expected close marker for OBJECT quando tenti di interrogare la tabella dopo averla JsonParseException creata. Per ulteriori informazioni, consulta JSON Data Files nella documentazione di SerDe OpenX su. GitHub

3. Scegli Esegui query. L'istruzione crea una tabella Athena denominata `r53_rlogs` le cui colonne rappresentano ciascuno dei campi nei dati dei log di Resolver.
4. Nell'editor di query della console Athena, eseguire la seguente query per verificare che la tabella sia stata creata.

```
SELECT * FROM "r53_rlogs" LIMIT 10
```

Esempio di partizionamento

L'esempio seguente mostra una dichiarazione `CREATE TABLE` per i log delle query di Resolver che utilizza la proiezione delle partizioni ed è partizionata per vpc e per data. Per maggiori informazioni sulla proiezione delle partizioni, consulta Proiezione delle partizioni con Amazon Athena.

```
CREATE EXTERNAL TABLE r53_rlogs (
  version string,
  account_id string,
  region string,
  vpc_id string,
  query_timestamp string,
  query_name string,
  query_type string,
  query_class string,
  rcode string,
  answers array<
    struct<
      Rdata: string,
      Type: string,
      Class: string>
    >,
  srcaddr string,
  srcport int,
  transport string,
  srcids struct<
    instance: string,
    resolver_endpoint: string
    >,
  firewall_rule_action string,
  firewall_rule_group_id string,
  firewall_domain_list_id string
)
PARTITIONED BY (
`date` string,
`vpc` string
)
ROW FORMAT SERDE      'org.openx.data.jsonserde.JsonSerDe'
STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT          'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION              's3://DOC-EXAMPLE-BUCKET/route53-query-logging/
AWSLogs/aws_account_id/vpcdnsquerylogs/'
TBLPROPERTIES(
```

```
'projection.enabled' = 'true',
'projection.vpc.type' = 'enum',
'projection.vpc.values' = 'vpc-6446ae02',
'projection.date.type' = 'date',
'projection.date.range' = '2023/06/26,NOW',
'projection.date.format' = 'yyyy/MM/dd',
'projection.date.interval' = '1',
'projection.date.interval.unit' = 'DAYS',
'storage.location.template' = 's3://DOC-EXAMPLE-BUCKET/route53-query-logging/
AWSLogs/aws_account_id/vpcdnsquerylogs/${vpc}/${date}/'
)
```

Query di esempio

Gli esempi seguenti mostrano alcune query che è possibile eseguire da Athena nei log delle query di Resolver.

Esempio 1: log di query in ordine query_timestamp decrescente

La query seguente visualizza i risultati del log in ordine `query_timestamp` decrescente.

```
SELECT * FROM "r53_rlogs"
ORDER BY query_timestamp DESC
```

Esempio 2: log di query all'interno dell'ora di inizio e di fine specificata

Le query riportate di seguito registrano log tra mezzanotte e le 8 del 24 settembre 2020. Sostituisci l'ora di inizio e di fine secondo le tue esigenze.

```
SELECT query_timestamp, srcids.instance, srcaddr, srcport, query_name, rcode
FROM "r53_rlogs"
WHERE (parse_datetime(query_timestamp,'yyyy-MM-dd''T''HH:mm:ss''Z')
     BETWEEN parse_datetime('2020-09-24-00:00:00','yyyy-MM-dd-HH:mm:ss')
     AND parse_datetime('2020-09-24-00:08:00','yyyy-MM-dd-HH:mm:ss'))
ORDER BY query_timestamp DESC
```

Esempio 3: log di query basati su un modello di nome query DNS specificato

La query seguente seleziona i registri il cui nome della query include la stringa "example.com".

```
SELECT query_timestamp, srcids.instance, srcaddr, srcport, query_name, rcode, answers
```

```
FROM "r53_rlogs"
WHERE query_name LIKE '%example.com%'
ORDER BY query_timestamp DESC
```

Esempio 4: richieste di log di query senza risposta

La query seguente seleziona le voci di log in cui la richiesta non ha ricevuto risposta.

```
SELECT query_timestamp, srcids.instance, srcaddr, srcport, query_name, rcode, answers
FROM "r53_rlogs"
WHERE cardinality(answers) = 0
```

Esempio 5: log di query con una risposta specifica

La query seguente mostra i log in cui il valore `answer.Rdata` ha l'indirizzo IP specificato.

```
SELECT query_timestamp, srcids.instance, srcaddr, srcport, query_name, rcode,
 answer.Rdata
FROM "r53_rlogs"
CROSS JOIN UNNEST(r53_rlogs.answers) as st(answer)
WHERE answer.Rdata='203.0.113.16';
```

Esecuzione di query sui registri eventi di Amazon SES

Puoi utilizzare Amazon Athena per eseguire query sui registri eventi di Servizio di e-mail semplice Amazon (Amazon SES).

Amazon SES è una piattaforma e-mail che offre un metodo comodo e conveniente per inviare e ricevere e-mail usando domini e indirizzi e-mail personali. Puoi monitorare le attività di invio di Amazon SES a un livello granulare utilizzando eventi, parametri e statistiche.

In base alle caratteristiche che definisci, puoi pubblicare eventi Amazon SES su Amazon CloudWatch, Amazon Data Firehose o Amazon Simple Notification Service. Dopo aver archiviato le informazioni in Amazon S3, è possibile eseguire query su Amazon Athena.

Per un esempio di `CREATE TABLE` istruzione Athena per i log di Amazon SES, che include i passaggi su come creare viste e appiattire gli array annidati nei dati dei log degli eventi di Amazon SES, consulta «Fase 3: Utilizzo di Amazon Athena per interrogare i log degli eventi SES» nel post del AWS blog Analyzing Amazon SES Event Data with Analytics Services. AWS

Esecuzione di query sui log di flusso Amazon VPC

I flussi di log Amazon Virtual Private Cloud acquisiscono informazioni sul traffico IP da e verso le interfacce di rete in un VPC. Utilizza i log per analizzare i modelli di traffico di rete e identificare le minacce e i rischi nella rete VPC.

Per eseguire query nel flusso di log di Amazon VPC, sono disponibili due opzioni:

- Console Amazon VPC: utilizza la funzionalità di integrazione Athena nella console Amazon VPC per generare un modello AWS CloudFormation che crei un database Athena, un gruppo di lavoro e una tabella di log di flusso con partizionamento per te. Il modello crea inoltre un set di query di flusso di log predefinite che può essere utilizzato per ottenere informazioni dettagliate sul traffico in transito attraverso il VPC.

  Per ulteriori informazioni su questo approccio, consulta Eseguire una query dei flussi di log tramite Amazon Athena nella Guida per l'utente di Amazon VPC.
- Console Amazon Athena: crea le tabelle e le query direttamente nella console Athena. Per maggiori informazioni, continua a leggere questa pagina.

Creazione ed esecuzione di query sulle tabelle per log di flusso VPC personalizzati

Prima di iniziare a eseguire query sui log in Athena, abilita i log di flusso VPC e configurali in modo che possano essere salvati nel bucket Amazon S3. Dopo aver creato i log, lasciali in esecuzione per qualche minuto per raccogliere alcuni dati. I log vengono creati in un formato di compressione GZIP su cui Athena consente di eseguire query direttamente.

Durante la creazione di un log di flusso, puoi utilizzare un formato personalizzato quando vuoi specificare quali campi restituire nel log di flusso e l'ordine in cui visualizzarli. Per ulteriori informazioni sui record dei log di flusso, consulta Record log di flusso nella Guida per l'utente di Amazon VPC.

Considerazioni generali

Quando si creano tabelle nei flussi di log di Athena per Amazon VPC, tenere in considerazione i seguenti punti:

- Per impostazione predefinita, in Athena, Parquet accederà alle colonne in base al nome. Per ulteriori informazioni, consulta Gestione degli aggiornamenti degli schemi.

- Utilizzare i nomi nei record del flusso di log per i nomi delle colonne in Athena. I nomi delle colonne nello schema Athena devono corrispondere esattamente ai nomi dei campi nel flusso di log Amazon VPC, con le seguenti differenze:
  - Sostituire i trattini nei nomi dei campi di log di Amazon VPC con i caratteri di sottolineatura nei nomi delle colonne Athena. In Athena, gli unici caratteri consentiti per i nomi di database, tabelle e colonne sono lettere minuscole, numeri e il carattere di sottolineatura. Per ulteriori informazioni, consulta Nomi database, tabella e colonne.
  - Escludere i nomi dei registri del flusso di log che rappresentano in Athena parole chiave riservate, racchiudendoli tra apici retroversi (`).
- I log di flusso VPC sono specifici. Account AWS Quando pubblichi i tuoi file di log su Amazon S3, il percorso creato da Amazon VPC in Amazon S3 include l'ID dell' Account AWS che è stato utilizzato per crearli. Per ulteriori informazioni, consulta la sezione relativa alla pubblicazione di registri di flusso in Amazon S3 nella Guida per l'utente di Amazon VPC.

Istruzione CREATE TABLE (CREA TABELLA) per il flusso di log di Amazon VPC

La procedura seguente consente di creare una tabella Amazon VPC per il flusso di log di Amazon VPC. Quando crei un log di flusso con un formato personalizzato, crea una tabella con campi che corrispondano a quelli specificati durante la creazione del log di flusso, nello stesso ordine in cui li hai specificati.

Per creare una tabella Athena per il flusso di log di Amazon VPC

1. Inserire un'istruzione DDL come la seguente nell'editor di query della console Athena, seguendo le linee guida riportate nella sezione Considerazioni generali. L'istruzione di esempio crea una tabella con le colonne per il flusso di log di Amazon VPC, versioni da 2 a 5, come documentato in Registri del flusso di log. Se si utilizza un set di colonne o un ordine di colonne diverso, modificare questa istruzione di conseguenza.

```
CREATE EXTERNAL TABLE IF NOT EXISTS `vpc_flow_logs` (
  version int,
  account_id string,
  interface_id string,
  srcaddr string,
  dstaddr string,
  srcport int,
  dstport int,
  protocol bigint,
```

```
  packets bigint,
  bytes bigint,
  start bigint,
  `end` bigint,
  action string,
  log_status string,
  vpc_id string,
  subnet_id string,
  instance_id string,
  tcp_flags int,
  type string,
  pkt_srcaddr string,
  pkt_dstaddr string,
  region string,
  az_id string,
  sublocation_type string,
  sublocation_id string,
  pkt_src_aws_service string,
  pkt_dst_aws_service string,
  flow_direction string,
  traffic_path int
)
PARTITIONED BY (`date` date)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ' '
LOCATION 's3://DOC-EXAMPLE-BUCKET/prefix/AWSLogs/{account_id}/
vpcflowlogs/{region_code}/'
TBLPROPERTIES ("skip.header.line.count"="1");
```

Notare i seguenti punti:

- La query specifica `ROW FORMAT DELIMITED` e omette di specificare un. SerDe Ciò significa che la query utilizza il metodo LazySimpleSerDe per CSV, TSV e file delimitati in modo personalizzato. In questa query, i campi terminano con uno spazio.
- La clausola `PARTITIONED BY` utilizza il tipo `date`. In questo modo è possibile utilizzare operatori matematici nelle query per selezionare ciò che è più vecchio o più recente rispetto a una determinata data.

> **Note**
>
> Poiché `date` è una parola chiave riservata nelle istruzioni DDL, questa viene preceduta da caratteri di apice inverso. Per ulteriori informazioni, consulta Parole chiave riservate.

- Per un log di flusso VPC con un formato personalizzato diverso, modifica i campi in modo che corrispondano a quelli specificati durante la creazione del log di flusso.

2. Modificare `LOCATION 's3://DOC-EXAMPLE-BUCKET/`*`prefix`*`/AWSLogs/`*`{account_id}`*`/vpcflowlogs/`*`{region_code}`*`/'` perché punti al bucket Amazon S3 che contiene i dati di log.
3. Eseguire la query nella console Athena. Una volta completata la query, Athena registra la tabella `vpc_flow_logs`, rendendo i dati in essa contenuti pronti per l'esecuzione di query.
4. Creare partizioni per poter leggere i dati, come nella seguente query di esempio. Questa query crea una singola partizione per una data specificata. Sostituire i segnaposto per data e posizione in base alle esigenze.

> **Note**
>
> Questa query crea unicamente una singola partizione, per la data specificata. Per automatizzare il processo, utilizzare uno script che esegue questa query e crea partizioni in questo modo per `year/month/day` oppure utilizzare un'istruzione `CREATE TABLE` che specifica la proiezione delle partizioni.

```
ALTER TABLE vpc_flow_logs
ADD PARTITION (`date`='YYYY-MM-dd')
LOCATION 's3://DOC-EXAMPLE-BUCKET/prefix/AWSLogs/{account_id}/
vpcflowlogs/{region_code}/YYYY/MM/dd';
```

Query di esempio per la tabella vpc_flow_logs

Utilizzare l'editor di query nella console Athena per eseguire istruzioni SQL sulla tabella creata. È possibile salvare le query, visualizzare le query precedenti o scaricare i risultati delle query in formato CSV. Negli esempi seguenti, sostituire `vpc_flow_logs` con il nome della tabella. Modificare i valori delle colonne e altre variabili in base alle proprie esigenze.

La query di esempio seguente elenca un massimo di 100 log di flusso per la data specificata.

```
SELECT *
FROM vpc_flow_logs
WHERE date = DATE('2020-05-04')
LIMIT 100;
```

La query seguente elenca tutte le connessioni TCP rifiutate e utilizza la colonna di partizione della data appena creata, `date`, per estrarne il giorno della settimana in cui si verificano questi eventi.

```
SELECT day_of_week(date) AS
  day,
  date,
  interface_id,
  srcaddr,
  action,
  protocol
FROM vpc_flow_logs
WHERE action = 'REJECT' AND protocol = 6
LIMIT 100;
```

Per vedere quale server riceve il numero più elevato di richieste HTTPS, utilizzare la seguente query. Conta il numero di pacchetti ricevuti sulla porta HTTPS 443, li raggruppa in base all'indirizzo IP di destinazione e restituisce i primi 10.

```
SELECT SUM(packets) AS
  packetcount,
  dstaddr
FROM vpc_flow_logs
WHERE dstport = 443 AND date > current_date - interval '7' day
GROUP BY dstaddr
ORDER BY packetcount DESC
LIMIT 10;
```

Creazione di tabelle per il flusso di log in formato Apache Parquet

La procedura seguente consente di creare una tabella Amazon VPC per il flusso di log di Amazon VPC in formato Apache Parquet.

Per creare una tabella Athena per il flusso di log di Amazon VPC in formato Parquet

1. Inserire un'istruzione DDL come la seguente nell'editor di query della console Athena, seguendo le linee guida riportate nella sezione Considerazioni generali. L'istruzione di esempio crea una tabella con le colonne per il flusso di log di Amazon VPC, versioni da 2 a 5, come documentato in Registri del flusso di log in formato Parquet, partizionato con Hive ogni ora. Se non disponi di partizioni orarie, rimuovi `hour` dalla clausola `PARTITIONED BY`.

```
CREATE EXTERNAL TABLE IF NOT EXISTS vpc_flow_logs_parquet (
  version int,
  account_id string,
  interface_id string,
  srcaddr string,
  dstaddr string,
  srcport int,
  dstport int,
  protocol bigint,
  packets bigint,
  bytes bigint,
  start bigint,
  `end` bigint,
  action string,
  log_status string,
  vpc_id string,
  subnet_id string,
  instance_id string,
  tcp_flags int,
  type string,
  pkt_srcaddr string,
  pkt_dstaddr string,
  region string,
  az_id string,
  sublocation_type string,
  sublocation_id string,
  pkt_src_aws_service string,
  pkt_dst_aws_service string,
  flow_direction string,
  traffic_path int
)
PARTITIONED BY (
  `aws-account-id` string,
  `aws-service` string,
  `aws-region` string,
```

```
  `year` string,
  `month` string,
  `day` string,
  `hour` string
)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET/prefix/AWSLogs/'
TBLPROPERTIES (
  'EXTERNAL'='true',
  'skip.header.line.count'='1'
  )
```

2. Modificare `LOCATION 's3://DOC-EXAMPLE-BUCKET/prefix/AWSLogs/'` perché punti al bucket Amazon S3 che contiene i dati di log.
3. Eseguire la query nella console Athena.
4. Se i dati sono in formato compatibile con Hive, esegui il seguente comando nella console Athena per aggiornare e caricare le partizioni Hive nel metastore. Una volta completata la query, è possibile eseguire query sui dati nella tabella `vpc_flow_logs_parquet`.

   ```
   MSCK REPAIR TABLE vpc_flow_logs_parquet
   ```

   Se non utilizzi dati compatibili con Hive, esegui ALTER TABLE ADD PARTITION per caricare le partizioni.

Per ulteriori informazioni sull'utilizzo di Athena per eseguire query sul log di flusso di Amazon VPC in formato Parquet, consulta il post Optimize performance and reduce costs for network analytics with VPC Flow Logs in Apache Parquet format nell'AWS Big Data Blog.

Creazione ed esecuzione di query su una tabella per il flusso di log di Amazon VPC tramite la proiezione delle partizioni

Utilizzare una istruzione `CREATE TABLE` come la seguente per creare una tabella, partizionarla e popolare automaticamente le partizioni utilizzando la proiezione delle partizioni. Sostituire il nome

della tabella `test_table_vpclogs` nell'esempio con il nome della propria tabella. Modificare la clausola `LOCATION` per specificare il bucket Amazon S3 che contiene i dati di log di Amazon VPC.

La seguente istruzione `CREATE TABLE` è per il flusso di log VPC fornito in formato di partizionamento in stile non Hive. L'esempio consente l'aggregazione di più account. Se stai centralizzando i log di flusso VPC da più account in un bucket Amazon S3, l'ID dell'account deve essere inserito nel percorso Amazon S3.

```
CREATE EXTERNAL TABLE IF NOT EXISTS test_table_vpclogs (
  version int,
  account_id string,
  interface_id string,
  srcaddr string,
  dstaddr string,
  srcport int,
  dstport int,
  protocol bigint,
  packets bigint,
  bytes bigint,
  start bigint,
  `end` bigint,
  action string,
  log_status string,
  vpc_id string,
  subnet_id string,
  instance_id string,
  tcp_flags int,
  type string,
  pkt_srcaddr string,
  pkt_dstaddr string,
  az_id string,
  sublocation_type string,
  sublocation_id string,
  pkt_src_aws_service string,
  pkt_dst_aws_service string,
  flow_direction string,
  traffic_path int
)
PARTITIONED BY (accid string, region string, day string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ' '
LOCATION '$LOCATION_OF_LOGS'
TBLPROPERTIES
```

```
(
"skip.header.line.count"="1",
"projection.enabled" = "true",
"projection.accid.type" = "enum",
"projection.accid.values" = "$ACCID_1,$ACCID_2",
"projection.region.type" = "enum",
"projection.region.values" = "$REGION_1,$REGION_2,$REGION_3",
"projection.day.type" = "date",
"projection.day.range" = "$START_RANGE,NOW",
"projection.day.format" = "yyyy/MM/dd",
"storage.location.template" = "s3://DOC-EXAMPLE-BUCKET/AWSLogs/${accid}/vpcflowlogs/
${region}/${day}"
)
```

Query di esempio per test_table_vpclogs

Nelle seguenti query di esempio viene interrogata la `test_table_vpclogs` creata dall'istruzione `CREATE TABLE` precedente. Sostituire `test_table_vpclogs` nelle query con il nome della propria tabella. Modificare i valori delle colonne e altre variabili in base alle proprie esigenze.

Per restituire le prime 100 voci di log di accesso in ordine cronologico per un determinato periodo di tempo, eseguire una query come la seguente.

```
SELECT *
FROM test_table_vpclogs
WHERE day >= '2021/02/01' AND day < '2021/02/28'
ORDER BY day ASC
LIMIT 100
```

Per visualizzare quale server riceve i dieci pacchetti HTTP principali per un determinato periodo di tempo, eseguire una query come la seguente. La query conta il numero di pacchetti ricevuti sulla porta HTTPS 443, li raggruppa in base all'indirizzo IP di destinazione e restituisce le prime 10 voci principali dalla settimana precedente.

```
SELECT SUM(packets) AS packetcount,
       dstaddr
FROM test_table_vpclogs
WHERE dstport = 443
  AND day >= '2021/03/01'
  AND day < '2021/03/31'
GROUP BY dstaddr
ORDER BY packetcount DESC
```

```
LIMIT 10
```

Per restituire i registri creati durante un determinato periodo di tempo, eseguire una query come la seguente.

```
SELECT interface_id,
       srcaddr,
       action,
       protocol,
       to_iso8601(from_unixtime(start)) AS start_time,
       to_iso8601(from_unixtime("end")) AS end_time
FROM test_table_vpclogs
WHERE DAY >= '2021/04/01'
  AND DAY < '2021/04/30'
```

Per restituire i registri di accesso per un indirizzo IP di origine tra i periodi di tempo specificati, eseguire una query come la seguente.

```
SELECT *
FROM test_table_vpclogs
WHERE srcaddr = '10.117.1.22'
  AND day >= '2021/02/01'
  AND day < '2021/02/28'
```

Per elencare le connessioni TCP rifiutate, eseguire una query come la seguente.

```
SELECT day,
       interface_id,
       srcaddr,
       action,
       protocol
FROM test_table_vpclogs
WHERE action = 'REJECT' AND protocol = 6 AND day >= '2021/02/01' AND day < '2021/02/28'
LIMIT 10
```

Per restituire i registri di accesso per l'intervallo di indirizzi IP che inizia con `10.`117, eseguire una query come la seguente.

```
SELECT *
FROM test_table_vpclogs
```

```
WHERE split_part(srcaddr,'.', 1)='10'
  AND split_part(srcaddr,'.', 2) ='117'
```

Per restituire i registri di accesso per un indirizzo IP di destinazione tra i periodi di tempo specificati, eseguire una query come la seguente.

```
SELECT *
FROM test_table_vpclogs
WHERE dstaddr = '10.0.1.14'
  AND day >= '2021/01/01'
  AND day < '2021/01/31'
```

Creazione di tabelle per il flusso di log in formato Apache Parquet usando la proiezione di partizione

La seguente istruzione CREATE TABLE di proiezione della partizione per i log di flusso VPC è in formato Apache Parquet, non è compatibile con Hive e partizionata per ora e per data anziché per giorno. Sostituire il nome della tabella `test_table_vpclogs_parquet` nell'esempio con il nome della propria tabella. Modificare la clausola `LOCATION` per specificare il bucket Amazon S3 che contiene i dati di log di Amazon VPC.

```
CREATE EXTERNAL TABLE IF NOT EXISTS test_table_vpclogs_parquet (
  version int,
  account_id string,
  interface_id string,
  srcaddr string,
  dstaddr string,
  srcport int,
  dstport int,
  protocol bigint,
  packets bigint,
  bytes bigint,
  start bigint,
  `end` bigint,
  action string,
  log_status string,
  vpc_id string,
  subnet_id string,
  instance_id string,
  tcp_flags int,
  type string,
  pkt_srcaddr string,
  pkt_dstaddr string,
```

```
  az_id string,
  sublocation_type string,
  sublocation_id string,
  pkt_src_aws_service string,
  pkt_dst_aws_service string,
  flow_direction string,
  traffic_path int
)
PARTITIONED BY (region string, date string, hour string)
ROW FORMAT SERDE
'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT
'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
OUTPUTFORMAT
'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION 's3://DOC-EXAMPLE-BUCKET/prefix/AWSLogs/{account_id}/vpcflowlogs/'
TBLPROPERTIES (
"EXTERNAL"="true",
"skip.header.line.count" = "1",
"projection.enabled" = "true",
"projection.region.type" = "enum",
"projection.region.values" = "us-east-1,us-west-2,ap-south-1,eu-west-1",
"projection.date.type" = "date",
"projection.date.range" = "2021/01/01,NOW",
"projection.date.format" = "yyyy/MM/dd",
"projection.hour.type" = "integer",
"projection.hour.range" = "00,23",
"projection.hour.digits" = "2",
"storage.location.template" = "s3://DOC-EXAMPLE-BUCKET/prefix/AWSLogs/${account_id}/
vpcflowlogs/${region}/${date}/${hour}"
)
```

Risorse aggiuntive

Per ulteriori informazioni sull'utilizzo di Athena per analizzare i log di flusso del VPC, consulta i seguenti articoli del blog sui big data AWS :

- Analizza i log di flusso VPC con l'integrazione con Amazon Athena point-and-click
- Analisi dei log di flusso VPC con Amazon Athena e Amazon QuickSight
- Optimize performance and reduce costs for network analytics with VPC Flow Logs in Apache Parquet format (Ottimizzazione delle prestazioni e riduzione dei costi per l'analisi di rete con i log di flusso VPC in formato Apache Parquet)

Interrogazione dei log AWS WAF

AWS WAF è un firewall per applicazioni Web che consente di monitorare e controllare le richieste HTTP e HTTPS che le applicazioni Web protette ricevono dai client. È possibile definire come gestire le richieste Web configurando le regole all'interno di una lista di controllo degli accessi AWS WAF Web (ACL). Si protegge quindi un'applicazione Web associandovi un'ACL Web. Esempi di risorse per applicazioni Web con cui puoi proteggerti AWS WAF includono CloudFront distribuzioni Amazon, API REST di Amazon API Gateway e Application Load Balancers. Per ulteriori informazioni in merito AWS WAF, consulta la guida per gli AWS WAFsviluppatori.AWS WAF

AWS WAF i log includono informazioni sul traffico analizzato dall'ACL Web, ad esempio l'ora in cui è AWS WAF stata ricevuta la richiesta dalla AWS risorsa, informazioni dettagliate sulla richiesta e l'azione relativa alla regola a cui corrisponde ogni richiesta.

È possibile configurare un ACL AWS WAF Web per pubblicare i log su una delle diverse destinazioni, dove è possibile interrogarli e visualizzarli. Per ulteriori informazioni sulla configurazione della registrazione Web ACL e sul contenuto dei AWS WAF log, consulta la sezione Registrazione del traffico ACL AWS WAF Web nella guida per gli sviluppatori.AWS WAF

Per un esempio di come aggregare AWS WAF i log in un repository centrale di data lake e interrogarli con Athena, consulta il post del AWS Big Data Blog Analyzing logs AWS WAF with Service OpenSearch , Amazon Athena e Amazon. QuickSight

Questo argomento fornisce due esempi di istruzioni `CREATE TABLE`: una che usa il partizionamento e una che non lo fa.

> **Note**
>
> Le istruzioni `CREATE TABLE` in questo argomento possono essere utilizzate per i registri AWS WAF v1 e v2. In v1, il campo `webaclid` contiene un ID. In v2, il campo `webaclid` contiene un ARN completo. Le istruzioni `CREATE TABLE` trattano questo contenuto in modo agnostico usando il tipo di dati `string`.

Argomenti

- Creazione di una tabella per i log AWS WAF S3 in Athena utilizzando la proiezione delle partizioni
- Creazione di una tabella per i AWS WAF log senza partizionamento
- Query di esempio per i log AWS WAF

Creazione di una tabella per i log AWS WAF S3 in Athena utilizzando la proiezione delle partizioni

Poiché AWS WAF i log hanno una struttura nota il cui schema di partizione è possibile specificare in anticipo, è possibile ridurre il tempo di esecuzione delle query e automatizzare la gestione delle partizioni utilizzando la funzionalità di proiezione delle partizioni Athena. La proiezione delle partizioni aggiunge automaticamente nuove partizioni man mano che vengono aggiunti nuovi dati. Ciò elimina la necessità di aggiungere manualmente le partizioni utilizzando `ALTER TABLE ADD PARTITION`.

L'`CREATE TABLE`istruzione di esempio seguente utilizza automaticamente la proiezione delle partizioni nei AWS WAF log da una data specificata fino ad oggi per quattro diverse regioni. AWS La clausola `PARTITION BY` in questo esempio esegue la partizione per regione e per data, ma è possibile modificarla in base alle proprie esigenze. Modifica i campi secondo necessità in base all'output del log. Nelle `storage.location.template` clausole `LOCATION` and, sostituisci i segnaposto *`bucket e AccountID`* con valori che identificano la posizione del bucket Amazon S3 dei log. AWS WAF Per `projection.day.range`, sostituire *`2021`*/*`01`*/*`01`* con la data di inizio che si desidera usare. Dopo aver eseguito la query con esito positivo, è possibile eseguire query sulla tabella. Non è necessario eseguire `ALTER TABLE ADD PARTITION` per caricare le partizioni.

```
CREATE EXTERNAL TABLE `waf_logs`(
  `timestamp` bigint,
  `formatversion` int,
  `webaclid` string,
  `terminatingruleid` string,
  `terminatingruletype` string,
  `action` string,
  `terminatingrulematchdetails` array <
                                  struct <
                                      conditiontype: string,
                                      sensitivitylevel: string,
                                      location: string,
                                      matcheddata: array < string >
                                        >
                                   >,
  `httpsourcename` string,
  `httpsourceid` string,
  `rulegrouplist` array <
                      struct <
                          rulegroupid: string,
                          terminatingrule: struct <
                                              ruleid: string,
                                              action: string,
```

```
                                                rulematchdetails: array <
                                                                        struct <
                                                                            conditiontype:
string,

sensitivitylevel: string,
                                                                            location:
string,
                                                                            matcheddata:
array < string >
                                                                               >
                                                                         >
                                                 >,
                                nonterminatingmatchingrules: array <
                                                                    struct <
                                                                        ruleid: string,
                                                                        action: string,
                                                                        overriddenaction:
string,
                                                                        rulematchdetails:
array <

struct <

   conditiontype: string,

   sensitivitylevel: string,

   location: string,

   matcheddata: array < string >

      >
                                                                         >,
                                                                        challengeresponse:
struct <

responsecode: string,

solvetimestamp: string
                                                                                    >,
                                                                        captcharesponse:
struct <
```

```
 responsecode: string,

 solvetimestamp: string
                                                                          >
                                                                    >
                                                                 >,
                          excludedrules: string
                            >
                       >,
`ratebasedrulelist` array <
                         struct <
                             ratebasedruleid: string,
                             limitkey: string,
                             maxrateallowed: int
                               >
                          >,
  `nonterminatingmatchingrules` array <
                                    struct <
                                        ruleid: string,
                                        action: string,
                                        rulematchdetails: array <
                                                             struct <
                                                                 conditiontype: string,
                                                                 sensitivitylevel:
 string,
                                                                 location: string,
                                                                 matcheddata: array <
 string >
                                                                    >
                                                             >,
                                        challengeresponse: struct <
                                                             responsecode: string,
                                                             solvetimestamp: string
                                                              >,
                                        captcharesponse: struct <
                                                             responsecode: string,
                                                             solvetimestamp: string
                                                              >
                                          >
                                     >,
  `requestheadersinserted` array <
                                 struct <
                                     name: string,
```

```
                                    value: string
                                      >
                                   >,
  `responsecodesent` string,
  `httprequest` struct <
                    clientip: string,
                    country: string,
                    headers: array <
                                struct <
                                    name: string,
                                    value: string
                                      >
                                 >,
                    uri: string,
                    args: string,
                    httpversion: string,
                    httpmethod: string,
                    requestid: string
                      >,
  `labels` array <
              struct <
                  name: string
                    >
               >,
  `captcharesponse` struct <
                      responsecode: string,
                      solvetimestamp: string,
                      failureReason: string
                        >,
  `challengeresponse` struct <
                      responsecode: string,
                      solvetimestamp: string,
                      failureReason: string
                      >,
  `ja3Fingerprint` string,
  `oversizefields` string,
  `requestbodysize` int,
  `requestbodysizeinspectedbywaf` int
)
PARTITIONED BY (
`region` string,
`date` string)
ROW FORMAT SERDE
  'org.openx.data.jsonserde.JsonSerDe'
```

```
STORED AS INPUTFORMAT
  'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET/AWSLogs/accountID/WAFLogs/region/DOC-EXAMPLE-WEBACL/'
TBLPROPERTIES(
 'projection.enabled' = 'true',
 'projection.region.type' = 'enum',
 'projection.region.values' = 'us-east-1,us-west-2,eu-central-1,eu-west-1',
 'projection.date.type' = 'date',
 'projection.date.range' = '2021/01/01,NOW',
 'projection.date.format' = 'yyyy/MM/dd',
 'projection.date.interval' = '1',
 'projection.date.interval.unit' = 'DAYS',
 'storage.location.template' = 's3://DOC-EXAMPLE-BUCKET/AWSLogs/accountID/WAFLogs/
${region}/DOC-EXAMPLE-WEBACL/${date}/')
```

Note

Il formato del percorso nella `LOCATION` clausola dell'esempio è standard ma può variare in base alla configurazione implementata. AWS WAF Ad esempio, il percorso dei AWS WAF log di esempio seguente riguarda una CloudFront distribuzione:

```
s3://DOC-EXAMPLE-BUCKET/AWSLogs/12345678910/WAFLogs/cloudfront/
cloudfronyt/2022/08/08/17/55/
```

Se riscontri problemi durante la creazione o l'interrogazione della tabella AWS WAF dei log, conferma la posizione dei dati di registro o del contatto. AWS Support

Per maggiori informazioni sulla proiezione delle partizioni, consulta Proiezione delle partizioni con Amazon Athena.

Creazione di una tabella per i AWS WAF log senza partizionamento

Questa sezione descrive come creare una tabella per i AWS WAF log senza partizionamento o proiezione delle partizioni.

> **Note**
>
> Per motivi di prestazioni e costi, non è consigliabile utilizzare uno schema non partizionato per le query. Per ulteriori informazioni, consulta i 10 migliori consigli per l'ottimizzazione delle prestazioni per Amazon Athena AWS nel blog Big Data.

Per creare la tabella AWS WAF

1. Copiare e incollare la seguente istruzione DDL nella console Athena. Modifica i campi secondo necessità in base all'output del log. Modifica il parametro `LOCATION` per il bucket Amazon S3 in cui sono archiviati i registri.

   Questa query utilizza il metodo OpenX JSON SerDe.

   > **Note**
   >
   > SerDe Si aspetta che ogni documento JSON si trovi su una singola riga di testo senza caratteri di terminazione di riga che separano i campi del record. Se il testo JSON è in un bel formato di stampa, potresti ricevere un messaggio di errore come HIVE_CURSOR_ERROR: Row is not a valid JSON Object o HIVE_CURSOR_ERROR:: Unexpected end-of-input: expected: expected close marker for OBJECT quando tenti di interrogare la tabella dopo averla JsonParseException creata. Per ulteriori informazioni, consulta JSON Data Files nella documentazione di SerDe OpenX su. GitHub

```
CREATE EXTERNAL TABLE `waf_logs`(
  `timestamp` bigint,
  `formatversion` int,
  `webaclid` string,
  `terminatingruleid` string,
  `terminatingruletype` string,
  `action` string,
  `terminatingrulematchdetails` array <
                                  struct <
                                      conditiontype: string,
                                      sensitivitylevel: string,
                                      location: string,
                                      matcheddata: array < string >
```

```
                                        >
                                    >,
 `httpsourcename` string,
 `httpsourceid` string,
 `rulegrouplist` array <
                     struct <
                         rulegroupid: string,
                         terminatingrule: struct <
                                         ruleid: string,
                                         action: string,
                                         rulematchdetails: array <
                                                             struct <
                                                                 conditiontype:
string,

sensitivitylevel: string,
                                                                 location:
string,
                                                                 matcheddata:
array < string >
                                                                    >
                                                              >
                                     >,
                         nonterminatingmatchingrules: array <
                                                         struct <
                                                             ruleid: string,
                                                             action: string,
                                                             overriddenaction:
string,
                                                             rulematchdetails:
array <

 struct <

     conditiontype: string,

     sensitivitylevel: string,

     location: string,

     matcheddata: array < string >

        >
                                                             >,
```

```
                                                                challengeresponse:
struct <

responsecode: string,

solvetimestamp: string
                                                                        >,
                                                                captcharesponse:
struct <

responsecode: string,

solvetimestamp: string
                                                                        >
                                                                    >
                                                            >,
                        excludedrules: string
                          >
                    >,
`ratebasedrulelist` array <
                        struct <
                            ratebasedruleid: string,
                            limitkey: string,
                            maxrateallowed: int
                              >
                         >,
  `nonterminatingmatchingrules` array <
                                    struct <
                                        ruleid: string,
                                        action: string,
                                        rulematchdetails: array <
                                                            struct <
                                                                conditiontype:
string,
                                                                sensitivitylevel:
string,
                                                                location: string,
                                                                matcheddata: array <
string >
                                                                   >
                                                            >,
                                        challengeresponse: struct <
                                                            responsecode: string,
                                                            solvetimestamp: string
```

```
                                                                    >,
                                        captcharesponse: struct <
                                                                    responsecode: string,
                                                                    solvetimestamp: string
                                                                     >
                                             >
                                    >,
`requestheadersinserted` array <
                                struct <
                                    name: string,
                                    value: string
                                      >
                                 >,
`responsecodesent` string,
`httprequest` struct <
                    clientip: string,
                    country: string,
                    headers: array <
                                struct <
                                    name: string,
                                    value: string
                                      >
                                 >,
                    uri: string,
                    args: string,
                    httpversion: string,
                    httpmethod: string,
                    requestid: string
                      >,
`labels` array <
                struct <
                    name: string
                      >
                 >,
`captcharesponse` struct <
                        responsecode: string,
                        solvetimestamp: string,
                        failureReason: string
                          >,
`challengeresponse` struct <
                        responsecode: string,
                        solvetimestamp: string,
                        failureReason: string
                        >,
```

```
  `ja3Fingerprint` string,
  `oversizefields` string,
  `requestbodysize` int,
  `requestbodysizeinspectedbywaf` int
)
ROW FORMAT SERDE 'org.openx.data.jsonserde.JsonSerDe'
STORED AS INPUTFORMAT 'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION 's3://DOC-EXAMPLE-BUCKET/prefix/'
```

2. Esegui l'istruzione `CREATE EXTERNAL TABLE` nell'editor di query della console Athena. Questo registra la tabella `waf_logs` e rende i dati in essa contenuti disponibili per le query da Athena.

Query di esempio per i log AWS WAF

Molte delle seguenti query di esempio utilizzano la tabella di proiezione delle partizioni creata nella sezione precedente di questo documento. Modificare il nome della tabella, i valori delle colonne e altre variabili negli esempi in base alle proprie esigenze. Per migliorare le prestazioni delle query e ridurre i costi, aggiungere la colonna della partizione nella condizione del filtro.

- Count the number of referrers that contain a specified term
- Count all matched IP addresses in the last 10 days that have matched excluded rules
- Group all counted managed rules by the number of times matched
- Group all counted custom rules by number of times matched

Utilizzo di data e ora

- Return the timestamp field in human-readable ISO 8601 format
- Return records from the last 24 hours
- Return records for a specified date range and IP address
- For a specified date range, count the number of IP addresses in five minute intervals
- Count the number of X-Forwarded-For IP in the last 10 days

Utilizzo di richieste e indirizzi bloccati

- Extract the top 100 IP addresses blocked by a specified rule type
- Count the number of times a request from a specified country has been blocked
- Count the number of times a request has been blocked, grouping by specific attributes
- Count the number of times a specific terminating rule ID has been matched
- Retrieve the top 100 IP addresses blocked during a specified date range

Example : Conta il numero di riferimenti contenenti un termine specificato

La query seguente conta il numero di riferimenti che contengono il termine "amazon" per l'intervallo di date specificato.

```
WITH test_dataset AS
  (SELECT header FROM waf_logs
    CROSS JOIN UNNEST(httprequest.headers) AS t(header) WHERE "date" >= '2021/03/01'
    AND "date" < '2021/03/31')
SELECT COUNT(*) referer_count
FROM test_dataset
WHERE LOWER(header.name)='referer' AND header.value LIKE '%amazon%'
```

Example : Conta tutti gli indirizzi IP corrispondenti negli ultimi 10 giorni che hanno soddisfatto le regole escluse

La query seguente conteggia il numero di volte in cui l'indirizzo IP corrisponde alla regola esclusa nel gruppo di regole.

```
WITH test_dataset AS
  (SELECT * FROM waf_logs
    CROSS JOIN UNNEST(rulegrouplist) AS t(allrulegroups))
SELECT
  COUNT(*) AS count,
  "httprequest"."clientip",
  "allrulegroups"."excludedrules",
  "allrulegroups"."ruleGroupId"
FROM test_dataset
WHERE allrulegroups.excludedrules IS NOT NULL AND from_unixtime(timestamp/1000) > now()
 - interval '10' day
```

```
GROUP BY "httprequest"."clientip", "allrulegroups"."ruleGroupId",
 "allrulegroups"."excludedrules"
ORDER BY count DESC
```

Example : Raggruppa tutte le regole gestite conteggiate in base al numero di corrispondenze

Se hai impostato le azioni delle regole del gruppo di regole su Count nella configurazione Web ACL prima del 27 ottobre 2022, AWS WAF hai salvato le sostituzioni nell'ACL Web JSON come. `excludedRules` Ora, l'impostazione JSON per sovrascrivere una regola su Count è nelle impostazioni `ruleAction0verrides`. Per ulteriori informazioni, consulta Sostituzione delle azioni nei gruppi di regole nella Guida per gli sviluppatori di AWS WAF . Per estrarre le regole gestite in modalità Count dalla nuova struttura di log, esegui una query su `nonTerminatingMatchingRules` nella sezione `ruleGroupList` anziché nel campo `excludedRules`, come nell'esempio seguente.

```
SELECT
 count(*) AS count,
 httpsourceid,
 httprequest.clientip,
 t.rulegroupid,
 t.nonTerminatingMatchingRules
FROM "waf_logs"
CROSS JOIN UNNEST(rulegrouplist) AS t(t)
WHERE action <> 'BLOCK' AND cardinality(t.nonTerminatingMatchingRules) > 0
GROUP BY t.nonTerminatingMatchingRules, action, httpsourceid, httprequest.clientip,
 t.rulegroupid
ORDER BY "count" DESC
Limit 50
```

Example : Raggruppa tutte le regole personalizzate contate in base al numero di corrispondenze

La seguente query raggruppa tutte le regole personalizzate contate in base al numero di volte in cui sono state confrontate.

```
SELECT
  count(*) AS count,
         httpsourceid,
         httprequest.clientip,
         t.ruleid,
         t.action
FROM "waf_logs"
CROSS JOIN UNNEST(nonterminatingmatchingrules) AS t(t)
WHERE action <> 'BLOCK' AND cardinality(nonTerminatingMatchingRules) > 0
```

```
GROUP BY t.ruleid, t.action, httpsourceid, httprequest.clientip
ORDER BY "count" DESC
Limit 50
```

Per informazioni sui percorsi dei log per le regole personalizzate e i gruppi di regole gestiti, consulta Controllo e regolazione nella Guida per gli sviluppatori di AWS WAF .

Utilizzo di data e ora

Example : Restituisci il campo timestamp in formato ISO 8601 leggibile dall'uomo

La seguente query utilizza le funzioni `from_unixtime` e `to_iso8601` per restituire il campo `timestamp` in formato ISO 8601 leggibile dalle persone (ad esempio, `2019-12-13T23:40:12.000Z` invece di `1576280412771`). La query restituisce anche il nome dell'origine HTTP, l'ID di origine e la richiesta.

```
SELECT to_iso8601(from_unixtime(timestamp / 1000)) as time_ISO_8601,
       httpsourcename,
       httpsourceid,
       httprequest
FROM waf_logs
LIMIT 10;
```

Example : Restituisci record delle ultime 24 ore

La query seguente utilizza un filtro nella clausola `WHERE` per restituire il nome dell'origine HTTP, l'ID origine HTTP e i campi di richiesta HTTP per i registri delle ultime 24 ore.

```
SELECT to_iso8601(from_unixtime(timestamp/1000)) AS time_ISO_8601,
       httpsourcename,
       httpsourceid,
       httprequest
FROM waf_logs
WHERE from_unixtime(timestamp/1000) > now() - interval '1' day
LIMIT 10;
```

Example : Restituisci i record per un intervallo di date e un indirizzo IP specificati

Nella query seguente sono elencati i registri in un intervallo di date specificato per un indirizzo IP client specificato.

```
SELECT *
```

```
FROM waf_logs
WHERE httprequest.clientip='53.21.198.66' AND "date" >= '2021/03/01' AND "date" <
 '2021/03/31'
```

Example : per un intervallo di date specificato, conta il numero di indirizzi IP in intervalli di cinque minuti

La query seguente conta, per un determinato intervallo di date, il numero di indirizzi IP in intervalli di cinque minuti.

```
WITH test_dataset AS
  (SELECT
     format_datetime(from_unixtime((timestamp/1000) -
 ((minute(from_unixtime(timestamp / 1000))%5) * 60)),'yyyy-MM-dd HH:mm') AS
 five_minutes_ts,
     "httprequest"."clientip"
     FROM waf_logs
     WHERE "date" >= '2021/03/01' AND "date" < '2021/03/31')
SELECT five_minutes_ts,"clientip",count(*) ip_count
FROM test_dataset
GROUP BY five_minutes_ts,"clientip"
```

Example : Conta il numero di IP X-Forwarded-For negli ultimi 10 giorni

La seguente query filtra le intestazioni della richiesta e conta il numero di IP X-Forwarded-For negli ultimi 10 giorni.

```
WITH test_dataset AS
  (SELECT header
   FROM waf_logs
   CROSS JOIN UNNEST (httprequest.headers) AS t(header)
   WHERE from_unixtime("timestamp"/1000) > now() - interval '10' DAY)
SELECT header.value AS ip,
       count(*) AS COUNT
FROM test_dataset
WHERE header.name='X-Forwarded-For'
GROUP BY header.value
ORDER BY COUNT DESC
```

Per ulteriori informazioni sulle funzioni data e ora, consulta Funzioni e operatori di data e ora nella documentazione Trino.

Utilizzo di richieste e indirizzi bloccati

Example : Estrai i primi 100 indirizzi IP bloccati da un tipo di regola specificato

La query seguente estrae e conta i primi 100 indirizzi IP bloccati dalla regola di terminazione `RATE_BASED` durante l'intervallo di date specificato.

```
SELECT COUNT(httpRequest.clientIp) as count,
httpRequest.clientIp
FROM waf_logs
WHERE terminatingruletype='RATE_BASED' AND action='BLOCK' and "date" >= '2021/03/01'
AND "date" < '2021/03/31'
GROUP BY httpRequest.clientIp
ORDER BY count DESC
LIMIT 100
```

Example : Conta il numero di volte in cui una richiesta proveniente da un Paese specificato è stata bloccata

La seguente query conteggia il numero di volte in cui la richiesta è arrivata da un indirizzo IP che appartiene all'Irlanda (IE) ed è stato bloccato dalla regola di terminazione `RATE_BASED`.

```
SELECT
  COUNT(httpRequest.country) as count,
  httpRequest.country
FROM waf_logs
WHERE
  terminatingruletype='RATE_BASED' AND
  httpRequest.country='IE'
GROUP BY httpRequest.country
ORDER BY count
LIMIT 100;
```

Example : Conta il numero di volte in cui una richiesta è stata bloccata, raggruppando per attributi specifici

La seguente query conta il numero di volte in cui la richiesta è stata bloccata, con risultati raggruppati per WebACL RuleId, ClientIP e URI di richiesta HTTP.

```
SELECT
  COUNT(*) AS count,
  webaclid,
```

```
  terminatingruleid,
  httprequest.clientip,
  httprequest.uri
FROM waf_logs
WHERE action='BLOCK'
GROUP BY webaclid, terminatingruleid, httprequest.clientip, httprequest.uri
ORDER BY count DESC
LIMIT 100;
```

Example : Conta il numero di volte in cui è stata trovata una corrispondenza con un determinato ID di regola di terminazione

La seguente query conteggia il numero di volte in cui un determinato ID regola di terminazione è stato corrisposto (`WHERE terminatingruleid='e9dd190d-7a43-4c06-bcea-409613d9506e'`). La query raggrupperà i risultati per WebACL, Operazione, ClientIP e URI della richiesta HTTP.

```
SELECT
  COUNT(*) AS count,
  webaclid,
  action,
  httprequest.clientip,
  httprequest.uri
FROM waf_logs
WHERE terminatingruleid='e9dd190d-7a43-4c06-bcea-409613d9506e'
GROUP BY webaclid, action, httprequest.clientip, httprequest.uri
ORDER BY count DESC
LIMIT 100;
```

Example : Recupera i primi 100 indirizzi IP bloccati durante un intervallo di date specificato

Nella query seguente vengono estratti i primi 100 indirizzi IP bloccati per un intervallo di date specificato. La query elenca anche il numero di volte in cui gli indirizzi IP sono stati bloccati.

```
SELECT "httprequest"."clientip", "count"(*) "ipcount", "httprequest"."country"
FROM waf_logs
WHERE "action" = 'BLOCK' and "date" >= '2021/03/01'
AND "date" < '2021/03/31'
GROUP BY "httprequest"."clientip", "httprequest"."country"
ORDER BY "ipcount" DESC limit 100
```

Per ulteriori informazioni sulle query dei log di Amazon S3, consulta i seguenti argomenti:

- Come posso analizzare i log di accesso del server Amazon S3 utilizzando Athena? nel Portale del sapere AWS
- Esecuzione di query sui log di accesso Amazon S3 per le richieste utilizzando Amazon Athena nella Guida per l'utente di Amazon Simple Storage Service
- Utilizzo di AWS CloudTrail per identificare le richieste Amazon S3 nella Guida per l'utente di Amazon Simple Storage Service

Esecuzione di query sui log del server Web archiviati in Amazon S3

È possibile utilizzare Athena per interrogare i log dei server Web archiviati in Amazon S3. Gli argomenti di questa sezione mostrano come creare tabelle in Athena per eseguire query sui log del server Web in formati diversi.

Argomenti

- Esecuzione di query sui log Apache archiviati in Amazon S3
- Esecuzione di query sui log Internet Information Server (IIS) archiviati in Amazon S3

Esecuzione di query sui log Apache archiviati in Amazon S3

Puoi utilizzarlo Amazon Athena per interrogare i file di log del server HTTP Apache archiviati nel tuo account Amazon S3. Questo argomento illustra come utilizzare schemi di tabelle per eseguire query sui file Log di accesso Apache nel formato di log comune.

I campi nel formato di log comune includono l'indirizzo IP del client, l'ID del client, l'ID utente, la marca temporale della richiesta ricevuta, il testo della richiesta client, il codice di stato del server e la dimensione dell'oggetto restituito al client.

I dati dell'esempio seguente mostrano il formato comune di log per Apache.

```
198.51.100.7 - Li [10/Oct/2019:13:55:36 -0700] "GET /logo.gif HTTP/1.0" 200 232
198.51.100.14 - Jorge [24/Nov/2019:10:49:52 -0700] "GET /index.html HTTP/1.1" 200 2165
198.51.100.22 - Mateo [27/Dec/2019:11:38:12 -0700] "GET /about.html HTTP/1.1" 200 1287
198.51.100.9 - Nikki [11/Jan/2020:11:40:11 -0700] "GET /image.png HTTP/1.1" 404 230
198.51.100.2 - Ana [15/Feb/2019:10:12:22 -0700] "GET /favicon.ico HTTP/1.1" 404 30
198.51.100.13 - Saanvi [14/Mar/2019:11:40:33 -0700] "GET /intro.html HTTP/1.1" 200 1608
198.51.100.11 - Xiulan [22/Apr/2019:10:51:34 -0700] "GET /group/index.html HTTP/1.1"
 200 1344
```

Creazione di una tabella in Athena per i log Apache

Prima di poter eseguire query sui log Apache archiviati in Amazon S3, è necessario creare uno schema di tabella per Athena in modo che possa leggere i dati di log. Per creare una tabella Athena per i log Apache, è possibile utilizzare - Grok SerDe. Per ulteriori informazioni sull'uso di Grok SerDe, consulta Writing grok custom classifiers nella Developer Guide.AWS Glue

Per creare una tabella in Athena per i log del server Web Apache

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Copiare e incollare la seguente istruzione DDL nella console Athena Query Editor. Modificare i valori in `LOCATION 's3://DOC-EXAMPLE-BUCKET/apache-log-folder/'` per puntare ai log Apache in Amazon S3.

```
CREATE EXTERNAL TABLE apache_logs (
  client_ip string,
  client_id string,
  user_id string,
  request_received_time string,
  client_request string,
  server_status string,
  returned_obj_size string
  )
ROW FORMAT SERDE
   'com.amazonaws.glue.serde.GrokSerDe'
WITH SERDEPROPERTIES (
   'input.format'='^%{IPV4:client_ip} %{DATA:client_id} %{USERNAME:user_id}
 %{GREEDYDATA:request_received_time} %{QUOTEDSTRING:client_request}
 %{DATA:server_status} %{DATA: returned_obj_size}$'
   )
STORED AS INPUTFORMAT
   'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
   'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
   's3://DOC-EXAMPLE-BUCKET/apache-log-folder/';
```

3. Eseguire la query nella console Athena per registrare la tabella `apache_logs`. Al termine della query, i registri sono pronti per le query da Athena.

Esempio di query di selezione per i log Apache

Example — Filtrare per errori 404

La query di esempio seguente seleziona l'ora di ricezione della richiesta, il testo della richiesta client e il codice di stato del server dalla tabella `apache_logs`. La clausola `WHERE` filtra per codice di stato HTTP `404` (pagina non trovata).

```
SELECT request_received_time, client_request, server_status
FROM apache_logs
WHERE server_status = '404'
```

L'immagine seguente mostra i risultati della query nell'editor di query Athena.

Results

	request_received_time	client_request	server_status
1	[11/Jan/2020:11:40:11 -0700]	GET /image.png HTTP/1.1	404
2	[15/Feb/2019:10:12:22 -0700]	GET /favicon.ico HTTP/1.1	404

Example — Filtrare per richieste riuscite

La query di esempio seguente seleziona l'ID utente, l'ora di ricezione della richiesta, il testo della richiesta client e il codice di stato del server dalla tabella `apache_logs`. I filtri della clausola `WHERE` per il codice di stato HTTP `200` (riuscito).

```
SELECT user_id, request_received_time, client_request, server_status
FROM apache_logs
WHERE server_status = '200'
```

L'immagine seguente mostra i risultati della query nell'editor di query Athena.

Results

	user_id	request_received_time	client_request	server_status
1	Li	[10/Oct/2019:13:55:36 -0700]	GET /logo.gif HTTP/1.0	200
2	Jorge	[24/Nov/2019:10:49:52 -0700]	GET /index.html HTTP/1.1	200
3	Mateo	[27/Dec/2019:11:38:12 -0700]	GET /about.html HTTP/1.1	200
4	Saanvi	[14/Mar/2019:11:40:33 -0700]	GET /intro.html HTTP/1.1	200
5	Xiulan	[22/Apr/2019:10:51:34 -0700]	GET /group/index.html HTTP/1.1	200

Example - Filtraggio per timestamp

L'esempio seguente cerca i record il cui tempo di ricezione della richiesta è maggiore del timestamp specificato.

```
SELECT * FROM apache_logs WHERE request_received_time > 10/Oct/2023:00:00:00
```

Esecuzione di query sui log Internet Information Server (IIS) archiviati in Amazon S3

Puoi usare Amazon Athena per eseguire query sui log del server Web Microsoft Internet Information Server (IIS) archiviati nell'account Amazon S3. Mentre IIS utilizza una varietà dei formati dei file di log, questo argomento illustra come creare schemi di tabella per eseguire una query sui log con formato W3C extended e IIS da Athena.

Poiché i formati di file di registro W3C esteso e IIS utilizzano delimitatori di caratteri singoli (rispettivamente spazi e virgole) e non hanno valori racchiusi tra virgolette, è possibile utilizzarli per creare tabelle Athena per LazySimpleSerDetali formati.

Formato dei file di log W3C Extended

Il formato dei dati del file di log W3C extended presenta campi separati da spazi. I campi visualizzati nei log W3C extended sono determinati da un amministratore del server Web che sceglie quali campi di log includere. Nell'esempio seguente i dati di log contengono i campi `date, time`, `c-ip`, `s-ip`, `cs-method`, `cs-uri-stem`, `sc-status`, `sc-bytes`, `cs-bytes`, `time-taken` e `cs-version`.

```
2020-01-19 22:48:39 203.0.113.5 198.51.100.2 GET /default.html 200 540 524 157 HTTP/1.0
2020-01-19 22:49:40 203.0.113.10 198.51.100.12 GET /index.html 200 420 324 164 HTTP/1.0
2020-01-19 22:50:12 203.0.113.12 198.51.100.4 GET /image.gif 200 324 320 358 HTTP/1.0
```

```
2020-01-19 22:51:44 203.0.113.15 198.51.100.16 GET /faq.html 200 330 324 288 HTTP/1.0
```

Creazione di una tabella in Athena per i log W3C Extended

Prima di poter eseguire una query sui W3C extended, è necessario creare uno schema di tabella in modo che Athena possa leggere i dati del log.

Per creare una tabella in Athena per i log W3C extended

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Incollare un'istruzione DDL come la seguente nella console Athena, osservando i seguenti punti:

   a. Aggiungere o rimuovere le colonne dell'esempio in modo che corrispondano ai campi dei log per cui si desidera eseguire la query.

   b. I nomi delle colonna nel formato di file di log W3C Extended contengono trattini (-). Tuttavia, in conformità con le convenzioni di denominazione di Athena, l'esempio di istruzione `CREATE TABLE` li sostituisce con trattini bassi (_).

   c. Per specificare uno spazio come separatore, utilizzare `FIELDS TERMINATED BY ' '`.

   d. Modificare i valori in `LOCATION 's3://DOC-EXAMPLE-BUCKET/w3c-log-folder/'` per puntare ai log W3C extended in Amazon S3.

```
CREATE EXTERNAL TABLE `iis_w3c_logs`(
  date_col string,
  time_col string,
  c_ip string,
  s_ip string,
  cs_method string,
  cs_uri_stem string,
  sc_status string,
  sc_bytes string,
  cs_bytes string,
  time_taken string,
  cs_version 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://DOC-EXAMPLE-BUCKET/w3c-log-folder/'
```

3. Eseguire la query nella console Athena per registrare la tabella `iis_w3c_logs`. Al termine della query, i registri sono pronti per le query da Athena.

Esempio di query di selezione dei log W3C Extended

Il seguente esempio di query seleziona la data, l'ora, la destinazione della richiesta e il tempo impiegato dalla tabella `iis_w3c_logs`. La clausola `WHERE` filtra per casi in cui il metodo HTTP è `GET` e il codice di stato HTTP è `200` (riuscito).

```
SELECT date_col, time_col, cs_uri_stem, time_taken
FROM iis_w3c_logs
WHERE cs_method = 'GET' AND sc_status = '200'
```

L'immagine seguente mostra i risultati della query nell'editor di query Athena.

Results

	date_col	time_col	cs_uri_stem	time_taken
1	2020-01-19	22:48:39	/default.html	157
2	2020-01-19	22:49:40	/index.html	164
3	2020-01-19	22:50:12	/image.gif	358
4	2020-01-19	22:51:44	/faq.html	288

Combinazione dei campi data e ora

I campi `date` e `time` delimitati da spazio sono voci separate nei dati di origine del log, ma è possibile combinarle in una marca temporale se lo si desidera. Usa le funzioni concat() e date_parse() in una query SELECT o CREATE TABLE AS SELECT per concatenare e convertire le colonne di data e ora in formato marca temporale. Nell'esempio seguente viene utilizzata una query CTAS per creare una nuova tabella con una colonna `derived_timestamp`.

```
CREATE TABLE iis_w3c_logs_w_timestamp AS
SELECT
  date_parse(concat(date_col,' ', time_col),'%Y-%m-%d %H:%i:%s') as derived_timestamp,
  c_ip,
```

```
  s_ip,
  cs_method,
  cs_uri_stem,
  sc_status,
  sc_bytes,
  cs_bytes,
  time_taken,
  cs_version
FROM iis_w3c_logs
```

Dopo aver creato la tabella, è possibile eseguire query direttamente sulla nuova colonna della marca temporale, come nell'esempio seguente.

```
SELECT derived_timestamp, cs_uri_stem, time_taken
FROM iis_w3c_logs_w_timestamp
WHERE cs_method = 'GET' AND sc_status = '200'
```

L'immagine seguente mostra i risultati della query.

Results

	derived_timestamp	cs_uri_stem	time_taken
1	2020-01-19 22:48:39.000	/default.html	157
2	2020-01-19 22:49:40.000	/index.html	164
3	2020-01-19 22:50:12.000	/image.gif	358
4	2020-01-19 22:51:44.000	/faq.html	288

Formato del file di log IIS

A differenza del formato W3C Extended, il formato del file di log IIS ha un insieme fisso di campi e include una virgola come separatore. LazySimpleSerDe Tratta la virgola come delimitatore e lo spazio dopo la virgola come inizio del campo successivo.

L'esempio seguente mostra dati di esempio nel formato del file di log IIS.

```
203.0.113.15, -, 2020-02-24, 22:48:38, W3SVC2, SERVER5, 198.51.100.4, 254, 501, 488,
 200, 0, GET, /index.htm, -,
203.0.113.4, -, 2020-02-24, 22:48:39, W3SVC2, SERVER6, 198.51.100.6, 147, 411, 388,
 200, 0, GET, /about.html, -,
```

```
203.0.113.11, -, 2020-02-24, 22:48:40, W3SVC2, SERVER7, 198.51.100.18, 170, 531, 468,
 200, 0, GET, /image.png, -,
203.0.113.8, -, 2020-02-24, 22:48:41, W3SVC2, SERVER8, 198.51.100.14, 125, 711, 868,
 200, 0, GET, /intro.htm, -,
```

Creazione di una tabella in Athena per i file di log IIS

Per eseguire le query sui log del formato del file di log IIS in Amazon S3, devi prima creare uno schema di tabella in modo che Athena possa leggere i dati del log.

Per creare una tabella in Athena per i registri del formato del file di log IIS

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Incollare la seguente istruzione DDL nella console Athena, osservando i seguenti punti:
   a. Per specificare il separatore decimale, utilizzare `FIELDS TERMINATED BY ','`.
   b. Modifica i valori in LOCATION 's3://DOC-EXAMPLE-BUCKET/ *iis-log-file-folder*/' in modo che puntino ai tuoi file di log in formato di log IIS in Amazon S3.

```
CREATE EXTERNAL TABLE `iis_format_logs`(
client_ip_address string,
user_name string,
request_date string,
request_time string,
service_and_instance string,
server_name string,
server_ip_address string,
time_taken_millisec string,
client_bytes_sent string,
server_bytes_sent string,
service_status_code string,
windows_status_code string,
request_type string,
target_of_operation string,
script_parameters 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://DOC-EXAMPLE-BUCKET/iis-log-file-folder/'
```

3. Eseguire la query nella console Athena per registrare la tabella `iis_format_logs`. Al termine della query, i registri sono pronti per le query da Athena.

Esempio di query di selezione nel formato di log IIS

Il seguente esempio di query seleziona la data della richiesta, l'ora della richiesta, la destinazione della richiesta e il tempo impiegato in millisecondi dalla tabella `iis_format_logs`. La clausola `WHERE` filtra per casi in cui il tipo di richiesta è `GET` e il codice di stato HTTP è `200` (riuscito). Nella query, si noti che gli spazi iniziali in `' GET'` e `' 200'` sono necessari per far sì che la query abbia esito positivo.

```
SELECT request_date, request_time, target_of_operation, time_taken_millisec
FROM iis_format_logs
WHERE request_type = ' GET' AND service_status_code = ' 200'
```

L'immagine seguente mostra i risultati della query dei dati di esempio.

Results

	request_date	request_time	target_of_operation	time_taken_millisec
1	2020-02-24	22:48:38	/index.htm	254
2	2020-02-24	22:48:39	/about.html	147
3	2020-02-24	22:48:40	/image.png	170
4	2020-02-24	22:48:41	/intro.htm	125

Formato del file di log NCSA

IIS utilizza anche il formato NCSA Logging, che ha un numero fisso di campi in formato testo ASCII separati da spazi. La struttura è simile al formato di log comune utilizzato per i log di accesso Apache. I campi nel formato di log comune NCSA includono l'indirizzo IP del client, l'ID del client (non usato di norma), l'ID utente/dominio, la marca temporale della richiesta ricevuta, il testo della richiesta client, il codice di stato del server e la dimensione dell'oggetto restituito al client.

L'esempio seguente mostra i dati nel formato di log comune NCSA come documentato per IIS.

```
198.51.100.7 - ExampleCorp\Li [10/Oct/2019:13:55:36 -0700] "GET /logo.gif HTTP/1.0" 200
 232
198.51.100.14 - AnyCompany\Jorge [24/Nov/2019:10:49:52 -0700] "GET /index.html
 HTTP/1.1" 200 2165
198.51.100.22 - ExampleCorp\Mateo [27/Dec/2019:11:38:12 -0700] "GET /about.html
 HTTP/1.1" 200 1287
198.51.100.9 - AnyCompany\Nikki [11/Jan/2020:11:40:11 -0700] "GET /image.png HTTP/1.1"
 404 230
198.51.100.2 - ExampleCorp\Ana [15/Feb/2019:10:12:22 -0700] "GET /favicon.ico HTTP/1.1"
 404 30
198.51.100.13 - AnyCompany\Saanvi [14/Mar/2019:11:40:33 -0700] "GET /intro.html
 HTTP/1.1" 200 1608
198.51.100.11 - ExampleCorp\Xiulan [22/Apr/2019:10:51:34 -0700] "GET /group/index.html
 HTTP/1.1" 200 1344
```

Creazione di una tabella in Athena per i log IIS NCSA

Per l'istruzione `CREATE TABLE`, è possibile utilizzare - Grok SerDe e un modello Grok simile a quello per i log dei server Web Apache. A differenza dei log di Apache, il modello Grok utilizza `%{DATA:user_id}` per il terzo campo invece di `%{USERNAME:user_id}` per tenere conto della presenza della barra rovesciata in `domain\user_id`. Per ulteriori informazioni sull'uso di Grok SerDe, consulta Writing grok custom classifiers nella Developer Guide.AWS Glue

Per creare una tabella in Athena per i log del server Web IIS NCSA

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Copiare e incollare la seguente istruzione DDL nella console Athena Query Editor. Modificare i valori in `LOCATION 's3://DOC-EXAMPLE-BUCKET/iis-ncsa-logs/'` per puntare ai log IIS NCSA in Amazon S3.

```
CREATE EXTERNAL TABLE iis_ncsa_logs(
  client_ip string,
  client_id string,
  user_id string,
  request_received_time string,
  client_request string,
  server_status string,
  returned_obj_size string
  )
ROW FORMAT SERDE
   'com.amazonaws.glue.serde.GrokSerDe'
```

```
WITH SERDEPROPERTIES (
   'input.format'='^%{IPV4:client_ip} %{DATA:client_id} %{DATA:user_id}
 %{GREEDYDATA:request_received_time} %{QUOTEDSTRING:client_request}
 %{DATA:server_status} %{DATA: returned_obj_size}$'
   )
STORED AS INPUTFORMAT
   'org.apache.hadoop.mapred.TextInputFormat'
OUTPUTFORMAT
   'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat'
LOCATION
   's3://DOC-EXAMPLE-BUCKET/iis-ncsa-logs/';
```

3. Eseguire la query nella console Athena per registrare la tabella `iis_ncsa_logs`. Al termine della query, i registri sono pronti per le query da Athena.

Esempio di query di selezione per i log IIS NCSA

Example — Filtrare per errori 404

La query di esempio seguente seleziona l'ora di ricezione della richiesta, il testo della richiesta client e il codice di stato del server dalla tabella `iis_ncsa_logs`. La clausola `WHERE` filtra per codice di stato HTTP `404` (pagina non trovata).

```
SELECT request_received_time, client_request, server_status
FROM iis_ncsa_logs
WHERE server_status = '404'
```

L'immagine seguente mostra i risultati della query nell'editor di query Athena.

Results

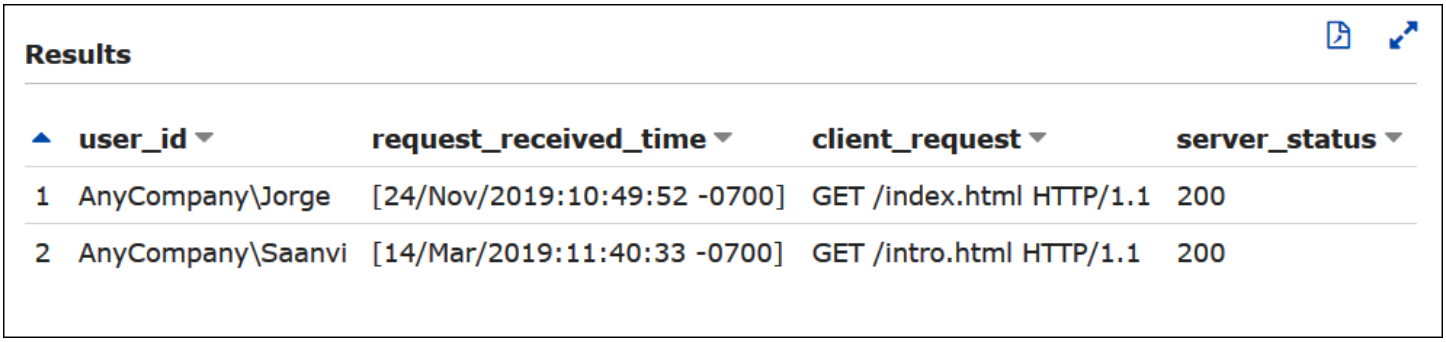
	request_received_time	client_request	server_status
1	[11/Jan/2020:11:40:11 -0700]	GET /image.png HTTP/1.1	404
2	[15/Feb/2019:10:12:22 -0700]	GET /favicon.ico HTTP/1.1	404

Example — Filtrare per richieste riuscite da un determinato dominio

La query di esempio seguente seleziona l'ID utente, l'ora di ricezione della richiesta, il testo della richiesta client e il codice di stato del server dalla tabella `iis_ncsa_logs`. La clausola `WHERE` filtra per richieste con codice di stato HTTP `200` (riuscito) dagli utenti nel dominio `AnyCompany`.

```
SELECT user_id, request_received_time, client_request, server_status
FROM iis_ncsa_logs
WHERE server_status = '200' AND user_id LIKE 'AnyCompany%'
```

L'immagine seguente mostra i risultati della query nell'editor di query Athena.



Results

	user_id	request_received_time	client_request	server_status
1	AnyCompany\Jorge	[24/Nov/2019:10:49:52 -0700]	GET /index.html HTTP/1.1	200
2	AnyCompany\Saanvi	[14/Mar/2019:11:40:33 -0700]	GET /intro.html HTTP/1.1	200

Utilizzo delle transazioni ACID di Athena

Il termine "transazioni ACID" si riferisce a un insieme di proprietà (atomicità, consistenza, isolamento e durabilità) che garantiscono l'integrità dei dati nelle transazioni del database. Le transazioni ACID consentono a più utenti di aggiungere ed eliminare contemporaneamente e in modo affidabile oggetti Amazon S3 in modo atomico, isolando al contempo le query esistenti mantenendo la coerenza di lettura per le query relative al data lake. Le transazioni ACID di Athena aggiungono il supporto di una tabella singola per le operazioni di inserimento, eliminazione, aggiornamento e spostamento temporale al linguaggio DML (Data Manipulation Language) di Athena SQL. Più utenti simultanei possono utilizzare le transazioni ACID di Athena per apportare modifiche affidabili a livello di riga ai dati Amazon S3. Le transazioni di Athena gestiscono automaticamente la semantica e il coordinamento del blocco e non richiedono una soluzione di blocco registri personalizzata.

Le transazioni ACID di Athena e la sintassi SQL familiare semplificano gli aggiornamenti dei dati aziendali e normativi. Ad esempio, per rispondere a una richiesta di cancellazione dei dati, è possibile eseguire un'operazione SQL `DELETE`. Per apportare correzioni manuali ai registri, è possibile utilizzare una singola istruzione `UPDATE`. Per recuperare i dati eliminati di recente, è possibile emettere query temporali tramite un'istruzione `SELECT`.

Poiché sono basate su formati di tabelle condivise, le transazioni ACID di Athena sono compatibili con altri servizi e motori come Amazon EMR e Apache Spark che supportano anche i formati di tabella condivisi.

Le transazioni di Athena sono disponibili tramite la console Athena, le operazioni API e i driver ODBC e JDBC.

Argomenti

- Esecuzione di query sulle tabelle Delta Lake di Linux Foundation
- Utilizzo di Athena per eseguire query sui set di dati Apache Hudi
- Utilizzo di tabelle Apache Iceberg

Esecuzione di query sulle tabelle Delta Lake di Linux Foundation

Delta Lake di Linux Foundation è un formato di tabella per la Big data/analisi. Puoi utilizzare Amazon Athena per leggere direttamente le tabelle Delta Lake archiviate in Amazon S3 senza dover generare file di manifesto o eseguire l'istruzione `MSCK REPAIR`.

Il formato Delta Lake archivia i valori minimi e massimi per colonna di ogni file di dati. L'implementazione di Athena utilizza queste informazioni per consentire di saltare i file sui predicati per non considerare i file indesiderati.

Considerazioni e limitazioni

L'assistenza di Delta Lake ad Athena ha le seguenti considerazioni e limitazioni:

- Solo tabelle con AWS Glue catalogo: il supporto nativo di Delta Lake è supportato solo tramite tabelle registrate con. AWS Glue Se hai una tabella Delta Lake registrata con un altro metastore, puoi comunque conservarla e trattarla come metastore principale. Poiché i metadati Delta Lake sono archiviati nel file system (ad esempio, in Amazon S3) anziché nel metastore, Athena richiede solo la AWS Glue proprietà location in per leggere le tabelle Delta Lake.
- Solo motore V3: le query Delta Lake sono supportate solo sulla versione 3 del motore Athena. È necessario assicurarsi che il gruppo di lavoro creato sia configurato per l'utilizzo della versione 3 del motore Athena.
- Versione del lettore Delta Lake: è supportato il protocollo di lettura Delta Lake fino alla versione 3.

- Mappatura delle colonne e timestampNTZ — Sono supportate la mappatura delle colonne Delta, che consente alle colonne della tabella Delta e alle colonne sottostanti del file Parquet di utilizzare nomi diversi, e i timestamp senza fuso orario (timestampNTZ).
- Supporto per le query temporali non disponibile: non è disponibile alcun supporto per le query che utilizzano le funzionalità temporali di Delta Lake.
- Sola lettura: le istruzioni DML di scrittura come `UPDATE`, `INSERT` o `DELETE` non sono supportate.
- Supporto di Lake Formation: l'integrazione di Lake Formation non è disponibile per tabelle Delta Lake con il relativo schema sincronizzato con AWS Glue. Per ulteriori informazioni, consulta Utilizzo AWS Lake Formation con Amazon Athena e Configurazione delle autorizzazioni per una tabella Delta Lake nella Guida per gli AWS Lake Formation sviluppatori.
- Supporto DDL limitato: sono supportate le seguenti istruzioni DDL: `CREATE EXTERNAL TABLE`, `SHOW COLUMNS`, `SHOW TBLPROPERTIES`, `SHOW PARTITIONS`, `SHOW CREATE TABLE` e `DESCRIBE`. Per informazioni sull'utilizzo dell'istruzione `CREATE EXTERNAL TABLE`, consulta la sezione Nozioni di base.
- Ignorare gli oggetti S3 Glacier non è supportato: se gli oggetti nella tabella Delta Lake di Linux Foundation si trovano in una classe di archiviazione Amazon S3 Glacier, l'impostazione della proprietà della tabella `read_restored_glacier_objects` su `false` non ha alcun effetto.

  Ad esempio, supponiamo di emettere il seguente comando:

  ```
  ALTER TABLE table_name SET TBLPROPERTIES ('read_restored_glacier_objects' = 'false')
  ```

  Per le tabelle Iceberg e Delta Lake, il comando produce l'errore Chiave delle proprietà della tabella non supportata: read_restored_glacier_objects. Per le tabelle Hudi, il comando `ALTER TABLE` non produce un errore, ma gli oggetti Amazon S3 Glacier ancora non verranno ignorati. L'esecuzione delle query `SELECT` dopo il comando `ALTER TABLE` continuerà a restituire tutti gli oggetti.

Tipi di dati supportati per colonne non partizionate

Per le colonne non partizionate, sono supportati tutti i tipi di dati supportati da Athena tranne `CHAR` (`CHAR` non è supportato dal protocollo Delta Lake stesso). I tipi di dati supportati includono:

```
boolean
tinyint
smallint
integer
bigint
```

```
double
float
decimal
varchar
string
binary
date
timestamp
array
map
struct
```

Tipi di dati supportati per colonne partizionate

Per le colonne partizionate, Athena supporta tabelle con i seguenti tipi di dati:

```
boolean
integer
smallint
tinyint
bigint
decimal
float
double
date
timestamp
varchar
```

Per ulteriori informazioni sui tipi di dati in Athena, consulta la pagina Tipi di dati in Amazon Athena.

Nozioni di base

Per essere interrogabile, la tua tabella Delta Lake deve esistere in. AWS Glue Se la tabella è in Amazon S3 ma non in AWS Glue, esegui un'`CREATE EXTERNAL TABLE`istruzione utilizzando la seguente sintassi. Se la tabella esiste già in AWS Glue (ad esempio, perché utilizzi Apache Spark o un altro motore con AWS Glue), puoi saltare questo passaggio.

```
CREATE EXTERNAL TABLE
  [db_name.]table_name
  LOCATION 's3://DOC-EXAMPLE-BUCKET/your-folder/'
  TBLPROPERTIES ('table_type' = 'DELTA')
```

Nota l'omissione delle definizioni delle colonne, delle SerDe librerie e di altre proprietà delle tabelle. A differenza delle tabelle Hive tradizionali, i metadati delle tabelle Delta Lake vengono dedotti dal registro delle transazioni di Delta Lake e sincronizzati direttamente con. AWS Glue

> **Note**
>
> Per le tabelle Delta Lake, non sono ammesse istruzioni `CREATE TABLE` che includono più delle proprietà `LOCATION` e `table_type`.

Lettura di tabelle Delta Lake

Per eseguire query su una tabella Delta Lake, utilizza la sintassi `SELECT` SQL standard:

```
[ WITH with_query [, ...] ]SELECT [ ALL | DISTINCT ] select_expression [, ...]
[ FROM from_item [, ...] ]
[ WHERE condition ]
[ GROUP BY [ ALL | DISTINCT ] grouping_element [, ...] ]
[ HAVING condition ]
[ { UNION | INTERSECT | EXCEPT } [ ALL | DISTINCT ] select ]
[ ORDER BY expression [ ASC | DESC ] [ NULLS FIRST | NULLS LAST] [, ...] ]
[ OFFSET count [ ROW | ROWS ] ]
[ LIMIT [ count | ALL ] ]
```

Per ulteriori informazioni sulla sintassi `SELECT`, consulta la pagina SELECT nella documentazione di Athena.

Il formato Delta Lake archivia i valori minimi e massimi per colonna di ogni file di dati. Athena utilizza queste informazioni per consentire di saltare i file sui predicati per non considerare i file superflui.

Sincronizzazione dei metadati Delta Lake

Athena sincronizza i metadati della tabella, inclusi schema, colonne di partizione e proprietà della tabella, con se AWS Glue usi Athena per creare la tua tabella Delta Lake. Con il passare del tempo, questi metadati possono perdere la sincronizzazione con i metadati della tabella sottostante nel log delle transazioni. Per mantenere aggiornata la tua tabella, puoi scegliere una delle seguenti opzioni:

- Usa il AWS Glue crawler per le tabelle Delta Lake. Per maggiori informazioni, consulta Introduzione al supporto nativo delle tabelle Delta Lake con AWS Glue i crawler nel AWS Big Data Blog e Scheduling an AWS Glue crawler nella Developer Guide. AWS Glue

- Elimina e ricrea la tabella in Athena.
- Usa l'SDK, la CLI AWS Glue o la console per aggiornare manualmente lo schema in. AWS Glue

Tieni presente che le seguenti funzionalità richiedono che AWS Glue lo schema abbia sempre lo stesso schema del log delle transazioni:

- Lake Formation
- Visualizzazioni
- Filtri di righe e colonne

Se il tuo flusso di lavoro non richiede nessuna di queste funzionalità e preferisci non mantenere questa compatibilità, puoi utilizzare `CREATE TABLE` DDL in Athena e aggiungere il percorso Amazon S3 come parametro in. SerDe AWS Glue

Per creare una tabella Delta Lake utilizzando Athena e console AWS Glue

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Nell'editor di query Athena, utilizza il seguente DDL per creare una tabella Delta Lake. Tiene presente che quando utilizzi questo metodo il valore per `TBLPROPERTIES` deve essere `'spark.sql.sources.provider' = 'delta'` e non `'table_type' = 'delta'`.

   Nota, inoltre, che questo stesso schema (con una sola colonna denominata `col` di tipo `array<string>`) viene inserito quando utilizzi Apache Spark (Athena per Apache Spark) o la maggior parte degli altri motori per creare una tabella.

   ```
   CREATE EXTERNAL TABLE
      [db_name.]table_name(col array<string>)
      LOCATION 's3://DOC-EXAMPLE-BUCKET/your-folder/'
      TBLPROPERTIES ('spark.sql.sources.provider' = 'delta')
   ```
3. Apri la AWS Glue console all'indirizzo https://console.aws.amazon.com/glue/.
4. Nel pannello di navigazione, seleziona Catalogo dati, Tabelle.
5. Nell'elenco delle tabelle, seleziona il link per la tabella.
6. Nella pagina della tabella seleziona Azioni, Modifica tabella.
7. Nella sezione Parametri Serde aggiungi la chiave **`path`** con il valore **`s3://DOC-EXAMPLE-BUCKET/your-folder/`**.

8. Seleziona Salva.

Risorse aggiuntive

Per una discussione sull'utilizzo delle tabelle Delta Lake AWS Glue e sulla loro interrogazione con Athena, consulta Gestire le operazioni sui dati UPSERT utilizzando Delta Lake open source AWS Glue e nel Big Data Blog.AWS

Utilizzo di Athena per eseguire query sui set di dati Apache Hudi

Apache Hudi è un framework open source per la gestione dei dati che semplifica l'elaborazione incrementale dei dati. Le operazioni di inserimento, aggiornamento, upsert ed eliminazione a livello di registro vengono elaborate in modo molto più granulare, riducendo il sovraccarico. `Upsert` si riferisce alla possibilità di inserire registri in un set di dati esistente se non esistono già o di aggiornarli.

Hudi gestisce gli eventi di inserimento e aggiornamento dei dati senza creare molti file di piccole dimensioni che possono causare problemi di prestazioni per l'analisi. Apache Hudi tiene traccia automaticamente delle modifiche e unisce i file in modo che abbiano dimensioni ottimali. Ciò evita la necessità di creare soluzioni personalizzate in grado di monitorare e riscrivere molti file di piccole dimensioni in meno file di grandi dimensioni.

I set di dati Hudi sono adatti per i seguenti casi d'uso:

- il rispetto delle normative sulla privacy come il Regolamento generale sulla protezione dei dati (GDPR) e la Legge sulla privacy dei consumatori della California (CCPA), che danno alle persone il diritto di rimuovere le informazioni personali o di modificare il modo in cui vengono utilizzati i loro dati.
- Utilizzo di dati in streaming da sensori e altri dispositivi IoT (Internet of Things) che richiedono specifici eventi di inserimento e aggiornamento dei dati.
- Implementazione di un sistema di acquisizione dei dati di modifica o change data capture (CDC).

I set di dati gestiti da Hudi sono archiviati in Amazon S3 tramite formati di archiviazione aperti. Attualmente, Athena può leggere set di dati Hudi compattati ma non scrivere dati Hudi. Athena supporta fino alla versione 0.8.0 di Hudi con la versione 2 del motore Athena e la versione 0.14.0 di Hudi con la versione 3 del motore Athena. È soggetta a modifiche. Athena non può garantire la compatibilità di lettura con le tabelle create con versioni successive di Hudi. Per ulteriori informazioni

sulle versioni del motore Athena, consulta Controllo delle versioni del motore di Athena. Per ulteriori informazioni sulle funzionalità e sul controllo delle versioni di Hudi, consulta la documentazione di Hudi sul sito Web di Apache.

Tipi di tabella set di dati Hudi

Un set di dati Hudi può essere tra i seguenti tipi:

- Copia in scrittura (CoW): i dati vengono memorizzati in un formato colonnare (Parquet) e ogni aggiornamento crea una nuova versione dei file durante una scrittura.
- Unisci in lettura (MoR) — I dati vengono archiviati utilizzando una combinazione di formati a colonne (Parquet) e basati su righe (Avro). Gli aggiornamenti vengono registrati nei file `delta` basati su righe e vengono compattati in base alle necessità per creare nuove versioni dei file colonnari.

Con i set di dati CoW, ogni volta che c'è un aggiornamento a un record, il file che contiene il record viene riscritto con i valori aggiornati. Quando si lavora con un set di dati MoR, ogniqualvolta è disponibile un aggiornamento Hudi scrive solo la riga per il registro modificato. MoR è più adatto per carichi di lavoro pesanti in scrittura o modifiche con meno letture. CoW è più adatto per carichi di lavoro pesanti di lettura su dati che cambiano meno frequentemente.

Hudi fornisce tre tipi di query per accedere ai dati:

- Query snapshot: query che vedono l'ultima snapshot della tabella a partire da una determinata operazione di commit o compattazione. Per le tabelle MoR, le query snapshot espongono lo stato più recente della tabella unendo i file di base e delta della parte di file più recente al momento della query.
- Query incrementali: le query vedono solo i nuovi dati scritti nella tabella, dal momento di un determinato commit/compattazione. Questo fornisce in modo efficace flussi di modifica per abilitare pipeline di dati incrementali.
- Leggi query ottimizzate: per le tabelle MoR, le query vedono i dati più recenti compattati. Per le tabelle CoW, le query vedono i dati più recenti impegnati.

Nella tabella seguente vengono illustrati i possibili tipi di query Hudi per ciascun tipo di tabella.

Tipo tabella	Possibili tipi di query Hudi
Copia in scrittura	snapshot, incrementale
Unisci in lettura	snapshot, incrementale, lettura ottimizzata

Attualmente, Athena supporta query snapshot e query di lettura ottimizzate, ma non query incrementali. Sulle tabelle MoR, tutti i dati esposti a query ottimizzate di lettura sono compattati. Ciò fornisce buone prestazioni ma non include i commit delta più recenti. Le query snapshot contengono i dati più aggiornati ma incorrono in un sovraccarico computazionale che rende queste query meno performanti.

Per ulteriori informazioni sui compromessi tra i tipi di tabella e query, consulta Tipi di tabella e query nella documentazione di Apache Hudi.

Cambiamento terminologico Hudi: le viste sono ora query

A partire dal rilascio della versione 0.5.1, Apache Hudi ha cambiato parte della sua terminologia. Quelle che erano precedentemente viste sono chiamate query nei rilasci successivi. Nella tabella seguente vengono riepilogate le modifiche tra i termini precedenti e quelli nuovi.

Vecchio termine	Nuovo termine
CoW: visualizz azione ottimizzata per la lettura MoR: visualizz azione in tempo reale	Query snapshot
Visualizzazione incrementale	Query incrementale

Vecchio termine	Nuovo termine
Visualizzazione ottimizzata per la lettura MoR	Query ottimizzata per la lettura

Tabelle dell'operazione Bootstrap

A partire da Apache Hudi v. 0.6.0, la funzione dell'operazione Bootstrap fornisce prestazioni migliori con i set di dati Parquet esistenti. Invece di riscrivere il set di dati, un'operazione Bootstrap può generare solo metadati, lasciando il set di dati in posizione.

Puoi usare Athena per interrogare le tabelle da un'operazione Bootstrap proprio come altre tabelle basate sui dati in Amazon S3. Nell'istruzione `CREATE TABLE`, specifica il percorso della tabella Hudi nella clausola `LOCATION`.

Per ulteriori informazioni sulla creazione di tabelle Hudi utilizzando l'operazione di bootstrap in Amazon EMR, consulta l'articolo Nuove funzionalità di Apache Hudi disponibili in Amazon EMR nel Big Data Blog. AWS

Elenco dei metadati Hudi

Apache Hudi dispone di una tabella di metadati contenente funzionalità di indicizzazione per migliorare le prestazioni, come l'elenco dei file, il salto dei dati utilizzando le statistiche delle colonne e un indice basato sul filtro Bloom.

Di queste funzionalità Athena attualmente supporta solo l'indice di elenco dei file. L'indice di elenco dei file elimina le chiamate al file system come "list files" recuperando le informazioni da un indice che mantiene una mappatura tra partizione e file. Ciò rimuove il bisogno di elencare in modo ricorsivo ogni singola partizione nel percorso della tabella per avere una visualizzazione del file system. Quando utilizzi set di dati di grandi dimensioni, questa indicizzazione riduce drasticamente la latenza che altrimenti si verificherebbe quando si ottiene l'elenco dei file durante la scrittura e le query. Inoltre, evita colli di bottiglia come restrizioni di limiti di richiesta nelle chiamate `LIST` Amazon S3.

> **Note**
>
> Al momento Athena non supporta il salto dei dati o l'indicizzazione del filtro Bloom.

Abilitazione della tabella di metadati Hudi

Per impostazione predefinita, l'elenco dei file basato su tabelle di metadati è disabilitato. Per abilitare la tabella di metadati Hudi e la relativa funzionalità di elenco dei file, imposta la proprietà della tabella `hudi.metadata-listing-enabled` su `TRUE`.

Esempio

L'esempio `ALTER TABLE SET TBLPROPERTIES` seguente abilita la tabella di metadati nella tabella di esempio `partition_cow`.

```
ALTER TABLE partition_cow SET TBLPROPERTIES('hudi.metadata-listing-enabled'='TRUE')
```

Considerazioni e limitazioni

- Athena non supporta query incrementali.
- Athena non supporta CTAS o INSERT INTO per i dati Hudi. Se desideri il supporto Athena per la scrittura di dataset Hudi, invia un feedback ad `<athena-feedback@amazon.com>`.

  Per ulteriori informazioni sulla scrittura di dati Hudi, vedere le seguenti risorse:

  - Utilizzo di un set di dati Hudi nella Guida al rilascio di Amazon EMR.
  - Scrittura di dati nella documentazione di Apache Hudi.
- L'utilizzo di MSCK REPAIR TABLE sulle tabelle Hudi in Athena non è supportato. Se devi caricare una tabella Hudi non creata in, usa. AWS GlueALTER TABLE ADD PARTITION
- Ignorare gli oggetti S3 Glacier non supportati: se gli oggetti nella tabella Apache Hudi si trovano in una classe di archiviazione Amazon S3 Glacier, l'impostazione della proprietà della tabella `read_restored_glacier_objects` su `false` non ha alcun effetto.

  Ad esempio, supponiamo di emettere il seguente comando:

  ```
  ALTER TABLE table_name SET TBLPROPERTIES ('read_restored_glacier_objects' = 'false')
  ```

  Per le tabelle Iceberg e Delta Lake, il comando produce l'errore Chiave delle proprietà della tabella non supportata: read_restored_glacier_objects. Per le tabelle Hudi, il comando `ALTER TABLE` non produce un errore, ma gli oggetti Amazon S3 Glacier ancora non verranno ignorati. L'esecuzione delle query `SELECT` dopo il comando `ALTER TABLE` continuerà a restituire tutti gli oggetti.

Risorse aggiuntive

Per ulteriori risorse sull'utilizzo di Apache Hudi con Athena, consulta le seguenti risorse.

Video

Il video seguente mostra come puoi usare Amazon Athena per interrogare un dataset Apache Hudi ottimizzato per la lettura nel tuo data lake basato su Amazon S3.

Esegui query sui set di dati di Apache Hudi con Amazon Athena.

Post del blog

I seguenti post del blog AWS Big Data includono descrizioni di come utilizzare Apache Hudi con Athena.

- Usa AWS Data Exchange per condividere senza problemi i set di dati Apache Hudi
- Crea un data lake near-real-time transazionale basato su Apache HUDI utilizzando Amazon AWS DMS Kinesis, AWS Glue streaming ETL e visualizzazione dei dati con Amazon QuickSight

Creazione di tabelle Hudi

Questa sezione fornisce esempi di istruzioni CREATE TABLE in Athena per tabelle partizionate e non partizionate di dati Hudi.

Se hai già delle tabelle Hudi create in AWS Glue, puoi interrogarle direttamente in Athena. Quando crei tabelle partizionate Hudi in Athena, è necessario eseguire `ALTER TABLE ADD PARTITION` per caricare i dati Hudi prima di poterli interrogare.

Esempi di creazione di tabelle Copia su scrittura (CoW)

Tabella CoW non partizionata

L'esempio seguente crea una tabella CoW non partizionata in Athena.

```
CREATE EXTERNAL TABLE `non_partition_cow`(
  `_hoodie_commit_time` string,
  `_hoodie_commit_seqno` string,
  `_hoodie_record_key` string,
  `_hoodie_partition_path` string,
  `_hoodie_file_name` string,
```

```
  `event_id` string,
  `event_time` string,
  `event_name` string,
  `event_guests` int,
  `event_type` string)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT
  'org.apache.hudi.hadoop.HoodieParquetInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET/folder/non_partition_cow/'
```

Tabella CoW partizionata

L'esempio seguente crea una tabella CoW partizionata in Athena.

```
CREATE EXTERNAL TABLE `partition_cow`(
  `_hoodie_commit_time` string,
  `_hoodie_commit_seqno` string,
  `_hoodie_record_key` string,
  `_hoodie_partition_path` string,
  `_hoodie_file_name` string,
  `event_id` string,
  `event_time` string,
  `event_name` string,
  `event_guests` int)
PARTITIONED BY (
  `event_type` string)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT
  'org.apache.hudi.hadoop.HoodieParquetInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET/folder/partition_cow/'
```

Il seguente esempio `ALTER TABLE ADD PARTITION` aggiunge due partizioni all'esempio di tabella `partition_cow`.

```
ALTER TABLE partition_cow ADD
```

```
  PARTITION (event_type = 'one') LOCATION 's3://DOC-EXAMPLE-BUCKET/folder/
partition_cow/one/'
  PARTITION (event_type = 'two') LOCATION 's3://DOC-EXAMPLE-BUCKET/folder/
partition_cow/two/'
```

Esempi di creazione di tabelle Unisci in lettura (MoR)

Hudi crea due tabelle nel metastore per MoR: una tabella per le query snapshot e una tabella per le query ottimizzate per la lettura. Entrambe le tabelle sono interrogabili. Nelle versioni Hudi precedenti alla 0.5.1, la tabella per le query ottimizzate per la lettura aveva il nome specificato al momento della creazione della tabella. A partire da Hudi versione 0.5.1, al nome della tabella viene aggiunto il suffisso `_ro` per impostazione predefinita. Il nome della tabella per le query snapshot è il nome specificato seguito da `_rt`.

Tabella Unisci in lettura (MoR) non partizionata

L'esempio seguente crea una tabella MoR non partizionata in Athena per le query ottimizzate per la lettura. Le query ottimizzate di lettura utilizzano il formato di input `HoodieParquetInputFormat`.

```
CREATE EXTERNAL TABLE `nonpartition_mor`(
  `_hoodie_commit_time` string,
  `_hoodie_commit_seqno` string,
  `_hoodie_record_key` string,
  `_hoodie_partition_path` string,
  `_hoodie_file_name` string,
  `event_id` string,
  `event_time` string,
  `event_name` string,
  `event_guests` int,
  `event_type` string)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT
  'org.apache.hudi.hadoop.HoodieParquetInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET/folder/nonpartition_mor/'
```

L'esempio seguente crea una tabella MoR non partizionata in Athena per le query snapshot. Per le query snapshot, utilizzare il formato di input `HoodieParquetRealtimeInputFormat`.

```
CREATE EXTERNAL TABLE `nonpartition_mor_rt`(
  `_hoodie_commit_time` string,
  `_hoodie_commit_seqno` string,
  `_hoodie_record_key` string,
  `_hoodie_partition_path` string,
  `_hoodie_file_name` string,
  `event_id` string,
  `event_time` string,
  `event_name` string,
  `event_guests` int,
  `event_type` string)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT
  'org.apache.hudi.hadoop.realtime.HoodieParquetRealtimeInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET/folder/nonpartition_mor/'
```

Tabella Unisci in lettura (MoR) partizionata

L'esempio seguente crea una tabella MoR partizionata in Athena per le query ottimizzate per la lettura.

```
CREATE EXTERNAL TABLE `partition_mor`(
  `_hoodie_commit_time` string,
  `_hoodie_commit_seqno` string,
  `_hoodie_record_key` string,
  `_hoodie_partition_path` string,
  `_hoodie_file_name` string,
  `event_id` string,
  `event_time` string,
  `event_name` string,
  `event_guests` int)
PARTITIONED BY (
  `event_type` string)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT
  'org.apache.hudi.hadoop.HoodieParquetInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
```

```
LOCATION
  's3://DOC-EXAMPLE-BUCKET/folder/partition_mor/'
```

Il seguente esempio `ALTER TABLE ADD PARTITION` aggiunge due partizioni all'esempio di tabella `partition_mor`.

```
ALTER TABLE partition_mor ADD
  PARTITION (event_type = 'one') LOCATION 's3://DOC-EXAMPLE-BUCKET/folder/
partition_mor/one/'
  PARTITION (event_type = 'two') LOCATION 's3://DOC-EXAMPLE-BUCKET/folder/
partition_mor/two/'
```

L'esempio seguente crea una tabella MoR partizionata in Athena per le query snapshot.

```
CREATE EXTERNAL TABLE `partition_mor_rt`(
  `_hoodie_commit_time` string,
  `_hoodie_commit_seqno` string,
  `_hoodie_record_key` string,
  `_hoodie_partition_path` string,
  `_hoodie_file_name` string,
  `event_id` string,
  `event_time` string,
  `event_name` string,
  `event_guests` int)
PARTITIONED BY (
  `event_type` string)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT
  'org.apache.hudi.hadoop.realtime.HoodieParquetRealtimeInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET/folder/partition_mor/'
```

Analogamente, il seguente esempio `ALTER TABLE ADD PARTITION` aggiunge due partizioni all'esempio di tabella `partition_mor_rt`.

```
ALTER TABLE partition_mor_rt ADD
  PARTITION (event_type = 'one') LOCATION 's3://DOC-EXAMPLE-BUCKET/folder/
partition_mor/one/'
```

```
  PARTITION (event_type = 'two') LOCATION 's3://DOC-EXAMPLE-BUCKET/folder/
partition_mor/two/'
```

Risorse aggiuntive

- Per informazioni sull'utilizzo di connettori AWS Glue personalizzati e processi AWS Glue 2.0 per creare una tabella Apache Hudi su cui è possibile eseguire query con Athena, vedere Scrittura su tabelle Apache Hudi utilizzando un connettore AWS Glue personalizzato nel Big Data Blog. AWS
- Per un articolo sull'utilizzo di Apache Hudi e Amazon Athena per creare un framework di elaborazione dati per un data lake, consulta Semplificare l'elaborazione operativa dei dati nei data lake AWS Glue utilizzando e Apache Hudi nel Big Data Blog. AWS Glue AWS

Utilizzo di tabelle Apache Iceberg

Athena supporta le query di lettura, viaggio nel tempo, scrittura e DDL per le tabelle Apache Iceberg che utilizzano il formato Apache Parquet per i dati e il catalogo per il loro metastore. AWS Glue

Apache Iceberg è un formato a tabella aperta per set di dati analitici di dimensioni molto grandi. Iceberg gestisce grandi raccolte di file come tabelle e supporta le moderne operazioni analitiche di data lake come inserimento a livello di record, aggiornamento, eliminazione e query temporali. La specifica di Iceberg consente un'evoluzione perfetta delle tabelle, come l'evoluzione di schemi e partizioni; inoltre è concepita per un utilizzo ottimizzato su Amazon S3. Iceberg aiuta inoltre a garantire la correttezza dei dati in scenari di scrittura simultanea.

Per ulteriori informazioni su Apache Iceberg, consulta https://iceberg.apache.org/.

Considerazioni e limitazioni

Il supporto di Athena per le tabelle Iceberg presenta le seguenti considerazioni e limitazioni:

- Supporto per la versione Iceberg — Athena supporta la versione 1.4.2 di Apache Iceberg.
- Solo tabelle con AWS Glue catalogo: solo le tabelle Iceberg create in base al AWS Glue catalogo in base alle specifiche definite dall'implementazione open source del catalogo Glue sono supportate da Athena.
- Supporto per il blocco delle tabelle AWS Glue solo da parte di: a differenza dell'implementazione open source del catalogo Glue, che supporta il blocco personalizzato tramite plug-in, Athena supporta AWS Glue solo il blocco ottimistico. L'utilizzo di Athena per modificare una tabella Iceberg

con qualsiasi altra implementazione di blocco causerà potenziali perdite di dati e interruzioni delle transazioni.

- Formati di file supportati: il supporto del formato di file Iceberg in Athena dipende dalla versione del motore Athena, come illustrato nella tabella seguente.

Versione del motore Athena	Parquet	ORC	Avro
2	Sì	No	No
3	Sì	Sì	Sì

- Tabelle Iceberg v2: Athena crea e opera solo su tabelle Iceberg v2. Per la differenza tra le tabelle v1 e v2, consulta Modifiche al tipo di formato nella documentazione di Apache Iceberg.
- Visualizzazione dei tipi di orari senza fuso orario: l'ora e il timestamp senza tipi di fuso orario vengono visualizzati in UTC. Se il fuso orario non è specificato in un'espressione di filtro su una colonna temporale, viene utilizzato UTC.
- Precisione dei dati relativi al timestamp: mentre Iceberg supporta una precisione al microsecondo per il tipo di dati del timestamp, Athena per i timestamp supporta solo una precisione al millisecondo sia in lettura che in scrittura. Per i dati nelle colonne relative al tempo riscritti durante le operazioni di compattazione manuale, Athena mantiene solo una precisione al millisecondo.
- Operazioni non supportate:le seguenti operazioni Athena non sono supportate per le tabelle Iceberg.
  - ALTER TABLE SET LOCATION
- Viste: utilizza `CREATE VIEW` per creare viste Athena come descritto nella pagina Utilizzo delle visualizzazioni. Se desideri utilizzare le specifiche di visualizzazione Iceberg per creare viste, contatta athena-feedback@amazon.com.
- Comandi di gestione TTF non supportati in AWS Lake Formation: sebbene sia possibile utilizzare Lake Formation per gestire le autorizzazioni di accesso in lettura per TransactionTable formati (TTF) come Apache Iceberg, Apache Hudi e Linux Foundation Delta Lake, non è possibile utilizzare Lake Formation per gestire le autorizzazioni per operazioni come `VACUUM` o con questi formati di tabella. `MERGE` `UPDATE` `OPTIMIZE` Per ulteriori informazioni sull'integrazione di Lake Formation con Athena, consulta Using AWS Lake Formation with Amazon Athena nella AWS Lake Formation Developer Guide.

- Partizionamento per campi annidati: il partizionamento per campi annidati non è supportato. Il tentativo di eseguire questa operazione genera il messaggio NOT_SUPPORTED: Il partizionamento per campo annidato non è supportato:*column_name.nested_field_name*.
- Ignorare gli oggetti S3 Glacier non supportati: se gli oggetti nella tabella Apache Iceberg si trovano in una classe di archiviazione Amazon S3 Glacier, l'impostazione della proprietà della tabella `read_restored_glacier_objects` su `false` non ha alcun effetto.

  Ad esempio, supponiamo di emettere il seguente comando:

  ```
  ALTER TABLE table_name SET TBLPROPERTIES ('read_restored_glacier_objects' = 'false')
  ```

  Per le tabelle Iceberg e Delta Lake, il comando produce l'errore Chiave delle proprietà della tabella non supportata: read_restored_glacier_objects. Per le tabelle Hudi, il comando `ALTER TABLE` non produce un errore, ma gli oggetti Amazon S3 Glacier ancora non verranno ignorati. L'esecuzione delle query `SELECT` dopo il comando `ALTER TABLE` continuerà a restituire tutti gli oggetti.

Se si desidera che Athena supporti una particolare funzionalità, inviare un feedback all'indirizzo athena-feedback@amazon.com.

Argomenti

Creazione di tabelle Iceberg

Per creare una tabella Iceberg da usare in Athena, puoi usare `CREATE TABLE` un'istruzione come documentato in questa pagina oppure puoi usare un crawler. AWS Glue

Utilizzo di una dichiarazione CREATE TABLE

Athena crea tabelle Iceberg v2. Per la differenza tra le tabelle v1 e v2, consulta Modifiche al tipo di formato nella documentazione di Apache Iceberg.

Athena `CREATE TABLE` crea una tabella Iceberg senza dati. È possibile eseguire una query su una tabella da sistemi esterni come Apache Spark direttamente se la tabella utilizza il Catalogo glue open source Iceberg. Non è necessario creare una tabella esterna.

> **Warning**
>
> L'esecuzione di `CREATE EXTERNAL TABLE` si traduce nel messaggio di errore External keyword not supported for table type ICEBERG (Parola chiave esterna non supportata per il tipo di tabella ICEBERG).

Per creare una tabella Iceberg da Athena, imposta la proprietà della tabella `'table_type'` su `'ICEBERG'` nella clausola `TBLPROPERTIES`, come nel seguente riepilogo della sintassi.

```
CREATE TABLE
  [db_name.]table_name (col_name data_type [COMMENT col_comment] [, ...] )
  [PARTITIONED BY (col_name | transform, ... )]
  LOCATION 's3://DOC-EXAMPLE-BUCKET/your-folder/'
  TBLPROPERTIES ( 'table_type' ='ICEBERG' [, property_name=property_value] )
```

Per informazioni sui tipi di dati su cui puoi eseguire query nelle tabelle Iceberg, consulta Tipi di dati supportati per tabelle Iceberg in Athena.

Partizionamento

Per creare tabelle Iceberg con partizioni, utilizza la sintassi `PARTITIONED BY`. Le colonne utilizzate per il partizionamento devono essere prima specificate nelle dichiarazioni delle colonne. All'interno della clausola `PARTITIONED BY`, il tipo di colonna non deve essere incluso. È inoltre possibile definire le trasformazioni delle partizioni nella sintassi `CREATE TABLE`. Per specificare più colonne per il partizionamento, separa le colonne con una virgola (`,`), come nell'esempio seguente.

```
CREATE TABLE iceberg_table (id bigint, data string, category string)
  PARTITIONED BY (category, bucket(16, id))
  LOCATION 's3://DOC-EXAMPLE-BUCKET/your-folder/'
```

```
  TBLPROPERTIES ( 'table_type' = 'ICEBERG' )
```

La tabella riportata di seguito mostra le funzioni di trasformazione delle partizioni disponibili.

Funzione	Descrizione	Tipi supportati
`year(ts)`	Partizione per anno	`date`, `timestamp`
`month(ts)`	Partizione per mese	`date`, `timestamp`
`day(ts)`	Partizione per giorno	`date`, `timestamp`
`hour(ts)`	Partizione per ora	`timestamp`
`bucket(N, col)`	Partizione per valore hash mod *N* bucket Questo è lo stesso concetto dell'hash bucketing per le tabelle Hive.	`int`, `long`, `decimal`, `date`, `timestamp` , `string`, `binary`
`truncate( L, col)`	Partizione per valore troncato su *L*	`int`, `long`, `decimal`, `string`

Athena supporta il partizionamento nascosto di Iceberg. Per ulteriori informazioni, consulta Partizionamento nascosto di Iceberg nella documentazione di Apache Iceberg.

Proprietà tabella

Questa sezione descrive le proprietà della tabella che è possibile specificare come coppie chiave-valore nella clausola `TBLPROPERTIES` dell'istruzione `CREATE TABLE`. Athena consente solo un elenco predefinito di coppie chiave-valore nelle proprietà della tabella per creare o modificare le tabelle Iceberg. Le tabelle seguenti mostrano le proprietà della tabella che è possibile specificare in anteprima. Per ulteriori informazioni sulle opzioni di compattazione, consulta Ottimizzazione delle tabelle Iceberg più avanti in questo documento. Se si desidera che Athena supporti una specifica proprietà di configurazione della tabella open source, inviare un feedback all'indirizzo athena-feedback@amazon.com.

format

Descrizione	Formato dei dati del file
Valori delle proprietà consentiti	Le combinazioni di formato di file supportati e compressione variano in base alla versione del motore Athena. Per ulteriori informazioni, consulta Supporto per la compressione delle tabelle Iceberg in base al formato di file.
Valore predefinito	parquet

write_compression

Descrizione	Codec di compressione dei file
Valori delle proprietà consentiti	Le combinazioni di formato di file supportati e compressione variano in base alla versione del motore Athena. Per ulteriori informazioni, consulta Supporto per la compressione delle tabelle Iceberg in base al formato di file.
Valore predefinito	La compressione di scrittura predefinita varia in base alla versione del motore Athena. Per ulteriori informazioni, consulta Supporto per la compressione delle tabelle Iceberg in base al formato di file.

optimize_rewrite_data_file_threshold

Descrizione	Configurazione specifica dell'ottimizzazione dei dati Se ci sono meno file di dati che richiedono un'ottimizzazione rispetto alla soglia specificata, i file non vengono riscritti. Ciò consente l'accumulo di più file di dati per produrre file più vicini alle dimensioni di destinazione e saltare i calcoli non necessari per risparmiare sui costi.
Valori delle proprietà consentiti	Un numero positivo. Deve essere inferiore a 50.
Valore predefinito	5

optimize_rewrite_delete_file_threshold

Descrizione	Configurazione specifica dell'ottimizzazione dei dati Se a un file di dati sono associati meno file di eliminazione rispetto alla soglia, il file di dati non viene riscritto. Ciò consente l'accumulo di più file di eliminazione per ogni file di dati per risparmiare sui costi.
Valori delle proprietà consentiti	Un numero positivo. Deve essere inferiore a 50.
Valore predefinito	2

vacuum_min_snapshots_to_keep

Descrizione	Numero minimo di snapshot da mantenere sul ramo principale di una tabella. Questo valore ha la precedenza sulla proprietà `vacuum_max_snapshot_age_seconds` . Se il numero minimo di snapshot rimanenti è più vecchio dell'età specificata da `vacuum_max_snapshot_age_seconds` , gli snapshot vengono conservati e il valore di `vacuum_max_snapshot_age_seconds` viene ignorato.
Valori delle proprietà consentiti	Un numero positivo.
Valore predefinito	1

vacuum_max_snapshot_age_seconds

Descrizione	Età massima degli snapshot da mantenere nel ramo principale. Questo valore viene ignorato se il numero minimo di snapshot rimanenti specificato da `vacuum_min_snapshots_to_keep` è più vecchio dell'età specificata. Questa proprietà di comportamento della tabella corrisponde alla proprietà nella configurazione di Apache `history.expire.max-snapshot-age-ms` Iceberg.

Valori delle proprietà consentiti	Un numero positivo.
Valore predefinito	432.000 secondi (5 giorni)

vacuum_max_metadata_files_to_keep

Descrizione	Il numero massimo di file di metadati precedenti da conservare nel ramo principale della tabella.
Valori delle proprietà consentiti	Un numero positivo.
Valore predefinito	100

Istruzione CREATE TABLE (CREA TABELLA) di esempio

Nell'esempio seguente viene creata una tabella Iceberg che ha tre colonne.

```
CREATE TABLE iceberg_table (
  id int,
  data string,
  category string)
PARTITIONED BY (category, bucket(16,id))
LOCATION 's3://DOC-EXAMPLE-BUCKET/iceberg-folder'
TBLPROPERTIES (
  'table_type'='ICEBERG',
  'format'='parquet',
  'write_compression'='snappy',
  'optimize_rewrite_delete_file_threshold'='10'
)
```

CREATE TABLE AS SELECT (CTAS)

Per informazioni sulla creazione di una tabella Iceberg utilizzando l'istruzione `CREATE TABLE AS`, consulta CREATE TABLE AS, con particolare attenzione alla sezione Proprietà tabella CTAS.

Utilizzo di un crawler AWS Glue

Puoi usare un AWS Glue crawler per registrare automaticamente le tue tabelle Iceberg in. AWS Glue Data Catalog Se desideri migrare da un altro catalogo Iceberg, puoi creare e pianificare un AWS Glue crawler e fornire i percorsi Amazon S3 in cui si trovano le tabelle Iceberg. Puoi specificare la profondità massima dei percorsi di Amazon S3 che il crawler AWS Glue può attraversare. Dopo aver pianificato un AWS Glue crawler, il crawler estrae le informazioni sullo schema e le aggiorna con le modifiche dello schema ogni volta che viene eseguito. AWS Glue Data Catalog Il AWS Glue crawler supporta la fusione dello schema tra istantanee e aggiorna la posizione più recente del file di metadati in. AWS Glue Data Catalog Per ulteriori informazioni, consulta Data Catalog and crawler in. AWS Glue

Gestione di tabelle Iceberg

Athena supporta le seguenti operazioni DDL della tabella per le tabelle Iceberg.

ALTER TABLE RENAME

Rinominare una tabella

Poiché i metadati della tabella di una tabella Iceberg sono archiviati in Amazon S3, è possibile aggiornare il database e il nome della tabella gestita da Iceberg senza influire sulle informazioni della tabella sottostante.

Riepilogo

```
ALTER TABLE [db_name.]table_name RENAME TO [new_db_name.]new_table_name
```

Esempio

```
ALTER TABLE my_db.my_table RENAME TO my_db2.my_table2
```

ALTER TABLE SET PROPERTIES

Aggiunge proprietà a una tabella Iceberg e imposta i relativi valori assegnati.

In conformità con le specifiche Iceberg, le proprietà della tabella sono archiviate nel file di metadati della tabella Iceberg anziché in AWS Glue. Athena non accetta proprietà personalizzate della tabella. Consulta la sezione Proprietà tabella per le coppie chiave-valore consentite. Se si desidera che Athena supporti una specifica proprietà di configurazione della tabella open source, inviare un feedback all'indirizzo athena-feedback@amazon.com.

Riepilogo

```
ALTER TABLE [db_name.]table_name SET TBLPROPERTIES ('property_name' =
 'property_value' [ , ... ])
```

Esempio

```
ALTER TABLE iceberg_table SET TBLPROPERTIES (
  'format'='parquet',
  'write_compression'='snappy',
  'optimize_rewrite_delete_file_threshold'='10'
)
```

ALTER TABLE UNSET PROPERTIES

Elimina le proprietà esistenti da una tabella Iceberg.

Riepilogo

```
ALTER TABLE [db_name.]table_name UNSET TBLPROPERTIES ('property_name' [ , ... ])
```

Esempio

```
ALTER TABLE iceberg_table UNSET TBLPROPERTIES ('write_compression')
```

DESCRIBE TABLE

Descrive le informazioni della tabella.

Riepilogo

```
DESCRIBE [FORMATTED] [db_name.]table_name
```

Quando l'opzione `FORMATTED` è specificata, l'output visualizza informazioni aggiuntive come la posizione della tabella e le proprietà.

Esempio

```
DESCRIBE iceberg_table
```

DROP TABLE

Elimina una tabella Iceberg.

> **Warning**
>
> Poiché le tabelle Iceberg sono considerate tabelle gestite in Athena, l'eliminazione di una tabella Iceberg rimuove tutti i dati anche dalla tabella.

Riepilogo

```
DROP TABLE [IF EXISTS] [db_name.]table_name
```

Esempio

```
DROP TABLE iceberg_table
```

SHOW CREATE TABLE

Visualizza un'istruzione DDL `CREATE TABLE` che può essere utilizzata per ricreare la tabella Iceberg in Athena. Se Athena non è in grado di riprodurre la struttura della tabella (ad esempio, poiché nella tabella sono specificate proprietà personalizzate), viene generato un errore NON SUPPORTATO.

Riepilogo

```
SHOW CREATE TABLE [db_name.]table_name
```

Esempio

```
SHOW CREATE TABLE iceberg_table
```

SHOW TABLE PROPERTIES (MOSTRA PROPRIETÀ TABELLA)

Mostra una o più proprietà della tabella di una tabella Iceberg. Vengono visualizzate solo le proprietà della tabella supportate da Athena.

Riepilogo

```
SHOW TBLPROPERTIES [db_name.]table_name [('property_name')]
```

Esempio

```
SHOW TBLPROPERTIES iceberg_table
```

Esecuzione di query sui metadati di tabelle Iceberg

In una query SELECT puoi utilizzare le seguenti proprietà dopo *table_name* per interrogare i metadati di tabelle Iceberg:

- $files: mostra i file di dati correnti di una tabella.
- $manifests: mostra i manifesti dei file correnti di una tabella.
- $history: mostra la cronologia di una tabella.
- $partitions: mostra le partizioni correnti di una tabella.
- $snapshots: mostra gli snapshot di una tabella.
- $refs: mostra i riferimenti di una tabella.

Sintassi

La seguente dichiarazione elenca i file di una tabella Iceberg.

```
SELECT * FROM "dbname"."tablename$files"
```

L'istruzione seguente elenca i manifesti di una tabella Iceberg.

```
SELECT * FROM "dbname"."tablename$manifests"
```

La seguente dichiarazione mostra la cronologia di una tabella Iceberg.

```
SELECT * FROM "dbname"."tablename$history"
```

L'esempio seguente mostra una partizione di una tabella Iceberg.

```
SELECT * FROM "dbname"."tablename$partitions"
```

L'esempio seguente elenca le istantanee di una tabella Iceberg.

```
SELECT * FROM "dbname"."tablename$snapshots"
```

L'esempio seguente mostra un riferimento per una tabella Iceberg.

```
SELECT * FROM "dbname"."tablename$refs"
```

Schema della tabella Iceberg in evoluzione

Gli aggiornamenti dello schema Iceberg sono modifiche solo ai metadati. Nessun file di dati viene modificato quando si esegue un aggiornamento dello schema.

Il formato Iceberg supporta le seguenti modifiche all'evoluzione dello schema:

- Add (Aggiungi): aggiunge una nuova colonna a una tabella o a uno `struct` nidificato.
- Drop (Elimina): rimuove una colonna esistente da una tabella o uno `struct` nidificato.
- Rename (Rinomina): rinomina una colonna o un campo esistente in uno `struct` nidificato.
- Riordina: Modifica l'ordine delle colonne.
- Type promotion (Promozione del tipo): amplia il tipo di colonna, il campo `struct`, la chiave `map`, il valore `map` o l'elemento `list`. Attualmente sono supportati i seguenti casi per le tabelle Iceberg:
  - da intero a intero grande
  - da float a double
  - aumento della precisione di un tipo decimale

ALTER TABLE ADD COLUMNS

Aggiunge una o più colonne a una tabella Iceberg esistente.

Riepilogo

```
ALTER TABLE [db_name.]table_name ADD COLUMNS (col_name data_type [,...])
```

Esempi

Nell'esempio seguente viene aggiunta una colonna `comment` di tipo `string` in una tabella Iceberg.

```
ALTER TABLE iceberg_table ADD COLUMNS (comment string)
```

Nell'esempio seguente viene aggiunta una colonna `point` di tipo `struct` in una tabella Iceberg.

```
ALTER TABLE iceberg_table
```

```
ADD COLUMNS (point struct<x: double, y: double>)
```

Nell'esempio seguente viene aggiunta una colonna `points` che è una matrice di strutture a una tabella Iceberg.

```
ALTER TABLE iceberg_table
ADD COLUMNS (points array<struct<x: double, y: double>>)
```

ALTER TABLE ADD COLUMNS

Elimina le proprietà esistenti da una tabella Iceberg.

Riepilogo

```
ALTER TABLE [db_name.]table_name DROP COLUMN col_name
```

Esempio

```
ALTER TABLE iceberg_table DROP COLUMN userid
```

ALTER TABLE CHANGE COLUMN

Modifica il nome, il tipo, l'ordine o il commento di una colonna.

> **Note**
>
> `ALTER TABLE REPLACE COLUMNS` non è supportato. Poiché `REPLACE COLUMNS` rimuove tutte le colonne e ne aggiunge di nuove, non è supportata per Iceberg. `CHANGE COLUMN` è la sintassi preferita per l'evoluzione dello schema.

Riepilogo

```
ALTER TABLE [db_name.]table_name
  CHANGE [COLUMN] col_old_name col_new_name column_type
  [COMMENT col_comment] [FIRST|AFTER column_name]
```

Esempio

```
ALTER TABLE iceberg_table CHANGE comment blog_comment string AFTER id
```

SHOW COLUMNS

Mostra le colonne in una tabella.

Riepilogo

```
SHOW COLUMNS (FROM|IN) [db_name.]table_name
```

Esempio

```
SHOW COLUMNS FROM iceberg_table
```

Interrogazione di tabelle Iceberg ed esecuzione di query temporali

Per eseguire query su un set di dati Iceberg, utilizza una istruzione `SELECT` standard come la seguente. Le query seguono le specifiche del formato Apache Iceberg v2 ed eseguono merge-on-read eliminazioni sia di posizione che di uguaglianza.

```
SELECT * FROM [db_name.]table_name [WHERE predicate]
```

Per ottimizzare i tempi delle query, tutti i predicati vengono "spinti" più vicino a dove si trovano i dati.

Query temporali e di versione

Ogni tabella Apache Iceberg conserva un manifesto con versioni degli oggetti Amazon S3 che comprende. Le versioni precedenti del manifesto possono essere utilizzate per le query temporali e di versione.

Le query temporali in Athena interrogano Amazon S3 relativamente a dati storici da uno snapshot coerente con una data e un'ora specificate. Le query di versione in Athena interrogano Amazon S3 relativamente a dati storici a partire da un ID snapshot specificato.

Query temporali

Per eseguire una query temporale, utilizza `FOR TIMESTAMP AS OF timestamp` dopo il nome della tabella nell'istruzione `SELECT`, come nel seguente esempio:

```
SELECT * FROM iceberg_table FOR TIMESTAMP AS OF timestamp
```

L'ora del sistema da specificare per il viaggio è un timestamp o un timestamp con un fuso orario. Se non specificato, Athena considera il valore come un timestamp nell'ora UTC.

Nell'esempio seguente, le interrogazioni sui viaggi nel tempo selezionano i CloudTrail dati per la data e l'ora specificate.

```
SELECT * FROM iceberg_table FOR TIMESTAMP AS OF TIMESTAMP '2020-01-01 10:00:00 UTC'
```

```
SELECT * FROM iceberg_table FOR TIMESTAMP AS OF (current_timestamp - interval '1' day)
```

Query di versione

Per eseguire una query di versione (ovvero visualizzare uno snapshot coerente a partire da una versione specificata), utilizza `FOR VERSION AS OF` *`version`* dopo il nome della tabella nell'istruzione `SELECT`, come nell'esempio seguente.

```
SELECT * FROM [db_name.]table_name FOR VERSION AS OF version
```

Il parametro *`version`* è l'ID snapshot `bigint` associato a una versione della tabella Iceberg.

La seguente query di versione di esempio seleziona i dati per la versione specificata.

```
SELECT * FROM iceberg_table FOR VERSION AS OF 949530903748831860
```

> **Note**
>
> Le clausole `FOR SYSTEM_TIME AS OF` e `FOR SYSTEM_VERSION AS OF` nella versione 2 del motore Athena sono state sostituite dalle clausole `FOR TIMESTAMP AS OF` e `FOR VERSION AS OF` nella versione 3 del motore Athena.

Recupero dell'ID snapshot

È possibile utilizzare la SnapshotUtilclasse Java fornita da Iceberg per recuperare l'ID dello snapshot Iceberg, come nell'esempio seguente.

```
import org.apache.iceberg.Table;
import org.apache.iceberg.aws.glue.GlueCatalog;
import org.apache.iceberg.catalog.TableIdentifier;
import org.apache.iceberg.util.SnapshotUtil;

import java.text.SimpleDateFormat;
```

```
import java.util.Date;

Catalog catalog = new GlueCatalog();

Map<String, String> properties = new HashMap<String, String>();
properties.put("warehouse", "s3://DOC-EXAMPLE-BUCKET/my-folder");
catalog.initialize("my_catalog", properties);

Date date = new SimpleDateFormat("yyyy/MM/dd HH:mm:ss").parse("2022/01/01 00:00:00");
long millis = date.getTime();

TableIdentifier name = TableIdentifier.of("db", "table");
Table table = catalog.loadTable(name);
long oldestSnapshotIdAfter2022 = SnapshotUtil.oldestAncestorAfter(table, millis);
```

Combinazione di query temporali e di versione

È possibile utilizzare la sintassi delle query temporali e di versione nella stessa query per specificare condizioni di temporizzazione e controllo delle versioni diverse, come nell'esempio seguente.

```
SELECT table1.*, table2.* FROM
  [db_name.]table_name FOR TIMESTAMP AS OF (current_timestamp - interval '1' day) AS
 table1
  FULL JOIN
  [db_name.]table_name FOR VERSION AS OF 5487432386996890161 AS table2
  ON table1.ts = table2.ts
  WHERE (table1.id IS NULL OR table2.id IS NULL)
```

Creazione ed esecuzione di query sulle viste con le tabelle Iceberg

Per creare ed eseguire query sulle viste Athena sulle tabelle Iceberg, utilizza le viste `CREATE VIEW` come descritto nella pagina Utilizzo delle visualizzazioni.

Esempio:

```
CREATE VIEW view1 AS SELECT * FROM iceberg_table
```

```
SELECT * FROM view1
```

Se desideri utilizzare le specifiche di visualizzazione Iceberg per creare viste, contatta athena-feedback@amazon.com.

Utilizzo del controllo granulare degli accessi di Lake Formation

La versione 3 del motore Athena supporta il controllo granulare degli accessi Lake Formation con le tabelle Iceberg, incluso il controllo degli accessi con sicurezza a livello di colonna e riga. Questo controllo dell'accesso funziona con le query temporali e con le tabelle che hanno eseguito l'evoluzione dello schema. Per ulteriori informazioni, consulta Controllo granulare degli accessi di Lake Formation e gruppi di lavoro Athena.

Se hai creato la tabella Iceberg al di fuori di Athena, utilizza l'SDK di Apache Iceberg versione 0.13.0 o successiva in modo che le informazioni della colonna della tabella Iceberg siano inserite nel AWS Glue Data Catalog. Se la tua tabella Iceberg non contiene informazioni sulle colonne in AWS Glue, puoi utilizzare l'istruzione ALTER TABLE SET PROPERTIES Athena o l'ultimo Iceberg SDK per correggere la tabella e aggiornare le informazioni sulle colonne in. AWS Glue

Aggiornamento dati di tabelle Iceberg

I dati della tabella Iceberg possono essere gestiti direttamente su Athena utilizzando le query `INSERT`, `UPDATE` e `DELETE`. Ogni transazione di gestione dei dati produce un nuovo snapshot, che può essere interrogato utilizzando una query temporale. Le istruzioni `UPDATE` e `DELETE` seguono le specifiche di eliminazione della posizione a livello di riga del formato Iceberg v2 e applicano l'isolamento della snapshot.

Utilizzare i seguenti comandi per eseguire operazioni di gestione dei dati sulle tabelle Iceberg.

INSERT INTO

Inserisce i dati in una tabella Iceberg. Ad Athena Iceberg `INSERT INTO` si applica una tariffa uguale alle query `INSERT INTO` attuali per le tabelle Hive esterne in base alla quantità di dati scansionati. Per inserire dati in una tabella Iceberg, utilizza la seguente sintassi, dove *`query`* può essere `VALUES (val1, val2, ...)` o `SELECT (col1, col2, …) FROM [`*`db_name`*`.]`*`table_name`* `WHERE` *`predicate`*. Per la sintassi SQL e i dettagli semantici, consulta la pagina INSERT INTO.

```
INSERT INTO [db_name.]table_name [(col1, col2, …)] query
```

Negli esempi seguenti vengono inseriti i valori nella tabella `iceberg_table`.

```
INSERT INTO iceberg_table VALUES (1,'a','c1')
```

```
INSERT INTO iceberg_table (col1, col2, ...) VALUES (val1, val2, ...)
```

```
INSERT INTO iceberg_table SELECT * FROM another_table
```

DELETE

Athena Iceberg `DELETE` scrive i file di eliminazione della posizione Iceberg in una tabella. Questa operazione è nota come eliminazione. merge-on-read A differenza di un'copy-on-writeeliminazione, un' merge-on-read eliminazione è più efficiente perché non riscrive i dati del file. Quando Athena legge i dati Iceberg, unisce i file di eliminazione della posizione Iceberg con i file di dati per produrre la visualizzazione più recente di una tabella. Per rimuovere questi file di eliminazione della posizione, puoi eseguire Azione di compattazione REWRITE DATA. Le operazioni `DELETE` vengono addebitate dalla quantità di dati analizzati. Per la sintassi, consulta DELETE.

L'esempio seguente elimina le righe da `iceberg_table` che hanno `c3` come valore per `category`.

```
DELETE FROM iceberg_table WHERE category='c3'
```

UPDATE

Athena Iceberg `UPDATE` scrive i file di eliminazione della posizione Iceberg e le righe appena aggiornate come file di dati nella stessa transazione. `UPDATE`può essere immaginato come una combinazione di `INSERT INTO` e `DELETE`. Le operazioni `UPDATE` vengono addebitate dalla quantità di dati analizzati. Per la sintassi, consulta UPDATE.

L'esempio seguente aggiorna i valori specificati nella tabella `iceberg_table`.

```
UPDATE iceberg_table SET category='c2' WHERE category='c1'
```

MERGE INTO

Aggiorna, elimina o inserisce in modo condizionale righe in una tabella Iceberg. Una singola istruzione può combinare operazioni di aggiornamento, eliminazione e inserimento. Per la sintassi, consulta MERGE INTO.

> **Note**
>
> L'istruzione `MERGE INTO` è transazionale ed è supportata solo per le tabelle Apache Iceberg nella versione 3 del motore Athena.

L'esempio seguente elimina tutti i clienti dalla tabella `t` che si trova nella tabella di origine `s`.

```
MERGE INTO accounts t USING monthly_accounts_update s
ON t.customer = s.customer
WHEN MATCHED
THEN DELETE
```

L'esempio seguente aggiorna la tabella di destinazione `t` con le informazioni sui clienti dalla tabella di origine `s`. Per le righe dei clienti nella tabella `t` che contengono righe relative ai clienti nella tabella `s`, l'esempio incrementa gli acquisti nella tabella t. Se la tabella `t` non corrisponde a una riga del cliente nella tabella `s`, l'esempio inserisce la riga del cliente dalla tabella `s` nella tabella `t`.

```
MERGE INTO accounts t USING monthly_accounts_update s
    ON (t.customer = s.customer)
    WHEN MATCHED
        THEN UPDATE SET purchases = s.purchases + t.purchases
    WHEN NOT MATCHED
        THEN INSERT (customer, purchases, address)
              VALUES(s.customer, s.purchases, s.address)
```

L'esempio seguente aggiorna in modo condizionale la tabella di destinazione `t` con le informazioni dalla tabella di origine `s`. L'esempio elimina qualsiasi riga di destinazione corrispondente il cui indirizzo di origine è Centreville. Per tutte le altre righe corrispondenti, l'esempio aggiunge gli acquisti di origine e imposta l'indirizzo di destinazione sull'indirizzo di origine. Se non c'è alcuna corrispondenza nella tabella di destinazione, l'esempio inserisce la riga dalla tabella di origine.

```
MERGE INTO accounts t USING monthly_accounts_update s
    ON (t.customer = s.customer)
    WHEN MATCHED AND s.address = 'Centreville'
        THEN DELETE
    WHEN MATCHED
        THEN UPDATE
            SET purchases = s.purchases + t.purchases, address = s.address
    WHEN NOT MATCHED
        THEN INSERT (customer, purchases, address)
              VALUES(s.customer, s.purchases, s.address)
```

Ottimizzazione delle tabelle Iceberg

Man mano che i dati si accumulano in una tabella Iceberg, le query diventano gradualmente meno efficienti a causa del maggiore tempo di elaborazione richiesto per aprire i file. Se la tabella contiene

file di eliminazione si verificano costi computazionali aggiuntivi. In Iceberg, i file eliminati archiviano le eliminazioni a livello di riga e il motore deve applicare le righe eliminate ai risultati della query.

Per ottimizzare le prestazioni delle query sulle tabelle Iceberg, Athena supporta la compattazione manuale come comando di manutenzione delle tabelle. Le compattazioni ottimizzano il layout strutturale della tabella senza alterare il contenuto della tabella.

OPTIMIZE

L'operazione di compattazione `OPTIMIZE table REWRITE DATA` riscrive i file di dati in un layout più ottimizzato in base alle dimensioni e al numero di file di eliminazione associati. Per i dettagli sulla sintassi e sulle proprietà della tabella, consulta la pagina OPTIMIZE.

Esempio

L'esempio seguente unisce i file di eliminazione in file di dati e produce file vicino alle dimensioni del file di destinazione dove il valore di `category` è `c1`.

```
OPTIMIZE iceberg_table REWRITE DATA USING BIN_PACK
  WHERE category = 'c1'
```

VACUUM

`VACUUM` esegue la scadenza degli snapshot e la rimozione dei file orfani. Queste operazioni riducono le dimensioni dei metadati e rimuovono i file non nello stato corrente della tabella che sono anche più vecchi del periodo di conservazione specificato per la tabella. Per i dettagli sulla sintassi, consulta la pagina VACUUM.

Esempio

L'esempio seguente utilizza una proprietà della tabella per configurare la tabella `iceberg_table` in modo che mantenga i dati degli ultimi tre giorni, quindi utilizza `VACUUM` per far scadere i vecchi snapshot e rimuovere i file orfani dalla tabella.

```
ALTER TABLE iceberg_table SET TBLPROPERTIES (
  'vacuum_max_snapshot_age_seconds'='259200'
)

VACUUM iceberg_table
```

Tipi di dati supportati per tabelle Iceberg in Athena

Athena può eseguire query su tabelle Iceberg che contengono i seguenti tipi di dati:

```
binary
boolean
date
decimal
double
float
int
list
long
map
string
struct
timestamp without time zone
```

Per ulteriori informazioni sui tipi di tabella Iceberg, consulta la pagina degli schemi per Iceberg nella documentazione di Apache.

La tabella riportata di seguito mostra la relazione tra i tipi di dati Athena e i tipi di dati della tabella Iceberg.

Tipo Iceberg	Tipo Athena	Note
`boolean`	`boolean`	
-	`tinyint`	Non supportato per tabelle Iceberg in Athena.
-	`smallint`	Non supportato per tabelle Iceberg in Athena.
`int`	`int`	Nelle istruzioni DML di Athena, questo tipo è `INTEGER`.
`long`	`bigint`	
`double`	`double`	
`float`	`float`	

Tipo Iceberg	Tipo Athena	Note
`decimal(P, S)`	`decimal(P, S)`	`P` è precisione, `S` è scala.
-	`char`	Non supportato per tabelle Iceberg in Athena.
`string`	`string`	Nelle istruzioni DML di Athena, questo tipo è `VARCHAR`.
`binary`	`binary`	
`date`	`date`	
`time`	-	Solo il timestamp Iceberg (senza fuso orario) è supportato per le istruzioni DDL di Athena Iceberg come `CREATE TABLE`, ma tutti i tipi di timestamp possono essere interrogati tramite Athena.
`timestamp`	`timestamp`	
`timestamp tz`	`timestamp tz`	
`list<E>`	`array`	
`map<K,V>`	`map`	
`struct<...>`	`struct`	
`fixed(L)`	-	Il tipo `fixed(L)` non è attualmente supportato in Athena.

Per ulteriori informazioni sui tipi di dati in Athena, consultare Tipi di dati in Amazon Athena.

Altre operazioni di Athena sulle tabelle Iceberg

Operazioni a livello di database

Quando utilizzi DROP DATABASE con l'opzione `CASCADE`, tutti i dati della tabella Iceberg vengono rimossi. Le seguenti operazioni DDL non hanno alcun effetto sulle tabelle Iceberg.

- CREATE DATABASE

- ALTER DATABASE SET DBPROPERTIES
- SHOW DATABASES
- SHOW TABLES
- SHOW VIEWS

Operazioni relative al partizionamento

Poiché le tabelle Iceberg utilizzano il partizionamento nascosto, non è necessario lavorare direttamente con le partizioni fisiche. Di conseguenza, le tabelle Iceberg di Athena non supportano le seguenti operazioni DDL relative alle partizioni:

- SHOW PARTITIONS
- ALTER TABLE ADD PARTITION
- ALTER TABLE DROP PARTITION
- ALTER TABLE RENAME PARTITION

Se si desidera vedere l'evoluzione della partizione Iceberg in Athena, inviare un feedback all'indirizzo athena-feedback@amazon.com.

Scaricamento tabella Iceberg

Le tabelle Iceberg possono essere scaricate nei file di una cartella su Amazon S3. Per informazioni, consulta UNLOAD.

RIPARAZIONE MSCK

Poiché le tabelle Iceberg tengono traccia delle informazioni sul layout della tabella, l'esecuzione di MSCK REPAIR TABLE come quella delle tabelle Hive non è necessaria e non è supportata.

Risorse aggiuntive

Il seguente articolo si trova nella documentazione di AWS Prescriptive Guidance.

- Utilizzo delle tabelle Apache Iceberg utilizzando Amazon Athena SQL

Per articoli approfonditi sull'utilizzo di Athena con le tabelle Apache Iceberg, consulta i seguenti post nel Blog sui Big Data di AWS .

- Implementa un processo CDC serverless con Apache Iceberg utilizzando Amazon DynamoDB e Amazon Athena
- Accelera l'ingegneria delle funzionalità di data science sui data lake transazionali tramite Amazon Athena con Apache Iceberg
- Crea un data lake Apache Iceberg utilizzando Amazon Athena, Amazon EMR e AWS Glue
- Esegui upsert in un data lake con Amazon Athena e Apache Iceberg
- Crea un data lake transazionale utilizzando Apache Iceberg e condividi dati tra account utilizzando Amazon Athena AWS GlueAWS Lake Formation
- Utilizza Apache Iceberg in un data lake per supportare l'elaborazione incrementale dei dati
- Crea un data lake Apache Iceberg allineato al GDPR in tempo reale
- Automatizza la replica delle fonti relazionali in un data lake transazionale con Apache Iceberg e AWS Glue
- Interagisci con le tabelle Apache Iceberg utilizzando Amazon Athena e autorizzazioni granulari tra account utilizzando AWS Lake Formation
- Crea un data lake transazionale serverless con Apache Iceberg, Amazon EMR Serverless e Amazon Athena

Sicurezza di Amazon Athena

La sicurezza del cloud AWS è la massima priorità. In qualità di AWS cliente, puoi beneficiare di un data center e di un'architettura di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra AWS te e te. Il modello di responsabilità condivisa descrive questo come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud: AWS è responsabile della protezione dell'infrastruttura che funziona Servizi AWS nel AWS cloud. AWS ti fornisce anche servizi che puoi utilizzare in modo sicuro. L'efficacia della nostra sicurezza è regolarmente testata e verificata da revisori di terze parti come parte dei programmi di conformitàAWS. Per ulteriori informazioni sui programmi di conformità che si applicano ad Athena, consulta Servizi AWS coperti dal programma di conformità.
- Sicurezza nel cloud: la tua responsabilità è determinata dall'uso Servizio AWS che utilizzi. L'utente è anche responsabile per altri fattori, tra cui la riservatezza dei dati, i requisiti dell'azienda, nonché le leggi e le normative applicabili.

Questa documentazione consente di comprendere come applicare il modello di responsabilità condivisa quando utilizzi Amazon Athena. I seguenti argomenti illustrano come configurare Athena per soddisfare gli obiettivi di sicurezza e conformità. Imparerai anche a usarne altri Servizi AWS che possono aiutarti a monitorare e proteggere le tue risorse Athena.

Argomenti

Protezione dei dati in Athena

Il modello di responsabilità AWS condivisa Modello si applica alla protezione dei dati in Amazon Athena. Come descritto in questo modello, AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutti i Cloud AWS. L'utente è responsabile del controllo dei contenuti ospitati su questa infrastruttura. L'utente è inoltre responsabile della configurazione della protezione e delle attività di gestione per i Servizi AWS utilizzati. Per ulteriori informazioni sulla privacy dei dati, vedi le Domande frequenti sulla privacy dei dati. Per informazioni sulla protezione dei dati in Europa, consulta il post del blog relativo al Modello di responsabilità condivisa AWS e GDPR nel Blog sulla sicurezza AWS .

Ai fini della protezione dei dati, consigliamo di proteggere Account AWS le credenziali e configurare i singoli utenti con AWS IAM Identity Center or AWS Identity and Access Management (IAM). In tal modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere i suoi compiti. Ti suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Utilizza l'autenticazione a più fattori (MFA) con ogni account.
- Usa SSL/TLS per comunicare con le risorse. AWS È richiesto TLS 1.2 ed è consigliato TLS 1.3.
- Configura l'API e la registrazione delle attività degli utenti con. AWS CloudTrail

- Utilizza soluzioni di AWS crittografia, insieme a tutti i controlli di sicurezza predefiniti all'interno Servizi AWS.
- Utilizza i servizi di sicurezza gestiti avanzati, come Amazon Macie, che aiutano a individuare e proteggere i dati sensibili archiviati in Amazon S3.
- Se hai bisogno di moduli crittografici convalidati FIPS 140-2 per l'accesso AWS tramite un'interfaccia a riga di comando o un'API, utilizza un endpoint FIPS. Per ulteriori informazioni sugli endpoint FIPS disponibili, consulta il Federal Information Processing Standard (FIPS) 140-2.

Ti consigliamo vivamente di non inserire mai informazioni riservate o sensibili, ad esempio gli indirizzi e-mail dei clienti, nei tag o nei campi di testo in formato libero, ad esempio nel campo Nome. Ciò include quando lavori con Athena o altri utenti Servizi AWS utilizzando la console, l'API o AWS gli AWS CLI SDK. I dati inseriti nei tag o nei campi di testo in formato libero utilizzati per i nomi possono essere utilizzati per i la fatturazione o i log di diagnostica. Quando fornisci un URL a un server esterno, ti suggeriamo vivamente di non includere informazioni sulle credenziali nell'URL per convalidare la tua richiesta al server.

Come ulteriore misura di sicurezza, puoi utilizzare la chiave di contesto di condizione globale aws:CalledVia per limitare le richieste solo a quelle effettuate da Athena. Per ulteriori informazioni, consulta Usare Athena con CalledVia le chiavi di contesto.

Protezione di più tipi di dati

Quando si utilizza Athena per la creazione di database e di tabelle, sono coinvolti vari tipi di dati. Questi tipi di dati includono i dati di origine archiviati in Amazon S3, i metadati per database e tabelle che crei quando esegui le query o il AWS Glue Crawler per scoprire dati, i dati dei risultati delle query e la cronologia delle query. Questa sezione illustra ciascun tipo di dati e fornisce linee guida su come proteggerli.

- Dati origine: i dati dei database e delle tabelle vengono archiviati in Amazon S3 e Athena non li modifica. Per ulteriori informazioni, consulta Protezione dei dati in Amazon S3 nella Guida per l'utente di Amazon Simple Storage Service. È possibile controllare l'accesso ai dati di origine ed è possibile crittografarli in Amazon S3. È possibile utilizzare Athena per creare tabelle in base ai set di dati crittografati in Amazon S3.
- Metadati di database e tabelle (schema): Athena schema-on-read utilizza la tecnologia, il che significa che le definizioni delle tabelle vengono applicate ai dati in Amazon S3 quando Athena esegue le query. Qualsiasi schema creato viene salvato automaticamente, a meno che non venga esplicitamente eliminato. In Athena è possibile modificare i metadati del Catalogo dati utilizzando

le dichiarazioni DDL. È possibile eliminare definizioni delle tabelle e schemi senza alcun impatto sui dati sottostanti archiviati in Amazon S3. I metadati per le tabelle e i database utilizzati in Athena vengono archiviati in AWS Glue Data Catalog.

Puoi definire politiche di accesso granulari a database e tabelle registrati in Using (IAM). AWS Glue Data Catalog AWS Identity and Access Management È anche possibile crittografare i metadati in AWS Glue Data Catalog. Se si esegue la crittografia dei metadati, utilizzare le autorizzazioni ai dati crittografati per l'accesso.

- Risultati delle query e cronologia delle query, incluse le query salvate: i risultati delle query vengono archiviati in una posizione in Amazon S3 che puoi specificare a livello globale o per ciascun gruppo di lavoro. Se non è specificata, Athena utilizza la posizione predefinita in ogni caso. Puoi controllare l'accesso ai bucket Amazon S3 in cui si archiviano i risultati delle query e le query salvate. Inoltre, è possibile scegliere di crittografare i risultati delle query archiviate in Amazon S3. Gli utenti devono disporre delle autorizzazioni adeguate per accedere alle posizioni in Amazon S3 e decrittare i file. Per ulteriori informazioni, consulta la sezione Crittografia dei risultati di query Athena archiviati in Amazon S3 riportata di seguito in questo documento.

  Athena conserva la cronologia delle query per 45 giorni. Puoi visualizzare la cronologia delle query utilizzando le API Athena, nella console e con. AWS CLI Per archiviare le query per più di 45 giorni, salvarle. Per proteggere l'accesso alle query salvate, è possibile usare i gruppi di lavoro in Athena, limitando l'accesso alle query salvate solo agli utenti che sono autorizzati a visualizzarle.

Argomenti

Crittografia a riposo

Puoi eseguire query su Amazon Athena su dati crittografati in Amazon S3 nella stessa Regione e in un numero limitato di Regioni. Puoi anche crittografare i risultati delle query in Amazon S3 e i dati nel Data Catalog AWS Glue .

È possibile crittografare le seguenti risorse in Athena:

- I risultati di tutte le query in Amazon S3, che Athena archivia in un percorso noto, ad esempio la posizione dei risultati in Amazon S3. È possibile crittografare i risultati delle query archiviati in Amazon S3, indipendentemente dal fatto che il set di dati sottostante sia crittografato in Amazon S3 o meno. Per informazioni, consulta Crittografia dei risultati di query Athena archiviati in Amazon S3.
- I dati nel Data AWS Glue Catalog. Per informazioni, consulta Autorizzazioni di accesso ai metadati crittografati nel catalogo dati di AWS Glue.

> **Note**
>
> Quando si utilizza Athena per leggere una tabella crittografata, Athena utilizza le opzioni di crittografia specificate per i dati della tabella, non l'opzione di crittografia per i risultati della query. Se sono configurati metodi o chiavi di crittografia separati per i risultati delle query e i dati della tabella, Athena legge i dati della tabella senza utilizzare l'opzione di crittografia e la chiave utilizzate per crittografare o decrittografare i risultati della query.
> Tuttavia, se si utilizza Athena per inserire dati in una tabella con dati crittografati, Athena utilizza la configurazione di crittografia specificata per i risultati della query per crittografare i dati inseriti. Ad esempio, se si specifica la `CSE_KMS` crittografia per i risultati delle query, Athena utilizza lo stesso ID AWS KMS chiave utilizzato per la crittografia dei risultati delle query per crittografare i dati della tabella inseriti. `CSE_KMS`

Argomenti

- Opzioni di crittografia supportate da Amazon S3
- Autorizzazioni di accesso a dati crittografati in Amazon S3
- Autorizzazioni di accesso ai metadati crittografati nel catalogo dati di AWS Glue
- Crittografia dei risultati di query Athena archiviati in Amazon S3
- Creazione di tabelle basate su set di dati crittografati in Amazon S3

Opzioni di crittografia supportate da Amazon S3

Athena supporta le seguenti opzioni di crittografia per set di dati e risultati di query in Amazon S3.

Tipo di crittografia	Descrizione	Supporto tra regioni
SSE-S3	Crittografia lato server (SSE) con una chiave gestita da Amazon S3.	Sì
SSE-KMS	Crittografia lato server (SSE) con una chiave gestita dal cliente. AWS Key Management Service **Note** Con questo tipo di crittografia, Athena non richiede di indicare che i dati vengono crittografati durante la creazione di una tabella.	Sì
CSE-KMS	Crittografia lato client (CSE) con chiave gestita dal cliente. AWS KMS In Athena, questa opzione richiede l'utilizzo di un'istruzione `CREATE TABLE` con una clausola `TBLPROPERTIES` che specifica `'has_encrypted_data'='true'` . Per ulteriori informazioni, consulta Creazione di tabelle basate su set di dati crittografati in Amazon S3.	No

Per ulteriori informazioni sulla AWS KMS crittografia con Amazon S3, consulta What is AWS Key Management Service and How Amazon Simple Storage Service (Amazon S3) nella Developer Guide AWS KMS.AWS Key Management Service Per ulteriori informazioni sull'utilizzo di SSE-KMS o CSE-KMS con Athena, consulta Lancio: Amazon Athena aggiunge il supporto per le query dei dati crittografati nel blog AWS Big Data.

Opzioni non supportate

Le seguenti opzioni di crittografia non sono supportate:

- SSE con chiavi fornite dal cliente (SSE-C)
- File crittografato lato client con una chiave gestita lato client
- Chiavi asimmetriche.

Per confrontare le opzioni di crittografia Amazon S3, consulta Protezione dei dati con la crittografia nella Guida per l'utente di Amazon Simple Storage Service.

Strumenti per la crittografia lato client

Per il file crittografato lato client, sono disponibili due strumenti:

- Client di crittografia Amazon S3: crittografa i dati solo per Amazon S3 ed è supportato da Athena.
- AWS Encryption SDK— L'SDK può essere utilizzato per crittografare i dati ovunque, AWS ma non è supportato direttamente da Athena.

Questi strumenti non sono compatibili e i dati crittografati con uno strumento non possono essere decrittati dall'altro. Athena supporta solo il client di crittografia Amazon S3. Se si utilizza l'SDK per crittografare i dati, è possibile eseguire query da Athena, ma i dati vengono restituiti come testo crittografato.

Se si desidera utilizzare Athena per interrogare i dati che sono stati crittografati con l'SDK di crittografia AWS , è necessario scaricare e decrittare i dati e quindi crittografarli nuovamente utilizzando il client di crittografia Amazon S3.

Autorizzazioni di accesso a dati crittografati in Amazon S3

A seconda del tipo di crittografia utilizzato in Amazon S3, potrebbe essere necessario aggiungere delle autorizzazioni, note anche come operazioni "Consenti", per le tue policy utilizzate in Athena:

- SSE-S3: se utilizzi SSE-S3 per la crittografia, gli utenti Athena non devono disporre di ulteriori autorizzazioni nelle loro policy. È sufficiente disporre delle autorizzazioni Amazon S3 adeguate per il percorso Amazon S3 corretto e per le operazioni Athena. Per ulteriori informazioni sulle policy che consentono di disporre delle autorizzazioni Athena e Amazon S3 adeguate, consulta le sezioni AWS politiche gestite per Amazon Athena e Accesso ad Amazon S3 da Athena.
- AWS KMS— Se si utilizza AWS KMS per la crittografia, gli utenti di Athena devono essere autorizzati a eseguire AWS KMS azioni particolari oltre alle autorizzazioni Athena e Amazon S3. Consenti queste azioni modificando la policy chiave per le CMK gestite dal AWS KMS cliente utilizzate per crittografare i dati in Amazon S3. Per aggiungere utenti chiave alle politiche AWS KMS chiave appropriate, puoi utilizzare la AWS KMS console all'indirizzo https://console.aws.amazon.com/kms. Per informazioni su come aggiungere un utente a una policy AWS KMS chiave, consulta Permette agli utenti chiave di utilizzare la CMK nella AWS Key Management Service Developer Guide.

> **Note**
>
> Gli amministratori di livello avanzato delle policy di chiavi possono modificare le policy stesse. `kms:Decrypt` è il livello minimo di operazioni consentite per un utente Athena affinché possa lavorare con un set di dati crittografato. Per utilizzare i risultati di query crittografati, i livelli minimi di operazioni consentite sono `kms:GenerateDataKey` e `kms:Decrypt`.

Quando si utilizza Athena per interrogare set di dati in Amazon S3 con un gran numero di oggetti crittografati con AWS KMS, AWS KMS può limitare i risultati delle query. Questo è più probabile che ciò accada quando è presente un numero elevato di piccoli oggetti. Athena esegue il backoff delle richieste di nuovi tentativi, tuttavia potrebbe comunque verificarsi un errore di limitazione. Se stai lavorando con un numero elevato di oggetti crittografati e riscontri questo problema, un'opzione è quella di abilitare le chiavi bucket Amazon S3 per ridurre il numero di chiamate a KMS. Per ulteriori informazioni, consulta Riduzione del costo di SSE-KMS con le chiavi bucket Amazon S3 nella Guida per l'utente di Amazon Simple Storage Service. Un'altra opzione è aumentare le quote di servizio per AWS KMS. Per ulteriori informazioni, consulta Quote nella Guida per gli sviluppatori di AWS Key Management Service .

Per informazioni sulla risoluzione dei problemi relativi alle autorizzazioni quando si utilizza Amazon S3 con Athena, consulta la sezione Autorizzazioni dell'argomento Risoluzione dei problemi in Athena.

Autorizzazioni di accesso ai metadati crittografati nel catalogo dati di AWS Glue

Se si crittografano i metadati in AWS Glue Data Catalog, è necessario aggiungere `"kms:GenerateDataKey"` e `"kms:Encrypt"` azioni alle politiche utilizzate per accedere ad Athena. `"kms:Decrypt"` Per informazioni, consulta Accesso da Athena ai metadati crittografati nel AWS Glue Data Catalog.

Crittografia dei risultati di query Athena archiviati in Amazon S3

Puoi impostare la crittografia dei risultati delle query utilizzando la console Athena o quando utilizzi JDBC o ODBC. I gruppi di lavoro consentono di applicare la crittografia dei risultati delle query.

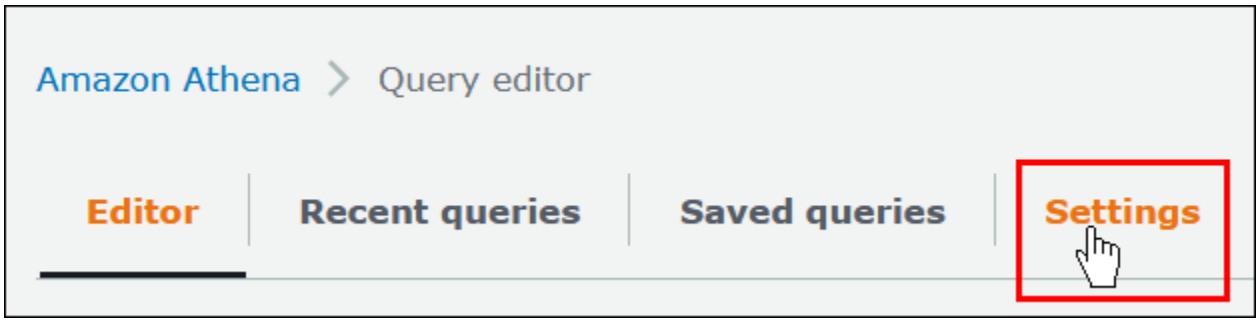
Dalla console puoi configurare le impostazioni per la crittografia dei risultati di query in due modi:

- Impostazioni lato client: quando utilizzi Impostazioni nella console o le operazioni API per indicare che vuoi crittografare i risultati delle query, usi le cosiddette impostazioni lato client. Le impostazioni lato client includono la posizione e la crittografia dei risultati delle query. Se le specifichi, vengono utilizzate a meno che non vengano sostituite dalle impostazioni del gruppo di lavoro.
- Impostazioni del gruppo di lavoro: quando si crea o si modifica un gruppo di lavoro e si seleziona il campo Override client-side settings (Ignora impostazioni lato client), tutte le query che vengono eseguite nel gruppo di lavoro usano la crittografia del gruppo di lavoro e le impostazioni del percorso dei risultati delle query. Per ulteriori informazioni, consulta Le impostazioni del gruppo di lavoro sostituiscono le impostazioni lato client.

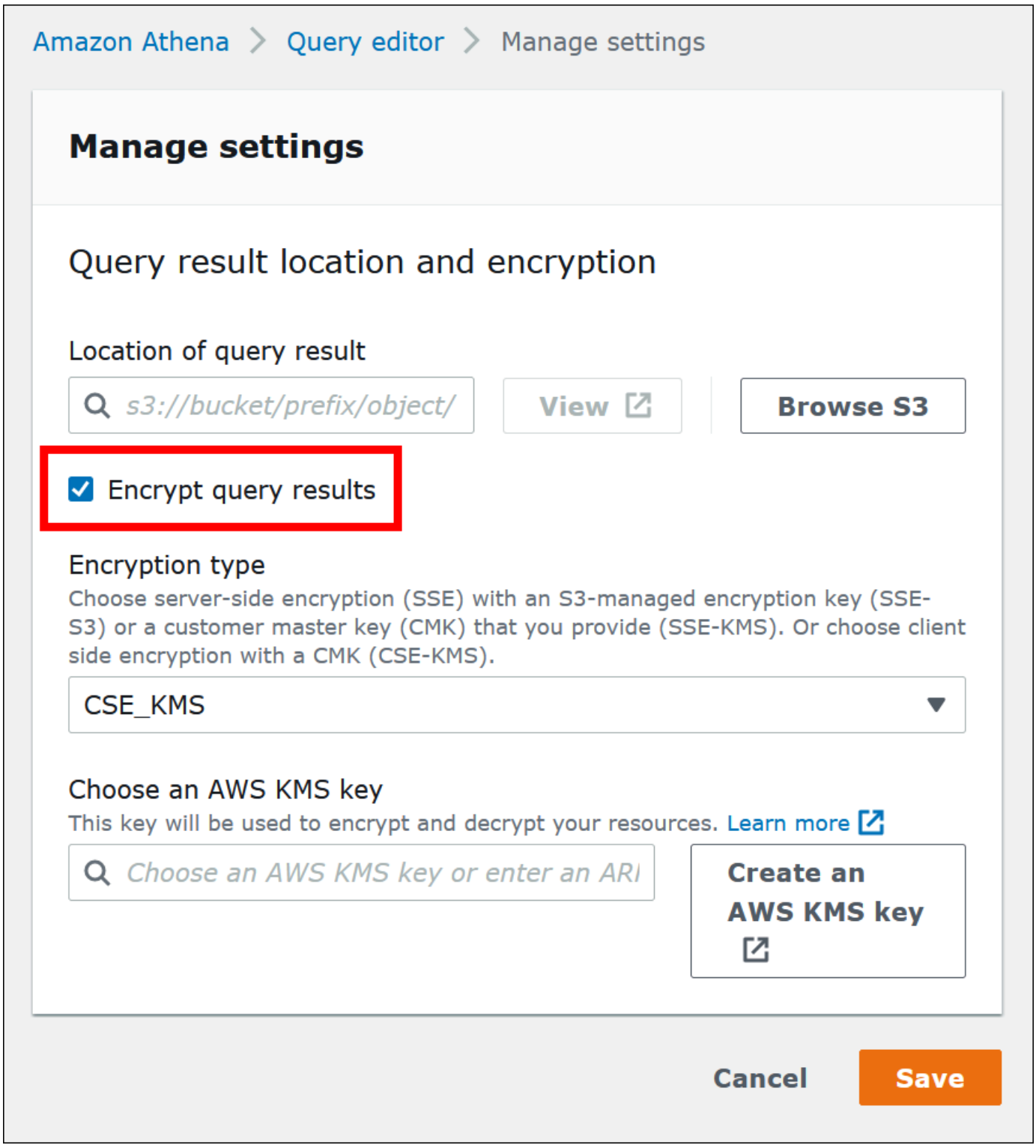
Per crittografare i risultati delle query archiviati in Amazon S3 con la console

> **Important**
>
> Se per il gruppo di lavoro è selezionato il campo Override client-side settings (Ignora impostazioni lato client), tutte le query nel gruppo di lavoro utilizzano le relative impostazioni. La configurazione della crittografia e il percorso dei risultati delle query specificati nella scheda Setting (Impostazioni) della console Athena, le operazioni API e i driver JDBC e ODBC non vengono utilizzati. Per ulteriori informazioni, consulta Le impostazioni del gruppo di lavoro sostituiscono le impostazioni lato client.

1. Nella console Athena, seleziona Impostazioni.



2. Scegli Gestisci.
3. Per Location of query result (Percorso dei risultati delle query), inserisci o scegli un percorso Amazon S3. Questo è il percorso Amazon S3 in cui sono archiviati i risultati della query.
4. Seleziona Encrypt query results (Esegui crittografia dei risultati delle query).



5. In Encryption type (Tipo di crittografia), seleziona CSE-KMS, SSE-KMS o SSE-S3. Di questi tre, CSE-KMS offre il livello di crittografia più elevato e SSE-S3 il più basso.
6. Se hai scelto SSE-KMS o CSE-KMS, specifica una chiave. AWS KMS

- Per Scegli una AWS KMS chiave, se il tuo account ha accesso a una chiave gestita AWS KMS dal cliente (CMK) esistente, scegli il relativo alias o inserisci una chiave AWS KMS ARN.
- Se il tuo account non ha accesso a una chiave gestita dal cliente (CMK) esistente, scegli Crea una AWS KMS chiave, quindi apri la console.AWS KMS Per ulteriori informazioni, consulta Creazione di chiavi nella Guida per gli sviluppatori di AWS Key Management Service .

> **Note**
>
> Athena supporta solo chiavi simmetriche per la lettura e la scrittura dei dati.

7. Torna alla console Athena e scegli la chiave creata dall'alias o dall'ARN.
8. Selezionare Salva.

Crittografia dei risultati delle query Athena quando si usa JDBC o ODBC

Se ci si connette utilizzando il driver JDBC o ODBC, è possibile configurare le opzioni del driver per specificare il tipo di crittografia da utilizzare e la posizione della directory di gestione temporanea di Amazon S3. Per configurare il driver JDBC o ODBC per crittografare i risultati delle query utilizzando i protocolli di crittografia supportati da Athena, consulta Connessione ad Amazon Athena con i driver ODBC e JDBC.

Creazione di tabelle basate su set di dati crittografati in Amazon S3

Quando crei una tabella, occorre indicare ad Athena che un set di dati è crittografato in Amazon S3. Non è invece necessario quando si utilizza SSE-KMS. Sia per SSE-S3 che per la crittografia AWS KMS , Athena determina come decrittografare il set di dati e creare la tabella, quindi non è necessario fornire informazioni chiave.

Gli utenti che eseguono le query, incluso l'utente che crea la tabella, devono disporre delle autorizzazioni descritte in precedenza in questo argomento.

> **Important**
>
> Se si utilizza Amazon EMR con EMRFS per caricare file Parquet crittografati, è necessario disabilitare i caricamenti in più parti impostando `fs.s3n.multipart.uploads.enabled` su `false`. Se non si effettua questa operazione, Athena non sarà in grado di determinare la lunghezza del file Parquet e si verificherà un errore HIVE_CANNOT_OPEN_SPLIT. Per

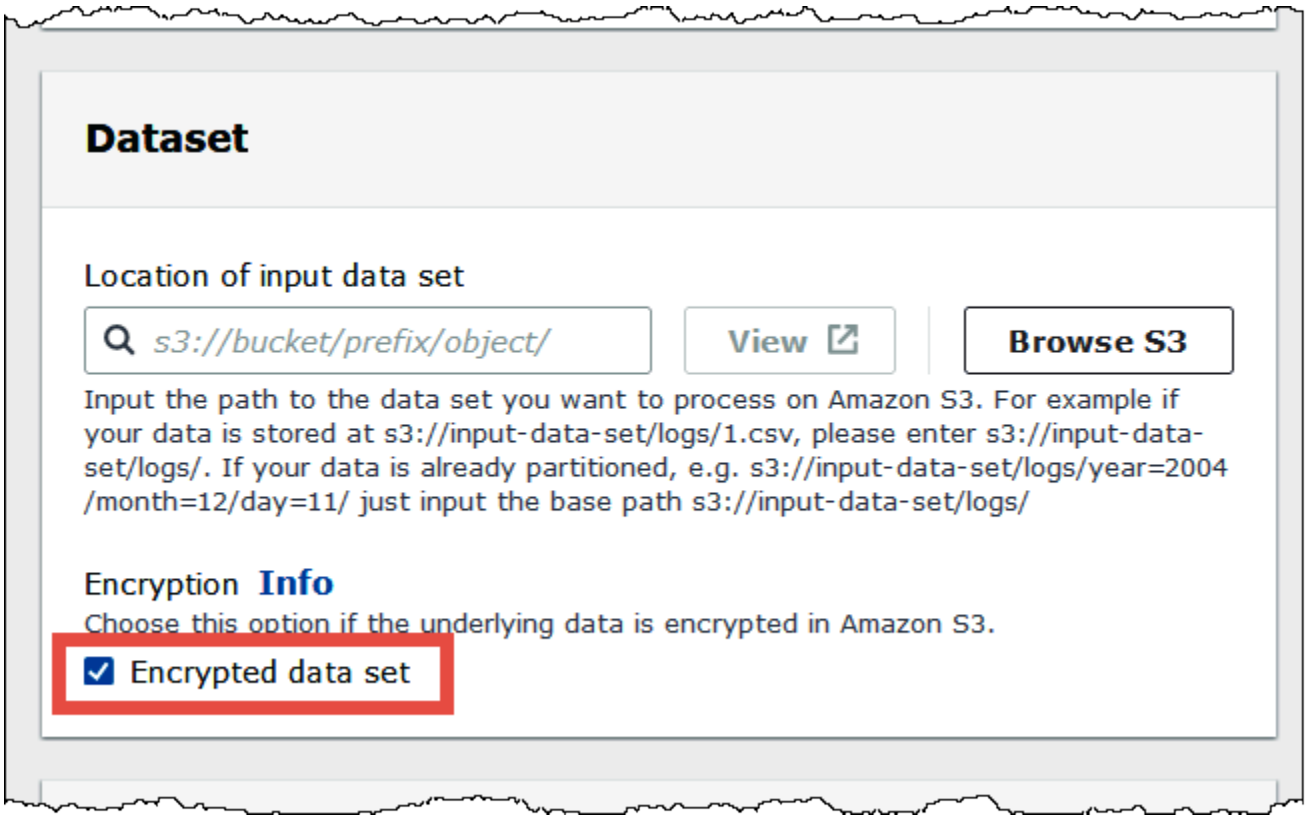
ulteriori informazioni, consulta Configurazione del caricamento in più parti per Amazon S3 nella Guida alla gestione di Amazon EMR.

Per indicare che il set di dati è crittografato in Amazon S3, esegui uno dei seguenti passaggi. Questo passaggio non è necessario se viene utilizzato SSE-KMS.

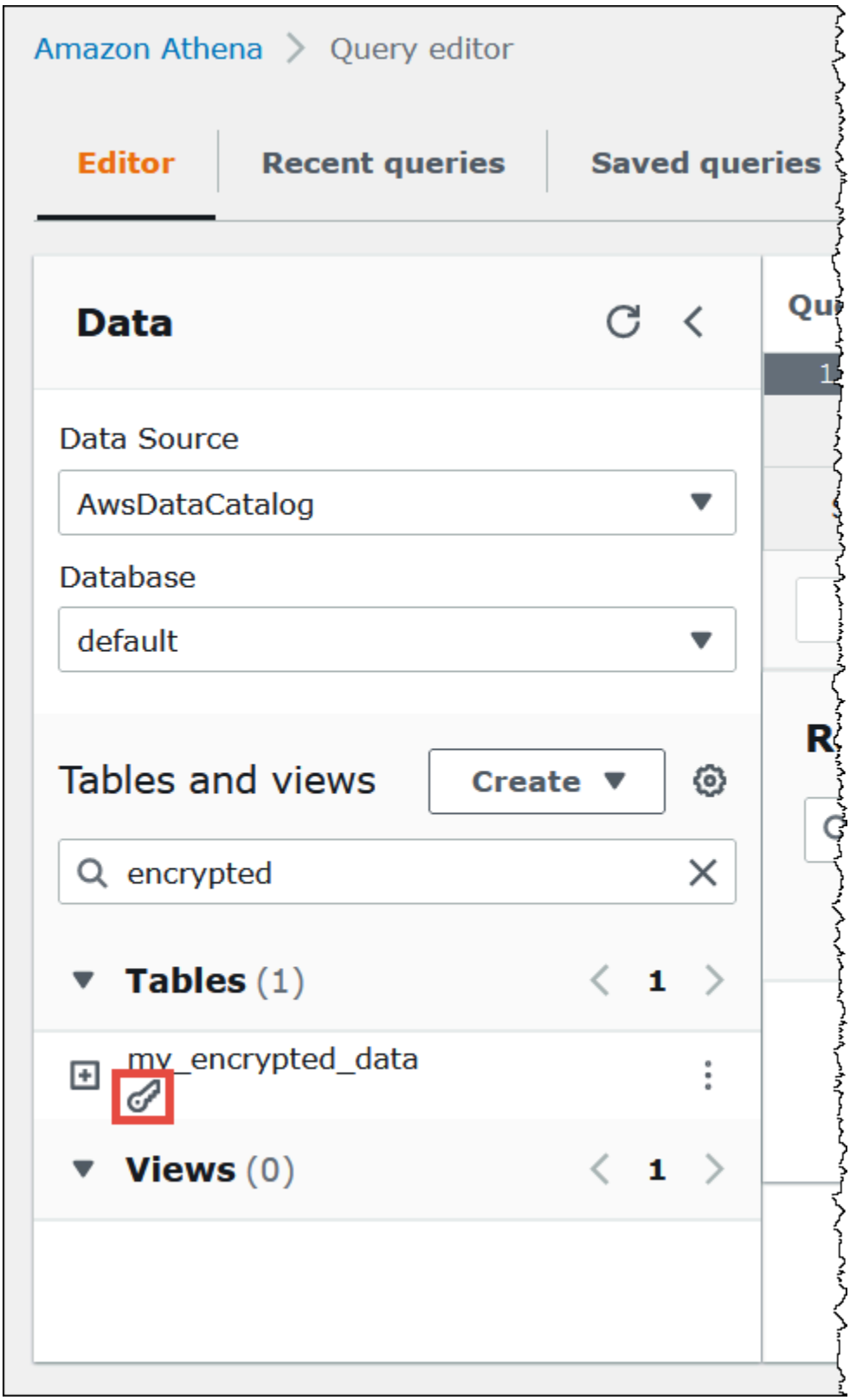
- In un'istruzione CREATE TABLE, usa una clausola `TBLPROPERTIES` che specifica `'has_encrypted_data'='true'`, come nel seguente esempio.

```
CREATE EXTERNAL TABLE 'my_encrypted_data' (
   `n_nationkey` int,
   `n_name` string,
   `n_regionkey` int,
   `n_comment` string)
ROW FORMAT SERDE
   'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT
   'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
LOCATION
   's3://DOC-EXAMPLE-BUCKET/folder_with_my_encrypted_data/'
TBLPROPERTIES (
   'has_encrypted_data'='true')
```

- Utilizza il driver JDBC e imposta il valore `TBLPROPERTIES` come illustrato nell'esempio precedente quando usi `statement.executeQuery()` per eseguire l'istruzione CREATE TABLE.
- Quando usi la console Athena per creare una tabella tramite un modello e specifichi una posizione della tabella, seleziona l'opzione Set di dati crittografati.



Nell'elenco di tabelle della console Athena, le tabelle crittografate mostrano un'icona con una chiave.


Amazon Athena
Query editor
Editor
Recent queries
Saved queries
Data
Data Source
AwsDataCatalog
Database
default
Tables and views
Create
encrypted
Tables (1)
1
my_encrypted_data
Views (0)
1

Crittografia in transito

Oltre a crittografare i dati inattivi in Amazon S3, Amazon Athena utilizza Transport Layer Security (TLS) per i dati in transito tra Athena e Amazon S3 e tra Athena e le applicazioni personalizzate che vi accedono.

È consigliabile consentire solo connessioni crittografate su HTTPS (TLS) utilizzando `aws:SecureTransport condition` sulle policy IAM del bucket Amazon S3.

I risultati delle query inviati ai client JDBC o ODBC sono crittografati con TLS. Per informazioni sulle versioni più recenti dei driver JDBC e ODBC e la relativa documentazione, vedere Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

Per i connettori di origine dati federati di Athena, il supporto della crittografia in transito tramite TLS dipende dal singolo connettore. Per informazioni, consulta la documentazione relativa ai singoli connettori di origine dati.

Gestione delle chiavi

Amazon Athena supporta AWS Key Management Service (AWS KMS) per crittografare i set di dati nei risultati delle query di Amazon S3 e Athena. AWS KMS utilizza chiavi gestite dai clienti (CMK) per crittografare gli oggetti Amazon S3 e si affida alla crittografia delle buste.

In AWS KMS, puoi eseguire le seguenti azioni:

- Creare chiavi
- Importare il materiale della chiave per le nuove CMK

> **Note**
>
> Athena supporta solo chiavi simmetriche per la lettura e la scrittura dei dati.

Per ulteriori informazioni, consulta Che cos'è AWS Key Management Service nella Guida per gli sviluppatori di AWS Key Management Service e Come Amazon Simple Storage Service usa AWS KMS. Per visualizzare le chiavi dell'account che AWS crea e gestisce per te, nel riquadro di navigazione, scegli chiavi AWS gestite.

Se stai caricando o accedendo a oggetti crittografati da SSE-KMS, usa la versione 4 di AWS Signature per una maggiore sicurezza. Per ulteriori informazioni, consulta Specifica di Signature Version nell'autenticazione delle richieste nella Guida per l'utente di Amazon Simple Storage Service.

Se i carichi di lavoro Athena crittografano una grande quantità di dati, è possibile utilizzare le chiavi bucket Amazon S3 per ridurre i costi. Per ulteriori informazioni, consulta Riduzione del costo di SSE-KMS con le chiavi bucket Amazon S3 nella Guida per l'utente di Amazon Simple Storage Service.

Riservatezza del traffico Internet

Il traffico è protetto sia tra Athena e le applicazioni on-premise che tra Athena e Amazon S3. Il traffico tra Athena e altri servizi, come AWS Glue e AWS Key Management Service, utilizza HTTPS per impostazione predefinita.

- Per il traffico tra Athena e i client locali e le applicazioni, i risultati delle query inviati ai client JDBC o ODBC vengono crittografati utilizzando Transport Layer Security (TLS).

  È possibile utilizzare una delle opzioni di connettività tra la tua rete privata e AWS:

  - Una connessione VPN da sito a sito AWS VPN . Per ulteriori informazioni, consulta Cos'è Site-to-Site VPN AWS VPN nella Guida per l'utente di AWS Site-to-Site VPN .
  - Una connessione. AWS Direct Connect Per ulteriori informazioni, consulta Che cos'è AWS Direct Connect? nella Guida per l'utente di AWS Direct Connect .
- Per il traffico tra Athena e i bucket Amazon S3, Transport Layer Security (TLS) crittografa gli oggetti in transito tra Athena e Amazon S3 e tra Athena e applicazioni dei clienti che vi accedono, è consigliabile consentire solo connessioni crittografate tramite HTTPS (TLS) utilizzando `aws:SecureTransport condition` sulle policy IAM del bucket Amazon S3. Sebbene Athena utilizzi attualmente l'endpoint pubblico per accedere ai dati nei bucket Amazon S3, ciò non significa che i dati attraversino la rete Internet pubblica. Tutto il traffico tra Athena e Amazon S3 viene instradato sulla rete e crittografato tramite AWS TLS.
- Programmi di conformità: Amazon Athena è conforme a diversi programmi di AWS conformità, tra cui SOC, PCI, FedRAMP e altri. Per ulteriori informazioni, consulta Servizi AWS coperti dal programma di conformità.

Gestione dell'identità e dell'accesso in Athena

Amazon Athena utilizza le policy AWS Identity and Access Management (IAM) per limitare l'accesso alle operazioni Athena. Per l'elenco completo delle autorizzazioni per Athena, consulta Operazioni, risorse e chiavi di condizione per Amazon Athena nella Referenza sulle autorizzazioni per il servizio.

Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Le autorizzazioni richieste per eseguire query Athena includono:

- Posizioni Amazon S3 in cui sono archiviati i dati sottostanti. Per maggiori informazioni, consulta Identity and Access Management in Amazon S3 nella Guida per l'utente di Amazon Simple Storage.
- Metadati e risorse archiviati in AWS Glue Data Catalog, ad esempio database e tabelle, incluse azioni aggiuntive per i metadati crittografati. Per ulteriori informazioni, consulta Impostazione delle autorizzazioni IAM per AWS Glue e Configurazione della crittografia in AWS Glue nella Guida per gli sviluppatori di AWS Glue .
- Operazioni API Athena. Per un elenco completo delle operazioni in Athena, consulta Operazioni nella Referenza per l'API Amazon Athena.

I seguenti argomenti forniscono ulteriori informazioni sulle autorizzazioni per aree specifiche di Athena.

Argomenti

- Consenti l'accesso a un connettore dati Athena per il metastore Hive esterno
- Consentire l'accesso alla funzione Lambda a metastore Hive esterni
- Esempio di policy di autorizzazione IAM per consentire la query federata Athena
- Esempio di policy di autorizzazione IAM per consentire le funzioni definite dall'utente (UDF) di Amazon Athena
- Autorizzazione per l'accesso per ML con Athena
- Abilitazione dell'accesso federato all'API Athena

AWS politiche gestite per Amazon Athena

Una politica AWS gestita è una politica autonoma creata e amministrata da. AWS AWS le politiche gestite sono progettate per fornire autorizzazioni per molti casi d'uso comuni, in modo da poter iniziare ad assegnare autorizzazioni a utenti, gruppi e ruoli.

Tieni presente che le policy AWS gestite potrebbero non concedere le autorizzazioni con il privilegio minimo per i tuoi casi d'uso specifici, poiché sono disponibili per tutti i clienti. AWS Ti consigliamo pertanto di ridurre ulteriormente le autorizzazioni definendo policy gestite dal cliente specifiche per i tuoi casi d'uso.

Non è possibile modificare le autorizzazioni definite nelle politiche gestite. AWS Se AWS aggiorna le autorizzazioni definite in una politica AWS gestita, l'aggiornamento ha effetto su tutte le identità principali (utenti, gruppi e ruoli) a cui è associata la politica. AWS è più probabile che aggiorni una policy AWS gestita quando ne Servizio AWS viene lanciata una nuova o quando diventano disponibili nuove operazioni API per i servizi esistenti.

Per ulteriori informazioni, consultare Policy gestite da AWSnella Guida per l'utente di IAM.

Considerazioni sull'utilizzo di policy gestite con Athena

Le policy gestite sono facili da utilizzare e vengono aggiornate automaticamente con le operazioni richieste con l'evolvere del servizio. Quando utilizzi le policy gestite con Athena, tieni presente quanto segue:

- Per consentire o negare operazioni di servizio Amazon Athena per te stesso o altri utenti che utilizzano AWS Identity and Access Management (IAM), devi allegare le policy basate su identità ai principali come utenti o gruppi.
- Ogni policy basata su identità è formata da istruzioni che definiscono le operazioni consentite o negate. Per ulteriori informazioni e step-by-step istruzioni su come allegare una policy a un utente,

consulta Allegare policy gestite nella IAM User Guide. Per un elenco delle operazioni, consulta la documentazione di riferimento dell'API di Amazon Athena.

- Le policy basate sulle identità gestite dal cliente e inline permettono di specificare operazioni Athena più dettagliate all'interno di una policy per ottimizzare l'accesso. Consigliamo di utilizzare la policy `AmazonAthenaFullAccess` come punto di partenza e successivamente concedere o negare operazioni specifiche elencate nella documentazione di riferimento dell'API di Amazon Athena. Per ulteriori informazioni sulle policy inline, consulta Policy gestite e policy inline nella Guida per l'utente di IAM.
- Se disponi anche di principali che si collegano utilizzando JDBC, devi fornire le credenziali del driver JDBC all'applicazione. Per ulteriori informazioni, consulta Accesso tramite connessioni JDBC e ODBC.
- Se hai crittografato il AWS Glue Data Catalog, devi specificare azioni aggiuntive nelle policy IAM basate sull'identità per Athena. Per ulteriori informazioni, consulta Accesso da Athena ai metadati crittografati nel AWS Glue Data Catalog.
- Se crei e usi i gruppi di lavoro, verifica che le tue policy includano l'accesso rilevante alle operazioni dei gruppi di lavoro. Per informazioni dettagliate, consulta the section called “ Policy IAM per l'accesso ai gruppi di lavoro” e the section called “Esempi di policy per i gruppi di lavoro”.

AWS politica gestita: AmazonAthenaFullAccess

La policy gestita `AmazonAthenaFullAccess` concede accesso completo ad Athena.

Per fornire l'accesso, aggiungi autorizzazioni ai tuoi utenti, gruppi o ruoli:

- Utenti e gruppi in AWS IAM Identity Center:

  Crea un set di autorizzazioni. Segui le istruzioni riportate nella pagina Create a permission set (Creazione di un set di autorizzazioni) nella Guida per l'utente di AWS IAM Identity Center .
- Utenti gestiti in IAM tramite un provider di identità:

  Crea un ruolo per la federazione delle identità. Segui le istruzioni riportate nella pagina Creating a role for a third-party identity provider (federation) (Creazione di un ruolo per un provider di identità di terze parti [federazione]) nella Guida per l'utente di IAM.
- Utenti IAM:
  - Crea un ruolo che l'utente possa assumere. Per istruzioni, consulta la pagina Creating a role for an IAM user (Creazione di un ruolo per un utente IAM) nella Guida per l'utente di IAM.

- (Non consigliato) Collega una policy direttamente a un utente o aggiungi un utente a un gruppo di utenti. Segui le istruzioni riportate nella pagina Aggiunta di autorizzazioni a un utente (console) nella Guida per l'utente di IAM.

Raggruppamenti di autorizzazioni

La policy `AmazonAthenaFullAccess` è raggruppata nei seguenti set di autorizzazioni.

- **`athena`**: consente ai principali di accedere alle risorse Athena.
- **`glue`**— Consente ai principali l'accesso a AWS Glue database, tabelle e partizioni. Ciò è necessario affinché il preside possa utilizzarlo AWS Glue Data Catalog con Athena.
- **`s3`**: consente al principale di scrivere e leggere i risultati delle query da Amazon S3, di leggere esempi di dati Athena disponibili pubblicamente che risiedono in Amazon S3 e di elencare i bucket. Questo è necessario in modo che il principale possa utilizzare Athena per lavorare con Amazon S3.
- **`sns`**: consente ai principali di elencare gli argomenti di Amazon SNS e ottenere gli attributi degli argomenti. Ciò consente ai responsabili di utilizzare gli argomenti Amazon SNS con Athena a scopo di monitoraggio e avviso.
- **`cloudwatch`**— Consente ai responsabili di creare, leggere ed eliminare CloudWatch allarmi. Per ulteriori informazioni, consulta Controllo dei costi e monitoraggio delle interrogazioni con metriche ed eventi CloudWatch .
- **`lakeformation`**: consente ai principali di richiedere credenziali temporanee per accedere ai dati in una posizione data lake registrata con Lake Formation. Per ulteriori informazioni, consulta Controlli dell'accesso ai dati sottostantinella Guida per sviluppatori AWS di Lake Formation.
- **`datazone`**— Consente ai mandanti di elencare DataZone progetti, domini e ambienti Amazon. Per informazioni sull'utilizzo DataZone in Athena, vedere. Utilizzo di Amazon DataZone in Athena
- **`pricing`**— Fornisce l'accesso a. AWS Billing and Cost Management Per ulteriori informazioni, GetProductsconsulta l'AWS Billing and Cost Management API Reference.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "BaseAthenaPermissions",
            "Effect": "Allow",
            "Action": [
                "athena:*"
```

```
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "BaseGluePermissions",
            "Effect": "Allow",
            "Action": [
                "glue:CreateDatabase",
                "glue:DeleteDatabase",
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:UpdateDatabase",
                "glue:CreateTable",
                "glue:DeleteTable",
                "glue:BatchDeleteTable",
                "glue:UpdateTable",
                "glue:GetTable",
                "glue:GetTables",
                "glue:BatchCreatePartition",
                "glue:CreatePartition",
                "glue:DeletePartition",
                "glue:BatchDeletePartition",
                "glue:UpdatePartition",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:BatchGetPartition",
                "glue:StartColumnStatisticsTaskRun",
                "glue:GetColumnStatisticsTaskRun",
                "glue:GetColumnStatisticsTaskRuns",
                "glue:GetCatalogImportStatus"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "BaseQueryResultsPermissions",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
```

```
                "s3:ListBucketMultipartUploads",
                "s3:ListMultipartUploadParts",
                "s3:AbortMultipartUpload",
                "s3:CreateBucket",
                "s3:PutObject",
                "s3:PutBucketPublicAccessBlock"
            ],
            "Resource": [
                "arn:aws:s3:::aws-athena-query-results-*"
            ]
        },
        {
            "Sid": "BaseAthenaExamplesPermissions",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::athena-examples*"
            ]
        },
        {
            "Sid": "BaseS3BucketPermissions",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketLocation",
                "s3:ListAllMyBuckets"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "BaseSNSPermissions",
            "Effect": "Allow",
            "Action": [
                "sns:ListTopics",
                "sns:GetTopicAttributes"
            ],
            "Resource": [
                "*"
            ]
```

```
        },
        {
            "Sid": "BaseCloudWatchPermissions",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:GetMetricData"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "BaseLakeFormationPermissions",
            "Effect": "Allow",
            "Action": [
                "lakeformation:GetDataAccess"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "BaseDataZonePermissions",
            "Effect": "Allow",
            "Action": [
                "datazone:ListDomains",
                "datazone:ListProjects",
                "datazone:ListAccountEnvironments"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Sid": "BasePricingPermissions",
            "Effect": "Allow",
            "Action": [
                "pricing:GetProducts"
            ],
            "Resource": [
                "*"
```

```
            ]
        }
    ]
}
```

AWS politica gestita: AWSQuicksightAthenaAccess

`AWSQuicksightAthenaAccess`concede l'accesso alle azioni QuickSight richieste da Amazon per l'integrazione con Athena. È possibile allegare la policy `AWSQuicksightAthenaAccess`alle identità IAM. Allega questa politica solo ai mandanti che utilizzano Amazon QuickSight con Athena. Questa policy include alcune operazioni per Athena che sono obsolete e non incluse nell'API pubblica corrente oppure che vengono utilizzate solo con i driver JDBC e ODBC.

Raggruppamenti di autorizzazioni

La policy `AWSQuicksightAthenaAccess` è raggruppata nei seguenti set di autorizzazioni.

- **`athena`**: consente al principale di eseguire query sulle risorse Athena.
- **`glue`**— Consente ai principali di accedere a AWS Glue database, tabelle e partizioni. Ciò è necessario affinché il preside possa utilizzarlo AWS Glue Data Catalog con Athena.
- **`s3`**: consente al principale di scrivere e leggere i risultati delle query da Amazon S3.
- **`lakeformation`**: consente ai principali di richiedere credenziali temporanee per accedere ai dati in una posizione data lake registrata con Lake Formation. Per ulteriori informazioni, consulta Controlli dell'accesso ai dati sottostantinella Guida per sviluppatori AWS di Lake Formation.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:BatchGetQueryExecution",
                "athena:GetQueryExecution",
                "athena:GetQueryResults",
                "athena:GetQueryResultsStream",
                "athena:ListQueryExecutions",
                "athena:StartQueryExecution",
                "athena:StopQueryExecution",
                "athena:ListWorkGroups",
                "athena:ListEngineVersions",
```

```
                "athena:GetWorkGroup",
                "athena:GetDataCatalog",
                "athena:GetDatabase",
                "athena:GetTableMetadata",
                "athena:ListDataCatalogs",
                "athena:ListDatabases",
                "athena:ListTableMetadata"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "glue:CreateDatabase",
                "glue:DeleteDatabase",
                "glue:GetDatabase",
                "glue:GetDatabases",
                "glue:UpdateDatabase",
                "glue:CreateTable",
                "glue:DeleteTable",
                "glue:BatchDeleteTable",
                "glue:UpdateTable",
                "glue:GetTable",
                "glue:GetTables",
                "glue:BatchCreatePartition",
                "glue:CreatePartition",
                "glue:DeletePartition",
                "glue:BatchDeletePartition",
                "glue:UpdatePartition",
                "glue:GetPartition",
                "glue:GetPartitions",
                "glue:BatchGetPartition"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
```

```
                "s3:ListBucket",
                "s3:ListBucketMultipartUploads",
                "s3:ListMultipartUploadParts",
                "s3:AbortMultipartUpload",
                "s3:CreateBucket",
                "s3:PutObject",
                "s3:PutBucketPublicAccessBlock"
            ],
            "Resource": [
                "arn:aws:s3:::aws-athena-query-results-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "lakeformation:GetDataAccess"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Athena aggiorna le policy gestite AWS

Visualizza i dettagli sugli aggiornamenti delle politiche AWS gestite per Athena da quando questo servizio ha iniziato a tenere traccia di queste modifiche.

Modifica	Descrizione	Data
AmazonAthenaFullAccess: aggiornamento a policy esistente	Consente ad Athena di utilizzare l' AWS Glue `GetCatalogImportStatus` API documentata pubblicamente per recuperare lo stato di importazione del catalogo.	18 giugno 2024
AmazonAthenaFullAccess: aggiornamento a policy esistente	Sono state aggiunte `datazone:ListAccountEnvironments` le autorizzazioni	3 gennaio 2024

Modifica	Descrizione	Data
	`datazone:ListDomains datazone:ListProjects` , e per consentire agli utenti di Athena di lavorare con domini, progetti e ambienti DataZone Amazon. Per ulteriori informazioni, consulta Utilizzo di Amazon DataZone in Athena.	
AmazonAthenaFullAccess: aggiornamento a policy esistente	Le `glue:GetColumnStatisticsTaskRuns` autorizzazioni `glue:StartColumnStatisticsTaskRun glue:GetColumnStatisticsTaskRun` , e sono state aggiunte per dare ad Athena il diritto di AWS Glue chiamare per recuperare le statistiche per la funzionalità di ottimizzazione basata sui costi. Per ulteriori informazioni, consulta Utilizzo dell'ottimizzatore basato sui costi.	3 gennaio 2024
AmazonAthenaFullAccess: aggiornamento a policy esistente	Athena ha aggiunto `pricing:GetProducts` per fornire l'accesso a AWS Billing and Cost Management. Per ulteriori informazioni, consulta GetProducts l'API Reference.AWS Billing and Cost Management	25 gennaio 2023

Modifica	Descrizione	Data
AmazonAthenaFullAccess: aggiornamento a policy esistente	Athena aggiunta `cloudwatch:GetMetricData` per recuperare i valori delle metriche CloudWatch. Per ulteriori informazioni, GetMetricDataconsulta Amazon CloudWatch API Reference.	14 novembre 2022
AmazonAthenaFullAccesse AWSQuicksightAthenaAccess— Aggiornamenti alle politiche esistenti	Athena ha aggiunto `s3:PutBucketPublicAccessBlock` per consentire il blocco dell'accesso pubblico sui bucket creati da Athena.	7 luglio 2021
Athena ha iniziato a monitorare le modifiche	Athena ha iniziato a tenere traccia delle modifiche per le sue politiche AWS gestite.	7 luglio 2021

Accesso tramite connessioni JDBC e ODBC

Per accedere a risorse Servizi AWS e risorse, come Athena e i bucket Amazon S3, fornisci le credenziali del driver JDBC o ODBC all'applicazione. Se stai utilizzando il driver JDBC oppure ODBC, accertati che la policy delle autorizzazioni IAM includa tutte le operazioni elencate in AWS politica gestita: AWSQuicksightAthenaAccess.

Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Metodi di autenticazione

I driver Athena JDBC e ODBC supportano l'autenticazione basata su SAML 2.0, inclusi i seguenti provider di identità:

- Active Directory Federation Services (AD FS)
- Azure Active Directory (AD)
- Okta

- PingFederate

Per ulteriori informazioni, consulta le guide all'installazione e alla configurazione dei rispettivi driver, scaricabili in formato PDF dalle pagine dei driver JDBC e ODBC. Per ulteriori informazioni correlate, consulta la seguente documentazione:

- Abilitazione dell'accesso federato all'API Athena
- Utilizzo di Lake Formation e dei driver Athena JDBC e ODBC per l'accesso federato ad Athena
- Configurazione di Single Sign-On tramite ODBC, SAML 2.0 e il provider di identità Okta

Per informazioni sulle versioni più recenti dei driver JDBC e ODBC e la relativa documentazione, vedere Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

Accesso ad Amazon S3 da Athena

Puoi concedere l'accesso alle posizioni Amazon S3 utilizzando le policy basate sulle identità, le policy delle risorse bucket, le policy dei punti di accesso o qualsiasi combinazione delle policy qui sopra. Quando gli attori interagiscono con Athena, le loro autorizzazioni passano attraverso Athena per determinare ciò a cui Athena può accedere. Ciò significa che gli utenti devono essere autorizzati ad accedere ai bucket Amazon S3 per potervi eseguire query con Athena.

Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Tieni presente che le richieste ad Amazon S3 provengono da un indirizzo IPv4 privato per Athena, non dall'IP di origine specificato in. `aws:SourceIp` Per questo motivo, non è possibile utilizzare la `aws:SourceIp` condizione per negare l'accesso alle azioni di Amazon S3 in una determinata policy IAM. Inoltre, non puoi limitare o consentire l'accesso alle risorse Amazon S3 in base alle chiavi di `aws:SourceVpce` condizione `aws:SourceVpc` o.

> **Note**
>
> I gruppi di lavoro Athena che utilizzano l'autenticazione Centro identità IAM richiedono che S3 Access Grants sia configurato per l'utilizzo delle identità con propagazione delle identità attendibili. Per maggiori informazioni, consulta la pagina S3 Access Grants and directory identities nella Guida per l'utente di Amazon Simple Storage Service.

Argomenti

Controllo dell'accesso ai bucket Amazon S3 con policy basate sull'identità

Le policy basate su identità sono collegate a un utente, un gruppo o un ruolo IAM. Queste policy consentono di specificare cosa può fare quell'identità (le sue autorizzazioni). Puoi utilizzare policy basate sull'identità per controllare l'accesso ai tuoi bucket Amazon S3.

La seguente politica basata sull'identità consente `Read` `Write` l'accesso agli oggetti in uno specifico bucket Amazon S3. Per utilizzare questa politica, sostituisci il testo segnaposto in *`corsivo con i tuoi valori.`*

```
{
"Version": "2012-10-17",
"Statement":
    [
        {
            "Sid": "ListObjectsInBucket",
            "Effect": "Allow",
            "Action": ["s3:ListBucket"],
            "Resource":
                ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource":
                ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"]
        }
    ]
}
```

Controllo dell'accesso ai bucket Amazon S3 con policy relative alle risorse dei bucket

Puoi utilizzare le policy dei bucket di Amazon S3 per proteggere l'accesso agli oggetti nei tuoi bucket in modo che solo gli utenti con le autorizzazioni appropriate possano accedervi. Per indicazioni sulla creazione di una policy Amazon S3, consulta Aggiungere una bucket policy utilizzando la console Amazon S3 nella Amazon S3 User Guide.

Il seguente esempio di politica di autorizzazione limita un utente alla lettura di oggetti che hanno il `environment: production` tag key e value. La politica di esempio utilizza la chiave `s3:ExistingObjectTag` condition per specificare la chiave e il valore del tag.

```
{
    "Version":"2012-10-17",
    "Statement":
    [
        {
            "Principal":{"AWS":"arn:aws:iam::111122223333:role/JohnDoe"
        },
            "Effect":"Allow",
            "Action": [ "s3:GetObject", "s3:GetObjectVersion" ],
            "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "Condition":
                {
                    "StringEquals":{"s3:ExistingObjectTag/environment":"production"
                }
    ]
}
```

Per altri esempi di policy relative ai bucket, consulta Esempi di policy relative ai bucket di Amazon S3 nella Amazon S3 User Guide.

Punti di accesso Amazon S3, alias dei punti di accesso e politiche dei punti di accesso

Se disponi di un set di dati condiviso in un bucket Amazon S3, mantenere una singola policy del bucket che gestisce l'accesso per centinaia di casi d'uso può essere difficile.

I punti di accesso al bucket Amazon S3 aiutano a risolvere questo problema. Un bucket può avere più punti di accesso, ciascuno con una policy che controlla l'accesso al bucket in modo diverso.

Per ogni punto di accesso creato, Amazon S3 genera un alias che rappresenta il punto di accesso. Poiché l'alias è nel formato del nome bucket Amazon S3, è possibile utilizzare l'alias nella clausola

`LOCATION` delle istruzioni `CREATE TABLE` in Athena. L'accesso di Athena al bucket è quindi controllato dalla policy per il punto di accesso che l'alias rappresenta.

Per ulteriori informazioni, consulta Posizione delle tabelle in Amazon S3 e Utilizzo dei punti di accesso nella Guida per l'utente di Amazon S3.

Utilizzo delle chiavi CalledVia contestuali

Per una maggiore sicurezza, è possibile utilizzare la chiave di contesto di condizione globale aws:CalledVia. La chiave `aws:CalledVia` contiene un elenco ordinato di ciascun servizio nella catena che ha effettuato le richieste per conto dell'entità principale. Specificando il nome del principale servizio Athena `athena.amazonaws.com` per la chiave di contesto `aws:CalledVia`, è possibile limitare le richieste solo a quelle effettuate da Athena. Per ulteriori informazioni, consulta Usare Athena con CalledVia le chiavi di contesto.

Altre risorse

Per ulteriori informazioni ed esempi su come concedere l'accesso ad Amazon S3, consulta le seguenti risorse:

- Procedure guidate di esempio: gestione degli accessi nella Guida per l'utente di Amazon S3.
- In che modo posso fornire l'accesso su più account agli oggetti che si trovano nei bucket Amazon S3? nel Knowledge Center. AWS
- Accesso tra account in Athena a bucket Amazon S3.

Accesso tra account in Athena a bucket Amazon S3

Uno scenario comune di Amazon Athena è la concessione dell'accesso agli utenti in un account diverso da quello del proprietario del bucket, in modo che possano eseguire le query. In questo caso, occorre utilizzare una policy dei bucket per concedere l'accesso.

> **Note**
>
> Per informazioni sull'accesso tra account diversi ai cataloghi di AWS Glue dati di Athena, vedere. Accesso tra account ai cataloghi dati AWS Glue

La seguente policy dei bucket di esempio, creata e applicata al bucket `s3://DOC-EXAMPLE-BUCKET` dal proprietario del bucket, concede l'accesso a tutti gli utenti nell'account `123456789123`, che è un account diverso.

```
{
   "Version": "2012-10-17",
   "Id": "MyPolicyID",
   "Statement": [
      {
          "Sid": "MyStatementSid",
          "Effect": "Allow",
          "Principal": {
             "AWS": "arn:aws:iam::123456789123:root"
          },
          "Action": [
             "s3:GetBucketLocation",
             "s3:GetObject",
             "s3:ListBucket",
             "s3:ListBucketMultipartUploads",
             "s3:ListMultipartUploadParts",
             "s3:AbortMultipartUpload"
          ],
          "Resource": [
             "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
             "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
          ]
       }
    ]
 }
```

Per concedere l'accesso a un determinato utente in un account, occorre sostituire la chiave `Principal` con una chiave che specifica l'utente anziché `root`. Ad esempio, per il profilo utente `Dave`, è necessario utilizzare `arn:aws:iam::123456789123:user/Dave`.

Accesso da più account a un bucket crittografato con una chiave personalizzata AWS KMS

Se disponi di un bucket Amazon S3 crittografato con una chiave personalizzata AWS Key Management Service (AWS KMS), potresti dover concedere l'accesso ad esso agli utenti di un altro account Amazon Web Services.

La concessione dell'accesso a un bucket AWS KMS crittografato nell'Account A a un utente nell'Account B richiede le seguenti autorizzazioni:

- La policy del bucket nell'Account A deve concedere l'accesso al ruolo assunto dall'Account B.
- La politica AWS KMS chiave dell'Account A deve concedere l'accesso al ruolo assunto dall'utente nell'Account B.
- Il ruolo AWS Identity and Access Management (IAM) assunto dall'Account B deve concedere l'accesso sia al bucket che alla chiave nell'Account A.

Nelle procedure seguenti viene descritto come concedere ciascuna di queste autorizzazioni.

Per concedere l'accesso al bucket nell'Account A all'utente nell'Account B

- Nell'Account A, esaminare la policy del bucket S3 e verificare che sia presente un'istruzione che consenta l'accesso dall'ID dell'Account B.

  Ad esempio, la seguente policy del bucket consente l'accesso `s3:GetObject` all'ID `111122223333` dell'account:

```
{
  "Id": "ExamplePolicy1",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExampleStmt1",
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
      "Principal": {
        "AWS": [
          "111122223333"
        ]
      }
    }
  ]
}
```

Per concedere l'accesso all'utente nell'account b dalla politica AWS KMS chiave nell'account a

1. Nella politica AWS KMS chiave per l'Account A, concedi il ruolo assunto dalle autorizzazioni dell'Account B alle seguenti azioni:

   - `kms:Encrypt`
   - `kms:Decrypt`
   - `kms:ReEncrypt*`
   - `kms:GenerateDataKey*`
   - `kms:DescribeKey`

   Nell'esempio seguente viene concesso l'accesso alla chiave a un solo ruolo IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowUseOfTheKey",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/role_name"
            },
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        }
    ]
}
```

2. Dall'Account A, esamina la politica chiave utilizzando la visualizzazione delle AWS Management Console politiche.
3. Nella policy della chiave, verificare che l'istruzione seguente elenchi l'Account B come principal.

```
"Sid": "Allow use of the key"
```

4. Se l'istruzione `"Sid": "Allow use of the key"` non è presente, attenersi alla seguente procedura:

   a. Visualizzare la policy della chiave utilizzando la visualizzazione predefinita della console.

   b. Aggiungere l'ID dell'Account B come account esterno con accesso alla chiave.

Concessione dell'accesso al bucket e alla chiave nell'Account A dal ruolo IAM assunto dall'Account B

1. Dall'Account B, apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Apri il ruolo IAM associato all'utente nell'Account B.
3. Esamina l'elenco delle policy di autorizzazione applicate al ruolo IAM.
4. Assicurarsi che venga applicata una policy che conceda l'accesso al bucket.

   L'istruzione di esempio seguente concede al ruolo IAM l'accesso alle operazioni `s3:GetObject` e `s3:PutObject` sul bucket `DOC-EXAMPLE-BUCKET`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExampleStmt2",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
    }
  ]
}
```

5. Assicurarsi che venga applicata una policy che consenta l'accesso alla chiave.

> **Note**
>
> Se il ruolo IAM assunto dall'Account B dispone già dell'accesso come amministratore, non è necessario concedere l'accesso alla chiave dalle policy IAM dell'utente.

L'istruzione di esempio seguente concede al ruolo IAM l'accesso per utilizzare la chiave `arn:aws:kms:us-west-2:123456789098:key/111aa2bb-333c-4d44-5555-a111bb2c33dd`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExampleStmt3",
      "Action": [
        "kms:Decrypt",
        "kms:DescribeKey",
        "kms:Encrypt",
        "kms:GenerateDataKey",
        "kms:ReEncrypt*"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:kms:us-west-2:123456789098:key/111aa2bb-333c-4d44-5555-a111bb2c33dd"
    }
  ]
}
```

Accesso tra account agli oggetti del bucket

Gli oggetti caricati da un account (Account C) diverso dall'account proprietario del bucket (Account A) potrebbero richiedere ACL esplicite a livello di oggetto che concedano l'accesso in lettura all'account di query (Account B). Per evitare questo requisito, l'Account C deve assumere un ruolo nell'Account A prima di collocare gli oggetti nel bucket dell'Account A. Per ulteriori informazioni, consulta Come posso fornire l'accesso tra account agli oggetti che si trovano nei bucket Amazon S3?.

Accesso granulare a database e tabelle in AWS Glue Data Catalog

Se utilizzi Amazon Athena, puoi definire politiche a livello di risorsa per gli oggetti Data Catalog di database e tabelle utilizzati in Athena. AWS Glue Data Catalog

> **Note**
>
> Il termine "controllo granulare degli accessi" si riferisce qui alla sicurezza a livello di database e tabella. Per informazioni sulla sicurezza a livello di colonna, riga e cella, consulta Data filtering and cell-level security in Lake Formation (Filtraggio dei dati e sicurezza a livello di cella in Lake Formation).

Puoi definire autorizzazioni a livello di risorse nelle policy IAM basate sulle identità.

> **Important**
>
> Questa sezione illustra le autorizzazioni a livello di risorsa nelle policy IAM basate su identità. Queste sono diverse dalle policy basate sulle risorse. Per ulteriori informazioni sulle differenze, consulta Policy basate sulle identità e policy basate su risorse nella Guida per l'utente di IAM.

Consulta i seguenti argomenti per queste operazioni:

Per eseguire questa operazione	Consulta l'argomento seguente
Creare una policy IAM che definisca l'accesso granulare alle risorse	Creazione di policy IAM nella Guida per l'utente di IAM.
Scopri le politiche basate sull'identità IAM utilizzate in AWS Glue	Policy basate sulle identità (policy IAM) nella Guida per lo sviluppatore di AWS Glue .

In questa sezione

Limitazioni

Considera le seguenti restrizioni quando utilizzi il controllo degli accessi granulare con AWS Glue Data Catalog e Athena:

- I gruppi di lavoro Athena abilitati per Centro identità IAM richiedono che Lake Formation sia configurato per l'utilizzo delle identità di Centro identità IAM. Per ulteriori informazioni, consulta la pagina Integrating IAM Identity Center nella Guida per gli sviluppatori di AWS Lake Formation .
- È possibile limitare l'accesso solo a database e tabelle. I controlli degli accessi granulari si applicano a livello di tabella e non è possibile limitare l'accesso a singole partizioni all'interno di una tabella. Per ulteriori informazioni, consulta Partizioni e versioni delle tabelle in AWS Glue.
- AWS Glue Data Catalog Contiene le seguenti risorse:`CATALOG`,`DATABASE`, `TABLE` e. `FUNCTION`

  > **Note**
  >
  > In questo elenco, le risorse comuni tra Athena e il AWS Glue Data Catalog sono `TABLE`DATABASE, e `CATALOG` per ogni account. `Function`è specifico per. AWS Glue Per operazioni di eliminazione in Athena, è necessario includere le autorizzazioni per le operazioni AWS Glue . Per informazioni, consulta Esempi di autorizzazioni granulari per tabelle e database.

  La gerarchia è la seguente: `CATALOG` è un predecessore di tutti i `DATABASES` in ogni account e ogni `DATABASE` è un predecessore per tutte le relative `TABLES` e `FUNCTIONS`. Ad esempio, per una tabella denominata `table_test` che appartiene a un database db nel catalogo nell'account, i suoi predecessori sono db e il catalogo nell'account. Per il database db, il predecessore è il catalogo nell'account e i relativi discendenti sono le tabelle e le funzioni. Per ulteriori informazioni sulla struttura gerarchica delle risorse, consulta la sezione relativa all'elenco degli ARN nel catalogo dati nella Guida per lo sviluppatore di AWS Glue .
- Per operazioni Athena di non eliminazione su una risorsa, come `CREATE DATABASE`, `CREATE TABLE`, `SHOW DATABASE`, `SHOW TABLE` o `ALTER TABLE`, hai bisogno delle autorizzazioni per chiamare tali operazioni sulla risorsa (tabella o database) e su tutti i predecessori della risorsa nel Catalogo dati. Ad esempio, per una tabella, i suoi predecessori sono i database di appartenenza e il catalogo dell'account. Per un database, il predecessore è il catalogo per questo account. Per informazioni, consulta Esempi di autorizzazioni granulari per tabelle e database.
- Per un'operazione di eliminazione in Athena, ad esempio `DROP DATABASE` o `DROP TABLE`, è necessaria anche l'autorizzazione a richiamare l'operazione di eliminazione su tutti i predecessori

e discendenti della risorsa nel catalogo dati. Ad esempio, per eliminare un database è necessario disporre di autorizzazioni per il database, il catalogo, che è il suo predecessore e tutte le tabelle e le funzioni definite dall'utente, che sono i discendenti. Una tabella non ha discendenti. Per eseguire `DROP TABLE`, è necessario disporre dell'autorizzazione per questa operazione sulla tabella, il database di appartenenza e il catalogo. Per informazioni, consulta Esempi di autorizzazioni granulari per tabelle e database.

AWS Glue accesso al catalogo e al database per Regione AWS

Affinché Athena funzioni con AWS Glue, è necessaria una politica che garantisca l'accesso al tuo database e AWS Glue Data Catalog al tuo account per Regione AWS . Per creare database, è richiesta anche l'autorizzazione `CreateDatabase`. Nel seguente esempio di policy, sostituisci l' Regione AWS Account AWS ID e il nome del database con quelli tuoi.

```
{
   "Sid": "DatabasePermissions",
   "Effect": "Allow",
   "Action": [
      "glue:GetDatabase",
      "glue:GetDatabases",
      "glue:CreateDatabase"
   ],
   "Resource": [
     "arn:aws:glue:us-east-1:123456789012:catalog",
     "arn:aws:glue:us-east-1:123456789012:database/default"
   ]
}
```

Partizioni e versioni delle tabelle in AWS Glue

In AWS Glue, le tabelle possono avere partizioni e versioni. Le versioni e le partizioni delle tabelle non sono considerate risorse indipendenti in. AWS Glue L'accesso alle versioni e alle partizioni delle tabelle è determinato dalla concessione dell'accesso nella tabella e nelle risorse predecessore per la tabella.

Ai fini del controllo granulare degli accessi vengono applicate le seguenti autorizzazioni di accesso:

- I controlli degli accessi granulari si applicano a livello di tabella. È possibile limitare l'accesso solo a database e tabelle. Ad esempio, se consenti l'accesso a una tabella partizionata, tale accesso si

applica a tutte le partizioni della tabella. Non puoi limitare l'accesso a singole partizioni all'interno di una tabella.

> **Important**
>
> Per eseguire azioni AWS Glue sulle partizioni, sono necessarie le autorizzazioni per le azioni di partizione a livello di catalogo, database e tabella. L'accesso alle partizioni all'interno di una tabella non è sufficiente. Ad esempio, per l'esecuzione di `GetPartitions` su una tabella `myTable` nel database `myDB`, è necessario concedere le autorizzazioni a `glue:GetPartitions` per il catalogo, il database `myDB` e le risorse `myTable`.

- I controlli degli accessi granulari non si applicano alle versioni delle tabelle. Come per le partizioni, l'accesso alle versioni precedenti di una tabella viene concesso tramite l'accesso alle API della versione di tabella AWS Glue presenti nella tabella e ai predecessori della tabella.

Per informazioni sulle AWS Glue autorizzazioni sulle azioni, consulta Autorizzazioni AWS Glue API: riferimento alle azioni e alle risorse nella Guida per gli sviluppatori.AWS Glue

Esempi di autorizzazioni granulari per tabelle e database

La tabella seguente elenca esempi di policy IAM basate sulle identità che consentono l'accesso granulare ai database e alle tabelle in Athena. Consigliamo di iniziare con questi esempi e, in base alle esigenze, regolarli per consentire o negare operazioni specifiche a particolari database e tabelle.

Questi esempi includono l'accesso a database e cataloghi in modo che Athena AWS Glue e io possiamo lavorare insieme. Per più AWS regioni, includi politiche simili per ciascuno dei tuoi database e cataloghi, una riga per ogni regione.

In questi esempi, sostituisci il database `example_db` e la tabella `test` con i nomi dei tuoi database e tabelle.

Istruzione DDL	Esempio di policy di accesso IAM che concede l'accesso alla risorsa

ALTER DATABASE	Consente di modificare le proprietà del database `example_db` .
	
	`{`
	`"Effect": "Allow",`
	`"Action": [`

Istruzione DDL	Esempio di policy di accesso IAM che concede l'accesso alla risorsa
	"glue:GetDatabase", "glue:UpdateDatabase"], "Resource": ["arn:aws:glue: *us-east-1* :*123456789012* :catalog", "arn:aws:glue: *us-east-1* :*123456789012* :database / *example_db* "] }

CREATE DATABASE	Consente di creare il database denominato `example_db` .
	{
	"Effect": "Allow",
	"Action": [
	"glue:GetDatabase",
	"glue:CreateDatabase"
	],
	"Resource": [
	"arn:aws:glue: *us-east-1* :*123456789012* :catalog",
	"arn:aws:glue: *us-east-1* :*123456789012* :database
	/ *example_db* "
	]
	}

Istruzione DDL	Esempio di policy di accesso IAM che concede l'accesso alla risorsa

CREATE TABLE	Consente di creare una tabella denominata `test` nel database `example_db` .
	
	`{`
	`"Sid": "DatabasePermissions",`
	`"Effect": "Allow",`
	`"Action": [`
	`"glue:GetDatabase",`
	`"glue:GetDatabases"`
	`],`
	`"Resource": [`
	`"arn:aws:glue: us-east-1 :123456789012 :catalog",`
	`"arn:aws:glue: us-east-1 :123456789012 :database / example_db "`
	`]`
	`},`
	`{`
	`"Sid": "TablePermissions",`
	`"Effect": "Allow",`
	`"Action": [`
	`"glue:GetTables",`
	`"glue:GetTable",`
	`"glue:GetPartitions",`
	`"glue:CreateTable"`
	`],`
	`"Resource": [`
	`"arn:aws:glue: us-east-1 :123456789012 :catalog",`
	`"arn:aws:glue: us-east-1 :123456789012 :database / example_db ",`
	`"arn:aws:glue: us-east-1 :123456789012 :table/example_db /test"`
	`]`
	`}`

Istruzione DDL	Esempio di policy di accesso IAM che concede l'accesso alla risorsa

DROP DATABASE	Consente di eliminare il database `example_db` , comprese tutte le tabelle in esso contenute.
	
	`{`
	`"Effect": "Allow",`
	`"Action": [`
	`"glue:GetDatabase",`
	`"glue:DeleteDatabase",`
	`"glue:GetTables",`
	`"glue:GetTable",`
	`"glue:DeleteTable"`
	`],`
	`"Resource": [`
	`"arn:aws:glue: us-east-1 :123456789012 :catalog",`
	`"arn:aws:glue: us-east-1 :123456789012 :database / example_db ",`
	`"arn:aws:glue: us-east-1 :123456789012 :table/example_db /*",`
	`"arn:aws:glue: us-east-1 :123456789012 :userDefinedFunction/ example_db /*"`
	`]`
	`}`

Istruzione DDL	Esempio di policy di accesso IAM che concede l'accesso alla risorsa

DROP TABLE	Consente di eliminare una tabella partizionata denominata `test` nel database `example_db` . Se la tabella non ha partizioni, non occorre includere operazioni di partizione.
	
	{
	"Effect": "Allow",
	"Action": [
	"glue:GetDatabase",
	"glue:GetTable",
	"glue:DeleteTable",
	"glue:GetPartitions",
	"glue:GetPartition",
	"glue:DeletePartition"
	],
	"Resource": [
	"arn:aws:glue: *us-east-1* :*123456789012* :catalog",
	"arn:aws:glue: *us-east-1* :*123456789012* :database / *example_db* ",
	"arn:aws:glue: *us-east-1* :*123456789012* :table/*example_db* /*test*"
	]
	}

Istruzione DDL	Esempio di policy di accesso IAM che concede l'accesso alla risorsa
MSCK REPAIR TABLE	Consente di aggiornare i metadati del catalogo dopo aver aggiunto le partizioni compatibili con Hive alla tabella denominata `test` nel database `example_db` . `{ "Effect": "Allow", "Action": ["glue:GetDatabase", "glue:CreateDatabase", "glue:GetTable", "glue:GetPartitions", "glue:GetPartition", "glue:BatchCreatePartition"], "Resource": ["arn:aws:glue: us-east-1 :123456789012 :catalog", "arn:aws:glue: us-east-1 :123456789012 :database / example_db ", "arn:aws:glue: us-east-1 :123456789 012 :table/example_db /test"] }`
SHOW DATABASES	Consente di elencare tutti i database nel AWS Glue Data Catalog. `{ "Effect": "Allow", "Action": [ "glue:GetDatabase", "glue:GetDatabases" ], "Resource": [ "arn:aws:glue: us-east-1 :123456789012 :catalog", "arn:aws:glue: us-east-1 :123456789012 :database/*" ] }`

Istruzione DDL	Esempio di policy di accesso IAM che concede l'accesso alla risorsa
SHOW TABLES	Consente di elencare tutte le tabelle nel database `example_db` . `{ "Effect": "Allow", "Action": [ "glue:GetDatabase", "glue:GetTables" ], "Resource": [ "arn:aws:glue: us-east-1 :123456789012 :catalog", "arn:aws:glue: us-east-1 :123456789012 :database / example_db ", "arn:aws:glue: us-east-1 :123456789012 :table/example_db /*" ] }`

Accesso tra account ai cataloghi dati AWS Glue

Puoi utilizzare la funzionalità di AWS Glue catalogo per più account di Athena per registrare un AWS Glue catalogo da un account diverso dal tuo. Dopo aver configurato le autorizzazioni IAM richieste per AWS Glue e aver registrato il catalogo come risorsa DataCatalogAthena, puoi utilizzare Athena per eseguire query tra account. Per informazioni sull'utilizzo della console Athena per registrare un catalogo da un altro account, consulta la sezione Registrazione e AWS Glue Data Catalog da un altro account.

Per ulteriori informazioni sull'accesso tra account in AWS Glue, consulta Garantire l'accesso su più account nella Developer Guide.AWS Glue

Prima di iniziare

Perché questa funzione utilizza API e funzionalità esistenti delle risorse `DataCatalog` di Athena per consentire l'accesso tra account, ti consigliamo di leggere le seguenti risorse prima di iniziare:

- Connessione alle origini dati- Contiene argomenti sull'utilizzo di Athena con sorgenti di AWS Glue cataloghi dati Hive o Lambda.

- Policy di esempio del catalogo dati: mostra come scrivere policy che controllano l'accesso ai cataloghi dati.
- Utilizzo di AWS CLI with Hive metastores- Mostra come utilizzare i metastore AWS CLI with Hive, ma contiene casi d'uso applicabili ad altre fonti di dati.

Considerazioni e limitazioni

Attualmente, l'accesso al AWS Glue catalogo tra più account di Athena presenta le seguenti limitazioni:

- La funzionalità è disponibile solo Regioni AWS laddove è supportata la versione 2 o successiva del motore Athena. Per ulteriori informazioni sulle versioni del motore Athena, consulta Controllo delle versioni del motore di Athena. Per aggiornare la versione del motore di un gruppo di lavoro, consulta Modifica delle versioni del motore Athena.
- Quando registri un altro account AWS Glue Data Catalog nel tuo account, crei una `DataCatalog` risorsa regionale collegata ai dati dell'altro account solo in quella particolare regione.
- Attualmente le istruzioni `CREATE VIEW` che includono un catalogo AWS Glue multi-account non sono supportate.
- I cataloghi crittografati utilizzando chiavi AWS gestite non possono essere interrogati su più account. Per i cataloghi da interrogare su più account, utilizza invece le chiavi gestite dai clienti (). `KMS_CMK` Per informazioni sulle differenze tra chiavi gestite dal cliente e chiavi AWS gestite, consulta Customer keys and AWS keys nella AWS Key Management Service Developer Guide.

Nozioni di base

Nello scenario seguente, l'account «mutuatario» (88886666) desidera eseguire una `SELECT` query che si riferisce al AWS Glue catalogo che appartiene all'account «proprietario» (9999), come nell'esempio seguente:

```
SELECT * FROM ownerCatalog.tpch1000.customer
```

Nella procedura seguente, i passaggi 1a e 1b mostrano come concedere all'account del mutuatario l'accesso alle AWS Glue risorse dell'account proprietario, sia dal lato del mutuatario che dal lato del proprietario. L'esempio concede l'accesso al database `tpch1000` e alla tabella `customer`. Modifica questi nomi di esempio in base alle proprie esigenze.

Fase 1a: Creare un ruolo di mutuatario con una politica di accesso alle risorse del proprietario AWS Glue

Per creare un ruolo di account mutuatario con una politica di accesso alle AWS Glue risorse dell'account proprietario, puoi utilizzare la console AWS Identity and Access Management (IAM) o l'API IAM. La seguente procedura usa la console IAM.

Per creare un ruolo e una politica del mutuatario per accedere alle risorse dell'account proprietario AWS Glue

1. Accedi alla console IAM all'indirizzo https://console.aws.amazon.com/iam/ dall'account del mutuatario.
2. Nel riquadro di navigazione, espandi Gestione accesso, quindi seleziona Policy.
3. Scegli Crea policy.
4. Per Editor di policy, scegli JSON.
5. Nell'editor delle policy, inserisci la seguente policy, quindi modificala in base alle tue esigenze:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "glue:*",
            "Resource": [
                "arn:aws:glue:us-east-1:999999999999:catalog",
                "arn:aws:glue:us-east-1:999999999999:database/tpch1000",
                "arn:aws:glue:us-east-1:999999999999:table/tpch1000/customer"
            ]
        }
    ]
}
```

6. Seleziona Successivo.
7. Nella pagina Rivedi e crea, in Nome della politica, inserisci un nome per la politica (ad esempio,**`CrossGluePolicyForBorrowerRole`**).
8. Scegli Crea policy.
9. Nel pannello di navigazione, seleziona Roles (Ruoli).
10. Selezionare Create role (Crea ruolo).

11. Nella pagina Seleziona entità attendibile, scegli Account AWS, quindi scegli Avanti.
12. Nella pagina Aggiungi autorizzazioni, inserisci il nome della politica che hai creato nella casella di ricerca (ad esempio,**`CrossGluePolicyForBorrowerRole`**).
13. Seleziona la casella di controllo accanto al nome della politica, quindi scegli Avanti.
14. Nella pagina Name, review, and create (Nome, revisione e creazione), per Role name (Nome ruolo) inserisci un nome per il ruolo (ad esempio **`CrossGlueBorrowerRole`**).
15. Scegli Crea ruolo.

Fase 1b: Creare una politica relativa al proprietario per concedere AWS Glue l'accesso al mutuatario

Per concedere AWS Glue l'accesso dall'account del proprietario (³9999) al ruolo del mutuatario, puoi utilizzare la console o il funzionamento dell' AWS Glue API. AWS Glue PutResourcePolicy La procedura seguente utilizza la console. AWS Glue

Per concedere AWS Glue l'accesso al conto del mutuatario da parte del proprietario

1. Accedi alla AWS Glue console all'indirizzo https://console.aws.amazon.com/glue/ dall'account del proprietario.
2. Nel riquadro di navigazione, espandi Data Catalog, quindi seleziona Impostazioni catalogo.
3. Nel campo Autorizzazioni inserisci una policy simile alla seguente. Per *`rolename`*, inserisci il ruolo che il mutuatario ha creato nel passaggio 1a (ad esempio,). **`CrossGlueBorrowerRole`** Se desideri aumentare l'ambito delle autorizzazioni, puoi utilizzare il carattere jolly * sia per il database che per i tipi di risorse della tabella.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::666666666666:user/username",
                    "arn:aws:iam::666666666666:role/rolename"
                ]
            },
            "Action": "glue:*",
            "Resource": [
                "arn:aws:glue:us-east-1:999999999999:catalog",
```

```
                "arn:aws:glue:us-east-1:999999999999:database/tpch1000",
                "arn:aws:glue:us-east-1:999999999999:table/tpch1000/customer"
            ]
        }
    ]
}
```

Al termine, ti consigliamo di utilizzare l'AWS Glue API per effettuare alcune chiamate di prova tra più account per confermare che le autorizzazioni siano configurate come previsto.

Fase 2: Il mutuatario registra l'account AWS Glue Data Catalog che appartiene al proprietario

La procedura seguente illustra come utilizzare la console Athena per configurare AWS Glue Data Catalog nell'account Amazon Web Services del proprietario come origine dei dati. Per informazioni sull'utilizzo delle operazioni API anziché sulla console per registrare il catalogo, consulta la sezione Utilizzo dell'API per registrare un catalogo dati Athena appartenente all'account del proprietario.

Per registrare un' AWS Glue Data Catalog appartenenza a un altro account

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Espandi Amministrazione, quindi seleziona Origini dati.
4. Nell'angolo in alto a destra, scegli Create data source (Crea origine dei dati).
5. Nella pagina Scegli un'origine dati, per Origini dati, seleziona S3 - AWS Glue Data Catalog, quindi scegli Avanti.

6. Nella pagina Inserisci dettagli origine dati, nella sezione AWS Glue Data Catalog, per Scegli un AWS Glue Data Catalog, seleziona AWS Glue Data Catalog in un altro account.
7. Per Dataset details (Dettagli del set di dati), fornisci le seguenti informazioni:
   - Nome origine dati: inserisci il nome che desideri utilizzare nelle query SQL per fare riferimento al catalogo dati nell'altro account.
   - Descrizione — (Facoltativo) Inserisci una descrizione del catalogo dati nell'altro account.
   - ID catalogo — Inserisci l'ID account Amazon Web Services a 12 cifre dell'account a cui appartiene il catalogo dati. L'ID dell'account Amazon Web Services è l'ID del catalogo.
8. (Facoltativo) Espandi Tag, quindi inserisci le coppie chiave-valore che intendi associare all'origine dati. Per ulteriori informazioni sui tag, consulta Assegnazione di tag alle risorse Athena.
9. Seleziona Successivo.
10. Nella pagina Review and create (Rivedi e crea), esamina le informazioni inserite, quindi scegli Create data source (Crea origine dei dati). La pagina Data source details (Dettagli sull'origine dei dati) elenca i database e i tag per il catalogo dati registrato.
11. Scegli Data sources (Origini dati). Il catalogo dati che hai registrato è elencato nella colonna Data source name (Nome origine dei dati).
12. Per visualizzare o modificare le informazioni sul catalogo dati, scegli il catalogo, quindi scegli Actions (Operazioni), Edit (Modifica).
13. Per eliminare il nuovo catalogo dati, scegli il catalogo, quindi scegli Actions (Operazioni), Delete (Elimina).

Passaggio 3: il mutuatario invia una query

Il mutuatario invia una query che fa riferimento al catalogo utilizzando il catalogo. banca dati. sintassi della *tabella*, come nell'esempio seguente:

```
SELECT * FROM ownerCatalog.tpch1000.customer
```

Invece di utilizzare la sintassi completa, il mutuatario può anche specificare il catalogo contestualmente passandolo tramite. QueryExecutionContext

Autorizzazioni Amazon S3 aggiuntive

- Se l'account del mutuatario utilizza una query Athena per scrivere nuovi dati in una tabella nell'account del proprietario, il proprietario non avrà automaticamente accesso a questi dati in Amazon S3, anche se la tabella esiste nell'account del proprietario. Questo perché il mutuatario è il proprietario dell'oggetto delle informazioni in Amazon S3, a meno che non sia configurato diversamente. Per concedere al proprietario l'accesso ai dati, imposta le autorizzazioni sugli oggetti di conseguenza come passaggio aggiuntivo.
- Alcune operazioni DDL tra account comeMSCK REPAIR TABLE richiedono autorizzazioni Amazon S3. Ad esempio, se l'account del mutuatario sta eseguendo un'`MSCK REPAIR`operazione su più account su una tabella dell'account del proprietario i cui dati sono contenuti in un bucket S3 dell'account proprietario, tale bucket deve concedere le autorizzazioni al ruolo assunto dal mutuatario affinché la query abbia esito positivo.

Per ulteriori informazioni sulla concessione di autorizzazioni per il bucket, consulta Come impostare le autorizzazioni ACL per un bucket nella Guida per l'utente di Amazon Simple Storage Service.

Utilizzo di un catalogo in modo dinamico

In alcuni casi potresti voler eseguire rapidamente test su un catalogo AWS Glue tra account senza la registrazione necessaria a tal fine. È possibile eseguire dinamicamente query tra account senza creare l'oggetto della risorsa `DataCatalog` se le autorizzazioni IAM e Amazon S3 richieste sono configurate correttamente come descritto in precedenza in questo documento.

Per fare riferimento esplicitamente a un catalogo senza registrazione, utilizza la sintassi nell'esempio seguente:

```
SELECT * FROM "glue:arn:aws:glue:us-east-1:999999999999:catalog".tpch1000.customer
```

Usa il formato"`glue:<arn>`", dove *`<arn>`* è l'ARN di AWS Glue Data Catalog che si desidera utilizzare. Nell'esempio, Athena utilizza questa sintassi per puntare dinamicamente al catalogo AWS Glue dati dell'account 9999 come se avessi creato separatamente un oggetto per esso. `DataCatalog`

Note per l'utilizzo di cataloghi dinamici

Quando utilizzi i cataloghi dinamici, ricorda i seguenti punti.

- L'utilizzo di un catalogo dinamico richiede le autorizzazioni IAM normalmente utilizzate per le operazioni dell'API Athena Catalogo dati. La differenza principale è che il nome della risorsa Catalogo dati segue la convenzione di denominazione `glue:*`.
- Il catalogo ARN deve appartenere alla stessa Regione in cui viene eseguita la query.
- Quando si utilizza un catalogo dinamico in una query o visualizzazione DML, racchiuderlo con virgolette doppie basse (\"). Quando si utilizza un catalogo dinamico in una query DDL, circondalo con caratteri di backtick (`).

Utilizzo dell'API per registrare un catalogo dati Athena appartenente all'account del proprietario

Invece di utilizzare la console Athena come descritto nel passaggio 2, puoi utilizzare le operazioni API per registrare il catalogo dati che appartiene all'account del proprietario.

Il creatore della DataCatalogrisorsa Athena deve disporre delle autorizzazioni necessarie per eseguire l'operazione API Athena. CreateDataCatalog A seconda delle esigenze, potrebbe essere necessario accedere a operazioni API aggiuntive. Per ulteriori informazioni, consulta Policy di esempio del catalogo dati.

Il seguente ente di `CreateDataCatalog` richiesta registra un AWS Glue catalogo per l'accesso su più account:

```
# Example CreateDataCatalog request to register a cross-account Glue catalog:
{
    "Description": "Cross-account Glue catalog",
    "Name": "ownerCatalog",
    "Parameters": {"catalog-id" : "999999999999"  # Owner's account ID
    },
    "Type": "GLUE"
}
```

Il codice di esempio seguente utilizza un client Java per creare l'oggetto `DataCatalog`.

```
# Sample code to create the DataCatalog through Java client
CreateDataCatalogRequest request = new CreateDataCatalogRequest()
    .withName("ownerCatalog")
    .withType(DataCatalogType.GLUE)
    .withParameters(ImmutableMap.of("catalog-id", "999999999999"));

athenaClient.createDataCatalog(request);
```

Dopo questi passaggi, il mutuatario dovrebbe vedere `ownerCatalog` quando chiama l'operazione API. ListDataCatalogs

Risorse aggiuntive

- Registrazione e AWS Glue Data Catalog da un altro account
- Configura l'accesso tra account a un account condiviso AWS Glue Data Catalog con Amazon Athena nella guida Prescriptive AWS Guidance Patterns.
- Esegui query su AWS Glue Data Catalog più account utilizzando Amazon Athena nel blog sui Big AWS Data
- Concessione dell'accesso multi-account nella Guida per gli sviluppatori di AWS Glue

Accesso da Athena ai metadati crittografati nel AWS Glue Data Catalog

Se utilizzi Amazon Athena, puoi abilitare la crittografia AWS Glue Data Catalog utilizzando la AWS Glue console o l'API. AWS Glue Data Catalog Per informazioni, consulta Crittografia del Data Catalog nella Guida per lo sviluppatore di AWS Glue .

Se AWS Glue Data Catalog è crittografato, è necessario aggiungere le seguenti azioni a tutte le politiche utilizzate per accedere ad Athena:

```
{
 "Version": "2012-10-17",
 "Statement": {
 "Effect": "Allow",
     "Action": [
           "kms:GenerateDataKey",
           "kms:Decrypt",
           "kms:Encrypt"
      ],
     "Resource": "(arn of the key used to encrypt the catalog)"
   }
}
```

Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Accesso a gruppi di lavoro e tag

Un gruppo di lavoro è una risorsa gestita da Athena. Pertanto, se la policy del gruppo di lavoro utilizza operazioni che accettano `workgroup` come input, è necessario specificare l'ARN del gruppo di lavoro nel modo seguente dove *`workgroup-name`* è il nome del gruppo di lavoro:

```
"Resource": [arn:aws:athena:region:AWSAcctID:workgroup/workgroup-name]
```

Ad esempio, per un gruppo di lavoro denominato `test_workgroup` nella Regione `us-west-2` per l'account Amazon Web Services `123456789012`, specifica il gruppo di lavoro come una risorsa utilizzando il seguente ARN:

```
"Resource":["arn:aws:athena:us-east-2:123456789012:workgroup/test_workgroup"]
```

Per accedere ai gruppi di lavoro abilitati alla propagazione dell'identità affidabile (TIP), gli utenti di IAM Identity Center devono essere assegnati a `IdentityCenterApplicationArn` ciò che viene restituito dalla risposta dell'azione API GetWorkGroupAthena.

- Per un elenco delle policy dei gruppi di lavoro, consulta the section called “Esempi di policy per i gruppi di lavoro”.
- Per un elenco delle policy basate su tag per i gruppi di lavoro, consulta Policy di controllo degli accessi IAM basate su tag.
- Per ulteriori informazioni sulla creazione di policy IAM per i gruppi di lavoro, consulta Policy IAM per l'accesso ai gruppi di lavoro.
- Per un elenco completo delle operazioni Amazon Athena, consulta i nomi delle operazioni API nella documentazione di riferimento dell'API Amazon Athena.
- Per ulteriori informazioni sulle policy IAM, consulta Creazione di policy con l'editor visivo nella Guida per l'utente di IAM.

Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Consenti l'accesso alle istruzioni preparate

Questo argomento tratta le autorizzazioni IAM per le istruzioni preparate in Amazon Athena. Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Per ulteriori informazioni sulle istruzioni preparate, consulta Utilizzo di query parametrizzate.

Le seguenti autorizzazioni IAM sono necessarie per creare, gestire ed eseguire istruzioni preparate.

```
athena:CreatePreparedStatement
athena:UpdatePreparedStatement
athena:GetPreparedStatement
athena:ListPreparedStatements
athena:DeletePreparedStatement
```

Utilizza queste autorizzazioni come illustrato nella tabella seguente.

Per farlo	Vanno concesse le seguenti autorizzazioni:
Esecuzione di una query PREPARE	athena:StartQueryExecution athena:CreatePreparedStatement
Riesegui una query PREPARE per aggiornare un'istruzione preparata esistente	athena:StartQueryExecution athena:UpdatePreparedStatement
Esegui una query EXECUTE	athena:StartQueryExecution athena:GetPreparedStatement
Esecuzione di una query DEALLOCATE PREPARE	athena:StartQueryExecution athena:DeletePreparedStatement

Esempio

Il criterio IAM di esempio seguente concede le autorizzazioni per gestire ed eseguire istruzioni preparate su un ID account e un gruppo di lavoro specifici.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:StartQueryExecution",
                "athena:CreatePreparedStatement",
```

```
                "athena:UpdatePreparedStatement",
                "athena:GetPreparedStatement",
                "athena:DeletePreparedStatement",
                "athena:ListPreparedStatements"
            ],
            "Resource": [
                "arn:aws:athena:*:111122223333:workgroup/<workgroup-name>"
            ]
        }
    ]
}
```

Usare Athena con CalledVia le chiavi di contesto

Quando un principale effettua una richiesta a AWS, AWS raccoglie le informazioni sulla richiesta in un contesto di richiesta che valuta e autorizza la richiesta. È possibile utilizzare l'elemento `Condition` di una policy JSON per confrontare le chiavi della richiesta con i valori chiave specificati nella policy. Le chiavi di condizione globali sono chiavi di condizione con un prefisso `aws:`.

La chiave di contesto aws:CalledVia

Puoi usare la chiave di contesto di condizione globale aws:CalledVia per confrontare i servizi nella policy con i servizi che hanno effettuato richieste per conto del principale IAM (utente o ruolo). Quando un principale effettua una richiesta a un Servizio AWS, tale servizio potrebbe utilizzare le credenziali del principale per effettuare richieste successive ad altri servizi. La chiave `aws:CalledVia` contiene un elenco ordinato di ciascun servizio nella catena che ha effettuato le richieste per conto dell'entità principale.

Specificando il nome principale del servizio per la chiave di `aws:CalledVia` contesto, è possibile rendere la chiave Servizio AWS di contesto specifica. Ad esempio, puoi utilizzare la chiave di condizione `aws:CalledVia` per limitare le richieste solo a quelle fatte da Athena. Per utilizzare la chiave di condizione `aws:CalledVia` in una policy con Athena, specifica il nome del principale del servizio Athena `athena.amazonaws.com`, come nell'esempio seguente.

```
...
    "Condition": {
        "ForAnyValue:StringEquals": {
            "aws:CalledVia": "athena.amazonaws.com"
        }
    }
...
```

Puoi utilizzare la chiave di contesto `aws:CalledVia` per garantire che i chiamanti abbiano accesso a una risorsa (come una funzione Lambda) solo se chiamano la risorsa da Athena.

> **Note**
>
> La chiave di contesto `aws:CalledVia` non è compatibile con la funzionalità di propagazione delle identità attendibili.

Aggiungi una chiave di CalledVia contesto opzionale per un accesso granulare a una funzione Lambda

Athena richiede che il chiamante abbia le autorizzazioni `lambda:InvokeFunction` per chiamare la funzione Lambda associata alla query. La seguente istruzione consente l'accesso a grana fine a una funzione Lambda in modo che l'utente possa utilizzare solo Athena per richiamare la funzione Lambda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor3",
            "Effect": "Allow",
            "Action": "lambda:InvokeFunction",
            "Resource": "arn:aws:lambda:us-
east-1:111122223333:function:OneAthenaLambdaFunction",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": "athena.amazonaws.com"
                }
            }
        }
    ]
}
```

L'esempio seguente mostra l'aggiunta dell'istruzione precedente a una policy che consente a un utente di eseguire e leggere una query federata. I principali autorizzati a eseguire queste operazioni possono eseguire query che specificano i cataloghi Athena associati a un'origine dati federata. Tuttavia, il principale non può accedere alla funzione Lambda associata a meno che la funzione non venga richiamata tramite Athena.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "athena:GetWorkGroup",
                "s3:PutObject",
                "s3:GetObject",
                "athena:StartQueryExecution",
                "s3:AbortMultipartUpload",
                "athena:StopQueryExecution",
                "athena:GetQueryExecution",
                "athena:GetQueryResults",
                "s3:ListMultipartUploadParts"
            ],
            "Resource": [
                "arn:aws:athena:*:111122223333:workgroup/WorkGroupName",
                "arn:aws:s3:::MyQueryResultsBucket/*",
                "arn:aws:s3:::MyLambdaSpillBucket/MyLambdaSpillPrefix*"
            ]
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": "athena:ListWorkGroups",
            "Resource": "*"
        },
        {
            "Sid": "VisualEditor2",
            "Effect": "Allow",
            "Action":
                [
                "s3:ListBucket",
                "s3:GetBucketLocation"
                ],
            "Resource": "arn:aws:s3:::MyLambdaSpillBucket"
        },
        {
            "Sid": "VisualEditor3",
            "Effect": "Allow",
            "Action": "lambda:InvokeFunction",
```

```
            "Resource": [
                "arn:aws:lambda:*:111122223333:function:OneAthenaLambdaFunction",
                "arn:aws:lambda:*:111122223333:function:AnotherAthenaLambdaFunction"
            ],
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:CalledVia": "athena.amazonaws.com"
                }
            }
        }
    ]
}
```

Per ulteriori informazioni sulle chiavi di contesto `CalledVia`, consulta Chiavi di contesto delle condizioni globali di AWS nella Guida per l'utente di IAM.

Consenti l'accesso a un connettore dati Athena per il metastore Hive esterno

Negli esempi di policy di autorizzazione in questo argomento vengono illustrate le operazioni consentite obbligatorie e le risorse per le quali sono consentite. Esamina attentamente queste policy e modificale in base si tuoi requisiti prima di collegare policy di autorizzazione simili a identità IAM.

- Example Policy to Allow an IAM Principal to Query Data Using Athena Data Connector for External Hive Metastore
- Example Policy to Allow an IAM Principal to Create an Athena Data Connector for External Hive Metastore

Example : consenti a un principale IAM di eseguire query sui dati utilizzando Athena Data Connector per il metastore Hive esterno

La seguente policy è collegata ai principali IAM oltre alla AWS politica gestita: AmazonAthenaFullAccess, che concede l'accesso completo a operazioni Athena.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "lambda:GetFunction",
```

```
                "lambda:GetLayerVersion",
                "lambda:InvokeFunction"
            ],
            "Resource": [
                "arn:aws:lambda:*:111122223333:function:MyAthenaLambdaFunction",
                "arn:aws:lambda:*:111122223333:function:AnotherAthenaLambdaFunction",
                "arn:aws:lambda:*:111122223333:layer:MyAthenaLambdaLayer:*"
            ]
        },
        {
            "Sid": "VisualEditor2",
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:PutObject",
                "s3:ListMultipartUploadParts",
                "s3:AbortMultipartUpload"
            ],
            "Resource": "arn:aws:s3:::MyLambdaSpillBucket/MyLambdaSpillLocation"
        }
    ]
}
```

Spiegazione delle autorizzazioni

Operazioni consentite	Spiegazione
`"s3:GetBucketLocation", "s3:GetObject", "s3:ListBucket", "s3:PutObject", "s3:ListMultipartUploadParts", "s3:AbortMultipartUpload"`	`s3`le azioni consentono la lettura e la scrittura sulla risorsa specificata come`"arn:aws:s3::: MyLambdaSpillBucket /MyLambdaSpillLocation "`, where *`MyLambdaSpillLocation`* identifica il bucket di fuoriuscita specificato nella configurazione della funzione o delle funzioni Lambda richiamate. L'identificatore di *`risorsa arn:aws:lambda: *: my:Layer AWSAcctId :: MyAthenaLambdaLayer *`* è richiesto solo se si utilizza un livello Lambda per creare dipendenze di runtime personali

Operazioni consentite	Spiegazione
	zzate per ridurre le dimensioni degli artefatti funzionali al momento della distribuzione. Il * nell'ultima posizione è un carattere jolly per la versione del livello.
`"lambda:GetFunction", "lambda:GetLayerVersion", "lambda:InvokeFunction"`	Consente alle query di AWS Lambda richiamare le funzioni specificate nel blocco. `Resource` Ad esempio`arn:aws:lambda:*: MyAWSAcctId :function : MyAthenaLambdaFunction` , where *`MyAthenaLambdaFunction`* specifica il nome di una funzione Lambda da richiamare. Più funzioni possono essere specificate come mostrato nell'esempio.

Example : consenti a un principale IAM di creare un Athena Data Connector per il metastore Hive esterno

La seguente policy è collegata ai principali IAM oltre alla AWS politica gestita: AmazonAthenaFullAccess, che concede l'accesso completo a operazioni Athena.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "lambda:GetFunction",
                "lambda:ListFunctions",
                "lambda:GetLayerVersion",
                "lambda:InvokeFunction",
                "lambda:CreateFunction",
                "lambda:DeleteFunction",
                "lambda:PublishLayerVersion",
                "lambda:DeleteLayerVersion",
                "lambda:UpdateFunctionConfiguration",
                "lambda:PutFunctionConcurrency",
```

```
                "lambda:DeleteFunctionConcurrency"
            ],
            "Resource": "arn:aws:lambda:*:111122223333:
 function: MyAthenaLambdaFunctionsPrefix*"
        }
    ]
}
```

Spiegazione delle autorizzazioni

Consente alle query di richiamare le AWS Lambda funzioni per le funzioni specificate nel blocco AWS Lambda . `Resource` Ad esempio`arn:aws:lambda:*:`*`MyAWSAcctId`*`:function:`*`MyAthenaLambdaFunction`*, where *`MyAthenaLambdaFunction`*specifica il nome di una funzione Lambda da richiamare. Più funzioni possono essere specificate come mostrato nell'esempio.

Consentire l'accesso alla funzione Lambda a metastore Hive esterni

Per richiamare una funzione Lambda nell'account, è necessario creare un ruolo con le seguenti autorizzazioni:

- `AWSLambdaVPCAccessExecutionRole`: un'autorizzazione per il ruolo di esecuzione di AWS Lambda per gestire le interfacce di rete elastiche che collegano la funzione a un VPC. Assicurarsi di disporre di un numero sufficiente di interfacce di rete e indirizzi IP disponibili.
- `AmazonAthenaFullAccess`— La policy AmazonAthenaFullAccessgestita garantisce l'accesso completo ad Athena.
- Una policy Amazon S3 per consentire alla funzione Lambda di scrivere su S3 e consentire ad Athena di leggere da S3.

Ad esempio, la seguente policy definisce l'autorizzazione per la posizione di spill `s3:\\mybucket\spill`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketLocation",
                "s3:GetObject",
```

```
                "s3:ListBucket",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/spill"
            ]
        }
    ]
}
```

Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Creazione di funzioni Lambda

Per creare una funzione Lambda nell'account, sono necessari i permessi di sviluppo delle funzioni o il ruolo `AWSLambdaFullAccess`. Per ulteriori informazioni, consulta Policy IAM basate su identità per AWS Lambda.

Poiché Athena utilizza le AWS Serverless Application Repository per creare le funzioni Lambda, il superutente o l'amministratore che crea le funzioni Lambda deve disporre anche di politiche IAM per consentire le query federate Athena.

Registrazione del catalogo e operazioni API sui metadati

Per accedere all'API di registrazione del catalogo e alle operazioni dell'API dei metadati, utilizza la policy gestita. AmazonAthenaFullAccess Se non si utilizza questa policy, aggiungere le seguenti operazioni API alle policy Athena:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:ListDataCatalogs",
                "athena:GetDataCatalog",
                "athena:CreateDataCatalog",
                "athena:UpdateDataCatalog",
                "athena:DeleteDataCatalog",
                "athena:GetDatabase",
                "athena:ListDatabases",
                "athena:GetTableMetadata",
```

```
                "athena:ListTableMetadata"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Invocazione di funzioni Lambda tra più Regioni

Per invocare una funzione Lambda in una Regione diversa da quella in cui si eseguono le query Athena, utilizzare l'ARN completo della funzione Lambda. Per impostazione predefinita, Athena invoca le funzioni Lambda definite nella stessa Regione. Se è necessario invocare una funzione Lambda per accedere a un metastore Hive in una Regione diversa dalla Regione in cui si eseguono le query Athena, è necessario fornire l'ARN completo della funzione Lambda.

Si supponga, ad esempio, di definire il catalogo `ehms` nella Regione `eu-central-1` Europa (Francoforte) per utilizzare la seguente funzione Lambda nella Regione Stati Uniti orientali (Virginia settentrionale).

```
arn:aws:lambda:us-east-1:111122223333:function:external-hms-service-new
```

Quando si specifica l'ARN completo in questo modo, Athena può invocare la funzione Lambda `external-hms-service-new` su `us-east-1` per recuperare i dati del metastore Hive da `eu-central-1`.

> **Note**
>
> Il catalogo `ehms` deve essere registrato nella stessa Regione in cui vengono eseguite le query Athena.

Invocazione Lambda tra Account Lambda

A volte potrebbe essere necessario richiedere l'accesso a un metastore Hive da un altro account. Ad esempio, per eseguire un metastore Hive, è possibile avviare un cluster EMR da un account diverso da quello utilizzato per le query Athena. Gruppi o team diversi potrebbero eseguire metastore Hive con account diversi all'interno del loro VPC. Oppure si potrebbe voler accedere ai metadati provenienti da vari metastore Hive di gruppi o team diversi.

Athena utilizza il supporto AWS Lambda per l'accesso tra account per abilitare l'accesso tra account per i metastore Hive.

> **Note**
>
> Si noti che l'accesso tra account per Athena implica normalmente l'accesso tra account sia per i metadati che per i dati in Amazon S3.

Si consideri il seguente scenario:

- L'account `111122223333` imposta la funzione Lambda `external-hms-service-new` su us-east-1 in Athena per accedere a un metastore Hive in esecuzione su un cluster EMR.
- L'account `111122223333` vuole consentire all'account 444455556666 di accedere ai dati del metastore Hive.

Per concedere `444455556666` all'account l'accesso alla funzione Lambda`external-hms-service-new`, account `111122223333` utilizza il comando seguente AWS CLI `add-permission`. Il comando è stato formattato per la leggibilità.

```
$ aws --profile perf-test lambda add-permission
      --function-name external-hms-service-new
      --region us-east-1
      --statement-id Id-ehms-invocation2
      --action "lambda:InvokeFunction"
      --principal arn:aws:iam::444455556666:user/perf1-test
{
    "Statement": "{\"Sid\":\"Id-ehms-invocation2\",
                  \"Effect\":\"Allow\",
                  \"Principal\":{\"AWS\":\"arn:aws:iam::444455556666:user/perf1-test
\"},
                  \"Action\":\"lambda:InvokeFunction\",
                  \"Resource\":\"arn:aws:lambda:us-
east-1:111122223333:function:external-hms-service-new\"}"
}
```

Per verificare l'autorizzazione Lambda, utilizzare il comando `get-policy`, come nell'esempio seguente. Il comando è stato formattato per la leggibilità.

```
$ aws --profile perf-test lambda get-policy
```

```
      --function-name arn:aws:lambda:us-east-1:111122223333:function:external-hms-
service-new
      --region us-east-1
{
    "RevisionId": "711e93ea-9851-44c8-a09f-5f2a2829d40f",
    "Policy": "{\"Version\":\"2012-10-17\",
                \"Id\":\"default\",
                \"Statement\":[{\"Sid\":\"Id-ehms-invocation2\",
                               \"Effect\":\"Allow\",
                               \"Principal\":{\"AWS\":
\"arn:aws:iam::444455556666:user/perf1-test\"},
                               \"Action\":\"lambda:InvokeFunction\",
                               \"Resource\":\"arn:aws:lambda:us-
east-1:111122223333:function:external-hms-service-new\"}]}"
}
```

Dopo aver aggiunto l'autorizzazione, è possibile utilizzare un ARN completo della funzione Lambda su `us-east-1` come mostrato di seguito quando si definisce un catalogo `ehms`:

```
arn:aws:lambda:us-east-1:111122223333:function:external-hms-service-new
```

Per informazioni sulla invocazione tra regioni, vedere Invocazione di funzioni Lambda tra più Regioni in questo argomento.

Concedere l'accesso tra account ai dati

Prima di poter eseguire query Athena, è necessario concedere l'accesso tra account ai dati in Amazon S3. Questa operazione può essere eseguita in uno dei seguenti modi:

- Aggiornare la policy dell'elenco di controllo dell'accesso del bucket Amazon S3 con un ID utente canonico.
- Aggiungere l'accesso tra account alla policy del bucket Amazon S3.

Ad esempio, aggiungere la policy seguente alla policy del bucket Amazon S3 nell'account `111122223333` per consentire all'account `444455556666` di leggere i dati dalla posizione Amazon S3 specificata.

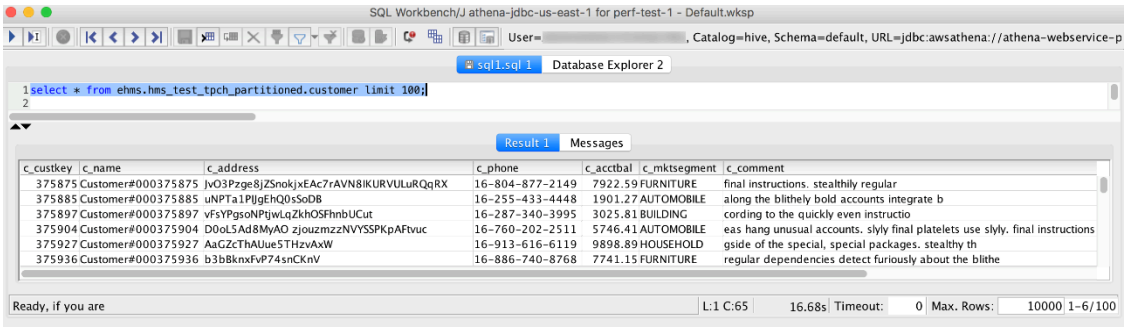
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "Stmt1234567890123",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::444455556666:user/perf1-test"
            },
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::athena-test/lambda/dataset/*"
        }
    ]
}
```

> **Note**
>
> Potrebbe essere necessario concedere l'accesso tra account ad Amazon S3 non solo ai dati, ma anche alla posizione di spill Amazon S3. La funzione Lambda esegue lo spillover dei dati aggiuntivi nella posizione di spill quando la dimensione dell'oggetto di risposta supera una determinata soglia. Vedere l'inizio di questo argomento per una policy di esempio.

Nell'esempio corrente, dopo aver concesso l'accesso tra account a `444455556666`,, `444455556666` può utilizzare il catalogo `ehms` in `account` per eseguire query sulle tabelle definite nell'account `111122223333`.

Nell'esempio seguente, il profilo di SQL Workbench `perf-test-1` è per l'account `444455556666`. La query utilizza il catalogo `ehms` per accedere al metastore Hive e ai dati Amazon S3 nell'account `111122223333`.



Esempio di policy di autorizzazione IAM per consentire la query federata Athena

Negli esempi di policy di autorizzazione in questo argomento vengono illustrate le operazioni consentite obbligatorie e le risorse per le quali sono consentite. Esamina attentamente queste policy e modificale in base ai tuoi requisiti prima di collegarle a identità IAM.

Per informazioni sul collegamento di policy alle identità IAM, consulta Aggiunta e rimozione di autorizzazioni alle identità IAM nella Guida per l'utente di IAM.

- Example policy to allow an IAM principal to run and return results using Athena Federated Query
- Example Policy to Allow an IAM Principal to Create a Data Source Connector

Example : consenti a un principale IAM di eseguire e restituire risultati utilizzando la query federata Athena

La seguente policy di autorizzazione basata su identità consente operazioni richieste da un utente o un altro principale IAM per utilizzare la query federata Athena. I principali autorizzati a eseguire queste operazioni sono in grado di eseguire query che specificano i cataloghi Athena associati a un'origine dati federata.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Athena",
            "Effect": "Allow",
            "Action": [
                "athena:GetDataCatalog",
                "athena:GetQueryExecution",
                "athena:GetQueryResults",
                "athena:GetWorkGroup",
                "athena:StartQueryExecution",
                "athena:StopQueryExecution"
            ],
            "Resource": [
                "arn:aws:athena:*:111122223333:workgroup/WorkgroupName",
                "arn:aws:athena:aws_region:111122223333:datacatalog/DataCatalogName"
            ]
        },
        {
            "Sid": "ListAthenaWorkGroups",
            "Effect": "Allow",
            "Action": "athena:ListWorkGroups",
            "Resource": "*"
        },
        {
            "Sid": "Lambda",
```

```
            "Effect": "Allow",
            "Action": "lambda:InvokeFunction",
            "Resource": [
                "arn:aws:lambda:*:111122223333:function:OneAthenaLambdaFunction",
                "arn:aws:lambda:*:111122223333:function:AnotherAthenaLambdaFunction"
            ]
        },
        {
            "Sid": "S3",
            "Effect": "Allow",
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:ListMultipartUploadParts",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::MyLambdaSpillBucket",
                "arn:aws:s3:::MyLambdaSpillBucket/*",
                "arn:aws:s3:::MyQueryResultsBucket",
                "arn:aws:s3:::MyQueryResultsBucket/*"
            ]
        }
    ]
}
```

Spiegazione delle autorizzazioni

Operazioni consentite	Spiegazione
`"athena:GetQueryExecution", "athena:GetQueryResults", "athena:GetWorkGroup", "athena:StartQueryExecution", "athena:StopQueryExecution"`	Autorizzazioni Athena richieste per eseguire query federate.
`"athena:GetDataCatalog", "athena:GetQueryExecution," "athena:GetQueryResults", "athena:GetWorkGroup",`	Autorizzazioni Athena necessarie per eseguire query di visualizzazione federate. L'`GetDataCatalog` azione è necessaria per le viste.

Operazioni consentite	Spiegazione
`"athena:StartQueryExecution", "athena:StopQueryExecution"`	
`"lambda:InvokeFunction"`	Consente alle interrogazioni di richiamare le AWS Lambda funzioni per le AWS Lambda funzioni specificate nel `Resource` blocco. Ad esempio`arn:aws:lambda:*: MyAWSAcctId :function: MyAthenaLambdaFunction` , where `MyAthenaLambdaFunction` specifica il nome di una funzione Lambda da richiamare. Come mostrato nell'esempio, è possibile specificare più funzioni.

Operazioni consentite	Spiegazione
`"s3:AbortMultipartUpload", "s3:GetBucketLocation", "s3:GetObject", "s3:ListBucket", "s3:ListMultipartUploadParts", "s3:PutObject"`	Le `s3:GetBucketLocation` autorizzazioni `s3:ListBucket` e sono necessarie per accedere al bucket di output delle query per i principali IAM in esecuzione. `StartQueryExecution` `s3:PutObject` `s3:ListMultipartUploadParts`, e `s3:AbortMultipartUpload` consentono di scrivere i risultati delle query in tutte le sottocartelle del bucket dei risultati della query come specificato dall'identificatore di `arn:aws:s3:::` *`MyQueryResultsBucket`* `/*` risorsa, dove si trova il bucket dei risultati *`MyQueryResultsBucket`* della query Athena. Per ulteriori informazioni, consulta Utilizzo dei risultati delle query, delle query recenti e dei file di output. `s3:GetObject` consente la lettura dei risultati delle query e della cronologia delle query per la risorsa specificata come`arn:aws:s3:::` *`MyQueryResultsBucket`*, where *`MyQueryResultsBucket`* è il bucket dei risultati delle query Athena. `s3:GetObject` consente inoltre la lettura dalla risorsa specificata come`"arn:aws:s3:::` *`MyLambdaSpillBucket`* */`*`MyLambdaSpillPrefix`*`*"`, where *`MyLambdaSpillPrefix`* è specificata nella configurazione della funzione o delle funzioni Lambda richiamate.

Example : consente a un principale IAM di creare un connettore origine dati

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "lambda:CreateFunction",
                "lambda:ListVersionsByFunction",
                "iam:CreateRole",
                "lambda:GetFunctionConfiguration",
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy",
                "lambda:PutFunctionConcurrency",
                "iam:PassRole",
                "iam:DetachRolePolicy",
                "lambda:ListTags",
                "iam:ListAttachedRolePolicies",
                "iam:DeleteRolePolicy",
                "lambda:DeleteFunction",
                "lambda:GetAlias",
                "iam:ListRolePolicies",
                "iam:GetRole",
                "iam:GetPolicy",
                "lambda:InvokeFunction",
                "lambda:GetFunction",
                "lambda:ListAliases",
                "lambda:UpdateFunctionConfiguration",
                "iam:DeleteRole",
                "lambda:UpdateFunctionCode",
                "s3:GetObject",
                "lambda:AddPermission",
                "iam:UpdateRole",
                "lambda:DeleteFunctionConcurrency",
                "lambda:RemovePermission",
                "iam:GetRolePolicy",
                "lambda:GetPolicy"
            ],
            "Resource": [

 "arn:aws:lambda:*:111122223333:function:MyAthenaLambdaFunctionsPrefix*",
                "arn:aws:s3:::awsserverlessrepo-changesets-1iiv3xa62ln3m/*",
```

```
                "arn:aws:iam::*:role/RoleName",
                "arn:aws:iam::111122223333:policy/*"
            ]
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "cloudformation:CreateUploadBucket",
                "cloudformation:DescribeStackDriftDetectionStatus",
                "cloudformation:ListExports",
                "cloudformation:ListStacks",
                "cloudformation:ListImports",
                "lambda:ListFunctions",
                "iam:ListRoles",
                "lambda:GetAccountSettings",
                "ec2:DescribeSecurityGroups",
                "cloudformation:EstimateTemplateCost",
                "ec2:DescribeVpcs",
                "lambda:ListEventSourceMappings",
                "cloudformation:DescribeAccountLimits",
                "ec2:DescribeSubnets",
                "cloudformation:CreateStackSet",
                "cloudformation:ValidateTemplate"
            ],
            "Resource": "*"
        },
        {
            "Sid": "VisualEditor2",
            "Effect": "Allow",
            "Action": "cloudformation:*",
            "Resource": [
                "arn:aws:cloudformation:*:111122223333:stack/aws-serverless-
repository-MyCFStackPrefix*/*",
                "arn:aws:cloudformation:*:111122223333:stack/
serverlessrepo-MyCFStackPrefix*/*",
                "arn:aws:cloudformation:*:*:transform/Serverless-*",
                "arn:aws:cloudformation:*:111122223333:stackset/aws-serverless-
repository-MyCFStackPrefix*:*",
                "arn:aws:cloudformation:*:111122223333:stackset/
serverlessrepo-MyCFStackPrefix*:*"
            ]
        },
        {
```

```
            "Sid": "VisualEditor3",
            "Effect": "Allow",
            "Action": "serverlessrepo:*",
            "Resource": "arn:aws:serverlessrepo:*:*:applications/*"
        }
    ]
}
```

Spiegazione delle autorizzazioni

Operazioni consentite	Spiegazione
`"lambda:CreateFunction", "lambda:ListVersionsByFunction", "lambda:GetFunctionConfiguration", "lambda:PutFunctionConcurrency", "lambda:ListTags", "lambda:DeleteFunction", "lambda:GetAlias", "lambda:InvokeFunction", "lambda:GetFunction", "lambda:ListAliases", "lambda:UpdateFunctionConfiguration", "lambda:UpdateFunctionCode", "lambda:AddPermission", "lambda:DeleteFunctionConcurrency", "lambda:RemovePermission", "lambda:GetPolicy" "lambda:GetAccountSettings", "lambda:ListFunctions", "lambda:ListEventSourceMappings",`	Consente la creazione e la gestione delle funzioni Lambda elencate come risorse. Nell'esempio, viene utilizzato un prefisso di nome nell'identificatore di risorsa`arn:aws:lambda:*:` *MyAWSAcctId* `:function:` *MyAthenaLambdaFunctionsPrefix* `*`, dove *MyAthenaLambdaFunctionsPrefix* viene utilizzato un prefisso condiviso nel nome di un gruppo di funzioni Lambda in modo che non sia necessario specificarle singolarmente come risorse. È possibile specificare una o più risorse della funzione Lambda.
`"s3:GetObject"`	Consente la lettura di un bucket che AWS Serverless Application Repository richiede quanto specificato dall'identificatore di risorsa. `arn:aws:s3:::awsserverlessrepo-changesets-` *1iiv3xa62ln3m* `/*` Questo bucket può essere specifico per l'account.

Operazioni consentite	Spiegazione
`"cloudformation:*"`	*Consente la creazione e la gestione degli AWS CloudFormation stack specificati dalla risorsa MyCF.StackPrefix* Questi stack e stackset sono il modo AWS Serverless Application Repository in cui distribuisce connettori e UDF.
`"serverlessrepo:*"`	Consente la ricerca, la visualizzazione, la pubblicazione e l'aggiornamento delle applicazioni nel campo specificato dall'identificatore di AWS Serverless Application Repository risorsa. `arn:aws:serverlessrepo:*:*:applications/*`

Esempio di policy di autorizzazione IAM per consentire le funzioni definite dall'utente (UDF) di Amazon Athena

Negli esempi di policy di autorizzazione in questo argomento vengono illustrate le operazioni consentite obbligatorie e le risorse per le quali sono consentite. Esamina attentamente queste policy e modificale in base si tuoi requisiti prima di collegare policy di autorizzazione simili a identità IAM.

- Example Policy to Allow an IAM Principal to Run and Return Queries that Contain an Athena UDF Statement
- Example Policy to Allow an IAM Principal to Create an Athena UDF

Example : consenti a un principale IAM di eseguire e restituire query contenenti un'istruzione UDF Athena

La seguente policy di autorizzazione basata su identità consente operazioni richieste da un utente o un altro principale IAM per eseguire query che utilizzano istruzioni UDF Athena.

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

```
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "athena:StartQueryExecution",
                "lambda:InvokeFunction",
                "athena:GetQueryResults",
                "s3:ListMultipartUploadParts",
                "athena:GetWorkGroup",
                "s3:PutObject",
                "s3:GetObject",
                "s3:AbortMultipartUpload",
                "athena:StopQueryExecution",
                "athena:GetQueryExecution",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:athena:*:MyAWSAcctId:workgroup/MyAthenaWorkGroup",
                "arn:aws:s3:::MyQueryResultsBucket/*",
                "arn:aws:lambda:*:MyAWSAcctId:function:OneAthenaLambdaFunction",
                "arn:aws:lambda:*:MyAWSAcctId:function:AnotherAthenaLambdaFunction"
            ]
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": "athena:ListWorkGroups",
            "Resource": "*"
        }
    ]
}
```

Spiegazione delle autorizzazioni

Operazioni consentite	Spiegazione
`"athena:StartQueryExecution", "athena:GetQueryResults", "athena:GetWorkGroup", "athena:StopQueryExecution", "athena:GetQueryExecution",`	Autorizzazioni Athena richieste per eseguire query nel gruppo di lavoro `MyAthenaWorkGroup` .

Operazioni consentite	Spiegazione
`"s3:PutObject", "s3:GetObject", "s3:AbortMultipartUpload"`	`s3:PutObject` e `s3:AbortMultipartUpload` consentono di scrivere i risultati delle query in tutte le sottocartelle del bucket dei risultati della query come specificato dall'identificatore di `arn:aws:s3:::` *`MyQueryResultsBucket`* `/*` risorsa, dove si trova il bucket dei risultati *`MyQueryResultsBucket`* della query Athena. Per ulteriori informazioni, consulta Utilizzo dei risultati delle query, delle query recenti e dei file di output. `s3:GetObject` consente la lettura dei risultati delle query e della cronologia delle query per la risorsa specificata come`arn:aws:s3:::` *`MyQueryResultsBucket`* , where *`MyQueryResultsBucket`* è il bucket dei risultati delle query Athena. Per ulteriori informazioni, consulta Utilizzo dei risultati delle query, delle query recenti e dei file di output. `s3:GetObject` consente inoltre la lettura dalla risorsa specificata come`"arn:aws:s3:::` *`MyLambdaSpillBucket`* `/`*`MyLambdaSpillPrefix`* `*"`, where *`MyLambdaSpillPrefix`* è specificata nella configurazione della funzione o delle funzioni Lambda richiamate.

Operazioni consentite	Spiegazione
`"lambda:InvokeFunction"`	Consente alle interrogazioni di richiamare AWS Lambda le funzioni specificate nel blocco. `Resource` Ad esempio `arn:aws:lambda:*: MyAWSAcctId :function : MyAthenaLambdaFunction` , where `MyAthenaLambdaFunction` specifica il nome di una funzione Lambda da richiamare. Più funzioni possono essere specificate come mostrato nell'esempio.

Example : consente a un principale IAM di creare un'UDF Athena

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "lambda:CreateFunction",
                "lambda:ListVersionsByFunction",
                "iam:CreateRole",
                "lambda:GetFunctionConfiguration",
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy",
                "lambda:PutFunctionConcurrency",
                "iam:PassRole",
                "iam:DetachRolePolicy",
                "lambda:ListTags",
                "iam:ListAttachedRolePolicies",
                "iam:DeleteRolePolicy",
                "lambda:DeleteFunction",
                "lambda:GetAlias",
                "iam:ListRolePolicies",
                "iam:GetRole",
                "iam:GetPolicy",
                "lambda:InvokeFunction",
                "lambda:GetFunction",
```

```
                "lambda:ListAliases",
                "lambda:UpdateFunctionConfiguration",
                "iam:DeleteRole",
                "lambda:UpdateFunctionCode",
                "s3:GetObject",
                "lambda:AddPermission",
                "iam:UpdateRole",
                "lambda:DeleteFunctionConcurrency",
                "lambda:RemovePermission",
                "iam:GetRolePolicy",
                "lambda:GetPolicy"
            ],
            "Resource": [

"arn:aws:lambda:*:111122223333:function:MyAthenaLambdaFunctionsPrefix*",
                "arn:aws:s3:::awsserverlessrepo-changesets-1iiv3xa62ln3m/*",
                "arn:aws:iam::*:role/RoleName",
                "arn:aws:iam::111122223333:policy/*"
            ]
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "cloudformation:CreateUploadBucket",
                "cloudformation:DescribeStackDriftDetectionStatus",
                "cloudformation:ListExports",
                "cloudformation:ListStacks",
                "cloudformation:ListImports",
                "lambda:ListFunctions",
                "iam:ListRoles",
                "lambda:GetAccountSettings",
                "ec2:DescribeSecurityGroups",
                "cloudformation:EstimateTemplateCost",
                "ec2:DescribeVpcs",
                "lambda:ListEventSourceMappings",
                "cloudformation:DescribeAccountLimits",
                "ec2:DescribeSubnets",
                "cloudformation:CreateStackSet",
                "cloudformation:ValidateTemplate"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "VisualEditor2",
            "Effect": "Allow",
            "Action": "cloudformation:*",
            "Resource": [
                "arn:aws:cloudformation:*:111122223333:stack/aws-serverless-
repository-MyCFStackPrefix*/*",
                "arn:aws:cloudformation:*:111122223333:stack/
serverlessrepo-MyCFStackPrefix*/*",
                "arn:aws:cloudformation:*:*:transform/Serverless-*",
                "arn:aws:cloudformation:*:111122223333:stackset/aws-serverless-
repository-MyCFStackPrefix*:*",
                "arn:aws:cloudformation:*:111122223333:stackset/
serverlessrepo-MyCFStackPrefix*:*"
            ]
        },
        {
            "Sid": "VisualEditor3",
            "Effect": "Allow",
            "Action": "serverlessrepo:*",
            "Resource": "arn:aws:serverlessrepo:*:*:applications/*"
        }
    ]
}
```

Spiegazione delle autorizzazioni

Operazioni consentite	Spiegazione
`"lambda:CreateFunction", "lambda:ListVersionsByFunction", "lambda:GetFunctionConfiguration", "lambda:PutFunctionConcurrency", "lambda:ListTags", "lambda:DeleteFunction", "lambda:GetAlias", "lambda:InvokeFunction", "lambda:GetFunction", "lambda:ListAliases", "lambda:UpdateFunctionConfigur ation", "lambda:UpdateFunctionCode", "lambda:AddPermission", "lambda:DeleteFunctionConcurrency",`	Consente la creazione e la gestione delle funzioni Lambda elencate come risorse. Nell'esempio, viene utilizzato un prefisso di nome nell'identificatore di risorsa`aarn:aws:lambda:*:` *MyAWSAcctId* `:function:` *MyAthenaLambdaFunctionsPrefix* `*`, dove *MyAthenaLambdaFunctionsPrefix* viene utilizzato un prefisso condiviso nel nome di un gruppo di funzioni Lambda in modo che non sia necessario specificarle singolarmente come risorse. È possibile specificare una o più risorse della funzione Lambda.

Operazioni consentite	Spiegazione
`"lambda:RemovePermission", "lambda:GetPolicy" "lambda:GetAccountSettings", "lambda:ListFunctions", "lambda:ListEventSourceMappings",`	
`"s3:GetObject"`	Consente la lettura di un bucket che AWS Serverless Application Repository richiede quanto specificato dall'identificatore di risorsa. `arn:aws:s3:::awsserverlessrepo-changesets-` *`1iiv3xa62ln3m`* `/*`
`"cloudformation:*"`	*Consente la creazione e la gestione degli AWS CloudFormation stack specificati dalla risorsa MyCF. StackPrefix* Questi stack e stackset sono il modo AWS Serverless Application Repository in cui distribuisce connettori e UDF.
`"serverlessrepo:*"`	Consente la ricerca, la visualizzazione, la pubblicazione e l'aggiornamento delle applicazioni nel campo specificato dall'identificatore di AWS Serverless Application Repository risorsa. `arn:aws:serverlessrepo:*:*:applications/*`

Autorizzazione per l'accesso per ML con Athena

I principali IAM che eseguono query Athena ML devono poter eseguire l'operazione `sagemaker:invokeEndpoint` per gli endpoint Sagemaker che utilizzano. Includi una dichiarazione di policy simile alla seguente nelle policy di autorizzazione basate su identità collegate a identità utente. Inoltre, allega la AWS politica gestita: AmazonAthenaFullAccess, che concede l'accesso completo alle operazioni Athena o a una policy inline modificata che consente un sottoinsieme di operazioni.

Sostituisci `arn:aws:sagemaker:`*`region`*`:`*`AWSAcctID`*`:`*`ModelEndpoint`* nell'esempio con l'ARN o gli ARN degli endpoint del modello da utilizzare nelle query. Per ulteriori informazioni, consulta Azioni, risorse e chiavi di condizione SageMaker nel Service Authorization Reference.

```
{
            "Effect": "Allow",
            "Action": [
                "sagemaker:invokeEndpoint"
            ],
            "Resource": "arn:aws:sagemaker:us-west-2:123456789012:workteam/public-
crowd/default"
}
```

Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Abilitazione dell'accesso federato all'API Athena

Questa sezione illustra l'accesso federato che consente a un utente o a un'applicazione client nell'organizzazione di chiamare le operazioni API di Amazon Athena. In questo caso, gli utenti dell'organizzazione non hanno accesso diretto ad Athena. Le credenziali utente vengono invece gestite all'esterno di AWS Microsoft Active Directory. Active Directory supporta SAML 2.0 (Security Assertion Markup Language 2.0).

Per autenticare gli utenti in questo scenario, è necessario utilizzare il driver JDBC o ODBC con supporto di SAML 2.0 per accedere ad Active Directory Federation Services (AD FS) 3.0 e consentire a un'applicazione client di chiamare le operazioni API di Athena.

Per ulteriori informazioni sul supporto SAML 2.0 su AWS, consulta Informazioni sulla federazione SAML 2.0 nella Guida per l'utente IAM.

> **Note**
>
> L'accesso federato all'API di Athena è supportato per un determinato tipo di provider di identità (IdP), Active Directory Federation Service (AD FS 3.0), parte di Windows Server. L'accesso federato non è compatibile con la funzionalità di propagazione delle identità attendibili di Centro identità IAM. L'accesso viene stabilito attraverso le versioni dei driver JDBC o ODBC che supportano SAML 2.0. Per informazioni, consulta Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

Argomenti

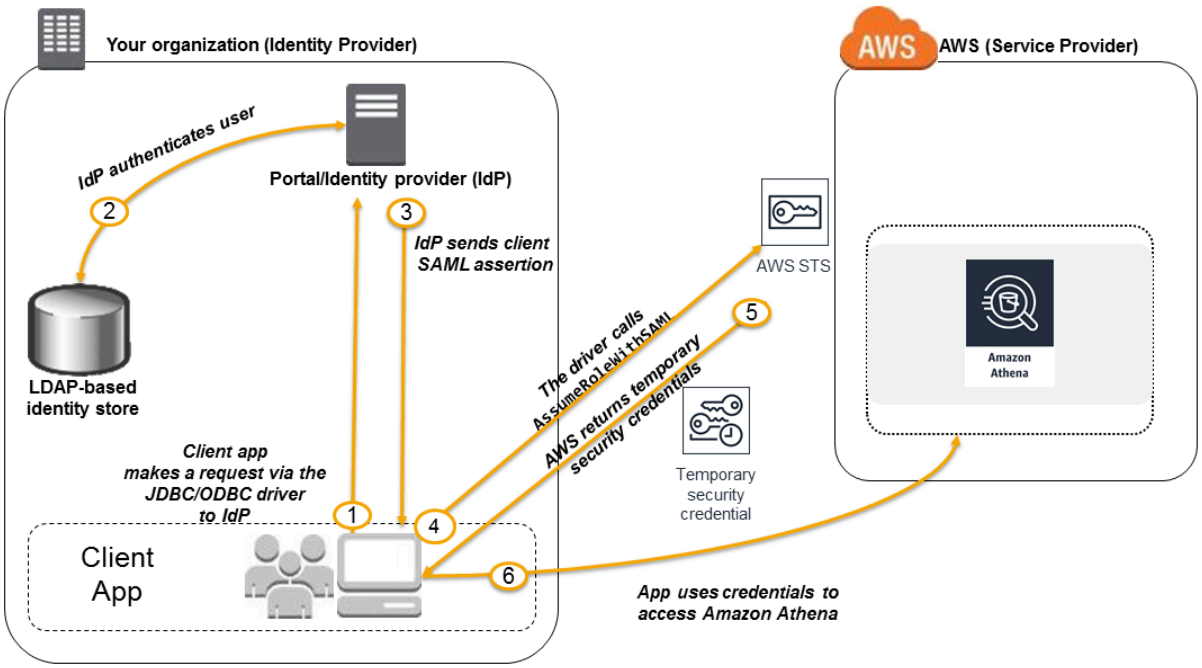
Prima di iniziare

Prima di iniziare, completa i seguenti prerequisiti:

- All'interno dell'organizzazione, installa e configura ADFS 3.0 come IdP.
- Installare e configurare le versioni più recenti disponibili dei driver JDBC e ODBC sui client che vengono utilizzati per l'accesso ad Athena. Il driver deve includere il supporto per l'accesso federato compatibile con SAML 2.0. Per informazioni, consulta Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

Diagramma architetturale

Il diagramma seguente illustra tale processo.



1. Un utente dell'organizzazione utilizza un'applicazione client con il driver JDBC o ODBC per richiedere l'autenticazione dall'IdP dell'organizzazione. L'IdP è ADFS 3.0.
2. L'IdP autentica l'utente rispetto ad Active Directory, che è l'archivio identità dell'organizzazione.
3. L'IdP crea un'asserzione SAML con informazioni sull'utente e invia l'asserzione all'applicazione client tramite il driver JDBC o ODBC.
4. Il driver JDBC o ODBC richiama l'operazione dell'API AWS Security Token Service AssumeRoleWithSAML, passandole i seguenti parametri:
   - L'ARN del fornitore SAML
   - L'ARN del ruolo da assumere
   - L'asserzione SAML dell'IdP

   Per ulteriori informazioni, consulta AssumeRoleWithSAML, nell'API Reference.AWS Security Token Service
5. La risposta API all'applicazione client tramite il driver JDBC o ODBC include le credenziali di sicurezza temporanee.
6. L'applicazione client utilizza le credenziali di sicurezza temporanee per chiamare le operazioni API Athena, consentendo agli utenti di accedere alle operazioni API Athena.

Procedura: accesso federato basato su SAML all'API Athena

Questa procedura stabilisce la fiducia tra l'IdP della tua organizzazione e il AWS tuo account per abilitare l'accesso federato basato su SAML alle operazioni dell'API Amazon Athena.

Per abilitare l'accesso federato all'API Athena:

1. Nella tua organizzazione, registrati AWS come fornitore di servizi (SP) nel tuo IdP. Questo processo è noto come relazione di trust. Per ulteriori informazioni, consultare Configurazione del provider di identità SAML 2.0 con una relazione di trust nella Guida per l'utente di IAM. Come parte di questa operazione, eseguire questi passaggi:

   a. Ottenere il documento di metadati SAML di esempio da questo URL: https://signin.aws.amazon.com/static/saml-metadata.xml.

   b. Nell'IdP della tua organizzazione (ADFS), genera un file XML di metadati equivalente che descriva il tuo IdP come provider di identità per. AWS Il file di metadati deve includere il nome dell'emittente, la data di creazione, la data di scadenza e le chiavi AWS utilizzate per convalidare le risposte di autenticazione (asserzioni) dell'organizzazione.

2. Nella console &IAM; creare un'entità provider di identità SAML. Per ulteriori informazioni, consulta Creazione di provider di identità SAML nella Guida per l'utente di IAM. Come parte di questo passaggio, eseguire queste operazioni:

   a. Aprire la console IAM all'indirizzo https://console.aws.amazon.com/iam/.

   b. Caricare il documento di metadati SAML generato dall'IdP (ADFS) al punto 1 di questa procedura.

3. Nella console IAM, creare uno o più ruoli IAM per l'IdP. Per ulteriori informazioni, consultare Creazione di un ruolo per un provider di identità di terze parti (federazione) nella Guida per l'utente di IAM. Come parte di questo passaggio, eseguire queste operazioni:

   - Nella policy di autorizzazione del ruolo, elencare le operazioni che gli utenti dell'organizzazione possono effettuare in AWS.
   - Nella policy di affidabilità del ruolo, impostare come principale l'entità provider SAML creata al punto 2 di questa procedura.

   Ciò stabilisce una relazione di fiducia tra l'organizzazione e. AWS

4. Nell'IdP dell'organizzazione (ADFS), definire le asserzioni che associano utenti o gruppi dell'organizzazione ai ruoli IAM. L'associazione di utenti e gruppi ai ruoli IAM è nota anche come regola di attestazione. Si noti che i diversi utenti e gruppi dell'organizzazione potrebbero essere mappati a diversi ruoli IAM.

   Per informazioni sulla configurazione della mappatura in ADFS, consulta il post del blog: Abilitare la federazione all' AWS utilizzo di Windows Active Directory, ADFS e SAML 2.0.

5. Installare e configurare il driver JDBC o ODBC con il supporto di SAML 2.0. Per informazioni, consulta Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

6. Specificare la stringa di connessione dall'applicazione al driver JDBC o ODBC. Per informazioni sulla stringa di connessione che deve essere utilizzata dall'applicazione, consultare l'argomento Utilizzo del provider di credenziali di Active Directory Federation Services (ADFS) nella Guida all'installazione e alla configurazione del driver JDBC o un argomento simile nella Guida all'installazione e alla configurazione del driver ODBC disponibile in formato PDF scaricabile dagli argomenti Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

   Segue un riepilogo generale della configurazione della stringa di connessione ai driver:

1. In `AwsCredentialsProviderClass configuration`, impostare `com.simba.athena.iamsupport.plugin.AdfsCredentialsProvider` per indicare che si desidera utilizzare l'autenticazione basata su SAML 2.0 tramite IdP ADFS.
2. Per `idp_host`, fornire il nome dell'host del server dell'IdP ADFS.
3. Per `idp_port`, fornire il numero di porta che l'IdP ADFS ascolta per la richiesta di asserzione SAML.
4. Per `UID` e `PWD`, fornire le credenziali utente di dominio AD. Quando si utilizza il driver su Windows, se `UID` e `PWD` non vengono forniti, il driver tenta di ottenere le credenziali utente dell'utente che ha effettuato l'accesso al computer Windows.
5. Facoltativamente, impostare `ssl_insecure` su `true`. In questo caso, il driver non controlla l'autenticità del certificato SSL per il server dell'IdP ADFS. L'impostazione di `true` è necessaria se il certificato SSL dell'IdP ADFS non è stato configurato in modo da essere considerato affidabile dal driver.
6. Per abilitare la mappatura di un utente o gruppo di dominio Active Directory a uno o più ruoli IAM (come menzionato al punto 4 di questa procedura), in `preferred_role` per la connessione JDBC o ODBC, specificare il ruolo IAM (ARN) da assumere per la connessione del driver. La specifica di `preferred_role` è facoltativa ed è utile se il ruolo non è il primo ruolo elencato nella regola di attestazione.

Come risultato della procedura, si verificano le seguenti operazioni:

1. Il driver JDBC o ODBC chiama l'API AWS STSAssumeRoleWithSAML e le trasmette le asserzioni, come mostrato nel passaggio 4 del diagramma di architettura.
2. AWS si assicura che la richiesta di assunzione del ruolo provenga dall'IdP a cui si fa riferimento nell'entità del provider SAML.
3. Se la richiesta ha esito positivo, l'operazione API AWS STS AssumeRoleWithSAML restituisce un set di credenziali di sicurezza temporanee, che l'applicazione client utilizza per effettuare richieste firmate ad Athena.

   L'applicazione ha ora le informazioni sull'utente corrente e può accedere ad Athena in modo programmatico.

Logging e monitoraggio in Athena

Per rilevare incidenti, ricevere avvisi quando si verificano incidenti, e rispondere, utilizza queste opzioni con Amazon Athena:

- Monitora Athena con AWS CloudTrail: AWS CloudTrailfornisce un registro delle azioni intraprese da un utente, un ruolo o un utente in Servizio AWS Athena. Consente di acquisire le chiamate dalla console Athena e le chiamate di codice alle operazioni API Athena come eventi. Questo consente di determinare la richiesta effettuata ad Athena, l'indirizzo IP da cui è stata eseguita la richiesta, l'autore della richiesta, il momento in cui è stata eseguita e altri dettagli. Per ulteriori informazioni, consulta Registrazione delle chiamate API Amazon Athena con AWS CloudTrail.

  Puoi anche usare Athena per interrogare i file di CloudTrail registro non solo per Athena, ma anche per altri. Servizi AWS Per ulteriori informazioni, consulta Interrogazione dei log AWS CloudTrail.
- Monitora l'utilizzo di Athena con e CloudTrail Amazon: QuickSight Amazon QuickSight è un servizio di business intelligence completamente gestito e basato sul cloud che ti consente di creare dashboard interattive a cui la tua organizzazione può accedere da qualsiasi dispositivo. Per un esempio di soluzione che utilizza CloudTrail Amazon QuickSight per monitorare l'utilizzo di Athena, consulta il post sul blog AWS Big Data How Realtor.com monitora l'utilizzo di Amazon Athena con e Amazon. AWS CloudTrail QuickSight
- Utilizzo EventBridge con Athena: Amazon EventBridge offre un flusso quasi in tempo reale di eventi di sistema che descrivono i cambiamenti nelle AWS risorse. EventBridge viene a conoscenza dei cambiamenti operativi man mano che si verificano, risponde ad essi e adotta le misure correttive necessarie, inviando messaggi per rispondere all'ambiente, attivando funzioni, apportando modifiche e acquisendo informazioni sullo stato. Gli eventi vengono emessi secondo il principio del massimo sforzo. Per ulteriori informazioni, consulta la sezione Guida introduttiva ad Amazon EventBridge nella Amazon EventBridge User Guide.
- Usa i gruppi di lavoro per separare utenti, team, applicazioni o carichi di lavoro e per impostare limiti di query e controllare i costi delle query: puoi visualizzare i parametri relativi alle query in Amazon CloudWatch, controllare i costi delle query configurando limiti sulla quantità di dati scansionati, creare soglie e attivare azioni, come gli allarmi Amazon SNS, quando queste soglie vengono superate. Per una procedura di alto livello, consulta la sezione Configurazione dei gruppi di lavoro. Utilizzare le autorizzazioni IAM a livello di risorsa per controllare l'accesso a un determinato gruppo di lavoro. Per ulteriori informazioni, consultare Utilizzo dei gruppi di lavoro per l'esecuzione di query e Controllo dei costi e monitoraggio delle interrogazioni con metriche ed eventi CloudWatch .

Argomenti

Registrazione delle chiamate API Amazon Athena con AWS CloudTrail

Athena è integrato con AWS CloudTrail, un servizio che fornisce una registrazione delle azioni intraprese da un utente, un ruolo o un membro di Servizio AWS Athena.

CloudTrail acquisisce tutte le chiamate API per Athena come eventi. Le chiamate acquisite includono le chiamate dalla console Athena e le chiamate di codice alle operazioni delle API Athena. Se crei un trail, puoi abilitare la distribuzione continua di CloudTrail eventi a un bucket Amazon S3, inclusi gli eventi per Athena. Se non configuri un percorso, puoi comunque visualizzare gli eventi più recenti nella CloudTrail console nella cronologia degli eventi.

Utilizzando le informazioni raccolte da CloudTrail, è possibile determinare la richiesta effettuata ad Athena, l'indirizzo IP da cui è stata effettuata la richiesta, chi ha effettuato la richiesta, quando è stata effettuata e dettagli aggiuntivi.

Per ulteriori informazioni CloudTrail, consulta la Guida per l'AWS CloudTrail utente.

È possibile utilizzare Athena per interrogare i file di CloudTrail registro da Athena stessa e da altri. Servizi AWS Per ulteriori informazioni Interrogazione dei log AWS CloudTrailJSON Hive SerDe, consulta il post sul blog AWS Big Data Usa le istruzioni CTAS con Amazon Athena per ridurre i costi e migliorare le prestazioni, che fornisce informazioni sull' CloudTrail utilizzo di Athena.

Informazioni su Athena in CloudTrail

CloudTrail è abilitato sul tuo account Amazon Web Services al momento della creazione dell'account. Quando si verifica un'attività in Athena, tale attività viene registrata in un CloudTrail evento insieme ad altri eventi di AWS servizio nella cronologia degli eventi. Puoi visualizzare, cercare e scaricare gli eventi recenti nell'account Amazon Web Services. Per ulteriori informazioni, consulta Visualizzazione degli eventi con la cronologia degli CloudTrail eventi.

Per una registrazione continua degli eventi nell'account Amazon Web Services che includa gli eventi per Athena, crea un percorso. Un trail consente di CloudTrail inviare file di log a un bucket Amazon S3. Per impostazione predefinita, quando si crea un percorso nella console, questo sarà valido in tutte le Regioni AWS. Il trail registra gli eventi di tutte le regioni della AWS partizione e consegna i file di log al bucket Amazon S3 specificato. Inoltre, puoi configurarne altri Servizi AWS per analizzare

ulteriormente e agire in base ai dati sugli eventi raccolti nei log. CloudTrail Per ulteriori informazioni, consulta gli argomenti seguenti:

- Panoramica della creazione di un percorso
- CloudTrail servizi e integrazioni supportati
- Configurazione delle notifiche Amazon SNS per CloudTrail
- Ricezione di file di CloudTrail registro da più regioni e ricezione di file di CloudTrail registro da più account

Tutte le azioni Athena vengono registrate CloudTrail e documentate nell'Amazon Athena API Reference. Ad esempio, le chiamate alle GetQueryResultsazioni StartQueryExecutione generano voci nei file di registro. CloudTrail

Ogni evento o voce di log contiene informazioni sull'utente che ha generato la richiesta. Le informazioni di identità consentono di determinare quanto segue:

- Se la richiesta è stata effettuata con credenziali utente root o AWS Identity and Access Management (IAM).
- Se la richiesta è stata effettuata con le credenziali di sicurezza temporanee per un ruolo o un utente federato.
- Se la richiesta è stata effettuata da un altro Servizio AWS.

Per ulteriori informazioni, vedete l'elemento CloudTrail userIdentity.

Informazioni sulle voci di file di log di Athena

Un trail è una configurazione che consente la distribuzione di eventi come file di log in un bucket Amazon S3 specificato dall'utente. CloudTrail i file di registro contengono una o più voci di registro. Un evento rappresenta una singola richiesta proveniente da qualsiasi fonte e include informazioni sull'azione richiesta, la data e l'ora dell'azione, i parametri della richiesta e così via. CloudTrail i file di registro non sono una traccia ordinata dello stack delle chiamate API pubbliche, quindi non vengono visualizzati in un ordine specifico.

> **Note**
>
> Per evitare la divulgazione involontaria di informazioni riservate, la voce `queryString` in entrambi i log `StartQueryExecution` e `CreateNamedQuery` ha un valore di

`***OMITTED***`. Si tratta di un'impostazione predefinita. Per accedere alla stringa di query effettiva, puoi utilizzare l'GetQueryExecutionAPI Athena e passare il valore di `responseElements.queryExecutionId` dal CloudTrail log.

Gli esempi seguenti mostrano le voci di CloudTrail registro per:

- StartQueryExecution(Riuscito)
- StartQueryExecution (Fallito)
- CreateNamedQuery

StartQueryExecution (riuscito)


```
{
 "eventVersion":"1.05",
 "userIdentity":{
    "type":"IAMUser",
    "principalId":"EXAMPLE_PRINCIPAL_ID",
    "arn":"arn:aws:iam::123456789012:user/johndoe",
    "accountId":"123456789012",
    "accessKeyId":"EXAMPLE_KEY_ID",
    "userName":"johndoe"
 },
 "eventTime":"2017-05-04T00:23:55Z",
 "eventSource":"athena.amazonaws.com",
 "eventName":"StartQueryExecution",
 "awsRegion":"us-east-1",
 "sourceIPAddress":"77.88.999.69",
 "userAgent":"aws-internal/3",
 "requestParameters":{
    "clientRequestToken":"16bc6e70-f972-4260-b18a-db1b623cb35c",
    "resultConfiguration":{
       "outputLocation":"s3://DOC-EXAMPLE-BUCKET/test/"
    },
    "queryString":"***OMITTED***"
 },
 "responseElements":{
    "queryExecutionId":"b621c254-74e0-48e3-9630-78ed857782f9"
 },
 "requestID":"f5039b01-305f-11e7-b146-c3fc56a7dc7a",
 "eventID":"c97cf8c8-6112-467a-8777-53bb38f83fd5",
```

```
 "eventType":"AwsApiCall",
 "recipientAccountId":"123456789012"
}
```

StartQueryExecution (fallito)

```
{
 "eventVersion":"1.05",
 "userIdentity":{
  "type":"IAMUser",
  "principalId":"EXAMPLE_PRINCIPAL_ID",
  "arn":"arn:aws:iam::123456789012:user/johndoe",
  "accountId":"123456789012",
  "accessKeyId":"EXAMPLE_KEY_ID",
  "userName":"johndoe"
  },
 "eventTime":"2017-05-04T00:21:57Z",
 "eventSource":"athena.amazonaws.com",
 "eventName":"StartQueryExecution",
 "awsRegion":"us-east-1",
 "sourceIPAddress":"77.88.999.69",
 "userAgent":"aws-internal/3",
 "errorCode":"InvalidRequestException",
 "errorMessage":"Invalid result configuration. Should specify either output location or
 result configuration",
 "requestParameters":{
  "clientRequestToken":"ca0e965f-d6d8-4277-8257-814a57f57446",
  "queryString":"***OMITTED***"
  },
 "responseElements":null,
 "requestID":"aefbc057-305f-11e7-9f39-bbc56d5d161e",
 "eventID":"6e1fc69b-d076-477e-8dec-024ee51488c4",
 "eventType":"AwsApiCall",
 "recipientAccountId":"123456789012"
}
```

CreateNamedQuery

```
{
  "eventVersion":"1.05",
  "userIdentity":{
     "type":"IAMUser",
     "principalId":"EXAMPLE_PRINCIPAL_ID",
```

```
      "arn":"arn:aws:iam::123456789012:user/johndoe",
      "accountId":"123456789012",
      "accessKeyId":"EXAMPLE_KEY_ID",
      "userName":"johndoe"
   },
   "eventTime":"2017-05-16T22:00:58Z",
   "eventSource":"athena.amazonaws.com",
   "eventName":"CreateNamedQuery",
   "awsRegion":"us-west-2",
   "sourceIPAddress":"77.88.999.69",
   "userAgent":"aws-cli/1.11.85 Python/2.7.10 Darwin/16.6.0 botocore/1.5.48",
   "requestParameters":{
      "name":"johndoetest",
      "queryString":"***OMITTED***",
      "database":"default",
      "clientRequestToken":"fc1ad880-69ee-4df0-bb0f-1770d9a539b1"
      },
   "responseElements":{
      "namedQueryId":"cdd0fe29-4787-4263-9188-a9c8db29f2d6"
      },
   "requestID":"2487dd96-3a83-11e7-8f67-c9de5ac76512",
   "eventID":"15e3d3b5-6c3b-4c7c-bc0b-36a8dd95227b",
   "eventType":"AwsApiCall",
   "recipientAccountId":"123456789012"
},
```


Convalida della conformità per Amazon Athena

Revisori di terze parti valutano la sicurezza e la conformità di Amazon Athena come parte di più programmi di conformità di AWS. Questi includono SOC, PCI, FedRAMP e altri.

Per un elenco di Servizi AWS che rientrano nell'ambito di programmi di conformità specifici, consulta Servizi AWS coperti dal programma di conformità. Per informazioni generali, consultare Programmi per la conformità di AWS.

Puoi scaricare i report di audit di terze parti utilizzando AWS Artifact. Per ulteriori informazioni, consulta l'argomento Download dei rapporti in AWS Artifact.

La responsabilità per la conformità quando utilizzi Athena è determinata dalla riservatezza dei dati, dagli obiettivi di conformità dell'azienda e dalle normative vigenti. AWS fornisce le risorse seguenti per semplificare la conformità:

- Guide Quick Start per la sicurezza e conformità: queste guide all'implementazione illustrano considerazioni relative all'architettura e forniscono fasi per l'implementazione di ambienti di base incentrati sulla sicurezza e sulla conformità su AWS.
- Architettare per la sicurezza e la conformità HIPAA su Amazon Web Services: questo whitepaper descrive come le aziende possono utilizzare AWS per creare applicazioni conformi alla normativa HIPAA.
- Risorse per la conformità di AWS: questa raccolta di workbook e guide potrebbe essere utile al tuo settore e alla tua posizione.
- AWS Config: questo Servizio AWS valuta il livello di conformità delle configurazioni delle risorse con pratiche interne, linee guida e regolamenti di settore.
- AWS Security Hub: questo Servizio AWS fornisce una visione completa dello stato di sicurezza all'interno di AWS che consente di verificare la conformità con gli standard e le best practice di sicurezza del settore.

Resilienza in Athena

L'infrastruttura AWS globale è costruita attorno a Regioni AWS zone di disponibilità. Regioni AWS forniscono più zone di disponibilità fisicamente separate e isolate, collegate con reti a bassa latenza, ad alto throughput e altamente ridondanti. Con le zone di disponibilità, è possibile progettare e gestire applicazioni e database che eseguono il failover automatico tra zone di disponibilità senza interruzioni. Le zone di disponibilità sono più disponibili, tolleranti ai guasti e scalabili rispetto alle infrastrutture tradizionali a data center singolo o multiplo.

Per ulteriori informazioni sulle zone di disponibilità, Regioni AWS consulta infrastruttura globale.AWS

Oltre all'infrastruttura AWS globale, Athena offre diverse funzionalità per supportare le esigenze di resilienza e backup dei dati.

Athena è un servizio serverless, perciò non è necessario configurare o gestire alcuna infrastruttura. Athena è altamente disponibile ed esegue le query utilizzando le risorse di elaborazione in più zone di disponibilità instradando automaticamente le query nel modo appropriato se una determinata zona di disponibilità non è raggiungibile. Athena utilizza Amazon S3 come archivio dati sottostante, rendendo i dati estremamente disponibili e durevoli. Amazon S3 fornisce un'infrastruttura durevole per archiviare dati importanti ed è progettato per una durabilità degli oggetti pari al 99,999999999%. I dati vengono archiviati in modo ridondante in più strutture e in più dispositivi all'interno di ogni struttura.

Sicurezza dell'infrastruttura in Athena

In quanto servizio gestito, Amazon Athena è protetto dalla sicurezza di rete AWS globale. Per informazioni sui servizi di AWS sicurezza e su come AWS protegge l'infrastruttura, consulta AWS Cloud Security. Per progettare il tuo AWS ambiente utilizzando le migliori pratiche per la sicurezza dell'infrastruttura, vedi Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Utilizzi chiamate API AWS pubblicate per accedere ad Athena attraverso la rete. I client devono supportare quanto segue:

- Transport Layer Security (TLS). È richiesto TLS 1.2 ed è consigliato TLS 1.3.
- Suite di cifratura con Perfect Forward Secrecy (PFS), ad esempio Ephemeral Diffie-Hellman (DHE) o Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La maggior parte dei sistemi moderni, come Java 7 e versioni successive, supporta tali modalità.

Inoltre, le richieste devono essere firmate utilizzando un ID chiave di accesso e una chiave di accesso segreta associata a un principale IAM. O puoi utilizzare AWS Security Token Service (AWS STS) per generare credenziali di sicurezza temporanee per sottoscrivere le richieste.

Usa le policy IAM per limitare l'accesso alle operazioni Athena. Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Le policy gestite da Athena sono facili da utilizzare e vengono aggiornate automaticamente con le operazioni richieste con l'evolvere del servizio. Le policy gestite dal cliente e le policy inline consentono di ottimizzare le policy specificando operazioni Athena più granulari all'interno della policy. Concedere l'accesso appropriato alla posizione Amazon S3 dei dati. Per informazioni dettagliate e gli scenari su come concedere l'accesso Amazon S3, consulta Procedure guidate di esempio: gestione dell'accesso nella Guida per gli sviluppatori di Amazon Simple Storage Service. Per ulteriori informazioni e un esempio di quali operazioni Amazon S3 consentire, consulta la policy del bucket di esempio in accesso multi-account.

Argomenti

- Connessione ad Amazon Athena utilizzando un endpoint VPC di interfaccia

Connessione ad Amazon Athena utilizzando un endpoint VPC di interfaccia

Puoi migliorare la posizione di sicurezza del VPC utilizzando un endpoint VPC di interfaccia (AWS PrivateLink) e un endpoint VPC di AWS Glue nel cloud privato virtuale (VPC). Un endpoint VPC di interfaccia migliora il livello di sicurezza offrendoti la possibilità di verificare quali destinazioni possono essere raggiunte dall'interno del VPC. Ogni endpoint VPC è rappresentato da una o più interfacce di rete elastiche (ENI) con indirizzi IP privati nelle sottoreti del VPC.

L'interfaccia VPC endpoint collega il tuo VPC direttamente ad Athena senza un gateway Internet, un dispositivo NAT, una connessione VPN o una connessione. AWS Direct Connect Le istanze presenti nel tuo VPC non richiedono indirizzi IP pubblici per comunicare con l'API Athena.

Per usare Athena tramite il VPC, devi connetterti da un'istanza che si trova all'interno del VPC o connettere la rete privata al VPC usando Amazon Virtual Private Network (VPN) o AWS Direct Connect. Per informazioni su Amazon VPN, consulta Connessioni VPN nella Guida per l'utente di Amazon Virtual Private Cloud. Per informazioni su AWS Direct Connect, consulta Creazione di una connessione nella Guida per l'utente.AWS Direct Connect

Athena supporta gli endpoint VPC ovunque siano disponibili sia Regioni AWS Amazon VPC che Athena.

È possibile creare un endpoint VPC di interfaccia per connettersi ad Athena utilizzando i AWS Management Console comandi or (). AWS Command Line Interface AWS CLI Per ulteriori informazioni, consulta Creazione di un endpoint di interfaccia.

Se dopo aver creato un endpoint VPC di interfaccia abiliti nomi host DNS privati per l'endpoint, l'endpoint di default di Athena (https://athena.*Region*.amazonaws.com) restituisce il tuo endpoint VPC.

Se non abiliti nomi host DNS privati, Amazon VPC fornisce un nome di endpoint DNS che puoi utilizzare nel formato seguente:

```
VPC_Endpoint_ID.athena.Region.vpce.amazonaws.com
```

Per ulteriori informazioni, consulta Interface VPC endpoints (AWS PrivateLink) nella Amazon VPC User Guide.

Athena supporta l'esecuzione di chiamate a tutte le sue operazioni API all'interno del VPC.

Creazione di una policy di endpoint VPC per Athena

Puoi creare una policy per gli endpoint VPC di Amazon per Athena per specificare delle restrizioni come quelle seguenti:

- Principale: il principale che può eseguire operazioni.
- Operazioni: le operazioni che possono essere eseguite.
- Risorse: le risorse sui cui si possono eseguire operazioni.
- Solo identità affidabili: utilizza la `aws:PrincipalOrgId` condizione per limitare l'accesso solo alle credenziali che fanno parte della tua organizzazione. AWS Questo può aiutare a impedire l'accesso da parte di principali non desiderati.
- Solo risorse affidabili: utilizza la condizione `aws:ResourceOrgId` per impedire l'accesso a risorse non desiderate.
- Solo identità e risorse affidabili: crea una policy combinata per un endpoint VPC che aiuti a impedire l'accesso a principali e risorse non desiderate.

Per ulteriori informazioni, consulta Controllare l'accesso ai servizi con endpoint VPC nella Amazon VPC User Guide e Appendice 2 — Esempi di policy per gli endpoint VPC nel white paper Building a data perimeter on. AWS AWS

Example – Policy degli endpoint VPC

L'esempio seguente consente le richieste in base alle identità dell'organizzazione alle risorse dell'organizzazione e consente le richieste dei responsabili del servizio. AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRequestsByOrgsIdentitiesToOrgsResources",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
```

```
                    "aws:PrincipalOrgID": "my-org-id",
                    "aws:ResourceOrgID": "my-org-id"
                }
            }
        },
        {
            "Sid": "AllowRequestsByAWSServicePrincipals",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "Bool": {
                    "aws:PrincipalIsAWSService": "true"
                }
            }
        }
    ]
}
```

Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Sottoreti condivise

Non puoi creare, descrivere, modificare o eliminare gli endpoint VPC nelle sottoreti condivise con te. Tuttavia, puoi utilizzare gli endpoint VPC in sottoreti condivise con te. Per informazioni sulla condivisione VPC, consulta la pagina Condivisione del VPC con altri account nella Guida per l'utente di Amazon VPC.

Analisi della configurazione e delle vulnerabilità in Athena

Athena è serverless, quindi non è necessaria alcuna infrastruttura da configurare o gestire. AWS gestisce le attività di sicurezza di base, come l'applicazione di patch al sistema operativo (OS) guest e al database, la configurazione del firewall e il disaster recovery. Queste procedure sono state riviste e certificate dalle terze parti appropriate. Per ulteriori dettagli, consulta le seguenti AWS risorse:

- Modello di responsabilità condivisa
- Best practice per sicurezza, identità e conformità.

Utilizzo di Athena per eseguire query sui dati registrati con AWS Lake Formation

AWS Lake Formation ti consente di definire e consolidare le policy di accesso a livello di database, tabella e colonna quando usi le query Athena per leggere i dati archiviati in Amazon S3. Lake Formation fornisce un livello di autorizzazione e governance sui dati archiviati in Amazon S3. Puoi utilizzare una gerarchia di autorizzazioni in Lake Formation per concedere o revocare le autorizzazioni per leggere oggetti del catalogo dati, ad esempio database, tabelle e colonne. Lake Formation semplifica la gestione delle autorizzazioni e consente di implementare il controllo granulare degli accessi (FGAC) per i dati.

Puoi utilizzare Athena per eseguire query sui dati registrati con Lake Formation e sui dati non registrati con Lake Formation.

Le autorizzazioni Lake Formation si applicano quando utilizzi Athena per eseguire query sui dati di origine dalle posizioni Amazon S3 registrate con Lake Formation. Le autorizzazioni Lake Formation si applicano anche quando si creano database e tabelle che puntano a posizioni di dati Amazon S3 registrate. Per utilizzare Athena con dati registrati utilizzando Lake Formation, Athena deve essere configurato per utilizzare AWS Glue Data Catalog.

Le autorizzazioni Lake Formation non si applicano durante la scrittura di oggetti su Amazon S3 né durante l'esecuzione di query sui dati archiviati in Amazon S3 o sui metadati che non sono registrati con Lake Formation. Per i dati di origine in Amazon S3 e i metadati non registrati con Lake Formation, l'accesso è determinato dalle politiche di autorizzazione IAM per Amazon S3 e dalle azioni. AWS Glue Le posizioni dei risultati delle query Athena in Amazon S3 non possono essere registrate con Lake Formation e le policy di autorizzazione IAM per Amazon S3 controllano l'accesso. Inoltre, le autorizzazioni Lake Formation non si applicano alla cronologia delle query Athena. Puoi utilizzare i gruppi di lavoro Athena per controllare l'accesso alla cronologia delle query.

Per altre informazioni su Lake Formation, consulta le Domande frequenti su Lake Formation e la Guida per gli sviluppatori di AWS Lake Formation.

Argomenti

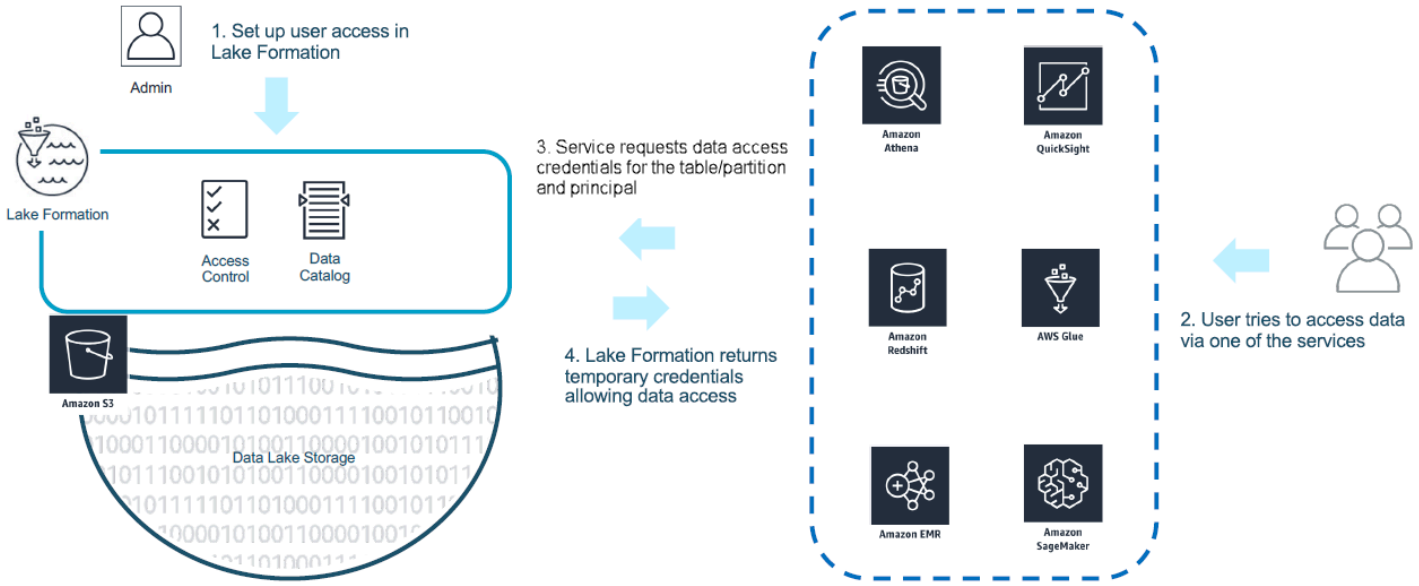
- Utilizzo di Lake Formation e dei driver Athena JDBC e ODBC per l'accesso federato ad Athena

Come Athena accede ai dati registrati con Lake Formation

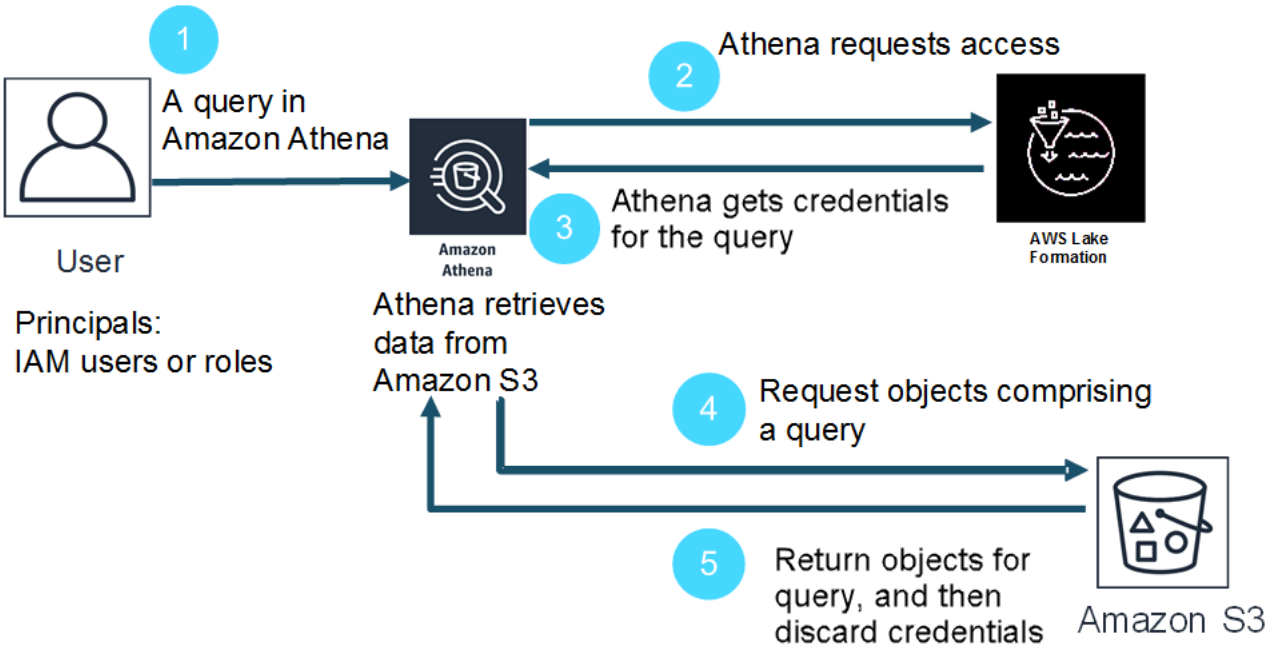
Il flusso di lavoro di accesso descritto in questa sezione si applica solo quando si eseguono query Athena su posizioni e oggetti metadati Amazon S3 registrati con Lake Formation. Per ulteriori informazioni, consulta Registrazione di un data lake nella Guida per gli sviluppatori di AWS Lake Formation . Oltre alla registrazione dei dati, l'amministratore Lake Formation applica le autorizzazioni Lake Formation che concedono o revocano l'accesso ai metadati nel catalogo dati e la posizione dei dati in Amazon S3. Per ulteriori informazioni, consulta Security and Access Control to Metadata and Data (Sicurezza e controllo degli accessi ai metadati e ai dati) nella Guida per gli sviluppatori di AWS Lake Formation .

Ogni volta che un'entità principale Athena (utente, gruppo o ruolo) esegue una query sui dati registrati utilizzando Lake Formation, Lake Formation verifica che l'entità principale disponga delle autorizzazioni Lake Formation appropriate per il database, la tabella e la posizione Amazon S3 appropriati per la query. Se l'entità principale ha accesso, Lake Formation fornisce credenziali temporanee ad Athena e la query viene eseguita.

Il seguente diagramma illustra il flusso descritto sopra.



Il diagramma seguente mostra come funziona il vending di credenziali in Athena sulla query-by-query base di un'ipotetica `SELECT` query su una tabella con una sede Amazon S3 registrata a Lake Formation:



1. Un'entità principale esegue una query SELECT in Athena.
2. Athena analizza la query e controlla le autorizzazioni Lake Formation per vedere se all'entità principale è stato concesso l'accesso alla tabella e alle colonne della tabella.
3. Se l'entità principale ha accesso, Athena richiede le credenziali da Lake Formation. Se l'entità principale non ha accesso, Athena invia un errore di accesso negato.
4. Lake Formation rilascia ad Athena le credenziali da utilizzare durante la lettura dei dati da Amazon S3, insieme all'elenco delle colonne consentite.
5. Athena utilizza le credenziali temporanee di Lake Formation per eseguire query sui dati da Amazon S3. Una volta completata la query, Athena elimina le credenziali.

Considerazioni e limitazioni sull'utilizzo di Athena per eseguire query sui dati registrati con Lake Formation

Considera quanto segue quando utilizzi Athena per eseguire query sui dati registrati in Lake Formation. Per altre informazioni, consulta Problemi noti per AWS Lake Formation nella Guida per gli sviluppatori di AWS Lake Formation .

Considerazioni e limitazioni

- Metadati delle colonne visibili agli utenti non autorizzati in alcune circostanze con Avro e custom SerDe

- Utilizzo delle autorizzazioni Lake Formation sulle visualizzazioni
- Controllo granulare degli accessi di Lake Formation e gruppi di lavoro Athena
- Posizione dei risultati delle query Athena in Amazon S3 non registrate con Lake Formation
- Utilizzo dei gruppi di lavoro Athena per limitare l'accesso alla cronologia delle query
- Accesso al catalogo dati multi-account
- Sedi Amazon S3 crittografate con CSE-KMS registrate presso Lake Formation
- Le posizioni dei dati partizionati registrate con Lake Formation devono essere nelle sottodirectory della tabella
- Le query Create Table As Select (CTAS) richiedono autorizzazioni di scrittura Amazon S3.
- L'autorizzazione DESCRIBE è necessaria nel database di default.

Metadati delle colonne visibili agli utenti non autorizzati in alcune circostanze con Avro e custom SerDe

L'autorizzazione a livello di colonna di Lake Formation impedisce agli utenti di accedere ai dati nelle colonne per le quali l'utente non dispone delle autorizzazioni Lake Formation. Tuttavia, in alcune situazioni, gli utenti sono in grado di accedere ai metadati che descrivono tutte le colonne della tabella, incluse le colonne per le quali non dispongono delle autorizzazioni per i dati.

Ciò si verifica quando i metadati delle colonne vengono archiviati nelle proprietà della tabella per le tabelle che utilizzano il formato di archiviazione Apache Avro o utilizzano un Serializer/Deserializer personalizzato (SerDe) in cui lo schema della tabella è definito nelle proprietà della tabella insieme alla definizione. SerDe Quando utilizzi Athena con Lake Formation, consigliamo di esaminare i contenuti delle proprietà della tabella che registri con Lake Formation e, se possibile, limitare le informazioni archiviate nelle proprietà della tabella per evitare che i metadati sensibili siano visibili agli utenti.

Utilizzo delle autorizzazioni Lake Formation sulle visualizzazioni

Per i dati registrati con Lake Formation, un utente Athena può creare una `VIEW` solo se dispone delle autorizzazioni Lake Formation per le tabelle, le colonne e le posizioni dei dati di origine Amazon S3 su cui si basa `VIEW`. Dopo aver creato una `VIEW` in Athena, le autorizzazioni Lake Formation possono essere applicate alla `VIEW`. Le autorizzazioni a livello di colonna non sono disponibili per `VIEW`. Gli utenti che dispongono delle autorizzazioni Lake Formation per una `VIEW` ma non dispongono delle autorizzazioni per la tabella e le colonne su cui era basata la visualizzazione non sono in grado di utilizzare la `VIEW` per eseguire query sui dati. Tuttavia, gli utenti con questa

combinazione di autorizzazioni sono in grado di utilizzare istruzioni come `DESCRIBE VIEW`, `SHOW CREATE VIEW`, e `SHOW COLUMNS` per visualizzare i metadati `VIEW`. Per questo motivo, assicurati di allineare le autorizzazioni Lake Formation per ogni `VIEW` con le autorizzazioni della tabella sottostante. I filtri delle celle definiti su una tabella non si applicano a `VIEW` per tale tabella. I nomi dei link alle risorse devono avere lo stesso nome della risorsa nell'account di origine. Esistono limitazioni aggiuntive quando si lavora con le viste in una configurazione tra più account. Per ulteriori informazioni su come importare le autorizzazioni per le visualizzazioni condivise tra account, consulta Accesso al catalogo dati multi-account.

Controllo granulare degli accessi di Lake Formation e gruppi di lavoro Athena

Gli utenti dello stesso gruppo di lavoro Athena possono visualizzare i dati che il controllo granulare degli accessi di Lake Formation ha configurato come accessibili da parte del gruppo di lavoro. Per ulteriori informazioni sull'utilizzo del controllo granulare degli accessi in Lake Formation, consulta l'articolo Gestone del controllo granulare degli accessi utilizzando AWS Lake Formation nel blog sui big data di AWS .

Posizione dei risultati delle query Athena in Amazon S3 non registrate con Lake Formation

Le posizioni dei risultati della query in Amazon S3 per Athena non possono essere registrate con Lake Formation. Le autorizzazioni Lake Formation non limitano l'accesso a queste posizioni. A meno che non si limiti l'accesso, gli utenti Athena possono accedere ai file dei risultati delle query e ai metadati quando non dispongono delle autorizzazioni Lake Formation per i dati. Per evitare questo problema, è consigliabile utilizzare i gruppi di lavoro per specificare la posizione dei risultati delle query e allineare l'appartenenza al gruppo di lavoro alle autorizzazioni Lake Formation. È quindi possibile utilizzare i criteri di autorizzazione IAM per limitare l'accesso ai percorsi dei risultati delle query Per ulteriori informazioni sui risultati delle query, consulta Utilizzo dei risultati delle query, delle query recenti e dei file di output.

Utilizzo dei gruppi di lavoro Athena per limitare l'accesso alla cronologia delle query

La cronologia delle query Athena espone un elenco di query salvate e stringhe di query complete. A meno che non si utilizzino gruppi di lavoro per separare l'accesso alle cronologie delle query, gli utenti di Athena che non sono autorizzati a eseguire query sui dati in Lake Formation possono visualizzare le stringhe di query eseguite su tali dati, inclusi i nomi delle colonne, i criteri di selezione ecc. È consigliabile utilizzare i gruppi di lavoro per separare le cronologie delle query e allineare l'appartenenza al gruppo di lavoro Athena alle autorizzazioni Lake Formation per limitare gli accessi. Per ulteriori informazioni, consulta Uso dei gruppi di lavoro per controllare l'accesso alle query e i costi.

Accesso al catalogo dati multi-account

Per accedere a un catalogo dati in un altro account, puoi utilizzare la funzionalità AWS Glue multi-account di Athena o imposta l'accesso tra account in Lake Formation.

Accesso al catalogo dati multi-account

Puoi utilizzare la funzionalità di catalogo AWS Glue multiaccount di Athena per registrare il catalogo nel tuo account. Questa funzionalità è disponibile solo nella versione 2 del motore Athena e versioni successive ed è limitata all'uso della stessa regione tra gli account. Per ulteriori informazioni, consulta Registrazione e AWS Glue Data Catalog da un altro account.

Se il Catalogo dati da condividere ha una politica delle risorse configurata in AWS Glue, questa deve essere aggiornata per consentire l'accesso AWS Resource Access Manager e concedere all'Account B le autorizzazioni per utilizzare il Catalogo dati dell'Account A, come nell'esempio seguente.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Principal": {
            "Service": "ram.amazonaws.com"
        },
        "Action": "glue:ShareResource",
        "Resource": [
          "arn:aws:glue:<REGION>:<ACCOUNT-A>:table/*/*",
          "arn:aws:glue:<REGION>:<ACCOUNT-A>:database/*",
          "arn:aws:glue:<REGION>:<ACCOUNT-A>:catalog"
        ]
    },
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::<ACCOUNT-B>:root"
        },
        "Action": "glue:*",
        "Resource": [
          "arn:aws:glue:<REGION>:<ACCOUNT-A>:table/*/*",
          "arn:aws:glue:<REGION>:<ACCOUNT-A>:database/*",
          "arn:aws:glue:<REGION>:<ACCOUNT-A>:catalog"
        ]
    }
    ]
```

```
}
```

Per ulteriori informazioni, consulta Accesso tra account ai cataloghi dati AWS Glue.

Configurazione dell'accesso multi-account in Lake Formation

AWS Lake Formation consente di utilizzare un singolo account per gestire un Data Catalog centralizzato. Puoi usare questa funzione per implementare l'accesso tra account nei metadati del Catalogo dati e nei dati sottostanti. Ad esempio, un account proprietario può concedere a un altro account (destinatario) l'autorizzazione `SELECT` per una tabella.

Affinché un database o una tabella condivisa appaia nell'Editor di query Athena, in Lake Formation crea un collegamento di risorse al database o alla tabella condivisa. Quando l'account del destinatario in Lake Formation interroga la tabella del proprietario, CloudTrailaggiunge l'evento di accesso ai dati ai registri sia per l'account del destinatario che per l'account del proprietario.

Per le visualizzazioni condivise, è importante considerare quanto segue:

- Le query vengono eseguite sui collegamenti alle risorse di destinazione, non sulla tabella o sulla visualizzazione di origine e quindi l'output viene condiviso con l'account di destinazione.
- Non è sufficiente condividere solo la visualizzazione. Tutte le tabelle coinvolte nella creazione della visualizzazione devono far parte della condivisione multi-account.
- Per una visualizzazione condivisa, il nome del collegamento alla risorsa deve corrispondere al nome della risorsa nell'account del proprietario. Se il nome non corrisponde, viene visualizzato un messaggio di errore come «Failed analyzing stored view» («awsdatacatalog»). *`my-lf-resource-link`*. *`my-lf-view`*': riga 3:3: si verifica lo schema *`schema_name`* not exist.

Per ulteriori informazioni sull'accesso multi-account in Lake Formation, consulta le informazioni seguenti nella Guida per gli sviluppatori di AWS Lake Formation :

Accesso multi-account

Come funzionano i link alle risorse in Lake Formation

Registrazione tra più account CloudTrail

Sedi Amazon S3 crittografate con CSE-KMS registrate presso Lake Formation

Le tabelle Open Table Format (OTF) come Apache Iceberg che presentano le seguenti caratteristiche non possono essere interrogate con Athena:

- Le tabelle si basano sulle posizioni dei dati di Amazon S3 registrate con Lake Formation.
- Gli oggetti in Amazon S3 sono crittografati utilizzando la crittografia lato client (CSE).
- La crittografia utilizza chiavi gestite dal AWS KMS cliente (). `CSE_KMS`

Per interrogare tabelle non OTF (crittografate con una `CSE_KMS` chiave), aggiungete il seguente blocco alla politica della AWS KMS chiave utilizzata per la crittografia CSE. <KMS_KEY_ARN>è l'ARN della AWS KMS chiave che crittografa i dati. <IAM-ROLE-ARN>è l'ARN del ruolo IAM che registra la posizione di Amazon S3 in Lake Formation.

```
{
    "Sid": "Allow use of the key",
    "Effect": "Allow",
    "Principal": {
        "AWS": "*"
    },
    "Action": "kms:Decrypt",
    "Resource": "<KMS-KEY-ARN>",
    "Condition": {
        "ArnLike": {
            "aws:PrincipalArn": "<IAM-ROLE-ARN>"
        }
    }
}
```

Le posizioni dei dati partizionati registrate con Lake Formation devono essere nelle sottodirectory della tabella

Le tabelle partizionate registrate con Lake Formation devono avere dati partizionati nelle directory che sono sottodirectory della tabella in Amazon S3. Ad esempio, una tabella con la posizione `s3://DOC-EXAMPLE-BUCKET/mytable` e le partizioni `s3://DOC-EXAMPLE-BUCKET/mytable/dt=2019-07-11`, `s3://DOC-EXAMPLE-BUCKET/mytable/dt=2019-07-12` e così via può essere registrata con Lake Formation ed è possibile eseguire query utilizzando Athena. D'altra parte, una tabella con la posizione `s3://DOC-EXAMPLE-BUCKET/mytable` e le partizioni situate in `s3://DOC-EXAMPLE-BUCKET/dt=2019-07-11`, `s3://DOC-EXAMPLE-BUCKET/dt=2019-07-12` e così via, non può essere registrata con Lake Formation. Poiché tali partizioni non sono sottodirectory di `s3://DOC-EXAMPLE-BUCKET/mytable`, non possono essere lette da Athena.

Le query Create Table As Select (CTAS) richiedono autorizzazioni di scrittura Amazon S3.

Create Table As Statements (CTAS) richiede l'accesso in scrittura alla posizione Amazon S3 delle tabelle. Per eseguire query CTAS sui dati registrati con Lake Formation, gli utenti di Athena devono disporre delle autorizzazioni IAM per scrivere nelle posizioni Amazon S3 delle tabelle, oltre alle autorizzazioni di Lake Formation appropriate per leggere le posizioni dei dati. Per ulteriori informazioni, consulta Creazione di una tabella dai risultati delle query (CTAS).

L'autorizzazione DESCRIBE è necessaria nel database di default.

L'autorizzazione DESCRIBE di Lake Formation è necessaria nel database `default` Il AWS CLI comando di esempio seguente concede l'`DESCRIBE`autorizzazione sul `default` database all'utente dell'account. `datalake_user1` AWS `111122223333`

```
aws lakeformation grant-permissions --principal
 DataLakePrincipalIdentifier=arn:aws:iam::111122223333:user/datalake_user1 --
permissions "DESCRIBE" --resource '{ "Database": {"Name":"default"}}
```

Per ulteriori informazioni, consulta Documentazione di riferimento sulle autorizzazioni Lake Formation nella Guida per gli sviluppatori di AWS Lake Formation .

Gestione delle autorizzazioni utente Lake Formation e Athena

Lake Formation fornisce le credenziali per eseguire query sui datastore Amazon S3 registrati con Lake Formation. Se in precedenza hai utilizzato le policy IAM per concedere o negare le autorizzazioni di lettura delle posizioni dei dati in Amazon S3, puoi utilizzare le autorizzazioni Lake Formation. Tuttavia, sono ancora necessarie altre autorizzazioni IAM.

Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Le sezioni seguenti riepilogano le autorizzazioni necessarie per utilizzare Athena per eseguire query sui dati registrati in Lake Formation. Per ulteriori informazioni, consulta Sicurezza in AWS Lake Formation nella Guida per gli sviluppatori AWS Lake Formation .

Riepilogo delle autorizzazioni

- Autorizzazioni basate su identità per Lake Formation e Athena
- Autorizzazioni Amazon S3 per le posizioni dei risultati delle query Athena
- Appartenenza dei gruppi di lavoro Athena alla cronologia delle query

- Autorizzazioni Lake Formation ai dati
- Autorizzazioni IAM per scrivere nelle posizioni Amazon S3
- Autorizzazioni per dati crittografati, metadati e risultati delle query Athena
- Autorizzazioni basate su risorse per i bucket Amazon S3 negli account esterni (facoltativo)

Autorizzazioni basate su identità per Lake Formation e Athena

Chiunque utilizzi Athena per eseguire query sui dati registrati con Lake Formation deve disporre di una policy di autorizzazione IAM che consente l'azione `lakeformation:GetDataAccess`. AWS politica gestita: AmazonAthenaFullAccess consente questa operazione. Se utilizzi policy inline, assicurati di aggiornare le policy di autorizzazione per consentire questa operazione.

In Lake Formation, un amministratore di data lake dispone delle autorizzazioni per creare oggetti metadati come database e tabelle, concedere autorizzazioni Lake Formation ad altri utenti e registrare nuove posizioni Amazon S3. Per registrare nuove posizioni, sono necessarie le autorizzazioni per il ruolo collegato ai servizi per Lake Formation. Per ulteriori informazioni, consulta Creare un amministratore di Data Lake e Autorizzazioni dei ruoli collegati ai servizi per Lake Formation nella Guida per gli sviluppatori di AWS Lake Formation .

Un utente Lake Formation può utilizzare Athena per eseguire query su database, tabelle, colonne di tabelle e datastore Amazon S3 sottostanti in base alle autorizzazioni LakeFormation concesse dagli amministratori di data lake. Gli utenti non possono creare database o tabelle o registrare nuove posizioni Amazon S3 con Lake Formation. Per ulteriori informazioni, consulta Create a Data Lake User (Creazione di un utente di data lake) nella Guida per gli sviluppatori di AWS Lake Formation .

In Athena, le policy di autorizzazione basate su identità, incluse quelle per i gruppi di lavoro Athena, controllano comunque l'accesso alle azioni Athena per gli utenti dell'account Amazon Web Services. Inoltre, l'accesso federato potrebbe essere fornito tramite l'autenticazione basata su SAML disponibile con i driver Athena. Per ulteriori informazioni, consulta Uso dei gruppi di lavoro per controllare l'accesso alle query e i costi, Policy IAM per l'accesso ai gruppi di lavoro e Abilitazione dell'accesso federato all'API Athena.

Per ulteriori informazioni, consulta Concedere autorizzazioni Lake Formation nella Guida per gli sviluppatori di AWS Lake Formation .

Autorizzazioni Amazon S3 per le posizioni dei risultati delle query Athena

Le posizioni dei risultati della query in Amazon S3 per Athena non possono essere registrate con Lake Formation. Le autorizzazioni Lake Formation non limitano l'accesso a queste posizioni. A meno

che non si limiti l'accesso, gli utenti Athena possono accedere ai file dei risultati delle query e ai metadati quando non dispongono delle autorizzazioni Lake Formation per i dati. Per evitare questo problema, è consigliabile utilizzare i gruppi di lavoro per specificare la posizione dei risultati delle query e allineare l'appartenenza al gruppo di lavoro alle autorizzazioni Lake Formation. È quindi possibile utilizzare i criteri di autorizzazione IAM per limitare l'accesso ai percorsi dei risultati delle query Per ulteriori informazioni sui risultati delle query, consulta Utilizzo dei risultati delle query, delle query recenti e dei file di output.

Appartenenza dei gruppi di lavoro Athena alla cronologia delle query

La cronologia delle query Athena espone un elenco di query salvate e stringhe di query complete. A meno che non si utilizzino gruppi di lavoro per separare l'accesso alle cronologie delle query, gli utenti di Athena che non sono autorizzati a eseguire query sui dati in Lake Formation possono visualizzare le stringhe di query eseguite su tali dati, inclusi i nomi delle colonne, i criteri di selezione ecc. È consigliabile utilizzare i gruppi di lavoro per separare le cronologie delle query e allineare l'appartenenza al gruppo di lavoro Athena alle autorizzazioni Lake Formation per limitare gli accessi. Per ulteriori informazioni, consulta Uso dei gruppi di lavoro per controllare l'accesso alle query e i costi.

Autorizzazioni Lake Formation ai dati

Oltre all'autorizzazione di base per utilizzare Lake Formation, gli utenti Athena devono disporre delle autorizzazioni Lake Formation per accedere alle risorse su cui eseguono query. Queste autorizzazioni vengono concesse e gestite da un amministratore Lake Formation. Per ulteriori informazioni, consulta Security and Access Control to Metadata and Data (Sicurezza e controllo degli accessi ai metadati e ai dati) nella Guida per gli sviluppatori di AWS Lake Formation .

Autorizzazioni IAM per scrivere nelle posizioni Amazon S3

Le autorizzazioni Lake Formation per Amazon S3 non includono la possibilità di scrivere su Amazon S3. Create Table As Statements (CTAS) richiede l'accesso in scrittura alla posizione Amazon S3 delle tabelle. Per eseguire query CTAS sui dati registrati con Lake Formation, gli utenti di Athena devono disporre delle autorizzazioni IAM per scrivere nelle posizioni Amazon S3 delle tabelle, oltre alle autorizzazioni di Lake Formation appropriate per leggere le posizioni dei dati. Per ulteriori informazioni, consulta Creazione di una tabella dai risultati delle query (CTAS).

Autorizzazioni per dati crittografati, metadati e risultati delle query Athena

I dati di origine sottostanti in Amazon S3 e i metadati nel catalogo dati registrati con Lake Formation possono essere crittografati. Non vi è alcuna modifica al modo in cui Athena gestisce la crittografia

dei risultati delle query quando si utilizza Athena per eseguire query sui dati registrati con Lake Formation. Per ulteriori informazioni, consulta Crittografia dei risultati di query Athena archiviati in Amazon S3.

- Crittografia dei dati di origine — È supportata la crittografia dei dati di origine delle posizioni dei dati di Amazon S3. Gli utenti Athena che eseguono query sulle posizioni Amazon S3 crittografate e registrate con Lake Formation hanno bisogno delle autorizzazioni per crittografare e decrittografare i dati. Per ulteriori informazioni sui requisiti, consultare Opzioni di crittografia supportate da Amazon S3 e Autorizzazioni di accesso a dati crittografati in Amazon S3.
- Crittografia dei metadati — È supportata la crittografia dei metadati nel catalogo dati. Per le entità principali che utilizzano le policy Athena basate su identità, è necessario consentire le operazioni `"kms:GenerateDataKey"`, `"kms:Decrypt"` e `"kms:Encrypt"` per la chiave utilizzata per la crittografia dei metadati. Per ulteriori informazioni, consulta la sezione Crittografia del catalogo dati nella Guida per gli sviluppatori di AWS Glue e Accesso da Athena ai metadati crittografati nel AWS Glue Data Catalog.

Autorizzazioni basate su risorse per i bucket Amazon S3 negli account esterni (facoltativo)

Per eseguire una query su una posizione dei dati di Amazon S3 in un account diverso, una policy IAM basata sulle risorse (policy bucket) deve consentire l'accesso alla posizione. Per ulteriori informazioni, consulta Accesso tra account in Athena a bucket Amazon S3.

Per informazioni sull'accesso a un catalogo dati in un altro account, consulta Accesso al catalogo dati multi-account.

Applicazione di autorizzazioni Lake Formation a database e tabelle esistenti

Se non hai mai utilizzato Athena e utilizzi Lake Formation per configurare l'accesso ai dati delle query, non devi configurare le policy IAM in modo che gli utenti possano leggere i dati Amazon S3 e creare i metadati. Puoi utilizzare Lake Formation per amministrare le autorizzazioni.

La registrazione dei dati con Lake Formation e l'aggiornamento delle policy di autorizzazione IAM non è un requisito. Se i dati non sono registrati con Lake Formation, gli utenti Athena che dispongono delle autorizzazioni appropriate in Amazon S3 e AWS Glue, se applicabile, possono continuare a interrogare i dati non registrati con Lake Formation.

Se hai utenti Athena esistenti che eseguono query su dati non registrati con Lake Formation, puoi aggiornare le autorizzazioni IAM per Amazon S3 e, se applicabile AWS Glue Data Catalog, in modo da poter utilizzare le autorizzazioni di Lake Formation per gestire l'accesso degli utenti in modo

centralizzato. Per l'autorizzazione a leggere le posizioni dei dati Amazon S3, puoi aggiornare le policy basate su risorse e basate su identità per rimuovere le autorizzazioni Amazon S3. Per l'accesso ai metadati, se hai configurato politiche a livello di risorsa per il controllo granulare degli accessi con, AWS Glue puoi invece utilizzare le autorizzazioni di Lake Formation per gestire l'accesso.

Per ulteriori informazioni, consulta Aggiornamento AWS Glue delle autorizzazioni relative ai dati al Accesso granulare a database e tabelle in AWS Glue Data Catalog modello nella Guida per gli sviluppatori. AWS Lake Formation AWS Lake Formation

Utilizzo di Lake Formation e dei driver Athena JDBC e ODBC per l'accesso federato ad Athena

I driver Athena JDBC e ODBC supportano la federazione basata su SAML 2.0 con Athena usando i provider di identità Okta e Microsoft Active Directory Federation Services (AD FS). Integrando Amazon Athena AWS Lake Formation con, abiliti l'autenticazione basata su SAML su Athena con credenziali aziendali. Con Lake Formation e AWS Identity and Access Management (IAM), puoi mantenere un controllo granulare degli accessi a livello di colonna sui dati disponibili per l'utente SAML. Con i driver Athena JDBC e ODBC, l'accesso federato è disponibile per l'accesso programmatico o al tool.

Per utilizzare Athena per accedere a un'origine dati controllata da Lake Formation, devi abilitare la federazione basata su SAML 2.0 configurando i ruoli di provider di identità (IdP) e (IAM). AWS Identity and Access Management Per informazioni dettagliate sulle fasi, consulta Tutorial: Configurazione dell'accesso federato per gli utenti Okta in Athena utilizzando Lake Formation e JDBC.

Prerequisiti

Per utilizzare Amazon Athena e Lake Formation per l'accesso federato, devi soddisfare i seguenti requisiti:

- Gestire le identità aziendali utilizzando un provider di identità esistente basato su SAML, ad esempio Okta o Microsoft Active Directory Federation Services (AD FS).
- Lo usi come archivio di metadati. AWS Glue Data Catalog
- Definisci e gestisci le autorizzazioni in Lake Formation per accedere a database, tabelle e colonne in AWS Glue Data Catalog. Per ulteriori informazioni, consulta la Guida per gli sviluppatori di AWS Lake Formation.
- Utilizzare la versione 2.0.14 o versioni successive del driver Athena JDBC o la versione 1.1.3 o versioni successive del driver Athena ODBC.

Considerazioni e limitazioni

Quando utilizzi il driver Athena JDBC o ODBC e Lake Formation per configurare l'accesso federato ad Athena, tieni presenti i seguenti punti:

- Attualmente, il driver Athena JDBC e i driver ODBC supportano i provider di identità Okta, Microsoft Active Directory Federation Services (AD FS) e Azure AD. Sebbene il driver JDBC Athena disponga di una classe SAML generica che può essere estesa per utilizzare altri provider di identità, il supporto per le estensioni personalizzate che consentono l'uso di altri provider di identità () IdPs con Athena potrebbe essere limitato.
- L'accesso federato che utilizza i driver JDBC e ODBC non è compatibile con la funzionalità di propagazione delle identità attendibili di Centro identità IAM.
- Attualmente, non è possibile utilizzare la console Athena per configurare il supporto per l'utilizzo di IdP e SAML con Athena. Per configurare questo supporto, utilizzare il provider di identità di terzi, le console di gestione Lake Formation e IAM e il client driver JDBC o ODBC.
- Comprendere la specificazione SAML 2.0 e come funziona con il tuo provider di identità prima di configurare il tuo provider di identità e SAML per l'utilizzo con Lake Formation e Athena.
- I provider SAML e i driver Athena JDBC e ODBC sono forniti da terze parti, pertanto l'assistenza AWS per problemi relativi al loro utilizzo potrebbe essere limitata.

Argomenti

- Tutorial: Configurazione dell'accesso federato per gli utenti Okta in Athena utilizzando Lake Formation e JDBC

Tutorial: Configurazione dell'accesso federato per gli utenti Okta in Athena utilizzando Lake Formation e JDBC

Questo tutorial mostra come configurare Okta AWS Lake Formation, le AWS Identity and Access Management autorizzazioni e il driver JDBC Athena per abilitare l'uso federato di Athena basato su SAML. Lake Formation fornisce un controllo di accesso dettagliato sui dati disponibili in Athena per l'utente basato su SAML. Per configurare questa configurazione, il tutorial utilizza la console per sviluppatori Okta, le console AWS IAM e Lake Formation e lo strumento SQL Workbench/J.

Prerequisiti

Questo tutorial presuppone che tu abbia eseguito quanto segue:

- Creazione di un account Amazon Web Services. Per creare un account, visita la homepage di Amazon Web Services.
- Impostare una posizione dei risultati delle query per Athena in Amazon S3.
- Registrazione di una posizione del bucket dati Amazon S3 con Lake Formation
- Definizione di un database e di tabelle nel Catalogo dati AWS Glue che puntano ai tuoi dati in Amazon S3.
  - Se non hai ancora definito una tabella, esegui un AWS Glue crawler o usa Athena per definire un database e una o più tabelle per i dati a cui desideri accedere.
  - Questo tutorial utilizza una tabella basata sul set di dati sui viaggi di NYC Taxi disponibile nel Registro di Open Data in AWS. Il tutorial utilizza il nome del database `tripdb` e il nome della tabella `nyctaxi`.

Fasi del tutorial

- Fase 1: creare un account Okta
- Fase 2: aggiungere gli utenti e i gruppi a Okta
- Fase 3: configurazione di un'applicazione Okta per l'autenticazione SAML
- Fase 4: Creare un AWS SAML Identity Provider e un ruolo IAM di accesso a Lake Formation
- Passaggio 5: aggiungere il ruolo IAM e il provider di identità SAML all'applicazione Okta
- Passaggio 6: concedere le autorizzazioni a utenti e gruppi tramite AWS Lake Formation
- Passaggio 7: Verificare l'accesso tramite il client Athena JDBC
- Conclusioni
- Risorse correlate

Fase 1: creare un account Okta

Questo tutorial utilizza Okta come provider di identità basato su SAML. Se non hai già un account Okta, puoi crearne uno gratuito. È necessario un account Okta per poter creare un'applicazione Okta per l'autenticazione SAML.

Per creare un account Okta

1. Per utilizzare Okta, andare alla pagina di iscrizione per sviluppatori Oktae creare un account di prova Okta gratuito. Il servizio Developer Edition è gratuito fino ai limiti specificati da Okta all'indirizzo developer.okta.com/pricing.

2. Quando ricevi l'e-mail di attivazione, attiva il tuo account.

   Ti verrà assegnato un nome di dominio Okta. Salva il nome di dominio come riferimento. Successivamente, si utilizza il nome di dominio (*< okta-idp-domain >*) nella stringa JDBC che si connette ad Athena.

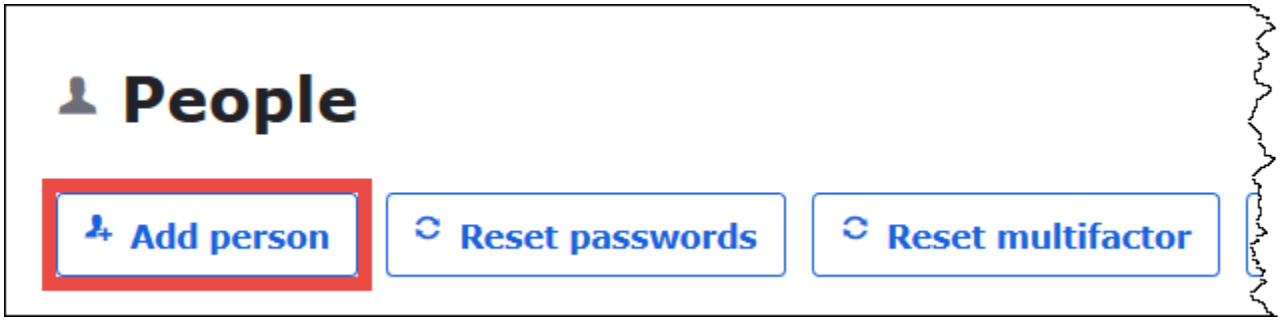
Fase 2: aggiungere gli utenti e i gruppi a Okta

In questo passaggio utilizzi la console Okta per i seguenti processi:

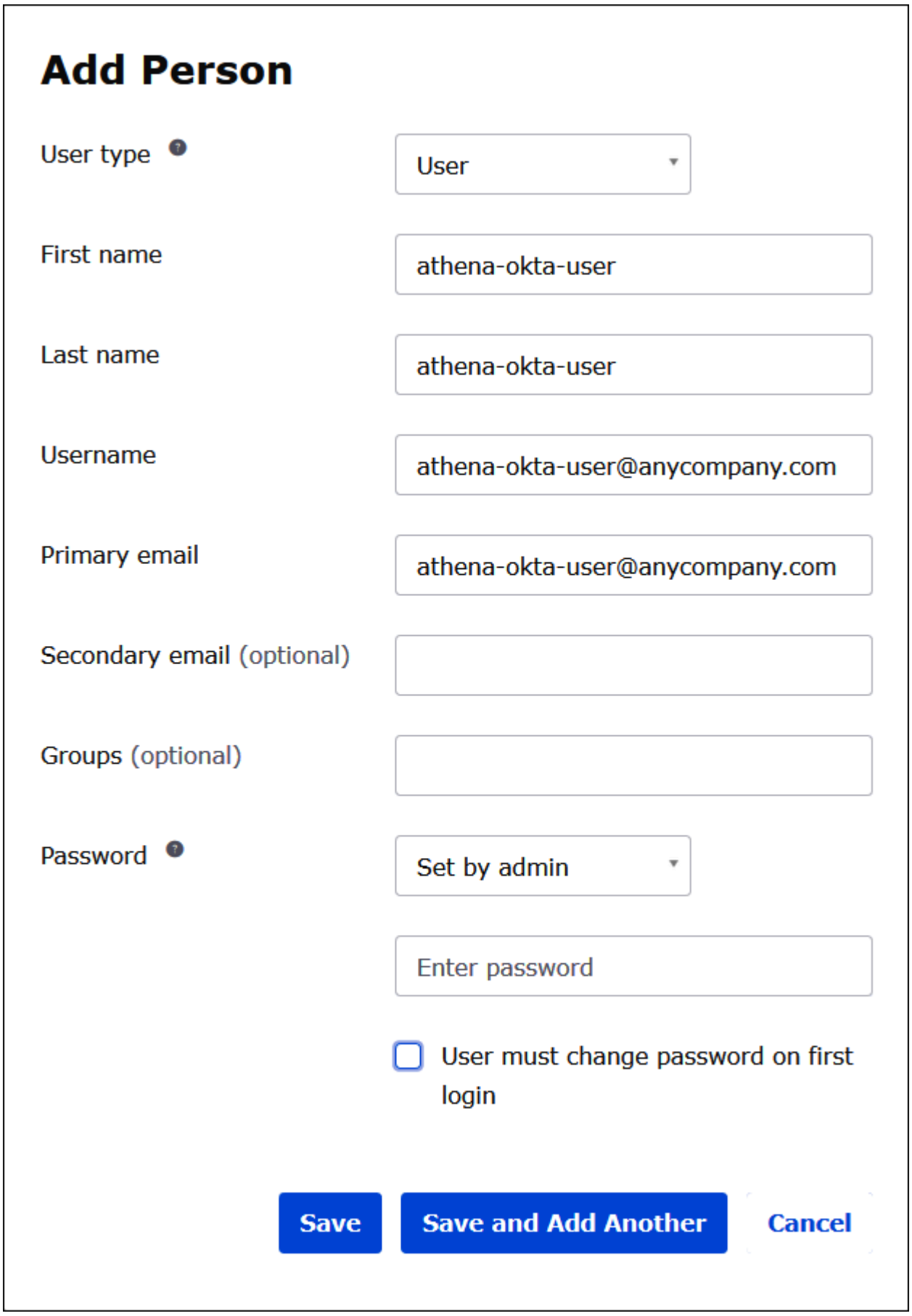
- Creare due utenti Okta.
- Creare due gruppi Okta.
- Aggiungere un utente Okta a ciascun gruppo Okta.

Per aggiungere utenti a Okta

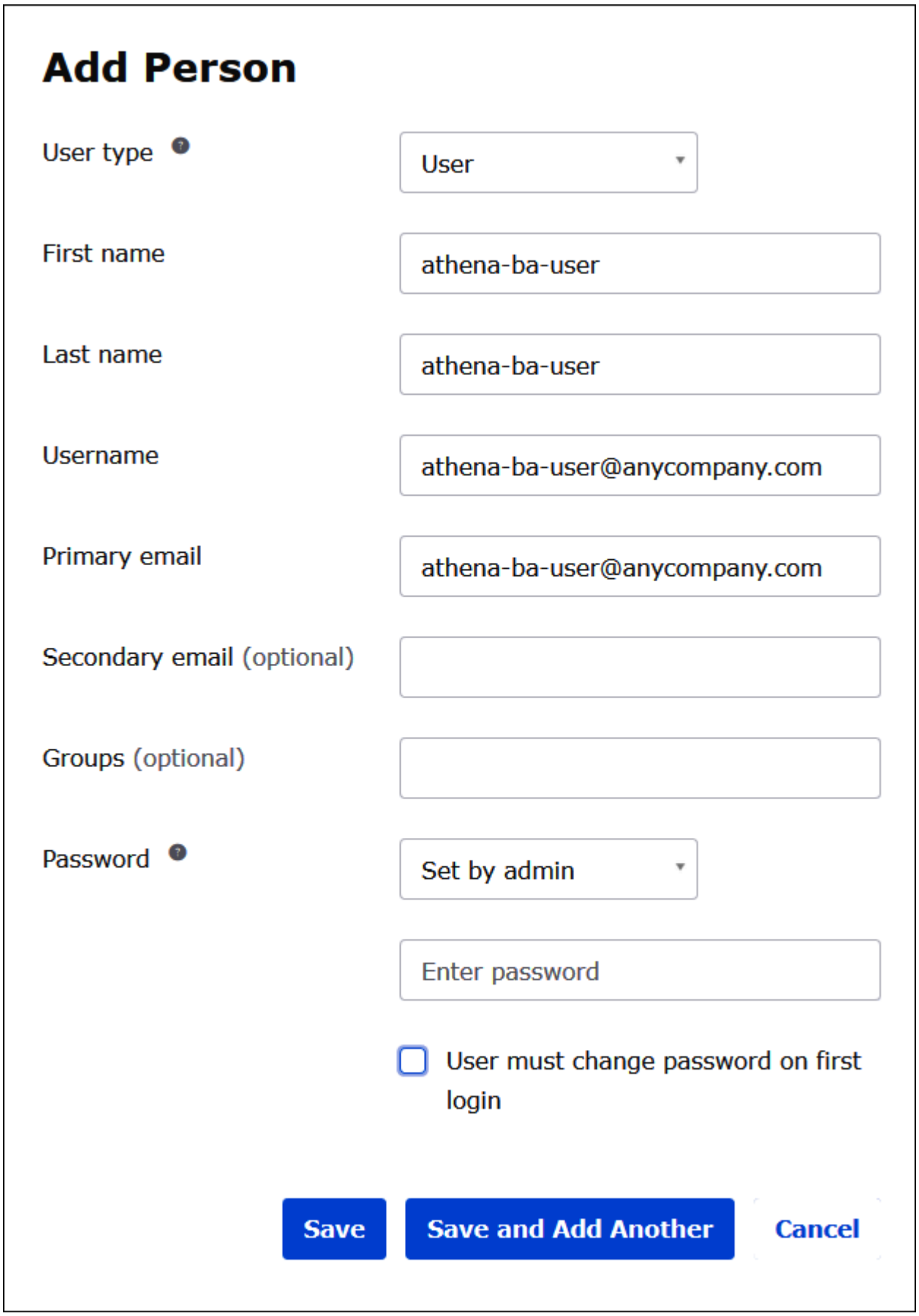
1. Dopo aver attivato l'account Okta, accedere come utente amministrativo al dominio Okta assegnato.
2. Nel pannello di navigazione a sinistra selezionare Directory e poi Persone.
3. Scegliere Aggiungi persona per aggiungere un nuovo utente che accederà ad Athena tramite il driver JDBC.



4. Nella finestra di dialogo Aggiungi persona, inserire le informazioni richieste.
   - Inserisci i valori per Nome e Cognome. In questo tutorial si utilizza *athena-okta-user*.
   - Specificare un Nome utente e un'indirizzo email principale. *Questo tutorial utilizza @anycompany .comathena-okta-user.*
   - Per Password, scegli Impostato dall'amministratore e quindi fornisci una password. Questo tutorial cancella l'opzione per L'utente deve cambiare la password al primo accesso; i requisiti di sicurezza possono variare.


Add Person
User type
User
First name
athena-okta-user
Last name
athena-okta-user
Username
athena-okta-user@anycompany.com
Primary email
athena-okta-user@anycompany.com
Secondary email (optional)
Groups (optional)
Password
Set by admin
Enter password
User must change password on first login
Save
Save and Add Another
Cancel

5. Scegliere Salva e aggiungi un altro.
6. Inserisci le informazioni per un altro utente. Questo esempio aggiunge l'utente business analyst *athena-ba-user@anycompany .com*.

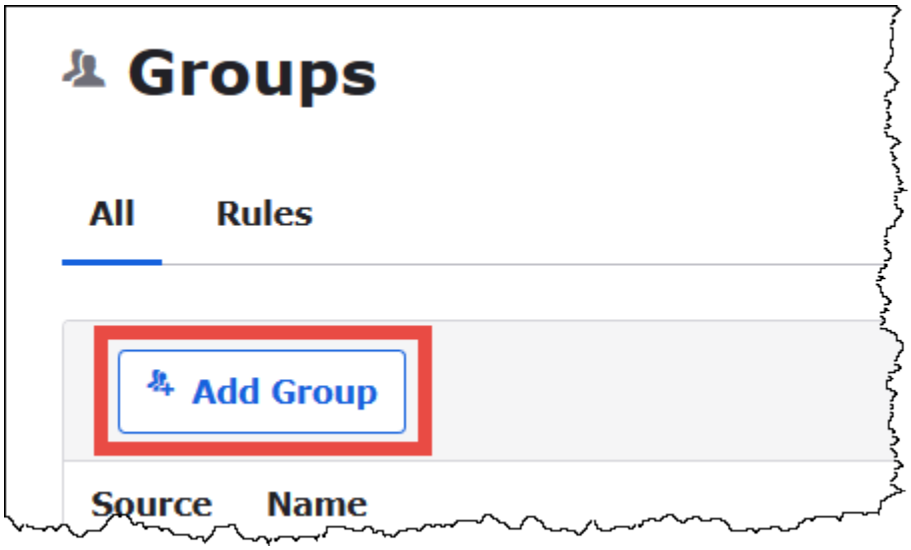

Add Person
User type
User
First name
athena-ba-user
Last name
athena-ba-user
Username
athena-ba-user@anycompany.com
Primary email
athena-ba-user@anycompany.com
Secondary email (optional)
Groups (optional)
Password
Set by admin
Enter password
User must change password on first login
Save
Save and Add Another
Cancel

7. Selezionare Salva.

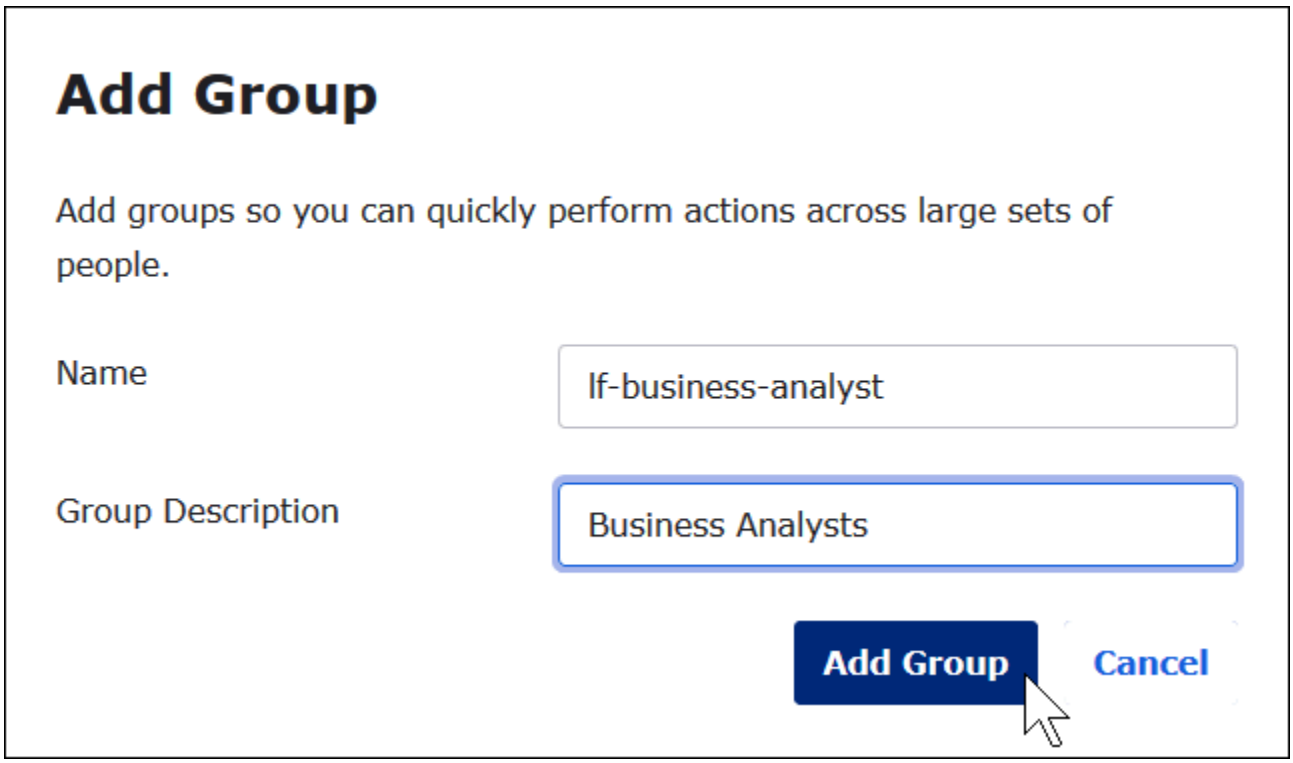
Nella procedura seguente, fornisci l'accesso a due gruppi Okta tramite il driver Athena JDBC aggiungendo un gruppo "Business Analysts" e un gruppo "Developer".

Per aggiungere gruppi Okta

1. Nel pannello di navigazione a sinistra scegli Directory e poi Persone.
2. Nella pagina Gruppi, scegli Aggiungi gruppo.



3. Nella finestra di dialogo Aggiungi gruppo, inserisci le informazioni richieste.
   - Per Nome, immetti *lf-business-analyst*.
   - Per Descrizione del gruppo, inserisci *Business Analyst*.

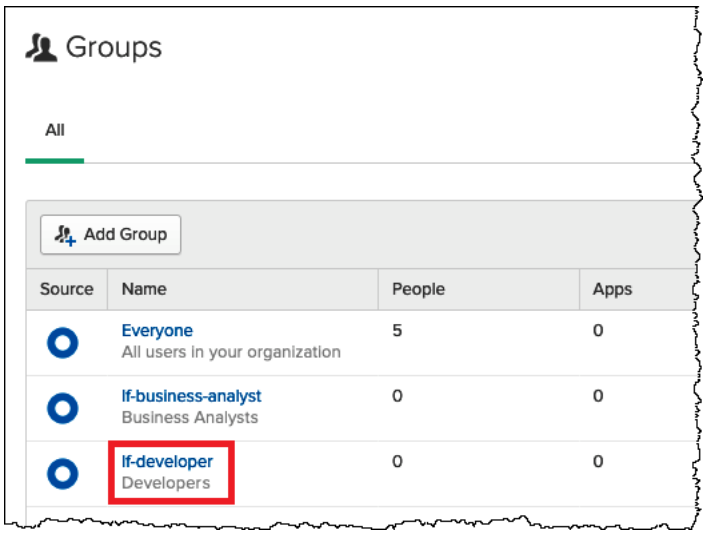


4. Scegli Add Group (Aggiungi gruppo).
5. Nella pagina Gruppi, scegliere di nuovo Aggiungi gruppo. Questa volta inserirai le informazioni per il gruppo Sviluppatori.
6. Inserisci le informazioni richieste.
   - Per Nome, inserire *lf-developer*.
   - Per Descrizione del gruppo, inserire *Sviluppatori*.
7. Scegli Add Group (Aggiungi gruppo).

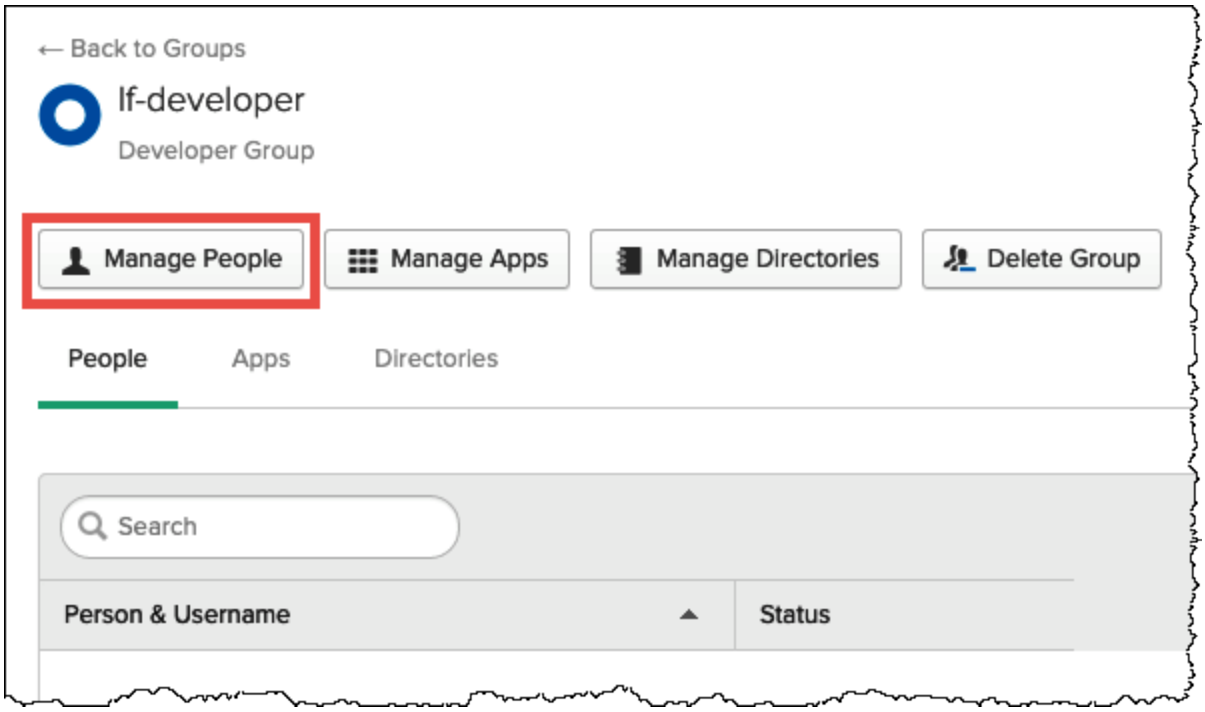
Ora che sono disponibili due utenti e due gruppi, è possibile aggiungere un utente a ciascun gruppo.

Per aggiungere utenti ai gruppi

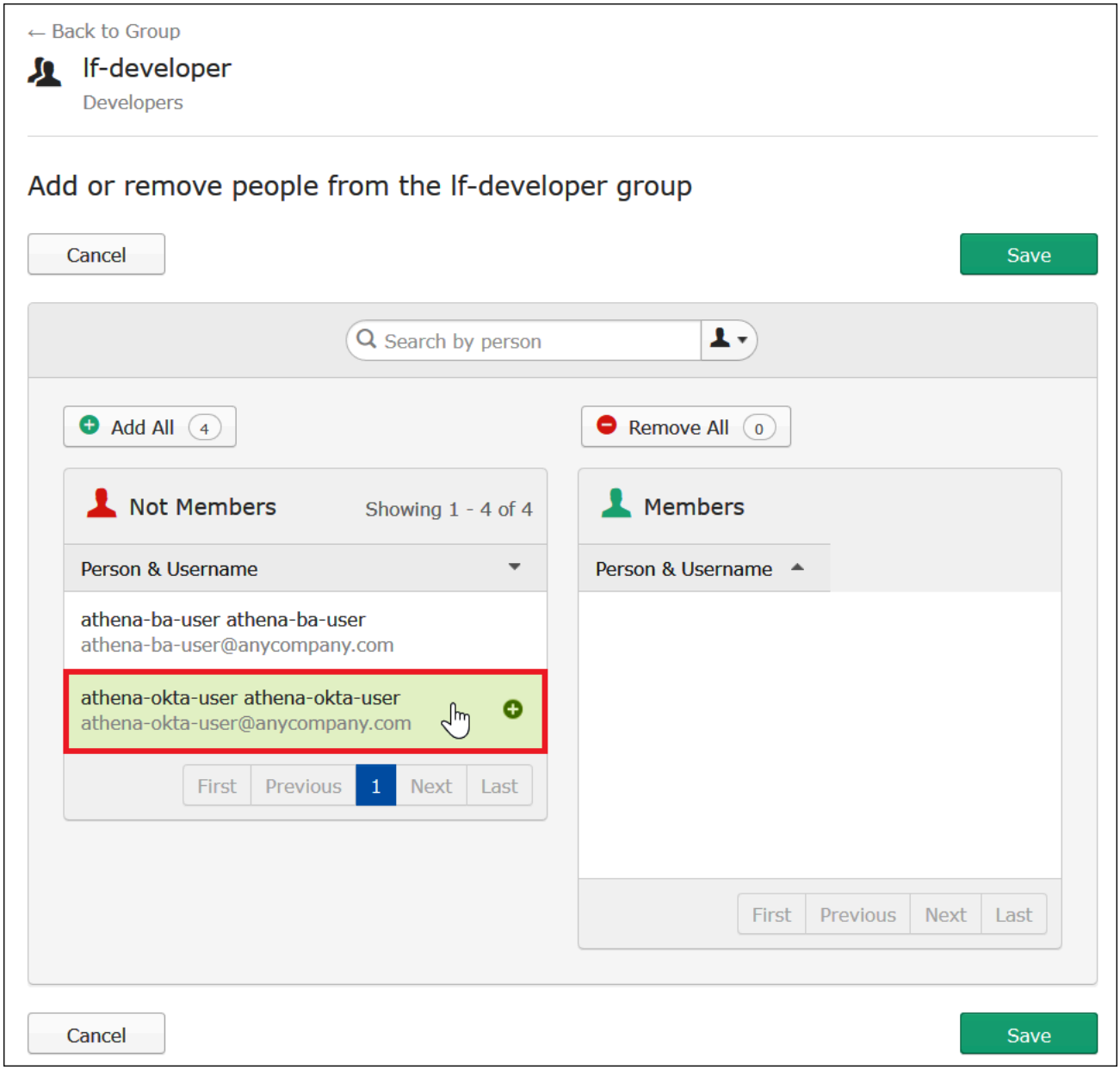
1. Nella pagina Gruppi scegli il gruppo lf-developer appena creato. Aggiungerai a questo gruppo uno degli utenti Okta che hai creato come sviluppatore.



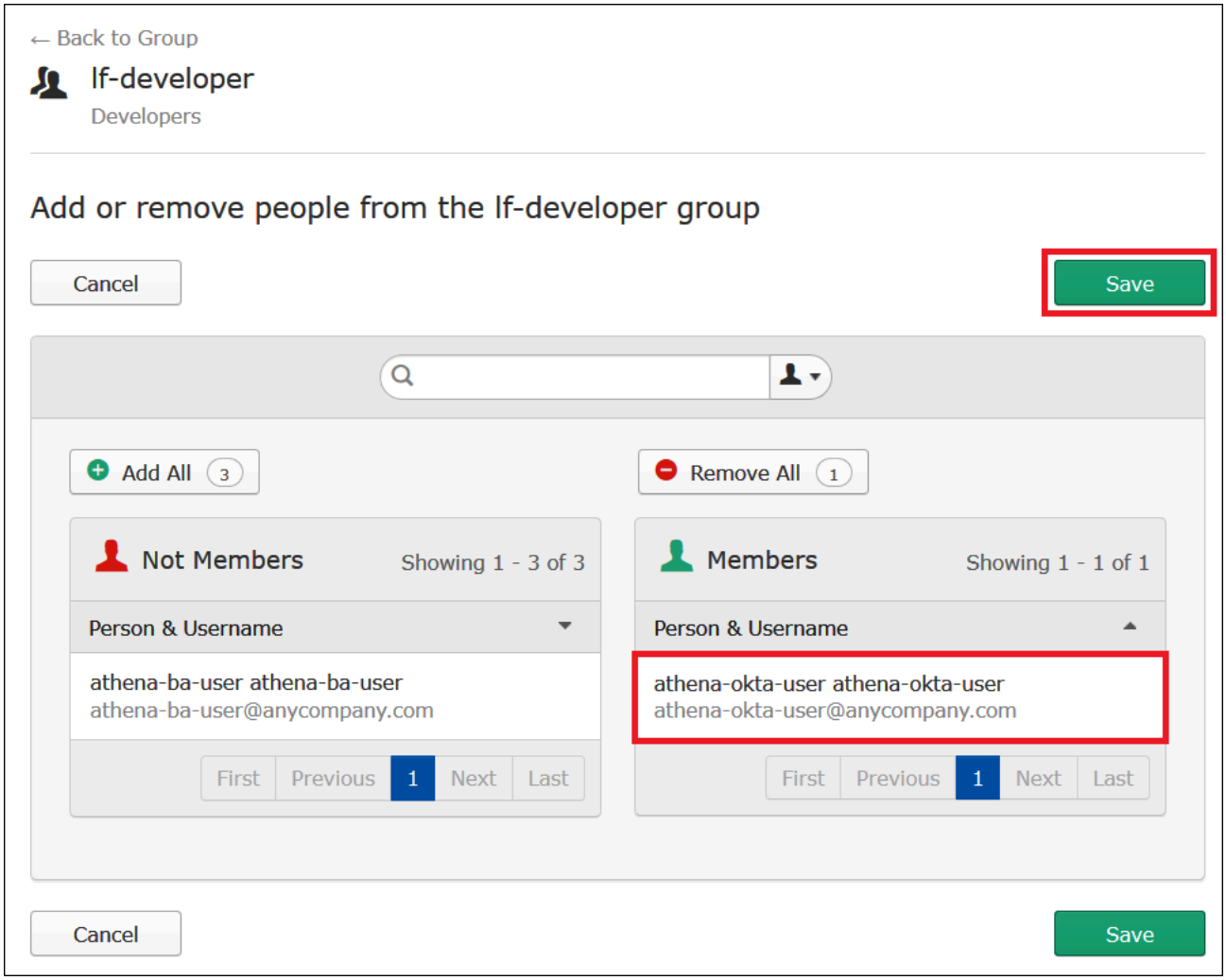
2. Scegli Gestisci persone.



3. Dall'elenco Non membri, scegli athena-okta-user.

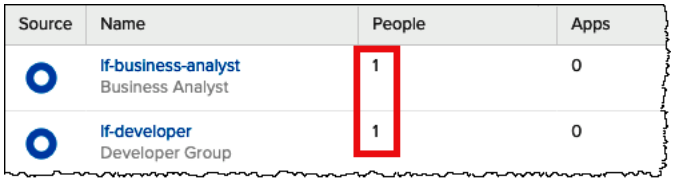


La voce per l'utente si sposta dall'elenco di sinistra Non membri all'elenco di destra Membri.



4. Selezionare Salva.
5. Scegli Torna al gruppo, oppure scegli Directory, quindi scegli Gruppi.
6. Scegli il lf-business-analystgruppo.
7. Scegli Gestisci persone.
8. Aggiungilo athena-ba-userall'elenco dei membri del lf-business-analystgruppo, quindi scegli Salva.
9. Scegli Torna al gruppo, oppure scegli Directory, Gruppi.

   La pagina Gruppi mostra ora che ogni gruppo ha un utente Okta.



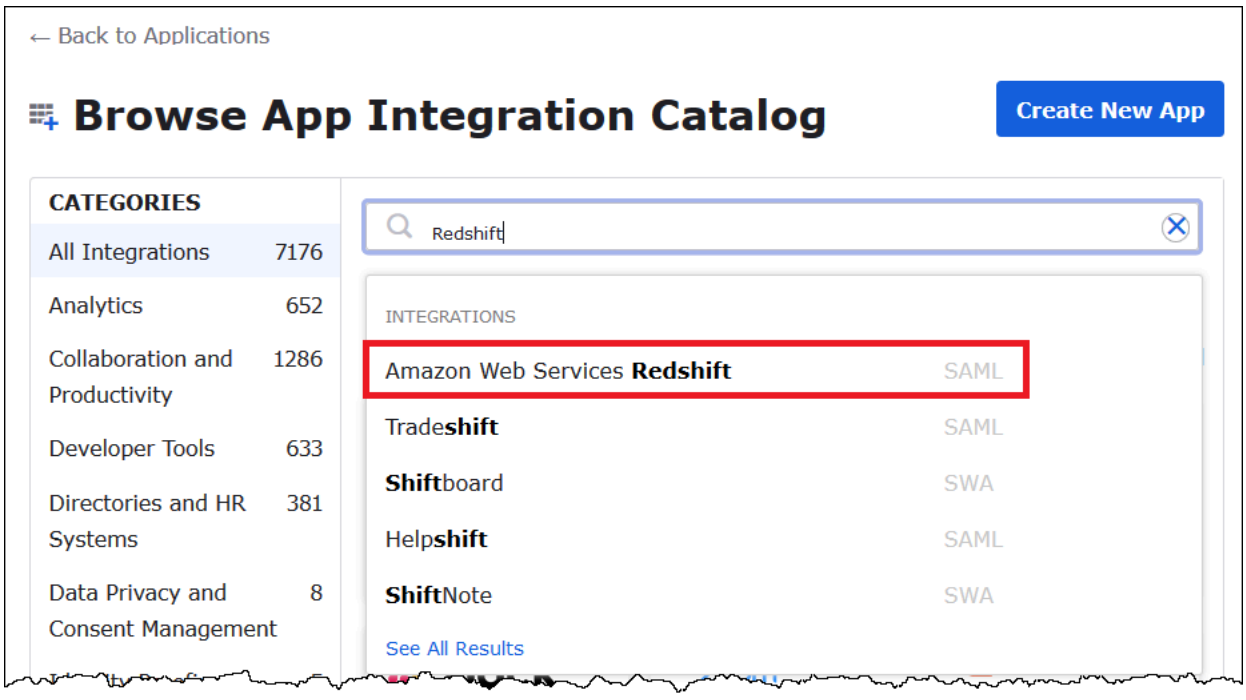
Fase 3: configurazione di un'applicazione Okta per l'autenticazione SAML

In questo passaggio utilizzi la console degli sviluppatori Okta per i seguenti processi:

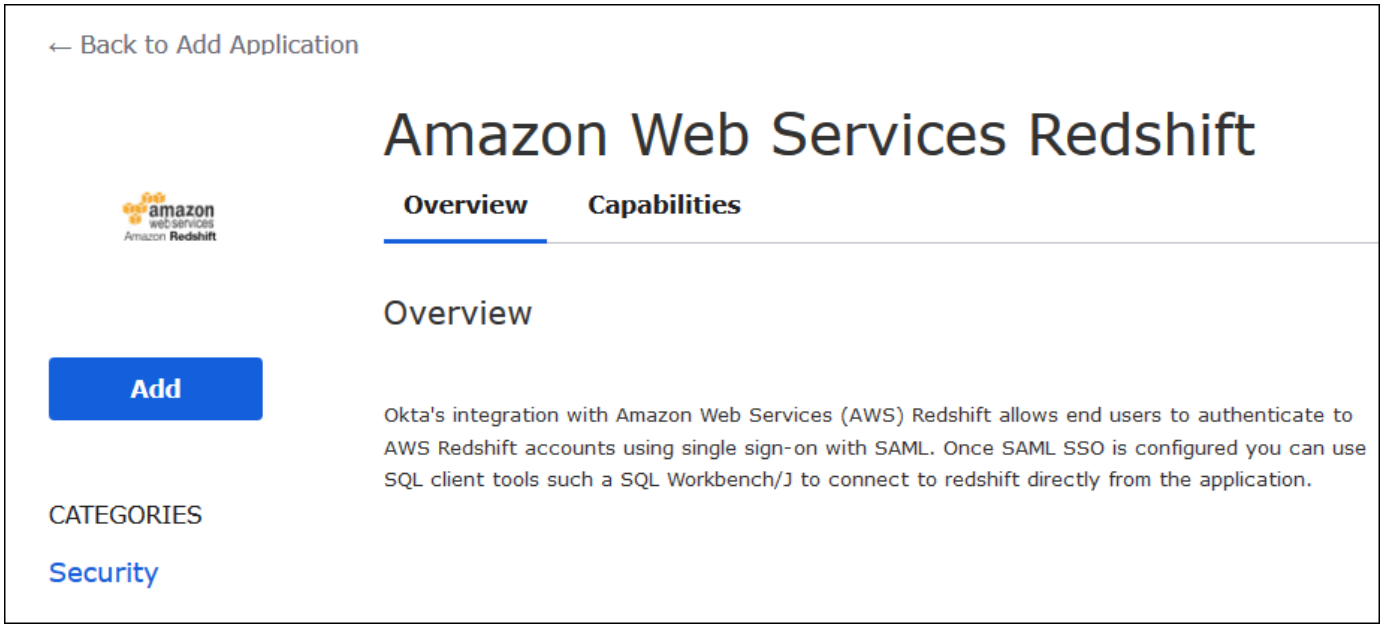
- Aggiungi un'applicazione SAML da utilizzare con AWS.
- Assegnare l'applicazione all'utente Okta.
- Assegnare l'applicazione a un gruppo Okta.
- Scaricare i metadati del provider di identità risultanti per un utilizzo successivo con AWS.

Per aggiungere un'applicazione per l'autenticazione SAML

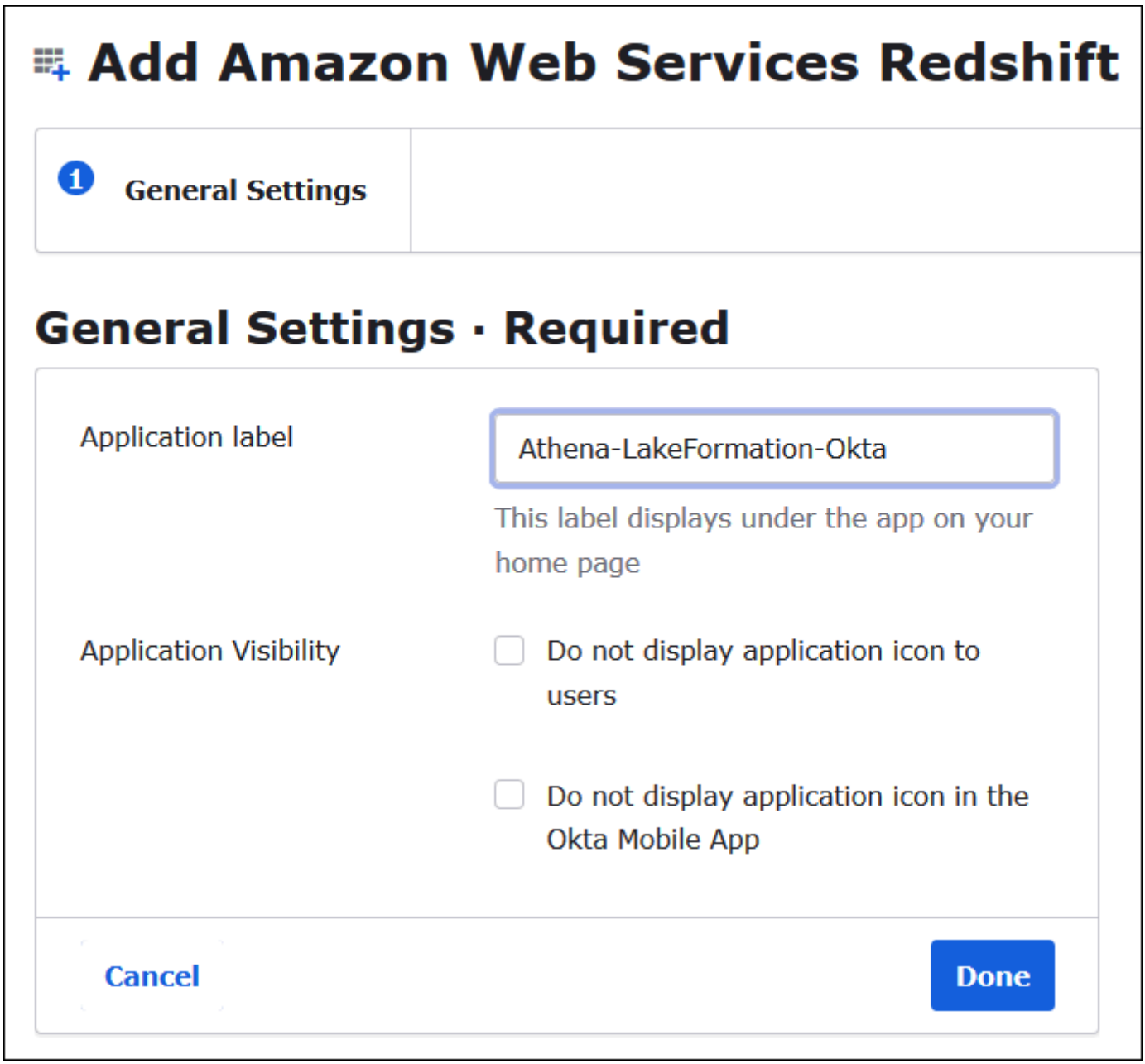
1. Nel pannello di navigazione Okta, selezionare Applicazioni, Applications in modo da poter configurare un'applicazione Okta per l'autenticazione SAML su Athena.
2. Fare clic su Sfoglia catalogo delle app.
3. Nella casella di ricerca immetti **`Redshift`**.
4. Scegliere Amazon Web Services Redshift. L'applicazione Okta in questo tutorial utilizza l'integrazione SAML esistente per Amazon Redshift.



5. Sulla pagina Amazon Web Services Redshift, scegliere Aggiungi per creare un'applicazione basata su SAML per Amazon Redshift.



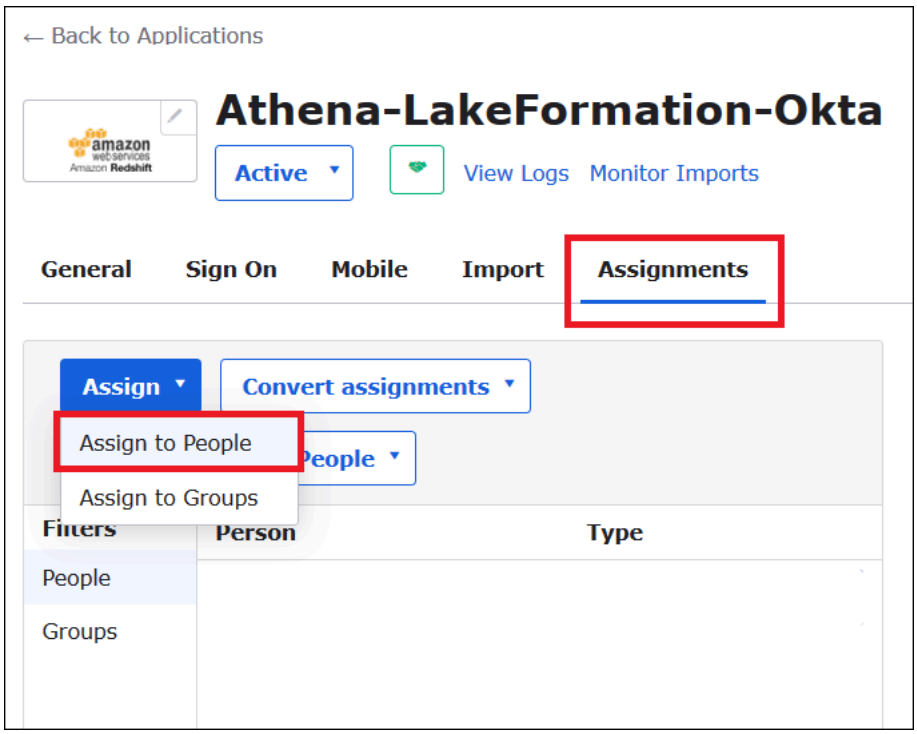
6. Per Etichetta dell'applicazione, inserire `Athena-LakeFormation-Okta`, quindi scegliere Fine.



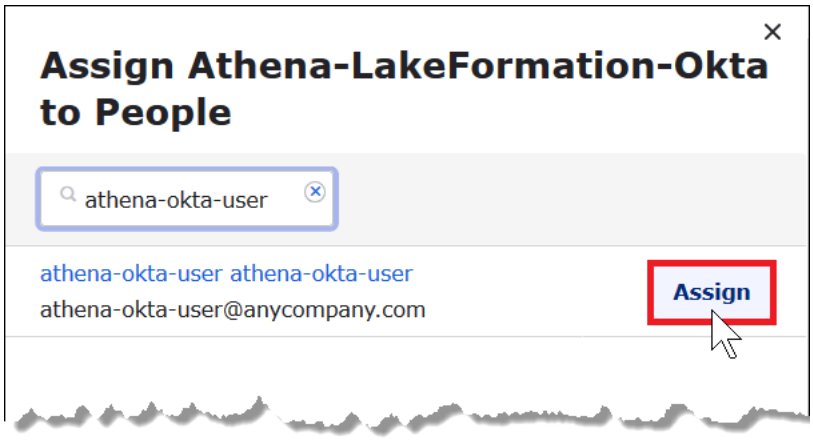
Dopo aver creato un'applicazione Okta, è possibile assegnarla agli utenti e ai gruppi creati.

Per assegnare l'applicazione a utenti e gruppi

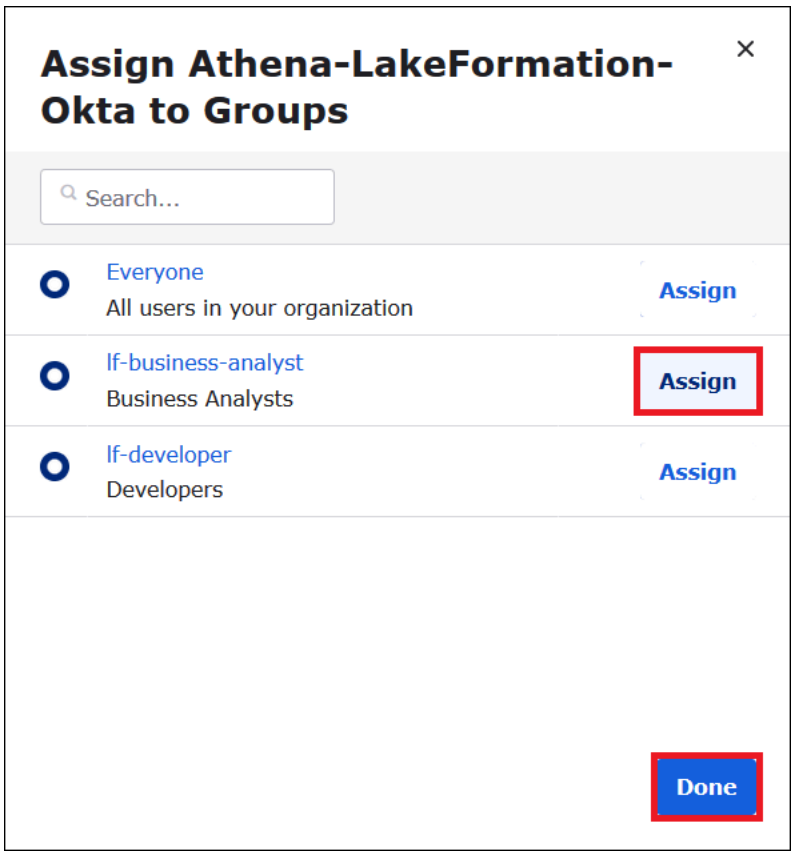
1. Nella pagina Applicazioni, scegli l'applicazione Athena- LakeFormation -Okta.
2. Nella tab Assegnazioni, scegliere Assegna,Assegna alle persone.



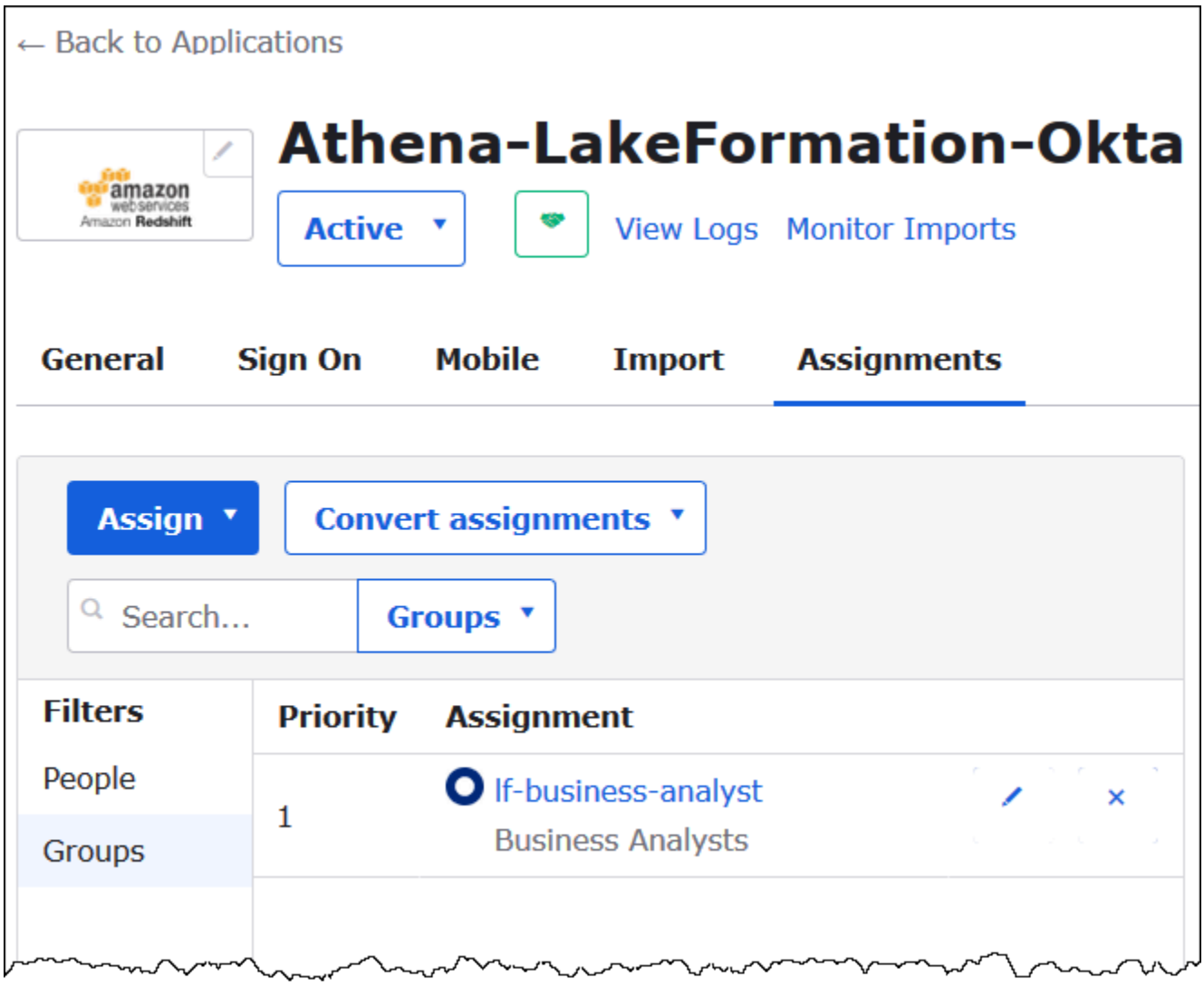
3. Nella finestra di dialogo Assegna Athena LakeFormation - -Okta a persone, trovate athena-okta-userl'utente che avete creato in precedenza.
4. Scegliere Assegna per assegnare l'utente all'applicazione.



5. Scegliere Salva e torna indietro.
6. Seleziona Fatto.
7. Nella scheda Assegnazioni dell'applicazione Athena LakeFormation - -Okta, scegliete Assegna, Assegna a gruppi.
8. Per lf-business-analyst, scegli Assegna per assegnare l'applicazione Athena- LakeFormation - Okta al lf-business-analystgruppo, quindi scegli Fine.



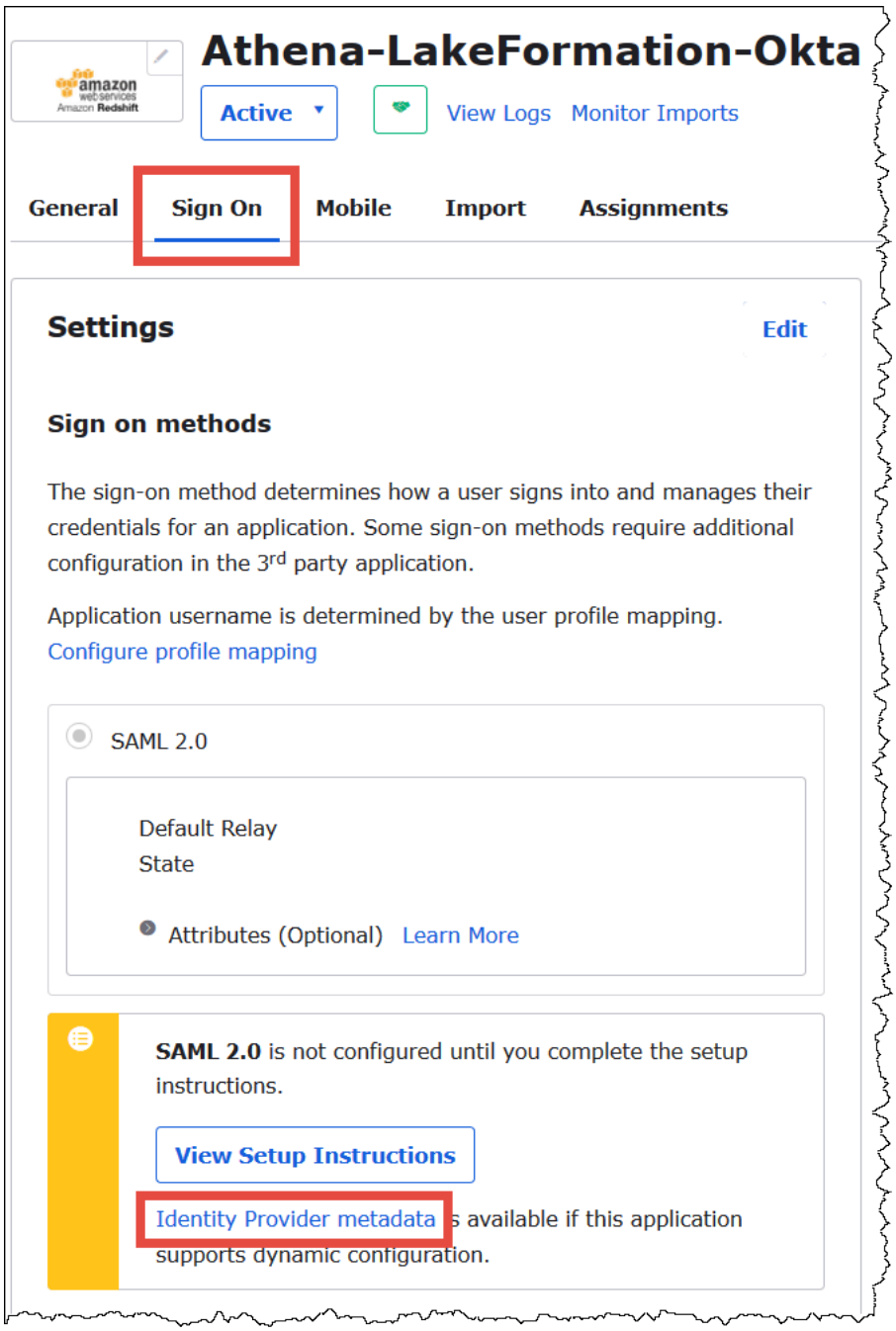
Il gruppo viene visualizzato nell'elenco dei gruppi per l'applicazione.



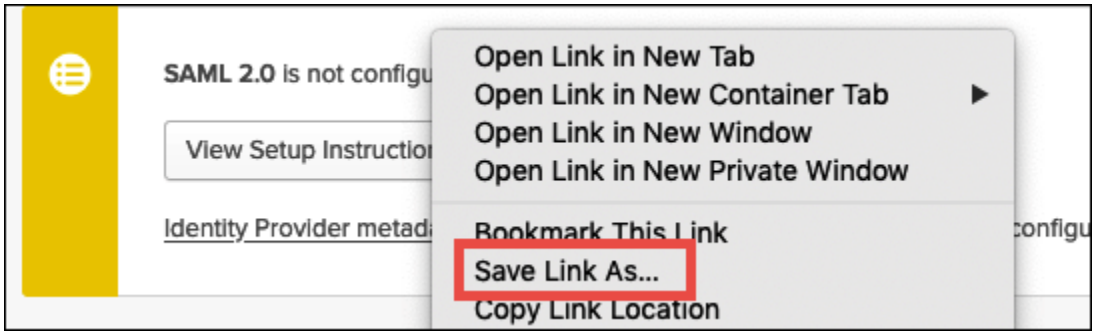
Ora puoi scaricare i metadati dell'applicazione del provider di identità per l'utilizzo con AWS.

Per scaricare i metadati dell'applicazione

1. Scegliere la tab dell'applicazione Okta Accesso e quindi fare clic con il pulsante destro su Metadati del provider di identità.



2. Scegliere Salva link con nome per salvare i metadati del provider di identità, che sono in formato XML, in un file. Assegnare un nome che si riconosce (ad esempio, `Athena-LakeFormation-idp-metadata.xml`).



Fase 4: Creare un AWS SAML Identity Provider e un ruolo IAM di accesso a Lake Formation

In questo passaggio, si utilizza la console AWS Identity and Access Management (IAM) per eseguire le seguenti attività:

- Creare un provider di identità per AWS.
- Creare un ruolo IAM per l'accesso a Lake Formation.
- Aggiungi la policy AmazonAthenaFullAccess gestita al ruolo.
- Aggiungi una politica per Lake Formation e AWS Glue al ruolo.
- Aggiungere una policy per i risultati delle query Athena al ruolo.

Per creare un provider di identità AWS SAML

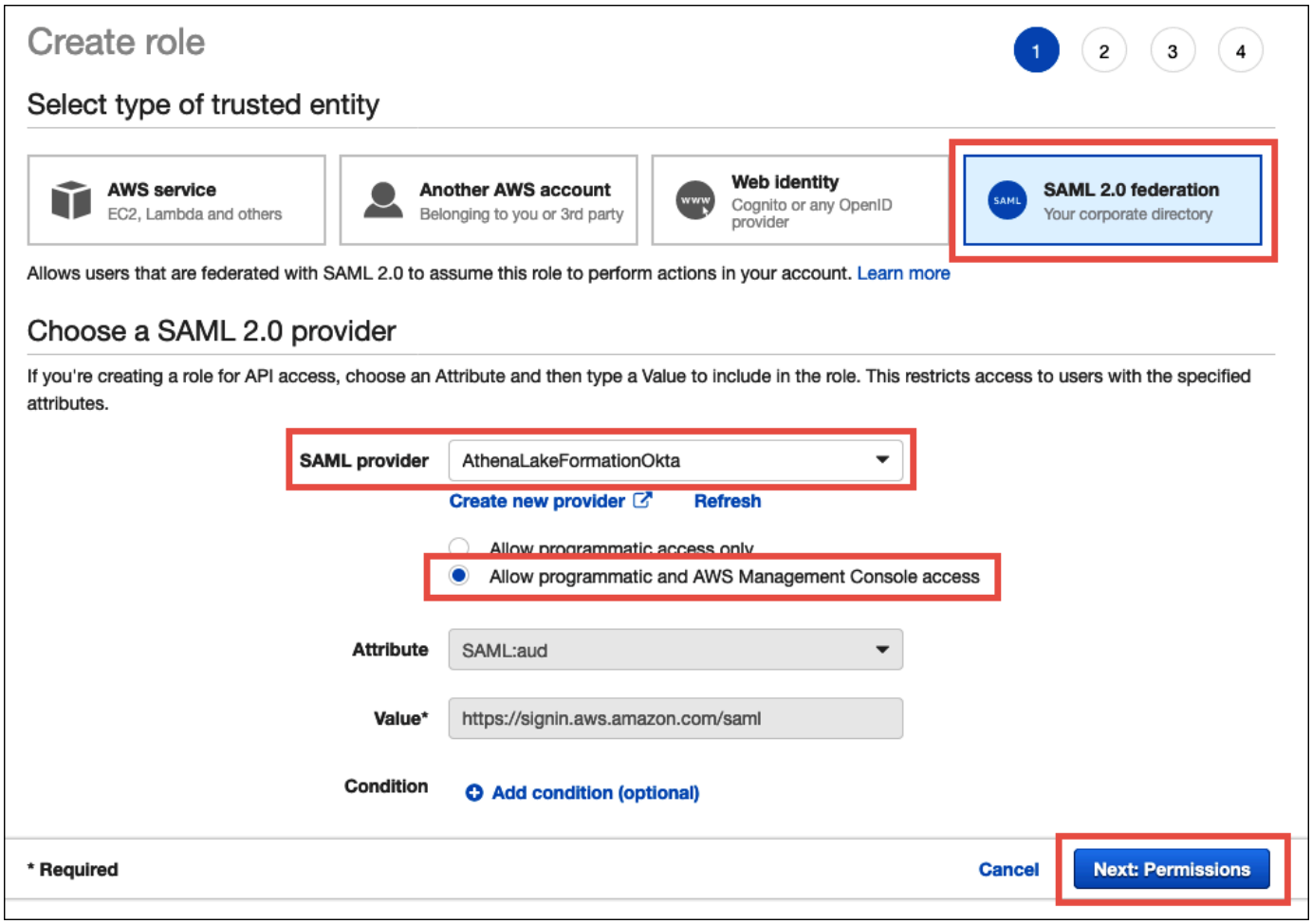
1. Accedi alla console Account Amazon Web Services comeamministratore dell'account Amazon Web Services e accedi alla console IAM (https://console.aws.amazon.com/iam/).
2. Nel pannello di navigazione, scegli Provider di identità, quindi clicca su Aggiungi provider.
3. Nella schermata Configura provider, inserisci le seguenti informazioni:
   - Per Tipo di provider, scegliere SAML.
   - Per Nome provider, inserire `AthenaLakeFormationOkta`.
   - Per Documento dei metadati, utilizzare l'opzione Scegli file per caricare il file XML dei metadati del provider di identità (IdP) scaricato.
4. Scegli Aggiungi provider.

Successivamente, crei un ruolo IAM per AWS Lake Formation l'accesso. Aggiungere due policy in linea al ruolo. Una policy fornisce le autorizzazioni per accedere a Lake Formation e alle AWS Glue

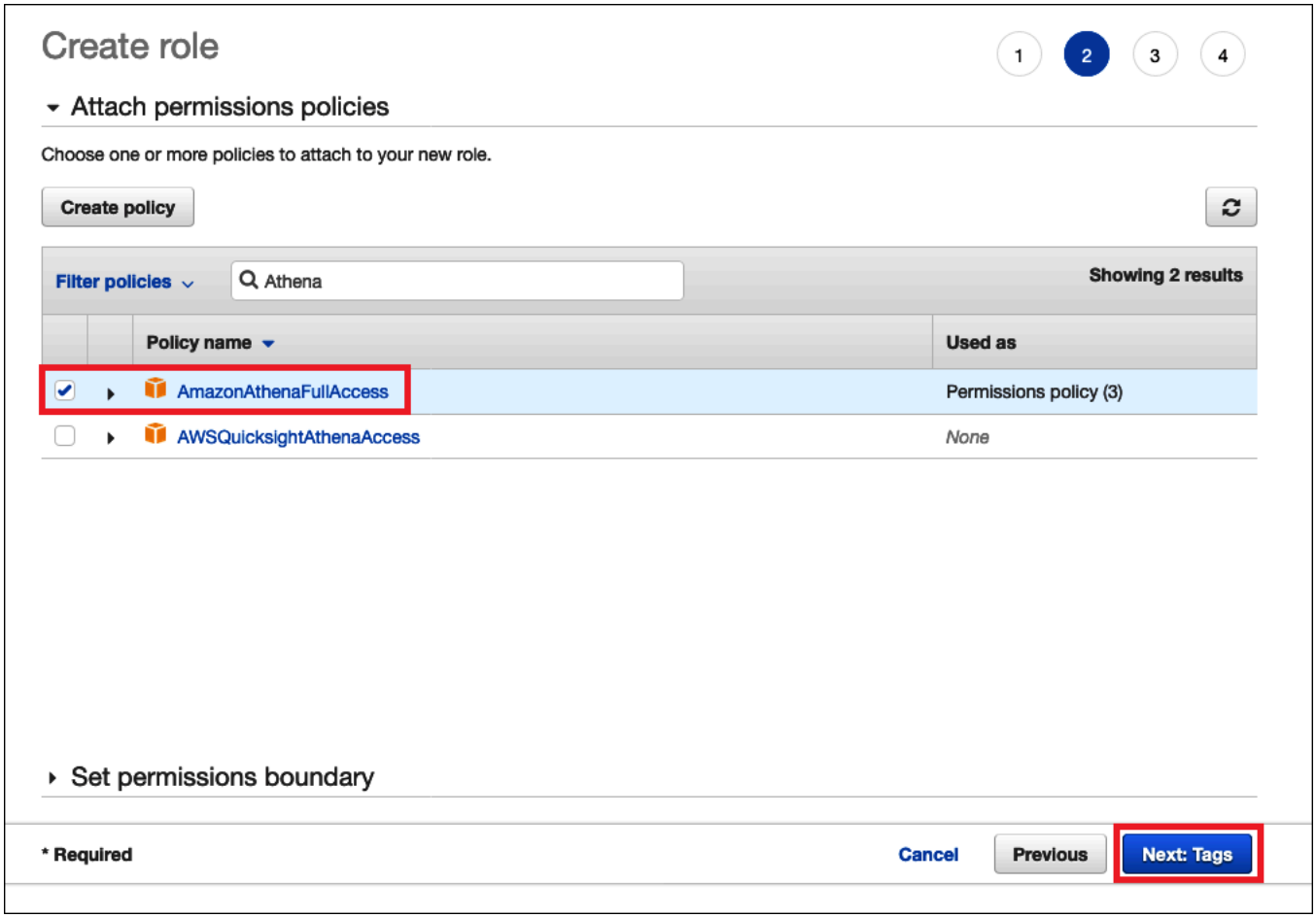
API. L'altra policy fornisce l'accesso ad Athena e alla posizione dei risultati della query Athena in Amazon S3.

Per creare un ruolo IAM per l'accesso AWS Lake Formation

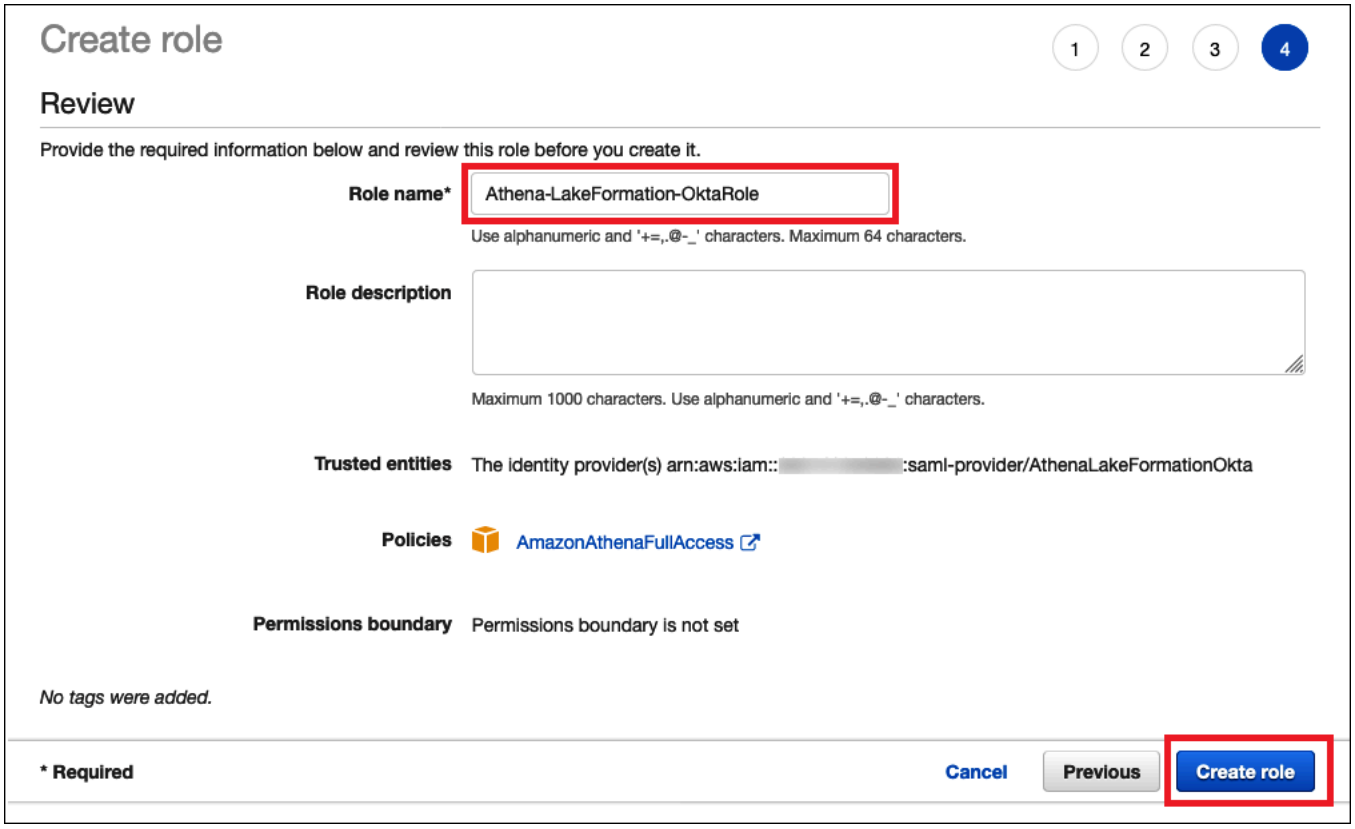
1. Nel pannello di navigazione della console IAM, scegliere Ruoli e quindi Crea ruolo.
2. Nella pagina Crea ruolo, procedere nel seguente modo:



   a. Per Seleziona tipo di entità attendibile, scegliere SAML 2.0 Federation.
   b. Per il provider SAML, seleziona AthenaLakeFormationOkta.
   c. Per il provider SAML, seleziona l'opzione Consenti accesso e programmazione. AWS Management Console
   d. Scegli Successivo: autorizzazioni.
3. Sulla pagina Allegare policy autorizzazioni, per Filtra policy inserire **`Athena`**.
4. Seleziona la politica AmazonAthenaFullAccessgestita, quindi scegli Avanti: tag.



5. Nella pagina Add tags (Aggiungi tag), scegliere Next: Review (Successivo: rivedi).
6. Nella pagina Revisione, in Nome ruolo, inserisci un nome per il ruolo (ad esempio, *Atena-LakeFormation - OktaRole*), quindi scegli Crea ruolo.



Successivamente, aggiungi policy in linea che consentono l'accesso ai risultati delle query di Lake Formation, AWS Glue API e Athena in Amazon S3.

Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Per aggiungere una politica in linea al ruolo di Lake Formation e AWS Glue

1. Nell'elenco dei ruoli nella console IAM, scegliere il `Athena-LakeFormation-OktaRole` appena creato.
2. Nella pagina Riepilogo per il ruolo, nella tab Autorizzazioni scegli Aggiungi policy in linea.
3. Nella pagina Create policy (Crea policy), scegli JSON.
4. Aggiungi una policy in linea come la seguente, che fornisce l'accesso a Lake Formation e all'API AWS Glue .

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
```

```
        "Action": [
            "lakeformation:GetDataAccess",
            "glue:GetTable",
            "glue:GetTables",
            "glue:GetDatabase",
            "glue:GetDatabases",
            "glue:CreateDatabase",
            "glue:GetUserDefinedFunction",
            "glue:GetUserDefinedFunctions"
        ],
        "Resource": "*"
    }
}
```

5. Scegli Verifica policy.
6. Per Nome, inserire un nome per la policy, ad esempio **`LakeFormationGlueInlinePolicy`**.
7. Scegli Crea policy.

Per aggiungere una policy in linea al ruolo per la posizione dei risultati delle query Athena

1. Nella pagina Riepilogo per il ruolo `Athena-LakeFormation-OktaRole`, nella tab Autorizzazioni scegli Aggiungi policy in linea.
2. Nella pagina Create policy (Crea policy), scegli JSON.
3. Aggiungi una policy in linea come la seguente, che consente al ruolo di accedere alla posizione dei risultati delle query Athena. Sostituisci i segnaposto *< athena-query-results-bucket >* nell'esempio con il nome del tuo bucket Amazon S3.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AthenaQueryResultsPermissionsForS3",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::<athena-query-results-bucket>",
```

```
                "arn:aws:s3:::<athena-query-results-bucket>/*"
            ]
        }
    ]
}
```

4. Scegli Verifica policy.
5. Per Nome, inserire un nome per la policy, ad esempio **`AthenaQueryResultsInlinePolicy`**.
6. Scegli Crea policy.

Successivamente, copia l'ARN del ruolo di accesso Lake Formation e l'ARN del provider SAML creato. Questi sono necessari quando configuri l'applicazione Okta SAML nella sezione successiva dell'esercitazione.

Per copiare l'ARN del ruolo e l'ARN del provider di identità SAML

1. Dalla console IAM, nella pagina Riepilogo per il ruolo `Athena-LakeFormation-OktaRole`, scegli l'icona Copia negli appunti accanto ad ARN ruolo. L'ARN ha il formato seguente:

```
arn:aws:iam::<account-id>:role/Athena-LakeFormation-OktaRole
```

2. Salvare l'ARN completo in modo sicuro per riferimento futuro.
3. Nel pannello di navigazione della console IAM, seleziona Provider di identità.
4. Scegli il provider. AthenaLakeFormationOkta
5. Nella pagina Riepilogo, scegli l'icona Copia negli appunti accanto ad ARN fornitore. L'ARN avrà un aspetto simile al seguente:

```
arn:aws:iam::<account-id>:saml-provider/AthenaLakeFormationOkta
```

6. Salvare l'ARN completo in modo sicuro per riferimento futuro.

Passaggio 5: aggiungere il ruolo IAM e il provider di identità SAML all'applicazione Okta

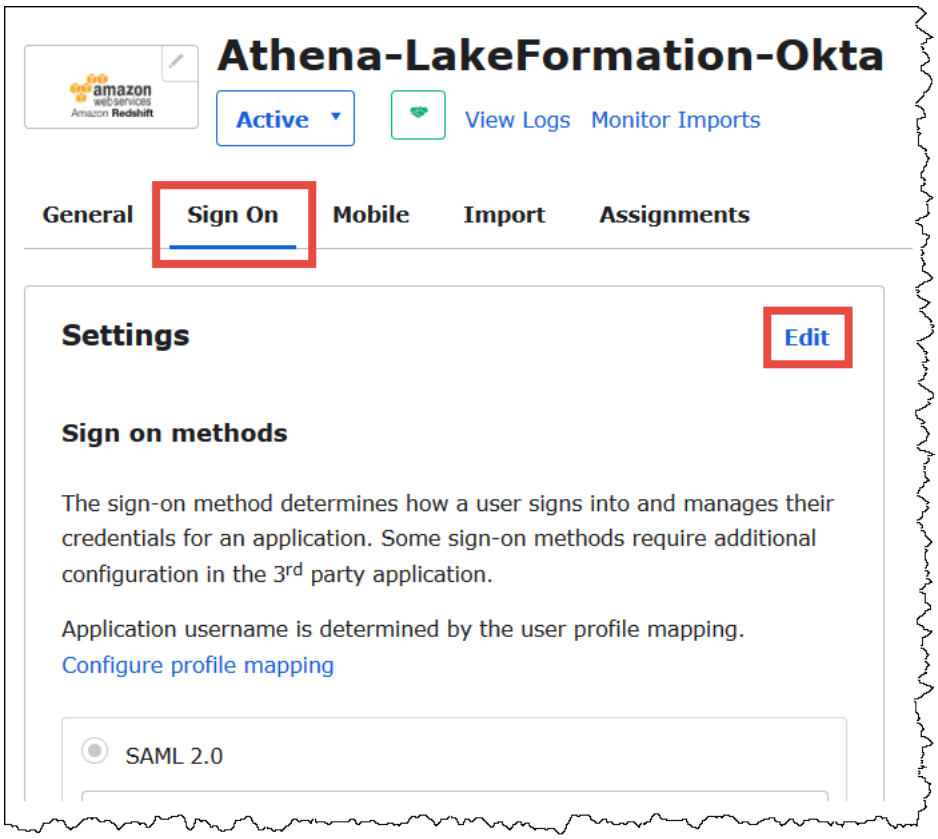
In questo passaggio, si torna alla console di sviluppo Okta ed eseguire le seguenti attività:

- Aggiungere attributi URL utente e gruppo Lake Formation all'applicazione Okta.
- Aggiungere l'ARN per il provider di identità e l'ARN per il ruolo IAM all'applicazione Okta.

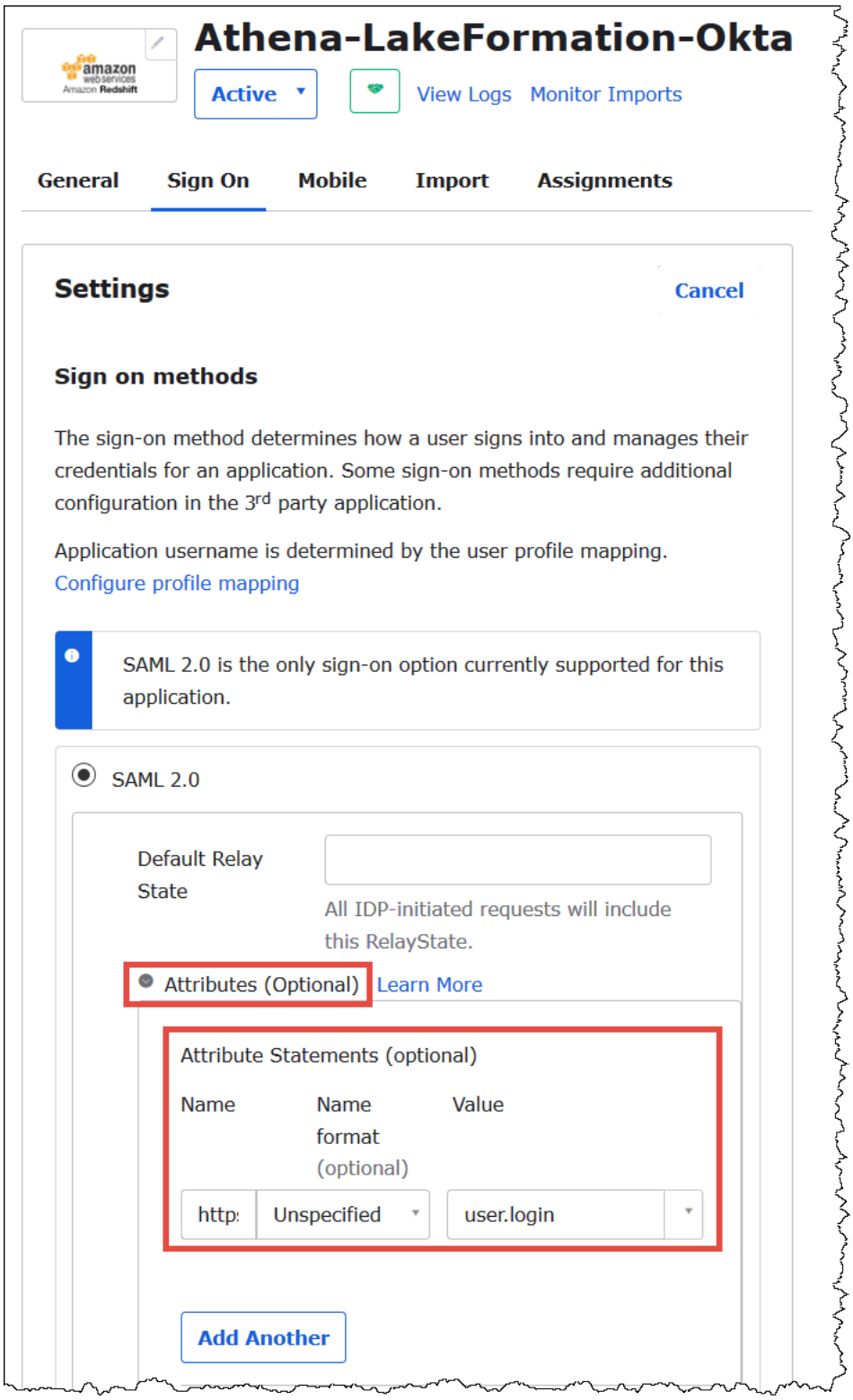
- Copiare l'ID dell'applicazione Okta. L'ID dell'applicazione Okta è obbligatorio nel profilo JDBC che si connette ad Athena.

Per aggiungere attributi URL utente e gruppo Lake Formation all'applicazione Okta

1. Accedere alla console degli sviluppatori Okta.
2. Scegli la tab Applicazioni, quindi scegli l'applicazione `Athena-LakeFormation-Okta`.
3. Scegliere nella tab Accedi per l'applicazione, quindi scegli Modifica.



4. Scegli Attributi (opzionali) per espanderla.



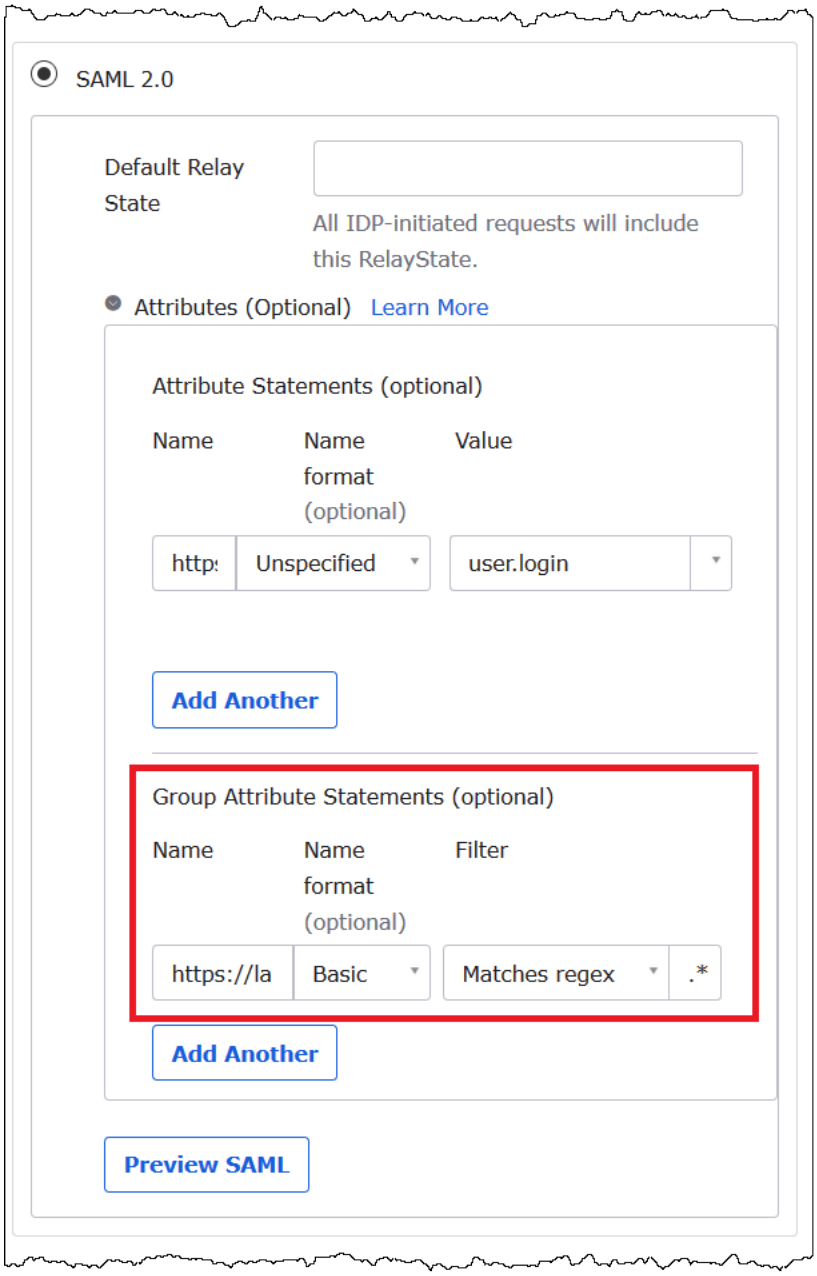
5. Per Istruzioni degli attributi (opzionali), aggiungi il seguente attributo:

   - Per Nome, immetti **`https://lakeformation.amazon.com/SAML/Attributes/Username`**.

- In Valore, inserire **`user.login`**.

6. Sotto Istruzioni degli attributi per i gruppi (opzionali), aggiungi il seguente attributo:

   - Per Nome, immetti **`https://lakeformation.amazon.com/SAML/Attributes/Groups`**.
   - Per Formato nome, inserisci **`Basic`**
   - Per Filtro, scegli Corrisponde alla regex, quindi inserisci **`.*`** nella casella filtro.

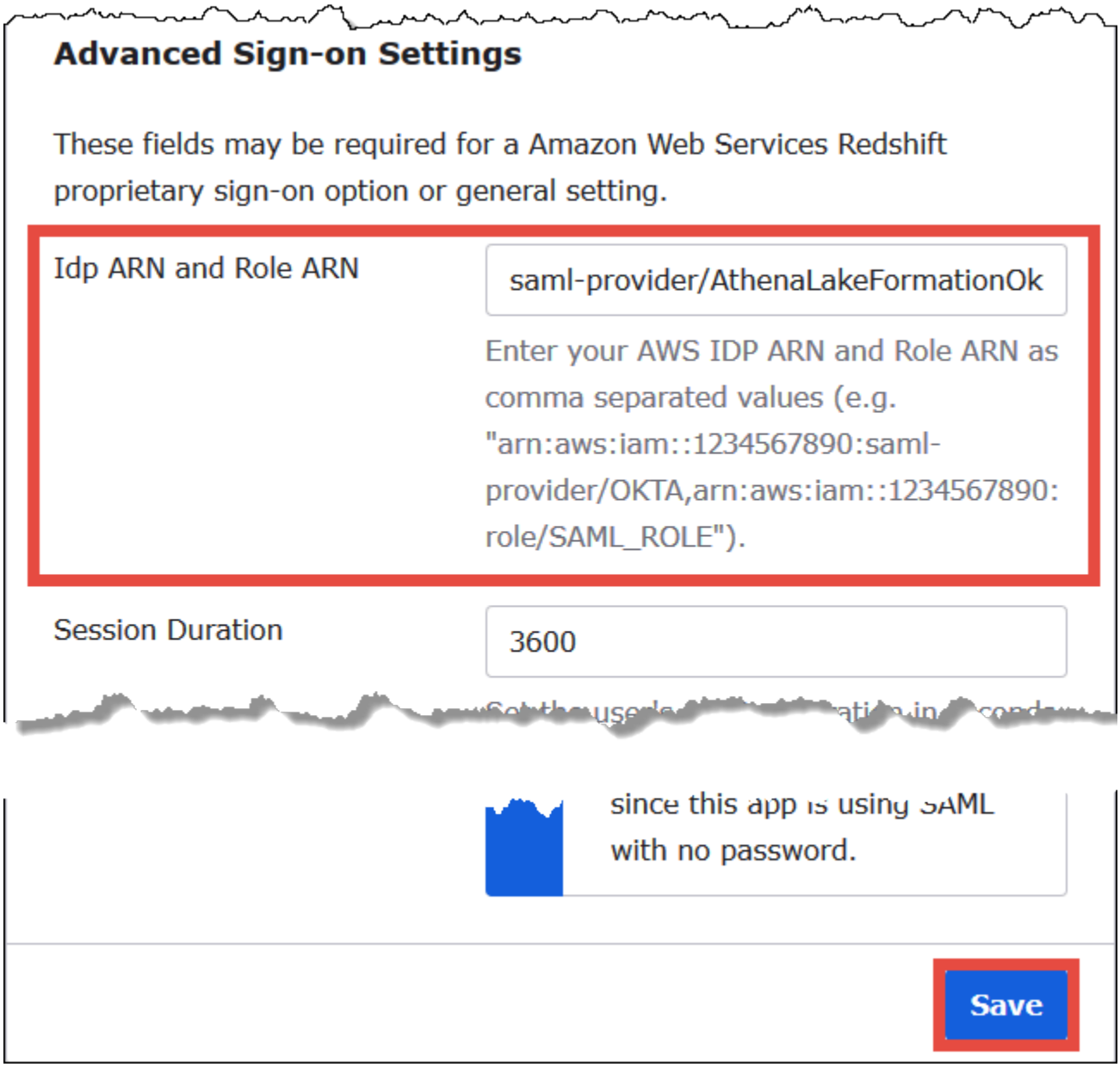


7. Scorri in basso fino alla sezione Impostazioni avanzate di accesso, in cui verranno aggiunti il provider di identità e gli ARN dei ruoli IAM all'applicazione Okta.

Per aggiungere gli ARN per il provider di identità e il ruolo IAM all'applicazione Okta

1. <saml-arn><role-arn>Per Idp ARN e Role ARN, immettere l'ARN del provider di AWS identità e il ruolo ARN come valori separati da virgole nel formato,. La stringa combinata dovrebbe essere simile alla seguente:

```
arn:aws:iam::<account-id>:saml-provider/
AthenaLakeFormationOkta,arn:aws:iam::<account-id>:role/Athena-LakeFormation-
OktaRole
```

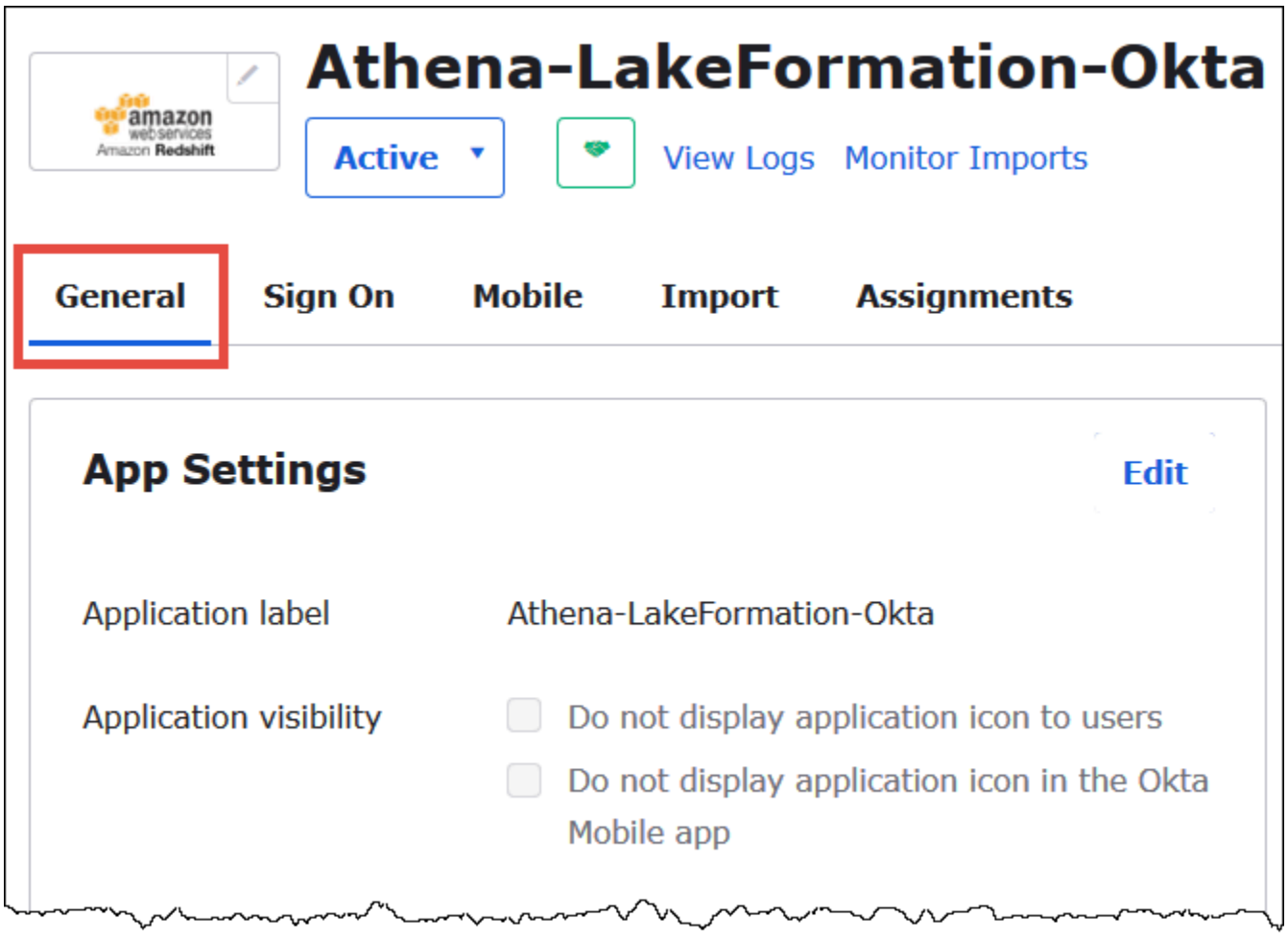


2. Selezionare Salva.

Successivamente, copiare l'ID dell'applicazione Okta. Ne avrai bisogno più tardi per la stringa JDBC che si connette ad Athena.

Per trovare e copiare l'ID dell'applicazione Okta

1. Selezionare la tab Generale dell'applicazione Okta.



2. Scorri in basso fino alla sezione Link per l'incorporamento dell'app.
3. Da Incorpora link, copia e salva in modo sicuro la porzione dell'ID dell'applicazione Okta dell'URL. L'ID dell'applicazione Okta è la parte dell'URL dopo `amazon_aws_redshift/` ma prima della barra in avanti successiva. Ad esempio, se l'URL contiene `amazon_aws_redshift/aaa/bbb`, l'ID dell'applicazione è `aaa`.

> **Note**
>
> Il link per l'incorporamento non può essere utilizzato per accedere direttamente alla console Athena per visualizzare i database. Le autorizzazioni Lake Formation per utenti e gruppi SAML vengono riconosciute solo quando si utilizza il driver JDBC oppure ODBC per inviare query ad Athena. Per visualizzare i database, puoi utilizzare lo strumento SQL Workbench/J, che utilizza il driver JDBC per connettersi ad Athena. Lo strumento SQL Workbench/J viene descritto in Passaggio 7: Verificare l'accesso tramite il client Athena JDBC.

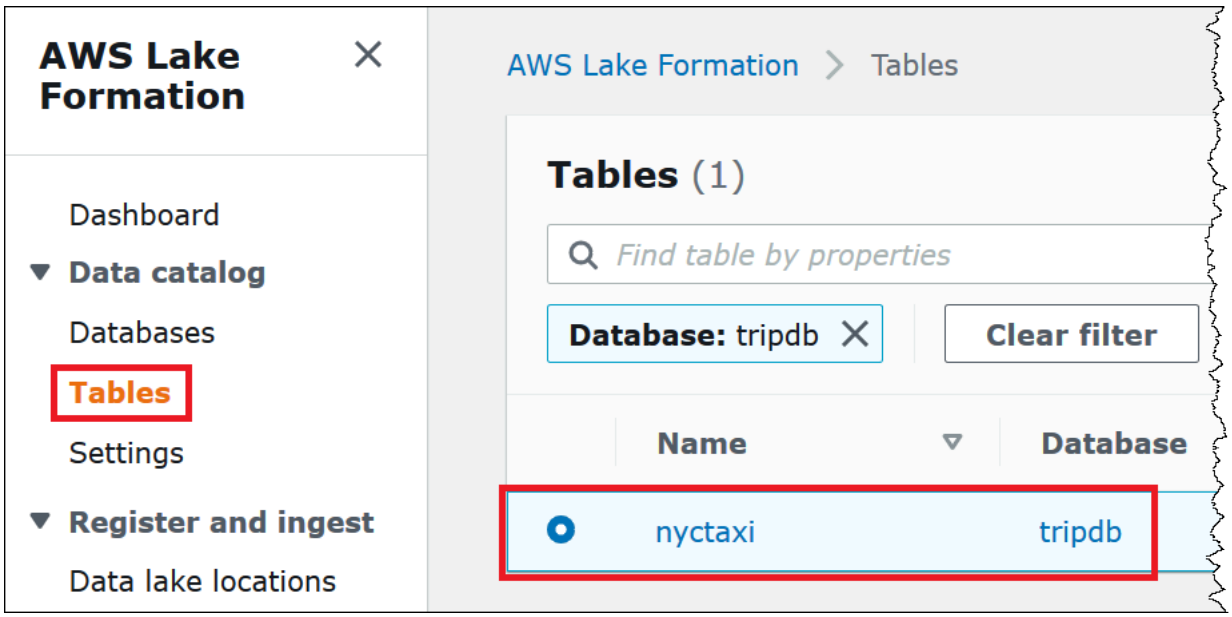
Passaggio 6: concedere le autorizzazioni a utenti e gruppi tramite AWS Lake Formation

In questo passaggio, utilizzi la console Lake Formation per concedere autorizzazioni per una tabella all'utente e al gruppo SAML. Si possono eseguire queste attività:

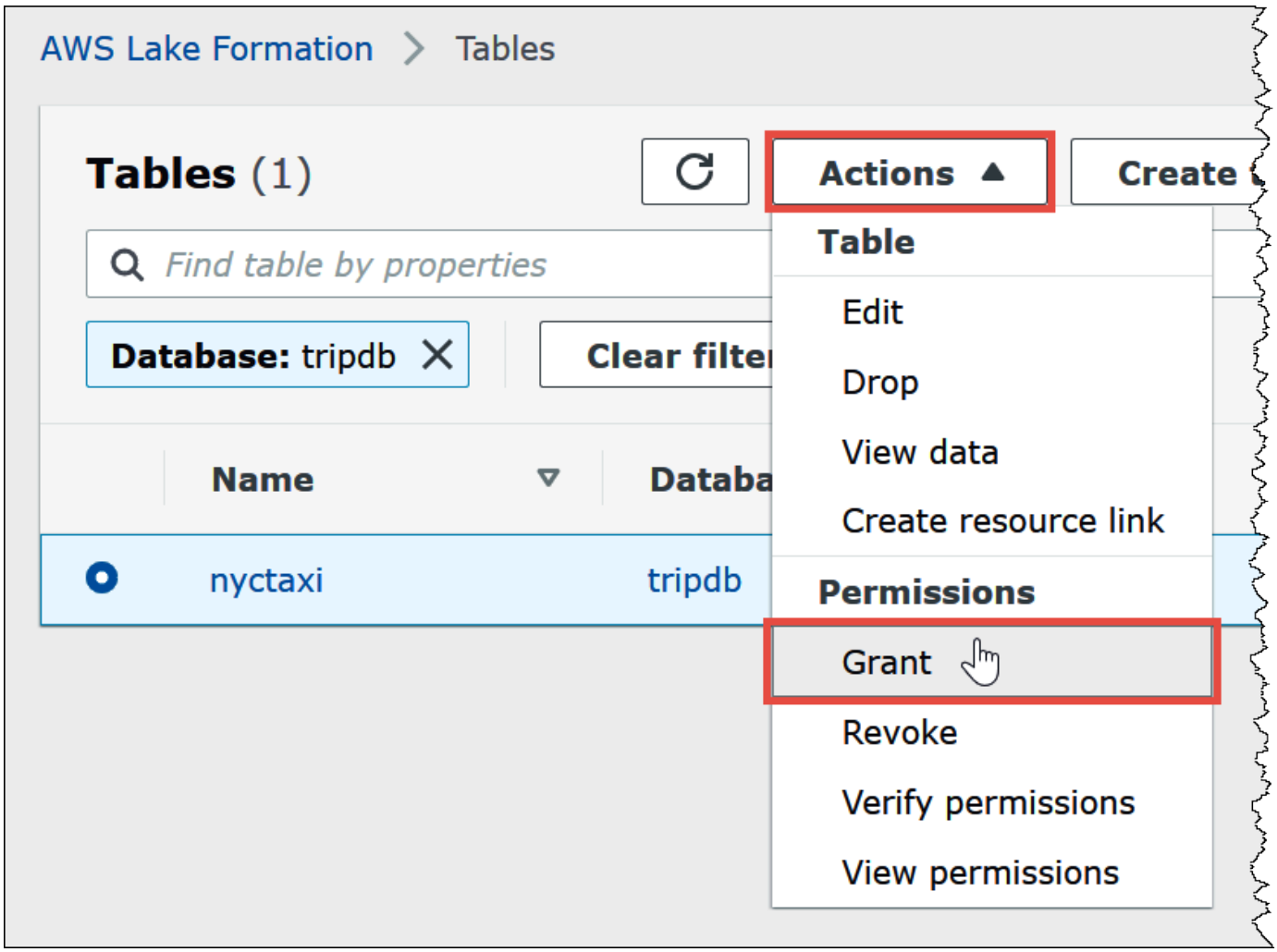
- Specificare l'ARN dell'utente Okta SAML e le autorizzazioni utente associate nella tabella.
- Specificare l'ARN del gruppo Okta SAML e le autorizzazioni gruppo associate nella tabella.
- Verificare le autorizzazioni concesse.

Per concedere autorizzazioni in Lake Formation per l'utente Okta

1. Accedi come amministratore data lake alla AWS Management Console.
2. Aprire la console Lake Formation all'indirizzo https://console.aws.amazon.com/lakeformation/.
3. Dal pannello di navigazione, seleziona Tabelle, quindi seleziona la tabella per la quale desideri concedere le autorizzazioni. Questo tutorial utilizza la tabella `nyctaxi` dal database `tripdb`.



4. Da Operazioni, scegliere Concessione.



5. Nella finestra di dialogo Concedi autorizzazioni inserisci le seguenti informazioni:

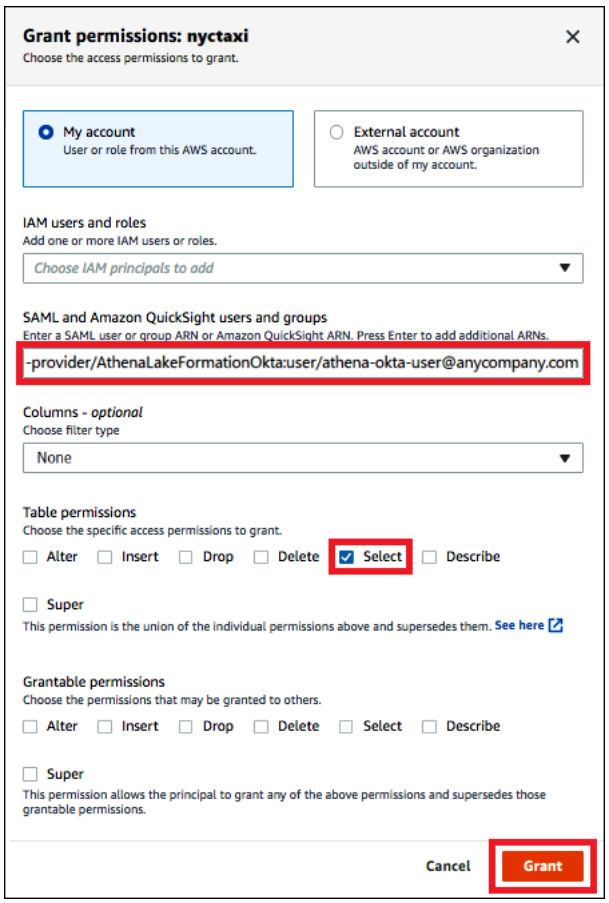
   a. In QuickSight Utenti e gruppi SAML e Amazon, inserisci l'ARN dell'utente SAML Okta nel seguente formato:

   ```
   arn:aws:iam::<account-id>:saml-provider/AthenaLakeFormationOkta:user/<athena-okta-user>@<anycompany.com>
   ```

   b. Per Colonne, per Scegli il tipo di filtro e, facoltativamente, scegli Includi colonne o Escludi colonne.

   c. Usa il menu a discesa Scegli una o più colonne sotto il filtro per specificare le colonne che desideri includere o escludere per o dall'utente.

   d. Per Autorizzazioni tabella, scegli Seleziona. Questa esercitazione concede solo l'autorizzazione `SELECT`; i requisiti possono variare.



6. Scegli Concessione.

Ora eseguire passaggi simili per il gruppo Okta.

Per concedere autorizzazioni in Lake Formation per il gruppo Okta

1. Sulla pagina Tabelle della console Lake Formation, assicurarsi che la tabella nyctaxi sia ancora selezionata.
2. Da Operazioni, scegliere Concessione.
3. Nella finestra di dialogo Concedi autorizzazioni inserisci le seguenti informazioni:

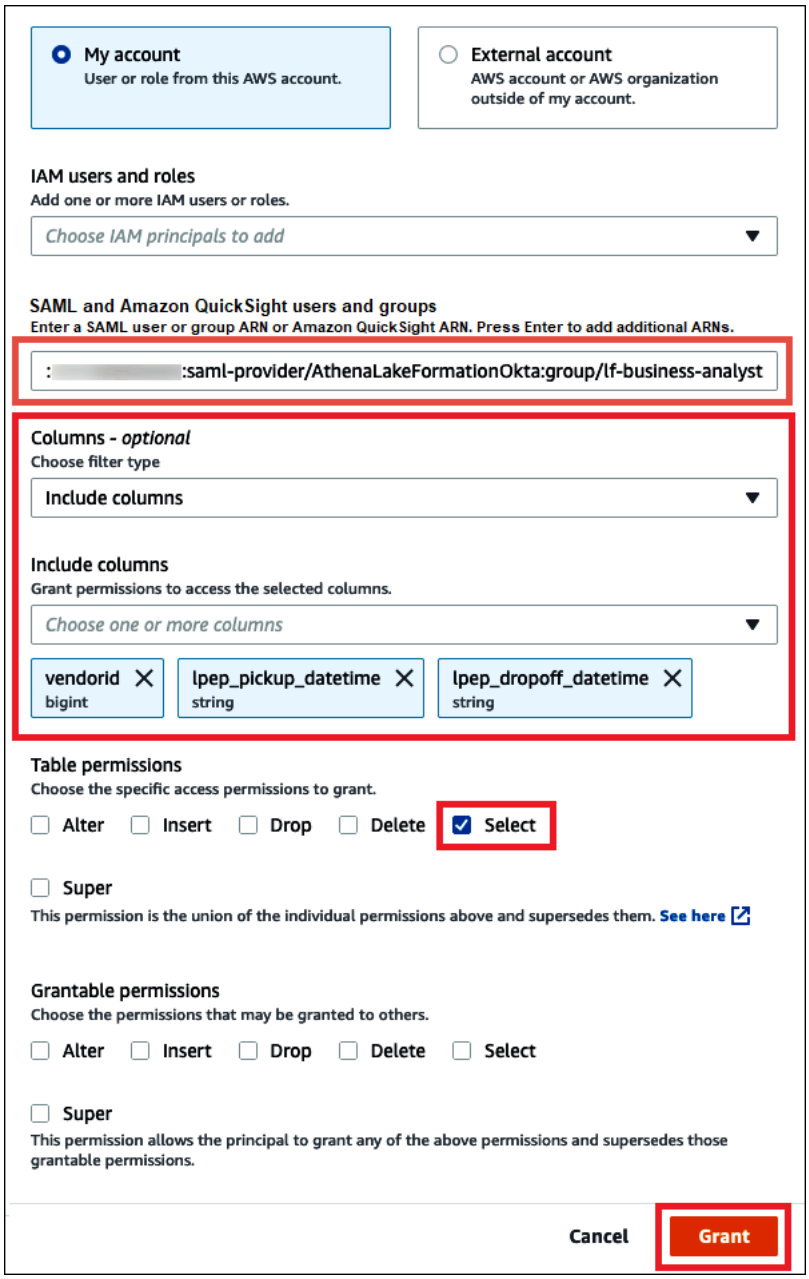
    a. In QuickSight Utenti e gruppi SAML e Amazon, inserisci l'ARN del gruppo Okta SAML nel seguente formato:

    ```
    arn:aws:iam::<account-id>:saml-provider/AthenaLakeFormationOkta:group/lf-business-analyst
    ```

    b. PerColonne,scegli il tipo di filtro, scegli Includi colonne.

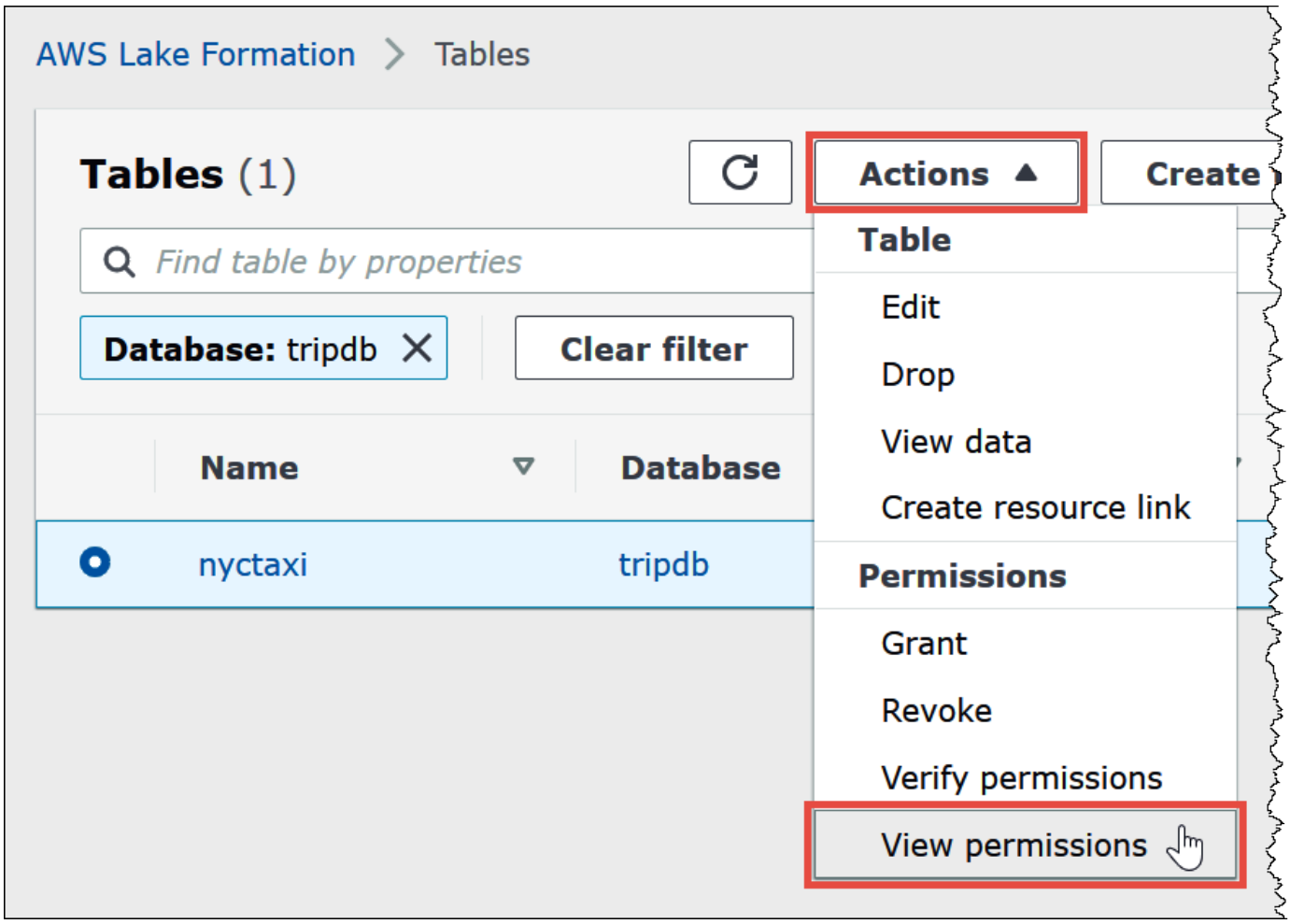
c. Per Scegli una o più colonne, scegliere le prime tre colonne della tabella.

d. Per Autorizzazioni tabella, scegli le autorizzazioni di accesso specifiche da concedere. Questa esercitazione concede solo l'autorizzazione `SELECT`; i requisiti possono variare.

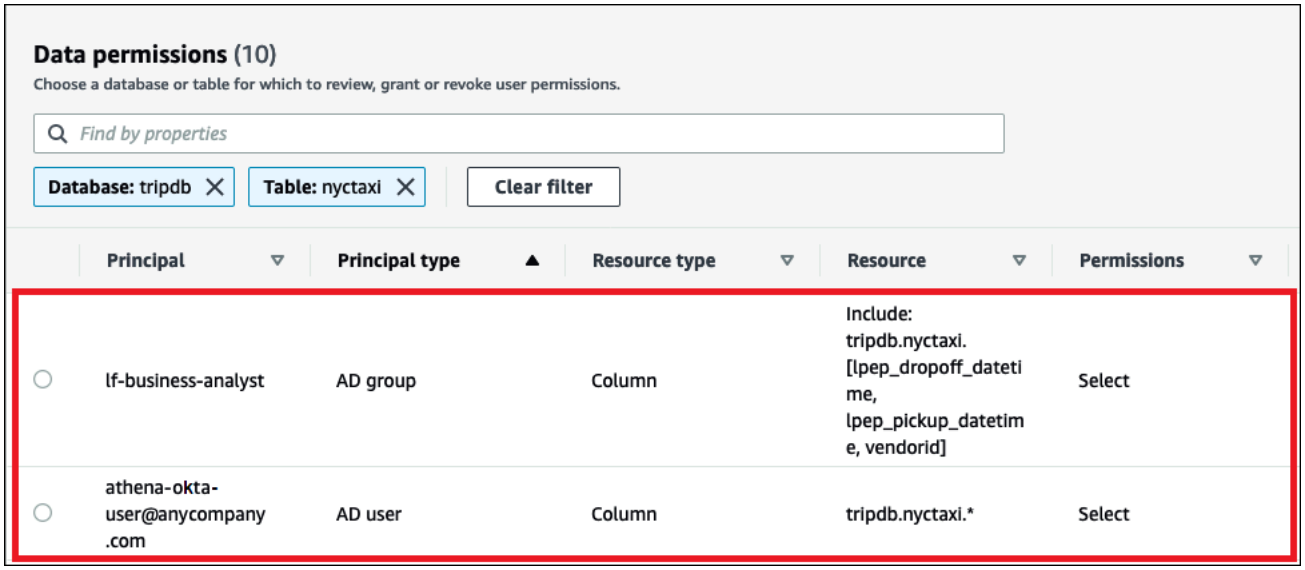


4. Scegli Concessione.

5. Per verificare le autorizzazioni concesse, scegli Azioni, Visualizza autorizzazioni.



La pagina delle autorizzazioni relative ai dati della `nyctaxi` tabella mostra le autorizzazioni per e il gruppo. athena-okta-userlf-business-analyst



Passaggio 7: Verificare l'accesso tramite il client Athena JDBC

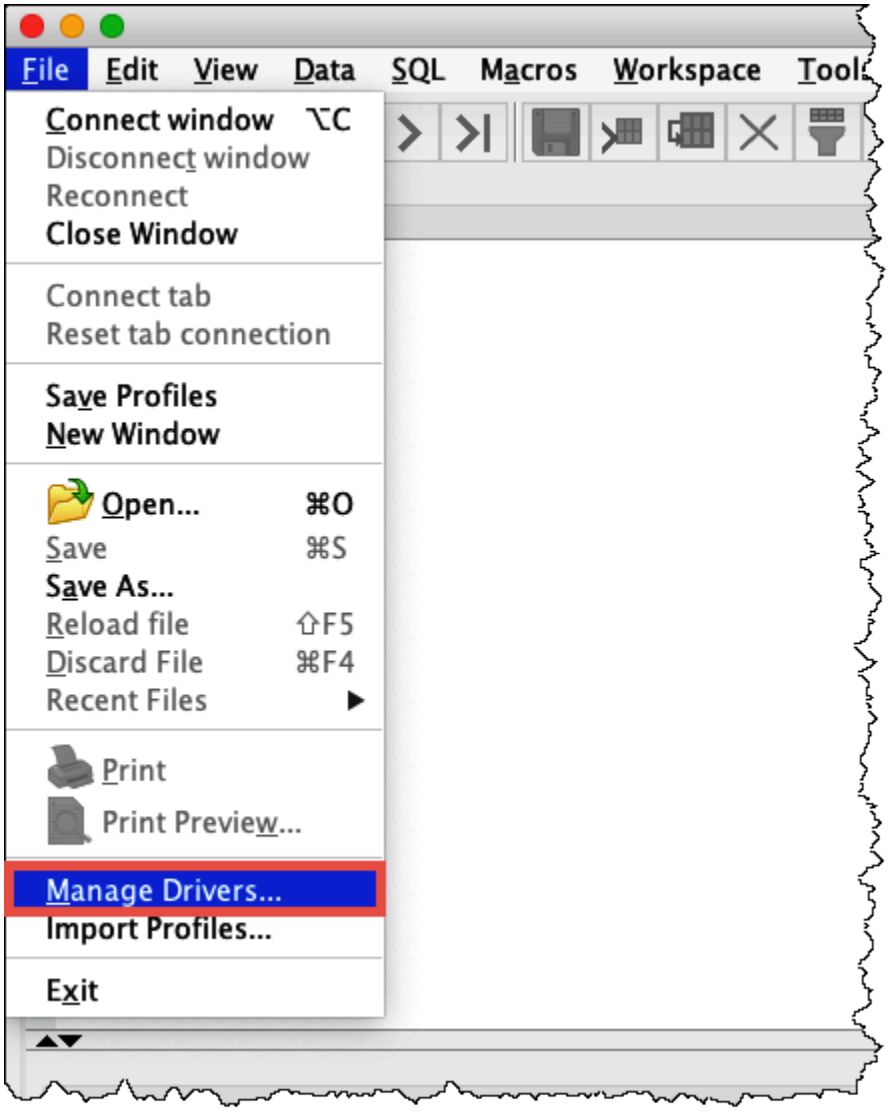
Ora puoi utilizzare un client JDBC per eseguire una connessione di prova ad Athena come utente Okta SAML.

In questa sezione esegui le seguenti attività:

- Prepara il client test: scarica il driver Athena JDBC, installa SQL Workbench e aggiungi il driver a Workbench. Questo tutorial utilizza SQL Workbench per accedere ad Athena tramite l'autenticazione Okta e per verificare le autorizzazioni Lake Formation.
- In SQL Workbench:
  - Creare una connessione per l'utente Athena Okta.
  - Eseguire query di test come utente Athena Okta.
  - Creare e testare una connessione per l'utente business analyst.
- Nella console Okta, aggiungere l'utente business analyst al gruppo di sviluppatori.
- Nella console Lake Formation, configurare le autorizzazioni di tabella per il gruppo di sviluppatori.
- In SQL Workbench, esegui query di test come utente business analyst e verifica in che modo la modifica delle autorizzazioni influisce sui risultati.

Per preparare il client di test

1. Scaricare ed estrarre il driver Athena JDBC compatibile con Lake Formation (2.0.14 o versione successiva) da Connessione ad Amazon Athena con JDBC.
2. Scaricare e installare lo strumento di query SQL gratuito SQL Workbench/J disponibile con una licenza Apache 2.0 modificata.
3. In SQL Workbench, scegliere File, quindi scegliere Gestisci driver.



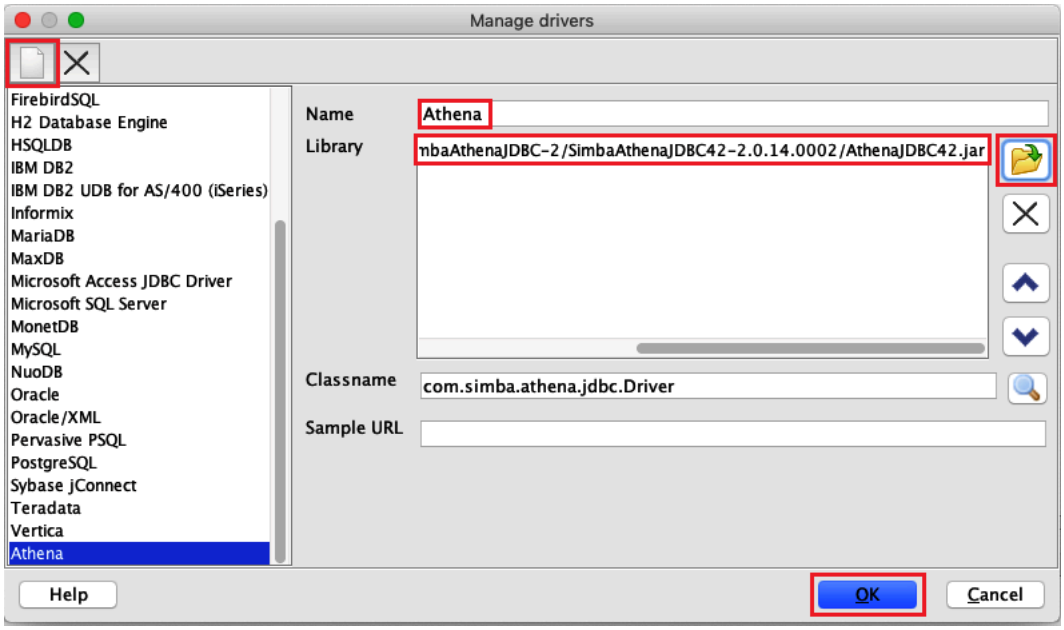
4. Nella finestra di dialogo Gestisci driver completare la seguente procedura:

   a. Scegliere l'icona del nuovo driver.

   b. Per Nome, immetti **`Athena`**.

   c. Per Libreria, scegliere il file `.jar` Simba Athena JDBC appena scaricato.

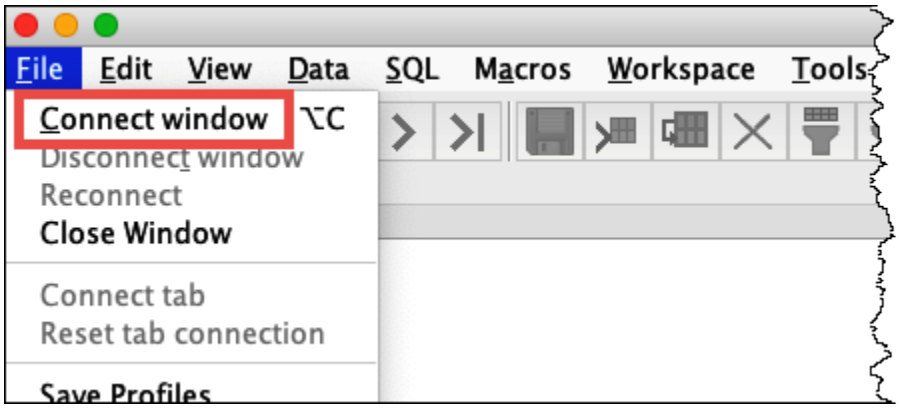
   d. Scegli OK.



Ora puoi creare e testare una connessione per l'utente Athena Okta.

Per creare una connessione per l'utente Okta

1. Scegliere File, Finestra Connect.



2. Nella finestra di dialogo Profilo di connessione creare una connessione inserendo le informazioni seguenti:

   - Nella casella Nome, inserire **`Athena_Okta_User_Connection`**.
   - Per Driver, scegliere il driver JDBC Simba Athena.
   - Per l'URL, esegui una delle seguenti operazioni:
     - Per utilizzare un URL di connessione, inserire una stringa di connessione a riga singola. L'esempio seguente aggiunge interruzioni di riga per migliorare la leggibilità.

```
jdbc:awsathena://AwsRegion=region-id;
S3OutputLocation=s3://DOC-EXAMPLE-BUCKET/athena_results;
AwsCredentialsProviderClass=com.simba.athena.iamsupport.plugin.OktaCredentialsProvider;
user=athena-okta-user@anycompany.com;
password=password;
idp_host=okta-idp-domain;
App_ID=okta-app-id;
SSL_Insecure=true;
LakeFormationEnabled=true;
```

- Per utilizzare un URL AWS basato sul profilo, procedi nel seguente modo:

  1. Configura un AWS profilo con un file di AWS credenziali come nell'esempio seguente.

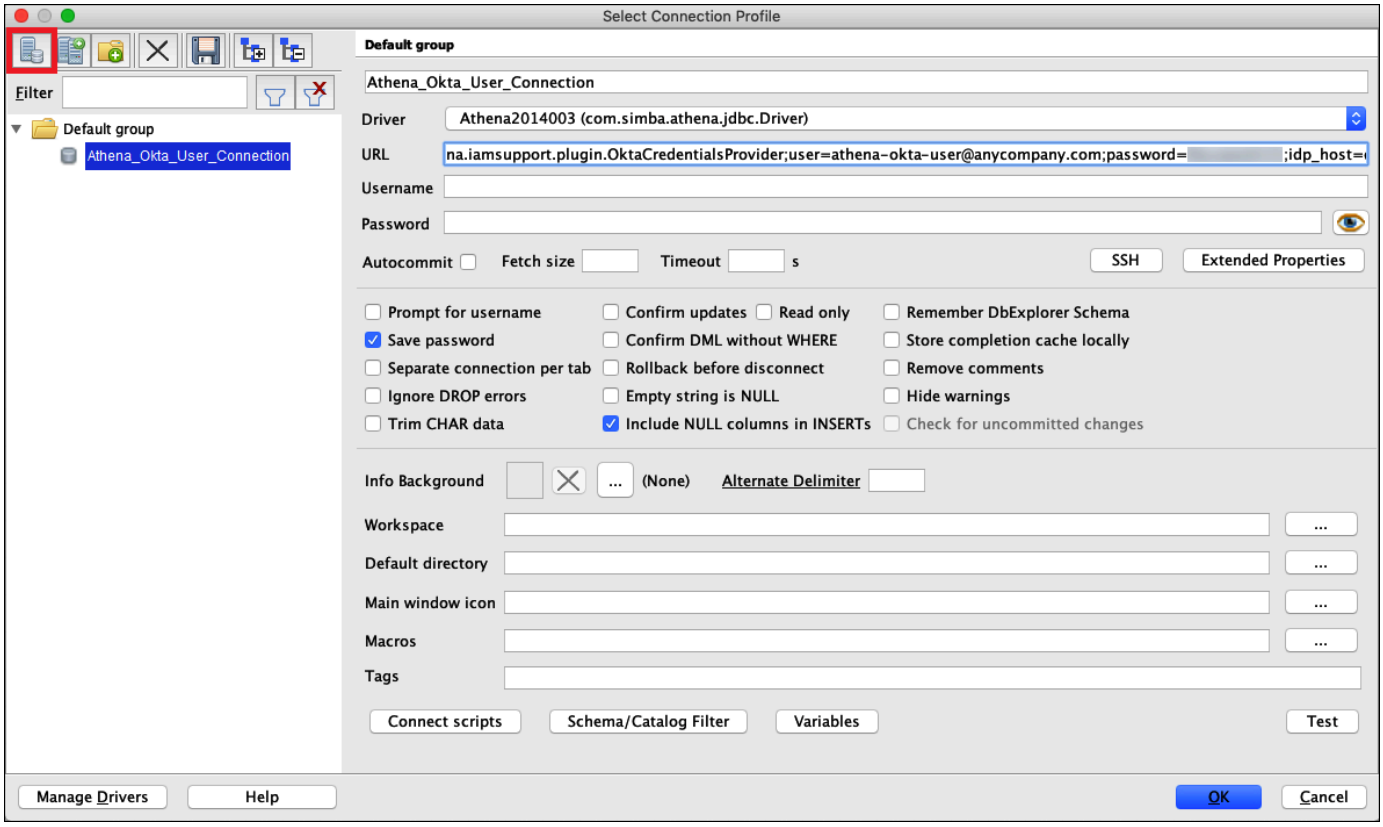
     ```
     [athena_lf_dev]
     plugin_name=com.simba.athena.iamsupport.plugin.OktaCredentialsProvider
     idp_host=okta-idp-domain
     app_id=okta-app-id
     uid=athena-okta-user@anycompany.com
     pwd=password
     ```

  2. Per URL, inserisci una stringa di connessione a riga singola come nel seguente esempio. L'esempio aggiunge interruzioni di riga per migliorare la leggibilità.

     ```
     jdbc:awsathena://AwsRegion=region-id;
     S3OutputLocation=s3://DOC-EXAMPLE-BUCKET/athena_results;
     profile=athena_lf_dev;
     SSL_Insecure=true;
     LakeFormationEnabled=true;
     ```

  Questi esempi sono rappresentazioni di base dell'URL necessario per connettersi ad Athena. Per l'elenco completo dei parametri supportati nell'URL, consulta la documentazione JDBC.

L'immagine seguente mostra un profilo di connessione SQL Workbench che utilizza un URL di connessione.

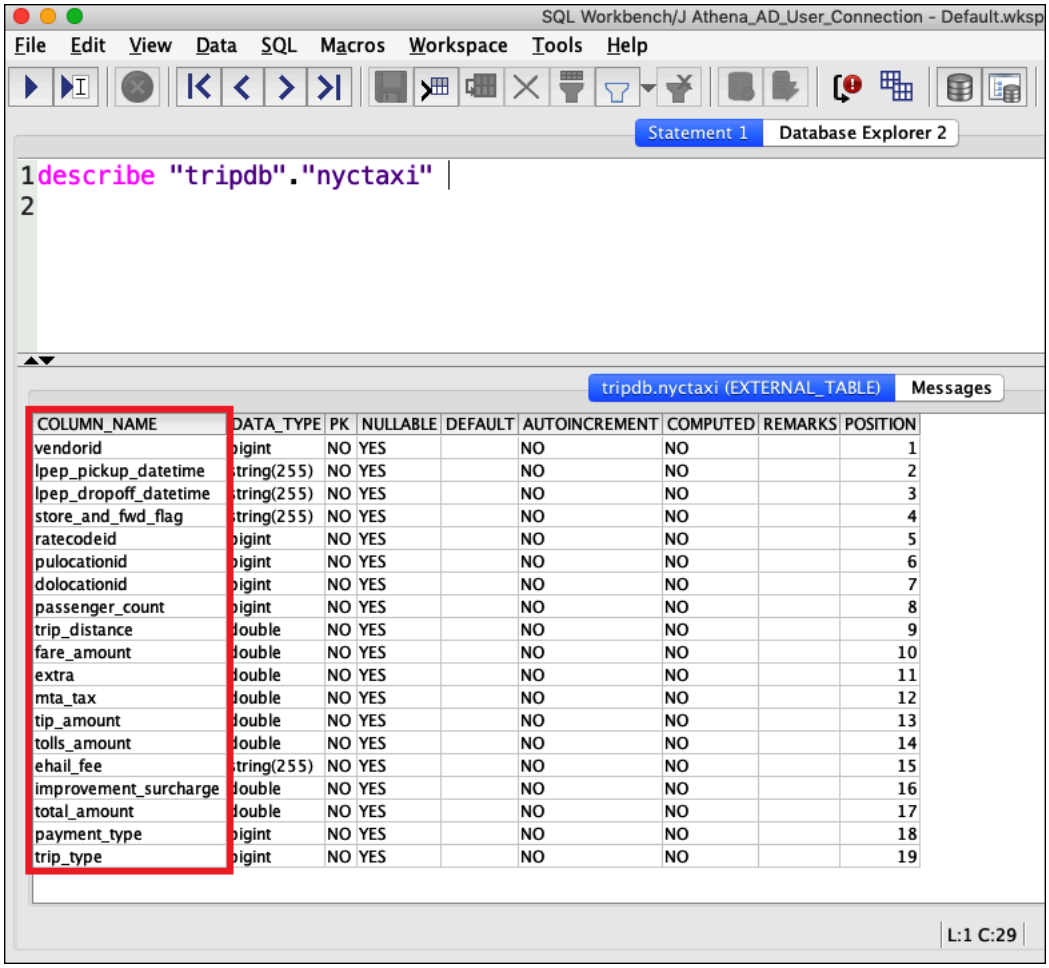


Dopo aver stabilito una connessione per l'utente Okta, è possibile testarla recuperando alcuni dati.

Per testare la connessione per l'utente Okta

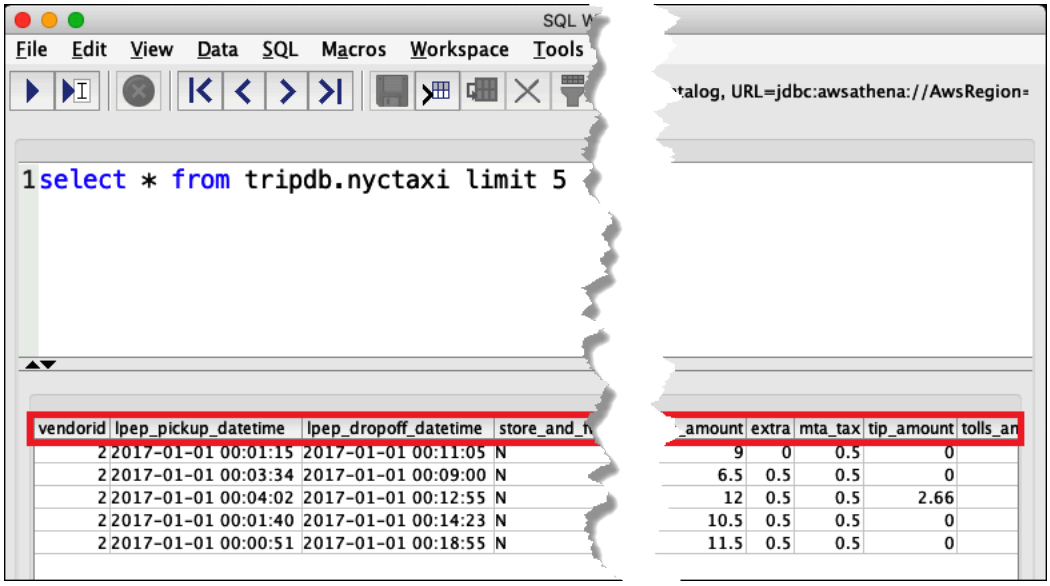
1. Scegli Test e quindi verifica che la connessione abbia esito positivo.
2. Dalla finestra Istruzione di SQL Workbench, eseguire il comando SQL `DESCRIBE` seguente. Verificare che siano mostrate tutte le colonne.

```
DESCRIBE "tripdb"."nyctaxi"
```



3. Dalla finestra Istruzione di SQL Workbench, eseguire il comando SQL SELECT seguente. Verificare che siano mostrate tutte le colonne.

```
SELECT * FROM tripdb.nyctaxi LIMIT 5
```



Successivamente, verifichi che athena-ba-user, come membro del lf-business-analystgruppo, abbia accesso solo alle prime tre colonne della tabella specificata in precedenza in Lake Formation.

Per verificare l'accesso per athena-ba-user

1. In SQL Workbench, nella finestra di dialogo Profilo di connessione creare un altro profilo di connessione.
   - Per il nome del profilo di connessione, inserire **`Athena_Okta_Group_Connection`**.
   - Per Driver, scegliere il driver JDBC Simba Athena.
   - Per l'URL, esegui una delle seguenti operazioni:
     - Per utilizzare un URL di connessione, inserire una stringa di connessione a riga singola. L'esempio seguente aggiunge interruzioni di riga per migliorare la leggibilità.

```
jdbc:awsathena://AwsRegion=region-id;
S3OutputLocation=s3://DOC-EXAMPLE-BUCKET/athena_results;
AwsCredentialsProviderClass=com.simba.athena.iamsupport.plugin.OktaCredentialsProvider;
user=athena-ba-user@anycompany.com;
password=password;
idp_host=okta-idp-domain;
App_ID=okta-application-id;
SSL_Insecure=true;
LakeFormationEnabled=true;
```

- Per utilizzare un URL AWS basato sul profilo, effettuate le seguenti operazioni:

  1. Configura un AWS profilo con un file di credenziali come nell'esempio seguente.

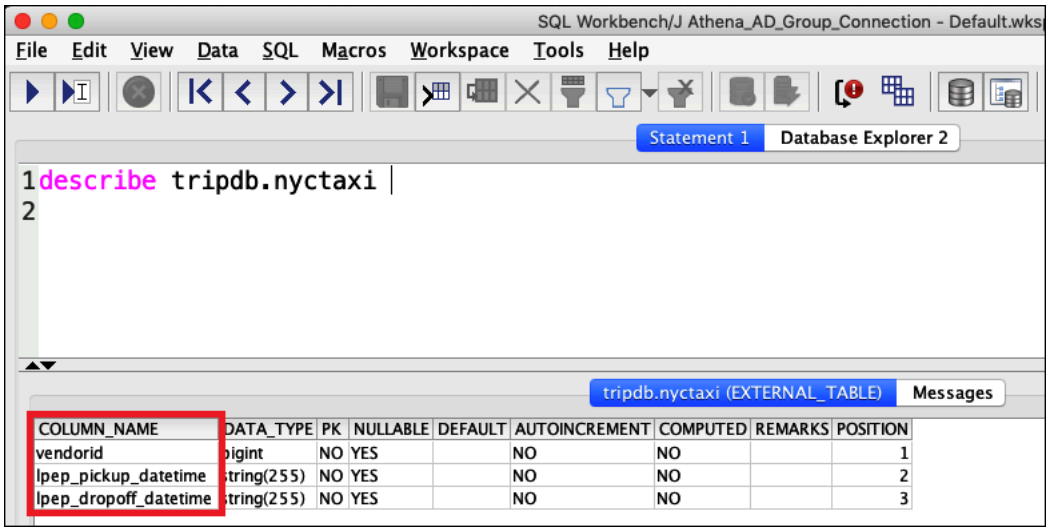
```
[athena_lf_ba]
plugin_name=com.simba.athena.iamsupport.plugin.OktaCredentialsProvider
idp_host=okta-idp-domain
app_id=okta-application-id
uid=athena-ba-user@anycompany.com
pwd=password
```

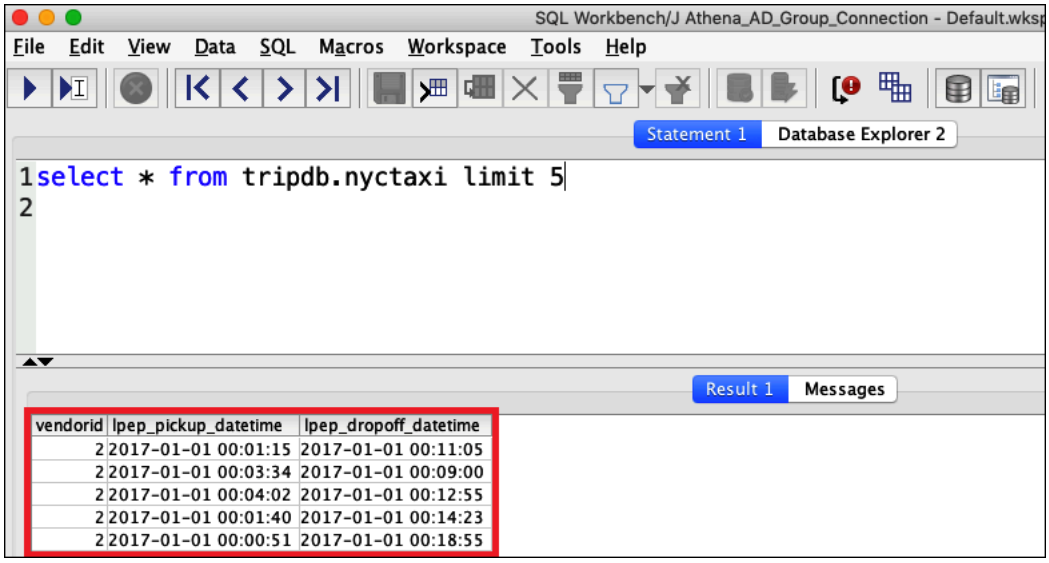
  2. Per URL, inserire una stringa di connessione a riga singola come la seguente. L'esempio aggiunge interruzioni di riga per migliorare la leggibilità.

```
jdbc:awsathena://AwsRegion=region-id;
S3OutputLocation=s3://DOC-EXAMPLE-BUCKET/athena_results;
profile=athena_lf_ba;
SSL_Insecure=true;
LakeFormationEnabled=true;
```

2. Scegli Test per confermare che la connessione funzioni.
3. Dalla finestra Istruzione SQL, eseguire gli stessi comandi SQL `DESCRIBE` e `SELECT` eseguiti in precedenza ed esaminare i risultati.

   Poiché athena-ba-userè un membro del lf-business-analystgruppo, vengono restituite solo le prime tre colonne specificate nella console Lake Formation.

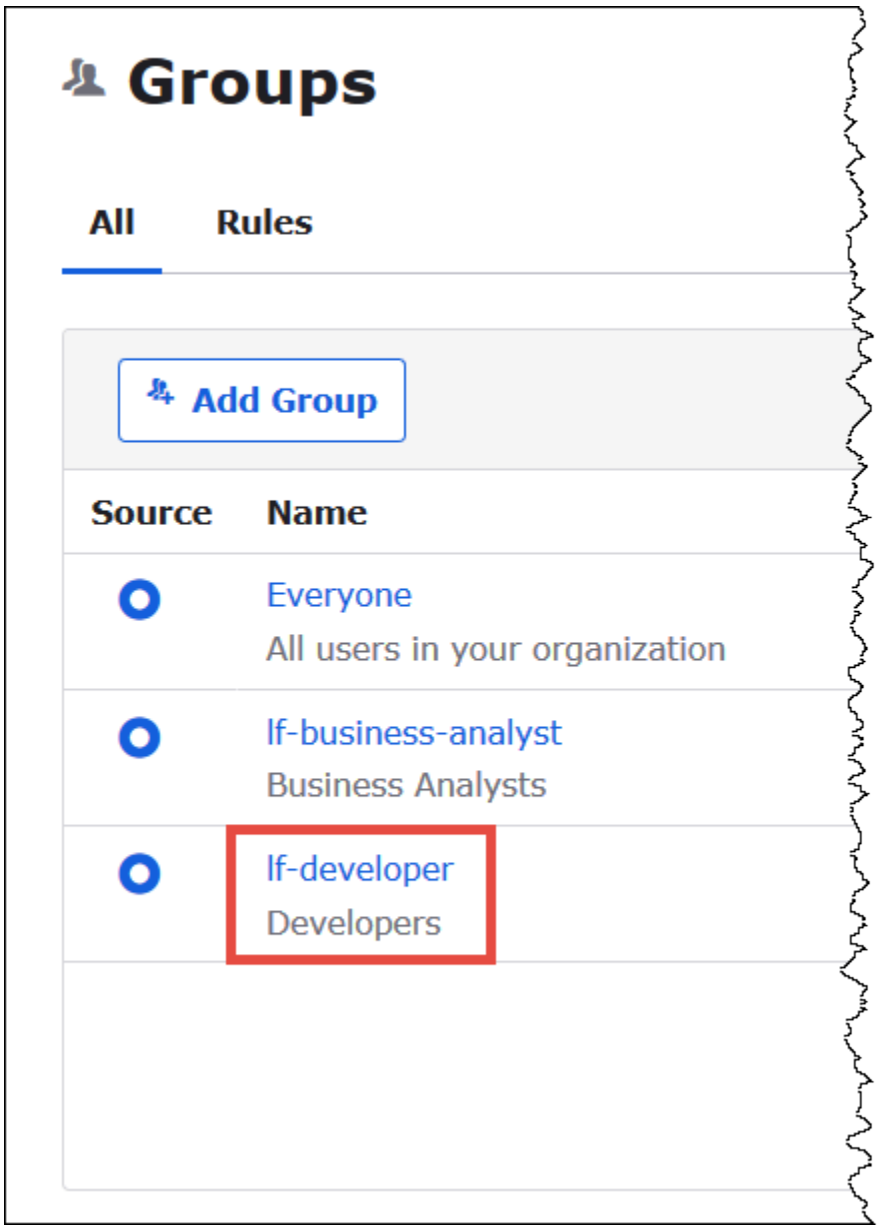




Successivamente, si torna alla console Okta per aggiungere `athena-ba-user` al gruppo Okta `lf-developer`.

Per aggiungere il athena-ba-user al gruppo lf-developer

1. Accedi alla console Okta come utente amministratore del dominio Okta assegnato.
2. Scegli Directory e quindi scegli Gruppi.
3. Nella pagina Gruppi scegli il gruppo lf-developer.



4. Scegli Gestisci persone.
5. Dall'elenco Non membri, scegli di aggiungerlo athena-ba-useral gruppo lf-developer.
6. Selezionare Salva.

Ora torna alla console Lake Formation per configurare le autorizzazioni di tabella per il gruppo lf-developer.

Per configurare i permessi delle tabelle per lf-developer-group

1. Accedi alla console Lake Formation come amministratore Data Lake.

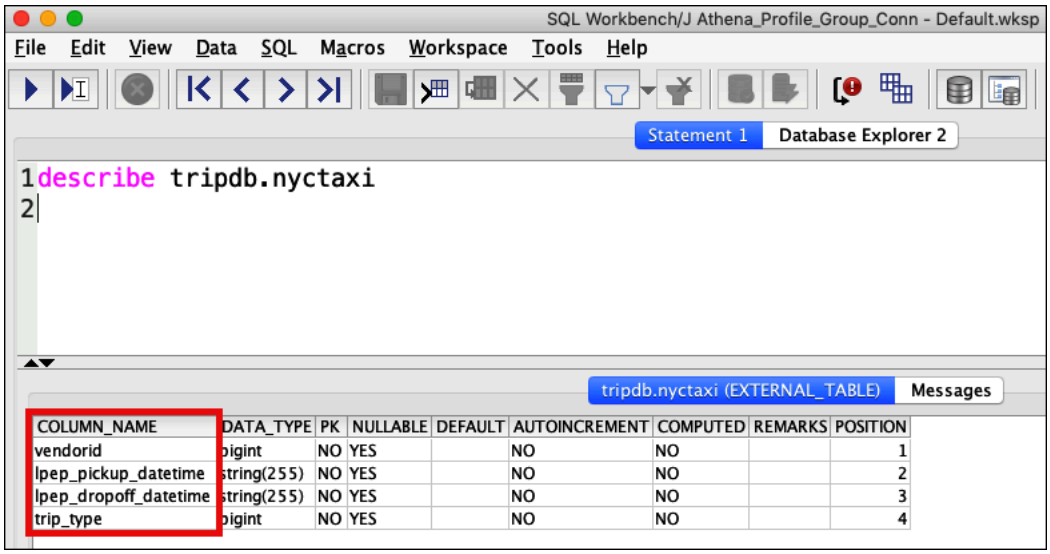
2. Nel pannello di navigazione, seleziona Tabelle.
3. Seleziona la tabella nyctaxi.
4. Scegli Operazioni, Concessione.
5. Nella finestra di dialogo Concessione di autorizzazioni inserire le informazioni riportate di seguito:
   - Per QuickSight utenti e gruppi SAML e Amazon, inserisci l'ARN del gruppo di sviluppatori Okta SAML lf-developer nel seguente formato:
   - PerColonne,scegli il tipo di filtro, scegli Includi colonne.
   - Seleziona la colonna trip_type.
   - Per Autorizzazioni tabella, scegli SELEZIONA.
6. Scegli Concessione.

Ora è possibile utilizzare SQL Workbench per verificare la modifica delle autorizzazioni per il gruppo lf-developer. La modifica dovrebbe riflettersi nei dati a disposizione di athena-ba-user, che ora è membro del gruppo lf-developer.

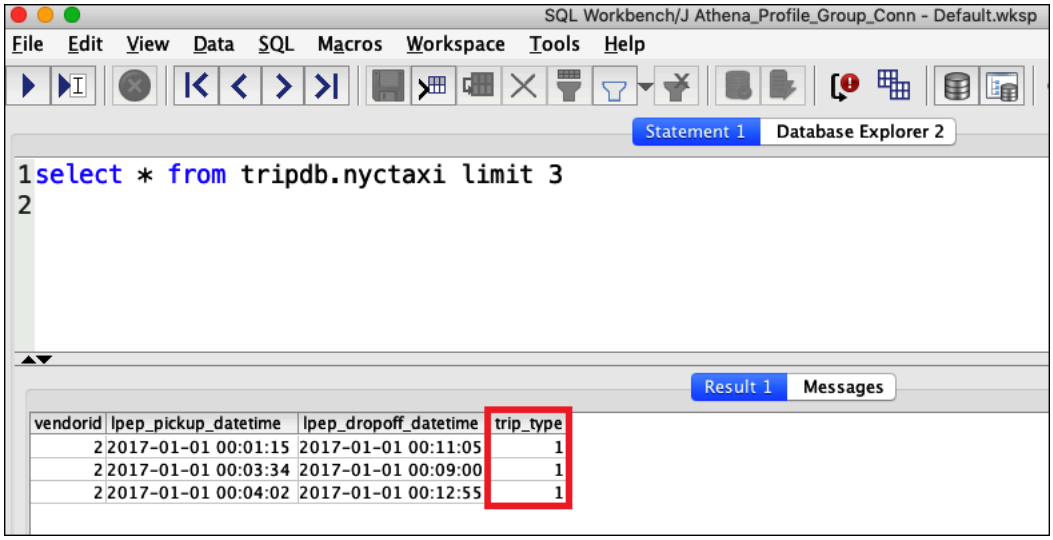
Per verificare la modifica delle autorizzazioni per athena-ba-user

1. Chiudere il programma SQL Workbench e quindi riaprirlo.
2. Connect al profilo per athena-ba-user.
3. Dalla finestra Istruzione, emettere le stesse istruzioni SQL eseguite in precedenza:

   Questa volta, viene mostrata la colonna trip_type.



Poiché ora athena-ba-userè membro sia di lf-developer che dei lf-business-analystgruppi, la combinazione dei permessi di Lake Formation per quei gruppi determina le colonne che vengono restituite.



Conclusioni

In questo tutorial hai configurato l'integrazione di Athena con l' AWS Lake Formation utilizzo di Okta come provider SAML. Hai utilizzato Lake Formation e IAM per controllare le risorse disponibili per l'utente SAML nel tuo data lake AWS Glue Data Catalog.

Risorse correlate

Per informazioni correlate, consulta le seguenti risorse.

- Connessione ad Amazon Athena con JDBC
- Abilitazione dell'accesso federato all'API Athena
- AWS Lake Formation Guida per gli sviluppatori
- Concessione e revoca delle autorizzazioni del catalogo dati nella Guida per gli sviluppatori di AWS Lake Formation .
- Provider di identità e federazione nella Guida per l'utente di IAM.
- Creazione di provider di identità SAML IAM nella Guida per l'utente di IAM.
- Abilitazione della federazione all' AWS utilizzo di Windows Active Directory, ADFS e SAML 2.0 sul Security Blog.AWS

Gestione dei carichi di lavoro

Puoi utilizzare le funzionalità del gruppo di lavoro di Athena, quali la gestione della capacità, l'ottimizzazione delle prestazioni, il supporto alla compressione, i tag e le service quotas per gestire il tuo carico di lavoro.

Argomenti

Uso dei gruppi di lavoro per controllare l'accesso alle query e i costi

Usa i gruppi di lavoro per separare utenti, team, applicazioni o carichi di lavoro e per impostare i limiti relativi alla quantità di dati che ogni query o l'intero gruppo di lavoro può elaborare e per tenere traccia dei costi. Poiché i gruppi di lavoro fungono da risorse, puoi utilizzare le policy basate su identità a livello di risorsa per controllare l'accesso a un determinato gruppo di lavoro. Puoi anche visualizzare i parametri relativi alle query in Amazon CloudWatch, controllare i costi configurando limiti alla quantità di dati scansionati, creare soglie e attivare azioni, come Amazon SNS, quando queste soglie vengono violate.

Per controllare ulteriormente i costi, puoi creare prenotazioni di capacità con il numero di unità di elaborazione dati specificato e aggiungere uno o più gruppi di lavoro alla prenotazione. Per ulteriori informazioni, consulta Gestione della capacità di elaborazione delle query.

I gruppi di lavoro si integrano con IAM CloudWatch, Amazon Simple Notification Service e i report sui AWS costi e sull'utilizzo come segue:

- Le policy IAM basate sull'identità con le autorizzazioni a livello di risorsa controllano chi è in grado di eseguire le query in un gruppo di lavoro.
- Athena pubblica le metriche delle query del gruppo di lavoro su CloudWatch, se abiliti le metriche di query.

- In Amazon SNS è possibile creare argomenti Amazon SNS che inviano allarmi a specifici utenti del gruppo di lavoro quando i controlli di utilizzo dei dati per le query in un gruppo di lavoro superano le soglie stabilite.
- Quando a un gruppo di lavoro assegni un tag configurato come tag di allocazione dei costi nella console Fatturazione e gestione costi, i costi associati all'esecuzione delle query in quel gruppo di lavoro vengono visualizzati nei Report costi e utilizzo con tale tag di allocazione dei costi.

Argomenti

- Utilizzo dei gruppi di lavoro per l'esecuzione di query
- Controllo dei costi e monitoraggio delle interrogazioni con metriche ed eventi CloudWatch

Consulta anche il post del blog AWS Big Data Separare le query e gestire i costi utilizzando i gruppi di lavoro di Amazon Athena, che mostra come utilizzare i gruppi di lavoro per separare i carichi di lavoro, controllare l'accesso degli utenti e gestire l'utilizzo e i costi delle query.

Utilizzo dei gruppi di lavoro per l'esecuzione di query

Ti consigliamo di usare i gruppi di lavoro per isolare le query per team, applicazioni o carichi di lavoro differenti. Ad esempio, è possibile creare gruppi di lavoro separati per due diversi team all'interno dell'organizzazione. È anche possibile separare i carichi di lavoro. Si possono ad esempio creare due gruppi di lavoro indipendenti, uno per le applicazioni pianificate automatizzate, come la generazione di report, e un altro per l'utilizzo ad hoc da parte degli analisti. È possibile passare da un gruppo di lavoro a un altro.

Argomenti

- Vantaggi dell'utilizzo dei gruppi di lavoro
- Funzionamento dei gruppi di lavoro
- Configurazione dei gruppi di lavoro
- Policy IAM per l'accesso ai gruppi di lavoro
- Impostazioni del gruppo di lavoro
- Gestione dei gruppi di lavoro
- Utilizzo dei gruppi di lavoro Athena abilitati per Centro identità IAM
- API per gruppi di lavoro Athena
- Risoluzione dei problemi relativi ai gruppi di lavoro

Vantaggi dell'utilizzo dei gruppi di lavoro

I gruppi di lavoro consentono di:

Isolare utenti, team applicazioni o carichi di lavoro in gruppi.	Ogni gruppo ha una propria cronologia delle query e un elenco di query salvate. Per ulteriori informazioni, consulta Funzionamento dei gruppi di lavoro. Per tutte le query nel gruppo di lavoro, puoi decidere di configurare impostazioni del gruppo di lavoro. Includono una posizione Amazon S3 per archiviare i risultati delle query, il proprietario del bucket previsto, la crittografia e il controllo degli oggetti scritti nel bucket dei risultati delle query. Puoi anche applicare le impostazioni del gruppo di lavoro. Per ulteriori informazioni, consulta Impostazioni del gruppo di lavoro.
Applicare vincoli di costo.	Puoi impostare due tipi di vincoli di costo per le query in un gruppo di lavoro: • Limite per query è una soglia per la quantità di dati analizzati per ogni query. Athena annulla le query quando superano la soglia specificata. Il limite si applica a tutte le query in esecuzione all'interno di un gruppo di lavoro. Puoi impostare un solo limite per query e aggiornarlo, se necessario. • Per-workgroup limit (Limite per gruppo di lavoro) è una soglia che puoi impostare per ciascun gruppo di lavoro per la quantità di dati analizzati dalle query nel gruppo di lavoro. Al superamento di una soglia, viene attivato un allarme Amazon SNS che attiva una determinata operazione, ad esempio l'invio di un'e-mail a un utente. Per ogni gruppo di lavoro si possono impostare più limiti a livello di gruppo di lavoro. Per informazioni dettagliate sulle fasi, consulta Impostazione dei limiti di controllo dell'utilizzo dei dati.
Tieni traccia delle metriche relative alle query per tutte le query	Per ogni query eseguita in un gruppo di lavoro, se configuri il gruppo di lavoro per la pubblicazione delle metriche, Athena le pubblica su CloudWatch. È possibile visualizzare i parametri delle query per ogni

dei gruppi di lavoro in. CloudWatch	gruppo di lavoro all'interno della console Athena. In CloudWatch, puoi creare dashboard personalizzate e impostare soglie e allarmi su queste metriche.

Funzionamento dei gruppi di lavoro

I gruppi di lavoro in Athena hanno le seguenti caratteristiche:

- Per impostazione predefinita, ogni account ha un gruppo di lavoro principale e le autorizzazioni predefinite consentono a tutti gli utenti autenticati di accedere a questo gruppo di lavoro. Il gruppo di lavoro principale non può essere eliminato.
- Ogni gruppo di lavoro creato mostra le query salvate e la cronologia delle query solo per le query eseguite al suo interno e non per tutte le query nell'account. In questo modo le tue query sono separate dalle altre presenti nell'account e puoi individuare in modo più efficace le tue query salvate e le query nella cronologia.
- Se un gruppo di lavoro viene disabilitato, non è possibile eseguire query al suo interno finché non viene abilitato. Le query inviate a un gruppo di lavoro disabilitato hanno esito negativo finché non viene nuovamente abilitato.
- Se si dispone delle autorizzazioni, è possibile eliminare un gruppo di lavoro vuoto e un gruppo di lavoro che contiene query salvate. In questo caso, prima di eliminare il gruppo di lavoro, Athena genera un avviso che indica che verranno eliminate le query salvate. Prima di eliminare un gruppo di lavoro a cui altri utenti hanno accesso, è bene accertarsi che i suoi utenti possono accedere ad altri gruppi in cui possano continuare a eseguire query.
- È possibile configurare le impostazioni a livello di gruppo di lavoro e imporne l'utilizzo da parte di tutte le query che vengono eseguite in un gruppo di lavoro. Le impostazioni includono la posizione dei risultati delle query in Amazon S3, il proprietario del bucket previsto, la crittografia e il controllo degli oggetti scritti nel bucket dei risultati della query.

> **Important**
>
> Quando si applicano le impostazioni a livello di gruppo di lavoro, tutte le query eseguite nel gruppo di lavoro utilizzano le impostazioni del gruppo di lavoro. Questo succede anche se le impostazioni lato client sono diverse da quelle del gruppo di lavoro. Per informazioni, consulta Le impostazioni del gruppo di lavoro sostituiscono le impostazioni lato client.

Limitazioni per i gruppi di lavoro

- Puoi creare fino a 1000 gruppi di lavoro per regione nell'account.
- Il gruppo di lavoro principale non può essere eliminato.
- Puoi aprire fino a dieci schede di query all'interno di ciascun gruppo di lavoro. Quando ti sposti tra i gruppi di lavoro, le schede di query rimangono aperte per un massimo di tre gruppi di lavoro.

Configurazione dei gruppi di lavoro

La configurazione dei gruppi di lavoro prevede la loro creazione e la definizione delle autorizzazioni per il loro utilizzo. Prima di tutto, decidi quali gruppi di lavoro sono necessari per la tua organizzazione e creali. Quindi, configura le policy IAM per i gruppi di lavoro che controllano l'accesso e le operazioni degli utenti su una risorsa `workgroup`. Gli utenti con l'accesso a questi gruppi di lavoro ora possono eseguire query al loro interno.

> **Note**
>
> Esegui queste attività per la configurazione dei gruppi di lavoro la prima volta che li utilizzi. Se il tuo account Athena usa già i gruppi di lavoro, ogni utente dell'account necessita delle autorizzazioni per l'esecuzione di query in uno o più gruppi di lavoro nell'account. Prima di eseguire query, controlla la policy IAM per stabilire a quali gruppi di lavoro puoi accedere, modifica la policy se necessario e passa a un gruppo di lavoro che intendi utilizzare.

Per impostazione predefinita, se non è stato creato alcun gruppo di lavoro, tutte le query nel tuo account vengono eseguite nel gruppo di lavoro principale.

Il gruppo di lavoro corrente viene visualizzato nella console Athena nell'opzione Workgroup (Gruppo di lavoro) in alto a destra. Puoi utilizzare questa opzione per passare da un gruppo di lavoro a un altro. Le query vengono eseguite nel gruppo di lavoro corrente. È possibile eseguire query nel contesto di un gruppo di lavoro nella console, attraverso le operazioni API, l'interfaccia a riga di comando o un'applicazione client utilizzando il driver JDBC o ODBC. Quando si ha accesso a un gruppo di lavoro, è possibile visualizzare le impostazioni, i parametri e i limiti di controllo dell'utilizzo dei dati del gruppo di lavoro. È inoltre possibile disporre di autorizzazioni aggiuntive per modificare le impostazioni e i limiti di controllo dell'utilizzo dei dati.

Per impostare i gruppi di lavoro

1. Decidere quali gruppi di lavoro creare. Ad esempio, puoi decidere quanto segue:

   - Chi può eseguire query in ciascun gruppo di lavoro e chi è il proprietario della configurazione del gruppo di lavoro. Questo determina le policy IAM create. Per ulteriori informazioni, consulta Policy IAM per l'accesso ai gruppi di lavoro.
   - Quali posizioni in Amazon S3 utilizzare per i risultati delle query che vengono eseguite in ciascun gruppo di lavoro. Una posizione deve esistere in Amazon S3 prima di poterla specificare per i risultati delle query del gruppo di lavoro. Tutti gli utenti che usano un gruppo di lavoro devono avere accesso a questa posizione. Per ulteriori informazioni, consulta Impostazioni del gruppo di lavoro.
   - Se il proprietario del bucket dei risultati della query Amazon S3 ha il pieno controllo dei nuovi oggetti scritti nel bucket. Ad esempio, se la posizione dei risultati della query è di proprietà di un altro account, puoi concedere la proprietà e il controllo completo dei risultati della query all'altro account. Per ulteriori informazioni, consulta. AclConfiguration
   - Specificate l'ID di Account AWS quello che ritenete sia il proprietario del bucket di posizione di output. Si tratta di una misura di sicurezza aggiuntiva facoltativa. Se l'ID account del proprietario del bucket non corrisponde all'ID specificato, i tentativi di output nel bucket avranno esito negativo. Per ulteriori informazioni, consulta Verifica della proprietà del bucket con condizione del proprietario del bucket nella Guida per l'utente di Amazon S3. Questa impostazione non si applica alle istruzioni CTAS, INSERT INTO o UNLOAD.
   - Quali impostazioni di crittografia sono necessarie e quali gruppi di lavoro hanno query che devono essere crittografate. È consigliabile creare gruppi di lavoro separati per le query crittografate e non crittografate. In questo modo, è possibile applicare la crittografia per un gruppo di lavoro applicabile a tutte le query che vengono eseguite al suo interno. Per ulteriori informazioni, consulta Crittografia dei risultati di query Athena archiviati in Amazon S3.
2. Creare gruppi di lavoro necessari e aggiungervi tag. Per le fasi, consulta Creare un gruppo di lavoro.
3. Creare policy IAM per utenti, gruppi o ruoli per abilitarne l'accesso ai gruppi di lavoro. Le policy definiscono l'appartenenza ai gruppi di lavoro e l'accesso alle operazioni su una risorsa `workgroup`. Per informazioni dettagliate sulle fasi, consulta Policy IAM per l'accesso ai gruppi di lavoro. Per esempi di policy JSON, consulta la sezione Accesso a gruppi di lavoro e tag.
4. Configurare le impostazioni del gruppo di lavoro Specifica un percorso in Amazon S3 per i risultati delle query e, facoltativamente, specifica il proprietario del bucket previsto, le impostazioni di crittografia e il controllo degli oggetti scritti nel bucket dei risultati della query.

Puoi applicare le impostazioni del gruppo di lavoro. Per ulteriori informazioni, consulta Impostazioni del gruppo di lavoro.

> **Important**
>
> Se le impostazioni lato client vengono ignorate, Athena utilizza le impostazioni del gruppo di lavoro. Questo incide sulle query eseguite nella console utilizzando il driver, l'interfaccia a riga di comando o le operazioni API.
> Durante l'esecuzione di query, l'automazione basata sulla disponibilità dei risultati in un determinato bucket Amazon S3 potrebbe interrompersi. È consigliabile informare gli utenti prima di impostare l'override. Quando le impostazioni del gruppo di lavoro sono impostate per l'override, puoi evitare di specificare le impostazioni lato client nei driver o nell'API.

5. Comunicare agli utenti quali gruppi di lavoro usare per l'esecuzione di query. Inviare un'e-mail per comunicare agli utenti dell'account i nomi dei gruppi di lavoro che possono utilizzare, le policy IAM necessarie e le impostazioni dei gruppi di lavoro.
6. Configurare limiti di controllo dei costi, detti anche limiti di controllo dell'utilizzo dei dati, per le query e i gruppi di lavoro. Per ricevere una notifica al superamento di una soglia, crea un argomento Amazon SNS e configura gli abbonamenti. Per i passaggi dettagliati, consulta Impostazione dei limiti di controllo dell'utilizzo dei dati e Nozioni di base su Amazon SNS nella Guida per gli sviluppatori di Amazon Simple Notification Service.
7. Passare al gruppo di lavoro in modo da poter eseguire le query. Per eseguire le query, passare al gruppo di lavoro appropriato. Per informazioni dettagliate sulle fasi, consulta the section called “Specificare un gruppo di lavoro in cui eseguire query”.

Policy IAM per l'accesso ai gruppi di lavoro

Per controllare l'accesso ai gruppi di lavoro, usa le autorizzazioni IAM a livello di risorsa o le policy IAM basate sull'identità. Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

> **Note**
>
> Per accedere a gruppi di lavoro affidabili abilitati alla propagazione delle identità, gli utenti di IAM Identity Center devono essere assegnati a `IdentityCenterApplicationArn` ciò che viene restituito dalla risposta dell'azione dell'API GetWorkGroupAthena.

La seguente procedura è specifica di Athena.

Per informazioni specifiche su IAM, segui i collegamenti elencati alla fine di questa sezione. Per informazioni sulle policy di esempio dei gruppi di lavoro JSON, consulta la sezione Esempi di policy per i gruppi di lavoro.

Per utilizzare l'editor visivo nella console IAM per creare una policy di gruppo di lavoro

1. Accedi AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione sulla sinistra, selezionare Policies (Policy) e fare clic su Create Policy (Crea policy).
3. Nella scheda Visual editor (Editor visivo), selezionare Choose a Service (Scegli un servizio). Quindi selezionare Athena per aggiungerlo alla policy.
4. Scegliere Select actions (Seleziona operazioni), quindi scegliere le operazioni da aggiungere alla policy. L'editor visivo mostra le operazioni disponibili in Athena. Per ulteriori informazioni, consulta Operazioni, risorse e chiavi di condizione per Amazon Athena nella Documentazione di riferimento per l'autorizzazione ai servizi.
5. Scegliere add actions (aggiungi operazioni) per immettere un'operazione oppure usare i caratteri jolly (*) per specificare più operazioni.

   Come impostazione predefinita, la policy che si sta creando utilizza le operazioni selezionate. Se si selezionano una o più operazioni che supportano le autorizzazioni a livello di risorsa per la risorsa `workgroup` in Athena, l'editor elenca la risorsa `workgroup`
6. Selezionare Resources (Risorse) per specificare i gruppi di lavoro per la policy. Per esempi di policy dei gruppi di lavoro JSON, consulta la sezione Esempi di policy per i gruppi di lavoro.
7. Specificare la risorsa `workgroup` come segue:

```
arn:aws:athena:<region>:<user-account>:workgroup/<workgroup-name>
```

8. Scegliere Review policy (Rivedi policy) e digitare i valori per Name (Nome) e Description (Descrizione) (facoltativa) per la policy che si sta creando. Esaminare il riepilogo della policy per accertarsi di disporre delle autorizzazioni desiderate.
9. Seleziona Crea policy per salvare la nuova policy.
10. Allega questa policy basata su un'identità a un utente, un gruppo o un ruolo.

Per ulteriori informazioni, consulta gli argomenti seguenti nella Referenza sull'autorizzazione del servizio e nella Guida per l'utente di IAM:

- Operazioni, risorse e chiavi di condizione per Amazon Athena
- Creazione di policy con l'editor visivo
- Aggiunta e rimozione delle policy IAM
- Controllo dell'accesso alle risorse

Per esempi di policy dei gruppi di lavoro JSON, consulta la sezione Esempi di policy per i gruppi di lavoro.

Per un elenco completo delle operazioni Amazon Athena, consulta i nomi delle operazioni API nella documentazione di riferimento dell'API Amazon Athena.

Esempi di policy per i gruppi di lavoro

Questa sezione include policy di esempio che puoi utilizzare per abilitare varie operazioni sui gruppi di lavoro. Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Un gruppo di lavoro è una risorsa IAM gestita da Athena. Pertanto, se la policy del gruppo di lavoro utilizza operazioni che accettano `workgroup` come input, è necessario specificare l'ARN del gruppo di lavoro nel modo seguente:

```
"Resource": [arn:aws:athena:<region>:<user-account>:workgroup/<workgroup-name>]
```

Dove *<workgroup-name>* è il nome del gruppo di lavoro. Ad esempio, per un gruppo di lavoro denominato `test_workgroup`, deve essere specificato come risorsa nel modo seguente:

```
"Resource": ["arn:aws:athena:us-east-1:123456789012:workgroup/test_workgroup"]
```

Per un elenco completo delle operazioni Amazon Athena, consulta i nomi delle operazioni API nella documentazione di riferimento dell'API Amazon Athena. Per ulteriori informazioni sulle policy IAM, consulta Creazione di policy con l'editor visivo nella Guida per l'utente di IAM. Per ulteriori informazioni sulla creazione di policy IAM per i gruppi di lavoro, consulta Policy IAM per l'accesso ai gruppi di lavoro.

- Example policy for full access to all workgroups
- Example policy for full access to a specified workgroup
- Example policy for running queries in a specified workgroup
- Example policy for running queries in the primary workgroup
- Example policy for management operations on a specified workgroup
- Example policy for listing workgroups
- Example policy for running and stopping queries in a specific workgroup
- Example policy for working with named queries in a specific workgroup
- Example policy for working with Spark notebooks

Example Esempio di policy per l'accesso completo a tutti i gruppi di lavoro

La policy seguente consente l'accesso completo a tutte le risorse gruppo di lavoro che potrebbero essere presenti nell'account. È consigliabile utilizzare questa policy per gli utenti nell'account che devono amministrare e gestire gruppi di lavoro per tutti gli altri utenti.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:*"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Example Esempio di policy per l'accesso completo a un determinato gruppo di lavoro

La policy seguente consente l'accesso completo a una specifica risorsa gruppo di lavoro denominata `workgroupA`. È possibile usare questa policy per gli utenti con controllo completo su un determinato gruppo di lavoro.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:ListEngineVersions",
                "athena:ListWorkGroups",
                "athena:ListDataCatalogs",
                "athena:ListDatabases",
                "athena:GetDatabase",
                "athena:ListTableMetadata",
                "athena:GetTableMetadata"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "athena:BatchGetQueryExecution",
                "athena:GetQueryExecution",
                "athena:ListQueryExecutions",
                "athena:StartQueryExecution",
                "athena:StopQueryExecution",
                "athena:GetQueryResults",
                "athena:GetQueryResultsStream",
                "athena:CreateNamedQuery",
                "athena:GetNamedQuery",
                "athena:BatchGetNamedQuery",
                "athena:ListNamedQueries",
                "athena:DeleteNamedQuery",
                "athena:CreatePreparedStatement",
                "athena:GetPreparedStatement",
                "athena:ListPreparedStatements",
                "athena:UpdatePreparedStatement",
                "athena:DeletePreparedStatement"
            ],
```

```
            "Resource": [
                "arn:aws:athena:us-east-1:123456789012:workgroup/workgroupA"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "athena:DeleteWorkGroup",
                "athena:UpdateWorkGroup",
                "athena:GetWorkGroup",
                "athena:CreateWorkGroup"
            ],
            "Resource": [
                "arn:aws:athena:us-east-1:123456789012:workgroup/workgroupA"
            ]
        }
    ]
}
```

Example Esempio di policy per l'esecuzione di query in un determinato gruppo di lavoro

Nella policy seguente, a un utente è consentito eseguire query nel gruppo `workgroupA` specificato e visualizzarle. All'utente non è consentito eseguire attività di gestione per il gruppo di lavoro, ad esempio l'aggiornamento o l'eliminazione.

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
            "Effect": "Allow",
            "Action": [
                "athena:ListEngineVersions",
                "athena:ListWorkGroups",
                "athena:ListDataCatalogs",
                "athena:ListDatabases",
                "athena:GetDatabase",
                "athena:ListTableMetadata",
                "athena:GetTableMetadata"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
```

```
            "Action": [
                "athena:GetWorkGroup",
                "athena:BatchGetQueryExecution",
                "athena:GetQueryExecution",
                "athena:ListQueryExecutions",
                "athena:StartQueryExecution",
                "athena:StopQueryExecution",
                "athena:GetQueryResults",
                "athena:GetQueryResultsStream",
                "athena:CreateNamedQuery",
                "athena:GetNamedQuery",
                "athena:BatchGetNamedQuery",
                "athena:ListNamedQueries",
                "athena:DeleteNamedQuery",
                "athena:CreatePreparedStatement",
                "athena:GetPreparedStatement",
                "athena:ListPreparedStatements",
                "athena:UpdatePreparedStatement",
                "athena:DeletePreparedStatement"
            ],
            "Resource": [
                "arn:aws:athena:us-east-1:123456789012:workgroup/workgroupA"
            ]
        }
    ]
}
```

Example Esempio di policy per l'esecuzione di query nel gruppo di lavoro principale

È possibile modificare l'esempio precedente per consentire a un determinato utente di eseguire query anche nel gruppo di lavoro principale.

> **Note**
>
> È consigliabile aggiungere la risorsa principale dei gruppi di lavoro a tutti gli utenti, che altrimenti sono configurati per l'esecuzione di query nei rispettivi gruppi di lavoro designati. L'aggiunta di questa risorsa alle policy degli utenti del gruppo di lavoro è utile in caso di eliminazione o disabilitazione dei rispettivi gruppi di lavoro designati. In questo caso, possono continuare a eseguire query nel gruppo di lavoro principale.

Per consentire agli utenti nel tuo account di eseguire query nel gruppo di lavoro principale, aggiungi una riga che contiene l'ARN del gruppo di lavoro principale alla sezione delle risorse di Example policy for running queries in a specified workgroup, come nel seguente esempio.

```
arn:aws:athena:us-east-1:123456789012:workgroup/primary"
```

Example Esempio di policy per le operazioni di gestione in un determinato gruppo di lavoro

Nel policy seguente, un utente può creare, eliminare, ottenere i dettagli e aggiornare un gruppo di lavoro `test_workgroup`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:ListEngineVersions"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "athena:CreateWorkGroup",
                "athena:GetWorkGroup",
                "athena:DeleteWorkGroup",
                "athena:UpdateWorkGroup"
            ],
            "Resource": [
                "arn:aws:athena:us-east-1:123456789012:workgroup/test_workgroup"
            ]
        }
    ]
}
```

Example Esempio di policy per elencare i gruppi di lavoro

La policy seguente consente a tutti gli utenti per elencare tutti i gruppi di lavoro:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:ListWorkGroups"
            ],
            "Resource": "*"
        }
    ]
}
```

Example Esempio di policy per l'esecuzione e l'interruzione di query in un determinato gruppo di lavoro

In questa policy, a un utente è consentito eseguire query nel gruppo di lavoro:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:StartQueryExecution",
                "athena:StopQueryExecution"
            ],
            "Resource": [
                "arn:aws:athena:us-east-1:123456789012:workgroup/test_workgroup"
            ]
        }
    ]
}
```

Example Esempio di policy per l'utilizzo delle query denominate in un determinato gruppo di lavoro

Nel seguente policy, un utente dispone delle autorizzazioni per creare, eliminare e ottenere informazioni sulle query denominate nel gruppo di lavoro specificato:

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

```
            "Effect": "Allow",
            "Action": [
                "athena:CreateNamedQuery",
                "athena:GetNamedQuery",
                "athena:DeleteNamedQuery"
            ],
            "Resource": [
                "arn:aws:athena:us-east-1:123456789012:workgroup/test_workgroup"
     ]
        }
    ]
}
```

Example Esempio di policy per l'utilizzo dei notebook Spark in Athena

Utilizza una policy come la seguente per lavorare con i notebook Spark in Athena.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCreatingWorkGroupWithDefaults",
            "Action": [
                "athena:CreateWorkGroup",
                "s3:CreateBucket",
                "iam:CreateRole",
                "iam:CreatePolicy",
                "iam:AttachRolePolicy",
                "s3:GetBucketLocation",
                "athena:ImportNotebook"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:athena:us-east-1:123456789012:workgroup/Demo*",
                "arn:aws:s3:::123456789012-us-east-1-athena-results-bucket-*",
                "arn:aws:iam::123456789012:role/service-role/
AWSAthenaSparkExecutionRole-*",
                "arn:aws:iam::123456789012:policy/service-role/
AWSAthenaSparkRolePolicy-*"
            ]
        },
        {
            "Sid": "AllowRunningCalculations",
            "Action": [
```

```
                "athena:ListWorkGroups",
                "athena:GetWorkGroup",
                "athena:StartSession",
                "athena:CreateNotebook",
                "athena:ListNotebookMetadata",
                "athena:ListNotebookSessions",
                "athena:GetSessionStatus",
                "athena:GetSession",
                "athena:GetNotebookMetadata",
                "athena:CreatePresignedNotebookUrl"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:athena:us-east-1:123456789012:workgroup/Demo*"
        },
        {
            "Sid": "AllowListWorkGroupAndEngineVersions",
            "Action": [
                "athena:ListWorkGroups",
                "athena:ListEngineVersions"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Impostazioni del gruppo di lavoro

Ogni gruppo di lavoro ha le impostazioni seguenti:

- Un nome univoco. Può contenere da 1 a 128 caratteri, inclusi i caratteri alfanumerici, trattini e caratteri di sottolineatura. Non è possibile modificare il nome di un gruppo di lavoro dopo che è stato creato. È comunque possibile creare un nuovo gruppo di lavoro con le stesse impostazioni e un nome diverso.
- Impostazioni valide per tutte le query eseguite nel gruppo di lavoro. ad esempio:
  - Un percorso in Amazon S3 per archiviare i risultati delle query per tutte le query che vengono eseguite in questo gruppo di lavoro. Questa posizione deve esistere prima di poterla specificare per il gruppo di lavoro durante la creazione. Per informazioni sulla creazione di un bucket Amazon S3, consulta la sezione Creazione di un bucket.
  - Controllo del proprietario del bucket dei risultati delle query: se il proprietario del bucket dei risultati della query Amazon S3 ha il pieno controllo dei nuovi oggetti scritti nel bucket.

Ad esempio, se la posizione dei risultati della query è di proprietà di un altro account, puoi concedere la proprietà e il controllo completo dei risultati della query all'altro account.

- Proprietario previsto del bucket: l'ID del bucket Account AWS che prevedi sia il proprietario del bucket dei risultati della query. Si tratta di una misura di sicurezza aggiuntiva. Se l'ID account del proprietario del bucket non corrisponde all'ID specificato, i tentativi di output nel bucket avranno esito negativo. Per informazioni dettagliate, consulta Verifica della proprietà del bucket con condizione del proprietario del bucket nella Guida per l'utente di Amazon S3.

  > **Note**
  >
  > L'impostazione prevista per il proprietario del bucket si applica solo al percorso di output di Amazon S3 specificato per i risultati delle query di Athena. Non si applica ad altri percorsi Amazon S3 come i percorsi dell'origine dati nei bucket Amazon S3 esterni, i percorsi relativi alle tabelle di destinazione con istruzioni `CTAS` e `INSERT INTO`, i percorsi di output delle istruzioni `UNLOAD`, le operazioni del bucket spill per le query federate o le query `SELECT` eseguite su una tabella in un altro account.

- Un'impostazione di crittografia, se si utilizza la crittografia per tutte le query del gruppo di lavoro. È possibile crittografare solo tutte le query in un gruppo di lavoro, non solo alcune. È preferibile creare gruppi di lavoro separati per contenere query crittografate o non crittografate.

Inoltre, il tuo gruppo di lavoro può ignorare le impostazioni lato client. Prima del rilascio dei gruppi di lavoro, puoi specificare le opzioni per la posizione e la crittografia dei risultati come parametri nel driver JDBC o ODBC oppure nella tab Proprietà della console Athena. Queste impostazioni possono anche essere specificate direttamente tramite operazioni API. Queste impostazioni vengono definite "impostazioni lato client". Con i gruppi di lavoro, puoi configurare queste impostazioni a livello di gruppo di lavoro per controllare le opzioni disponibili a livello di client. L'applicazione delle impostazioni a livello di gruppo di lavoro evita inoltre agli utenti di dover configurare singolarmente le impostazioni lato client. Se per il gruppo di lavoro selezioni l'opzioni Ignora impostazioni lato client, le query utilizzano le impostazioni del gruppo di lavoro e ignorano le impostazioni lato client.

Se si seleziona Override Client-Side Settings (Ignora impostazioni lato client), l'utente riceve una notifica sulla console che segnala la modifica delle impostazioni. Se vengono applicate le impostazioni del gruppo di lavoro in questo modo, gli utenti possono omettere le impostazioni lato client corrispondenti. Quindi, le query eseguite nella console utilizzano le impostazioni del gruppo di lavoro anche se sono presenti impostazioni lato client. Inoltre, quando le interrogazioni nel gruppo di lavoro vengono eseguite tramite API Operations o driver JDBC o ODBC, le impostazioni lato client,

come la posizione e la crittografia dei risultati delle query, vengono sostituite dalle impostazioni del gruppo di lavoro. AWS CLI Per verificare le impostazioni per il gruppo di lavoro, puoi visualizzare i dettagli del gruppo di lavoro.

È anche possibile impostare limiti di query per le query nei gruppi di lavoro.

Le impostazioni del gruppo di lavoro sostituiscono le impostazioni lato client

Nelle finestre di dialogo Create workgroup (Crea gruppo di lavoro) e Edit workgroup (Modifica gruppo di lavoro) è presente un campo denominato Override client-side settings (Ignora impostazioni lato client). Questo campo è deselezionato per impostazione predefinita. Se viene selezionato o lasciato deselezionato, Athena effettua le seguenti operazioni:

- Se l'opzione Ignora impostazioni lato client non è selezionata, le impostazioni del gruppo di lavoro non vengono applicate a livello client. Quando l'opzione Ignora impostazioni lato client non è selezionata per il gruppo di lavoro, Athena utilizza le impostazioni client per tutte le query eseguite nel gruppo di lavoro, incluse la posizione dei risultati delle query, la persona proprietaria del bucket, la crittografia e il controllo degli oggetti scritti nel bucket dei risultati della query. Ogni utente può specificare le impostazioni lato client nel menu Impostazioni nella console. Se le impostazioni lato client non sono impostate, si applicano le impostazioni a livello di gruppo di lavoro. Se si utilizzano le AWS CLI azioni API o i driver JDBC e ODBC per eseguire query in un gruppo di lavoro che non sostituisce le impostazioni lato client, le query utilizzano le impostazioni specificate nelle query.
- Se l'opzione Sovrascrivi le impostazioni lato client è selezionata, le impostazioni del gruppo di lavoro vengono applicate a livello di gruppo di lavoro per tutti i client nel gruppo di lavoro. Quando l'opzione Ignora impostazioni lato client è selezionata per il gruppo di lavoro, Athena utilizza le impostazioni del gruppo di lavoro per tutte le query eseguite nel gruppo di lavoro, incluse la posizione dei risultati delle query, la persona proprietaria del bucket, la crittografia e il controllo degli oggetti scritti nel bucket dei risultati della query. Le impostazioni del gruppo di lavoro hanno la precedenza su qualsiasi impostazione lato client specificata per una query quando si utilizza la console, le azioni API o i driver JDBC o ODBC.

  Se sovrascrivi le impostazioni lato client, alla successiva apertura della console Athena da parte tua o di un altro utente del gruppo di lavoro, verrà visualizzata una finestra in cui si segnala che le query all'interno del gruppo di lavoro usano le relative impostazioni e si chiede di confermare tale modifica.

> **Important**
>
> Se utilizzi le azioni API AWS CLI, i o i driver JDBC e ODBC per eseguire query in un gruppo di lavoro che sovrascrivono le impostazioni lato client, assicurati di omettere le impostazioni lato client nelle query o di aggiornarle in modo che corrispondano alle impostazioni del gruppo di lavoro. Se si specificano impostazioni lato client nelle query ma le si esegue in un gruppo di lavoro che sostituisce le impostazioni, le query verranno eseguite, ma verranno utilizzate le impostazioni del gruppo di lavoro. Per informazioni sulla visualizzazione delle impostazioni per un gruppo di lavoro, consulta Visualizzare i dettagli del gruppo di lavoro.

Gestione dei gruppi di lavoro

All'indirizzo https://console.aws.amazon.com/athena/ puoi possibile eseguire i seguenti processi:

Dichiarazione	Descrizione
Creare un gruppo di lavoro	Creare un nuovo gruppo di lavoro.
Modificare un gruppo di lavoro	Modificare un gruppo di lavoro e le relative impostazioni. Non è possibile modificare il nome di un gruppo di lavoro, ma è possibile creare un nuovo gruppo di lavoro con le stesse impostazioni e un nome diverso.
Visualizzare i dettagli del gruppo di lavoro	Visualizzare i dettagli del gruppo, ad esempio il nome, la descrizione, i limiti di utilizzo dei dati, la posizione dei risultati delle query, il proprieta rio del bucket previsto, la crittografia e il controllo degli oggetti scritti nel bucket dei risultati delle query. È anche possibile verificare se il gruppo di lavoro applica le proprie impostazioni, se è selezionata l'opzione Override client-side settings (Ignora impostazioni lato client).
Eliminare un gruppo di lavoro	Eliminare un gruppo di lavoro. Se si elimina un gruppo di lavoro, vengono eliminati la cronologia delle query, le query salvate, le impostazioni del gruppo di lavoro e i controlli dei limiti dei dati a

Dichiarazione	Descrizione
	livello di query. I controlli dei limiti di dati a livello di gruppo di lavoro rimangono attivi e puoi eliminarli singolarmente. CloudWatch Il gruppo di lavoro principale non può essere eliminato.
Passaggio da un gruppo di lavoro a un altro	Spostarsi tra i gruppi di lavoro a cui si ha accesso.
Copiare una query salvata tra gruppi di lavoro.	Copiare una query salvata tra gruppi di lavoro. È possibile eseguire questa operazione se, ad esempio, è stata creata una query in un gruppo di lavoro in anteprima e si desidera renderla disponibile in un gruppo di lavoro non in anteprima.
Abilitare e disabilitare un gruppo di lavoro	Abilitare o disabilitare un gruppo di lavoro. Quando un gruppo di lavoro è disabilitato, gli utenti non possono eseguire query o creare nuove query denominate. Se si dispone dell'accesso, è comunque possibile visualizzare i parametri, i controlli dei limiti di utilizzo dei dati, le impostazioni del gruppo di lavoro, la cronologia delle query e le query salvate.
Specificare un gruppo di lavoro in cui eseguire query	Prima di eseguire query, è necessario indicare ad Athena quale gruppo di lavoro utilizzare. È necessario disporre delle autorizzazioni per il gruppo di lavoro.
Creazione di un gruppo di lavoro Athena che utilizza l'autenticazione Centro identità IAM	Per utilizzare le identità di Centro identità IAM con Athena, è necessari o creare un gruppo di lavoro abilitato per Centro identità IAM. Dopo aver creato il gruppo di lavoro, è possibile utilizzare la console o l'API di Centro identità IAM per assegnare utenti o gruppi Centro identità IAM al gruppo stesso.

Creare un gruppo di lavoro

Per creare un gruppo di lavoro è necessario disporre delle autorizzazioni per le operazioni API `CreateWorkgroup`. Consulta Accesso a gruppi di lavoro e tag e Policy IAM per l'accesso ai gruppi di lavoro. Se aggiungi tag, devi anche aggiungere autorizzazioni a `TagResource`. Per informazioni, consulta Esempi di policy relative ai tag per gruppi di lavoro.

Per creare un gruppo di lavoro nella console

1. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

2. Nel pannello di navigazione della console Athena, scegli Workgroups (Gruppi di lavoro).
3. Nella pagina Gruppi di lavoro scegli Crea gruppo di lavoro.
4. Nella pagina Create workgroup (Crea gruppo di lavoro) compilare i campi nel modo seguente:

Campo	Descrizione
Workgroup name (Nome del gruppo di lavoro)	Obbligatorio. Immettere un nome univoco per il gruppo di lavoro. Usare un numero di caratteri compreso tra 1 e 128. (A-Z,a-z, 0-9,_,-,.). Questo nome non può essere modificato.
Descrizione	Facoltativo. Immettere una descrizione per il gruppo di lavoro. Può contenere fino a 1024 caratteri.
Choose the type of engine (Scegli tipo di motore)	Scegli Athena SQL se desideri eseguire query SQL ad hoc sui dati in Amazon S3 o utilizza un connettore di origine dati predefinito per eseguire query federate su una varietà di origini dati esterne ad Amazon S3. È possibile eseguire query utilizzando l'editor di query Athena, la AWS CLI o le API Athena. Scegli Apache Spark se desideri creare, modificare ed eseguire applicazioni per notebook Jupyter utilizzando Python e Apache Spark. I notebook Jupyter contengono un elenco di celle che

Campo	Descrizione
	possono includere codice, testo, Markdown, matematica, grafici e rich media. Le celle vengono eseguite secondo l'ordine come calcoli in una sessione interattiva del notebook in Athena. Per ulteriori informazioni sulla creazione e sulla configurazione di un gruppo di lavoro abilitato per Spark, consulta la pagina Creazione di un gruppo di lavoro abilitato a Spark in Athena. Dopo avere creato un gruppo di lavoro, il relativo motore di analisi può essere aggiornato (ad esempio, dalla versione 2 del motore Athena alla versione 3 del motore Athena), ma il tipo di motore non può essere modificato. Ad esempio, un gruppo di lavoro del motore Athena versione 3 non può essere modificato in un gruppo di lavoro PySpark del motore versione 3.
Aggiornamento del motore di query	Scegli come aggiornare il tuo gruppo di lavoro quando viene rilasciata una nuova versione del motore Athena. Puoi lasciare che Athena decida quando eseguire l'aggiornamento o specificare manualmente una versione del motore. Per ulteriori informazioni, consulta Controllo delle versioni del motore di Athena.
Modalità di autentica zione	Scegli AWS Identity and Access Management (IAM) per utilizzare l'autenticazione o la federazione IAM per il gruppo di lavoro. Scegli Centro identità IAM se desideri supportare le identità della forza lavoro, come utenti e gruppi, fornite da provider di identità SAML 2.0, come Microsoft Active Directory. Per ulteriori informazioni, consulta la pagina Trusted identity propagation across applications nella Guida per l'utente di AWS IAM Identity Center .
Ruolo di servizio per l'accesso a Centro identità IAM	Athena richiede le autorizzazioni IAM per accedere a Centro identità IAM per tuo conto. Per ulteriori informazioni sui ruoli di servizio IAM, consulta la pagina Creating a role to delegate permissions to an AWS service nella Guida per l'utente di IAM.

Campo	Descrizione
Percorso dei risultati delle query	Facoltativo. Inserire il percorso di un bucket o prefisso Amazon S3. Questo bucket e prefisso devono esistere prima di specificarli. **Note** Se si eseguono query nella console, è facoltativo specifica re la posizione dei risultati delle query. Se non viene specificata per il gruppo di lavoro o in Settings (Impostaz ioni), Athena usa la posizione predefinita per i risultati delle query. Se esegui le query con l'API o i driver, devi specifica re la posizione dei risultati delle query in almeno una delle due posizioni: per le singole query con o per il gruppo di lavoro OutputLocation, con. WorkGroupConfiguration
Proprietario previsto del bucket	Facoltativo. Inserisci l'ID del bucket di posizione di output Account AWS che ti aspetti sia il proprietario. Si tratta di una misura di sicurezza aggiuntiva. Se l'ID account del proprietario del bucket non corrisponde all'ID specificato, i tentativi di output nel bucket avranno esito negativo. Per informazioni dettagliate, consulta Verifica della proprietà del bucket con condizione del proprietario del bucket nella Guida per l'utente di Amazon S3. **Note** L'impostazione prevista per il proprietario del bucket si applica solo al percorso di output di Amazon S3 specifica to per i risultati delle query di Athena. Non si applica ad altri percorsi Amazon S3 come i percorsi dell'origine dati nei bucket Amazon S3 esterni, i percorsi relativi alle tabelle di destinazione con istruzioni `CTAS` e `INSERT INTO`, i percorsi di output delle istruzioni `UNLOAD`, le operazioni del bucket spill per le query federate o le query `SELECT` eseguite su una tabella in un altro account.

Campo	Descrizione
Assegna al proprietario del bucket il controllo completo ai risultati delle query	Questo campo è deselezionato per impostazione predefinita. Se lo selezioni e le ACL sono abilitate per il bucket della posizione dei risultati della query, concedi il controllo di accesso completo ai risultati della query al proprietario del bucket. Ad esempio, se la posizione dei risultati della query è di proprietà di un altro account, puoi utilizzare questa opzione per concedere la proprietà e il controllo completo dei risultati della query all'altro account. Se l'impostazione di S3 Object Ownership del bucket è Bucket owner preferred (Preferita dal proprietario del bucket), il proprieta rio del bucket possiede anche tutti gli oggetti dei risultati della query scritti da questo gruppo di lavoro. Ad esempio, se il gruppo di lavoro di un account esterno abilita questa opzione e imposta la posizione dei risultati della query sul bucket Amazon S3 del tuo account con S3 Object Ownership impostato su Bucket owner preferred (Preferit a dal proprietario del bucket), possiedi e hai il pieno controllo di accesso ai risultati delle query del gruppo di lavoro esterno. Selezionando questa opzione, quando l'impostazione di S3 Object Ownership del bucket dei risultati della query è Bucket owner enforced (Applicata dal proprietario del bucket), non si avrà alcun effetto. Per ulteriori informazioni, consulta Object Ownership settings (Impostazioni di Object Ownership) nella Guida per l'utente di Amazon S3.
Encrypt query results (Esegui la crittografia dei risultati delle query)	Facoltativo. I risultati archiviati in Amazon S3 vengono crittogra fati. Se è selezionato, tutte le query nel gruppo di lavoro vengono crittografate. Se è selezionato, è possibile selezionare Encryption type (Tipo di crittografia), Encryption key (Chiave di crittografia) e immettere KMS Key ARN (ARN chiave KMS). Se non si dispone della chiave, aprire la console AWS KMS per crearla. Per ulteriori informazioni, consulta Creazione di chiavi nella Guida per gli sviluppatori di AWS Key Management Service .

Campo	Descrizione
Imposta ***encryption_type*** come crittografia minima	Facoltativo. Selezionate questa opzione per applicare un tipo minimo di crittografia per i risultati delle query per tutti gli utenti del gruppo di lavoro. Selezionando questa opzione viene visualizzata una tabella con la gerarchia dei tipi di crittografia. La tabella mostra anche quali tipi di crittografia saranno autorizzati a utilizzare gli utenti del gruppo di lavoro quando si specifica un particolare tipo di crittografia come minimo. Per utilizzare questa opzione, Ignora impostazioni lato client non deve essere selezionata. Per ulteriori informazioni, consulta Configurazione della crittografia minima per un gruppo di lavoro.
Abilitazione di S3 Access Grants	Questo campo è selezionato per impostazione predefinita quando si sceglie Centro identità IAM come modalità di autenticazione. Se selezionata, questa opzione applica le autorizzazioni basate su utenti o gruppi di Centro identità IAM alle posizioni Amazon S3.
Creazione di un prefisso S3 basato sull'identità utente	Quando questa opzione è selezionata, Athena crea un prefisso Amazon S3 quando memorizza i risultati delle query. Il prefisso si basa sull'identità utente Centro identità IAM dell'utente.
Pubblica le metriche delle query su CloudWatch	Questo campo è selezionato per impostazione predefinita. Pubblicare i parametri di query in CloudWatch. Per informazioni, consulta Monitoraggio delle query Athena con metriche CloudWatch .
Override client-side settings (Ignora impostazioni lato client)	Questo campo è deselezionato per impostazione predefinita. Se viene selezionato, le impostazioni del gruppo di lavoro si applicano a tutte le query nel gruppo di lavoro e sostituiscono le impostazioni lato client. Per ulteriori informazioni, consulta Le impostazioni del gruppo di lavoro sostituiscono le impostazioni lato client.

Campo	Descrizione
Requester Pays S3 buckets (Bucket S3 con Pagamento a carico del richiedente)	Facoltativo. Scegli Turn on queries on requester pays buckets in Amazon S3 (Attiva query su bucket con pagamento a carico del richiedente in Amazon S3) se gli utenti del gruppo di lavoro eseguiranno le query sui dati archiviati nei bucket Amazon S3 configurati come Requester Pays (Pagamento a carico del richiedente). All'account dell'utente che esegue la query vengono addebitati i costi di accesso e trasferimento dei dati applicabili associati alla query. Per ulteriori informazioni, consulta Bucket con pagamento a carico del richiedente nella Guida per l'utente di Amazon Simple Storage Service.
Gestione del controllo dell'utilizzo dei dati a livello di query	Facoltativo. Imposta il limite per la quantità massima di dati che una query può analizzare. È possibile impostare un solo limite per query per un gruppo di lavoro. Il limite si applica a tutte le query nel gruppo di lavoro. Se una query supera tale limite, verrà annullata. Per ulteriori informazioni, consulta Impostazione dei limiti di controllo dell'utilizzo dei dati.
Avvisi sull'utilizzo dei dati a livello di gruppo di lavoro	Facoltativo. Impostare più soglie di avviso quando le query in esecuzione in questo gruppo di lavoro analizzano una determina ta quantità di dati entro un periodo specifico. Gli avvisi vengono implementati utilizzando Amazon CloudWatch alarms e si applicano a tutte le query del gruppo di lavoro. Per ulteriori informazioni, consulta Using Amazon CloudWatch alarms nella Amazon CloudWatch User Guide.
Tag	Facoltativo. Aggiungere uno o più tag a un gruppo di lavoro. Un tag è un'etichetta che viene assegnata a una risorsa gruppo di lavoro in Athena. È formato da una chiave e da un valore. Utilizza le migliori pratiche di AWS etichettatura per creare un set coerente di tag e classificare i gruppi di lavoro per scopo, proprietario o ambiente. È anche possibile utilizzare i tag nelle policy IAM e per controlla re i costi di fatturazione. Non utilizzare chiavi di tag duplicate in uno stesso gruppo di lavoro. Per ulteriori informazioni, consulta the section called “Assegnazione di tag alle risorse”.

5. Selezionare Create workgroup (Crea gruppo di lavoro). Il gruppo di lavoro viene visualizzato nell'elenco nella pagina Workgroups (Gruppi di lavoro).

Puoi anche utilizzare l'operazione CreateWorkGroupAPI per creare un gruppo di lavoro.

> **Important**
>
> Dopo aver creato i gruppi di lavoro, crea Policy IAM per l'accesso ai gruppi di lavoro IAM che consentano di eseguire operazioni correlate al gruppo di lavoro.

Modificare un gruppo di lavoro

Per modificare un gruppo di lavoro è necessario disporre delle autorizzazioni per le operazioni API `UpdateWorkgroup`. Consulta Accesso a gruppi di lavoro e tag e Policy IAM per l'accesso ai gruppi di lavoro. Se aggiungi o modifichi tag, devi anche disporre delle autorizzazioni per `TagResource`. Per informazioni, consulta Esempi di policy relative ai tag per gruppi di lavoro.

Per modificare un gruppo di lavoro nella console

1. Nel pannello di navigazione della console Athena, scegli Workgroups (Gruppi di lavoro).
2. Nella pagina Workgroups (Gruppi di lavoro), seleziona il gruppo di lavoro da modificare.
3. Scegli Actions (Operazioni), Edit (Modifica).
4. Modifica i campi come desiderato. Per l'elenco dei campi, consulta la sezione relativa alla creazione di un gruppo di lavoro. È possibile modificare tutti i campi tranne il nome del gruppo di lavoro. Se è necessario cambiare il nome, creare un altro gruppo di lavoro con il nuovo nome e le stesse impostazioni.
5. Seleziona Salvataggio delle modifiche. Il gruppo di lavoro aggiornato viene visualizzato nell'elenco nella pagina Workgroups (Gruppi di lavoro).

Visualizzare i dettagli del gruppo di lavoro

Per ogni gruppo di lavoro, è possibile visualizzare i relativi dettagli. I dettagli includono il nome, la descrizione, se è abilitato o disabilitato e le impostazioni utilizzate per le query eseguite nel gruppo di lavoro, che includono la posizione dei risultati delle query, il proprietario del bucket previsto, la crittografia e il controllo degli oggetti scritti nel bucket dei risultati delle query. Se un gruppo di lavoro ha limiti di utilizzo dei dati, vengono visualizzati anche tali limiti.

Per visualizzare i dettagli del gruppo di lavoro

1. Nel pannello di navigazione della console Athena, scegli Workgroups (Gruppi di lavoro).
2. Nella pagina Workgroups (Gruppi di lavoro), scegli il collegamento del gruppo di lavoro che desideri visualizzare. Viene visualizzata la pagina Overview Details (Dettagli della panoramica) del gruppo di lavoro.

Eliminare un gruppo di lavoro

Per eliminare un gruppo di lavoro è necessario disporre delle apposite autorizzazioni. Il gruppo di lavoro principale non può essere eliminato.

Se disponi delle autorizzazioni, puoi eliminare un gruppo di lavoro vuoto in qualsiasi momento. Puoi anche eliminare un gruppo di lavoro che contiene query salvate. In questo caso, prima di procedere con l'eliminazione del gruppo di lavoro, Athena presenta un avviso che indica che verranno eliminate le query salvate.

Se elimini un gruppo di lavoro di cui fai parte, lo stato attivo passa al gruppo di lavoro principale nella console. Se hai accesso a tale gruppo, puoi eseguire query e visualizzare le relative impostazioni.

Se elimini un gruppo di lavoro, le sue impostazioni e i controlli dei limiti dei dati a livello di query vengono eliminati. I controlli dei limiti di dati a livello di gruppo di lavoro rimangono attivi e CloudWatch, se necessario, è possibile eliminarli da lì.

> **Important**
>
> Prima di eliminare un gruppo di lavoro, verifica che i suoi utenti facciano parte anche di altri gruppi di lavoro in cui possano continuare a eseguire query. Se le policy IAM degli utenti consentono l'esecuzione di query solo in questo gruppo di lavoro e lo elimini, non saranno più autorizzati a eseguire query. Per ulteriori informazioni, consulta Example policy for running queries in the primary workgroup.

Per eliminare un gruppo di lavoro nella console

1. Nel pannello di navigazione della console Athena, scegli Workgroups (Gruppi di lavoro).
2. Nella pagina Workgroups (Gruppi di lavoro), seleziona il gruppo di lavoro da eliminare.
3. Scegli Operazioni > Elimina.

4. Alla richiesta di conferma Delete workgroup (Elimina gruppo di lavoro), inserisci il nome del gruppo di lavoro, quindi scegli Delete (Elimina).

Per eliminare un gruppo di lavoro con l'operazione API, utilizzare l'operazione `DeleteWorkGroup`.

Passaggio da un gruppo di lavoro a un altro

Per passare da un gruppo di lavoro a un altro è necessario disporre delle autorizzazioni per entrambi.

Puoi aprire fino a dieci schede di query all'interno di ciascun gruppo di lavoro. Quando ti sposti tra i gruppi di lavoro, le schede di query rimangono aperte per un massimo di tre gruppi di lavoro.

Per passare da un gruppo di lavoro a un altro

1. Nella console Athena, utilizza l'opzione Workgroup (Gruppo di lavoro) per scegliere un gruppo di lavoro.
2. Se viene visualizzata la finestra di dialogo Workgroup ***`workgroup-name`*** settings (Impostazioni del gruppo di lavoro nome-gruppo-di-lavoro), scegli Acknowledge (Accetta).

L'opzione Workgroup (Gruppo di lavoro) mostra il nome del gruppo di lavoro a cui si è passati. Ora è possibile eseguire query in questo gruppo di lavoro.

Copiare una query salvata tra gruppi di lavoro.

Attualmente, la console Athena non ha la possibilità di copiare direttamente una query salvata da un gruppo di lavoro a un altro, ma è possibile eseguire la stessa operazione manualmente utilizzando la procedura riportata di seguito.

Per copiare una query salvata tra gruppi di lavoro

1. Nella console Athena, dal gruppo di lavoro da cui copiare la query, scegli la tab Query salvate.
2. Scegli il collegamento della query salvata da copiare. Athena apre la query nell'editor di query.
3. Nell'editor di query seleziona il testo della query, quindi premi **`Ctrl+C`** per copiarlo.
4. Passa al gruppo di lavoro di destinazione oppure creare un gruppo di lavoro, quindi passa ad esso.
5. Apri una nuova scheda nell'editor di query e quindi premi **`Ctrl+V`** per incollare il testo nella nuova scheda.

6. Nell'editor di query, scegli Salva con nome per salvare la query nel gruppo di lavoro di destinazione.
7. Nella finestra di dialogo Scegli un nome inserisci un nome per la query e una descrizione facoltativa.
8. Selezionare Salva.

Abilitare e disabilitare un gruppo di lavoro

Se disponi delle autorizzazioni appropriate, puoi abilitare o disabilitare i gruppi di lavoro nella console, utilizzando le operazioni API o con i driver JDBC e ODBC.

Per abilitare o disabilitare un gruppo di lavoro

1. Nel pannello di navigazione della console Athena, scegli Workgroups (Gruppi di lavoro).
2. Nella pagina Workgroups (Gruppi di lavoro), scegli il collegamento per il gruppo di lavoro.
3. In alto a destra, seleziona Enable workgroup (Abilita gruppo di lavoro) o Disable workgroup (Disabilita gruppo di lavoro).
4. Alla richiesta di conferma, scegli Enable (Abilita) o Disable (Disabilita). Se si disabilita un gruppo di lavoro, gli utenti non possono eseguire query al suo interno o creare nuove query denominate. Se si abilita un gruppo di lavoro, gli utenti possono utilizzarlo per l'esecuzione di query.

Specificare un gruppo di lavoro in cui eseguire query

Per specificare un gruppo di lavoro da utilizzare, è necessario disporre delle relative autorizzazioni.

Per specificare il gruppo di lavoro da utilizzare

1. Verificare che le proprie autorizzazioni consentano di eseguire query nel gruppo di lavoro che si prevede di utilizzare. Per ulteriori informazioni, consulta the section called “ Policy IAM per l'accesso ai gruppi di lavoro”.
2. Per specificare il gruppo di lavoro, utilizzare una di queste opzioni:
   - Se si utilizza la console Athena, impostare il gruppo di lavoro passando da un gruppo di lavoro a un altro.
   - Se utilizzano le operazioni dell'API Athena, specificare il nome del gruppo di lavoro nell'operazione API. Ad esempio, è possibile impostare il nome del gruppo di lavoro nel StartQueryExecutionmodo seguente:

```
StartQueryExecutionRequest startQueryExecutionRequest = new
 StartQueryExecutionRequest()
              .withQueryString(ExampleConstants.ATHENA_SAMPLE_QUERY)
              .withQueryExecutionContext(queryExecutionContext)
              .withWorkGroup(WorkgroupName)
```

- Se si utilizza il driver JDBC o ODBC, impostare il nome del gruppo di lavoro nella stringa di connessione utilizzando il parametro di configurazione `Workgroup`. Il driver passa il nome del gruppo di lavoro ad Athena. Specificare il parametro del gruppo di lavoro nella stringa di connessione come nell'esempio seguente:

```
jdbc:awsathena://AwsRegion=<AWSREGION>;UID=<ACCESSKEY>;
PWD=<SECRETKEY>;S3OutputLocation=s3://DOC-EXAMPLE-BUCKET/<athena-
output>-<AWSREGION>/;
Workgroup=<WORKGROUPNAME>;
```

Configurazione della crittografia minima per un gruppo di lavoro

In qualità di amministratore di un gruppo di lavoro Athena SQL, puoi applicare un livello minimo di crittografia in Amazon S3 per tutti i risultati delle query del gruppo di lavoro. Puoi usare questa funzionalità per assicurarti che i risultati delle query non vengano mai memorizzati in un bucket Amazon S3 in uno stato non crittografato.

Quando gli utenti di un gruppo di lavoro con crittografia minima abilitata inviano una query, possono impostare la crittografia solo sul livello minimo configurato o su un livello superiore se disponibile. Athena crittografa i risultati delle query al livello specificato quando l'utente esegue la query o al livello impostato nel gruppo di lavoro.

Sono disponibili i seguenti livelli:

- Base: crittografia lato server Amazon S3 con chiavi gestite da Amazon S3 (SSE_S3).
- Intermedia: crittografia lato server con chiavi gestite da KMS (SSE_KMS).
- Avanzata: crittografia lato client con chiavi gestite da KMS (CSE_KMS).

Considerazioni e limitazioni

- La funzionalità di crittografia minima non è disponibile per i gruppi di lavoro abilitati su Apache Spark.

- La funzionalità di crittografia minima è funzionale solo quando il gruppo di lavoro non abilita l'opzione Ignora impostazioni lato client.
- Se il gruppo di lavoro ha l'opzione Ignora impostazioni lato client abilitata, prevale l'impostazione di crittografia del gruppo di lavoro e l'impostazione di crittografia minima non ha alcun effetto.
- L'attivazione di questa funzionalità non comporta alcun costo.

Abilitazione della crittografia minima per un gruppo di lavoro

È possibile abilitare un livello minimo di crittografia per i risultati delle query dal gruppo di lavoro Athena SQL quando si crea o si aggiorna il gruppo di lavoro. A tale scopo, puoi utilizzare la console Athena, l'API Athena o. AWS CLI

Utilizzo della console Athena per abilitare la crittografia minima

Per iniziare a creare o modificare il tuo gruppo di lavoro con la console Athena, consulta Creazione di un gruppo di lavoro o Modifica di un gruppo di lavoro. Durante la configurazione del gruppo di lavoro, utilizza i seguenti passaggi per abilitare la crittografia minima.

Per configurare il livello minimo di crittografia per i risultati delle query del gruppo di lavoro

1. Nella sezione Configurazioni aggiuntive, espandi Impostazioni.
2. Deseleziona l'opzione Ignora le impostazioni lato client o verifica che non sia selezionata.
3. Nella sezione Configurazioni aggiuntive, espandi Configurazione dei risultati delle query.
4. Seleziona l'opzione Crittografa risultati query.
5. Per Tipo di crittografia, seleziona il metodo di crittografia che vuoi che Athena utilizzi per i risultati delle query del gruppo di lavoro (SSE_S3, SSE_KMS oppure CSE_KMS). Questi tipi di crittografia corrispondono ai livelli di sicurezza di base, intermedi e avanzati.
6. Per applicare il metodo di crittografia che hai scelto come livello minimo di crittografia per tutti gli utenti, seleziona Imposta ***`encryption_method`*** come crittografia minima.

   Quando selezioni questa opzione, una tabella mostra la gerarchia di crittografia e i livelli di crittografia consentiti agli utenti quando il tipo di crittografia scelto diventa il minimo.
7. Dopo aver creato il gruppo di lavoro o aggiornato la configurazione del gruppo di lavoro, scegli Crea gruppo di lavoro o Salva modifiche.

Utilizzando l'API Athena o AWS CLI per abilitare la crittografia minima

Quando utilizzi l'UpdateWorkGroupAPI CreateWorkGroupor per creare o aggiornare un gruppo di lavoro Athena SQL, imposta su EnforceWorkGroupConfigurationfalse, su e usa EnableMinimumEncryptionConfigurationEncryptionOptionper true specificare il tipo di crittografia.

In AWS CLI, utilizza il update-work-groupcomando create-work-groupor con i --configuration-updates parametri --configuration or e specifica le opzioni corrispondenti a quelle per l'API.

Utilizzo dei gruppi di lavoro Athena abilitati per Centro identità IAM

La funzionalità di propagazione affidabile delle identità di AWS IAM Identity Center consente di utilizzare le identità della forza lavoro tra i servizi di analisi. AWS La propagazione delle identità attendibili evita di dover eseguire configurazioni di provider di identità specifici del servizio o configurazioni di ruoli IAM.

Con Centro identità IAM, è possibile gestire la sicurezza degli accessi per le identità della forza lavoro, note anche come utenti della forza lavoro. IAM Identity Center offre un posto in cui è possibile creare o connettere gli utenti della forza lavoro e gestire centralmente il loro accesso su tutti gli account e le applicazioni. AWS Per assegnare a questi utenti l'accesso agli Account AWS, puoi utilizzare le autorizzazioni multi-account. Per assegnare agli utenti l'accesso alle applicazioni abilitate a Centro identità IAM, alle applicazioni cloud e alle applicazioni Security Assertion Markup Language (SAML 2.0) dei clienti, puoi utilizzare le assegnazioni delle applicazioni. Per ulteriori informazioni, consulta la pagina Trusted identity propagation across applications nella Guida per l'utente di AWS IAM Identity Center .

Attualmente, il supporto Athena SQL per la propagazione delle identità attendibili consente di utilizzare la stessa identità per Amazon EMR Studio e l'interfaccia Athena SQL in EMR Studio. Per utilizzare le identità di Centro identità IAM con Athena SQL in EMR Studio, è necessario creare gruppi di lavoro abilitati per Centro identità IAM in Athena. Dopodiché, è possibile utilizzare la console o l'API di Centro identità IAM per assegnare utenti o gruppi di Centro identità IAM ai gruppi di lavoro Athena abilitati per Centro identità IAM. Le query provenienti da un gruppo di lavoro Athena che utilizza la propagazione delle identità attendibili devono essere eseguite dall'interfaccia Athena SQL in uno studio EMR abilitato per Centro identità IAM.

Considerazioni e limitazioni

Quando utilizzi la propagazione delle identità attendibili con Amazon Athena, considera i seguenti punti:

- Non è possibile modificare il metodo di autenticazione per il gruppo di lavoro dopo la creazione dello stesso.
  - I gruppi di lavoro SQL Athena esistenti non possono essere modificati per supportare i gruppi di lavoro abilitati per Centro identità IAM.
  - I gruppi di lavoro abilitati per Centro identità IAM non possono essere modificati per supportare le autorizzazioni IAM a livello di risorsa o le policy IAM basate sull'identità.
- Per accedere a gruppi di lavoro affidabili abilitati alla propagazione delle identità, gli utenti di IAM Identity Center devono essere assegnati a `IdentityCenterApplicationArn` ciò che viene restituito dalla risposta dell'azione dell'API GetWorkGroupAthena.
- Amazon S3 Access Grants deve essere configurato per l'utilizzo delle identità con propagazione delle identità attendibili. Per maggiori informazioni, consulta la pagina S3 Access Grants and corporate directory identities nella Guida per l'utente di Amazon S3.
- I gruppi di lavoro Athena abilitati per Centro identità IAM richiedono che Lake Formation sia configurato per l'utilizzo delle identità di Centro identità IAM. Per informazioni sulla configurazione, consulta la pagina Integrating IAM Identity Center nella Guida per gli sviluppatori di AWS Lake Formation .
- Per impostazione predefinita, nei gruppi di lavoro che utilizzano la propagazione delle identità attendibili le query scadono dopo 30 minuti. È possibile richiedere un aumento del timeout delle query, ma nei gruppi di lavoro con propagazione delle identità attendibili le query possono essere eseguite al massimo entro un'ora.
- Le modifiche alle autorizzazioni di utenti o gruppi nei gruppi di lavoro con propagazione delle identità attendibili possono richiedere fino a un'ora per avere effetto.
- Le query in un gruppo di lavoro Athena che utilizza la propagazione delle identità attendibili non possono essere eseguite direttamente dalla console Athena. Devono essere eseguite dall'interfaccia Athena in uno studio EMR con Centro identità IAM abilitato. Per ulteriori informazioni sull'utilizzo di Athena in EMR Studio, consulta la pagina Use the Amazon Athena SQL editor in EMR Studio nella Guida per la gestione di Amazon EMR.
- La propagazione delle identità attendibili non è compatibile con le seguenti funzionalità di Athena.
  - Chiavi di contesto `aws:CalledVia`.
  - Athena per i gruppi di lavoro Spark.
  - Accesso federato all'API di Athena.
  - Accesso federato ad Athena utilizzando Lake Formation e i driver Athena JDBC e ODBC.
- Puoi utilizzare la propagazione delle identità affidabili con Athena solo nei seguenti casi: Regioni AWS

- `us-east-2`: Stati Uniti orientali (Ohio)
- `us-east-1`: Stati Uniti orientali (Virginia settentrionale)
- `us-west-1`: Stati Uniti occidentali (California settentrionale)
- `us-west-2`: Stati Uniti occidentali (Oregon)
- `af-south-1`: Africa (Città del Capo)
- `ap-east-1`: Asia Pacifico (Hong Kong)
- `ap-southeast-3`: Asia Pacifico (Giacarta)
- `ap-south-1`: Asia Pacifico (Mumbai)
- `ap-northeast-3`: Asia Pacifico (Osaka-Locale)
- `ap-northeast-2`: Asia Pacifico (Seoul)
- `ap-southeast-1`: Asia Pacifico (Singapore)
- `ap-southeast-2`: Asia Pacifico (Sydney)
- `ap-northeast-1`: Asia Pacifico (Tokyo)
- `ca-central-1`: Canada (Centrale)
- `eu-central-1`: Europa (Francoforte)
- `eu-west-1`: Europa (Irlanda)
- `eu-west-2`: Europa (Londra)
- `eu-south-1`: Europa (Milano)
- `eu-west-3`: Europa (Parigi)
- `eu-north-1`: Europa (Stoccolma)
- `me-south-1`: Medio Oriente (Bahrein)
- `sa-east-1`: Sud America (San Paolo)

Autorizzazioni richieste

L'utente IAM dell'amministratore che crea il gruppo di lavoro abilitato per Centro identità IAM nella console Athena deve avere le seguenti policy collegate.

- La policy gestita `AmazonAthenaFullAccess`. Per informazioni dettagliate, vedi AWS politica gestita: AmazonAthenaFullAccess.
- La seguente policy inline che consente le operazioni di IAM e Centro identità IAM:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:createRole",
                "iam:CreatePolicy",
                "iam:AttachRolePolicy",
                "iam:ListRoles",
                "iam:PassRole",
                "identitystore:ListUsers",
                "identitystore:ListGroups",
                "identitystore:CreateUser",
                "identitystore:CreateGroup",
                "sso:ListInstances",
                "sso:CreateInstance",
                "sso:DeleteInstance",
                "sso:DescribeUser",
                "sso:DescribeGroup",
                "sso:ListTrustedTokenIssuers",
                "sso:DescribeTrustedTokenIssuer",
                "sso:ListApplicationAssignments",
                "sso:DescribeRegisteredRegions",
                "sso:GetManagedApplicationInstance",
                "sso:GetSharedSsoConfiguration",
                "sso:PutApplicationAssignmentConfiguration",
                "sso:CreateApplication",
                "sso:DeleteApplication",
                "sso:PutApplicationGrant",
                "sso:PutApplicationAuthenticationMethod",
                "sso:PutApplicationAccessScope",
                "sso:ListDirectoryAssociations",
                "sso:CreateApplicationAssignment",
                "sso:DeleteApplicationAssignment",
                "organizations:ListDelegatedAdministrators",
                "organizations:DescribeAccount",
                "organizations:DescribeOrganization",
                "organizations:CreateOrganization",
                "sso-directory:SearchUsers",
                "sso-directory:SearchGroups",
                "sso-directory:CreateUser"
            ],
            "Effect": "Allow",
```

```
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Creazione di un gruppo di lavoro Athena abilitato per Centro identità IAM

La procedura seguente mostra i passaggi e le opzioni relative alla creazione di un gruppo di lavoro Athena abilitato per Centro identità IAM. Per una descrizione delle altre opzioni di configurazione disponibili per i gruppi di lavoro Athena, consulta la pagina Creare un gruppo di lavoro.

Creazione di un gruppo di lavoro abilitato per il SSO nella console Athena

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Nel pannello di navigazione della console Athena, scegli Workgroups (Gruppi di lavoro).
3. Nella pagina Gruppi di lavoro scegli Crea gruppo di lavoro.
4. Nella pagina Crea gruppo di lavoro, in Nome gruppo di lavoro, inserisci un nome per il gruppo di lavoro.
5. Per Motore di analisi, utilizza l'impostazione predefinita Athena SQL.
6. Per Autenticazione, scegli Centro identità IAM.
7. Per Ruolo di servizio per l'accesso a Centro identità IAM, scegli un ruolo di servizio esistente o creane uno nuovo.

   Athena richiede le autorizzazioni per accedere a Centro identità IAM per tuo conto. A tale scopo, Athena ha bisogno di un ruolo di servizio. Un ruolo di servizio è un ruolo IAM che gestisci e che autorizza un AWS servizio ad accedere ad altri AWS servizi per tuo conto. Per ulteriori informazioni, consulta Creazione di un ruolo per delegare le autorizzazioni a un AWS servizio nella Guida per l'utente IAM.
8. Espandi Configurazione dei risultati delle query, quindi inserisci o scegli un percorso Amazon S3 per Posizione del risultato delle query.
9. (Facoltativo) Seleziona Crittografa i risultati delle query.
10. (Facoltativo) Seleziona Crea prefisso S3 basato sull'identità utente.

    Quando crei un gruppo di lavoro abilitato per Centro identità IAM, l'opzione Abilita S3 Access Grants è selezionata per impostazione predefinita. Puoi utilizzare Amazon S3 Access Grants per

controllare l'accesso alle posizioni (prefissi) dei risultati delle query Athena in Amazon S3. Per maggiori informazioni su Amazon S3 Access Grants, consulta la pagina Managing access with Amazon S3 Access Grants.

Nei gruppi di lavoro Athena che utilizzano l'autenticazione Centro identità IAM, puoi abilitare la creazione di posizioni dei risultati delle query basate sull'identità che sono regolate da Amazon S3 Access Grants. Questi prefissi Amazon S3 basati sull'identità utente consentono agli utenti di un gruppo di lavoro Athena di mantenere i risultati delle query isolati dagli altri utenti del medesimo gruppo di lavoro.

Quando abiliti l'opzione del prefisso utente, Athena aggiunge l'ID utente come prefisso del percorso Amazon S3 alla posizione di output dei risultati della query per il gruppo di lavoro (ad esempio, `s3://DOC-EXAMPLE-BUCKET/${user_id}`). Per utilizzare questa funzionalità, devi configurare Access Grants in modo da concedere soltanto all'utente l'autorizzazione alla posizione che ha il prefisso `user_id`. Per un esempio di politica del ruolo di localizzazione di Amazon S3 Access Grants che limita l'accesso ai risultati delle query Athena, consulta. Esempio di politica relativa ai ruoli

> **Note**
>
> La selezione dell'opzione del prefisso S3 dell'identità utente abilita automaticamente l'opzione di sovrascrittura delle impostazioni lato client per il gruppo di lavoro, come descritto nel passaggio successivo. L'opzione di sovrascrittura delle impostazioni lato client è un requisito per la funzionalità del prefisso dell'identità utente.

11. Espandi Impostazioni, quindi verifica che sia selezionata l'opzione Sovrascrivi le impostazioni lato client.

    Quando si seleziona Sovrascrivi le impostazioni lato client, le impostazioni del gruppo di lavoro vengono applicate a livello di gruppo di lavoro per tutti i client nel gruppo stesso. Per ulteriori informazioni, consulta Le impostazioni del gruppo di lavoro sostituiscono le impostazioni lato client.
12. (Facoltativo) Effettua tutte le altre impostazioni di configurazione necessarie come descritto in Creare un gruppo di lavoro.
13. Selezionare Create workgroup (Crea gruppo di lavoro).
14. Utilizza la sezione Workgroups della console Athena per assegnare utenti o gruppi dalla tua directory IAM Identity Center al tuo gruppo di lavoro Athena abilitato per IAM Identity Center.

Esempio di politica relativa ai ruoli

L'esempio seguente mostra una policy per un ruolo da associare a una posizione Amazon S3 Access Grant che limita l'accesso ai risultati delle query Athena.

```
{
    "Statement": [{
        "Action": ["s3:*"],
        "Condition": {
            "ArnNotEquals": {
                "s3:AccessGrantsInstanceArn": "arn:aws:s3:${region}:${account}:access-
grants/default"
            },
            "StringNotEquals": {
                "aws:ResourceAccount": "${account}"
            }
        },
        "Effect": "Deny",
        "Resource": "*",
        "Sid": "ExplicitDenyS3"
    }, {
        "Action": ["kms:*"],
        "Effect": "Deny",
        "NotResource": "arn:aws:kms:${region}:${account}:key/${keyid}",
        "Sid": "ExplictDenyKMS"
    }, {
        "Action": ["s3:ListMultipartUploadParts", "s3:GetObject"],
        "Condition": {
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": "arn:aws:s3:${region}:${account}:access-
grants/default"
            },
            "StringEquals": {
                "aws:ResourceAccount": "${account}"
            }
        },
        "Effect": "Allow",
        "Resource": "arn:aws:s3:::ATHENA-QUERY-RESULT-LOCATION/${identitystore:UserId}/
*",
        "Sid": "ObjectLevelReadPermissions"
    }, {
        "Action": ["s3:PutObject", "s3:AbortMultipartUpload"],
        "Condition": {
            "ArnEquals": {
```

```
                "s3:AccessGrantsInstanceArn": "arn:aws:s3:${region}:${account}:access-
grants/default"
            },
            "StringEquals": {
                "aws:ResourceAccount": "${account}"
            }
        },
        "Effect": "Allow",
        "Resource": "arn:aws:s3:::ATHENA-QUERY-RESULT-LOCATION/${identitystore:UserId}/
*",
        "Sid": "ObjectLevelWritePermissions"
    }, {
        "Action": "s3:ListBucket",
        "Condition": {
            "ArnEquals": {
                "s3:AccessGrantsInstanceArn": "arn:aws:s3:${region}:${account}:access-
grants/default"
            },
            "StringEquals": {
                "aws:ResourceAccount": "${account}"
            },
            "StringLikeIfExists": {
                "s3:prefix": ["${identitystore:UserId}", "${identitystore:UserId}/*"]
            }
        },
        "Effect": "Allow",
        "Resource": "arn:aws:s3:::ATHENA-QUERY-RESULT-LOCATION",
        "Sid": "BucketLevelReadPermissions"
    }, {
        "Action": ["kms:GenerateDataKey", "kms:Decrypt"],
        "Effect": "Allow",
        "Resource": "arn:aws:kms:${region}:${account}:key/${keyid}",
        "Sid": "KMSPermissions"
    }],
    "Version": "2012-10-17"
}
```

API per gruppi di lavoro Athena

Di seguito sono elencate alcune delle operazioni REST API utilizzate per i gruppi di lavoro Athena. In tutte le operazioni seguenti, tranne `ListWorkGroups`, è necessario specificare un gruppo di lavoro. In altre operazioni, ad esempio `StartQueryExecution`, il parametro del gruppo di lavoro

è facoltativo e le operazioni non sono elencate qui. Per un elenco delle operazioni, consulta la documentazione di riferimento dell'API Amazon Athena.

- CreateWorkGroup
- DeleteWorkGroup
- GetWorkGroup
- ListWorkGroups
- UpdateWorkGroup

Risoluzione dei problemi relativi ai gruppi di lavoro

Utilizza i seguenti suggerimenti per risolvere i problemi relativi ai gruppi di lavoro.

- Controlla le autorizzazioni per i singoli utenti nel tuo account. Devono avere accesso alla posizione per i risultati delle query e al gruppo di lavoro in cui intendono eseguire query. Se vogliono passare da un gruppo di lavoro a un altro, anche loro devono disporre delle autorizzazioni per entrambi. Per informazioni, consulta Policy IAM per l'accesso ai gruppi di lavoro.
- Presta attenzione al contesto nella console Athena, per individuare il gruppo di lavoro in cui eseguirai le query. Se usi il driver, assicurati di impostare il gruppo di lavoro su quello di cui hai bisogno. Per informazioni, consulta the section called “Specificare un gruppo di lavoro in cui eseguire query”.
- Se utilizzi l'API o i driver per eseguire le query, devi specificare la posizione dei risultati delle query utilizzando uno dei seguenti modi: per le singole query, usa (lato client). OutputLocation Nel gruppo di lavoro, usa. WorkGroupConfiguration Se la posizione non viene specificata in nessuno dei due modi, Athena genera un errore durante il runtime della query.
- Se sostituisci le impostazioni lato client con le impostazioni del gruppo di lavoro, è possibile che si verifichino errori relativi alla posizione dei risultati delle query. Ad esempio, un utente del gruppo di lavoro potrebbe non avere le autorizzazioni necessarie per la posizione del gruppo di lavoro in Amazon S3 per archiviare i risultati delle query. In questo caso, aggiungi le autorizzazioni necessarie.
- I gruppi di lavoro introducono cambiamenti di comportamento delle operazioni API. Le chiamate alle seguenti operazioni API esistenti richiedono che gli utenti nel tuo account dispongano delle autorizzazioni basate su risorse in IAM per i gruppi di lavoro in cui vengono effettuate. Se non esistono autorizzazioni per il gruppo di lavoro e per le azioni del gruppo di lavoro, vengono generate le seguenti

azioni APIAccessDeniedException: CreateNamedQueryDeleteNamedQuery, GetNamedQuery, ListNamedQueries, StartQueryExecution, StopQueryExecution, ListQueryExecutionsGetQueryExecutionGetQueryResults, e GetQueryResultsStream(questa azione API è disponibile solo per l'uso con il driver e non è altrimenti esposta per uso pubblico). Per ulteriori informazioni, consulta Operazioni, risorse e chiavi di condizione per Amazon Athena nella Documentazione di riferimento per l'autorizzazione ai servizi.

Le chiamate alle operazioni BatchGetQueryExecutione all'BatchGetNamedQueryAPI restituiscono informazioni solo sulle query eseguite nei gruppi di lavoro a cui gli utenti hanno accesso. Se l'utente non ha accesso al gruppo di lavoro, queste operazioni API restituiscono gli ID di query non autorizzati nel contesto dell'elenco di ID non elaborati. Per ulteriori informazioni, consulta the section called “ API per gruppi di lavoro Athena”.

- Se il gruppo di lavoro in cui viene eseguita una query è configurato con una posizione dei risultati di query applicata, non specificare e specifichi una `external_location` per la query CTAS. Athena genera un errore e una query che specifica una `external_location` in questo caso. Ad esempio, la query ha esito negativo se si sostituiscono le impostazioni lato client per la posizione dei risultati delle query impostando il gruppo di lavoro in modo da utilizzare la propria posizione: `CREATE TABLE <DB>.<TABLE1> WITH (format='Parquet', external_location='s3://DOC-EXAMPLE-BUCKET/test/') AS SELECT * FROM <DB>.<TABLE2> LIMIT 10;`

Potrebbero essere visualizzati i seguenti errori. Questa tabella fornisce un elenco di errori correlati ai gruppi di lavoro e suggerisce le soluzioni.

Errori dei gruppi di lavoro

Errore	Si verifica quando...
query state CANCELED. È stato superato il limite di byte scansionati.	Una query raggiunge un limite di dati a livello di query e viene annullata. Prova a riscriver e la query in modo che legga una quantità inferiore di dati oppure rivolgiti all'ammin istratore dell'account.
Utente: arn:aws:iam: :12345678 9012:user/abc non è autorizzato a eseguire: athena: on resource: arn:aws:athena:us-east- 1:1234567	Un utente esegue una query in un gruppo di lavoro, ma non è autorizzato ad accedervi. Aggiorna la policy per avere accesso al gruppo di lavoro.

Errore	Si verifica quando...
89012:workgroup/workgroupname StartQueryExecution	
INPUT NON VALIDO. WorkGroup <name>è disabilitato.	Un utente esegue una query in un gruppo di lavoro, ma il gruppo di lavoro è disabilitato. Il gruppo di lavoro potrebbe essere stato disabilit ato dall'amministratore. È anche possibile che tu non sia autorizzato ad accedervi. In entrambi i casi, contatta un amministratore che dispone dell'accesso per la modifica dei gruppi di lavoro.
INVALID_INPUT. WorkGroup <name>non è stato trovato.	Un utente esegue una query in un gruppo di lavoro, ma il gruppo di lavoro non esiste. Questo può accadere se il gruppo di lavoro è stato eliminato. Passa a un altro gruppo di lavoro per eseguire la query.
InvalidRequestException: quando si chiama l' StartQueryExecutionoperazione: non viene fornita alcuna posizione di output. È richiesta una posizione di output tramite l'impostazione di configurazione dei risultati del gruppo di lavoro o come input API.	Un utente esegue una query con l'API senza specificare la posizione per i risultati delle query. È necessario impostare la posizione di output per i risultati delle query utilizzando uno dei due modi: per le singole query, utilizzan do OutputLocation(lato client) o nel gruppo di lavoro, utilizzando. WorkGroupConfiguration
La query Crea tabella come selezione non è riuscita perché è stata inviata con una proprietà 'external_location' a un gruppo di lavoro Athena che impone una posizione di output centraliz zata per tutte le query. Rimuovi la proprietà 'external_location' e invia nuovamente la query.	Se il gruppo di lavoro in cui viene eseguita una query è configurato con una posizione dei risultati di query applicata e specifichi `external_location` per la query CTAS. In questo caso, rimuovi `external_location` ed esegui di nuovo la query.

Errore	Si verifica quando...
Impossibile creare una istruzione preparata *nome_istruzione_preparata* . Il numero di dichiarazioni preparate in questo gruppo di lavoro supera il limite di 1.000.	Il gruppo di lavoro contiene più del limite di 1.000 istruzioni preparate. Per risolvere il problema, utilizza DEALLOCATE PREPARE per rimuovere una o più istruzioni preparate dal gruppo di lavoro. In alternativa, crea un nuovo gruppo di lavoro.

Controllo dei costi e monitoraggio delle interrogazioni con metriche ed eventi CloudWatch

I gruppi di lavoro consentono di impostare limiti di controllo dell'utilizzo dei dati per query o per gruppo di lavoro, impostare allarmi quando tali limiti vengono superati e pubblicare le metriche delle query su. CloudWatch

In ogni gruppo di lavoro è possibile:

- Configurare Data usage controls (Controlli di utilizzo dei dati) a livello di query e di gruppo di lavoro e definire le operazioni da eseguire se le query superano le soglie.
- Visualizza e analizza le metriche delle query e pubblicale su. CloudWatch Se crei un gruppo di lavoro nella console, l'impostazione per la pubblicazione delle metriche CloudWatch viene selezionata automaticamente. Se si utilizzano le operazioni API, è necessario abilitare la pubblicazione dei parametri. Quando i parametri vengono pubblicati, sono visualizzati nella scheda Metrics (Parametri) del riquadro Workgroups (Gruppi di lavoro). I parametri sono disabilitati per impostazione predefinita per il gruppo di lavoro principale.

Video

Il video seguente mostra come creare dashboard personalizzate e impostare allarmi e trigger sulle metriche in. CloudWatch È possibile utilizzare pannelli di controllo prepopolati direttamente dalla console Athena per utilizzare questi parametri di query.

Monitoraggio delle query di Amazon Athena tramite Amazon CloudWatch

Argomenti

- Abilitazione delle metriche di interrogazione CloudWatch

- Monitoraggio delle query Athena con metriche CloudWatch
- Monitoraggio delle query Athena con eventi Amazon EventBridge
- Monitoraggio dei parametri di utilizzo di Athena
- Impostazione dei limiti di controllo dell'utilizzo dei dati

Abilitazione delle metriche di interrogazione CloudWatch

Quando crei un gruppo di lavoro nella console, l'impostazione per la pubblicazione delle metriche delle query su CloudWatch è selezionata per impostazione predefinita.

Per abilitare o disabilitare i parametri di query nella console Athena per un gruppo di lavoro

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Nel pannello di navigazione, seleziona Workgroups (Gruppi di lavoro).
4. Scegli il collegamento del gruppo di lavoro da modificare.
5. Nella pagina dei dettagli per il gruppo di lavoro, scegli Edit (Modifica).
6. Nella sezione Impostazioni, seleziona o deseleziona Pubblica le metriche delle query su. AWS CloudWatch

Se utilizzi le operazioni API, l'interfaccia a riga di comando o l'applicazione client con il driver JDBC per creare gruppi di lavoro, per abilitare la pubblicazione delle metriche delle query, imposta su

in. `PublishCloudWatchMetricsEnabled` `true` WorkGroupConfiguration L'esempio seguente mostra solo la configurazione dei parametri senza altre configurazioni:

```
"WorkGroupConfiguration": {
      "PublishCloudWatchMetricsEnabled": "true"
     ....
     }
```

Monitoraggio delle query Athena con metriche CloudWatch

Athena pubblica le metriche relative alle query su CloudWatch Amazon, quando è selezionata l'opzione Pubblica metriche di query su. CloudWatch Puoi creare dashboard personalizzate, impostare allarmi e trigger sulle metriche o utilizzare dashboard precompilate CloudWatch direttamente dalla console Athena.

Quando si abilitano i parametri di query per le query nei gruppi di lavoro, i parametri vengono visualizzati nella tab Parametri del pannello Gruppi di lavoro per ogni gruppo di lavoro nella console Athena.

Athena pubblica le seguenti metriche sulla console: CloudWatch

- `DPUAllocated`: il numero totale di DPU (unità di elaborazione dati) fornite in una prenotazione della capacità per eseguire le query.
- `DPUConsumed`: il numero di DPU utilizzate attivamente dalle query in uno stato `RUNNING` in un determinato momento di una prenotazione. Parametro emesso solo quando il gruppo di lavoro è associato a una prenotazione della capacità e include tutti i gruppi di lavoro associati a una prenotazione.
- `DPUCount`: il numero massimo di DPU utilizzate dalla query, pubblicate esattamente una volta completata la query.
- `EngineExecutionTime`: il numero di millisecondi necessari per l'esecuzione della query.
- `ProcessedBytes`: il numero di byte scansionati da Athena per query DML.
- `QueryPlanningTime`: il numero di millisecondi richiesti da Athena per pianificare il flusso di elaborazione delle query.
- `QueryQueueTime`: il numero di millisecondi di permanenza della query nella coda di query in attesa delle risorse.
- `ServicePreProcessingTime`: il numero di millisecondi richiesti da Athena per pre-elaborare la query prima di inviarla al motore di query.

- `ServiceProcessingTime`: il numero di millisecondi richiesti da Athena per elaborare i risultati della query dopo che il motore di query ha terminato l'esecuzione della query.
- `TotalExecutionTime`: il numero di millisecondi richiesti da Athena per eseguire una query DDL o DML.

Per le descrizioni più complete, consulta Elenco di CloudWatch metriche e dimensioni per Athena più avanti in questo documento.

Questi parametri hanno le seguenti dimensioni:

- `CapacityReservation`: il nome della prenotazione della capacità utilizzata per eseguire la query, se applicabile.
- `QueryState` – `SUCCEEDED`, `FAILED`, o `CANCELED`
- `QueryType` – `DML`, `DDL`, o `UTILITY`
- `WorkGroup` – nome del gruppo di lavoro

Athena pubblica la seguente metrica sulla CloudWatch console nel namespace: `AmazonAthenaForApacheSpark`

- `DPUCount`: il numero di DPU utilizzate durante la sessione per eseguire i calcoli.

Questo parametro possiede le seguenti dimensioni:

- `SessionId`: l'ID della sessione in cui vengono inviati i calcoli.
- `WorkGroup`: il nome del gruppo di lavoro.

Per ulteriori informazioni, consulta Elenco di CloudWatch metriche e dimensioni per Athena più avanti in questo argomento. Per ulteriori informazioni sui parametri di utilizzo di Athena, consulta Monitoraggio dei parametri di utilizzo di Athena.

Per visualizzare i parametri di query per un gruppo di lavoro nella console

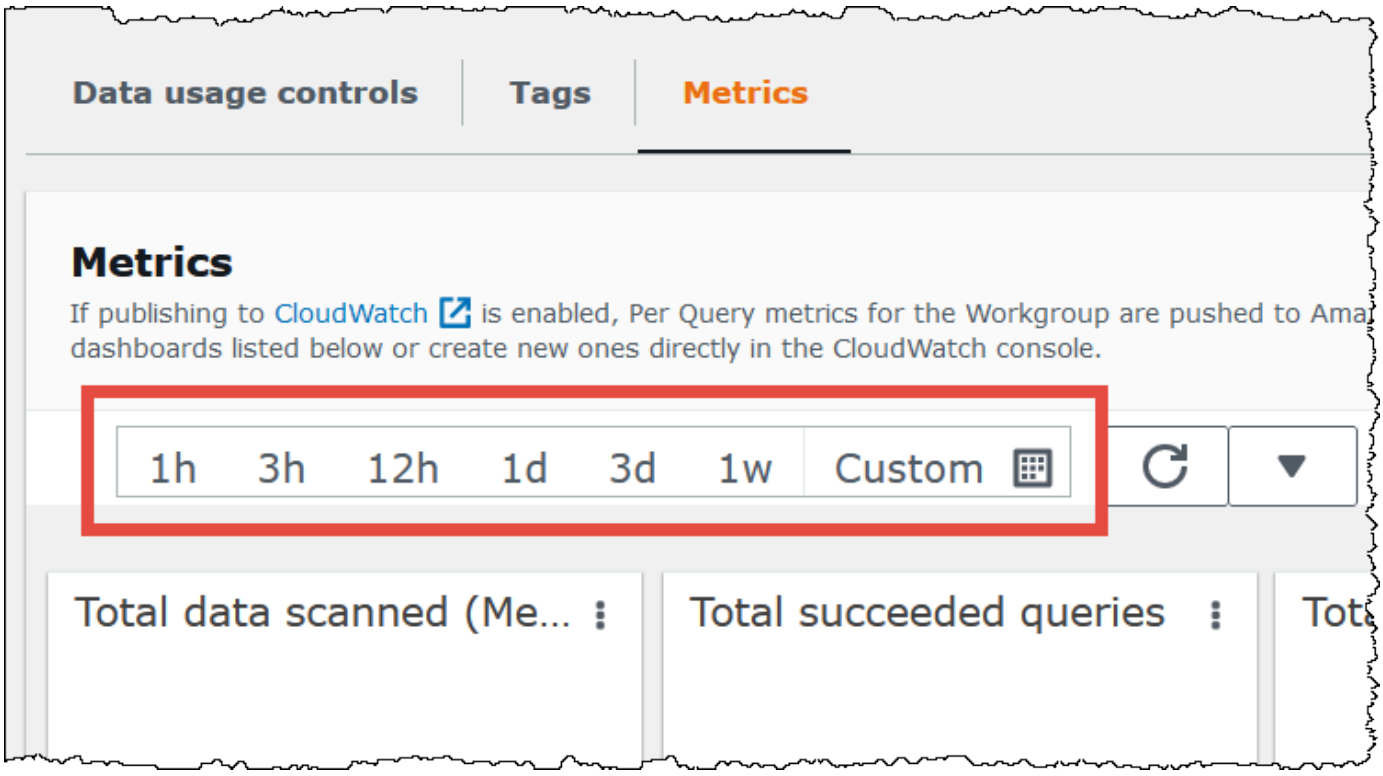
1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Nel pannello di navigazione, seleziona Workgroups (Gruppi di lavoro).
4. Scegli il gruppo di lavoro desiderato dall'elenco, quindi seleziona la scheda Metrics (Parametri).

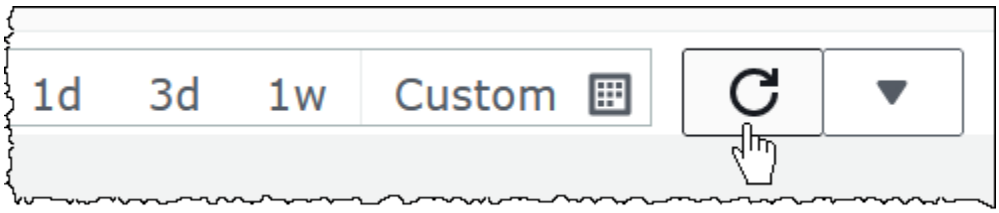
   Viene visualizzato il pannello di controllo dei parametri.

   > **Note**
   >
   > Se i parametri sono stati abilitati per il gruppo di lavoro e/o recentemente non c'è stata alcuna attività di query, i grafici sul pannello di controllo potrebbero essere vuoti. L'attività di interrogazione viene recuperata in CloudWatch base all'intervallo specificato nel passaggio successivo.

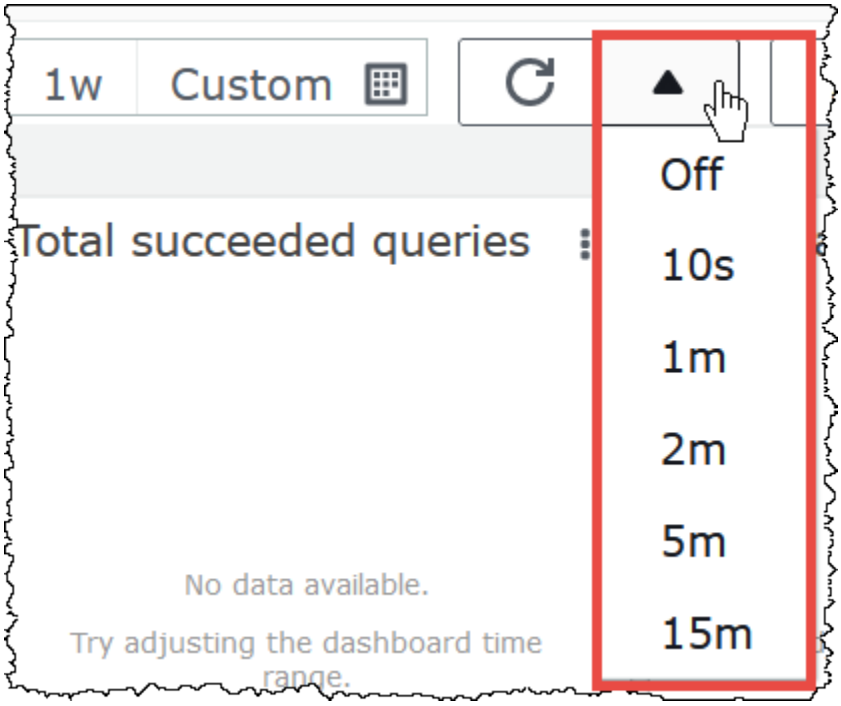
5. Nella sezione Metriche, scegli l'intervallo di metriche che Athena deve utilizzare per recuperare le metriche della query o specifica un intervallo personalizzato. CloudWatch



6. Per aggiornare i parametri visualizzati, scegliere l'icona di aggiornamento.



7. Fai clic sulla freccia accanto all'icona di aggiornamento per scegliere la frequenza di aggiornamento per la visualizzazione dei parametri.



Per visualizzare le metriche nella console Amazon CloudWatch

1. Apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Nel pannello di navigazione, seleziona Metrics (Parametri), All metrics (Tutti i parametri).
3. Seleziona lo spazio dei nomi AWS/Athena.

Per visualizzare i parametri con CLI

- Esegui una di queste operazioni:
  - Per elencare i parametri per Athena, apri un prompt dei comandi e usa il comando seguente:

```
aws cloudwatch list-metrics --namespace "AWS/Athena"
```

  - Per visualizzare un elenco di tutti i parametri disponibili, usare il comando seguente:

```
aws cloudwatch list-metrics"
```

Elenco di CloudWatch metriche e dimensioni per Athena

Se hai abilitato le CloudWatch metriche in Athena, invia le seguenti metriche CloudWatch a ciascun gruppo di lavoro. I parametri seguenti utilizzano lo spazio dei nomi `AWS/Athena`.

Nome parametro	Descrizione
DPUAllocated	Il numero totale di DPU (unità di elaborazione dati) fornite in una prenotazione della capacità per eseguire le query.
DPUConsumed	Il numero di DPU utilizzate attivamente dalle query in uno stato `RUNNING` in un determinato momento di una prenotazione. Questo parametro viene emesso solo quando il gruppo di lavoro è associato a una prenotazione della capacità e include tutti i gruppi di lavoro associati a una prenotazione. Se sposti un gruppo di lavoro da una prenotazione all'altra, il parametro include i dati del momento in cui il gruppo di lavoro apparteneva alla prima prenotazione. Per informazioni delle prenotazioni della capacità, consulta Gestione della capacità di elaborazione delle query.
DPUCount	Il numero massimo di DPU utilizzate dalla query, pubblicate esattamente una volta completata la query. Questo parametro viene emesso solo per i gruppi di lavoro collegati a una prenotazi one della capacità.
EngineExecutionTime	Il numero di millisecondi necessari per l'esecuzione della query.
ProcessedBytes	Il numero di byte analizzati da Athena per query DML. Per le query che sono state annullate (dagli utenti o automaticamente se hanno raggiunto il limite), questo valore include la quantità di dati analizzati prima dell'annullamento. Questo parametro non viene segnalato per le query DDL o CTAS.
QueryPlanningTime	Il numero di millisecondi richiesti da Athena per pianificare il flusso di elaborazione delle query. Include il tempo impiegato per recuperare le partizioni della tabella dall'origine dati; Tieni presente che, poiché il motore di query esegue la pianificazione

Nome parametro	Descrizione
	delle query, il tempo di pianificazione delle query è un sottoinsi eme di. EngineExecutionTime
QueryQueueTime	Il numero di millisecondi di permanenza della query nella coda di query in attesa delle risorse. Se si verificano errori temporanei, la query può essere aggiunta automaticamente alla coda.
ServicePreProcessingTime	Il numero di millisecondi richiesti da Athena per pre-elaborare la query prima di inviarla al motore di query.
ServiceProcessingTime	Il numero di millisecondi richiesti da Athena per elaborare i risultati della query dopo che il motore di query ha terminato l'esecuzione della query.
TotalExecutionTime	Il numero di millisecondi richiesti da Athena per eseguire una query DDL o DML. TotalExecutionTime include QueryQueu eTime, QueryPlanningTime EngineExecutionTime, e ServicePr ocessingTime.

Questi parametri per Athena possiedono le seguenti dimensioni.

Dimensione	Descrizione
CapacityReservation	Il nome della prenotazione della capacità utilizzata per eseguire la query, se applicabile. Quando non viene utilizzata una prenotazione della capacità, questa dimensione non restituisce dati.
QueryState	Lo stato della query. Statistiche valide: SUCCEEDED, FAILED o CANCELED.
QueryType	Il tipo di query. Statistiche valide: `DDL`, `DML` o `UTILITY`. Il tipo di istruzione di query che è stata eseguita. `DDL` indica le istruzioni di query DDL (Data Definition Language). `DML` indica istruzioni di query DML

Dimensione	Descrizione
	(Data Manipulation Language), ad esempio `CREATE TABLE AS SELECT`. `UTILITY` indica istruzioni di query diverse da DDL e DML, ad esempio `SHOW CREATE TABLE` o `DESCRIBE TABLE`.
WorkGroup	Il nome del gruppo di lavoro.

Monitoraggio delle query Athena con eventi Amazon EventBridge

Puoi utilizzare Amazon Athena con Amazon EventBridge per ricevere notifiche in tempo reale sullo stato delle tue richieste. Quando una query a cui hai inviato lo stato di transizione, Athena pubblica un evento EventBridge contenente informazioni sulla transizione dello stato della query. Puoi scrivere semplici regole per gli eventi che ti interessano e intraprendere azioni automatiche quando un evento corrisponde a una regola. Ad esempio, è possibile creare una regola che richiami una AWS Lambda funzione quando una query raggiunge uno stato terminale. Gli eventi vengono emessi secondo il principio del massimo sforzo.

Prima di creare regole per gli eventi per Athena, tuttavia, dovresti assicurarti di:

- Acquisisci familiarità con eventi, regole e obiettivi in. EventBridge Per ulteriori informazioni, consulta What Is Amazon EventBridge? Per ulteriori informazioni su come configurare le regole, consulta la sezione Guida introduttiva ad Amazon EventBridge.
- Creare la destinazione o le destinazioni da utilizzare nelle regole degli eventi.

> **Note**
>
> Athena offre attualmente un tipo di evento, Athena Query State Change, ma può aggiungere altri tipi di eventi e dettagli. Se deserializzi in modo programmatico dati JSON di eventi, assicurati che l'applicazione sia in grado di gestire proprietà sconosciute se proprietà aggiuntive verranno aggiunte.

Formato di un evento Athena

Di seguito è riportato il modello di base per un evento Amazon Athena.

```
{
```

```
    "source":[
        "aws.athena"
    ],
    "detail-type":[
        "Athena Query State Change"
    ],
    "detail":{
        "currentState":[
            "SUCCEEDED"
        ]
    }
}
```

Evento di modifica dello stato di query Athena

Nell'esempio seguente viene illustrato un evento Athena Query State Change con un valore di `currentState` pari a `SUCCEEDED`.

```
{
    "version":"0",
    "id":"abcdef00-1234-5678-9abc-def012345678",
    "detail-type":"Athena Query State Change",
    "source":"aws.athena",
    "account":"123456789012",
    "time":"2019-10-06T09:30:10Z",
    "region":"us-east-1",
    "resources":[

    ],
    "detail":{
        "versionId":"0",
        "currentState":"SUCCEEDED",
        "previousState":"RUNNING",
        "statementType":"DDL",
        "queryExecutionId":"01234567-0123-0123-0123-012345678901",
        "workgroupName":"primary",
        "sequenceNumber":"3"
    }
}
```

Nell'esempio seguente viene illustrato un evento Athena Query State Change con un valore di `currentState` pari a `FAILED`. Il blocco `athenaError` appare solo quando `currentState` è

FAILED. Per informazioni sui valori per errorCategory e errorType, consulta la pagina catalogo degli errori Athena.

```
{
    "version":"0",
    "id":"abcdef00-1234-5678-9abc-def012345678",
    "detail-type":"Athena Query State Change",
    "source":"aws.athena",
    "account":"123456789012",
    "time":"2019-10-06T09:30:10Z",
    "region":"us-east-1",
    "resources":[
    ],
    "detail":{
        "athenaError": {
            "errorCategory": 2.0, //Value depends on nature of exception
            "errorType": 1306.0, //Type depends on nature of exception
            "errorMessage": "Amazon S3 bucket not found", //Message depends on nature
 of exception
            "retryable":false //Retryable value depends on nature of exception
        },
        "versionId":"0",
        "currentState": "FAILED",
        "previousState": "RUNNING",
        "statementType":"DML",
        "queryExecutionId":"01234567-0123-0123-0123-012345678901",
        "workgroupName":"primary",
        "sequenceNumber":"3"
    }
}
```

Proprietà di output

L'output JSON include le seguenti proprietà.

Proprietà	Descrizione
athenaError	Appare solo quando currentState è FAILED. Contiene informazioni sull'errore che si è verificato, tra cui la categoria, il tipo e il messaggio di errore, nonché se è possibile ripetere l'operazione che ha portato all'error e. I valori per ciascuno di questi campi dipendono dalla natura dell'erro

Proprietà	Descrizione
	re. Per informazioni sui valori per `errorCategory` e `errorType`, consulta la pagina catalogo degli errori Athena.
`versionId`	Il numero di versione per lo schema dell'oggetto di dettaglio.
`currentState`	Lo stato in cui la query è passata al momento dell'evento.
`previousState`	Lo stato da cui la query è transitata al momento dell'evento.
`statementType`	Il tipo di istruzione di query che è stata eseguita.
`queryExecutionId`	Identificatore univoco per la query in esecuzione.
`workgroupName`	Il nome del gruppo di lavoro in cui è stata eseguita la query.
`sequenceNumber`	Un numero crescente monotonicamente che consente la deduplicazione e l'ordinamento di eventi in ingresso che coinvolgono l'esecuzione di una query singola. Quando vengono pubblicati eventi duplicati per la stessa transizione di stato, il valore `sequenceNumber` è lo stesso. Quando una query sperimenta una transizione di stato più di una volta, ad esempio query che verificano richieste rare, è possibile utilizzare `sequenceNumber` per ordinare eventi con valori `currentState` e `previousState` identici.

Esempio

Nell'esempio seguente vengono pubblicati gli eventi in un argomento Amazon SNS sottoscritto. Quando Athena viene interrogato, si riceve un'e-mail. Nell'esempio si presuppone che l'argomento Amazon SNS esista e che sia stato sottoscritto.

Per pubblicare eventi Athena in un argomento Amazon SNS

1. Crea il target per il tuo argomento Amazon SNS. Concedi agli EventBridge eventi Service Principal `events.amazonaws.com` l'autorizzazione a pubblicare sul tuo argomento Amazon SNS, come nell'esempio seguente.

```
{
    "Effect":"Allow",
    "Principal":{
        "Service":"events.amazonaws.com"
    },
    "Action":"sns:Publish",
    "Resource":"arn:aws:sns:us-east-1:111111111111:your-sns-topic"
}
```

2. Utilizzate il AWS CLI `events put-rule` comando per creare una regola per gli eventi Athena, come nell'esempio seguente.

```
aws events put-rule --name {ruleName} --event-pattern '{"source": ["aws.athena"]}'
```

3. Utilizza il AWS CLI `events put-targets` comando per allegare l'argomento di destinazione di Amazon SNS alla regola, come nell'esempio seguente.

```
aws events put-targets --rule {ruleName} --targets Id=1,Arn=arn:aws:sns:us-
east-1:111111111111:your-sns-topic
```

4. Interrogare Athena e osservare il target invocato. Dovresti ricevere le e-mail corrispondenti dall'argomento Amazon SNS.

Utilizzo Notifiche all'utente AWS con Amazon Athena

Puoi utilizzare Notifiche all'utente AWS per configurare i canali di consegna per ricevere notifiche sugli eventi Amazon Athena. L'utente riceverà una notifica quando un evento corrisponde a una regola specificata. È possibile ricevere notifiche per gli eventi tramite più canali, tra cui e-mail, notifiche chat AWS Chatbot o notifiche push AWS Console Mobile Application. Puoi anche visualizzare le notifiche nel Centro notifiche della console. Notifiche all'utente supporta l'aggregazione, che può ridurre il numero di notifiche ricevute durante eventi specifici.

Per ulteriori informazioni, consulta la Guida per l'utente Notifiche all'utente AWS .

Monitoraggio dei parametri di utilizzo di Athena

Puoi utilizzare le metriche di CloudWatch utilizzo per fornire visibilità sul modo in cui il tuo account utilizza le risorse visualizzando l'utilizzo corrente del servizio su CloudWatch grafici e dashboard.

Per Athena, le metriche di disponibilità dell'utilizzo corrispondono alle Servizio AWS quote per Athena. È possibile configurare gli allarmi che avvisano quando l'uso si avvicina a una quota di servizio. Per ulteriori informazioni sulle Service Quotas di Athena, consulta Service Quotas (Quote di Servizio). Per ulteriori informazioni sui parametri di AWS utilizzo, consulta i parametri di AWS utilizzo nella Amazon CloudWatch User Guide.

Athena pubblica i seguenti parametri nello spazio dei nomi `AWS/Usage`.

Nome parametro	Descrizione
`ResourceCount`	La somma di tutte le query in coda e in esecuzione per account, separate Regione AWS per tipo di query (DML o DDL). Maximum è l'unica statistica utile per questa metrica. Questo parametro viene pubblicato periodicamente ogni minuto. Se viene eseguita alcuna query, il parametro non segnala nulla (nemmeno 0). Questo parametro viene pubblicato solo se al momento della sua esecuzione, sono in esecuzione anche le query attive.

Le seguenti dimensioni vengono utilizzate per perfezionare i parametri di utilizzo pubblicati da Athena.

Dimensione	Descrizione
`Service`	Il nome della risorsa Servizio AWS che contiene la risorsa. Per Athena, il valore di questa dimensione è `Athena`.
`Resource`	Il tipo di risorsa in esecuzione. Il valore della risorsa per l'utilizzo della query Athena è `ActiveQueryCount` .
`Type`	Il tipo di entità che viene segnalato. Attualmente, l'unico valore valido per i parametri di utilizzo di Athena è `Resource`.
`Class`	La classe della risorsa monitorata. Per Athena, `Class` può essere `DML` o `DDL`.

Visualizzazione dei parametri di utilizzo delle risorse Athena nella console CloudWatch

Puoi utilizzare la CloudWatch console per visualizzare un grafico delle metriche di utilizzo di Athena e configurare allarmi che ti avvisano quando l'utilizzo si avvicina a una quota di servizio.

Come visualizzare i parametri di utilizzo delle risorse Athena

1. Apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Nel pannello di navigazione, seleziona Metrics (Parametri), All metrics (Tutti i parametri).
3. Seleziona Utilizzo, quindi seleziona Per risorsa AWS .

   Viene visualizzato l'elenco dei parametri di utilizzo delle quote di servizio.
4. Seleziona la casella di controllo che si trova accanto a Athena e. ActiveQueryCount
5. Seleziona la scheda Graphed metrics (Parametri nel grafico).

   Il grafico qui sopra mostra l'utilizzo corrente della AWS risorsa.

Per informazioni sull'aggiunta di quote di servizio al grafico e sull'impostazione di un allarme che ti avvisa se ti avvicini alla quota di servizio, consulta Visualizzazione delle quote di servizio e impostazione degli allarmi nella Amazon User Guide. CloudWatch Per informazioni sull'impostazione dei limiti di utilizzo per gruppo di lavoro, consultare Impostazione dei limiti di controllo dell'utilizzo dei dati.

Impostazione dei limiti di controllo dell'utilizzo dei dati

Athena consente di impostare due tipi di controlli dei costi: limite per query e limite per gruppo di lavoro. Per ogni gruppo di lavoro è possibile impostare un solo limite per query e più limiti per gruppo di lavoro.

- Il limite di controllo a livello di query specifica la quantità totale di dati analizzati per query. Se una delle query eseguite nel gruppo di lavoro supera il limite, viene annullata. È possibile creare un solo limite di controllo a livello di query in un gruppo di lavoro, che viene applicato a tutte le query eseguite al suo interno. Se necessario, è possibile modificare il limite. Per una procedura dettagliata, consulta Per creare un controllo di utilizzo dei dati a livello di query.
- Il limite di controllo dell'utilizzo dei dati a livello di gruppo di lavoro specifica la quantità totale di dati analizzati per tutte le query eseguite nel gruppo di lavoro nel periodo di tempo specificato. È possibile creare più limiti per gruppo di lavoro. Il limite di query a livello di gruppo di lavoro consente

di impostare più soglie su aggregati orari o giornalieri dei dati analizzati dalle query in esecuzione nel gruppo di lavoro.

Se la quantità aggregata dei dati analizzati supera la soglia, puoi inviare una notifica a un argomento Amazon SNS. Per farlo, puoi configurare un allarme Amazon SNS e un'operazione nella console Athena per notificare il superamento del limite a un amministratore. Per una procedura dettagliata, consulta Per creare un controllo di utilizzo dei dati a livello di gruppo di lavoro. Puoi anche creare un allarme e un'azione su qualsiasi metrica pubblicata da Athena dalla console. CloudWatch Ad esempio, è possibile impostare un avviso al raggiungimento di un numero di query non riuscite. Questo avviso può determinare l'invio di un'e-mail a un amministratore se il numero supera una determinata soglia. Se il limite viene superato, viene inviata una notifica di allarme Amazon SNS agli utenti specificati.

Altre operazioni intraprese:

- Richiamare una funzione Lambda. Per ulteriori informazioni, consulta Invocazione di funzioni Lambda tramite le notifiche di Amazon SNS nella Guida per sviluppatori di Amazon Simple Queue Service.
- Disabilita il gruppo di lavoro, per interrompere l'esecuzione di ulteriori query. Per le fasi, consulta Abilitare e disabilitare un gruppo di lavoro.

I limiti a livello di query e di gruppo di lavoro sono indipendenti. Viene effettuata una determinata operazione ogni volta che uno dei due limiti viene superato. Se due o più utenti eseguono query contemporaneamente nello stesso gruppo di lavoro, è possibile che le singole query non superino i limiti specificati, ma che la somma totale dei dati analizzati superi il limite di utilizzo dei dati a livello di gruppo di lavoro. In questo caso, viene inviato un allarme Amazon SNS all'utente.

Per creare un controllo di utilizzo dei dati a livello di query

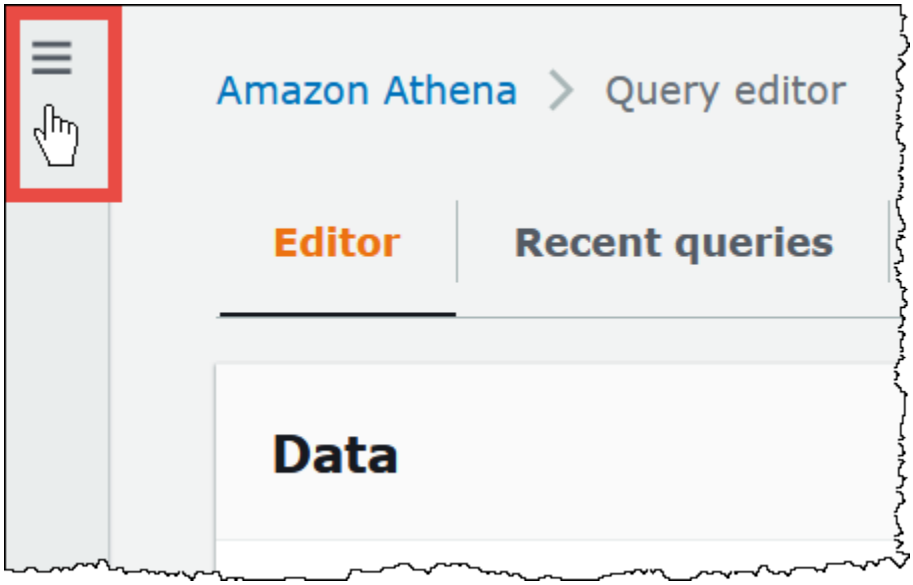
Il limite di controllo a livello di query specifica la quantità totale di dati analizzati per query. Se una delle query eseguite nel gruppo di lavoro supera il limite, viene annullata. Il costo delle query annullate viene calcolato in base ai prezzi di Amazon Athena.

> **Note**
>
> Per le query annullate o non riuscite, Athena potrebbe avere già scritto risultati parziali in Amazon S3. In questi casi, Athena non elimina risultati parziali dal prefisso Amazon S3 in cui vengono archiviati i risultati. Devi rimuovere il prefisso Amazon S3 con risultati parziali. Athena utilizza caricamenti in più parti di Amazon S3 per scrivere dati Amazon S3. È

consigliabile impostare la policy del ciclo di vita del bucket in modo da terminare i caricamenti in più parti in caso di query non riuscite. Per ulteriori informazioni, consulta Interruzione dei caricamenti in più parti incompleti utilizzando una policy per il ciclo di vita del bucket nella Guida per l'utente di Amazon Simple Storage Service.

È possibile creare un solo limite di controllo a livello di query in un gruppo di lavoro, che viene applicato a tutte le query eseguite al suo interno. Se necessario, è possibile modificare il limite.

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.



3. Nel pannello di navigazione, seleziona Workgroups (Gruppi di lavoro).
4. Scegli il nome del gruppo di lavoro dall'elenco.
5. Dalla scheda Data usage controls (Controlli di utilizzo dei dati), nella sezione Per query data usage control (Controllo dell'utilizzo dei dati a livello di query), scegli Manage (Gestisci).
6. Nella pagina Manage per query data usage control (Gestione del controllo dell'utilizzo dei dati a livello di query), specifica i seguenti valori:

   - Per Data limits (Limiti dati) specificare un valore compreso tra 10 MB (minimo) e 7 EB (massimo).

> **Note**
>
> Questi sono i limiti imposti dalla console per i controlli di utilizzo dei dati all'interno dei gruppi di lavoro. Non rappresentano limiti delle query in Athena.

- Per le unità, seleziona il valore dell'unità nell'elenco a discesa, ad esempio, Kilobytes KB (Kilobyte KB) o Exabytes EB(Exabyte EB).

L'operazione di default prevede che la query venga annullata se supera il limite. Questa impostazione non può essere modificata.

7. Selezionare Salva.

Creazione o modifica di un avviso sull'utilizzo dei dati a livello di gruppo di lavoro

È possibile impostare più soglie di avviso quando le query in esecuzione in un gruppo di lavoro analizzano una determinata quantità di dati entro un periodo specifico. Gli avvisi vengono implementati utilizzando gli CloudWatch allarmi di Amazon e si applicano a tutte le query del gruppo di lavoro. Quando viene raggiunta una soglia, è possibile fare in modo che Amazon SNS invii un'e-mail agli utenti specificati. Le query non vengono cancellate automaticamente quando viene raggiunta una soglia.

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.
3. Nel pannello di navigazione, seleziona Workgroups (Gruppi di lavoro).
4. Scegli il nome del gruppo di lavoro dall'elenco.
5. Scegli Edit (Modifica) per modificare le impostazioni del gruppo di lavoro.
6. Scorri fino ad Workgroup data usage alerts - optional (Avvisi sull'utilizzo dei dati a livello di gruppo di lavoro: facoltativo) ed espandi la sezione.
7. Scegli Add a layer (Aggiungi avviso).
8. In Data usage threshold configuration (Configurazione della soglia di utilizzo dei dati), specifica i valori come segue:

   - In Data threshold (Soglia dati) specifica un numero, quindi seleziona un valore unitario dall'elenco a discesa.

- In Time period (Periodo di tempo), scegli un periodo di tempo dall'elenco a discesa.
- In SNS topic selection (Selezione di un argomento SNS), scegli un argomento Amazon SNS dall'elenco a discesa. In alternativa, scegli Create SNS topic (Crea un argomento SNS) per passare direttamente alla console Amazon SNS, creare l'argomento Amazon SNS e configurare un abbonamento per uno degli utenti nel tuo account Athena. Per ulteriori informazioni, consulta Nozioni di base su Amazon SNS nella Guida per gli sviluppatori di Amazon Simple Notification Service.

9. Scegli Add alert (Aggiungi avviso) se stai creando un nuovo avviso oppure Save (Salva) per salvare un avviso esistente.

Gestione della capacità di elaborazione delle query

Utilizza le prenotazioni della capacità per ottenere capacità di elaborazione dedicata per le query eseguite in Athena. Con le prenotazioni della capacità, puoi usufruire delle funzionalità di gestione dei carichi di lavoro che ti aiutano a stabilire le priorità, controllare e dimensionare i carichi di lavoro interattivi più importanti. Ad esempio, puoi aggiungere capacità in qualsiasi momento per aumentare il numero di query eseguibili contemporaneamente, controllare quali carichi di lavoro possono utilizzare la capacità e condividere la capacità tra i carichi di lavoro. La capacità è completamente gestita da Athena e mantenuta per tutto il tempo necessario. La configurazione è semplice e non sono necessarie modifiche alle dichiarazioni SQL.

Per ottenere la capacità di elaborazione delle query, è necessario creare una prenotazione della capacità, specificare il numero di unità di elaborazione dati (DPU) necessarie e assegnare uno o più gruppi di lavoro alla prenotazione.

I gruppi di lavoro svolgono un ruolo importante quando utilizzi le prenotazioni della capacità. I gruppi di lavoro consentono di organizzare le query in raggruppamenti logici. Con le prenotazioni della capacità, puoi assegnare selettivamente capacità ai gruppi di lavoro in modo da controllare la modalità di fatturazione e il comportamento delle richieste per ciascun gruppo di lavoro. Per ulteriori informazioni sui gruppi di lavoro, consulta Uso dei gruppi di lavoro per controllare l'accesso alle query e i costi.

L'assegnazione dei gruppi di lavoro alle prenotazioni consente di dare priorità alle query inviate ai gruppi di lavoro assegnati. Ad esempio, puoi allocare capacità a un gruppo di lavoro utilizzato per le query di report finanziari urgenti in modo da isolare tali query da quelle meno critiche di un altro gruppo di lavoro. Ciò consente l'esecuzione coerente delle query per carichi di lavoro critici, consentendo al contempo l'esecuzione indipendente di altri carichi di lavoro.

Puoi utilizzare contemporaneamente prenotazioni della capacità e gruppi di lavoro per soddisfare requisiti diversi. Di seguito sono riportati alcuni scenari di esempio:

- Isolamento: per isolare un carico di lavoro importante, assegna a una prenotazione un singolo gruppo di lavoro. Solo le query del gruppo di lavoro assegnato utilizzano la capacità di elaborazione della prenotazione scelta.
- Condivisione: più carichi di lavoro utilizzano capacità da una singola prenotazione. Ad esempio, se per un set specifico di carichi di lavoro vuoi che i tuoi costi mensili siano prevedibili, puoi assegnare più gruppi di lavoro a una singola prenotazione. I gruppi di lavoro assegnati condividono la capacità di prenotazione.
- Modello misto: è possibile utilizzare le prenotazioni della capacità e la fatturazione in base alla query nello stesso momento nello stesso account. Ad esempio, per garantire l'esecuzione affidabile delle query che supportano un'applicazione di produzione, si assegna un gruppo di lavoro per tali query a una prenotazione della capacità. Per sviluppare le query prima di spostarle nel gruppo di lavoro di produzione, utilizza un gruppo di lavoro separato che non faccia parte di una prenotazione e utilizza la fatturazione in base alla query.

Informazioni su DPU

La capacità è misurata in unità di elaborazione dati (DPU). Le DPU rappresentano le risorse di calcolo e memoria utilizzate da Athena per accedere ed elaborare i dati per conto tuo. Una DPU fornisce 4 vCPUs e 16 GB di memoria. Il numero di DPU specificato influenza il numero di query che puoi eseguire simultaneamente. Ad esempio, una prenotazione con 256 DPU può consentire circa il doppio del numero di query simultanee rispetto a una prenotazione con 128 DPU.

Puoi creare fino a 100 prenotazioni della capacità con un massimo di 1.000 DPU totali per account e regione. Il numero minimo di DPU che puoi richiedere è 24. Se hai bisogno di più di 1.000 DPU per il tuo caso d'uso, contatta athena-feedback@amazon.com.

Per informazioni sulla stima dei requisiti di capacità, consulta Determinazione dei requisiti di capacità. Per informazioni sui prezzi, consulta Prezzi di Amazon Athena.

Considerazioni e limitazioni

- La funzionalità richiede la versione 3 del motore Athena.
- Puoi assegnare un singolo gruppo di lavoro al massimo a una prenotazione alla volta e puoi aggiungere un massimo di 20 gruppi di lavoro a una prenotazione.

- Non è possibile aggiungere gruppi di lavoro compatibili con Spark a una prenotazione della capacità.
- Per eliminare un gruppo di lavoro assegnato a una prenotazione, per prima cosa rimuovi il gruppo di lavoro dalla prenotazione.
- Il numero minimo di DPU che è possibile fornire è 24.
- Puoi creare fino a 100 prenotazioni della capacità con un massimo di 1.000 DPU totali per account e regione.
- Le richieste di capacità non sono garantite e possono richiedere fino a 30 minuti per essere completate.
- È previsto un periodo minimo di fatturazione di 1 ora per prenotazione. Dopo 1 ora, la capacità viene fatturata al minuto. Per informazioni sui prezzi, consulta Prezzi di Amazon Athena.
- La capacità riservata non è trasferibile a un'altra prenotazione della capacità Account AWS, oppure. Regione AWS
- Le query DDL sulle prenotazioni della capacità consumano DPU.
- Le query eseguite sulla capacità assegnata non vengono considerate nel conteggio dei limiti di query attive per DDL e DML.
- Se tutte le DPU sono in uso, le query inviate vengono messe in coda. Tali query non vengono rifiutate e non vengono inoltrate alla capacità on-demand.
- La `DPUConsumed` CloudWatch metrica è per gruppo di lavoro anziché per prenotazione. Pertanto, se sposti un gruppo di lavoro da una prenotazione all'altra, il parametro `DPUConsumed` include i dati del momento in cui il gruppo di lavoro apparteneva alla prima prenotazione. Per ulteriori informazioni sull'utilizzo delle CloudWatch metriche in Athena, consulta. Monitoraggio delle query Athena con metriche CloudWatch
- Attualmente, la funzionalità è disponibile nelle seguenti versioni: Regioni AWS
  - Stati Uniti orientali (Virginia settentrionale)
  - Stati Uniti orientali (Ohio)
  - US West (Oregon)
  - Asia Pacifico (Singapore)
  - Asia Pacifico (Sydney)
  - Asia Pacifico (Tokyo)
  - Europa (Irlanda)
  - Europa (Spagna)

- Europa (Stoccolma)
- Sud America (San Paolo)

Argomenti

Determinazione dei requisiti di capacità

Prima di creare una prenotazione della capacità, puoi stimare la capacità richiesta in modo da assegnarle il numero corretto di DPU. Inoltre, dopo aver utilizzato una prenotazione, potresti voler verificare se la capacità della prenotazione è insufficiente o eccessiva. Questo argomento descrive le tecniche che è possibile utilizzare per effettuare queste stime e descrive anche alcuni AWS strumenti per valutare l'utilizzo e i costi.

Argomenti

Stima della capacità richiesta

Nella stima dei requisiti di capacità, è utile considerare due punti di vista: quanta capacità potrebbe richiedere una determinata query e quanta capacità potrebbe essere necessaria in generale.

Stima dei requisiti di capacità per query

Per stabilire il numero di DPU che una query potrebbe richiedere, puoi attenerti le seguenti linee guida:

- Le query DDL consumano 4 DPU.

- Le query DML consumano in genere tra le 4 e le 124 DPU.

Athena stabilisce il numero delle DPU richieste da una query DML al momento dell'invio della query. Il numero varia in base alla dimensione dei dati, al formato di archiviazione, al costrutto della query e ad altri fattori. In genere, Athena tenta di selezionare il numero delle DPU più basso ed efficiente. Athena, se stabilisce che è necessaria una maggiore potenza di calcolo per il corretto completamento della query, aumenta il numero delle DPU assegnate alla query.

Stima dei requisiti di capacità specifici del carico di lavoro

Per determinare la capacità necessaria per eseguire più query contemporaneamente, considera le linee guida generali riportate nella tabella seguente:

Query simultanee	DPU richieste
10	40 o più
20	96 o più
30 o più	240 o più

Tieni presente che il numero effettivo di DPU di cui hai bisogno dipende dai tuoi obiettivi e dai modelli di analisi. Ad esempio, se desideri che le query vengano avviate immediatamente senza fare la coda, stabilisci un picco di richiesta di query simultanee, quindi fornisci il numero di DPU adatto.

Puoi effettuare il provisioning di un numero inferiore di DPU rispetto alla domanda di picco, ma quando si verifica un picco di domanda è possibile che si verifichi l'accodamento. Quando si verifica un accodamento, Athena mantiene le query in coda e le esegue quando la capacità diventa disponibile.

Se il tuo obiettivo è eseguire query entro un budget fisso, puoi utilizzare il Calcolatore dei prezzi AWS per stabilire il numero di DPU che rientrano nel budget.

Infine, ricorda che la dimensione dei dati, il formato di archiviazione e il modo di scrittura di una query influiscono sulle DPU richieste da una query. Per aumentare le prestazioni delle query, puoi comprimere o partizionare i dati o convertirli in formati colonnari. Per ulteriori informazioni, consulta Ottimizzazione delle prestazioni in Athena.

Segnali che è necessaria una capacità maggiore

Messaggi di errore relativi alla capacità insufficiente e l'accodamento delle query indicano che la capacità assegnata è inadeguata.

Se le query non vanno a buon fine e viene visualizzato un messaggio di errore relativo ad una capacità insufficiente, probabilmente il numero di DPU di prenotazione della capacità è troppo basso per la query. Ad esempio, se hai una prenotazione con 24 DPU ed esegui una query che richiede più di 24 DPU, la query non riuscirà. Per monitorare questo errore di query, puoi utilizzare gli eventi di EventBridge Athena. Prova ad aggiungere altre DPU e poi a eseguire nuovamente la query.

Se molte query sono accodate, significa che la capacità è utilizzata del tutto da altre query. Per ridurre la coda, effettua una delle seguenti operazioni:

- Aggiungi DPU alla tua prenotazione per aumentare la simultaneità delle query.
- Rimuovi i gruppi di lavoro dalla tua prenotazione per liberare spazio per altre query.

Per verificare l'eccessiva coda delle query, utilizza la CloudWatchmetrica del tempo di coda delle query Athena per i gruppi di lavoro inclusi nella prenotazione di capacità. Se il valore è superiore alla soglia preferita, puoi aggiungere DPU alla prenotazione della capacità.

Verifica della capacità inattiva

Per verificare la capacità inattiva, puoi ridurre il numero di DPU nella prenotazione o aumentare il corrispondente carico di lavoro e quindi osservare i risultati.

Come verificare la capacità inattiva

1. Esegui una di queste operazioni:
   - Riduci il numero di DPU nella prenotazione (riduci risorse disponibili)
   - Aggiungi gruppi di lavoro alla prenotazione (aumenta il carico di lavoro)
2. Viene utilizzato per misurare il tempo di coda CloudWatchdelle interrogazioni.
3. Se il tempo di coda aumenta oltre il livello desiderato, effettua una delle seguenti operazioni
   - Rimuovi i gruppi di lavoro
   - Aggiungi DPU alla tua prenotazione della capacità
4. Dopo ogni modifica, controlla le prestazioni e il tempo di coda delle query.
5. Continua a regolare il carico di lavoro e/o il numero di DPU per ottenere l'equilibrio desiderato.

Se non vuoi mantenere la capacità al di fuori del periodo di tempo preferito, puoi annullare la prenotazione e crearne un'altra in un secondo momento. Tuttavia, anche se hai annullato di recente la capacità di un'altra prenotazione, le richieste di nuova capacità non sono garantite; inoltre nuove prenotazioni richiedono tempo per essere create.

Strumenti per valutare i requisiti di capacità e i costi

Puoi utilizzare i seguenti servizi e funzionalità AWS per misurare l'utilizzo e i costi di Athena.

CloudWatchmetriche

Puoi configurare Athena per pubblicare metriche relative alle query su Amazon CloudWatch a livello di gruppo di lavoro. Dopo l'abilitazione dei parametri del gruppo di lavoro, nella console Athena, nella pagina dei dettagli del gruppo di lavoro, vengono visualizzati i parametri delle query del gruppo di lavoro.

Per informazioni sulle metriche di Athena pubblicate su CloudWatch e sulle relative dimensioni, vedere. Monitoraggio delle query Athena con metriche CloudWatch

CloudWatch metriche di utilizzo

Puoi utilizzare le metriche di CloudWatch utilizzo per fornire visibilità sul modo in cui il tuo account utilizza le risorse visualizzando l'utilizzo corrente del servizio su CloudWatch grafici e dashboard. Per Athena, le metriche di disponibilità dell'utilizzo corrispondono alle quote di AWS servizio per Athena. È possibile configurare gli allarmi che avvisano quando l'uso si avvicina a una quota di servizio.

Per ulteriori informazioni, consulta Monitoraggio dei parametri di utilizzo di Athena.

EventBridgeEventi Amazon

Puoi utilizzare Amazon Athena con Amazon EventBridge per ricevere notifiche in tempo reale sullo stato delle tue richieste. Quando una query inviata modifica lo stato, Athena pubblica un evento in EventBridge cui sono contenute informazioni sulla transizione dello stato della query. Puoi scrivere semplici regole per gli eventi che ti interessano e intraprendere azioni automatiche quando un evento corrisponde a una regola.

Per ulteriori informazioni, consulta le risorse seguenti.

- Monitoraggio delle query Athena con eventi Amazon EventBridge
- Che cos'è Amazon EventBridge?
- EventBridge Eventi Amazon

Tag

In Athena le prenotazioni della capacità supportano i tag. Un tag è formato da una chiave e da un valore. Per tenere traccia dei costi in Athena, puoi utilizzare i tag di allocazione dei AWS costi generati. AWS utilizza i tag di allocazione dei costi per organizzare i costi delle risorse nel rapporto costi e utilizzo. Ciò semplifica la categorizzazione e il monitoraggio dei costi di AWS . Per attivare i tag di allocazione dei costi per Athena, utilizza la console di AWS Billing and Cost Management.

Per ulteriori informazioni, consulta le risorse seguenti.

- Assegnazione di tag alle risorse Athena
- Attivazione dei tag di allocazione dei AWS costi generati
- Utilizzo dei tag per l'allocazione dei costi AWS

Creazione di prenotazioni della capacità

Per iniziare, crea una prenotazione della capacità con il numero di DPU richiesto, quindi assegni uno o più gruppi di lavoro che utilizzeranno tale capacità per le loro query. Puoi modificare la capacità in un secondo momento, in base alle tue esigenze, per fornire prestazioni più coerenti o gestire meglio i costi. Per informazioni sulla stima dei requisiti di capacità, consulta Determinazione dei requisiti di capacità.

> **Important**
>
> Le richieste di capacità non sono garantite e possono richiedere fino a 30 minuti per essere completate.

Creazione di una prenotazione della capacità

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.
3. Scegli Amministrazione, Prenotazioni della capacità.
4. Scegli Crea prenotazione della capacità.
5. Nella pagina Crea prenotazione della capacità, per Nome prenotazione della capacità, inserisci il nome. Il nome deve essere univoco, da 1 a 128 caratteri e comprendere solo i seguenti caratteri

A-z, A-z, 0-9, _(trattino basso), .(punto) e -(trattino). Dopo aver creato la prenotazione, non puoi modificare il nome.

6. Per DPU, scegli o inserisci il numero di unità di elaborazione dati (DPU) desiderato in incrementi di 4. Per ulteriori informazioni, consulta Informazioni su DPU.
7. (Facoltativo) Espandi l'opzione Tag, quindi seleziona Aggiungi nuovo tag per aggiungere una o più coppie chiave/valore personalizzate da associare alla risorsa di prenotazione della capacità. Per ulteriori informazioni, consulta Assegnazione di tag alle risorse Athena.
8. Scegli Rivedi.
9. Alla richiesta di conferma della creazione della capacità, conferma il numero di DPU e altre informazioni. Regione AWS Se accetti, seleziona Invia.

   Nella pagina dei dettagli, lo stato della prenotazione della capacità viene visualizzato come In sospeso. Quando la capacità della prenotazione è disponibile per eseguire le query, il relativo stato viene visualizzato come Attivo.

A questo punto, puoi aggiungere uno o più gruppi di lavoro alla tua prenotazione. Per le fasi, consulta Aggiunta di gruppi di lavoro a una prenotazione.

Gestione delle prenotazioni

Puoi visualizzare e gestire le prenotazioni della capacità nella pagina Prenotazioni della capacità. Puoi eseguire attività di gestione come l'incremento o la riduzione delle DPU, la modifica delle assegnazioni dei gruppi di lavoro e l'assegnazione di tag o l'annullamento delle prenotazioni.

Visualizzazione e gestione delle prenotazioni della capacità

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.
3. Seleziona Amministrazione, Prenotazioni della capacità.
4. Nella pagina delle prenotazioni della capacità, puoi eseguire le seguenti attività:

   - Per creare una prenotazione della capacità, scegli Crea prenotazione della capacità.
   - Utilizza la casella di ricerca per filtrare le prenotazioni per nome o numero di DPU.
   - Seleziona il menu a discesa dello stato per filtrare in base allo stato di prenotazione della capacità (ad esempio Attivo o Annullato). Per ulteriori informazioni sullo stato delle prenotazioni, consulta Comprensione dello stato di prenotazione.

- Per visualizzare i dettagli di una prenotazione della capacità, scegli il link relativo alla prenotazione. La pagina dei dettagli della prenotazione include opzioni per modificare la capacità, aggiungere gruppi di lavoro, rimuovere gruppi di lavoro e annullare la prenotazione.
- Per modificare una prenotazione (ad esempio, aggiungendo o rimuovendo DPU), seleziona il pulsante relativo alla prenotazione, quindi seleziona Modifica.
- Per annullare una prenotazione, seleziona il relativo pulsante, quindi seleziona Annulla.

Comprensione dello stato di prenotazione

La tabella seguente descrive i possibili valori di stato per una prenotazione della capacità.

Stato	Descrizione
Pending (In attesa)	Athena sta elaborando la tua richiesta di capacità. La capacità non è pronta per eseguire query.
Active (Attivo)	La capacità è disponibile per eseguire query.
Failed (Non riuscito)	La richiesta di capacità non è stata completata correttamente. Tieni presente che l'adempimento delle richieste di capacità non è garantito . Le prenotazioni non riuscite vengono conteggiate ai fini dei limiti DPU dell'account. Per interrompere l'utilizzo, è necessario annullare la prenotazione.
Aggiornamento in sospeso	Athena sta elaborando una modifica alla prenotazione. Questo stato si presenta ad esempio dopo aver modificato la prenotazione per aggiungere o rimuovere DPU.
Annullamento in corso	Athena sta elaborando una richiesta di cancellazione della prenotazi one. Le query ancora in esecuzione nei gruppi di lavoro che utilizzav ano la prenotazione possono essere completate, ma le altre query del gruppo di lavoro utilizzeranno la capacità (non fornita) on demand.
Annullato	L'annullamento della prenotazione della capacità è stato completato. Le prenotazioni annullate rimangono nella console per 45 giorni. Dopo 45 giorni, Athena eliminerà la prenotazione. Durante i 45 giorni, non puoi ridefinire o riutilizzare la prenotazione, ma puoi fare riferimento ai tag e visualizzarne i dettagli per riferimenti cronologici.

Stato	Descrizione
	Non è garantito che la capacità annullata sia prenotabile in futuro. La capacità non può essere trasferita a un'altra prenotazione, Account AWS oppure. Regione AWS

Informazioni sulle DPU attive e sulle DPU di destinazione

Nell'elenco delle prenotazioni della capacità nella console Athena, la prenotazione mostra due valori DPU: DPU attive e DPU di destinazione.

- DPU attive: il numero di DPU disponibili nella prenotazione per l'esecuzione di query. Ad esempio, se richiedi 100 DPU e la richiesta viene soddisfatta, nelle DPU attive viene visualizzato 100.
- DPU di destinazione: il numero di DPU verso cui è in corso il trasferimento della prenotazione. Le DPU di destinazione mostrano un valore diverso dalle DPU attive quando viene creata una prenotazione o un aumento o una diminuzione del numero di DPU sono in sospeso.

Ad esempio, dopo aver inviato una richiesta per creare una prenotazione con 24 DPU, lo stato della prenotazione sarà In sospeso, le DPU attive saranno 0 e le DPU di destinazione saranno 24.

Se hai una prenotazione con 100 DPU e la modifichi per richiedere un aumento di 20 DPU, loStato indicherà Aggiornamento in sospeso, le DPU attive saranno 100 e le DPU di destinazione saranno 120.

Se hai una prenotazione con 100 DPU e la modifichi per richiedere una riduzione di 20 DPU, loStato indicherà Aggiornamento in sospeso, le DPU attive saranno 100 e le DPU di destinazione saranno 80.

Durante queste transizioni, Athena lavora attivamente per acquisire o ridurre il numero di DPU in base alla richiesta dell'utente. Quando la DPU attiva diventa uguale alla DPU di destinazione, il numero obiettivo è stato raggiunto e nessuna modifica è in sospeso.

Per recuperare questi valori a livello di codice, puoi chiamare l'GetCapacityReservationazione API. L'API fa riferimento alle DPU attive e alle DPU di destinazione come `AllocatedDpus` e `TargetDpus`.

Argomenti

- Modifica di prenotazioni della capacità

- Aggiunta di gruppi di lavoro a una prenotazione
- Rimozione di un gruppo di lavoro da una prenotazione
- Annullamento di una prenotazione della capacità
- Eliminazione di una prenotazione della capacità

Modifica di prenotazioni della capacità

Dopo aver creato una prenotazione della capacità, puoi modificarne il numero di DPU e aggiungere o rimuovere i relativi tag personalizzati.

Per modificare una prenotazione della capacità

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.
3. Seleziona Amministrazione, Prenotazioni della capacità.
4. Nell'elenco delle prenotazioni della capacità, effettua una delle seguenti operazioni:
   - Seleziona il pulsante accanto alla prenotazione, quindi scegli Modifica.
   - Seleziona il link di prenotazione, quindi seleziona Modifica.
5. In DPU scegli o inserisci il numero di unità di elaborazione dati desiderato in incrementi di 4. Il numero minimo di DPU che puoi avere è 24. Per ulteriori informazioni, consulta Informazioni su DPU.

   > **Note**
   >
   > È possibile aggiungere DPU a una prenotazione della capacità esistente in qualsiasi momento. Tuttavia, non è possibile ridurre il numero di DPU fino a 1 ora dopo la creazione della prenotazione o l'aggiunta di DPU ad essa.

6. (Facoltativo) In Tag seleziona Rimuovi per rimuovere un tag oppure seleziona Aggiungi nuovo tag per aggiungere un nuovo tag.
7. Scegli Invia. La pagina dei dettagli della prenotazione mostra la configurazione aggiornata.

Aggiunta di gruppi di lavoro a una prenotazione

Dopo aver creato una prenotazione della capacità, puoi aggiungere fino a 20 gruppi di lavoro alla prenotazione. L'aggiunta di un gruppo di lavoro a una prenotazione comunica ad Athena quali query devono essere eseguite sulla capacità riservata dell'utente. Le query provenienti da gruppi di lavoro non associati a una prenotazione continuano a essere eseguite utilizzando il modello di prezzo predefinito per terabyte (TB) scansionato.

Quando una prenotazione include due o più gruppi di lavoro, le query provenienti da tali gruppi di lavoro possono utilizzare la capacità della prenotazione. Puoi aggiungere e rimuovere regole gruppi di lavoro in qualunque momento. Quando aggiungi o rimuovi gruppi di lavoro, le query in esecuzione non vengono interrotte.

Quando la prenotazione è in sospeso, le query dei gruppi di lavoro aggiunti continuano a essere eseguite utilizzando il modello di prezzo predefinito per terabyte (TB) scansionato fino a quando la prenotazione non diventa attiva.

Come aggiungere uno o più gruppi di lavoro alla prenotazione della capacità

1. Nella pagina dei dettagli della prenotazione della capacità, seleziona Aggiungi gruppi di lavoro.
2. Nella pagina Aggiungi gruppi di lavoro, seleziona i gruppi di lavoro che intendi aggiungere, quindi seleziona Aggiungi gruppi di lavoro. Non è possibile assegnare un gruppo di lavoro a più di una prenotazione.

   La pagina dei dettagli della prenotazione della capacità elenca i gruppi di lavoro aggiunti. Le query eseguite in tali gruppi di lavoro utilizzeranno la capacità che hai riservato quando la prenotazione è attiva.

Rimozione di un gruppo di lavoro da una prenotazione

Se non hai più bisogno di capacità da destinare a un gruppo di lavoro o intendi spostare un gruppo di lavoro nella sua prenotazione, puoi rimuoverlo in qualsiasi momento. La rimozione di un gruppo di lavoro da una prenotazione è un processo semplice. Dopo aver rimosso un gruppo di lavoro da una prenotazione, per impostazione predefinita le richieste del gruppo di lavoro rimosso utilizzano la capacità (non fornita) on demand e vengono fatturate in base ai terabyte (TB) scansionati.

Rimozione di uno o più gruppi di lavoro da una prenotazione

1. Nella pagina dei dettagli per la prenotazione della capacità, seleziona i gruppi di lavoro che intendi rimuovere.
2. Seleziona Rimuovi gruppi di lavoro. La richiesta Rimuovere i gruppi di lavoro? ti comunica che tutte le query attualmente attive termineranno prima che il gruppo di lavoro venga rimosso dalla prenotazione..
3. Scegli Rimuovi. La pagina dei dettagli della prenotazione della capacità mostra che i gruppi di lavoro rimossi non sono più presenti.

Annullamento di una prenotazione della capacità

Se non desideri più utilizzare una prenotazione della capacità, puoi annullarla. Le query ancora in esecuzione nei gruppi di lavoro che utilizzavano la prenotazione potranno terminare, ma le altre query del gruppo di lavoro non utilizzeranno più la prenotazione.

> **Note**
>
> Non è garantito che la capacità annullata sia prenotabile in futuro. La capacità non può essere trasferita su un'altra prenotazione, oppure. Account AWS Regione AWS

Annullamento di una prenotazione della capacità

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.
3. Seleziona Amministrazione, Prenotazioni della capacità.
4. Nell'elenco delle prenotazioni della capacità, effettua una delle seguenti operazioni:
   - Seleziona il pulsante accanto alla prenotazione, quindi seleziona Annulla.
   - Seleziona il link di prenotazione, quindi seleziona Annulla prenotazione della capacità.
5. Nella sezione Annullare la prenotazione della capacità? richiedi, inserisci annulla, quindi scegli Annulla prenotazione della capacità.

   Lo stato della prenotazione cambia in Annullamento e un banner sullo stato di avanzamento ti comunica che è in corso l'annullamento.

Una volta completato l'annullamento, la prenotazione della capacità rimane invariata, ma lo stato viene visualizzato come Annullato. La prenotazione verrà cancellata 45 giorni dopo l'annullamento. Durante i 45 giorni, non puoi ridefinire o riutilizzare una prenotazione che è stata annullata, ma puoi fare riferimento ai relativi tag e visualizzarla come riferimento cronologico.

Eliminazione di una prenotazione della capacità

Se desideri rimuovere tutti i riferimenti a una prenotazione della capacità annullata, puoi eliminare la prenotazione. Una prenotazione deve essere annullata prima di poter essere eliminata. Una prenotazione eliminata viene immediatamente rimossa dal tuo account e non può più essere referenziata, nemmeno dal relativo ARN.

Come eliminare una prenotazione della capacità

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.
3. Seleziona Amministrazione, Prenotazioni della capacità.
4. Nell'elenco delle prenotazioni della capacità, effettua una delle seguenti operazioni:
   - Seleziona il pulsante accanto alla prenotazione annullata, quindi scegli Operazioni, Elimina.
   - Seleziona il link di prenotazione, quindi seleziona Elimina.
5. Quando viene richiesto Eliminare la prenotazione della capacità?, seleziona Elimina.

   Un banner ti informa che la prenotazione della capacità è stata eliminata con successo. La prenotazione eliminata non appare più nell'elenco delle prenotazioni della capacità.

Policy IAM per le prenotazioni della capacità

Per controllare l'accesso alle prenotazione della capacità, utilizza le autorizzazioni IAM a livello di risorsa o le policy IAM basate sull'identità. Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

La seguente procedura è specifica di Athena.

Per informazioni specifiche su IAM, segui i collegamenti elencati alla fine di questa sezione. Per informazioni sulle policy JSON di esempio relative alle prenotazioni della capacità, consulta Esempi di policy di prenotazione della capacità.

Come utilizzare l'editor visivo nella console IAM per creare una policy di prenotazione della capacità

1. Accedi alla AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione sulla sinistra, selezionare Policies (Policy) e fare clic su Create Policy (Crea policy).
3. Nella scheda Visual editor (Editor visivo), selezionare Choose a Service (Scegli un servizio). Quindi selezionare Athena per aggiungerlo alla policy.
4. Scegliere Select actions (Seleziona operazioni), quindi scegliere le operazioni da aggiungere alla policy. L'editor visivo mostra le operazioni disponibili in Athena. Per ulteriori informazioni, consulta Operazioni, risorse e chiavi di condizione per Amazon Athena nella Documentazione di riferimento per l'autorizzazione ai servizi.
5. Seleziona add actions per immettere un'operazione specifica oppure usa i caratteri jolly (*) per specificare più operazioni.

   Come impostazione predefinita, la policy che si sta creando utilizza le operazioni selezionate. Se si selezionano una o più operazioni che supportano le autorizzazioni a livello di risorsa per la risorsa `capacity-reservation` in Athena, l'editor elenca la risorsa `capacity-reservation`
6. Seleziona Risorse per specificare le prenotazioni della capacità specifiche della tua policy. Per le policy JSON di esempio relative alla prenotazione della capacità, consulta Esempi di policy di prenotazione della capacità.
7. Specificare la risorsa `capacity-reservation` come segue:

   ```
   arn:aws:athena:<region>:<user-account>:capacity-reservation/<capacity-reservation-name>
   ```
8. Scegliere Review policy (Rivedi policy) e digitare i valori per Name (Nome) e Description (Descrizione) (facoltativa) per la policy che si sta creando. Esaminare il riepilogo della policy per accertarsi di disporre delle autorizzazioni desiderate.
9. Selezionare Create policy (Crea policy) per salvare la nuova policy.
10. Allega questa policy basata su un'identità a un utente, un gruppo o un ruolo.

Per ulteriori informazioni, consulta gli argomenti seguenti nella Referenza sull'autorizzazione del servizio e nella Guida per l'utente di IAM:

- Operazioni, risorse e chiavi di condizione per Amazon Athena
- Creazione di policy con l'editor visivo
- Aggiunta e rimozione delle policy IAM
- Controllo dell'accesso alle risorse

Per le policy JSON di esempio relative alla prenotazione della capacità, consulta Esempi di policy di prenotazione della capacità.

Per un elenco completo delle operazioni Amazon Athena, consulta i nomi delle operazioni API nella documentazione di riferimento dell'API Amazon Athena.

Esempi di policy di prenotazione della capacità

Questa sezione include policy di esempio che puoi utilizzare per abilitare varie operazioni sulle prenotazioni della capacità. Ogni volta che si utilizzano le policy IAM, assicurati di seguire le best practice IAM. Per ulteriori informazioni, consulta Best Practice di sicurezza in IAM nella Guida per l'utente di IAM.

Una prenotazione della capacità è una risorsa IAM gestita da Athena. Pertanto, se la policy della prenotazione della capacità utilizza operazioni che accettano `capacity-reservation` come input, è necessario specificare l'ARN della prenotazione della capacità nel modo seguente:

```
"Resource": [arn:aws:athena:<region>:<user-account>:capacity-reservation/<capacity-reservation-name>]
```

*<capacity-reservation-name>*Dov'è il nome della tua prenotazione della capacità. Ad esempio, per una prenotazione della capacità denominata `test_capacity_reservation`, specificala come risorsa nel modo seguente:

```
"Resource": ["arn:aws:athena:us-east-1:123456789012:capacity-reservation/
test_capacity_reservation"]
```

Per un elenco completo delle operazioni Amazon Athena, consulta i nomi delle operazioni API nella documentazione di riferimento dell'API Amazon Athena. Per ulteriori informazioni sulle policy IAM, consulta Creazione di policy con l'editor visivo nella Guida per l'utente di IAM.

- Example policy to list capacity reservations
- Example policy for management operations

Example Esempio di policy per elencare le prenotazioni della capacità

La policy seguente consente a tutti gli utenti di elencare tutte le prenotazioni di capacità.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:ListCapacityReservations"
            ],
            "Resource": "*"
        }
    ]
}
```

Example Esempio di policy per le operazioni di gestione

La seguente policy consente a un utente di creare, annullare, ottenere dettagli e aggiornare la prenotazione della capacità `test_capacity_reservation`. La policy consente inoltre a un utente di assegnare `workgroupA` e `workgroupB` a `test_capacity_reservation`.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect": "Allow",
         "Action": [
             "athena:CreateCapacityReservation",
             "athena:GetCapacityReservation",
             "athena:CancelCapacityReservation",
             "athena:UpdateCapacityReservation",
             "athena:GetCapacityAssignmentConfiguration",
             "athena:PutCapacityAssignmentConfiguration"
         ],
         "Resource": [
             "arn:aws:athena:us-east-1:123456789012:capacity-
reservation/test_capacity_reservation",
```

```
            "arn:aws:athena:us-east-1:123456789012:workgroup/workgroupA",
            "arn:aws:athena:us-east-1:123456789012:workgroup/workgroupB"
        ]
    }
  ]
}
```

API di prenotazione della capacità in Athena

L'elenco seguente contiene collegamenti di riferimento alle operazioni API relative alla prenotazione della capacità in Athena. Per le strutture dei dati e altre operazioni dell'API Athena, consulta la documentazione di riferimento dell'API Amazon Athena.

- CancelCapacityReservation
- CreateCapacityReservation
- GetCapacityAssignmentConfiguration
- GetCapacityReservation
- ListCapacityReservations
- PutCapacityAssignmentConfiguration
- UpdateCapacityReservation

Ottimizzazione delle prestazioni in Athena

Questo argomento fornisce informazioni generali e suggerimenti specifici per migliorare le prestazioni delle tue query Athena e su come rimediare agli errori relativi ai limiti e all'utilizzo delle risorse.

Quote del servizio

Athena applica delle quote per metriche come il tempo di esecuzione delle query, il numero di query simultanee in un account e i tassi di richiesta delle API. Per ulteriori informazioni su queste quote, consulta Service Quotas (Quote di Servizio). Il superamento di queste quote comporta l'esito negativo di una query, al momento dell'invio o durante l'esecuzione della query.

Molti dei suggerimenti per l'ottimizzazione delle prestazioni contenuti in questa pagina possono contribuire a ridurre il tempo di esecuzione delle query. L'ottimizzazione libera spazio in modo da consentire l'esecuzione di più query entro la quota di simultaneità e impedisce che le query vengano annullate perché in esecuzione da troppo tempo.

Le quote sul numero di query e richieste API simultanee si intendono per e. Account AWS Regione AWS Ti consigliamo di eseguire un carico di lavoro per Account AWS (o di utilizzare prenotazioni di capacità assegnate separatamente) per evitare che i carichi di lavoro competano per la stessa quota.

Se esegui due carichi di lavoro nello stesso account, uno dei carichi di lavoro può eseguire un picco di query. Ciò può comportare la limitazione o il blocco dell'esecuzione delle query del carico di lavoro rimanente. Per evitare ciò, puoi spostare i carichi di lavoro in account separati per assegnare a ciascun carico di lavoro una relativa quota di simultaneità. La creazione di una prenotazione della capacità assegnata uno o entrambi i carichi di lavoro consente di raggiungere lo stesso obiettivo.

Quote in altri servizi

Quando Athena esegue una query, può chiamare altri servizi che applicano le quote. Durante l'esecuzione delle query, Athena può effettuare chiamate API verso Amazon S3 e AWS altri servizi come IAM e. AWS Glue Data Catalog AWS KMS Se si utilizzano interrogazioni federate, viene chiamata anche Athena. AWS Lambda Tutti questi servizi hanno rispettivi limiti e quote che possono essere superati. L'esecuzione di una query, quando rileva errori dovuti a questi servizi, non va a buon fine e include l'errore del servizio di origine. Gli errori rimediabili vengono ritentati, ma le query possono comunque avere esito negativo se il problema non si risolve automaticamente in tempo. Assicurati di leggere attentamente i messaggi degli errori per comprendere se questi provengono da Athena o da un altro servizio. In questo documento sono trattati alcuni errori rilevanti.

Per ulteriori informazioni su come rimediare agli errori provocati dalle service quotas di Amazon S3, consulta Come evitare di avere un numero eccessivo di file più avanti in questo documento. Per ulteriori informazioni sull'ottimizzazione delle prestazioni di Amazon S3, consulta Modelli di concezione delle best practice: ottimizzazione delle prestazioni di Amazon S3 nella Guida per l'utente di Amazon S3.

Limiti delle risorse

Athena esegue le query in un motore di query distribuito. Quando invii una query, il pianificatore di query del motore Athena stima la capacità di calcolo richiesta per l’esecuzione della query e prepara di conseguenza un cluster di nodi di calcolo. Alcune query, come le query DDL, sono eseguite su un solo nodo. Le query complesse su set di dati di grandi dimensioni sono eseguite su cluster molto più grandi. I nodi sono uniformi, con le stesse configurazioni di memoria, CPU e disco. Athena si dimensiona orizzontalmente, non verticalmente, per elaborare query più impegnative.

Talvolta le richieste di una query superano le risorse disponibili al cluster che esegue la query. Quando ciò accade, la query non va a buon fine e viene visualizzato l'errore La query ha esaurito le risorse con questo fattore di scala.

La risorsa più comunemente esaurita è la memoria, ma in rari casi anche lo spazio su disco può essere esaurito. In genere si presentano errori di memoria quando il motore esegue una funzione join o finestra, ma possono presentarsi anche durante conteggi e aggregazioni distinti.

Una query, che non sia riuscita e abbia presentato l'errore di "risorse esaurite", potrebbe avere esito positivo quando la esegui nuovamente. L'esecuzione delle query non è deterministica. Fattori quali il tempo necessario al caricamento di dati e il modo di distribuzione sui nodi di set di dati intermedi possono comportare un diverso utilizzo delle risorse. Ad esempio, immagina una query che unisce due tabelle e che presenta una forte asimmetria nella distribuzione dei valori per la condizione di join. Una query di questo tipo può avere esito positivo per la maggior parte del tempo, ma talvolta può non riuscire quando i valori più comuni risultano essere elaborati dallo stesso nodo.

Per evitare che le query superino le risorse disponibili, segui i consigli per ottimizzare le prestazioni cui si è accennato in questo documento. In particolare, relativamente ai consigli sulle modalità di ottimizzazione delle query che esauriscono le risorse disponibili, consulta Ottimizzazione dei join, Ottimizzazione delle funzioni delle finestre e Ottimizzazione delle query mediante approssimazioni.

Tecniche di ottimizzazione delle query

Attieniti alle tecniche di ottimizzazione delle query descritte in questa sezione per velocizzare l'esecuzione delle query o per rimediare al problema delle query che superano i limiti di risorse in Athena.

Ottimizzazione dei join

Le strategie per eseguire join in un motore di query distribuito sono svariate. Due delle più comuni sono gli hash join distribuiti e le query con condizioni di join complesse.

Hash join distribuito

Il tipo più comune di join utilizza come condizione di join un confronto di uguaglianza. Athena esegue questo tipo di join come hash join distribuito.

In un hash join distribuito, il motore crea una tabella di ricerca (tabella hash) da uno dei lati del join. Questo lato è denominato il lato build. I record del lato build sono distribuiti tra i nodi. Ogni nodo costruisce una tabella di ricerca per il relativo sottoinsieme. L'altro lato del join, denominato lato probe, è quindi trasmesso in streaming attraverso i nodi. I record dal lato probe sono distribuiti sui nodi nello stesso modo del lato build. Ciò consente a ciascun nodo di eseguire il join cercando i record corrispondenti nella relativa tabella di ricerca.

Quando le tabelle di ricerca create dal lato build del join non rientrano nella memoria, le query possono non andare a buon fine. Anche se la dimensione totale del lato di build è inferiore alla memoria disponibile, se la distribuzione dei record presenta un'asimmetria significativa, le query possono comunque avere esito negativo. In un caso estremo, è possibile che tutti i record abbiano per la condizione di join lo stesso valore e che sia richiesto che vengano immessi nella memoria su un singolo nodo. Anche una query con asimmetria minore può avere esito negativo se un set di valori viene inviato allo stesso nodo e la somma dei valori supera la quantità superiore di memoria disponibile. I nodi hanno la capacità di versare i record su disco, tuttavia il versamento rallenta l'esecuzione delle query e può essere insufficiente per evitare che la query abbia esito negativo.

Athena tenta di riordinare i join per utilizzare la relazione più grande come lato probe e la relazione più piccola come lato build. Tuttavia Athena, poiché non gestisce i dati nelle tabelle, dispone di informazioni limitate e spesso deve presumere che la prima tabella sia la più grande e la seconda tabella sia la più piccola.

Quando scrivi join con condizioni di join basate sull'uguaglianza, presumi che la tabella a sinistra della parola chiave `JOIN` sia il lato probe e la tabella a destra sia il lato build. Assicurati che la tabella a destra, il lato build, sia la tabella di dimensioni inferiori. Se non è possibile ridurre le dimensione del lato build del join affinché rientri nella memoria, considera la possibile di eseguire più query che uniscano sottoinsiemi della tabella build.

Altri tipi di join

Le query con condizioni di join complesse (ad esempio, query che utilizzano `LIKE` o altri operatori) sono spesso impegnative dal punto di vista computazionale. > Nel peggiore dei casi, tutti i record di un lato del join devono essere confrontati con tutti i record sull'altro lato del join. Poiché il tempo di esecuzione aumenta con il quadrato del numero di record, si rischia che tali query superino il tempo di esecuzione massimo.

Per sapere in in anticipo in che modo Athena eseguirà la query, puoi utilizzare la dichiarazione `EXPLAIN`. Per ulteriori informazioni, consulta Utilizzo di EXPLAIN e EXPLAIN ANALYZE in Athena e Capire i risultati dell'istruzione EXPLAIN di Athena.

Ottimizzazione delle funzioni delle finestre

Le funzioni delle finestre, poiché richiedono molte risorse, possono rallentare o addirittura far fallire le query con il messaggio La query ha esaurito le risorse con questo fattore di scala. Le funzioni finestra conservano in memoria tutti i record su cui operano per calcolarne il risultato. Quando la finestra è di dimensioni molto grandi, la funzione finestra può esaurire la memoria.

Per assicurarti che le query vengano eseguite entro i limiti di memoria disponibili, riduci le dimensioni delle finestre su cui operano le funzioni finestra. A tale scopo, puoi aggiungere una clausola `PARTITIONED BY` o restringere l'ambito delle clausole di partizionamento esistenti.

Utilizza invece funzioni diverse dalle finestre

A volte le query con le funzioni finestra possono essere riscritte senza funzioni finestra. Ad esempio, invece di utilizzare `row_number` per trovare i record top `N`, puoi utilizzare `ORDER BY` e `LIMIT`. Anziché utilizzare `row_number` o `rank` per deduplicare i record, puoi utilizzare funzioni aggregate come max_by, min_by e arbitrary.

Ad esempio, considera di disporre di un set di dati con aggiornamenti da un sensore. Il sensore comunica regolarmente lo stato della batteria e include alcuni metadati quali la posizione. Se vuoi conoscere lo stato più recente della batteria di ogni sensore e la sua posizione, puoi utilizzare questa query:

```
SELECT sensor_id,
       arbitrary(location) AS location,
       max_by(battery_status, updated_at) AS battery_status
FROM sensor_readings
GROUP BY sensor_id
```

Poiché i metadati, ad esempio la posizione, sono gli stessi per ogni record, puoi utilizzare la funzione `arbitrary` per selezionare qualsiasi valore dal gruppo.

Per conoscere lo stato più recente della batteria, puoi usare la funzione `max_by`. La funzione `max_by` seleziona il valore di una colonna dal record in cui è stato trovato il valore massimo di un'altra colonna. In questo caso, la funzione restituisce lo stato della batteria del record con l'orario dell'ultimo aggiornamento all'interno del gruppo. Questa query viene eseguita più velocemente e utilizza meno memoria rispetto a una query equivalente con una funzione finestra.

Ottimizzazione delle aggregazioni

Athena, quando esegue un'aggregazione, distribuisce i record tra i nodi worker utilizzando le colonne nella clausola `GROUP BY`. Per rendere il più efficiente possibile l'abbinamento dei record ai gruppi, i nodi cercano di conservare i record in memoria ma, se necessario, li trasferiscono su disco.

Inoltre, nelle clausole `GROUP BY` è consigliabile evitare di includere colonne ridondanti. Poiché un numero inferiore di colonne richiede una quantità di memoria inferiore, è più efficiente una query che

descrive un gruppo con un numero inferiore di colonne. Le colonne numeriche utilizzano inoltre una quantità di memoria inferiore rispetto alle stringhe. Ad esempio, quando aggreghi un set di dati che ha sia un ID di categoria numerico sia un nome di categoria, nella clausola `GROUP BY` utilizza solo la colonna ID di categoria.

A volte le query includono colonne nella clausola `GROUP BY` per ovviare al fatto che una colonna debba far parte della clausola `GROUP BY` o di un'espressione aggregata. Se questa regola non viene seguita, puoi ricevere un messaggio di errore come il seguente:

EXPRESSION_NOT_AGGREGATE: riga 1:8: 'category' deve essere un'espressione aggregata o apparire nella clausola GROUP BY

Per evitare di dover aggiungere colonne ridondanti alla `GROUP BY` clausola, puoi utilizzare la funzione arbitraria, come nell'esempio che segue.

```
SELECT country_id,
       arbitrary(country_name) AS country_name,
       COUNT(*) AS city_count
FROM world_cities
GROUP BY country_id
```

La funzione `ARBITRARY` restituisce un valore arbitrario dal gruppo. La funzione è utile quando sai che tutti i record del gruppo hanno lo stesso valore per una colonna, ma il valore non identifica il gruppo.

Ottimizzazione delle query top N

La clausola `ORDER BY` restituisce i risultati di una query disponendoli in ordine. Athena utilizza l'ordinamento distribuito per eseguire l'operazione di ordinamento in parallelo su più nodi.

Se non è strettamente necessario ordinare i risultati, evita di aggiungere una clausola `ORDER BY`. Inoltre, se non sono strettamente necessarie, evita di aggiungere informazioni `ORDER BY` alle query interne. In molti casi, il pianificatore di query può rimuovere l'ordinamento ridondante, ma ciò non è garantito. Un'eccezione a questa regola è prevista se una query interna esegue un'operazione top `N`, ad esempio la ricerca dei valori `N` più recenti o dei valori `N` più comuni.

Athena, quando vede `ORDER BY` insieme a `LIMIT`, capisce che stai eseguendo una query top `N` e mette in atto operazioni dedicate di conseguenza.

> **Note**
>
> Sebbene Athena sia spesso in grado di rilevare anche funzioni finestra come `row_number` che utilizza top `N`, è consigliabile la versione più semplice che utilizza `ORDER BY` e `LIMIT`. Per ulteriori informazioni, consulta Ottimizzazione delle funzioni delle finestre.

Come includere solo le colonne obbligatorie

Se una colonna non è strettamente necessaria, non includerla nella tua query. Minore è la quantità di dati che una query deve elaborare, maggiore sarà la velocità di esecuzione. Ciò riduce sia la quantità di memoria richiesta sia la quantità di dati da inviare tra i nodi. Se utilizzi un formato di file colonnare, la riduzione del numero di colonne fa comporta una diminuzione anche della quantità di dati letti da Amazon S3.

In Athena non sono previsti limiti specifici al numero di colonne in un risultato, tuttavia le modalità di esecuzione delle query limita la possibile dimensione combinata delle colonne. La dimensione combinata delle colonne include i relativi nomi e tipi.

Ad esempio, l'errore seguente è causato da una relazione che supera il limite di dimensione di un descrittore di relazione:

GENERIC_INTERNAL_ERROR: io.airlift.bytecode. CompilationException

Per risolvere il problema, riduci il numero di colonne nella query o crea sottoquery e utilizza `JOIN` che recuperi una quantità inferiore di dati. Se hai query che eseguono `SELECT *` nella query più esterna, dovresti cambiare il `*` a un elenco che presenti solo le colonne di cui hai bisogno.

Ottimizzazione delle query mediante approssimazioni

Athena supporta funzioni di aggregazione di approssimazione per il conteggio di valori distinti, dei valori più frequenti, di percentili (comprese le mediane approssimative) e per la creazione di istogrammi. Utilizza queste funzioni ogni volta che non hai bisogno di valori esatti.

A differenza delle operazioni `COUNT(DISTINCT col)`, approx_distinct utilizza molta meno memoria e funziona più velocemente. Allo stesso modo, l'utilizzo di numeric_histogram, anziché di histogram, comporta metodi approssimativi e quindi una quantità di memoria inferiore.

Ottimizzazione LIKE

Puoi usare `LIKE` per trovare stringhe corrispondenti, ma per le stringhe lunghe richiede molto calcolo. La funzione regexp_like è nella maggior parte dei casi un'alternativa più veloce e offre anche maggiore flessibilità.

Spesso puoi ottimizzare una ricerca ancorando la sottostringa che stai cercando. Ad esempio, se vuoi cercare un prefisso, è preferibile utilizzare '*substr*%' anziché '%*substr*%'. Oppure, se stai usando `regexp_like`, '^*substr*'.

Utilizzo di UNION ALL anziché di UNION

`UNION ALL` e `UNION` sono due modi di combinare i risultati di due query in un unico risultato. `UNION ALL` concatena i record della prima query con la seconda e `UNION` fa lo stesso, ma rimuove anche i duplicati. `UNION` deve elaborare tutti i record e trovare i duplicati, il che richiede molta memoria e calcolo, ma `UNION ALL` è un'operazione relativamente rapida. A meno che non sia necessario deduplicare i record, utilizza `UNION ALL` per ottenere prestazioni ottimali.

Utilizzo di UNLOAD per set di risultati di grandi dimensioni

Quando prevedi che i risultati di una query siano di grandi dimensioni (ad esempio, decine di migliaia di righe o più), utilizza UNLOAD per esportare i risultati. Nella maggior parte dei casi, questa operazione è più veloce rispetto all'esecuzione di una normale query e l'utilizzo di `UNLOAD` offre anche maggior controllo sull'output.

Al termine dell'esecuzione di una query, Athena archivia su Amazon S3 il risultato in un unico file CSV non compresso. Questa operazione richiede più tempo di `UNLOAD`, non solo perché il risultato non è compresso, ma anche perché l'operazione non può essere parallelizzata. Al contrario, `UNLOAD` scrive i risultati direttamente dai nodi worker e utilizza appieno il parallelismo del cluster di elaborazione. Inoltre, puoi configurare `UNLOAD` in modo che scriva i risultati in formato compresso e in altri formati di file come JSON e Parquet.

Per ulteriori informazioni, consulta UNLOAD.

Utilizzo di CTAS o Glue ETL per materializzare le aggregazioni utilizzate di frequente

La "materializzazione" di una query è un modo per accelerare le prestazioni delle query archiviando i risultati di query complesse precalcolate (ad esempio aggregazioni e join) per riutilizzarli nelle query successive.

Se molte delle query includono gli stessi join e le stesse aggregazioni, puoi materializzare la sottoquery comune come una nuova tabella e quindi eseguire query su quella tabella. Puoi creare la

nuova tabella con Creazione di una tabella dai risultati delle query (CTAS) o con uno strumento ETL dedicato come Glue ETL.

Ad esempio, considera di avere una dashboard con widget che mostrano diversi aspetti di un set di dati di ordini. Ogni widget ha una propria query, ma tutte queste condividono gli stessi join e filtri. Una tabella di ordini viene unita a una tabella di elementi, inoltre è presente un filtro per mostrare solo gli ultimi tre mesi. Se identifichi le funzionalità comuni di queste query, puoi creare una nuova tabella utilizzabile dai widget. Ciò riduce la duplicazione e migliora le prestazioni. Lo svantaggio è che devi mantenere aggiornata la nuova tabella.

Riutilizzo dei risultati query

Accade spesso che la stessa query venga eseguita più volte in un breve periodo. Ad esempio, ciò può succedere quando più persone aprono la stessa dashboard di dati. Quando esegui una query, puoi fare in modo che Athena riutilizzi i risultati calcolati in precedenza. Specifica l'età massima dei risultati da riutilizzare. Se la stessa query è stata eseguita in precedenza entro tale intervallo di tempo, Athena restituisce tali risultati anziché eseguire nuovamente la query. Per ulteriori informazioni, consulta Riutilizzo dei risultati delle query nella Guida per l'utente di Amazon Athena e il blog Riduci i costi e migliora le prestazioni delle query con Amazon Athena Query Result Reuse nel Blog sui Big Data di AWS .

Tecniche di ottimizzazione query

Le prestazioni dipendono non solo dalle query, ma anche, in misura significativa, dal modo di organizzazione del set di dati nonché dal formato di file e dalla compressione utilizzati dal set di dati.

Come partizionare i dati

Il partizionamento divide la tabella in parti e mantiene insieme i dati correlati in base a proprietà quali data, paese o regione. Le chiavi di partizione fungono da colonne virtuali. Al momento della creazione della tabella, definisci le chiavi di partizione, quindi utilizzale per filtrare le query. Quando filtri in base alle colonne delle chiavi di partizione, vengono letti solo i dati delle partizioni corrispondenti. Ad esempio, se il set di dati è partizionato in base alla data e la query ha un filtro corrispondente solo all'ultima settimana, vengono letti solo i dati dell'ultima settimana. Per ulteriori informazioni sulle partizioni, consulta Partizionamento dei dati in Athena.

Scegli le chiavi di partizione che supporteranno le query

Poiché il partizionamento ha un impatto significativo sulle prestazioni delle query, quando pianifichi il set di dati e le tabelle, assicurati di considerare attentamente le modalità di partizionamento. Un

numero eccessivo di chiavi di partizione può comportare set di dati frammentati con un numero eccessivo di file e file di dimensioni troppo piccole. Viceversa, se il numero delle chiavi di partizione è troppo basso o il partizionamento è inesistente, le query eseguono la scansione di una quantità di dati superiore a quella necessaria.

Evita di eseguire l'ottimizzazione delle query rare

Una buona strategia consiste nell'eseguire l'ottimizzazione delle query più comuni ed evitare l'ottimizzazione delle query rare. Ad esempio, se le tue query riguardano intervalli di giorni, non partizionarle in base all'ora, anche se alcune query filtrano a quel livello. Se i dati presentano una colonna con timestamp granulare, le query rare che filtrano per ora possono utilizzare la colonna timestamp. Anche se in casi rari viene scansionata una quantità di dati leggermente superiore a quella necessaria, ridurre le prestazioni complessive a vantaggio dei casi rari di solito non è un buon compromesso.

Per ridurre la quantità di dati che le query devono scansionare e quindi migliorare le prestazioni, utilizza un formato di file colonnare e mantieni i record ordinati. Invece di partizionarli per ora, mantieni i record ordinati per timestamp. Per le query su finestre temporali più brevi, l'ordinamento per timestamp è efficiente quasi quanto il partizionamento per ora. Inoltre, l'ordinamento per timestamp in genere non pregiudica le prestazioni delle query su finestre temporali contate in giorni. Per ulteriori informazioni, consulta Utilizzo di formati di file colonnari.

Tieni presente che le query su tabelle con decine di migliaia di partizioni offrono prestazioni migliori se sono presenti predicati su tutte le chiavi di partizione. Questo è un motivo in più per pianificare lo schema di partizionamento per le query più comuni. Per ulteriori informazioni, consulta Partizioni di query per uguaglianza.

Utilizzo della proiezione di partizioni

La proiezione delle partizioni è una funzionalità di Athena che memorizza le informazioni sulle partizioni non in AWS Glue Data Catalog, ma come regole nelle proprietà della tabella in. AWS Glue Athena, quando pianifica una query su una tabella configurata con la proiezione delle partizioni, legge le regole di proiezione delle partizioni della tabella. Athena calcola le partizioni da leggere in memoria in base alla query e alle regole invece di cercare le partizioni in AWS Glue Data Catalog.

Oltre a semplificare la gestione delle partizioni, la proiezione delle partizioni può migliorare le prestazioni dei set di dati che presentano un numero elevato di partizioni. Quando una query include intervalli anziché valori specifici per le chiavi di partizione, la ricerca delle partizioni corrispondenti nel catalogo richiede tanto più tempo quanto maggiori sono le partizioni. Con la proiezione delle

partizioni, il filtro può essere calcolato nella memoria senza andare al catalogo e può essere molto più veloce.

In determinate circostanze, la proiezione delle partizioni può comportare prestazioni peggiori. Un esempio in tal senso è la situazione che si presenta quando una tabella è "sparsa". Una tabella sparsa non contiene dati per tutte le permutazioni dei valori delle chiavi di partizione descritti dalla configurazione di proiezione delle partizioni. Con una tabella sparsa, il set di partizioni calcolato dalla query e la configurazione di proiezione delle partizioni sono tutti elencati su Amazon S3, anche quando non contengono dati.

Quando usi la proiezione delle partizioni, assicurati di includere i predicati su tutte le chiavi delle partizioni. Restringi l'ambito dei valori possibili per evitare elenchi Amazon S3 non necessari. Immagina una chiave di partizione con un intervallo di un milione di valori e una query che non abbia filtri su quella chiave di partizione. Per eseguire la query, Athena deve eseguire almeno un milione di operazioni sugli elenchi Amazon S3. Le query più veloci quelle eseguite su valori specifici, indipendentemente dal fatto che utilizzi la proiezione delle partizioni o memorizzi le informazioni delle partizioni nel catalogo. Per ulteriori informazioni, consulta Partizioni di query per uguaglianza.

Quando configuri una tabella per la proiezione delle partizioni, assicurati che gli intervalli specificati siano ragionevoli. Se una query non include un predicato su una chiave di partizione, tutti i valori nell'intervallo vengono utilizzati per tale chiave. Se il set di dati è stato creato in una data specifica, per qualsiasi intervallo di date utilizza quella data come punto di inizio. Utilizza `NOW` come fine degli intervalli di data. Evita gli intervalli numerici con un numero elevato di valori e considera invece l'utilizzo del tipo injected.

Per maggiori informazioni sulla proiezione delle partizioni, consulta Proiezione delle partizioni con Amazon Athena.

Utilizzo degli indici di partizioni

Gli indici di partizione sono una funzionalità AWS Glue Data Catalog che migliora le prestazioni di ricerca delle partizioni per le tabelle con un numero elevato di partizioni.

L'elenco delle partizioni nel catalogo è simile a una tabella in un database relazionale. La tabella contiene colonne per le chiavi delle partizioni e una colonna aggiuntiva per la posizione delle partizioni. Quando si esegue una query su una tabella partizionata, i percorsi delle partizioni vengono ricercati mediante la scansione di questa tabella.

Proprio come con i database relazionali, puoi aumentare le prestazioni delle query aggiungendo indici. Puoi aggiungere più indici per supportare diversi modelli di query. L'indice di AWS Glue Data

Catalog partizione supporta sia gli operatori di uguaglianza che quelli di confronto>, come, >= e combinati con l'operatore. < AND Per ulteriori informazioni, consulta Lavorare con gli indici delle partizioni AWS Glue nella Guida per gli AWS Glue sviluppatori e Migliorare le prestazioni delle query di Amazon Athena utilizzando gli indici delle AWS Glue Data Catalog partizioni nel Big Data Blog.AWS

Utilizzo costante di STRING come tipo per le chiavi di partizione

Quando esegui una query sulle chiavi delle partizioni, ricorda che in Athena le chiavi delle partizioni devono essere di tipo `STRING` affinché in AWS Glue sia eseguito il pushdown del filtraggio delle partizioni. Se il numero di partizioni non è basso, l'utilizzo di altri tipi può comportare prestazioni peggiori. Se i valori delle chiavi di partizione sono simili a date o numeri, convertili nel tipo appropriato della query.

Rimozione delle partizioni vecchie e vuote

Se rimuovi dati da una partizione su Amazon S3 (ad esempio, utilizzando il ciclo di vita di Amazon S3), devi rimuovere anche la voce della partizione da AWS Glue Data Catalog. Durante la pianificazione delle query, su Amazon S3 viene elencata qualsiasi partizione corrispondente alla query. Se hai molte partizioni vuote, il sovraccarico di elencare queste partizioni può essere dannoso.

Inoltre, se disponi di molte migliaia di partizioni, valuta la possibilità di rimuovere i metadati delle partizioni dei dati vecchi che non sono più rilevanti. Ad esempio, se le query non scansionano mai dati più vecchi di un anno, puoi rimuovere periodicamente i metadati delle partizioni più vecchie. Se il numero di partizioni aumenta fino a decine di migliaia, la rimozione delle partizioni inutilizzate può velocizzare le query che non includono predicati su tutte le chiavi delle partizioni. Per informazioni sull'inclusione dei predicati su tutte le chiavi delle partizioni nelle query, consulta Partizioni di query per uguaglianza.

Partizioni di query per uguaglianza

Le query che includono i predicati di uguaglianza su tutte le chiavi delle partizioni sono eseguite più velocemente perché possono essere caricati direttamente i metadati delle partizioni. Evita le query in cui una o più chiavi di partizione sono prive di predicato o in cui il predicato seleziona un intervallo di valori. Per tali query, è necessario filtrare l'elenco di tutte le partizioni per trovare i valori corrispondenti. Per la maggior parte delle tabelle, il sovraccarico è minimo, ma per le tabelle con decine di migliaia o più di partizioni, il sovraccarico può diventare significativo.

Se non è possibile riscrivere le query per filtrare le partizioni in base all'uguaglianza, puoi provare la proiezione delle partizioni. Per ulteriori informazioni, consulta Utilizzo della proiezione di partizioni.

Come evitare di usare MSCK REPAIR TABLE per la manutenzione delle partizioni

Poiché può richiedere molto tempo, aggiunge solo nuove partizioni e non rimuove le vecchie partizioni, l'esecuzione `MSCK REPAIR TABLE` non è un modo efficiente per gestire le partizioni (consulta Considerazioni e limitazioni).

Le partizioni sono gestite in modo migliore utilizzando manualmente le API di AWS Glue Data Catalog, ALTER TABLE ADD PARTITION, o i crawler di AWS Glue. In alternativa, puoi utilizzare la proiezione delle partizioni, che rimuove del tutto il bisogno di gestire le partizioni. Per ulteriori informazioni, consulta Proiezione delle partizioni con Amazon Athena.

Come verificare che le query siano compatibili con lo schema di partizionamento

Puoi verificare in anticipo quali partizioni saranno scansionate da una query utilizzando la dichiarazione `EXPLAIN`. Utilizza la parola chiave `EXPLAIN` come prefisso della query, quindi cerca il frammento di origine (ad esempio `Fragment 2 [SOURCE]`) per ogni tabella nella parte inferiore dell'output `EXPLAIN`. Cerca le assegnazioni in cui il lato destro è definito come chiave di partizione. La riga sottostante include un elenco di tutti i valori per la chiave di partizione da scansionare durante l'esecuzione della query.

Ad esempio, supponi di avere una query su una tabella con una chiave di partizione `dt` e di anteporre alla query il prefisso `EXPLAIN`. Se i valori della query sono date e un filtro seleziona un intervallo di tre giorni, l'output `EXPLAIN` potrebbe essere simile a quanto segue:

```
dt := dt:string:PARTITION_KEY
    :: [[2023-06-11], [2023-06-12], [2023-06-13]]
```

L'output `EXPLAIN` mostra che il pianificatore ha trovato tre valori per questa chiave di partizione corrispondenti alla query. Ti mostra anche quali sono questi valori. Per ulteriori informazioni sull'utilizzo di `EXPLAIN` consulta Utilizzo di EXPLAIN e EXPLAIN ANALYZE in Athena e Capire i risultati dell'istruzione EXPLAIN di Athena.

Utilizzo di formati di file colonnari

I formati di file colonnari come Parquet e ORC sono concepiti per carichi di lavoro di analisi distribuiti. Tali formati organizzano i dati per colonna anziché per riga. L'organizzazione dei dati in formato colonnare presenta i seguenti vantaggi:

- Vengono caricate solo le colonne necessarie alla query

- La quantità complessiva di dati da caricare è ridotta
- I valori delle colonne sono memorizzati insieme, così i dati possono essere compressi in modo efficiente
- I file possono contenere metadati che consentono al motore di evitare il caricamento di dati non necessari

Un esempio del possibile utilizzo dei metadati dei file è il fatto che questi possano contenere informazioni sui valori minimi e massimi in una pagina di dati. Se i valori richiesti non rientrano nell'intervallo indicato nei metadati, la pagina può essere saltata.

Un modo per utilizzare questi metadati per migliorare le prestazioni consiste nell'assicurare che i dati all'interno dei file vengano ordinati. Ad esempio, supponi di avere delle query che cercano i record la cui voce `created_at` rientra in un breve intervallo di tempo. Se i dati sono ordinati per colonna `created_at`, Athena può utilizzare i valori minimi e massimi nei metadati del file per ignorare le parti non necessarie dei file di dati.

Quando utilizzi formati di file colonnari, assicurati che i file non siano di dimensioni troppo piccole. Come indicato in Come evitare di avere un numero eccessivo di file, i set di dati con molti file di piccole dimensioni comportano problemi di prestazioni. Ciò è particolarmente vero con i formati di file colonnari. Per i file di piccole dimensioni, è più importante il sovraccarico del formato di file colonnari rispetto ai vantaggi.

Nota che Parquet e ORC sono organizzati internamente per gruppi di file (Parquet) e stripe (ORC). La dimensione predefinita per i gruppi di righe è pari a 128 MB e per le stripe è pari a 64 MB. Se hai molte colonne, puoi aumentare il gruppo di righe e le dimensioni delle stripe per migliorare le prestazioni. Non è consigliabile ridurre la dimensione del gruppo di righe o di stripe a un valore inferiore a quelli predefiniti.

Per convertire altri formati di dati in Parquet o ORC, puoi usare AWS Glue ETL o Athena. Per ulteriori informazioni sull'utilizzo di Athena per ETL, consulta Utilizzo di CTAS e INSERT INTO per ETL e analisi dei dati.

Compressione di dati

Athena supporta un'ampia serie di formati di compressione. L'esecuzione di query su dati compressi è più veloce e anche più economica perché i costi sono commisurati al numero di byte scansionati prima della decompressione.

Il formato gzip offre buoni rapporti di compressione e supporta molti altri strumenti e servizi. Il formato zstd (Zstandard) è un formato di compressione più recente con un buon equilibrio tra prestazioni e rapporto di compressione.

Quando comprimi file di testo come dati JSON e CSV, cerca di mantenere un equilibrio tra il numero di file e la dimensione di questi. Per la maggior parte dei formati di compressione è necessario che il lettore legga i file dall'inizio. In generale, ciò significa che i file di testo compressi non possono essere elaborati in parallelo. I file non compressi di grandi dimensioni sono spesso suddivisi tra i worker per avere maggiore parallelismo durante l'elaborazione delle query, ma ciò non è possibile con la maggior parte dei formati di compressione.

Come detto in Come evitare di avere un numero eccessivo di file, è preferibile non avere né un numero di file troppo alto né troppo basso. Poiché il numero di file rappresenta il limite del numero di lavoratori che possono elaborare la query, questa regola è particolarmente vera per i file compressi.

Per ulteriori informazioni sull'utilizzo della compressione in Athena, consulta Supporto della compressione in Athena.

Come utilizzare il raggruppamento in bucket delle ricerche su chiavi con cardinalità elevata

Il raggruppamento in bucket è una tecnica per distribuire i record in file separati in base al valore di una delle colonne. Questo modo garantisce che tutti i record con lo stesso valore si trovino nello stesso file. Il raggruppamento in bucket è utile quando hai una chiave con cardinalità elevata e quando molte delle query cercano valori specifici della chiave.

Ad esempio, supponi di eseguire una query su un set di record per un utente specifico. Se i dati sono raggruppati in bucket in base all'ID utente, Athena per un ID specifico conosce in anticipo quali file contengono record e quali file no. Ciò consente ad Athena di leggere solo i file che possono contenere l'ID, riducendo notevolmente la quantità di dati letta. Inoltre, ciò riduce il tempo di elaborazione che altrimenti sarebbe necessario per cercare l'ID specifico tra i dati.

Svantaggi del raggruppamento in bucket

Il raggruppamento in bucket è meno utile quando le query cercano spesso diversi valori nella colonna del bucket in cui sono raggruppati i dati. Maggiore è il numero di valori richiesti, maggiore è la probabilità che tutti o la maggior parte dei file debbano essere letti. Ad esempio, se hai tre bucket e una query cerca tre valori diversi, potrebbe essere necessario leggere tutti i file. Il raggruppamento in bucket funziona meglio quando le query cercano valori singoli.

Per ulteriori informazioni, consulta Partizionamento e arrotolamento in Athena.

Come evitare di avere un numero eccessivo di file

I set di dati composti da molti file di piccole dimensioni comportano prestazioni complessive delle query scadenti. Athena, quando pianifica una query, elenca tutte i percorsi delle partizioni, operazione che richiede tempo. La gestione e la richiesta di ogni file comportano anche un sovraccarico computazionale. Pertanto, il caricamento di un singolo file di dimensioni maggiori da Amazon S3 è più veloce rispetto al caricamento degli stessi record da molti file di dimensioni minori.

In casi estremi, potresti incontrare dei limiti al servizio Amazon S3. Amazon S3 supporta fino a 5.500 richieste al secondo su una singola partizione di indice. Inizialmente, un bucket viene trattato come una singola partizione di indice, ma all'aumentare del carico delle richieste, può essere suddiviso in più partizioni di indice.

Amazon S3 esamina i modelli di richiesta e le suddivisioni in base ai prefissi delle chiavi. Se il set di dati è composto da molte migliaia di file, le richieste provenienti da Athena possono superare la quota delle richieste. Anche con un numero inferiore di file, la quota può essere superata se vengono eseguite più query simultanee sullo stesso set di dati. Altre applicazioni che accedono agli stessi file possono contribuire al numero totale di richieste.

Quando la frequenza di richiesta `limit` viene superata, Amazon S3 restituisce il seguente errore. Questo errore è incluso in Athena nelle informazioni sullo stato della query.

SlowDown: Riduci il tasso di richiesta

Per risolvere il problema, stabilisci innanzitutto se l'errore è provocato da una singola query o da più query che leggono gli stessi file. In quest'ultimo caso, coordina l'esecuzione delle query in modo che queste non vengano eseguite contemporaneamente. A tale scopo, aggiungi un meccanismo di accodamento o ritenta l'applicazione.

Se l'esecuzione di una singola query provoca l'errore, prova a combinare file di dati o a modificare la query per leggere un numero inferiore di file. Il momento migliore per combinare file di piccole dimensioni è prima che questi vengano scritti. A tal fine, tieni conto delle seguenti tecniche:

- Modifica il processo di scrittura dei file per scrivere file di maggiori dimensioni. Ad esempio, puoi memorizzare i record nel buffer per un periodo più lungo prima che siano scritti.
- Memorizza i file in un percorso su Amazon S3 e utilizza uno strumento come Glue ETL per combinarli in file più grandi. Quindi, sposta i file più grandi nel percorso a cui punta la tabella. Per ulteriori informazioni, consulta Leggere i file di input in gruppi più grandi nella Guida per gli AWS Glue sviluppatori e Come posso configurare un processo AWS Glue ETL per generare file più grandi? nel Knowledge Center AWS di re:POST.

- Riduzione del numero di chiavi di partizione. Quando hai troppe chiavi di partizione, ogni partizione potrebbe avere solo pochi record, il che comporta un numero eccessivo di file di piccole dimensioni. Per informazioni su come stabilire quali partizioni creare, consulta Scegli le chiavi di partizione che supporteranno le query.

Come evitare gerarchie di archiviazione aggiuntive oltre alla partizione

Per evitare un sovraccarico determinato dalla pianificazione delle query, archivia i file in una struttura piatta in ogni percorso della partizione. Non utilizzare gerarchie di directory aggiuntive.

Athena, quando pianifica una query, elenca tutti i file in tutte le partizioni corrispondenti alla query. Sebbene Amazon S3 non disponga di directory di per sé, la convenzione prevede di interpretare la barra `/` come un separatore di directory. Athena, quando elenca le posizioni delle partizioni, elenca in modo ricorsivo tutte le directory individuate. Quando i file all'interno di una partizione sono organizzati in una gerarchia, si verificano più cicli di elenchi.

Quando tutti i file si trovano direttamente nel percorso della partizione, nella maggior parte dei casi è sufficiente eseguire una sola operazione sull'elenco. Tuttavia, se hai più di 1000 file in una partizione, sono necessarie più operazioni di elenco sequenziali perché Amazon S3 restituisce solo 1000 oggetti per operazione di elenco. La presenza di più di 1000 file in una partizione può creare anche altri problemi di prestazioni, più gravi. Per ulteriori informazioni, consulta Come evitare di avere un numero eccessivo di file.

Utilizzo di SymlinkTextInputFormat solo quando necessario

L'utilizzo della tecnica SymlinkTextInputFormat può essere un modo per aggirare le situazioni in cui i file di una tabella non sono organizzati in modo ordinato in partizioni. Ad esempio, i symlink possono essere utili quando tutti i file hanno lo stesso prefisso o quando i file con schemi diversi si trovano nella stessa posizione.

Tuttavia, l'utilizzo dei symlink aggiunge livelli di riferimenti indiretti all'esecuzione della query. Questi livelli di rifirimenti indiretti influiscono sulle prestazioni complessive. I file symlink devono essere letti e le posizioni che definiscono devono essere elencate. Ciò aggiunge diversi round trip ad Amazon S3 che le normali tabelle Hive non richiedono. In conclusione, dovresti utilizzare `SymlinkTextInputFormat` solo quando non sono disponibili opzioni migliori come la riorganizzazione dei file.

Risorse aggiuntive

Per ulteriori informazioni sull'ottimizzazione delle prestazioni in Athena, prendi in considerazione le seguenti risorse:

- Leggi il post sul blog AWS Big Data I 10 migliori consigli di ottimizzazione delle prestazioni per Amazon Athena
- Per un articolo sull'utilizzo del pushdown del predicato per migliorare le prestazioni nelle query federate, consulta Miglioramento le query federate con il pushdown del predicato in Amazon Athena nel Blog sui Big Data di AWS .
- Per un articolo sulle ottimizzazioni delle prestazioni nel motore di query Athena, consulta Esegui le query 3 volte più velocemente con un risparmio sui costi fino al 70% sull'ultimo motore Amazon Athena nel Big Data Blog.AWS
- Leggi altri post di Athena nel blog sui AWS big data
- Fare domanda su AWS re:Post utilizzando il tag Amazon Athena
- Consulta gli argomenti di Athena nel centro di conoscenza AWS
- Contatti AWS Support (in AWS Management Console, fai clic su Support, Support Center)

Come prevenire la limitazione (della larghezza di banda della rete) di Amazon S3

La limitazione (della larghezza di banda della rete) è il processo di limitazione della velocità con cui utilizzi un servizio, un'applicazione o un sistema. Inoltre AWS, puoi utilizzare la limitazione per evitare un uso eccessivo del servizio Amazon S3 e aumentare la disponibilità e la reattività di Amazon S3 per tutti gli utenti. Tuttavia, poiché la limitazione limitazione (della larghezza di banda della rete) riduce la velocità con cui i dati possono essere trasferiti da o verso Amazon S3, è importante tenere a mente di evitare che le interazioni siano limitate.

Riduzione della limitazione (della larghezza di banda della rete) a livello di servizio

Per evitare la limitazione (della larghezza di banda della rete) di Amazon S3 a livello di servizio, puoi monitorare l'utilizzo e modificare le service quotas oppure puoi utilizzare determinate tecniche, come il partizionamento. Di seguito sono riportate alcune delle condizioni che possono portare alla limitazione (della larghezza di banda della rete):

- Superamento dei limiti di richiesta API da parte del tuo account: Amazon S3 ha limiti di richiesta API predefiniti basati sul tipo di account e sull'utilizzo. Se per un singolo oggetto superi il numero

massimo di richieste al secondo, le tue richieste potrebbero essere sottoposte alle limitazione (della larghezza di banda della rete) per evitare il sovraccarico del servizio Amazon S3.

- Partizionamento insufficiente dei dati: se non partizioni correttamente i dati e trasferisci una grande quantità di dati, Amazon S3 può limitare le richieste. Per ulteriori informazioni sul partizionamento, consulta la sezione Utilizzo del partizionamento in questo documento.
- Numero elevato di oggetti di piccole dimensioni: se possibile, evita molti file di piccole dimensioni. Amazon S3 ha un limite di 5500 richieste GET al secondo per prefisso partizionato e le tue query Athena hanno lo stesso limite. Se in una singola query analizzi milioni di oggetti di piccole dimensioni, è probabile che la tua query venga limitata da Amazon S3.

Per evitare un eccesso di scansione, puoi utilizzare AWS Glue ETL per compattare periodicamente i file oppure partizionare la tabella e aggiungere filtri per le chiavi di partizione. Per ulteriori informazioni, consulta le risorse seguenti.

- Come posso configurare un processo AWS Glue ETL per generare file più grandi? (Centro di AWS conoscenza)
- Leggere i file di input in gruppi più grandi (Guida per AWS Glue gli sviluppatori)

Ottimizzazione delle tabelle

La strutturazione dei dati è importante in caso di problemi di limitazione (della larghezza di banda della rete). Sebbene Amazon S3 sia in grado di gestire grandi quantità di dati, talvolta sopraggiunge una limitazione (della larghezza di banda della rete) a causa del modo in cui i dati sono strutturati.

Le sezioni seguenti mostrano alcuni suggerimenti su come strutturare i dati in Amazon S3 per evitare problemi di limitazione (della larghezza di banda della rete).

Utilizzo del partizionamento

Puoi utilizzare il partizionamento per ridurre le limitazioni (della larghezza di banda della rete) contenendo la quantità di dati a cui accedere in un dato momento. Partizionando i dati su colonne specifiche, puoi distribuire le richieste in modo uniforme su più oggetti e ridurre il numero di richieste per un singolo oggetto. La riduzione della quantità di dati da analizzare migliora le prestazioni delle query e riduce i costi.

Quando crei una tabella, puoi definire le partizioni che fungono da colonne virtuali. Per creare una tabella con partizioni in una dichiarazione `CREATE TABLE`, utilizza la clausola `PARTITIONED BY (`*`column_name data_type`*`)` in modo da definire le chiavi per partizionare i dati.

Per limitare le partizioni analizzate da una query, puoi specificarle come predicati in una clausola `WHERE` della query. Pertanto, le colonne utilizzate frequentemente come filtri sono adatte al partizionamento. In genere è consigliabile partizionare i dati in base agli intervalli di tempo, il che spesso determina uno schema di partizionamento a più livelli.

Nota che anche per il partizionamento sono previsti dei costi. Quando aumenti il numero di partizioni nella tabella, viene aumentato anche il tempo necessario a recuperare ed elaborare i metadati delle partizioni. Pertanto, un partizionamento eccessivo può eliminare i vantaggi derivanti da un partizionamento più oculato. Se i dati sono fortemente disallineati rispetto a un valore di partizione e la maggior parte delle query utilizza tale valore, potresti andare incontro a un sovraccarico aggiuntivo.

Per ulteriori informazioni sul partizionamento in Athena, consulta Cos'è il partizionamento?.

Come raggruppare i dati nel bucket

Un altro modo per partizionare i dati consiste nel raggrupparli in bucket all'interno di una singola partizione. Per raggruppare dati nel bucket specifica una o più colonne che contengono le righe che vuoi raggruppare insieme. Quindi sistema quelle righe in più bucket. In questo modo puoi eseguire una query solo sul bucket da leggere, il che riduce il numero di righe di dati da analizzare.

Quando selezioni una colonna da utilizzare per il raggruppamento in bucket, seleziona la colonna con cardinalità elevata (ovvero con molti valori distinti), che è distribuita uniformemente e che viene spesso utilizzata per filtrare i dati. Un esempio di colonna valida da utilizzare per il raggruppamento in bucket è una chiave primaria, ad esempio una colonna di ID.

Per ulteriori informazioni sul raggruppamento in bucket su Athena, consulta Cos'è il raggruppamento in bucket?.

Usa gli AWS Glue indici di partizione

È possibile utilizzare gli indici di AWS Glue partizione per organizzare i dati in una tabella in base ai valori di una o più partizioni. AWS Glue gli indici di partizione possono ridurre il numero di trasferimenti di dati, la quantità di elaborazione dei dati e il tempo di elaborazione delle query.

Un indice di AWS Glue partizione è un file di metadati che contiene informazioni sulle partizioni della tabella, incluse le chiavi di partizione e i relativi valori. L'indice delle partizioni viene archiviato in un bucket Amazon S3 e viene aggiornato automaticamente non appena vengono AWS Glue aggiunte nuove partizioni alla tabella.

Quando è presente un indice di AWS Glue partizione, le query tentano di recuperare un sottoinsieme delle partizioni invece di caricare tutte le partizioni della tabella. Le query sono eseguite solo sul sottoinsieme di dati rilevante per la query.

Quando si crea una tabella in AWS Glue, è possibile creare un indice di partizione su qualsiasi combinazione di chiavi di partizione definite nella tabella. Dopo aver creato uno o più indici di partizione su una tabella, è necessario aggiungere una proprietà alla tabella che abiliti il filtraggio delle partizioni. Quindi, puoi eseguire query sulla tabella da Athena.

Per informazioni sulla creazione di indici di partizione in AWS Glue, consulta Working with partition indexes nella Developer Guide. AWS GlueAWS Glue Per informazioni su come aggiungere una proprietà di tabella per abilitare il filtraggio delle partizioni, consulta AWS Glue indicizzazione e filtraggio delle partizioni.

Come utilizzare la compressione di dati e la suddivisione dei file

La compressione dei dati può velocizzare le query in maniera significativa se i file hanno dimensioni ottimali o se possono essere suddivisi in gruppi logici. In genere, i rapporti di compressione più elevati richiedono più cicli di CPU per comprimere e decomprimere i dati. Per Athena, è consigliabile utilizzare Apache Parquet o Apache ORC, che comprimono i dati per impostazione predefinita. Per ulteriori informazioni sulla compressione dei dati in Athena, consulta Supporto della compressione in Athena.

La suddivisione dei file aumenta il parallelismo consentendo ad Athena di distribuire il processo di lettura di un singolo file tra più lettori. Se un singolo file non è suddivisibile, può leggerlo solo un lettore mentre gli altri lettori sono inattivi. Apache Parquet e Apache ORC supportano anche i file suddivisibili.

Utilizzo di archivi dati colonnari ottimizzati

Le prestazioni delle query Athena migliorano in modo significativo se converti i dati in un formato colonnare. Quando generi file colonnari, una tecnica di ottimizzazione da considerare consiste nell'ordinare i dati in base alla chiave di partizione.

Apache Parquet e Apache ORC sono gli archivi dati colonnari e open source comunemente utilizzati. Per informazioni sulla conversione dell'origine dati esistente di Amazon S3 in uno di questi formati, consulta Conversione in formati colonnari.

Utilizzo di una dimensione di blocco Parquet o di una dimensione di stripe ORC maggiore

Parquet e ORC dispongono di parametri di archiviazione di dati che puoi regolare per l'ottimizzazione. In Parquet, puoi ottimizzare la dimensione dei blocchi. In ORC, puoi ottimizzare la dimensione delle stripe. Più grande è il blocco o la stripe, maggiore è il numero di righe che puoi memorizzare in ciascuno di essi. Per impostazione predefinita, la dimensione del blocco Parquet è di 128 MB e la dimensione della stripe ORC è di 64 MB.

Se una stripe ORC è inferiore a 8 MB (il valore predefinito di `hive.orc.max_buffer_size`), Athena legge l'intera stripe ORC. Questo è il compromesso a cui Athena scende tra la selettività delle colonne e le operazioni di input/output al secondo per stripe di dimensioni inferiori.

Se hai tabelle con un numero molto elevato di colonne, dimensioni ridotte di blocchi o stripe possono comportare una scansione di più dati di quanto sia necessario. In questi casi, le dimensioni maggiori di un blocco possono essere più efficienti.

Utilizzo di ORC per tipi complessi

Al momento, quando esegui una query su colonne archiviate in Parquet con tipi di dati complessi (`array`, `map` o `struct`), Athena legge un'intera riga di dati anziché leggere selettivamente solo le colonne specificate. Si tratta di un problema noto di Athena. Per ovviare al problema, prendi in considerazione la possibilità di utilizzare ORC.

Scelta di un algoritmo di compressione

Un altro parametro che puoi configurare è l'algoritmo di compressione sui blocchi di dati. Per informazioni sugli algoritmi di compressione supportati per Parquet e ORC in Athena, consulta Supporto della compressione Athena.

Per ulteriori informazioni sull'ottimizzazione dei formati di storage a colonne in Athena, consulta la sezione «Ottimizzazione della generazione di archivi dati a colonne» nel post del blog AWS Big Data I 10 migliori consigli per l'ottimizzazione delle prestazioni per Amazon Athena.

Utilizzo delle tabelle Iceberg

Apache Iceberg è un formato a tabella aperta per set di dati analitici di dimensioni molto grandi concepito per un utilizzo ottimizzato su Amazon S3. Puoi usare le tabelle Iceberg per ridurre la limitazione (della larghezza di banda della rete) in Amazon S3.

Le tabelle Iceberg ti offrono i seguenti vantaggi:

- Puoi partizionare le tabelle Iceberg su una o più colonne. Ciò ottimizza l'accesso ai dati e riduce la quantità di dati che le query devono scansionare.
- La modalità di archiviazione di oggetti Iceberg, poiché ottimizza l'utilizzo delle tabelle Iceberg in Amazon S3, può elaborare grandi volumi di dati e carichi di lavoro di query pesanti.
- Le tabelle Iceberg in modalità di archiviazione di oggetti sono scalabili, resistenti agli errori e durevoli, il che può aiutare a ridurre la limitazione (della larghezza di banda della rete).
- Con il supporto delle transazioni ACID più utenti possono aggiungere ed eliminare oggetti Amazon S3 in modo atomico.

Per ulteriori informazioni su Apache Iceberg, consulta Apache Iceberg. Per informazioni sull'utilizzo delle tabelle Apache Iceberg in Athena, consulta la sezione Utilizzo delle tabelle Iceberg.

Ottimizzazione delle query

Utilizza i suggerimenti di questa sezione per ottimizzare le query SQL in Athena.

Utilizza LIMIT con la clausola ORDER BY

La clausola `ORDER BY` restituisce i dati disponendoli in ordine. Ciò richiede che Athena invii tutte le righe di dati a un singolo nodo worker e che quindi ordini le righe. Questo tipo di query può essere eseguito a lungo o addirittura non andare a buon fine.

Per migliorare l'efficienza delle query, esamina i valori *N* superiori o inferiori, quindi utilizza anche una clausola `LIMIT`. Ciò riduce in modo significativo il costo dell'ordinamento, inviando sia l'ordinamento sia le restrizioni ai singoli nodi worker anziché a un singolo worker.

Ottimizzazione delle clausole JOIN

Quando effettui il join di due tabelle, Athena distribuisce la tabella a destra ai nodi worker, quindi trasmette la tabella a sinistra per eseguire il join.

Per questo motivo, specifica la tabella di maggiori dimensioni sul lato sinistro del join e la tabella di dimensioni inferiori sul lato destro del join. In questo modo, Athena utilizza meno memoria ed esegue la query con una latenza inferiore.

Inoltre, nota quanto segue:

- Quando utilizzi più comandi `JOIN`, specifica le tabelle dalla più grande alla più piccola.

- Evita i cross join a meno che non siano richiesti dalla query.

Ottimizzazione delle clausole GROUP BY

L'operatore `GROUP BY` distribuisce le righe ai nodi worker in base alle colonne `GROUP BY`. Queste colonne hanno un riferimento in memoria e i valori sono confrontati man mano che le righe sono acquisite. I valori sono aggregati insieme quando si trova una corrispondenza con la colonna `GROUP BY`. Considerando il modo di funzionamento di questo processo, è consigliabile ordinare le colonne dalla cardinalità più alta a quella più bassa.

Utilizzo di numeri anziché di stringhe

Poiché i numeri richiedono meno memoria e sono più veloci da elaborare rispetto alle stringhe, se possibile utilizza numeri anziché stringhe.

Limitazione del numero di colonne

Per ridurre la quantità totale di memoria necessaria per archiviare i dati, limita il numero di colonne specificato nella dichiarazione `SELECT`.

Utilizzo di espressioni regolari anziché di LIKE

Le query che includono clausole quali `LIKE '%string%'` su stringhe di grandi dimensioni possono essere molto intense dal punto di vista computazionale. Quando filtri in base a più valori su una colonna di stringhe, utilizza la funzione regexp_like() e un'espressione regolare. Ciò è particolarmente utile quando confronti un lungo elenco di valori.

Utilizzo della clausola LIMIT

Quando esegui una query, anziché selezionare tutte le colonne, puoi utilizzare la clausola `LIMIT` per restituire solo le colonne necessarie. In questo modo si riducono le dimensioni del set di dati elaborato tramite la pipeline di esecuzione delle query. Le clausole `LIMIT` sono più utili quando esegui query su tabelle con un gran numero di colonne basate su stringhe. Le clausole `LIMIT` sono utili anche quando esegui più join o aggregazioni sulle query.

Risorse aggiuntive

Modelli di concezione di best practice: ottimizzazione delle prestazioni di Amazon S3 nella Guida per l'utente di Amazon Simple Storage Service.

Ottimizzazione delle prestazioni in Athena

Supporto della compressione in Athena

Argomenti

Athena supporta diversi formati di compressione per la lettura e la scrittura di dati, inclusa la lettura da una tabella che utilizza più formati di compressione. Ad esempio, Athena può leggere correttamente i dati in una tabella che utilizza il formato file Parquet quando alcuni file Parquet vengono compressi con Snappy e altri file Parquet vengono compressi con GZIP. Lo stesso principio vale per i formati di archiviazione ORC, file di testo e JSON.

Athena supporta i seguenti formati di compressione:

- BZIP2: formato che utilizza l'algoritmo Burrows-Wheeler.
- DEFLATE: algoritmo di compressione basato su LZSS e codifica Huffman. Deflate è rilevante solo per il formato file Avro.
- GZIP: algoritmo di compressione basato su Deflate. Per le tabelle Hive nel motore Athena versioni 2 e 3 e le tabelle Iceberg nel motore Athena versione 2, il formato di compressione in scrittura predefinito per i file nei formati di archiviazione dei file di testo e Parquet è GZIP. I file nel formato `tar.gz` non sono supportati.
- LZ4: questo membro della famiglia Lempel-Ziv 77 (LZ7) si concentra sulla velocità di compressione e decompressione piuttosto che sulla compressione massima dei dati. LZ4 ha i seguenti formati di framing:
  - LZ4 raw/senza frame: una implementazione standard senza frame del formato di compressione a blocchi LZ4. Per ulteriori informazioni, vedere la descrizione del formato di blocco LZ4 su. GitHub
  - LZ4 con frame: la consueta implementazione di framing di LZ4. Per ulteriori informazioni, vedere la descrizione del formato del frame LZ4 su. GitHub
  - Compatibile con Hadoop LZ4: l'implementazione di Apache Hadoop di LZ4. Questa implementazione include la compressione LZ4 con la classe .java. BlockCompressorStream

- LZO: formato che utilizza l'algoritmo Lempel-Ziv-Oberhumer, che si concentra sull'elevata velocità di compressione e decompressione anziché sulla compressione massima dei dati. LZO ha due implementazioni:
  - LZO standard: per ulteriori informazioni, consulta il riassunto su LZO sul sito Web di Oberhumer.
  - Compatibile con LZO con hadoop: questa implementazione racchiude l'algoritmo LZO con la classe .java. BlockCompressorStream
- SNAPPY: algoritmo di compressione che fa parte della famiglia Lempel-Ziv 77 (LZ7). Snappy si concentra sull'elevata velocità di compressione e decompressione piuttosto che sulla compressione massima dei dati.
- ZLIB: basato su Deflate, ZLIB è il formato di compressione della scrittura di default per i file nel formato di archiviazione dati ORC. Per ulteriori informazioni, consulta la pagina zlib su. GitHub
- ZSTD: l'algoritmo di compressione dei dati in tempo reale Zstandard è un algoritmo di compressione veloce che fornisce rapporti di compressione elevati. La libreria Zstandard (ZSTD) viene fornita come software open source tramite una licenza BSD. ZSTD è la compressione predefinita per le tabelle Iceberg. Quando si scrivono dati compressi ZSTD, Athena utilizza il livello di compressione ZSTD 3 per impostazione predefinita. Per ulteriori informazioni sui livelli di compressione ZSTD in Athena, consulta Utilizzo dei livelli di compressione ZSTD in Athena.

Specificare i formati di compressione

Quando si scrivono istruzioni CREATE TABLE o CTAS, è possibile specificare le proprietà di compressione che specificano il tipo di compressione da utilizzare quando Athena scrive su tali tabelle.

- Per CTAS, consultare Proprietà tabella CTAS. Per alcuni esempi, consulta Esempi di query CTAS.
- Per CREATE TABLE, consultare ALTER TABLE SET TBLPROPERTIES per un elenco delle proprietà della tabella di compressione.

Specificare l'assenza di compressione

Le istruzioni CREATE TABLE supportano la scrittura di file non compressi. Per scrivere file non compressi, utilizza la sintassi seguente:

- CREATE TABLE (file di testo o JSON): in `TBLPROPERTIES`, specifica `write.compression = NONE`.

- CREATE TABLE (Parquet): in `TBLPROPERTIES`, specifica `parquet.compression = UNCOMPRESSED`.
- CREATE TABLE (ORC): in `TBLPROPERTIES`, specifica `orc.compress = NONE`.

Note e risorse

- Attualmente, estensioni di file in lettere maiuscole come `.GZ` o `.BZIP2` non sono riconosciute da Athena. Evitare di utilizzare set di dati con estensioni di file in lettere maiuscole o rinominare le estensioni dei file di dati in lettere minuscole.
- Per i dati in CSV, TSV e JSON, Athena determina il tipo di compressione dall'estensione del file. Se non è presente alcuna estensione di file, i dati vengono gestiti da Athena come testo normale non compresso. Se i dati sono compressi, verifica che il nome del file includa l'estensione della compressione, ad esempio `gz`.
- Il formato di file ZIP non è supportato.
- Per interrogare i log di Amazon Data Firehose da Athena, i formati supportati includono la compressione GZIP o i file ORC con compressione SNAPPY.
- Per ulteriori informazioni sull'uso della compressione, consulta la sezione 3 («Comprimi e dividi file») del post del blog AWS Big Data I 10 migliori consigli per l'ottimizzazione delle prestazioni per Amazon Athena.

Supporto per la compressione delle tabelle Hive in base al formato di file

Il supporto alla compressione Hive in Athena dipende dalla versione del motore.

Supporto della compressione Hive nella versione 3 del motore Athena

Nella tabella seguente viene riepilogato il supporto del formato di compressione nella versione 3 del motore Athena per i formati file di archiviazione in Apache Hive. Il formato del file di testo include TSV, CSV, JSON e SerDes personalizzati per il testo. "Yes" (Sì) o "No" in una cella si applicano allo stesso modo alle operazioni di lettura e di scrittura, tranne dove indicato. Ai fini di questa tabella, CREATE TABLE, CTAS e INSERT INTO sono considerate operazioni di scrittura. Per ulteriori informazioni sui livelli di compressione ZSTD in Athena, consulta Utilizzo dei livelli di compressione ZSTD in Athena.

	Avro	Ion	ORC	Parquet	File di testo
BZIP2	Sì	Sì	No	No	Sì
DEFLATE	Sì	No	No	No	No
GZIP	No	Sì	No	Sì	Sì
LZ4	No	Sì	Sì	Sì	Sì
LZO	No	Scrittura - No Lettura - Sì	No	Sì	Scrittura - No Lettura - Sì
SNAPPY	Sì	Sì	Sì	Sì	Sì
ZLIB	No	No	Sì	No	No
ZSTD	Sì	Sì	Sì	Sì	Sì
NONE	Sì	Sì	Sì	Sì	Sì

Supporto della compressione Hive nella versione 2 del motore Athena

Nella tabella seguente viene riepilogato il supporto del formato di compressione nella versione 2 del motore Athena per Apache Hive. Il formato del file di testo include TSV, CSV, JSON e SerDes personalizzati per il testo. "Yes" (Sì) o "No" in una cella si applicano allo stesso modo alle operazioni di lettura e di scrittura, tranne dove indicato. Ai fini di questa tabella, CREATE TABLE, CTAS e INSERT INTO sono considerate operazioni di scrittura.

	Avro	Ion	ORC	Parquet	File di testo
BZIP2	Sì	Sì	No	No	Sì
DEFLATE	Sì	No	No	No	No

	Avro	Ion	ORC	Parquet	File di testo
GZIP	No	Sì	No	Sì	Sì
LZ4	No	No	Sì	Scrittura: sì Lettura: no	Scrittura - No Lettura - Sì
LZO	No	Scrittura - No Lettura - Sì	No	Sì	Scrittura - No Lettura - Sì
SNAPPY	Sì	Sì	Sì	Sì	Sì
ZLIB	No	No	Sì	No	No
ZSTD	No	Sì	Sì	Sì	Sì
NONE	Sì	Sì	Sì	Sì	Sì

Supporto per la compressione delle tabelle Iceberg in base al formato di file

Il supporto alla compressione in Athena Iceberg dipende dalla versione del motore.

Supporto della compressione Iceberg nella versione 3 del motore Athena

Nella tabella seguente viene riepilogato il supporto del formato di compressione nella versione 3 del motore Athena per i formati file di archiviazione in Apache Iceberg. "Yes" (Sì) o "No" in una cella si applicano allo stesso modo alle operazioni di lettura e di scrittura, tranne dove indicato. Ai fini di questa tabella, CREATE TABLE, CTAS e INSERT INTO sono considerate operazioni di scrittura. Il formato di archiviazione predefinito per Iceberg nella versione 3 del motore Athena è Parquet. Il formato di compressione predefinito per Iceberg nella versione 3 del motore Athena è ZSTD. Per ulteriori informazioni sui livelli di compressione ZSTD in Athena, consulta Utilizzo dei livelli di compressione ZSTD in Athena.

	Avro	ORC	Parquet (predefinito)
BZIP2	No	No	No
GZIP	Sì	No	Sì
LZ4	No	Sì	No
SNAPPY	Sì	Sì	Sì
ZLIB	No	Sì	No
ZSTD	Sì	Sì	Sì (impostazione predefinita)
NONE	Sì (specifica `None` o `Deflate`)	Sì	Sì (specifica `None` o `Uncompressed`)

Supporto della compressione Iceberg nella versione 2 del motore Athena

Nella tabella seguente viene riepilogato il supporto del formato di compressione nella versione 2 del motore Athena per Apache Iceberg. "Yes" (Sì) o "No" in una cella si applicano allo stesso modo alle operazioni di lettura e di scrittura, tranne dove indicato. Ai fini di questa tabella, CREATE TABLE, CTAS e INSERT INTO sono considerate operazioni di scrittura. Il formato di archiviazione predefinito per Iceberg nella versione 2 del motore Athena è Parquet. Il formato di compressione predefinito per Iceberg nella versione 2 del motore Athena è GZIP.

	Avro (Non supportato)	ORC (Non supportato)	Parquet (predefinito)
BZIP2	No	No	No
GZIP	No	No	Sì (impostazione predefinita)
LZ4	No	No	No
SNAPPY	No	No	Sì
ZLIB	No	No	No

	Avro (Non supportato)	ORC (Non supportato)	Parquet (predefinito)
ZSTD	No	No	Sì
NONE	No	No	Sì

Utilizzo dei livelli di compressione ZSTD in Athena

L'algoritmo di compressione dei dati in tempo reale Zstandard è un algoritmo di compressione veloce che fornisce rapporti di compressione elevati. La libreria Zstandard (ZSTD) è un software open source e utilizza una licenza BSD. Athena supporta la lettura e la scrittura di dati compressi ZSTD ORC, Parquet e file di testo.

È possibile utilizzare i livelli di compressione ZSTD per regolare il rapporto di compressione e la velocità in base alle proprie esigenze. La libreria ZSTD supporta livelli di compressione da 1 a 22. Per impostazione predefinita, Athena utilizza il livello di compressione ZSTD 3.

I livelli di compressione forniscono compromessi granulari tra la velocità di compressione e la quantità di compressione raggiunta. Livelli di compressione più bassi offrono una maggiore velocità ma file di dimensioni maggiori. Ad esempio, puoi usare il livello 1 se la velocità è più importante e il livello 22 se invece è più importante la dimensione. Il livello 3 è adatto a numerosi casi d'uso ed è l'impostazione predefinita. Usa i livelli superiori a 19 con cautela poiché richiedono più memoria. La libreria ZSTD offre anche livelli di compressione negativi che estendono la gamma di velocità e rapporti di compressione. Per ulteriori informazioni, consulta RFC di compressione Zstandard.

L'abbondanza di livelli di compressione offre notevoli opportunità per una regolazione precisa. Tuttavia, quando decidi il livello di compressione assicurati di misurare i tuoi dati e di prendere in considerazione dei compromessi. Si consiglia di utilizzare il livello predefinito 3 o un livello compreso tra 6 e 9 per un ragionevole compromesso tra velocità di compressione e dimensione dei dati compressi. Utilizza i livelli pari o superiori a 20 per i casi in cui le dimensioni sono più importanti e la velocità di compressione non è un problema.

Considerazioni e limitazioni

Quando utilizzi un livello di compressione ZSTD in Athena, considera i seguenti punti.

- La proprietà `compression_level` ZSTD è supportata solo nella versione 3 del motore Athena.

- La proprietà `compression_level` ZSTD è supportata per le istruzioni`ALTER TABLE`, `CREATE TABLE`, `CREATE TABLE AS` (CTAS) e `UNLOAD`.
- La proprietà `compression_level` è facoltativa.
- La proprietà `compression_level` è supportata solo per la compressione ZSTD.
- I livelli di compressione possibili sono compresi tra 1 e 22.
- Il livello di compressione predefinito è 3.

Per informazioni sul supporto della compressione Apache Hive ZSTD in Athena, consulta Supporto per la compressione delle tabelle Hive in base al formato di file. Per informazioni sul supporto della compressione Apache Iceberg ZSTD in Athena, consulta Supporto per la compressione delle tabelle Iceberg in base al formato di file.

Specifica dei livelli di compressione ZSTD

Per specificare il livello di compressione ZSTD per le istruzioni`ALTER TABLE`, `CREATE TABLE`, `CREATE TABLE AS` e `UNLOAD`, utilizza la proprietà `compression_level`. Per specificare la compressione ZSTD stessa, è necessario utilizzare la singola proprietà di compressione utilizzata dalla sintassi dell'istruzione.

ALTER TABLE SET TBLPROPERTIES

Nella clausola `SET TBLPROPERTIES` dell'istruzione ALTER TABLE SET TBLPROPERTIES, specifica la compressione ZSTD utilizzando `'write.compression' = ' ZSTD'` o `'parquet.compression' = 'ZSTD'`. Quindi utilizza la proprietà `compression_level` per specificare un valore compreso tra 1 e 22 (ad esempio, `'compression_level' = 5`). Se non si specifica una proprietà del livello di compressione, il livello di compressione predefinito sarà 3.

Esempio

L'esempio seguente modifica la tabella `existing_table` per utilizzare il formato file Parquet con compressione ZSTD e livello di compressione ZSTD 4. Nota che il valore del livello di compressione deve essere immesso come stringa anziché come numero intero.

```
ALTER TABLE existing_table
SET TBLPROPERTIES ('parquet.compression' = 'ZSTD', 'compression_level' = 4)
```

CREATE TABLE

Nella clausola `TBLPROPERTIES` dell'istruzione CREATE TABLE, specifica `'write.compression' = 'ZSTD'` o `'parquet.compression' = 'ZSTD'`, quindi utilizza `compression_level = compression_level` e specifica un valore compreso tra 1 e 22. Se la proprietà `compression_level` non viene specificata, il livello di compressione predefinito sarà 3.

Esempio

L'esempio seguente crea una tabella nel formato file Parquet con compressione ZSTD e livello di compressione ZSTD 4.

```
CREATE EXTERNAL TABLE new_table (
  `col0` string COMMENT '',
  `col1` string COMMENT ''
)
STORED AS PARQUET
LOCATION 's3://DOC-EXAMPLE-BUCKET/'
TBLPROPERTIES ('write.compression' = 'ZSTD', 'compression_level' = 4)
```

CREATE TABLE AS (CTAS)

Nella clausola `WITH` dell'istruzione CREATE TABLE AS, specifica `write_compression = 'ZSTD'` o `parquet_compression = 'ZSTD'`, quindi utilizza `compression_level = compression_level` e specifica un valore compreso tra 1 e 22. Se la proprietà `compression_level` non viene specificata, il livello di compressione predefinito sarà 3.

Esempio

L'esempio CTAS seguente specifica Parquet come formato file con compressione ZSTD e livello di compressione ZSTD 4.

```
CREATE TABLE new_table
WITH ( format = 'PARQUET', write_compression = 'ZSTD', compression_level = 4)
AS SELECT * FROM old_table
```

UNLOAD

Nella clausola `WITH` dell'istruzione UNLOAD, specifica `compression = 'ZSTD'`, quindi utilizza `compression_level = compression_level` e specifica un valore compreso tra 1 e 22. Se la proprietà `compression_level` non viene specificata, il livello di compressione predefinito sarà 3.

Esempio

L'esempio seguente scarica i risultati della query nella posizione specificata utilizzando il formato file Parquet, la compressione ZSTD e il livello di compressione ZSTD 4.

```
UNLOAD (SELECT * FROM old_table)
TO 's3://DOC-EXAMPLE-BUCKET/'
WITH (format = 'PARQUET', compression = 'ZSTD', compression_level = 4)
```

Assegnazione di tag alle risorse Athena

Un tag consiste di una chiave e di un valore che definisci. Quando applichi tag a una risorsa Athena, assegni dei metadati personalizzati a tale risorsa. Puoi usare i tag per categorizzare le risorse AWS in diversi modi, ad esempio per scopo, proprietario o ambiente. In Athena, risorse come gruppi di lavoro, cataloghi di dati e prenotazioni di capacità sono risorse etichettabili. Ad esempio, puoi creare un set di tag per i gruppi di lavoro nel tuo account per monitorare i proprietari del gruppo di lavoro o identificare i gruppi di lavoro in base al loro scopo. Se abiliti i tag anche come tag di allocazione dei costi nella console Fatturazione e gestione costi, i costi associati all'esecuzione delle query vengono visualizzati nei Report costi e utilizzo con tale tag di allocazione dei costi. Ti consigliamo di usare le best practice di tagging di AWS per creare un set di tag coerente per soddisfare i requisiti dell'organizzazione.

Puoi lavorare con i tag utilizzando la console Athena o le operazioni API.

Argomenti

- Nozioni di base sui tag
- Limitazioni applicate ai tag
- Utilizzo dei tag nei gruppi di lavoro nella console
- Utilizzo delle operazioni relative ai tag
- Policy di controllo degli accessi IAM basate su tag

Nozioni di base sui tag

Un tag è un'etichetta che assegni a una risorsa Athena. Ogni tag è composto da una chiave e da un valore opzionale, entrambi personalizzabili.

I tag consentono di suddividere le risorse AWS in categorie in diversi modi. Ad esempio, puoi definire un set di tag per i gruppi di lavoro del tuo account e monitorare così il proprietario o lo scopo di ogni gruppo di lavoro.

Puoi aggiungere tag durante la creazione di un nuovo gruppo di lavoro o catalogo dati Athena oppure aggiungere, modificare o rimuovere tag a tali risorse. Puoi modificare un tag nella console. Per usare le operazioni API, per modificare un tag rimuovi il tag precedente e aggiungine uno nuovo. Se elimini una risorsa, verranno eliminati anche tutti i tag associati alla risorsa.

Athena non assegna automaticamente i tag alle risorse. Puoi modificare chiavi e valori di tag e rimuovere tag da una risorsa in qualsiasi momento. Puoi impostare il valore di un tag su una stringa vuota, ma non su null. Non aggiungere chiavi tag duplicate alla stessa risorsa. In caso contrario, Athena genera un messaggio di errore. Se utilizzi l'operazione TagResource per contrassegnare una risorsa utilizzando una chiave tag esistente, il nuovo valore tag sovrascrive il valore precedente.

In IAM puoi controllare quali utenti nel tuo account Amazon Web Services dispongono dell'autorizzazione per creare, modificare, eliminare o elencare i tag. Per ulteriori informazioni, consulta Policy di controllo degli accessi IAM basate su tag.

Per un elenco completo delle operazioni dei tag Amazon Athena, consulta i nomi delle operazioni API nella documentazione di riferimento dell'API Amazon Athena.

Puoi usare i tag per la fatturazione. Per ulteriori informazioni, consulta la sezione relativa all'utilizzo dei tag per la fatturazione nella Guida per l'utente di AWS Billing and Cost Management.

Per ulteriori informazioni, consulta Limitazioni applicate ai tag.

Limitazioni applicate ai tag

I tag hanno le restrizioni seguenti:

- In Athena, è possibile contrassegnare gruppi di lavoro e cataloghi dati. Non è possibile aggiungere tag alle query.
- Il numero massimo di tag per risorsa è 50. Per rientrare nel limite, esamina ed eliminare i tag inutilizzati.
- Per ciascuna risorsa, ogni chiave del tag deve essere univoca e ogni chiave del tag può avere un solo valore. Non aggiungere chiavi di tag duplicate contemporaneamente alla stessa risorsa. In caso contrario, Athena genera un messaggio di errore. Se aggiungi un tag a una risorsa usando una chiave di tag esistente in un'operazione `TagResource` separata, il nuovo valore di tag sovrascrive il valore precedente.

- La lunghezza della chiave di tag è compresa tra 1 e 128 caratteri Unicode in formato UTF-8.
- La lunghezza del valore del tag è compresa tra 0 e 256 caratteri Unicode in formato UTF-8.

  Le operazioni di tagging, come l'aggiunta, la modifica, la rimozione o l'elenco di tag, richiedono l'impostazione di un ARN per la risorsa gruppo di lavoro.
- Athena consente di utilizzare lettere, numeri, spazi rappresentati in formato UTF-8 e i caratteri seguenti: + - = . _ : / @.
- Per le chiavi e i valori dei tag viene fatta la distinzione tra maiuscole e minuscole.
- Il prefisso `"aws:"` nelle chiavi tag è riservato per l'uso da parte di AWS. Non è possibile modificare né eliminare le chiavi di tag con tale prefisso. I tag con questo prefisso non vengono conteggiati per il limite del numero di tag per risorsa.
- I tag che assegni sono disponibili solo per il tuo account Amazon Web Services.

Utilizzo dei tag nei gruppi di lavoro nella console

Tramite la console Athena puoi vedere quali tag sono utilizzati da ogni gruppo di lavoro nel tuo account. Puoi visualizzare i tag solo in base al gruppo di lavoro. Puoi anche utilizzare la console Athena per applicare, modificare o rimuovere i tag da un gruppo di lavoro per volta.

Puoi eseguire ricerche nei gruppi di lavoro utilizzando i tag che hai creato.

Argomenti

Visualizzazione dei tag per singoli gruppi di lavoro

Per visualizzare i tag per un singolo gruppo di lavoro nella console Athena

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Nel menu di navigazione, scegli Workgroups (Gruppi di lavoro), quindi seleziona il gruppo di lavoro desiderato.
4. Completa una delle seguenti operazioni:
   - Seleziona la scheda Tags (Tag). Se l'elenco dei tag è lungo, usa la casella di ricerca.
   - Scegli Edit (Modifica), quindi scorri verso il basso fino alla sezione Tags (Tag).

Aggiunta ed eliminazione di tag in un singolo gruppo di lavoro

Puoi gestire i tag per un singolo gruppo di lavoro direttamente dalla scheda Workgroups (Gruppi di lavoro).

> **Note**
>
> Se desideri che gli utenti aggiungano tag quando creano un gruppo di lavoro nella console o passino tag quando utilizzano l'azione CreateWorkGroup, assicurati di concedere agli utenti le autorizzazioni IAM per le operazioni TagResource e CreateWorkGroup.

Per aggiungere un tag durante la creazione di un nuovo gruppo di lavoro

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Nel menu di navigazione scegli Workgroups (Gruppi di lavoro).

3. Scegli Create workgroup (Crea gruppo di lavoro) e inserisci i valori necessari. Per informazioni dettagliate sulle fasi, consulta Creare un gruppo di lavoro.
4. Nella sezione Tags (Tag) aggiungi uno o più tag specificando le chiavi e i valori. Non aggiungere chiavi di tag duplicate contemporaneamente allo stesso gruppo di lavoro. In caso contrario, Athena genera un messaggio di errore. Per ulteriori informazioni, consulta Limitazioni applicate ai tag.
5. Al termine, scegli Create Workgroup (Crea gruppo di lavoro).

Per aggiungere o modificare un tag in un gruppo di lavoro esistente

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Nel pannello di navigazione, seleziona Workgroups (Gruppi di lavoro).
3. Scegli il gruppo di lavoro da modificare.
4. Completa una delle seguenti operazioni:
   - Scegliere la scheda Tags (Tag) quindi scegliere Manage tags (Gestisci tag).
   - Scegli Edit (Modifica), quindi scorri verso il basso fino alla sezione Tags (Tag).
5. Specifica una chiave e un valore per ogni tag. Per ulteriori informazioni, consulta Limitazioni applicate ai tag.
6. Seleziona Salva.

Per eliminare un tag da un singolo gruppo di lavoro

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Nel pannello di navigazione, seleziona Workgroups (Gruppi di lavoro).
3. Scegli il gruppo di lavoro da modificare.
4. Completa una delle seguenti operazioni:
   - Scegliere la scheda Tags (Tag) quindi scegliere Manage tags (Gestisci tag).
   - Scegli Edit (Modifica), quindi scorri verso il basso fino alla sezione Tags (Tag).
5. Nell'elenco dei tag seleziona Remove (Rimuovi) per il tag che desideri eliminare, quindi scegli Save (Salva).

Utilizzo delle operazioni relative ai tag

Utilizzare le seguenti operazioni relative ai tag per aggiungere, rimuovere o elencare tag su una risorsa.

API	CLI	Descrizione dell'operazione
`TagResource`	`tag-resource`	Aggiungere o sovrascrivere uno o più tag sulla risorsa con l'ARN specificato.
`UntagResource`	`untag-resource`	Eliminare uno o più tag dalla risorsa con l'ARN specificato.
`ListTagsForResource`	`list-tags-for-resource`	Elencare uno o più tag per la risorsa con l'ARN specificato.

Aggiunta di tag durante la creazione di una risorsa

Per aggiungere tag quando si crea un gruppo di lavoro o un catalogo dati, utilizzare il parametro `tags` con le operazioni API `CreateWorkGroup` o `CreateDataCatalog` oppure con i comandi AWS CLI `create-work-group` o `create-data-catalog`.

Gestione dei tag mediante operazioni API

Negli esempi di questa sezione viene illustrato come utilizzare le operazioni API dei tag per gestire i tag in gruppi di lavoro e cataloghi dati. Gli esempi sono nel linguaggio di programmazione Java.

Example TagResource

Nell'esempio seguente vengono aggiunti due tag al gruppo di lavoro `workgroupA`:

```
List<Tag> tags = new ArrayList<>();
tags.add(new Tag().withKey("tagKey1").withValue("tagValue1"));
tags.add(new Tag().withKey("tagKey2").withValue("tagValue2"));

TagResourceRequest request = new TagResourceRequest()
    .withResourceARN("arn:aws:athena:us-east-1:123456789012:workgroup/workgroupA")
    .withTags(tags);
```

```
client.tagResource(request);
```

Nell'esempio seguente vengono aggiunti due tag al catalogo dati `datacatalogA`:

```
List<Tag> tags = new ArrayList<>();
tags.add(new Tag().withKey("tagKey1").withValue("tagValue1"));
tags.add(new Tag().withKey("tagKey2").withValue("tagValue2"));

TagResourceRequest request = new TagResourceRequest()
    .withResourceARN("arn:aws:athena:us-east-1:123456789012:datacatalog/datacatalogA")
    .withTags(tags);

client.tagResource(request);
```

> **Note**
>
> Non aggiungere chiavi tag duplicate alla stessa risorsa. In caso contrario, Athena genera un messaggio di errore. Se aggiungi un tag a una risorsa usando una chiave di tag esistente in un'operazione `TagResource` separata, il nuovo valore di tag sovrascrive il valore precedente.

Example UntagResource

L'esempio seguente rimuove `tagKey2` dal gruppo di lavoro `workgroupA`:

```
List<String> tagKeys = new ArrayList<>();
tagKeys.add("tagKey2");

UntagResourceRequest request = new UntagResourceRequest()
    .withResourceARN("arn:aws:athena:us-east-1:123456789012:workgroup/workgroupA")
    .withTagKeys(tagKeys);

client.untagResource(request);
```

L'esempio seguente rimuove `tagKey2` dal catalogo dati `datacatalogA`:

```
List<String> tagKeys = new ArrayList<>();
tagKeys.add("tagKey2");

UntagResourceRequest request = new UntagResourceRequest()
```

```
    .withResourceARN("arn:aws:athena:us-east-1:123456789012:datacatalog/datacatalogA")
    .withTagKeys(tagKeys);

client.untagResource(request);
```

Example ListTagsForResource

Nell'esempio seguente vengono elencati i tag per il gruppo di lavoro `workgroupA`:

```
ListTagsForResourceRequest request = new ListTagsForResourceRequest()
    .withResourceARN("arn:aws:athena:us-east-1:123456789012:workgroup/workgroupA");

ListTagsForResourceResult result = client.listTagsForResource(request);

List<Tag> resultTags = result.getTags();
```

Nell'esempio seguente vengono elencati i tag per il catalogo dati `datacatalogA`:

```
ListTagsForResourceRequest request = new ListTagsForResourceRequest()
    .withResourceARN("arn:aws:athena:us-east-1:123456789012:datacatalog/datacatalogA");

ListTagsForResourceResult result = client.listTagsForResource(request);

List<Tag> resultTags = result.getTags();
```

Gestione dei tag con la AWS CLI

Nelle sezioni seguenti viene illustrato come utilizzare AWS CLI per creare e gestire tag nei cataloghi dati.

Aggiunta di tag a una risorsa: tag-resource

Il comando `tag-resource` aggiunge uno o più tag a una risorsa specificata

Sintassi

```
aws athena tag-resource --resource-arn
arn:aws:athena:region:account_id:datacatalog/catalog_name --tags
Key=string,Value=string Key=string,Value=string
```

Il parametro `--resource-arn` specifica la risorsa a cui vengono aggiunti i tag. Il parametro `--tags` specifica un elenco di coppie chiave-valore separate da spazi da aggiungere come tag alla risorsa.

Example

Nell'esempio seguente vengono aggiunti tag al catalogo dati `mydatacatalog`.

```
aws athena tag-resource --resource-arn arn:aws:athena:us-
east-1:111122223333:datacatalog/mydatacatalog --tags Key=Color,Value=Orange
 Key=Time,Value=Now
```

Per mostrare il risultato, utilizzare il comando `list-tags-for-resource`.

Per informazioni sull'aggiunta di tag durante l'utilizzo del comando `create-data-catalog`, consulta la sezione Registrazione di un catalogo: C reate-data-catalog.

Elencazione dei tag per una risorsa: list-tags-for-resource

Il comando `list-tags-for-resource` elenca i tag per la risorsa specificata.

Sintassi

`aws athena list-tags-for-resource --resource-arn arn:aws:athena:`*`region`*`:`*`account_id`*`:datacatalog/`*`catalog_name`*

Il parametro `--resource-arn` specifica la risorsa per la quale sono elencati i tag.

Nell'esempio seguente vengono elencati i tag per il catalogo dati `mydatacatalog`.

```
aws athena list-tags-for-resource --resource-arn arn:aws:athena:us-
east-1:111122223333:datacatalog/mydatacatalog
```

Il seguente risultato di esempio è in formato JSON.

```
{
    "Tags": [
        {
            "Key": "Time",
            "Value": "Now"
        },
        {
            "Key": "Color",
            "Value": "Orange"
        }
    ]
}
```

Rimozione di tag da una risorsa: untag-resource

Il comando `untag-resource` rimuove le chiavi tag specificate e i relativi valori associati dalla risorsa specificata.

Sintassi

```
aws athena untag-resource --resource-arn
arn:aws:athena:region:account_id:datacatalog/catalog_name --tag-keys
key_name [key_name ...]
```

Il parametro `--resource-arn` specifica la risorsa da cui vengono rimossi i tag. Il parametro `--tag-keys` accetta un elenco separato da spazi di nomi di chiavi. Per ogni nome di chiave specificato, il comando `untag-resource` rimuove sia la chiave che il suo valore.

Nell'esempio seguente vengono rimosse le chiavi `Color` e `Time` nonché i relativi valori dalla risorsa del catalogo `mydatacatalog`.

```
aws athena untag-resource --resource-arn arn:aws:athena:us-
east-1:111122223333:datacatalog/mydatacatalog --tag-keys Color Time
```

Policy di controllo degli accessi IAM basate su tag

La presenza di tag consente di scrivere una policy IAM che include il blocco `Condition` per controllare l'accesso a una risorsa in base ai relativi tag.

Esempi di policy relative ai tag per gruppi di lavoro

Example 1. Policy di tagging di base

La policy IAM seguente consente di eseguire query e interagire con i tag per il gruppo di lavoro denominato `workgroupA`:

```
{
    "Version": "2012-10-17",
    "Statement": [
       {
            "Effect": "Allow",
            "Action": [
                "athena:ListWorkGroups",
                "athena:ListEngineVersions",
                "athena:ListDataCatalogs",
```

```
                "athena:ListDatabases",
                "athena:GetDatabase",
                "athena:ListTableMetadata",
                "athena:GetTableMetadata"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "athena:GetWorkGroup",
                "athena:TagResource",
                "athena:UntagResource",
                "athena:ListTagsForResource",
                "athena:StartQueryExecution",
                "athena:GetQueryExecution",
                "athena:BatchGetQueryExecution",
                "athena:ListQueryExecutions",
                "athena:StopQueryExecution",
                "athena:GetQueryResults",
                "athena:GetQueryResultsStream",
                "athena:CreateNamedQuery",
                "athena:GetNamedQuery",
                "athena:BatchGetNamedQuery",
                "athena:ListNamedQueries",
                "athena:DeleteNamedQuery",
                "athena:CreatePreparedStatement",
                "athena:GetPreparedStatement",
                "athena:ListPreparedStatements",
                "athena:UpdatePreparedStatement",
                "athena:DeletePreparedStatement"
            ],
            "Resource": "arn:aws:athena:us-east-1:123456789012:workgroup/workgroupA"
        }
    ]
}
```

Example 2: Blocco di policy che nega operazioni su un gruppo di lavoro in base alla coppia chiave-valore del tag

I tag associati a una risorsa esistente sono definiti tag di risorsa. I tag delle risorse consentono di scrivere blocchi di policy come i seguenti che negano le operazioni elencate su qualsiasi gruppo di lavoro contrassegnato con una coppia chiave-valore come `stack`, `production`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "athena:GetWorkGroup",
                "athena:UpdateWorkGroup",
                "athena:DeleteWorkGroup",
                "athena:TagResource",
                "athena:UntagResource",
                "athena:ListTagsForResource",
                "athena:StartQueryExecution",
                "athena:GetQueryExecution",
                "athena:BatchGetQueryExecution",
                "athena:ListQueryExecutions",
                "athena:StopQueryExecution",
                "athena:GetQueryResults",
                "athena:GetQueryResultsStream",
                "athena:CreateNamedQuery",
                "athena:GetNamedQuery",
                "athena:BatchGetNamedQuery",
                "athena:ListNamedQueries",
                "athena:DeleteNamedQuery",
                "athena:CreatePreparedStatement",
                "athena:GetPreparedStatement",
                "athena:ListPreparedStatements",
                "athena:UpdatePreparedStatement",
                "athena:DeletePreparedStatement"
            ],
            "Resource": "arn:aws:athena:us-east-1:123456789012:workgroup/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/stack": "production"
                }
            }
        }
    ]
}
```

Example 3: Blocco di policy che limita le richieste di operazioni di modifica tag a determinati tag

I tag passati come parametri alle operazioni che modificano i tag (ad esempio `TagResource`, `UntagResource`, o `CreateWorkGroup` con tag) vengono definiti tag di richiesta. Il seguente blocco di policy di esempio consente l'operazione `CreateWorkGroup` solo se uno dei tag passati ha la chiave `costcenter` e il valore `1`, `2`, o `3`.

> **Note**
>
> Se desideri autorizzare un ruolo IAM a passare i tag come parte di un'operazione `CreateWorkGroup`, assicurati di concedere al ruolo le autorizzazioni per le operazioni `TagResource` e `CreateWorkGroup`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:CreateWorkGroup",
                "athena:TagResource"
            ],
            "Resource": "arn:aws:athena:us-east-1:123456789012:workgroup/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/costcenter": [
                        "1",
                        "2",
                        "3"
                    ]
                }
            }
        }
    ]
}
```

Esempi di policy relative ai tag per cataloghi dati

Example 1. Policy di tagging di base

La seguente policy IAM consente di interagire con i tag per il catalogo dati denominato `datacatalogA`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:ListWorkGroups",
                "athena:ListEngineVersions",
                "athena:ListDataCatalogs",
                "athena:ListDatabases",
                "athena:GetDatabase",
                "athena:ListTableMetadata",
                "athena:GetTableMetadata"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "athena:GetWorkGroup",
                "athena:TagResource",
                "athena:UntagResource",
                "athena:ListTagsForResource",
                "athena:StartQueryExecution",
                "athena:GetQueryExecution",
                "athena:BatchGetQueryExecution",
                "athena:ListQueryExecutions",
                "athena:StopQueryExecution",
                "athena:GetQueryResults",
                "athena:GetQueryResultsStream",
                "athena:CreateNamedQuery",
                "athena:GetNamedQuery",
                "athena:BatchGetNamedQuery",
                "athena:ListNamedQueries",
                "athena:DeleteNamedQuery"
            ],
            "Resource": [
```

```
                "arn:aws:athena:us-east-1:123456789012:workgroup/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "athena:CreateDataCatalog",
                "athena:GetDataCatalog",
                "athena:UpdateDataCatalog",
                "athena:DeleteDataCatalog",
                "athena:ListDatabases",
                "athena:GetDatabase",
                "athena:ListTableMetadata",
                "athena:GetTableMetadata",
                "athena:TagResource",
                "athena:UntagResource",
                "athena:ListTagsForResource"
            ],
            "Resource": "arn:aws:athena:us-
east-1:123456789012:datacatalog/datacatalogA"
        }
    ]
}
```

Example 2: Blocco di policy che nega operazioni su un catalogo dati in base alla coppia chiave-valore del tag

È possibile utilizzare i tag delle risorse per scrivere blocchi di policy che negano operazioni specifiche nei cataloghi dati contrassegnati con coppie chiave-valore di tag specifiche. La seguente policy di esempio nega le operazioni sui cataloghi dati con la coppia chiave-valore di tag `stack`, `production`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "athena:CreateDataCatalog",
                "athena:GetDataCatalog",
                "athena:UpdateDataCatalog",
                "athena:DeleteDataCatalog",
                "athena:GetDatabase",
```

```
                "athena:ListDatabases",
                "athena:GetTableMetadata",
                "athena:ListTableMetadata",
                "athena:StartQueryExecution",
                "athena:TagResource",
                "athena:UntagResource",
                "athena:ListTagsForResource"
            ],
            "Resource": "arn:aws:athena:us-east-1:123456789012:datacatalog/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/stack": "production"
                }
            }
        }
    ]
}
```

Example 3: Blocco di policy che limita le richieste di operazioni di modifica tag a determinati tag

I tag passati come parametri alle operazioni che modificano i tag (ad esempio `TagResource`, `UntagResource`, o `CreateDataCatalog` con tag) vengono definiti tag di richiesta. Il seguente blocco di policy di esempio consente l'operazione `CreateDataCatalog` solo se uno dei tag passati ha la chiave `costcenter` e il valore 1, 2, o 3.

> **Note**
>
> Se desideri autorizzare un ruolo IAM a passare i tag come parte di un'operazione `CreateDataCatalog`, assicurati di concedere al ruolo le autorizzazioni per le operazioni `TagResource` e `CreateDataCatalog`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "athena:CreateDataCatalog",
                "athena:TagResource"
            ],
            "Resource": "arn:aws:athena:us-east-1:123456789012:datacatalog/*",
```

```
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/costcenter": [
                        "1",
                        "2",
                        "3"
                    ]
                }
            }
        }
    ]
}
```

Service Quotas (Quote di Servizio)

> **Note**
>
> La console Service Quotas fornisce le informazioni sulle quote di Amazon Athena. Puoi anche utilizzare la console Service Quotas per richiedere aumenti delle quote per le quote regolabili. Per le limitazioni dello schema relative a AWS Glue , consulta la pagina Endpoint e quote di AWS Glue. Per informazioni generali sulle quote AWS di servizio, vedere le quote AWS di servizio in. Riferimenti generali di AWS

Query

Il tuo account dispone delle seguenti quote relative alle query per Amazon Athena. Per informazioni dettagliate, consulta la pagina Endpoint e quote di Amazon Athena della Riferimenti generali di AWS.

- Active DDL queries (Query DDL attive): il numero di query DDL attive. Le query DDL includono le query `CREATE TABLE` e `ALTER TABLE ADD PARTITION`.
- DDL query timeout (Timeout delle query DDL): il periodo di tempo massimo in minuti per cui una query DDL può essere eseguita prima che venga annullata.
- Active DML queries (Query DML attive): il numero di query DML attive. Le query DML includono le query `SELECT`, `CREATE TABLE AS` (CTAS) e `INSERT INTO`. Le quote specifiche variano in base alla regione AWS .
- Timeout delle query DML: il periodo di tempo massimo in minuti per cui una query DML può essere eseguita prima che venga annullata. Puoi richiedere un aumento di questo timeout fino a un massimo di 240 minuti.

Per richiedere aumenti delle quote, puoi utilizzare la console Service Quotas di Athena.

Athena elabora le query assegnando risorse in base al carico di servizio complessivo e al numero di richieste in entrata. Le query potrebbero essere temporaneamente accodate prima dell'esecuzione. I processi asincroni raccolgono le query dalle code e le eseguono sulle risorse fisiche non appena le risorse diventano disponibili e per tutto il tempo in cui la configurazione dell'account lo consente.

Una quota di query DML o DDL include sia query in esecuzione che in coda. Ad esempio, se la quota di query DML è 25 e il totale delle query in esecuzione e in coda è 26, la query 26 genererà un errore. TooManyRequestsException

> **Note**
>
> Se intendi controllare direttamente la simultaneità delle query eseguite in Athena, puoi utilizzare le prenotazioni della capacità. Per ulteriori informazioni, consulta Gestione della capacità di elaborazione delle query.

Lunghezza della stringa di query

La lunghezza massima consentita per la stringa di query è 262144 byte, dove le stringhe sono codificate in UTF-8. Non si tratta di una quota regolabile. Tuttavia, è possibile ovviare a questa limitazione suddividendo query lunghe in query più piccole. Per ulteriori informazioni, consulta Come faccio ad aumentare la lunghezza massima della stringa di query in Athena? nel Portale del sapere di AWS .

Gruppi di lavoro

Quando si lavora con gruppi di lavoro Athena, ricordare i seguenti punti:

- Le quote di servizio Athena vengono condivise tra tutti i gruppi di lavoro di un account.
- Il numero massimo di gruppi di lavoro che puoi creare per regione in un account è 1000.
- Il numero massimo di istruzioni preparate in un gruppo di lavoro è 1.000.
- Il numero massimo di tag per gruppo di lavoro è 50. Per ulteriori informazioni, consulta Limitazioni applicate ai tag.

Database, tabelle e partizioni

- Se utilizzi il AWS Glue Data Catalog con Athena, consulta AWS Glue endpoint e quote per le quote di servizio su tabelle, database e partizioni, ad esempio il numero massimo di database o tabelle per account.
  - Sebbene Athena supporti l'interrogazione di AWS Glue tabelle con 10 milioni di partizioni, Athena non è in grado di leggere più di 1 milione di partizioni in una singola scansione.
- Se non si utilizza AWS Glue Data Catalog, il numero di partizioni per tabella è 20.000. È possibile richiedere un aumento della quota.

Bucket Amazon S3

Quando utilizzi i bucket Amazon S3, ricorda i seguenti punti:

- Amazon S3 ha una quota di servizio predefinita di 100 bucket per account.
- Athena richiede un bucket separato per registrare i risultati.
- Tuttavia, è possibile richiedere un aumento della quota fino a 1.000 bucket Amazon S3 per account AWS .

Quote di chiamate API per account

Le API Athena hanno le seguenti quote predefinite per il numero di chiamate alle API per account (non per query):

Nome API	Numero predefinito di chiamate al secondo	Capacità di ottimizzazione
`BatchGetNamedQuery` , `ListNamedQueries` , `ListQueryExecutions`	5	Fino a 10
`CreateNamedQuery` , `DeleteNamedQuery` , `GetNamedQuery`	5	Fino a 20
`BatchGetQueryExecution`	20	Fino a 40
`StartQueryExecution` , `StopQueryExecution`	20	Fino a 80

Nome API	Numero predefinito di chiamate al secondo	Capacità di ottimizzazione
`GetQueryExecution` , `GetQueryResults`	100	Fino a 200

Ad esempio, per `StartQueryExecution` è possibile effettuare fino a 20 chiamate al secondo. Inoltre, se questa API non viene chiamata per 4 secondi, il tuo account accumula una capacità di ottimizzazione di un massimo di 80 chiamate. In questo caso, la tua applicazione è in grado di effettuare fino a 80 chiamate a questa API in modalità burst.

Se utilizzi una di queste API e superi la quota predefinita per il numero di chiamate al secondo o la capacità burst del tuo account, l'API Athena emette un errore simile al seguente: "": Si è verificato un errore (ThrottlingException) durante la chiamata all'operazioneClientError: <API_name>Rate exceeded». Riduci il numero di chiamate al secondo o la capacità di ottimizzazione per l'API per questo account.

La quota di Athena delle chiamate API per account non può essere modificata nella console Service Quotas di Athena. Per richiedere un aumento della quota per le chiamate API di Athena, vai alla pagina AWS Support Aumento del limite di servizio, compila e invia il modulo.

Controllo delle versioni del motore di Athena

Athena rilascia occasionalmente una nuova versione del motore per fornire prestazioni migliorate, funzionalità e correzioni di codice. Quando è disponibile una nuova versione del motore, Athena ti avvisa nella console Athena e in AWS Health Dashboard. You ti AWS Health Dashboard invia notifiche sugli eventi che possono influire sui tuoi AWS servizi o sul tuo account. Per ulteriori informazioni su AWS Health Dashboard, vedere Guida introduttiva a. AWS Health Dashboard

Il controllo delle versioni del motore è configurato per gruppo di lavoro. Puoi utilizzare i gruppi di lavoro per controllare il motore di query utilizzato dalle tue query e se consentire ad Athena di aggiornare automaticamente i tuoi gruppi di lavoro. Il motore di query in uso è mostrato nell'editor di query, nella pagina dei dettagli del gruppo di lavoro ed è disponibile tramite le API Athena.

- Per impostazione predefinita, i gruppi di lavoro sono configurati per l'aggiornamento automatico. Quando un gruppo di lavoro è impostato per l'aggiornamento automatico, Athena aggiorna il gruppo di lavoro automaticamente a meno che non rilevi incompatibilità.

- Se configuri un gruppo di lavoro per utilizzare una determinata versione, Athena non cambierà la versione del gruppo di lavoro.

In entrambi i casi, Athena aggiorna i gruppi di lavoro quando una versione non è più disponibile. Athena comunica AWS Health Dashboardall'utente quando una versione del motore non sarà più disponibile. You ti AWS Health Dashboard avvisa in merito a eventi che possono influire sui tuoi AWS servizi o sul tuo account. Per ulteriori informazioni su AWS Health Dashboard, vedere Guida introduttiva a. AWS Health Dashboard

Quando inizi a utilizzare una nuova versione del motore, un piccolo sottoinsieme di query potrebbe interrompersi per cause di incompatibilità. Le ultime modifiche vengono annunciate quando viene rilasciata una nuova versione di Athena. È necessario utilizzare i gruppi di lavoro per testare le query prima dell'aggiornamento, creando un gruppo di lavoro di test che utilizzi il nuovo motore o eseguendo un test di aggiornamento di un gruppo di lavoro esistente. Per ulteriori informazioni, consulta Test delle query in anticipo rispetto all'aggiornamento della versione del motore.

Argomenti

- Modifica delle versioni del motore Athena
- Documentazione di riferimento della versione del motore Athena

Modifica delle versioni del motore Athena

Athena rilascia occasionalmente una nuova versione del motore per fornire prestazioni migliorate, nuove funzionalità e correzioni di codice. Quando è disponibile una nuova versione del motore, Athena ti avvisa nella console. Puoi scegliere se lasciare che Athena decida quando eseguire l'aggiornamento o specificare manualmente una versione del motore Athena per gruppo di lavoro.

Argomenti

- Ricerca della versione del motore di query per un gruppo di lavoro
- Modifica della versione del motore nella console Athena
- Cambiare la versione del motore utilizzando il AWS CLI
- Specifica della versione del motore durante la creazione di un gruppo di lavoro
- Test delle query in anticipo rispetto all'aggiornamento della versione del motore
- Risoluzione dei problemi delle query non riuscite

Ricerca della versione del motore di query per un gruppo di lavoro

È possibile utilizzare anche la pagina Workgroups (Gruppi di lavoro) per trovare la versione corrente del motore per qualsiasi gruppo di lavoro.

Per trovare la versione corrente del motore per qualsiasi gruppo di lavoro

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Nel pannello di navigazione della console Athena, scegli Workgroups (Gruppi di lavoro).
4. Nella pagina Workgroups (Gruppi di lavoro), trova il gruppo di lavoro che desideri. Nella colonna Query engine version (Versione del motore di query) per il gruppo di lavoro viene visualizzata la versione del motore di query.

Modifica della versione del motore nella console Athena

Quando è disponibile una nuova versione del motore, è possibile scegliere se consentire ad Athena di decidere quando aggiornare il gruppo di lavoro o specificare manualmente la versione del motore Athena utilizzata dal gruppo di lavoro. Se attualmente è disponibile solo una versione, non è possibile specificare manualmente una versione diversa.

> **Note**
>
> Per modificare la versione del motore per un gruppo di lavoro, è necessario disporre dell'autorizzazione a eseguire il l'azione `athena:ListEngineVersions` sul gruppo di lavoro. Per esempi di policy IAM, consulta Esempi di policy per i gruppi di lavoro.

Delega ad Athena l'aggiornamento del gruppo di lavoro

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.
3. Nel pannello di navigazione della console, scegli Workgroups (Gruppi di lavoro).
4. Nell'elenco dei gruppi di lavoro, scegli il link per il gruppo di lavoro che desideri configurare.
5. Scegli Modifica.
6. Nella sezione Query engine version (Versione del motore di query), per Update query engine (Aggiornamento del motore di query), scegli Automatic (Automatico) per delegare ad Athena l'aggiornamento del gruppo di lavoro. Si tratta dell'impostazione di default.
7. Seleziona Salvataggio delle modifiche.

   Nell'elenco dei gruppi di lavoro, Query engine update status (Stato di aggiornamento del motore di query) per il gruppo di lavoro mostra Automatic (Automatico).

Per scegliere manualmente una versione del motore

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.
3. Nel pannello di navigazione della console, scegli Workgroups (Gruppi di lavoro).
4. Nell'elenco dei gruppi di lavoro, scegli il link per il gruppo di lavoro che desideri configurare.
5. Scegli Modifica.
6. Nella sezione Query engine version (Versione del motore di query), per Update query engine (Aggiornamento del motore di query), scegli Manual (Manuale) per scegliere manualmente una versione del motore.
7. Utilizza l'opzione Query engine version (Versione del motore di query) per scegliere la versione del motore che intendi utilizzare nel gruppo di lavoro. Non puoi specificare una versione diversa del motore se non è disponibile.

8. Seleziona Salvataggio delle modifiche.

   Nell'elenco dei gruppi di lavoro, Query engine update status (Stato di aggiornamento del motore di query) per il gruppo di lavoro mostra Manual (Manuale).

Cambiare la versione del motore utilizzando il AWS CLI

Per modificare la versione del motore utilizzando il AWS CLI, utilizzate la sintassi riportata nell'esempio seguente.

```
aws athena update-work-group --work-group workgroup-name --configuration-updates
 EngineVersion={SelectedEngineVersion='Athena engine version 3'}
```

Specifica della versione del motore durante la creazione di un gruppo di lavoro

Quando si crea un gruppo di lavoro, è possibile specificare la versione del motore utilizzata dal gruppo di lavoro o consentire ad Athena di decidere quando aggiornare il gruppo di lavoro. Se è disponibile una nuova versione del motore, una best practice consiste nel creare un gruppo di lavoro per testare il nuovo motore prima di aggiornare gli altri gruppi di lavoro. Per specificare la versione del motore per un gruppo di lavoro, è necessario disporre dell'autorizzazione `athena:ListEngineVersions` sul gruppo di lavoro. Per esempi di policy IAM, consulta Esempi di policy per i gruppi di lavoro.

Per specificare la versione del motore quando si crea un gruppo di lavoro

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.
3. Nel pannello di navigazione della console, scegli Workgroups (Gruppi di lavoro).
4. Nella pagina Gruppi di lavoro scegli Crea gruppo di lavoro.
5. Nella pagina Create workgroup (Crea gruppo di lavoro), nella sezione Query engine version (Versione del motore di query), effettua una delle seguenti operazioni:

   - Scegli Automatic (Automatico) per delegare ad Athena l'aggiornamento del gruppo di lavoro. Si tratta dell'impostazione di default.
   - Scegli Manual per scegliere manualmente una versione del motore diversa se disponibile.
6. Inserisci le informazioni per gli altri campi, se necessario. Per informazioni su altri campi, consulta Creare un gruppo di lavoro.

7. Selezionare Create workgroup (Crea gruppo di lavoro).

Test delle query in anticipo rispetto all'aggiornamento della versione del motore

Quando un gruppo di lavoro viene aggiornato a una nuova versione del motore, alcune delle query potrebbero interrompersi a causa di incompatibilità. Per assicurarsi che l'aggiornamento della versione del motore vada senza intoppi, è possibile testare le query in anticipo.

Per testare le query prima dell'aggiornamento della versione del motore

1. Verificare la versione del motore del gruppo di lavoro che si sta utilizzando. La versione del motore che si sta utilizzando viene visualizzata nella pagina Workgroups (Gruppi di lavoro) nella colonna Query engine version (Versione del motore di query) per il gruppo di lavoro. Per ulteriori informazioni, consulta Ricerca della versione del motore di query per un gruppo di lavoro.
2. Crea un gruppo di lavoro di prova che utilizzi la nuova versione del motore. Per ulteriori informazioni, consulta Specifica della versione del motore durante la creazione di un gruppo di lavoro.
3. Utilizza il nuovo gruppo di lavoro per eseguire le query che desideri verificare.
4. Se una query non riesce, utilizzare Documentazione di riferimento della versione del motore Athena per verificare l'interruzione delle modifiche che potrebbero influire sulla query. Alcune modifiche potrebbero richiedere l'aggiornamento della sintassi delle query.
5. Se le domande continuano a fallire, contatta AWS Support per ricevere assistenza. Nella AWS Management Console, scegli Support (Supporto), Support Center (Centro assistenza) oppure fai una domanda su AWS re:Post utilizzando il tag Amazon Athena.

Risoluzione dei problemi delle query non riuscite

Se una query ha esito negativo dopo un aggiornamento della versione del motore, utilizzare Documentazione di riferimento della versione del motore Athena per verificare l'interruzione delle modifiche, incluse le modifiche che potrebbero influire sulla sintassi delle query.

Se le tue domande continuano a fallire, contatta AWS Support per ricevere assistenza. Nella AWS Management Console, scegli Support, Support Center o fai una domanda su AWS re:post utilizzando il tag Amazon Athena.

Documentazione di riferimento della versione del motore Athena

Questa sezione elenca le modifiche apportate al motore di query Athena.

Argomenti

Versione 3 del motore Athena

Per la versione 3 del motore, Athena ha introdotto un approccio di integrazione continua per la gestione del software open source volto a migliorare la simultaneità dei progetti Trino e Presto e ottenere così un accesso più rapido ai miglioramenti da parte della community, integrati e ottimizzati all'interno del motore Athena.

La versione 3 del motore Athena supporta tutte le funzionalità della versione 2. Questo documento evidenzia le principali differenze tra la versione 2 e la versione 3 del motore Athena. Per ulteriori informazioni, consulta l'articolo del Blog sui Big Data di AWS Come fare l'upgrade alla versione 3 del motore Athena per aumentare le prestazioni delle query e accedere a più funzionalità di analisi.

Nozioni di base

Per iniziare, crea un nuovo gruppo di lavoro Athena che utilizza la versione 3 del motore Athena o configura un gruppo di lavoro esistente per l'utilizzo della versione 3. Qualsiasi gruppo di lavoro Athena può eseguire l'aggiornamento dalla versione 2 alla versione 3 del motore, senza compromettere la possibilità di inviare query.

Per ulteriori informazioni, consulta Modifica delle versioni del motore Athena.

Miglioramenti e nuove funzioni

Le caratteristiche e gli aggiornamenti elencati includono sia i miglioramenti apportati da Athena sia le funzionalità incorporate dal progetto open source di Trino. Per un elenco esaustivo degli operatori e delle funzioni di query SQL, consulta la documentazione di Trino.

Funzionalità aggiunte

Supporto dell'algoritmo del bucket di Apache Spark

Athena può leggere i bucket generati dall'algoritmo hash di Spark. Per specificare che i dati sono stati scritti originariamente dall'algoritmo hash di Spark, inserisci `('bucketing_format'='spark')` nella clausola `TBLPROPERTIES` dell'istruzione `CREATE TABLE`. Se questa proprietà non viene specificata, si utilizza l'algoritmo hash di Hive.

```
CREATE EXTERNAL TABLE `spark_bucket_table`(
  `id` int,
  `name` string
  )
CLUSTERED BY (`name`)
INTO 8 BUCKETS
STORED AS PARQUET
LOCATION
  's3://DOC-EXAMPLE-BUCKET/to/bucketed/table/'
TBLPROPERTIES ('bucketing_format'='spark')
```

Funzioni aggiunte

Le funzioni riportate in questa sezione sono nuove nella versione 3 del motore Athena.

Funzioni di aggregazione

listagg(x, separator): Restituisce i valori di input concatenati, separati dalla stringa del separatore.

```
SELECT listagg(value, ',') WITHIN GROUP (ORDER BY value) csv_value
FROM (VALUES 'a', 'c', 'b') t(value);
```

Funzioni di array

contains_sequence(x, seq): Restituisce true se la matrice x contiene tutte le sequenze di matrice come sottoinsieme sequenziale (tutti i valori nello stesso ordine consecutivo).

```
SELECT contains_sequence(ARRAY [1,2,3,4,5,6], ARRAY[1,2]);
```

Funzioni binarie

murmur3 (binary) — Calcola l'hash MurmurHash 3 a 128 bit del file binario.

```
SELECT murmur3(from_base64('aaaaaa'));
```

Funzioni di conversione

format_number(number): Restituisce una stringa formattata utilizzando un simbolo di unità.

```
SELECT format_number(123456); -- '123K'
```

```
SELECT format_number(1000000); -- '1M'
```

Funzioni di data e ora

timezone_hour(timestamp): Restituisce l'ora della differenza del fuso orario dal timestamp.

```
SELECT EXTRACT(TIMEZONE_HOUR FROM TIMESTAMP '2020-05-10 12:34:56 +08:35');
```

timezone_minute(timestamp): Restituisce il minuto della differenza del fuso orario dal timestamp.

```
SELECT EXTRACT(TIMEZONE_MINUTE FROM TIMESTAMP '2020-05-10 12:34:56 +08:35');
```

Funzioni geospaziali

to_encoded_polyline(Geometry): codifica una linestring o un multipunto in una polilinea.

```
SELECT to_encoded_polyline(ST_GeometryFromText(
```

```
   'LINESTRING (-120.2 38.5, -120.95 40.7, -126.453 43.252)'));
```

from_encoded_polyline(varchar): decodifica una polilinea in una linestring.

```
SELECT ST_AsText(from_encoded_polyline('_p~iF~ps|U_ulLnnqC_mqNvxq`@'));
```

to_geojson_geometry (SphericalGeography) — Restituisce la geografia sferica specificata in formato GeoJSON.

```
SELECT to_geojson_geometry(to_spherical_geography(ST_GeometryFromText(
   'LINESTRING (0 0, 1 2, 3 4)')));
```

from_geojson_geometry(varchar): Restituisce l'oggetto di tipo geografia sferica dalla rappresentazione GeoJSON, eliminando chiavi/valori non geometrici. `Feature` e `FeatureCollection` non sono supportati.

```
SELECT
 from_geojson_geometry(to_geojson_geometry(to_spherical_geography(ST_GeometryFromText(
   'LINESTRING (0 0, 1 2, 3 4)'))));
```

geometry_nearest_points(Geometry, Geometry): Restituisce i punti di ogni geometria più vicini tra loro. Se una delle due geometrie è vuota, restituisce NULL. In caso contrario, restituisce una riga di due oggetti `Point` con la distanza minima di due punti qualsiasi sulle geometrie. Il primo punto e il secondo punto provengono rispettivamente dal primo e dal secondo argomento di geometria. Se ci sono più coppie con la stessa distanza minima, una coppia viene scelta arbitrariamente.

```
SELECT geometry_nearest_points(ST_GeometryFromText(
   'LINESTRING (50 100, 50 200)'), ST_GeometryFromText(
   'LINESTRING (10 10, 20 20)'));
```

Funzioni set digest

make_set_digest(x): compone tutti i valori di input di x in un setdigest.

```
SELECT make_set_digest(value) FROM (VALUES 1, 2, 3) T(value);
```

Funzioni stringa

soundex(char): Restituisce una stringa di caratteri che contiene la rappresentazione fonetica di char.

```
SELECT name
FROM nation
WHERE SOUNDEX(name) = SOUNDEX('CHYNA'); -- CHINA
```

concat_ws(string0, string1, ..., stringN): Restituisce la concatenazione di `string1, string2, ..., string`N utilizzando `string0` come separatore. Se `string0` è null, allora il valore restituito è null. Negli argomenti, tutti i valori null indicati dopo il separatore vengono ignorati.

```
SELECT concat_ws(',', 'def', 'pqr', 'mno');
```

Funzioni finestra

GROUPS: aggiunge il supporto per le cornici delle finestre in base ai gruppi.

```
SELECT array_agg(a) OVER(
   ORDER BY a ASC NULLS FIRST GROUPS BETWEEN 1 PRECEDING AND 2 FOLLOWING)
FROM (VALUES 3, 3, 3, 2, 2, 1, null, null) T(a);
```

Miglioramenti in termini di prestazioni.

La versione 3 del motore Athena include alcuni miglioramenti in termini di prestazioni, tra cui i seguenti.

- Recupero più rapido dei metadati AWS Glue delle tabelle: le query che coinvolgono più tabelle ridurranno i tempi di pianificazione delle query.
- Filtro dinamico per JOIN DESTRO: il filtro dinamico è ora abilitato per le proprietà JOIN DESTRO con condizioni di join di uguaglianza, come nell'esempio seguente.

  ```
  SELECT *
  FROM lineitem RIGHT JOIN tpch.tiny.supplier
  ON lineitem.suppkey = supplier.suppkey
  WHERE supplier.name = 'abc';
  ```

- Istruzioni preparate di grandi dimensioni: sono state aumentate fino a 2 MB le dimensioni predefinite dell'intestazione di richiesta/risposta HTTP, per consentire istruzioni preparate di grandi dimensioni.
- approx_percentile (): la funzione `approx_percentile` ora utilizza `tdigest` invece di `qdigest` per recuperare valori quantili approssimativi dalle distribuzioni. Ciò si traduce in prestazioni più

elevate e un minore utilizzo della memoria. Nota che a seguito di questa modifica, la funzione restituisce risultati diversi rispetto alla versione 2 del motore Athena. Per ulteriori informazioni, consulta La funzione approx_percentile restituisce risultati diversi.

Miglioramenti in termini di affidabilità

L'utilizzo generale della memoria del motore e il monitoraggio nella versione 3 del motore Athena sono stati migliorati. Le query di grandi dimensioni sono meno soggette a errori causati dagli arresti anomali dei nodi.

Miglioramenti in termini di sintassi delle query

INTERSECT ALL: è stato aggiunto il supporto per `INTERSECT ALL`.

```
SELECT * FROM (VALUES 1, 2, 3, 4) INTERSECT ALL SELECT * FROM (VALUES 3, 4);
```

EXCEPT ALL: è stato aggiunto il supporto per `EXCEPT ALL`.

```
SELECT * FROM (VALUES 1, 2, 3, 4) EXCEPT ALL SELECT * FROM (VALUES 3, 4);
```

RANGE PRECEDING: è stato aggiunto il supporto per `RANGE PRECEDING` nelle funzioni finestra.

```
SELECT sum(x) over (order by x range 1 preceding)
FROM (values (1), (1), (2), (2)) t(x);
```

MATCH_RECOGNIZE: è stato aggiunto il supporto per la corrispondenza dei modelli di riga, come nell'esempio seguente.

```
SELECT m.id AS row_id, m.match, m.val, m.label
FROM (VALUES(1, 90),(2, 80),(3, 70),(4, 70)) t(id, value)
MATCH_RECOGNIZE (
        ORDER BY id
        MEASURES match_number() AS match,
        RUNNING LAST(value) AS val,
        classifier() AS label
        ALL ROWS PER MATCH
        AFTER MATCH SKIP PAST LAST ROW
        PATTERN (() | A) DEFINE A AS true
) AS m;
```

Miglioramenti in termini di formato e tipo di dati

La versione 3 del motore Athena presenta i seguenti miglioramenti in termini di formato e tipo di dati.

- LZ4 e ZSTD: è stato aggiunto il supporto per la lettura di dati Parquet compressi LZ4 e ZSTD. È stato aggiunto il supporto per la scrittura di dati ORC compressi ZSTD.
- Tabelle basate su collegamenti simbolici: è stato aggiunto il supporto per la creazione di tabelle basate su collegamenti simbolici per file Avro. Di seguito è riportato un esempio.

```
CREATE TABLE test_avro_symlink
ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.avro.AvroSerDe'
...
INPUTFORMAT 'org.apache.hadoop.hive.ql.io.SymlinkTextInputFormat'
```

- SphericalGeography— Il SphericalGeography tipo fornisce il supporto nativo per le caratteristiche spaziali rappresentate su coordinate geografiche (a volte chiamate coordinate geodetiche, lat/lon o lon/lat). Le coordinate geografiche sono coordinate sferiche espresse in unità angolari (gradi).

  La funzione `to_spherical_geography` restituisce le coordinate geografiche (sferiche) a partire da coordinate geometriche (piane), come nell'esempio seguente.

```
SELECT to_spherical_geography(ST_GeometryFromText(
   'LINESTRING (-40.2 28.9, -40.2 31.9, -37.2 31.9)'));
```

Modifiche importanti

Quando si esegue la migrazione dalla versione 2 del motore Athena alla versione 3, alcune modifiche possono influire sullo schema della tabella, sulla sintassi o sull'utilizzo del tipo di dati. Questa sezione elenca i messaggi di errore associati e offre alcuni suggerimenti per le soluzioni alternative.

Modifiche alla sintassi delle query

IGNORE NULLS non può essere utilizzato con funzioni di finestra senza valori

Messaggio di errore: Impossibile specificare la clausola di trattamento null per la funzione `bool_or`.

Causa: ora `IGNORE NULLS` può essere utilizzato solo con le funzioni di valore `first_value`, `last_value`, `nth_value`, `lead` e `lag`. Questa modifica è stata apportata per conformarsi alle specifiche SQL ANSI.

Soluzione consigliata: rimuovi `IGNORE NULLS` dalle funzioni finestra prive di valori nelle stringhe di query.

La funzione CONCAT deve avere due o più argomenti

Messaggio di errore: INVALID_FUNCTION_ARGUMENT: There must be two or more concatenation arguments (Argomento della funzione non valido: devono essere presenti due o più argomenti di concatenazione)

Causa: in precedenza, la funzione della stringa `CONCAT` accettava un singolo argomento. Nella versione 3 del motore Athena, la funzione `CONCAT` richiede un minimo di due argomenti.

Soluzione consigliata: modifica le occorrenze di `CONCAT(str)` in `CONCAT(str, '')`.

Nella versione 3 del motore Athena, le funzioni non possono avere più di 127 argomenti. Per ulteriori informazioni, consulta Troppi argomenti per la chiamata di funzione.

La funzione approx_percentile restituisce risultati diversi

La funzione `approx_percentile` restituisce risultati diversi nella versione 3 del motore Athena rispetto alla versione 2 del motore Athena.

Messaggio di errore: nessuno.

Causa: la funzione `approx_percentile` è soggetta a modifiche di versione.

> **Important**
>
> Poiché i risultati della funzione `approx_percentile` sono approssimazioni e le approssimazioni sono soggette a modifiche da una versione all'altra, non è consigliabile fare affidamento sulla funzione `approx_percentile` per applicazioni critiche.

Soluzione consigliata: per approssimare il comportamento della versione 2 del motore Athena, puoi utilizzare un set diverso di funzioni `approx_percentile` nella versione 3 del motore Athena. Ad esempio, supponi di avere la seguente query nella versione 2 del motore Athena:

```
SELECT approx_percentile(somecol, 2E-1)
```

Per ottenere approssimativamente lo stesso risultato nella versione 3 del motore Athena, puoi provare le funzioni `qdigest_agg` e `value_at_quantile`, come nell'esempio seguente. Tieni presente che, anche con questa soluzione alternativa, lo stesso comportamento non è garantito.

```
SELECT value_at_quantile(qdigest_agg(somecol, 1), 2E-1)
```

La funzione geospaziale non supporta l'input varbinary

Messaggio di errore: FUNCTION_NOT_FOUND for st_XXX (Funzione non trovata per st_XXX)

Causa: alcune funzioni geospaziali non supportano più il tipo di input `VARBINARY` legacy o le firme delle funzioni correlate al testo.

Soluzione consigliata: utilizza le funzioni geospaziali per convertire i tipi di input in tipi supportati. I tipi di input supportati sono indicati nel messaggio di errore.

Nelle clausole GROUP BY, le colonne annidate devono essere racchiuse tra virgolette

Messaggio di errore "*`column_name`*"."*`nested_column`*" deve essere un'espressione aggregata o apparire nella clausola GROUP BY

Causa: la versione 3 del motore Athena richiede che i nomi delle colonne annidate nelle clausole `GROUP BY` siano tra virgolette doppie. Ad esempio, la seguente query produce l'errore perché, nella clausola `GROUP BY`, `user.name` non è tra virgolette.

```
SELECT "user"."name" FROM dataset
GROUP BY user.name
```

Soluzione consigliata: inserisci le virgolette doppie attorno ai nomi delle colonne annidate nelle clausole `GROUP BY`, come nell'esempio seguente.

```
SELECT "user"."name" FROM dataset
GROUP BY "user"."name"
```

FilterNode Errore imprevisto durante l'utilizzo di OPTIMIZE su una tabella Iceberg

Messaggio di errore: è stato FilterNoderilevato un problema imprevisto nel piano; probabilmente il connettore non è stato in grado di gestire l'espressione WHERE fornita.

Causa: l'`OPTIMIZE`istruzione eseguita sulla tabella Iceberg utilizzava una `WHERE` clausola che includeva una colonna non di partizione nell'espressione di filtro.

Soluzione consigliata: l'`OPTIMIZE`istruzione supporta il filtraggio solo per partizioni. Quando esegui `OPTIMIZE` su tabelle partizionate, includi solo le colonne delle partizioni nella clausola. `WHERE` Se eseguite `OPTIMIZE` su una tabella non partizionata, non specificate alcuna clausola. `WHERE`

Ordine degli argomenti della funzione Log()

Nella versione 2 del motore Athena, l'ordine degli argomenti per la funzione `log()` era `log(`*`value`*`,` *`base`*`)`. Nella versione 3 del motore Athena, questo l'ordine è cambiato in `log(`*`base`*`, `*`value`*`)` secondo gli standard SQL.

La funzione minute() non supporta gli intervalli anno-mese

Messaggio di errore: Unexpected parameters (interval year to month) for function minute. (Parametri non previsti [intervallo anno-mese] per i minuti della funzione.) Expected: minute(timestamp with time zone) , minute(time with time zone) , minute(timestamp) , minute(time) , minute(interval day to second). (Valori previsti: minute[timestamp with time zone], minute[time with time zone], minute[timestamp], minute[time], minute[interval day to second])

Causa: nella versione 3 del motore Athena, i controlli relativi al tipo di dati sono stati resi più precisi per `EXTRACT` in base alle specifiche SQL ANSI.

Soluzione consigliata: aggiorna le query per assicurarti che i tipi corrispondano alle firme delle funzioni suggerite.

Le espressioni ORDER BY devono essere visualizzate nell'elenco SELECT

Messaggio di errore: For SELECT DISTINCT, ORDER BY expressions must appear in SELECT list (Per SELECT DISTINCT, le espressioni ORDER BY devono essere visualizzate nell'elenco SELECT)

Causa: in una clausola `SELECT` viene utilizzato un alias di tabella errato.

Soluzione consigliata: verifica che tutte le colonne dell'espressione `ORDER BY` contengano i riferimenti corretti nella clausola `SELECT DISTINCT`.

Errore di query durante il confronto di più colonne restituite da una sottoquery

Esempio di messaggio di errore: l'espressione del valore e il risultato della sottoquery devono essere dello stesso tipo: row(varchar, varchar) vs row(row(varchar, varchar))

Causa: A causa di un aggiornamento della sintassi nella versione 3 del motore Athena, questo errore si verifica quando una query tenta di confrontare più valori restituiti da una sottoquery e l'istruzione `SELECT` della sottoquery racchiude l'elenco di colonne tra parentesi, come nell'esempio seguente.

```
SELECT *
FROM table1
WHERE (t1_col1, t1_col2)
IN (SELECT (t2_col1, t2_col2) FROM table2)
```

Soluzione: nella versione 3 del motore Athena, rimuovi la parentesi attorno all'elenco di colonne nell'istruzione `SELECT` della sottoquery, come nella seguente query di esempio aggiornata.

```
SELECT *
FROM table1
WHERE (t1_col1, t1_col2)
IN (SELECT t2_col1, t2_col2 FROM table2)
```

SKIP è una parola riservata per le query DML

La parola `SKIP` ora è una parola riservata per le query DML come `SELECT`. Per utilizzare la parola `SKIP` come identificatore in una query DML, racchiuderla tra virgolette doppie.

Per ulteriori informazioni sulle parole riservate in Athena, consulta Parole chiave riservate.

Le clausole SYSTEM_TIME e SYSTEM_VERSION sono obsolete per le query temporali

Messaggio di errore: mismatched input 'SYSTEM_TIME'. (Input 'SYSTEM_TIME' non corrispondente.) Expecting (Valore atteso): 'TIMESTAMP', 'VERSION'

Causa: nella versione 2 del motore Athena, le tabelle Iceberg utilizzavano le clausole `FOR SYSTEM_TIME AS OF` e `FOR SYSTEM_VERSION AS OF` per le query temporali su timestamp e versione. La versione 3 del motore Athena utilizza le clausole `FOR TIMESTAMP AS OF` e `FOR VERSION AS OF`.

Soluzione consigliata: aggiorna la query SQL per utilizzare le clausole `TIMESTAMP AS OF` e `VERSION AS OF` per le operazioni temporali, come negli esempi seguenti.

Query temporale per timestamp:

```
SELECT * FROM TABLE FOR TIMESTAMP AS OF (current_timestamp - interval '1' day)
```

Query temporale per versione:

```
SELECT * FROM TABLE FOR VERSION AS OF 949530903748831860
```

Numero di argomenti eccessivo per un costruttore di array

Messaggio di errore: TOO_MANY_ARGUMENTS: Numero di argomenti eccessivo per il costruttore di array.

Causa: il numero massimo di elementi in un costruttore di array è ora impostato a 254.

Soluzione consigliata: suddividi gli elementi in più matrici con 254 o meno elementi ciascuna e utilizza la funzione `CONCAT` per concatenare le matrici, come nell'esempio seguente.

```
CONCAT(
ARRAY[x1,x2,x3...x254],
ARRAY[y1,y2,y3...y254],
...
)
```

Identificatore delimitato di lunghezza zero non consentito

Messaggio di errore: Zero-length delimited identifier not allowed. (Identificatore delimitato di lunghezza zero non consentito.)

Causa: una query utilizzava una stringa vuota come alias di colonna.

Soluzione consigliata: aggiorna la query in modo che utilizzi un alias non vuoto per la colonna.

Modifiche a livello di elaborazione dati

Convalida del bucket

Messaggio di errore: HIVE_INVALID_BUCKET_FILES: la tabella Hive è danneggiata.

Causa: la tabella potrebbe essere danneggiata. Per garantire la correttezza delle query per le tabelle con periodi fissi, la versione 3 del motore Athena consente una verifica aggiuntiva sulle tabelle con periodi fissi per garantire la correttezza delle query ed evitare errori imprevisti durante il runtime.

Soluzione consigliata: ricrea la tabella utilizzando la versione 3 del motore Athena.

La trasmissione di uno struct in JSON ora restituisce i nomi dei campi

Quando trasmetti uno `struct` a JSON in una query `SELECT` nella versione 3 del motore Athena, la trasmissione ora restituisce sia i nomi dei campi che i valori (ad esempio `"useragent":null`) anziché solo i valori (ad esempio, `null`).

Modifica dell'applicazione della sicurezza a livello di colonna delle tabelle Iceberg

Messaggio di errore: Access Denied: Cannot select from columns (Accesso negato: impossibile selezionare dalle colonne)

Causa: la tabella Iceberg è stata creata all'esterno di Athena e utilizza una versione dell'SDK di Apache Iceberg precedente alla 0.13.0. Poiché le versioni precedenti dell'SDK non compilavano le colonne in AWS Glue, Lake Formation non era in grado di determinare le colonne autorizzate per l'accesso.

Soluzione consigliata: esegui un aggiornamento utilizzando l'istruzione ALTER TABLE SET PROPERTIES di Athena o utilizza l'ultima versione dell'SDK di Iceberg per correggere la tabella e aggiornare le informazioni della colonna in AWS Glue.

I valori Null nei tipi di dati elenco vengono ora propagati alle UDF

Messaggio di errore: Null Pointer Exception (Eccezione del puntatore null)

Causa: questo problema può verificarsi se utilizzi il connettore UDF con una funzione Lambda definita dall'utente.

La versione 2 del motore Athena ha filtrato i valori null nei tipi di dati elenco che sono stati trasferiti a una funzione definita dall'utente. Nella versione 3 del motore Athena, i valori null vengono ora mantenuti e trasmessi all'UDF. Ciò può causare un'eccezione del puntatore null se la funzione definita dall'utente tenta di annullare l'elemento null senza eseguire una verifica.

Ad esempio, se i dati `[null, 1, null, 2, 3, 4]` sono presenti in un'origine dei dati come DynamoDB, alla funzione Lambda definita dall'utente vengono tramessi gli elementi seguenti:

Versione 2 del motore Athena: `[1,2,3,4]`

Versione 3 del motore Athena: `[null, 1, null, 2, 3, 4]`

Soluzione consigliata: assicurati che la funzione Lambda definita dall'utente gestisca gli elementi null per i tipi di dati elenco.

Le sottostringhe delle matrici di caratteri non contengono più spazi riempiti

Messaggio di errore: non viene generato alcun errore, ma la stringa restituita non contiene più spazi riempiti. Ad esempio, `substr(char[20],1,100)` ora restituisce una stringa con lunghezza 20 anziché 100.

Soluzione consigliata: non è richiesta alcuna azione.

Coercizione del tipo di colonna decimale non supportata

Messaggi di errore*: HIVE_CURSOR_ERROR: Impossibile leggere il file Parquet: s3://DOC-EXAMPLE-BUCKET/ path/file_name .parquet o Unsupported column type (varchar) per la colonna Parquet ([column_name]*

Causa: la versione 2 del motore Athena ha avuto successo occasionalmente (ma spesso ha avuto esito negativo) nel tentativo di coercizione dei tipi di dati da `varchar` al numero decimale. Poiché la versione 3 del motore Athena dispone di una convalida del tipo che verifica che questo sia compatibile prima di provare a leggere il valore, questi tentativi di coercizione ora falliscono sempre.

Soluzione consigliata: sia per Athena engine versione 2 che per Athena engine versione 3, modificate lo schema in modo AWS Glue da utilizzare un tipo di dati numerico anziché `varchar` per le colonne decimali nei file Parquet. Scansiona nuovamente i dati e assicurati che il nuovo tipo di dati della colonna sia di tipo decimale oppure ricrea manualmente la tabella in Athena e usa la sintassi `decimal(`*`precision`*`, `*`scale`*`)` per specificare un tipo di dati per la colonna decimal.

I valori Float o double NaN non possono più essere convertiti a bigint

Messaggio di errore: INVALID_CAST_ARGUMENT: impossibile convertire real/double NaN in bigint

Causa: nella versione 3 del motore Athena, `NaN` non può più convertire a 0 come `bigint`.

Soluzione consigliata: assicurati che i valori `NaN` non siano presenti nelle colonne `float` o `double` quando converti in `bigint`.

modifica del tipo di ritorno della funzione uuid ()

Il problema seguente riguarda sia le tabelle che le viste.

Messaggio di errore: Tipo di Hive non supportato: uuid

Causa: nella versione 2 del motore Athena, la funzione `uuid()` restituiva una stringa, ma nella versione 3 del motore Athena restituisce un UUID generato in modo pseudo casuale (tipo 4). Poiché il tipo di dati della colonna UUID non è supportato in Athena, la funzione `uuid()` non può più essere utilizzata direttamente nelle query CTAS per generare colonne UUID nella versione 3 del motore Athena.

Ad esempio, la seguente istruzione `CREATE TABLE` viene completata correttamente nella versione 2 del motore Athena ma restituisce NOT_SUPPORTED: tipo di Hive non supportata: uuid nella versione 3 del motore Athena:

```
CREATE TABLE uuid_table AS
   SELECT uuid() AS myuuid
```

Analogamente, la seguente istruzione `CREATE VIEW` viene completata correttamente nella versione 2 del motore Athena ma restituisce Tipo di colonna non valido per la colonna myuuid: Tipo di Hive non supportato: uuid nella versione 3 del motore Athena:

```
CREATE VIEW uuid_view AS
   SELECT uuid() AS myuuid
```

Quando su una visualizzazione così creata nella versione 2 del motore Athena viene eseguita una query nella versione 3 del motore Athena, si verifica un errore simile al seguente:

VIEW_IS_STALE: line 1:15: View 'awsdatacatalog.mydatabase.uuid_view' è obsoleta o in stato non valido: la colonna [myuuid] di tipo uuid proiettata dalla visualizzazione della query nella posizione 0 non può essere forzata alla colonna [myuuid] di tipo varchar memorizzata nella definizione della visualizzazione

Soluzione consigliata: quando crei la tabella o la visualizzazione, utilizza la funzione `cast()` per convertire l'output di `uuid()` in `varchar`, come negli esempi seguenti:

```
CREATE TABLE uuid_table AS
   SELECT CAST(uuid() AS VARCHAR) AS myuuid
```

```
CREATE VIEW uuid_view AS
   SELECT CAST(uuid() AS VARCHAR) AS myuuid
```

Problemi di coercizione CHAR e VARCHAR

Usa le soluzioni alternative in questa sezione se riscontri problemi di coercizione con `varchar` e `char` nella versione 3 del motore Athena. Se non riesci a utilizzare queste soluzioni alternative, contatta. AWS Support

Errore della funzione CONCAT con ingressi misti CHAR e VARCHAR

Problema: la seguente query ha esito positivo sulla versione 2 del motore Athena.

```
SELECT concat(CAST('abc' AS VARCHAR(20)), '12', CAST('a' AS CHAR(1)))
```

Tuttavia, nella versione 3 del motore Athena, la stessa query ha esito negativo con quanto segue:

Messaggio di errore: FUNCTION_NOT_FOUND: riga 1:8: Parametri imprevisti (varchar(20), varchar(2), char(1)) per la funzione concat. Previsto: concat(char(x), char(y)), concat(array(E), E) E, concat(E, array(E)) E, concat(array(E)) E, concat(varchar), concat(varbinary)

Soluzione consigliata: quando utilizzi la funzione `concat`, converti su `char` o `varchar`, ma non su una combinazione di entrambi.

SQL || errore di concatenazione con gli input CHAR e VARCHAR

Nella versione 3 del motore Athena, l'operatore di concatenazione `||` a doppia barra verticale richiede input come input `varchar`. Gli input non possono essere una combinazione di tipi `varchar` e `char`.

Messaggio di errore: TYPE_NOT_FOUND: riga 1:26: Tipo sconosciuto: char(65537)

Causa: una query che utilizza `||` per concatenare `char` e `varchar` può generare l'errore, come nell'esempio seguente.

```
SELECT CAST('a' AS CHAR) || CAST('b' AS VARCHAR)
```

Soluzione consigliata: concatena `varchar` con `varchar`, come nell'esempio seguente.

```
SELECT CAST('a' AS VARCHAR) || CAST('b' AS VARCHAR)
```

Errore nelle query CHAR e VARCHAR UNION

Messaggio di errore: NOT_SUPPORTED: tipo di hive non supportato: char (65536). Tipi di CHAR supportati: CHAR (<=255)

Causa: una query che tenta di combinare `char` e `varchar`, come nell'esempio seguente:

```
CREATE TABLE t1 (c1) AS SELECT CAST('a' as CHAR) as c1 UNION ALL SELECT CAST('b' AS
 VARCHAR) AS c1
```

Soluzione consigliata: nella query di esempio, converti `'a'` come `varchar` anzichéchar.

Spazi vuoti indesiderati dopo la coercizione CHAR o VARCHAR

Nella versione 3 del motore Athena, quando i dati di `char(X)` e `varchar` vengono forzati a un unico tipo quando formano una matrice o una singola colonna, `char(65535)` è il tipo di destinazione e ogni campo contiene molti spazi finali indesiderati.

Causa: la versione 3 del motore Athena costringe `varchar` e `char(X)` a `char(65535)`, quindi riempe a destra i dati con spazi.

Soluzione consigliata: converti esplicitamente ogni campo in `varchar`.

Modifiche al timestamp

Modifica del comportamento durante la trasmissione di un timestamp con fuso orario varchar

Nella versione 2 del motore Athena, la trasmissione di un `Timestamp` con fuso orario `varchar` causava la modifica di alcuni valori letterali del fuso orario (ad esempio, `US/Eastern` modificato in `America/New_York`). Questo comportamento non si verifica nella versione 3 del motore Athena.

L'overflow del timestamp della data genera un errore

Messaggio di errore: Millis overflow: XXX (Overflow di millisecondi: XXX)

Causa: poiché le date ISO8601 non erano controllate per l'overflow nella versione 2 del motore Athena, alcune date hanno prodotto un timestamp negativo. La versione 3 del motore Athena verifica la presenza di questo overflow e genera un'eccezione.

Soluzione consigliata: assicurati che il timestamp sia compreso nell'intervallo.

Fusi orari politici con TIME non supportati

Messaggio di errore: INVALID LITERAL (Valore letterale non valido)

Causa: query come `SELECT TIME '13:21:32.424 America/Los_Angeles'`.

Soluzione consigliata: evita di utilizzare fusi orari politici con `TIME`.

La mancata corrispondenza di precisione nelle colonne Timestamp causa un errore di serializzazione

Messaggio di errore: SERIALIZATION_ERROR: Could not serialize column '*COLUMNZ*' of type 'timestamp(3)' at position *X*:*Y* (Errore di serializzazione: impossibile serializzare la colonna "COLUMNZ" di tipo "timestamp(3)" nella posizione X:Y)

COLUMNZ è il nome di output della colonna che causa il problema. I numeri *X*:*Y* indicano la posizione della colonna nell'output.

Causa: la versione 3 del motore Athena verifica che la precisione dei timestamp nei dati corrisponda a quella specificata per il tipo di dati della colonna nelle specifiche della tabella. Attualmente, questa

precisione è sempre pari a 3. Se i dati hanno una precisione maggiore, le query hanno esito negativo e viene restituito l'errore.

Soluzione consigliata: controlla i dati per assicurarti che i timestamp abbiano una precisione al millisecondo.

Precisione del timestamp errata nelle query UNLOAD e CTAS per le tabelle Iceberg

Messaggio di errore: precisione del timestamp errata per il timestamp (6); la precisione configurata è MILLISECONDI

Causa: la versione 3 del motore Athena verifica che la precisione dei timestamp nei dati corrisponda a quella specificata per il tipo di dati della colonna nelle specifiche della tabella. Attualmente, questa precisione è sempre pari a 3. Se i dati hanno una precisione maggiore di questa (ad esempio, microsecondi anziché millisecondi), le query possono avere esito negativo con l'errore rilevato.

Soluzione: per risolvere questo problema, innanzitutto `CAST` la precisione del timestamp a 6, come nel seguente esempio CTAS che crea una tabella Iceberg. Nota che la precisione deve essere specificata come 6 anziché 3 per evitare l'errore Precisione (3) timestamp non supportata per Iceberg.

```
CREATE TABLE my_iceberg_ctas
WITH (table_type = 'ICEBERG', location = 's3://DOC-EXAMPLE-BUCKET/table_ctas/',
format = 'PARQUET')
AS SELECT id, CAST(dt AS timestamp(6)) AS "dt"
FROM my_iceberg
```

Quindi, poiché Athena non supporta il timestamp 6, converti nuovamente il valore in timestamp (ad esempio, in una visualizzazione). Il seguente esempio crea una visualizzazione dalla tabella `my_iceberg_ctas`.

```
CREATE OR REPLACE VIEW my_iceberg_ctas_view AS
SELECT cast(dt AS timestamp) AS dt
FROM my_iceberg_ctas
```

Ora la lettura del valore di tipo Long come timestamp o viceversa causa nei file ORC un errore indicante un file ORC non valido

Messaggio di errore: Error opening Hive split 'FILE (SPLIT POSITION)' Malformed ORC file. (Errore durante l'apertura del file ORC non valido "FILE [SPLIT POSITION]" della suddivisione di Hive.)

Cannot read SQL type timestamp from ORC stream .long_type of type LONG (Impossibile leggere il timestamp di tipo SQL dal flusso ORC .long_type di tipo LONG)

Causa: la versione 3 del motore Athena ora rifiuta la coercizione implicita dal tipo di dati `Long` a `Timestamp` o da `Timestamp` a `Long`. In precedenza, i valori `Long` venivano convertiti implicitamente in timestamp come se fossero millisecondi di epoca.

Soluzione consigliata: utilizza la funzione `from_unixtime` per trasmettere in modo esplicito la colonna o utilizza la funzione `from_unixtime` per creare una colonna aggiuntiva per le query future.

Ora e intervallo anno-mese non supportati

Messaggio di errore: TYPE MISMATCH (Tipo non corrispondente)

Causa: la versione 3 del motore Athena non supporta l'ora e l'intervallo anno-mese (ad esempio, `SELECT TIME '01:00' + INTERVAL '3' MONTH`).

Overflow del timestamp per il formato Parquet int96

Messaggio di errore: Nanos non valido timeOfDay

Causa: overflow del timestamp per il formato Parquet `int96`.

Soluzione consigliata: identifica i file specifici che presentano il problema. Quindi genera nuovamente il file di dati con una up-to-date libreria Parquet ben nota o usa Athena CTAS. Se il problema persiste, contatta l'assistenza di Athena indicando il modo in cui vengono generati i file di dati.

Spazio richiesto tra i valori di data e ora per la conversione da una stringa a un timestamp

Messaggio di errore: INVALID_CAST_ARGUMENT: impossibile convertire il valore in timestamp.

Causa: la versione 3 del motore Athena non accetta più un trattino come separatore valido tra i valori di data e ora nella stringa di input di `cast`. Ad esempio, la seguente query funziona nella versione 2 del motore Athena ma non nella versione 3 del motore Athena:

```
SELECT CAST('2021-06-06-23:38:46' AS timestamp) AS this_time
```

Soluzione consigliata: nella versione 3 del motore Athena, sostituisci il trattino tra la data e l'ora con uno spazio, come nell'esempio seguente.

```
SELECT CAST('2021-06-06 23:38:46' AS timestamp) AS this_time
```

modifica del valore di ritorno del timestamp to_iso8601 ()

Messaggio di errore: nessuno

Causa: nella versione 2 del motore Athena, la funzione `to_iso8601` restituisce un timestamp con fuso orario anche se il valore passato alla funzione non include il fuso orario. Nella versione 3 del motore Athena, la funzione `to_iso8601` restituisce un timestamp con fuso orario solo quando l'argomento passato include il fuso orario.

Ad esempio, la seguente query passa la data corrente alla funzione `to_iso8601` due volte: prima come timestamp con fuso orario e poi come timestamp.

```
SELECT TO_ISO8601(CAST(CURRENT_DATE AS TIMESTAMP WITH TIME ZONE)),
 TO_ISO8601(CAST(CURRENT_DATE AS TIMESTAMP))
```

L'output seguente mostra il risultato della query in ogni motore.

Versione 2 del motore Athena:

#	_col0	_col1
1	2023-02-24T00:00:00.000Z	2023-02-24T00:00:00.000Z

Versione 3 del motore Athena:

#	_col0	_col1
1	2023-02-24T00:00:00.000Z	2023-02-24T00:00:00.000

Soluzione consigliata: per replicare il comportamento precedente, puoi passare il valore del timestamp alla funzione `with_timezone` prima di passarlo `ato_iso8601`, come nell'esempio seguente:

```
SELECT to_iso8601(with_timezone(TIMESTAMP '2023-01-01 00:00:00.000', 'UTC'))
```

Risultato

#	_col0
1	2023-01-01T00:00:00.000Z

at_timezone () il primo parametro deve specificare una data

Problema: nella versione 3 del motore Athena, la funzione `at_timezone` non può assumere un valore `time_with_timezone` come primo parametro.

Causa: senza informazioni sulla data, non è possibile determinare se il valore passato è l'ora legale o l'ora solare. Ad esempio, `at_timezone('12:00:00 UTC', 'America/Los_Angeles')` è ambiguo poiché non è possibile determinare se il valore passato è Pacific Daylight Time (PDT) o Pacific Standard Time (PST).

Limitazioni

La versione 3 del motore Athena presenta le limitazioni seguenti.

- Prestazioni delle query: molte query vengono eseguite più velocemente sulla versione 3 del motore Athena, ma alcuni piani di query possono differire dalla versione 2. Di conseguenza, alcune query possono differire in termini di latenza o costi.
- Connettori Trino e Presto: i connettori Trino e Presto non sono supportati. Utilizzare Amazon Athena Federated Query per collegare le origini dati. Per ulteriori informazioni, consulta Utilizzo di Amazon Athena Federated Query.
- Esecuzione a tolleranza di errore: l'esecuzione a tolleranza di errore di Trino (Trino Tardigrade) non è supportata.
- Limite dei parametri della funzione: le funzioni non possono richiedere più di 127 parametri. Per ulteriori informazioni, consulta Troppi argomenti per la chiamata di funzione.

I seguenti limiti sono stati introdotti nella versione 2 del motore Athena per garantire che le query non abbiano esito negativo a causa di limitazioni di risorse. Questi limiti non sono configurabili dagli utenti.

- Numero degli elementi del risultato— Il numero di elementi del risultato n è limitato a 10.000 o meno per le seguenti funzioni: `min(col, n)`, `max(col, n)`, `min_by(col1, col2, n)` e `max_by(col1, col2, n)`.
- GROUPING SETS – Il numero massimo di sezioni in un gruppo di raggruppamenti è 2048.

- Lunghezza massima della riga del file di testo: la lunghezza massima della riga predefinita per i file di testo è 200 MB.
- Dimensione massima del risultato della funzione di sequenza — La dimensione massima del risultato di una funzione di sequenza è 50.000 voci. Ad esempio, `SELECT sequence(0,45000,1)` ha esito positivo, ma `SELECT sequence(0,55000,1)` ha esito negativo con il messaggio di errore Il risultato della funzione di sequenza non deve avere più di 50.000 voci. Questo limite è valido per tutti i tipi di input per funzioni di sequenza, inclusi i le marche temporali.

Versione 2 del motore Athena

La versione 2 del motore Athena introduce le seguenti modifiche.

- Miglioramenti e nuove funzioni
  - Miglioramenti a raggruppamento, unione e sottoquery
  - Miglioramenti al tipo di dati
  - Funzioni aggiunte
  - Miglioramenti in termini di prestazioni.
  - Miglioramenti relativi a JSON
- Modifiche importanti
  - Correzioni di bug
  - Modifiche alle funzioni geospaziali
  - Conformità ad ANSI SQL
  - Funzioni sostituite
  - Limiti

Miglioramenti e nuove funzioni

- EXPLAIN ed EXPLAIN ANALYZE: è possibile utilizzare l'istruzione `EXPLAIN` in Athena per visualizzare il piano di esecuzione per le query SQL. Utilizzare `EXPLAIN ANALYZE` per visualizzare il piano di esecuzione distribuito per le query SQL e il costo di ogni operazione. Per ulteriori informazioni, consulta Utilizzo di EXPLAIN e EXPLAIN ANALYZE in Athena.
- Query federate— Le query federate sono supportate nella versione 2 del motore Athena. Per ulteriori informazioni, consulta Utilizzo di Amazon Athena Federated Query.

- Funzioni geospaziali— Sono state aggiunte più di 25 funzioni geospaziali. Per ulteriori informazioni, consulta Nuove funzioni geospaziali nella versione 2 del motore Athena.
- Schema nidificato— È stato aggiunto il supporto per la lettura dello schema nidificato, che riduce i costi.
- Istruzioni preparate: utilizzare le istruzioni preparate per l'esecuzione ripetuta della stessa query con parametri di query diversi. Un'istruzione preparata contiene parametri segnaposto i cui valori vengono passati in fase di runtime. Le istruzioni preparate aiutano a prevenire attacchi SQL injection. Per ulteriori informazioni, consulta Utilizzo di query parametrizzate.
- Supporto dell'evoluzione dello schema— È stato aggiunto il supporto dell'evoluzione dello schema per i dati in formato Parquet.
  - Aggiunto il supporto per la lettura di colonne di tipo array, mappa o riga da partizioni in cui lo schema di partizione è diverso dallo schema della tabella. Ciò può verificarsi quando lo schema della tabella è stato aggiornato dopo la creazione della partizione. I tipi di colonna modificati devono essere compatibili. Per i tipi di riga, i campi finali possono essere aggiunti o eliminati, ma i campi corrispondenti (in ordine ordinale) devono avere lo stesso nome.
  - I file ORC possono ora avere colonne struct con campi mancanti. Ciò consente di modificare lo schema della tabella senza riscrivere i file ORC.
  - Le colonne struct ORC sono ora mappate per nome anziché in ordine numerico. Questo gestisce correttamente i campi struct mancanti o extra nel file ORC.
- SQL OFFSET: la clausola SQL `OFFSET` è ora supportata nelle istruzioni `SELECT`. Per ulteriori informazioni, consulta SELECT.
- Istruzione UNLOAD: è possibile utilizzare l'istruzione `UNLOAD` per scrivere l'output di una query `SELECT` nei formati PARQUET, ORC, AVRO e JSON. Per ulteriori informazioni, consulta UNLOAD.

Miglioramenti a raggruppamento, unione e sottoquery

- Raggruppamento complesso— Aggiunto il supporto per operazioni di raggruppamento complesse.
- Sottoquery correlate— Aggiunto il supporto per le sottoquery correlate in`IN` e per sottoquery correlate che richiedono costrizioni.
- CROSS JOIN – Aggiunto il supporto per `CROSS JOIN` vs. tabelle derivate `LATERAL`.
- GROUPING SETS – Aggiunto il supporto perle clausole `ORDER BY` nelle aggregazioni per le query che utilizzano `GROUPING SETS`.
- Lambda expressions — Aggiunto il supporto per il dereferenziamento dei campi riga nelle espressioni Lambda.

- Null values in semijoins — Aggiunto il supporto per i valori nulli sul lato sinistro di un semijoin (ovvero un predicato `IN` con sottoquery).
- Spatial joins — Aggiunto il supporto per i join spaziali di trasmissione e i join spaziali di sinistra.
- Spill to disk — Per operazioni `INNER JOIN` e `LEFT JOIN` con uso intensivo di memoria, Athena scarica i risultati delle operazioni intermedie su disco. Ciò consente l'esecuzione di query che richiedono grandi quantità di memoria.

Miglioramenti al tipo di dati

- INT for INTEGER – Aggiunto il supporto per `INT` come alias peril tipo di dati `INTEGER`.
- INTERVAL types — Aggiunto il supporto per il casting ai tipi `INTERVAL`.
- IPADDRESS — È stato aggiunto un nuovo `IPADDRESS` tipo per rappresentare gli indirizzi IP nelle query DML. Aggiunto il supporto per il casting tra il tipo `VARBINARY` e `IPADDRESS`. Il `IPADDRESS` tipo non è riconosciuto nelle query DDL.
- IS DISTINCT FROM — Aggiunto il supporto `IS DISTINCT FROM`per i tipi `JSON` e `IPADDRESS`.
- Null equality checks — Verifica di uguaglianza per valori nulli in `ARRAY`, `MAP` e`ROW`. Sono ora supportate le strutture dati. Ad esempio, l'espressione `ARRAY ['1', '3', null] = ARRAY ['1', '2', null]` restituisce `false`. In precedenza, un elemento null restituiva il messaggio di errore Confronto non supportato.
- Row type coercion — È ora consentita la coercizione tra i tipi di riga indipendentemente dai nomi dei campi. In precedenza, un tipo di riga era coercibile a un altro solo se il nome del campo nel tipo di origine corrispondeva al tipo di destinazione o quando il tipo di destinazione aveva un nome di campo anonimo.
- Time subtraction — Sottrazione implementata per tutti i tipi `TIME` e `TIMESTAMP`.
- Unicode — Aggiunto il supporto per le sequenze Unicode con escape nelle stringhe letterali.
- VARBINARY concatenation — Aggiunto il supporto per la concatenazione dei valori `VARBINARY`.

  Funzioni del valore finestra: le funzioni del valore finestra ora supportano `IGNORE NULLS` e `RESPECT NULLS`.

Tipi di input aggiuntivi per le funzioni

Le seguenti funzioni ora accettano tipi di input aggiuntivi. Per ulteriori informazioni su ciascuna funzione, visita il link corrispondente alla documentazione di Presto.

- approx_distinct() – La funzione approx_distinct() supporta ora i seguenti tipi: `INTEGER`, `SMALLINT`, `TINYINT`, `DECIMAL`, `REAL`, `DATE`, `TIMESTAMP`, `TIMESTAMP WITH TIME ZONE`, `TIME`, `TIME WITH TIME ZONE`, `IPADDRESS` e `CHAR`.
- avg(), sum() – Le funzioni di aggregazione avg() e sum() ora supportano il tipo di dati `INTERVAL`.
- lpad(), rpad() – Le funzioni lpad e rpad ora funzionano sugli input `VARBINARY`.
- min(), max() – Le funzioni di aggregazione min() e max() ora consentono tipi di input sconosciuti al momento dell'analisi delle query in modo da poter utilizzare le funzioni con valori letterali `NULL`.
- regexp_replace()— Variante della funzione regexp_replace()aggiunta che può eseguire una funzione Lambda per ogni sostituzione.
- sequenza ()— Aggiunte varianti `DATE` per la funzione sequence(), inclusa la variante con un incremento di fase implicito di un giorno.
- ST_Area()— La funzione geospaziale ST_Area() ora supporta tutti i tipi di geometria.
- substr() – La funzione substr ora funziona sugli input `VARBINARY`.
- zip_with() – Gli array di lunghezza non corrispondente possono ora essere utilizzati con zip_with(). Le posizioni mancanti sono riempite con null. In precedenza, veniva generato un errore quando venivano passati array di lunghezze diverse. Questa modifica può rendere difficile distinguere tra valori originariamente nulli dai valori aggiunti per tamponare gli array alla stessa lunghezza.

Funzioni aggiunte

Il seguente elenco contiene le nuove funzioni disponibili a partire dalla versione 2 del motore Athena. L'elenco non include funzioni geospaziali. Per un elenco delle funzioni geospaziali, consulta Nuove funzioni geospaziali nella versione 2 del motore Athena.

Per ulteriori informazioni su ciascuna funzione, visita il link corrispondente alla documentazione di Presto.

Funzioni di aggregazione

reduce_agg()

Operatori e funzioni della matrice

array_sort(): variante di questa funzione aggiunta che prende una funzione Lambda come comparatore.

ngrams()

Funzioni binarie e operatori

from_big_endian_32()

from_ieee754_32()

from_ieee754_64()

hmac_md5()

hmac_sha1()

hmac_sha256()

hmac_sha512()

spooky_hash_v2_32()

spooky_hash_v2_64()

to_big_endian_32()

to_ieee754_32()

to_ieee754_64()

Operatori e funzioni data e ora

millisecondi()

parse_duration()

to_milliseconds()

Operatori e funzioni mappa

multimap_from_entries()

Operatori e funzioni matematiche

inverse_normal_cdf()

wilson_interval_lower()

wilson_interval_upper()

Funzioni di digest quantile

Aggiunti funzioni digest quantile e il tipo digest quantile `qdigest`.

Operatori e funzioni di stringa

hamming_distance()

split_to_multimap()

Miglioramenti in termini di prestazioni.

Le prestazioni delle seguenti caratteristiche sono migliorate nella versione 2 del motore Athena.

Prestazioni delle query

- Prestazioni broadcast join: miglioramento delle prestazioni broadcast join applicando l'eliminazione dinamica delle partizioni nel nodo worker.
- Tabelle bucket — Prestazioni migliorate per la scrittura su tabelle a periodi fissi quando i dati in scrittura sono già partizionati in modo appropriato (ad esempio, quando l'output proviene da un join a periodi fissi).
- DISTINCT — Prestazioni migliorate per alcune query che utilizzano `DISTINCT`.

  Filtro dinamico ed eliminazione delle partizioni: i miglioramenti apportati aumentano le prestazioni e riducono la quantità di dati scansionati nelle query.
- Operazioni di filtraggio e proiezione — Le operazioni di filtro e proiezione sono ora sempre elaborate per colonne, se possibile. Il motore sfrutta automaticamente le codificazioni del dizionario, laddove efficace.
- Raccolta di scambi — Prestazioni migliorate per le query con raccolta di scambi.
- Aggregazioni globali – Prestazioni migliorate per alcune query che eseguono aggregazioni globali filtrate.
- GROUPING SETS, CUBE, ROLLUP – Prestazioni migliorate per query che coinvolgono `GROUPING SETS`, `CUBE` o `ROLLUP`, che è possibile utilizzare per aggregare più set di colonne in una singola query.
- Filtri altamente selettivi — Migliorate le prestazioni delle query con filtri altamente selettivi.
- Operazioni JOIN e AGGREGATE: le prestazioni delle operazioni `JOIN` e `AGGREGATE` sono state migliorate.

- LIKE— Migliorate le prestazioni delle query che utilizzano i predicati `LIKE` sulle colonne delle tabelle `information_schema`.
- ORDER BY e LIMIT— Miglioramento dei piani, delle prestazioni e dell'utilizzo della memoria per le query che coinvolgono `ORDER BY` e `LIMIT` per evitare inutili scambi di dati.
- ORDER BY – Le operazioni `ORDER BY` sono ora distribuite per impostazione predefinita, abilitando clausole `ORDER BY` più ampie da utilizzare.
- Conversioni di tipo ROW — Prestazioni migliorate durante la conversione tra tipi `ROW`.
- Tipi strutturali – Prestazioni migliorate delle query che elaborano tipi strutturali e contengono scansioni, join, aggregazioni o scritture di tabelle.
- Scansioni delle tabelle: è stata introdotta una regola di ottimizzazione per evitare scansioni di tabelle duplicate in alcuni casi.
- UNION – Prestazioni migliorate per le query `UNION`.

Prestazioni della pianificazione delle query

- Prestazioni di pianificazione – Prestazioni migliorate di pianificazione per query che uniscono più tabelle con un numero elevato di colonne.
- Valutazioni di predicato — Miglioramento delle prestazioni di valutazione dei predicati durante il pushdown dei predicati nella pianificazione.
- Supporto pushdown dei predicati per il casting — Supporto del pushdown del predicato per il predicato *`<column>`* `IN` *`<values list>`*, dove i valori nell'elenco dei valori richiedono il casting per corrispondere al tipo di colonna.
- Inferenza dei predicati e pushdown — Inferenza dei predicati e pushdown estesi per le query che utilizzano un predicato *`<symbol>`* `IN` *`<subquery>`*.
- Timeout: risolto un bug che in rari casi poteva causare timeout di pianificazione delle query.

Prestazioni di join

- Unisci con colonne della mappa – Miglioramento delle prestazioni dei join e delle aggregazioni che includono colonne della mappa.
- Unisci solo con condizioni di non uguaglianza — Migliorate le prestazioni dei join con solo condizioni di non uguaglianza utilizzando un loop join nidificato invece di un hash join.
- Outer joins — Il tipo di distribuzione join viene ora selezionato automaticamente per le query che coinvolgono outer join.

- Intervallo per un'unione di funzioni — Miglioramento delle prestazioni dei join in cui la condizione è un intervallo su una funzione (ad esempio, `a JOIN b ON b.x < f(a.x) AND b.x > g(a.x)`).
- S pill-to-disk — Risolti i bug e i problemi di memoria spill-to-disk correlati per migliorare le prestazioni e ridurre gli errori di memoria nelle operazioni. `JOIN`

Prestazioni delle sottoquery

- Sottoquery EXISTS correlate — Prestazioni migliorate delle sottoquery `EXISTS`correlate.
- Sottoquery correlate con equità — Supporto migliorato per sottoquery correlate contenenti predicati di uguaglianza.
- Sottoquery correlate con disuguaglianze correlate: prestazioni migliorate per sottoquery correlate che contengono disuguaglianze.
- Aggregazioni count(*) su sottoquery: miglioramento delle prestazioni di aggregazioni `count(*)` su sottoquery con cardinalità costante nota.
- Propagazione filtro query esterno: miglioramento delle prestazioni delle sottoquery correlate quando i filtri dalla query esterna possono essere propagate alla sottoquery.

Prestazioni delle funzioni

- Funzioni finestra aggregate: miglioramento delle prestazioni delle funzioni finestra aggregate.
- element_at(): miglioramento delle prestazioni di `element_at()` affinché le mappe siano a tempo costante piuttosto che proporzionali alle dimensioni della mappa.
- grouping(): miglioramento delle prestazioni per query che coinvolgono `grouping()`.
- Casting JSON: miglioramento delle prestazioni del casting dai tipi `JSON` a `ARRAY` o`MAP`.
- Funzioni di restituzione della mappa — Migliorate le prestazioni delle funzioni che restituiscono le mappe.
- M ap-to-map casting: sono state migliorate le prestazioni del map-to-map cast.
- min() e max() – Le funzioni `min()` e `max()` sono state ottimizzate per evitare la creazione di oggetti non necessari, riducendo così il sovraccarico di garbage collection.
- row_number() – Miglioramento delle prestazioni e dell'utilizzo della memoria per le query utilizzando `row_number()` seguito da un filtro sui numeri di riga generati.
- Funzioni finestra — Miglioramento delle prestazioni delle query contenenti funzioni di finestra con clausole `PARTITION BY` e `ORDER BY` identiche.

- Funzioni finestra — Miglioramento delle prestazioni di alcune funzioni finestra (ad esempio,`LAG`) che hanno specifiche simili.

Prestazioni geospaziali

- Serializzazione della geometria — Migliorate le prestazioni di serializzazione dei valori della geometria.
- Funzioni geospaziali — Migliorate le prestazioni di `ST_Intersects()`, `ST_Contains()`, `ST_Touches()`, `ST_Within()`, `ST_Overlaps()`, `ST_Disjoint()`, `transform_values()`, `ST_XMin()`, `ST_XMax()`, `ST_YMin()`, `ST_YMax()`, `ST_Crosses()` e `array_intersect()`.
- ST_Distance() — Miglioramento delle prestazioni delle query join che coinvolgono la funzione `ST_Distance()`.
- ST_Intersection() – Ottimizzata la funzione `ST_Intersection()` per i rettangoli allineati con gli assi delle coordinate (ad esempio, i poligoni prodotti dalle funzioni `ST_Envelope()` e `bing_tile_polygon()`).

Miglioramenti relativi a JSON

Funzioni della mappa

- Prestazioni migliorate del pedice della mappa da `O(n)` a `O(1)` in tutti i casi. In precedenza, solo le mappe prodotte da determinate funzioni e lettori hanno beneficiato di questo miglioramento.
- Aggiunte le funzioni `map_from_entries()` e `map_entries()`.

Casting

- Aggiunta la possibilità di eseguire il cast su`JSON` da `REAL`, `TINYINT` o `SMALLINT`.
- È ora possibile eseguire il cast di `JSON` su `ROW` anche se `JSON` non contiene tutti i campi nella `ROW`.
- Prestazioni migliorate di `CAST(json_parse(...) AS ...)`.
- Migliorate le prestazioni del casting dai tipi `JSON` a`ARRAY` o `MAP`.

Nuove funzioni JSON

- is_json_scalar()

Modifiche importanti

Le modifiche di rottura includono correzioni di bug, modifiche alle funzioni geospaziali, funzioni sostituite e l'introduzione di limiti. Miglioramenti nella conformità SQL ANSI possono interrompere le query che dipendevano da comportamenti non standard.

Correzioni di bug

Le seguenti modifiche correggono i problemi comportamentali che hanno causato l'esecuzione corretta delle query, ma con risultati imprecisi.

- fixed_len_byte_array Parquet columns are now accepted as DECIMAL – Le query su colonne Parquet di tipo `fixed_len_byte_array` hanno esito positivo e restituiscono valori corretti se sono annotati come `DECIMAL` nello schema Parquet. Le query su colonne `fixed_len_byte_array` senza l'annotazione `DECIMAL` hanno esito negativo e restituiscono un errore. In precedenza, le query sulle colonne `fixed_len_byte_array` senza l'annotazione DECIMAL avevano esito positivo ma restituivano valori incomprensibili.
- json_parse() non ignora più i caratteri finali— In precedenza, input come`[1,2]abc` sarebbero stati analizzati come `[1,2]`. L'utilizzo dei caratteri finali ora restituisce il messaggio di errore Impossibile convertire '[1, 2] abc' in JSON.
- precisione decimale round() corretta – `round(x, d)` ora arrotonda correttamente `x`quando `x` è un DECIMALE o quando `x` è un DECIMALE con scala 0 e `d` è un integer negativo. In precedenza, non si verificava alcun arrotondamento in questi casi.
- round(x, d) e truncate(x, d) –Il parametro d nella firma delle funzioni `round(x, d)` e `truncate(x, d)` è ora di tipo `INTEGER`. In precedenza, d poteva essere di tipo `BIGINT`.
- map() con chiavi duplicate – `map()` ora genera un errore sulle chiavi duplicate piuttosto che produrre silenziosamente una mappa danneggiata. Le query che attualmente costruiscono valori di mappa utilizzando chiavi duplicate ora non hanno esito positivo con un errore.
- map_from_entries() genera un errore con voci nulle – `map_from_entries()` ora genera un errore quando l'array di input contiene una voce nulla. Le query che costruiscono una mappa trasmettendo `NULL` come valore ora hanno esito negativo.
- Tabelle — Le tabelle con tipi di partizione non supportati non possono più essere create.
- Miglioramento della stabilità numerica nelle funzioni statistiche — La stabilità numerica per le funzioni statistiche `corr()`, `covar_samp()`, `regr_intercept()` e `regr_slope()` è stata migliorata.

- La precisione TIMESTAMP definita in Parquet è ora rispettata — La precisione dei valori `TIMESTAMP` e la precisione definita per la colonna `TIMESTAMP` nello schema Parquet devono ora corrispondere. Le precisioni non corrispondenti generano marche temporali errati.
- Informazioni fuso orario — Le informazioni sul fuso orario vengono ora calcolate utilizzando il pacchetto java.time dell'SDK Java 1.8.
- Tipi di dati SUM of INTERVAL_DAY_TO_SECOND e INTERVAL_YEAR_TO_MONTH - Non è più possibile utilizzare `SUM(NULL)` direttamente. Per utilizzare`SUM(NULL)`, esegui il casting di `NULL` a un tipo di dati come `BIGINT`, `DECIMAL`, `REAL`, `DOUBLE`, `INTERVAL_DAY_TO_SECOND` o `INTERVAL_YEAR_TO_MONTH`.

Modifiche alle funzioni geospaziali

Le modifiche apportate alle funzioni geospaziali includono quanto segue.

- Modifiche al nome delle funzioni — Alcuni nomi di funzioni sono cambiati. Per ulteriori informazioni, consulta Modifiche ai nomi delle funzioni geospaziali nella versione 2 del motore Athena.
- Input VARBINARY — Il tipo `VARBINARY` non è più direttamente supportato per l'input alle funzioni geospaziali. Ad esempio, per calcolare direttamente l'area di una geometria, la geometria deve ora essere inserita in `VARCHAR` o `GEOMETRY`. La soluzione alternativa consiste nell'utilizzare le funzioni di trasformazione, come negli esempi seguenti.
  - Per utilizzare `ST_area()` per calcolare l'area per l'input `VARBINARY` in formato Well-Known Binary (WKB), per prima cosa passa l'input a `ST_GeomFromBinary()`, ad esempio:

    ```
    ST_area(ST_GeomFromBinary(<wkb_varbinary_value>))
    ```

  - Per utilizzare `ST_area()` per calcolare l'area per l'input `VARBINARY` in formato binario legacy, per prima cosa passa l'input alla funzione `ST_GeomFromLegacyBinary()`, ad esempio:

    ```
    ST_area(ST_GeomFromLegacyBinary(<legacy_varbinary_value>))
    ```

- ST_ ExteriorRing () e ST_Polygon () — `ST_ExteriorRing()`e `ST_Polygon()`ora accettano solo poligoni come input. In precedenza, queste funzioni accettavano erroneamente altre geometrie.
- ST_Distance()— Come richiesto dalle Specificazioni SQL/MM, la funzione `ST_Distance()` ora restituisce `NULL` se uno degli input è una geometria vuota. In precedenza, veniva restituito `NaN`.

Conformità ad ANSI SQL

I seguenti problemi di sintassi e comportamentali sono stati corretti per seguire lo standard ANSI SQL.

- Operazioni cast() — Le operazioni cast() da REAL o DOUBLE a DECIMAL sono ora conformi allo standard SQL. Ad esempio, `cast (double '10000000000000000000000000000000' as decimal(38))` prima restituiva `100000000000000005366162204393472`, mentre ora restituisce `100000000000000000000000000000000`.
- JOIN ... USING– `JOIN ... USING` ora è conforme alla semantica SQL standard. In precedenza, `JOIN ... USING` richiedeva di qualificare il nome della tabella in colonne e la colonna di entrambe le tabelle era presente nell'output. Le qualificazioni della tabella ora non sono valide e la colonna è presente una sola volta nell'output.
- Letterali di tipo ROW rimossi — Il formato letterale di tipo ROW `ROW<int, int>(1, 2)` non è più supportato. Usa invece la sintassi `ROW(1 int, 2 int)`.
- Semantica di aggregazione raggruppata — Le aggregazioni raggruppate usano la semantica `IS NOT DISTINCT FROM` piuttosto che semantica dell'uguaglianza. Le aggregazioni raggruppate ora restituiscono risultati corretti e mostrano prestazioni migliorate durante il raggruppamento su valori con virgola mobile `NaN`. È supportato il raggruppamento su tipi di mappa, elenco e riga che contengono valori nulli.
- I tipi con virgolette non sono più consentiti — In conformità con lo standard ANSI SQL, i tipi di dati non possono più essere racchiusi tra virgolette. Ad esempio, `SELECT "date" '2020-02-02'` non è più una query valida. Utilizza invece la sintassi `SELECT date '2020-02-02'`.
- Accesso ai campi riga anonimi — Non è più possibile accedere ai campi riga anonimi utilizzando la sintassi `[.field0, .field1, ...]`.
- Operazioni di raggruppamento complesse — Le complesse operazioni di raggruppamento `GROUPING SETS`, `CUBE` e `ROLLUP` non supportano il raggruppamento su espressioni composte da colonne di input. Solo ammessi solo i nomi di colonna.

Funzioni sostituite

Le funzioni seguenti non sono più supportate e sono state sostituite da sintassi che produce gli stessi risultati.

- information_schema.__internal_partitions__: l'uso di `__internal_partitions__` non è più supportato nel motore Athena versione 2. Per una sintassi equivalente, utilizza `SELECT * FROM`

"`<table_name>`$partitions" o `SHOW PARTITIONS`. Per ulteriori informazioni, consulta Elencare le partizioni per una tabella specifica.

- Sostituite le funzioni geospaziali — Per un elenco delle funzioni geospaziali i cui nomi sono cambiati, consulta Modifiche ai nomi delle funzioni geospaziali nella versione 2 del motore Athena.

Limiti

I seguenti limiti sono stati introdotti nella versione 2 del motore Athena per garantire che le query non abbiano esito negativo a causa di limitazioni di risorse. Questi limiti non sono configurabili dagli utenti.

- Numero degli elementi del risultato— Il numero di elementi del risultato n è limitato a 10.000 o meno per le seguenti funzioni: `min(col, n)`, `max(col, n)`, `min_by(col1, col2, n)` e `max_by(col1, col2, n)`.
- GROUPING SETS – Il numero massimo di sezioni in un gruppo di raggruppamenti è 2048.
- Lunghezza massima della riga del file di testo: la lunghezza massima della riga predefinita per i file di testo è 200 MB.
- Dimensione massima del risultato della funzione di sequenza — La dimensione massima del risultato di una funzione di sequenza è 50.000 voci. Ad esempio, `SELECT sequence(0,45000,1)` ha esito positivo, ma `SELECT sequence(0,55000,1)` ha esito negativo con il messaggio di errore Il risultato della funzione di sequenza non deve avere più di 50.000 voci. Questo limite è valido per tutti i tipi di input per funzioni di sequenza, inclusi i le marche temporali.

Documentazione di riferimento SQL per Athena

Amazon Athena supporta un sottoinsieme di istruzioni, funzioni, operatori e tipi di dati DDL (Data Definition Language) e DML (Data Manipulation Language). Con alcune eccezioni, Athena DDL è basato su HiveQL DDL e Athena DML è basato su Trino. Per ulteriori informazioni sulle versioni del motore Athena, consulta Controllo delle versioni del motore di Athena.

Argomenti

- Tipi di dati in Amazon Athena
- Query, funzioni e operatori DML
- Istruzioni DDL
- Considerazioni e restrizioni per le query SQL in Amazon Athena

Tipi di dati in Amazon Athena

Quando si esegue`CREATE TABLE`, si specificano i nomi delle colonne e il tipo di dati che ogni colonna può contenere. Le tabelle create vengono archiviate in AWS Glue Data Catalog.

Per facilitare l'interoperabilità con altri motori di query, Athena utilizza i nomi dei tipi di dati Apache Hive per istruzioni DDL come. `CREATE TABLE` Per le query DML come`SELECT`, e`CTAS`, `INSERT INTO` Athena utilizza i nomi dei tipi di dati Trino. La tabella seguente mostra i tipi di dati supportati in Athena. Laddove i tipi DDL e DML differiscono in termini di nome, disponibilità o sintassi, vengono visualizzati in colonne separate.

DDL	DML	Descrizione
BOOLEAN		I valori validi sono `true` e `false`.
TINYINT		Un intero con segno a 8 bit in formato complementare a due, con un valore minimo di -2 ^{7} e un valore massimo di 2 ^{7} -1.
SMALLINT		Un intero con segno a 16 bit in formato complementare a due, con un valore minimo di -2 ^{15} e un valore massimo di 2 15 -1.
INT, INTEGER		Un valore con segno a 32 bit in formato complementare a due, con un valore minimo di -2 ^{31} e un valore massimo di 2 31 -1.
BIGINT		Un intero con segno a 64 bit in formato complemento a due, con un valore minimo di -2 ^{63} e un valore massimo di 2 ^{63} -1.
FLOAT	REAL	Un numero a virgola mobile a precisione singola firmato a 32 bit. L'intervallo è compreso tra 1,40129846432481707e-45 e 3,40282346638528860e+38, positivo o negativo. Segue lo standard IEEE per l'aritmetica a virgola mobile (IEEE 754).
DOUBLE		Un numero a virgola mobile a doppia precisione firmato a 64 bit. L'intervallo è compreso tra 4,94065645841246544e-324d e 1,79769313486231570e+308d, positivo o negativo. Segue lo standard IEEE per l'aritmetica a virgola mobile (IEEE 754).

DDL	DML	Descrizione
DECIMAL (precisione, scala)		*precision* è il numero totale di cifre. *scale*(opzionale) è il numero di cifre nella parte frazionaria con un valore predefinito di 0. Ad esempio, è possibile usare il tipo di queste definizioni: `decimal(11,5)` , `decimal(15)` . Il valore massimo per la *precisione* è 38 mentre il valore massimo per la *scala* è 38.
CHAR, CHAR (lunghezza)		Dati di caratteri a lunghezza fissa, con una lunghezza specificata compresa tra 1 e 255, come char (10). Se viene specificata la *lunghezza* , le stringhe vengono troncate alla lunghezza specificata durante la lettura. Se la stringa di dati sottostante è più lunga, la stringa di dati sottostante rimane invariata. Per ulteriori informazioni, consulta la sezione relativa a tipo di dati Hive CHAR.
STRING	VARCHAR	Dati di caratteri a lunghezza variabile.
VARCHAR (lunghezza)		Dati di caratteri a lunghezza variabile con una lunghezza massima di lettura. Le stringhe vengono troncate alla lunghezza specificata durante la lettura. Se la stringa di dati sottostante è più lunga, la stringa di dati sottostante rimane invariata.
BINARY	VARBINARY	Dati binari a lunghezza variabile.
TIME		Un'ora del giorno con precisione al millisecondo.
Non disponibile	*ORA (precisione)*	Un'ora del giorno con una precisione specifica. `TIME(3)`è equivalente a`TIME`.
Non disponibile	TIME WITH TIME ZONE	Un'ora del giorno in un fuso orario. I fusi orari devono essere specificati come offset rispetto all'UTC.
DATE		Una data di calendario con anno, mese e giorno.

DDL	DML	Descrizione
TIMESTAMP	TIMESTAMP, TIMESTAMP SENZA FUSO ORARIO	Data e ora del giorno del calendario con precisione al millisecondo.
Non disponibile	*TIMESTAMP (precisione), TIMESTAMP (precisione) SENZA FUSO ORARIO*	Una data e un'ora del giorno del calendario con una precisione specifica. `TIMESTAMP(3)` è equivalente a`TIMESTAMP` .
Non disponibile	TIMESTAMP WITH TIME ZONE	Data e ora del giorno del calendario in un fuso orario. I fusi orari possono essere specificati come offset rispetto all'UTC, come nomi di fusi orari IANA o utilizzando UTC, UT, Z o GMT.
Non disponibile	*TIMESTAMP (PRECISIONE) CON FUSO ORARIO*	Una data e un'ora del giorno del calendario con una precisione specifica, in un fuso orario.
Non disponibile	INTERVAL YEAR TO MONTH	Un intervallo di uno o più mesi interi
Non disponibile	INTERVAL DAY TO SECOND	Un intervallo di uno o più secondi, minuti, ore o giorni
ARRAY< element_type >	*ARRAY [tipo_elemento]*	Una matrice di valori. Tutti i valori devono avere lo stesso tipo.

DDL	DML	Descrizione
MAP< tipo_chiave, tipo_valore >	*MAP (tipo_chiave, tipo_valore)*	Una mappa in cui è possibile cercare i valori per chiave. Tutte le chiavi devono avere lo stesso valore e tutti i valori devono avere lo stesso valore.
STRUCT< field_name_1: field_type_1, field_name_2 : field_type_2 ,... >	*ROW (nome_campo_1 field_type_1, nome_campo_2 field_type_2 ,...)*	Una struttura di dati con campi denominati e relativi valori.
Non disponibile	JSON	Tipo di valore JSON, che può essere un oggetto JSON, un array JSON, un numero JSON, una stringa JSON o. `true` `false` `null`
Non disponibile	UUID	Un UUID (Identificatore univoco universale).
Non disponibile	INDIRIZZO IP	Un indirizzo IPv4 o IPv6.
Non disponibile	HyperLogLog	Questi tipi di dati supportano funzioni interne approssimative. Per ulteriori informazioni su ciascun tipo, visita il link alla voce corrispondente nella documentazione di Trino.
	P4 HyperLogLog	
	SetDigest	
	QDigest	
	TDigest	

Esempi di tipi di dati

La tabella seguente mostra esempi di valori letterali per i tipi di dati DML.

Tipo di dati	Esempi
BOOLEAN	`true` `false`
TINYINT	`TINYINT '123'`
SMALLINT	`SMALLINT '123'`
INT, INTEGER	`123456790`
BIGINT	`BIGINT '1234567890'` `2147483648`
REAL	`'123456.78'`
DOUBLE	`1.234`
DECIMAL (precisione, scala)	`DECIMAL '123.456'`
CHAR, CHAR (lunghezza)	`CHAR 'hello world'`, `CHAR 'hello ''world''!'`
VARCHAR, VARCHAR (lunghezza)	`VARCHAR 'hello world'`, `VARCHAR 'hello ''world''!'`
VARBINARY	`X'00 01 02'`
TEMPO, TEMPO (precisione)	`TIME '10:11:12'` , `TIME '10:11:12.345'`
TIME WITH TIME ZONE	`TIME '10:11:12.345 -06:00'`
DATE	`DATE '2024-03-25'`
TIMESTAMP, TIMESTAMP SENZA FUSO ORARIO, TIMESTAMP (precisio	`TIMESTAMP '2024-03-25 11:12:13'` , `TIMESTAMP '2024-03-25 11:12:13.456'`

Tipo di dati	Esempi
ne), TIMESTAMP (precisione) SENZA FUSO ORARIO	
TIMESTAMP CON FUSO ORARIO, TIMESTAMP (PRECISIONE) CON FUSO ORARIO	`TIMESTAMP '2024-03-25 11:12:13.456 Europe/Berlin'`
INTERVAL YEAR TO MONTH	`INTERVAL '3' MONTH`
INTERVAL DAY TO SECOND	`INTERVAL '2' DAY`
ARRAY [tipo_elemento]	`ARRAY['one', 'two', 'three']`
MAP (tipo_chiave, tipo_valore)	`MAP(ARRAY['one', 'two', 'three'], ARRAY[1, 2, 3])` Nota che le mappe vengono create da una matrice di chiavi e da una matrice di valori.

ROW (field_name_1 field_type_1, field_name_2 field_type_2 ,...)	`ROW('one', 'two', 'three')`
	
	Nota che le righe create in questo modo non hanno nomi di colonna. Per aggiungere nomi di colonna, puoi usare`CAST`, come nell'esempio seguente:
	
	`CAST(ROW(1, 2, 3) AS ROW(one INT, two INT, three INT))`

JSON	`JSON '{"one":1, "two": 2, "three": 3}'`
UUID	`UUID '12345678-90ab-cdef-1234-567890abcdef'`

Tipo di dati	Esempi

INDIRIZZO IP	`IPADDRESS '10.0.0.1'`
	
	`IPADDRESS '2001:db8::1'`

Considerazioni sui tipi di dati

Limiti di dimensione

Per i tipi di dati che non specificano un limite di dimensione, tieni presente che esiste un limite pratico di 32 MB per tutti i dati in una singola riga. Per ulteriori informazioni, consulta Row or column size limitation in Considerazioni e restrizioni per le query SQL in Amazon Athena.

CHAR e VARCHAR

Un `CHAR(n)` valore ha sempre un numero di caratteri. *n* Ad esempio, se si trasmette 'abc' a`CHAR(7)`, vengono aggiunti 4 spazi finali.

I confronti di `CHAR` valori includono spazi iniziali e finali.

Se viene specificata una lunghezza per `CHAR` o`VARCHAR`, le stringhe vengono troncate alla lunghezza specificata durante la lettura. Se la stringa di dati sottostante è più lunga, la stringa di dati sottostante rimane invariata.

Per evitare una virgoletta singola in una `CHAR` o`VARCHAR`, usa una virgoletta singola aggiuntiva.

Per eseguire il cast di un tipo di dati non stringa in una stringa in una query DML, `VARCHAR` esegui il cast sul tipo di dati.

Per utilizzare la `substr` funzione per restituire una sottostringa di lunghezza specificata da un tipo di `CHAR` dati, dovete prima eseguire il cast del `CHAR` valore come. `VARCHAR` Nell'esempio seguente, `col1` utilizza il tipo di `CHAR` dati.

```
substr(CAST(col1 AS VARCHAR), 1, 4)
```

DECIMAL

Per specificare i valori decimali come valori letterali nelle `SELECT` query, ad esempio quando si selezionano righe con un valore decimale specifico, è possibile specificare il `DECIMAL` tipo ed

elencare il valore decimale come letterale tra virgolette singole nella query, come negli esempi seguenti.

```
SELECT * FROM my_table
WHERE decimal_value = DECIMAL '0.12'
```

```
SELECT DECIMAL '44.6' + DECIMAL '77.2'
```

Utilizzo dei dati con timestamp

Questa sezione descrive alcune considerazioni sull'utilizzo dei dati con timestamp in Athena.

> **Note**
>
> Il trattamento dei timestamp è leggermente cambiato tra la versione 2 e la versione 3 del motore Athena. Per informazioni sugli errori relativi ai timestamp che possono verificarsi nella versione 3 del motore Athena e sulle soluzioni suggerite, consultare Modifiche al timestamp nei riferimenti Versione 3 del motore Athena.

Formato per la scrittura di dati timestamp su oggetti Amazon S3

Il formato in cui i dati timestamp devono essere scritti negli oggetti Amazon S3 dipende sia dal tipo di dati della colonna che SerDedalla libreria utilizzata.

- Se hai una colonna di tabella di tipo `DATE`, Athena si aspetta che la colonna o la proprietà corrispondente dei dati sia una stringa in formato `YYYY-MM-DD` ISO o un tipo di data integrato come quelli per Parquet o ORC.
- Se hai una colonna di tabella di tipo `TIME`, Athena si aspetta che la colonna o la proprietà corrispondente dei dati sia una stringa in formato `HH:MM:SS` ISO o un tipo di ora integrato come quelli per Parquet o ORC.
- Se hai una colonna di tabella di tipo `TIMESTAMP`, Athena si aspetta che la colonna o la proprietà corrispondente dei dati sia una stringa nel formato `YYYY-MM-DD HH:MM:SS.SSS` (nota lo spazio tra la data e l'ora) o un tipo di ora integrato come quelli per Parquet, ORC o Ion.

> **Note**
>
> I timestamp SerDe OpenCSV sono un'eccezione e devono essere codificati come epoche UNIX con risoluzione in millisecondi.

Come garantire che i dati partizionati nel tempo corrispondano al campo timestamp di un record

Il produttore dei dati deve assicurarsi che i valori della partizione siano allineati con i dati all'interno della partizione. Ad esempio, se i dati hanno una `timestamp` proprietà e si utilizza Firehose per caricare i dati in Amazon S3, è necessario utilizzare il partizionamento dinamico perché il partizionamento predefinito di Firehose è. wall-clock-based

Usa string come tipo di dati per le chiavi di partizione

Per motivi di prestazioni, è preferibile utilizzare `STRING` come tipo di dati per le chiavi di partizione. Anche se Athena riconosce i valori delle partizioni nel formato `YYYY-MM-DD` come date quando si utilizza il tipo `DATE`, ciò può portare a prestazioni scadenti. Per questo motivo, si consiglia di utilizzare invece il tipo di dati `STRING` per le chiavi di partizione.

Come scrivere query per campi timestamp che sono anche partizionati in base all'ora

Il modo in cui si scrivono le query per i campi di indicazione temporale che sono partizionati in base all'ora dipende dal tipo di tabella su cui si intende eseguire la query.

Tavoli Hive

Con le tabelle Hive più comunemente utilizzate in Athena, il motore di query non conosce le relazioni tra colonne e chiavi di partizione. Per questo motivo, devi sempre aggiungere predicati nelle tue query sia per la colonna che per la chiave di partizione.

Ad esempio, supponi di avere una colonna `event_time` e una chiave di partizione `event_date` e di voler eseguire query sugli eventi tra le 23:00 e le 03:00. In questo caso, è necessario includere i predicati nella query sia per la colonna che per la chiave di partizione, come nell'esempio seguente.

```
WHERE event_time BETWEEN start_time AND end_time
  AND event_date BETWEEN start_time_date AND end_time_date
```

Tavoli Iceberg

Con le tabelle Iceberg, puoi utilizzare valori di partizione calcolati, il che semplifica le tue query. Ad esempio, supponiamo che la tabella Iceberg sia stata creata con una clausola come la seguente: `PARTITIONED BY`

```
PARTITIONED BY (event_date month(event_time))
```

In questo caso, il motore di query elimina automaticamente le partizioni in base ai valori dei predicati `event_time`. Per questo motivo, la query deve solo specificare un predicato per `event_time`, come nell'esempio seguente.

```
WHERE event_time BETWEEN start_time AND end_time
```

Per ulteriori informazioni, consulta Creazione di tabelle Iceberg.

Query, funzioni e operatori DML

Il motore di query DML Athena supporta in genere la sintassi Trino e Presto, aggiungendo inoltre i propri miglioramenti. Athena non supporta tutte le funzioni di Trino e Presto. Per ulteriori informazioni, consulta gli argomenti per le specifiche istruzioni in questa sezione e in Considerazioni e limitazioni. Per informazioni sulle funzioni, consulta Funzioni in Amazon Athena. Per ulteriori informazioni sulle versioni del motore Athena, consulta Controllo delle versioni del motore di Athena.

Per informazioni sulle istruzioni DDL, consulta Istruzioni DDL. Per un elenco delle istruzioni DDL non supportate, consulta DDL non supportato.

SELECT

Recupera righe di dati da zero o più tabelle.

> **Note**
>
> In questo argomento vengono fornite informazioni di riepilogo per riferimento. Informazioni complete sull'utilizzo di `SELECT` e il linguaggio SQL non rientrano nell'ambito di questa documentazione. Per informazioni sull'utilizzo di SQL specifico per Athena, consulta Considerazioni e restrizioni per le query SQL in Amazon Athena e Esecuzione di query SQL con Amazon Athena. Per vedere esempio di creazione di un database o di una tabella e di esecuzione di una query `SELECT` sulla tabella di Athena, consulta Nozioni di base.

Riepilogo

```
[ WITH with_query [, ...] ]
SELECT [ ALL | DISTINCT ] select_expression [, ...]
[ FROM from_item [, ...] ]
[ WHERE condition ]
[ GROUP BY [ ALL | DISTINCT ] grouping_element [, ...] ]
[ HAVING condition ]
[ { UNION | INTERSECT | EXCEPT } [ ALL | DISTINCT ] select ]
[ ORDER BY expression [ ASC | DESC ] [ NULLS FIRST | NULLS LAST] [, ...] ]
[ OFFSET count [ ROW | ROWS ] ]
[ LIMIT [ count | ALL ] ]
```

> **Note**
>
> Le parole riservate nelle istruzioni SQL SELECT devono essere racchiuse tra virgolette doppie. Per ulteriori informazioni, consulta Elenco delle parole chiave riservate nelle istruzioni SQL SELECT.

Parametri

[WITH with_query [, ...]]

È possibile utilizzare `WITH` per appiattire query nidificate o per semplificare le sottoquery.

L'utilizzo della clausola `WITH` per creare query ricorsive è supportato a partire dalla versione 3 del motore Athena. La profondità massima di ricorsione è pari a 10.

La clausola `WITH` precede l'elenco `SELECT` in una query e definisce una o più sottoquery da usare all'interno della query `SELECT`.

Ogni sottoquery definisce una tabella temporanea, simile a una definizione vista, cui è possibile fare riferimento nella clausola `FROM`. Le tabelle vengono utilizzate solo quando la query è in esecuzione.

La sintassi `with_query` è:

```
subquery_table_name [ ( column_name [, ...] ) ] AS (subquery)
```

Dove:

- `subquery_table_name` è un univoco per una tabella temporanea che definisce i risultati della sottoquery della clausola `WITH`. A ogni `subquery` deve essere assegnato un nome di tabella a cui è possibile fare riferimento nella clausola `FROM`.
- `column_name [, ...]` è un elenco opzionale dei nomi di colonna di output. Il numero dei nomi di colonna specificati deve essere uguale o inferiore al numero delle colonne definite dalla `subquery`.
- `subquery` è un'istruzione di query.

[TUTTI | DISTINCT] select_expression

`select_expression` determina la righe da selezionare. A `select_expression` può utilizzare uno dei seguenti formati:

```
expression [ [ AS ] column_alias ] [, ...]
```

```
row_expression.* [ AS ( column_alias [, ...] ) ]
```

```
relation.*
```

```
*
```

- La `expression [ [ AS ] column_alias ]` sintassi specifica una colonna di output. La `[AS] column_alias` sintassi opzionale specifica un nome di titolo personalizzato da utilizzare per la colonna nell'output.
- For`row_expression.* [ AS ( column_alias [, ...] ) ]`, `row_expression` è un'espressione arbitraria del tipo di dati. `ROW` I campi della riga definiscono le colonne di output da includere nel risultato.
- Perché`relation.*`, le colonne di `relation` sono incluse nel risultato. Questa sintassi non consente l'uso di alias di colonna.
- L'asterisco `*` specifica che tutte le colonne devono essere incluse nel set di risultati.
- Nel set di risultati, l'ordine delle colonne è lo stesso dell'ordine in cui vengono specificate dall'espressione select. Se un'espressione select restituisce più colonne, l'ordine delle colonne segue l'ordine utilizzato nella relazione di origine o nell'espressione del tipo di riga.
- Quando vengono specificati alias di colonna, gli alias sostituiscono i nomi di campi di colonna o riga preesistenti. Se l'espressione select non ha nomi di colonna, nell'output vengono visualizzati i nomi di colonna anonimi con indicizzazione zero (`_col0`,,`_col1`). `_col2, ...`

- Il valore predefinito è `ALL`. L'utilizzo di `ALL` viene gestito come se fosse omesso; vengono selezionate tutte le righe per tutte le colonne e duplicati vengono conservati.
- Usa `DISTINCT` per restituire solo i valori distinti quando una colonna contiene valori duplicati.

FROM from_item [, ...]

Indica l'input alla query, dove `from_item` può essere una vista, un costrutto JOIN o una sottoquery, come descritto di seguito.

`from_item` può essere:

- `table_name [ [ AS ] alias [ (column_alias [, ...]) ] ]`

  Dove `table_name` è il nome della tabella di destinazione da cui selezionare le righe; `alias` è il nome da assegnare all'output dell'istruzione `SELECT` e `column_alias` definisce le colonne per l'`alias` specificato.

-O-

- `join_type from_item [ ON join_condition | USING ( join_column [, ...] ) ]`

  Dove `join_type` è una tra le seguenti opzioni:

  - `[ INNER ] JOIN`
  - `LEFT [ OUTER ] JOIN`
  - `RIGHT [ OUTER ] JOIN`
  - `FULL [ OUTER ] JOIN`
  - `CROSS JOIN`
  - `ON join_condition | USING (join_column [, ...])` Dove l'utilizzo di `join_condition` consente di specificare nomi di colonna per le chiavi JOIN in più tabelle e l'utilizzo di `join_column` richiede che `join_column` esista in entrambe le tabelle.

[condizione WHERE]

Filtra i risultati in base alla `condition` specificata, dove `condition` in genere ha la seguente sintassi.

```
column_name operator value [[[AND | OR] column_name operator value] ...]
```

L'*`operatore`* può essere uno dei comparatori =, >, <, >=, <=, <>, !=.

Le seguenti espressioni di sottoquery possono essere utilizzate anche nella clausola WHERE.

- [NOT] BETWEEN *integer_A* AND *integer_B* — Specifica un intervallo tra due numeri interi, come nell'esempio seguente. Se il tipo di dati della colonna è varchar, la colonna deve essere prima creata su un numero intero.

```
SELECT DISTINCT processid FROM "webdata"."impressions"
WHERE cast(processid as int) BETWEEN 1500 and 1800
ORDER BY processid
```

- [NOT] LIKE *value*— Cerca il modello specificato. Utilizza il segno di percentuale (%) come carattere jolly, come nell'esempio seguente.

```
SELECT * FROM "webdata"."impressions"
WHERE referrer LIKE '%.org'
```

- [NOT] IN (*value*[, *value*[, ...]) – Specifica un elenco di possibili valori per una colonna, come nell'esempio seguente.

```
SELECT * FROM "webdata"."impressions"
WHERE referrer IN ('example.com','example.net','example.org')
```

[GROUP BY [ALL | DISTINCT] grouping_expressions [,...]]

Suddivide l'output dell'istruzione SELECT in righe con valori corrispondenti.

ALL e DISTINCT determinano se i set duplicati di set di raggruppamento producono ciascuno righe di output diverse. Se omesso, viene utilizzato ALL.

grouping_expressions consente di eseguire operazioni di raggruppamento complesse. È possibile utilizzare operazioni di raggruppamento complesse per eseguire un'analisi che richiede di aggregare più serie di colonne in una query singola.

L'elemento grouping_expressions può essere qualsiasi funzione, come SUM, AVG o COUNT, eseguita sulle colonne di input.

Le espressioni GROUP BY possono raggruppare l'output in base ai nomi di colonna di input che non appaiono nell'output dell'istruzione SELECT.

Tutte le espressioni di output devono essere funzioni aggregate o colonne presenti nella clausola GROUP BY.

È possibile utilizzare una singola query per eseguire un'analisi che richiede di aggregare più serie di colonne.

Athena supporta aggregazioni complesse usando `GROUPING SETS`, `CUBE` e `ROLLUP`. `GROUP BY GROUPING SETS` specifica più elenchi di colonne su cui eseguire il raggruppamento. `GROUP BY CUBE` genera tutti i possibili set di raggruppamento per un determinato set di colonne. `GROUP BY ROLLUP` genera tutti i possibili subtotali per un determinato set di colonne. Le operazioni di raggruppamento complesse non supportano il raggruppamento su espressioni composte da colonne di input. Solo ammessi solo i nomi di colonna.

Spesso è possibile utilizzare `UNION ALL` per ottenere gli stessi risultati ottenuti in queste operazioni `GROUP BY`, ma le query che utilizzano `GROUP BY` hanno il vantaggio di leggere i dati una sola volta, mentre `UNION ALL` legge i dati sottostanti tre volte e potrebbe produrre risultati incoerenti quando l'origine dati è soggetta a modifiche.

[condizione HAVING]

Utilizzata con funzioni aggregate e con la clausola `GROUP BY`. Controlla quali gruppi sono selezionati, eliminando i gruppi che non soddisfano la `condition`. Questo filtraggio si verifica dopo che i gruppi e gli aggregati sono stati calcolati.

[{ UNION | INTERSECT | EXCEPT } [ALL | DISTINCT] union_query]]

`UNION`, `INTERSECT` e `EXCEPT` combinano i risultati di più di un'istruzione `SELECT` in una singola query. `ALL` o `DISTINCT` controlla l'unicità delle righe incluse nel set finale di risultati.

`UNION` combina le righe risultanti dalla prima query con le righe risultanti dalla seconda query. Per eliminare i duplicati, `UNION` crea una tabella hash, che consuma memoria. Per prestazioni migliori, prendere in considerazione l'utilizzo di `UNION ALL` se la query non richiede l'eliminazione dei duplicati. Più clausole `UNION` vengono elaborate da sinistra a destra, a meno che non si utilizzino le parentesi per definire esplicitamente l'ordine di elaborazione.

`INTERSECT` restituisce solo le righe presenti nei risultati sia della prima che della seconda query.

`EXCEPT` restituisce le righe dai risultati della prima query, escludendo le righe trovate dalla seconda query.

`ALL` fa sì che tutte le righe siano incluse, anche se le righe sono identiche.

`DISTINCT` fa sì che solo righe univoche siano incluse nel set di risultati combinati.

[espressione ORDER BY [ASC | DESC] [NULLS FIRST | NULLS LAST] [, ...]]

Ordina un set di risultati da uno o più output `expression`.

Quando la clausola contiene più espressioni, il set di risultati è ordinato in base alla prima `expression`. Quindi la seconda `expression` viene applicata a righe con valori corrispondenti dalla prima espressione e così via.

Ogni `expression` può specificare colonne di output da `SELECT` o un numero ordinale per una colonna di output per posizione, a partire da uno.

`ORDER BY` viene valutato come ultimo passaggio dopo qualsiasi clausola `GROUP BY` o `HAVING`. `ASC` e `DESC` determinano se i risultati sono in ordine crescente o decrescente.

L'ordine predefinito nullo è `NULLS LAST`, indipendentemente dall'ordinamento crescente o decrescente.

[OFFSET count [ROW | ROWS]]

Utilizzare la clausola `OFFSET` per eliminare un certo numero di righe iniziali dal set di risultati. Se la clausola `ORDER BY` è presente, la clausola `OFFSET` viene valutata su un set di risultati ordinato e il set rimane ordinato dopo che le righe ignorate sono state scartate. Se la query non ha la clausola `ORDER BY`, è arbitrario quali righe vengono scartate. Se il conteggio specificato da `OFFSET` è uguale o maggiore delle dimensioni del set di risultati, il risultato finale è vuoto.

LIMIT [count | ALL]

Limita il numero di righe del set di risultati a `count`. `LIMIT ALL` è lo stesso omettendo la clausola `LIMIT`. Se la query non dispone di alcuna clausola `ORDER BY`, i risultati sono arbitrari.

TABLESAMPLE [BERNOULLI | SYSTEM] (percentuale)

Operatore facoltativo per selezionare righe da una tabella in base a un metodo di campionamento.

`BERNOULLI` seleziona ogni riga che deve essere nella tabella di esempio con una probabilità di `percentage`. Tutti i blocchi fisici della tabella vengono scansionati e determinate righe vengono ignorate in base a un confronto tra la `percentage` del campione e un valore casuale calcolato in fase di runtime.

Con `SYSTEM`, la tabella viene suddivisa in segmenti logici di dati e viene campionata a questa granularità.

Vengono selezionate tutte le righe da un determinato segmento oppure il segmento viene ignorato in base a un confronto tra la `percentage` campione e un valore casuale calcolato in fase di

runtime. Il campionamento SYSTEM dipende dal connettore. Questo metodo non garantisce la campionatura indipendente delle probabilità.

[UNNEST (array_or_map) [WITH ORDINALITY]]

Espande una matrice o una mappa in una relazione. Le matrice vengono espanse in una singola colonna. Le mappe vengono espanse in due colonne (chiave, valore).

È possibile utilizzare UNNEST con più argomenti, che vengono espansi in più colonne con un numero di righe pari all'argomento di cardinalità più alto.

Alle altre colonne vengono aggiunti degli zeri.

La clausola WITH ORDINALITY aggiunge un colonna di ordinalità alla fine.

UNNEST viene in genere utilizzato con un JOIN e può fare riferimento a colonne dalle relazioni a sinistra del JOIN.

Ottenere le posizioni dei file per i dati di origine in Amazon S3

Per visualizzare la posizione del file Amazon S3 per i dati in una riga di tabella, puoi utilizzare "$path" in una query SELECT, come nell'esempio seguente:

```
SELECT "$path" FROM "my_database"."my_table" WHERE year=2019;
```

Questa query restituisce un risultato come il seguente:

```
s3://DOC-EXAMPLE-BUCKET/datasets_mytable/year=2019/data_file1.json
```

Per restituire un elenco ordinato e univoco dei percorsi del nome file S3 per i dati in una tabella, è possibile utilizzare SELECT DISTINCT e ORDER BY, come nell'esempio seguente.

```
SELECT DISTINCT "$path" AS data_source_file
FROM sampledb.elb_logs
ORDER By data_source_file ASC
```

Per restituire solo i nomi dei file senza il percorso, è possibile trasmettere "$path" come parametro per una funzione regexp_extract, come nell'esempio seguente.

```
SELECT DISTINCT regexp_extract("$path", '[^/]+$') AS data_source_file
```

```
FROM sampledb.elb_logs
ORDER By data_source_file ASC
```

Per restituire i dati da un file specifico, specificare il file nella clausola `WHERE`, come nell'esempio seguente.

```
SELECT *,"$path" FROM my_database.my_table WHERE "$path" = 's3://DOC-EXAMPLE-BUCKET/
my_table/my_partition/file-01.csv'
```

Per ulteriori informazioni ed esempi, consulta l'articolo del Portale del sapere In che modo è possibile visualizzare il file sorgente Amazon S3 per cercare una riga di una tabella Athena?.

> **Note**
>
> In Athena, le colonne di metadati nascoste Hive o Iceberg e `$bucket`, `$file_modified_time`, `$file_size` e `$partition` non sono supportate per le viste.

Escape delle virgolette singole

Per eseguire una procedura di escape di virgolette singole, precederla con altre virgolette, come nell'esempio seguente. Non confondere questo con doppie virgolette.

```
Select 'O''Reilly'
```

Risultati

```
O'Reilly
```

Risorse aggiuntive

Per ulteriori informazioni sull'utilizzo delle istruzioni `SELECT` in Athena, consulta le seguenti risorse.

Per informazioni su questo	Consulta questo
Esecuzione di query in Athena	Esecuzione di query SQL con Amazon Athena
Utilizzo di `SELECT` per creare una tabella	Creazione di una tabella dai risultati delle query (CTAS)

Per informazioni su questo	Consulta questo
Inserimento di dati da una query `SELECT` in un'altra tabella	INSERT INTO
Utilizzo di funzioni integrate nelle istruzioni `SELECT`	Funzioni in Amazon Athena
Utilizzo di funzioni definite dall'utente nelle istruzioni `SELECT`	Esecuzione di query con funzioni definite dall'utente
Esecuzione di query sui metadati del catalogo dati	Esecuzione di query AWS Glue Data Catalog

INSERT INTO

Inserisce nuove righe in una tabella di destinazione in base a un'istruzione di query `SELECT` eseguita su una tabella di origine o in base a un set di `VALUES` fornito come parte dell'istruzione. Quando la tabella di origine si basa sui dati sottostanti in un formato, ad esempio CSV o JSON, e la tabella di destinazione si basa su un altro formato, ad esempio Parquet o ORC, puoi utilizzare le query `INSERT INTO` per trasformare i dati selezionati nel formato della tabella di destinazione.

Considerazioni e limitazioni

Considera le informazioni seguenti durante l'utilizzo delle query `INSERT` con Athena.

- Quando si esegue una query `INSERT` su una tabella con i dati sottostanti crittografati in Amazon S3, i file di output scritti dalla query `INSERT` non vengono crittografati per impostazione predefinita. Ti consigliamo di crittografare i risultati delle query `INSERT` se stai inserendo nelle tabelle dati crittografati.

  Per ulteriori informazioni sulla crittografia dei risultati delle query tramite la console, consulta Crittografia dei risultati di query Athena archiviati in Amazon S3. Per abilitare la crittografia utilizzando l'API AWS CLI o Athena, utilizza `EncryptionConfiguration` le proprietà dell'StartQueryExecutionazione per specificare le opzioni di crittografia di Amazon S3 in base ai tuoi requisiti.
- Per le istruzioni `INSERT INTO`, l'impostazione prevista per il proprietario del bucket non si applica alla posizione della tabella di destinazione in Amazon S3. L'impostazione prevista per il proprietario

del bucket si applica solo al percorso di output di Amazon S3 specificato per i risultati delle query di Athena. Per ulteriori informazioni, consulta Specificare una posizione dei risultati delle query utilizzando la console Athena.

- Per le istruzioni `INSERT INTO` conformi all'ACID, consulta la sezione `INSERT INTO` della pagina Aggiornamento dati di tabelle Iceberg.

Formati supportati e SerDes

È possibile eseguire un'`INSERT`interrogazione su tabelle create a partire da dati con i seguenti formati e SerDes.

Formato dei dati	SerDe
Avro	org.apache.hadoop.hive.serde2.avro. AvroSerDe
Ion	com.amazon.ionhiveserde. IonHiveSerDe
JSON	org.apache.hive.hcatalog.data. JsonSerDe
ORC	org.apache.hadoop.hive.ql.io.orc. OrcSerde
Parquet	org.apache.hadoop.hive.ql.io.parquet.serde. ParquetHiveSerDe
File di testo	org.apache.hadoop.hive.serde2.lazy. LazySimpleSerDe **Note** Sono supportati CSV, TSV e file con delimitatori personalizzati.

Tabelle con bucket non supportate

`INSERT INTO` non è supportato nelle tabelle con bucket. Per ulteriori informazioni, consulta Partizionamento e arrotolamento in Athena.

Query federate non supportate

`INSERT INTO` non è supportato per le query federate. Il tentativo di eseguire questa operazione potrebbe generare il messaggio di errore This operation is currently not supported for external catalogs (Questa operazione non è attualmente supportata per i cataloghi esterni). Per informazioni sulle query federate, consultare Utilizzo di Amazon Athena Federated Query.

Partizionamento

Tieni in considerazione i punti esposti in questa sezione quando utilizzi il partizionamento con le query `INSERT INTO` o `CREATE TABLE AS SELECT`.

Limiti

L'istruzione `INSERT INTO` supporta la scrittura di un massimo di 100 partizioni nella tabella di destinazione. Se si esegue la clausola `SELECT` su una tabella con più di 100 partizioni, la query non riesce a meno che la query `SELECT` non sia limitata a massimo 100 partizioni.

Per informazioni su una soluzione alternativa per questa limitazione, consulta Utilizzo di CTAS e INSERT INTO per aggirare il limite di 100 partizioni.

Ordinamento colonne

Le istruzioni `INSERT INTO` o `CREATE TABLE AS SELECT` prevedono che la colonna partizionata sia l'ultima colonna dell'elenco delle colonne proiettate in un'istruzione `SELECT`.

Se la tabella di origine non è partizionata o partizionata su colonne diverse rispetto alla tabella di destinazione, query come `INSERT INTO` *`destination_table`* `SELECT * FROM` *`source_table`* considerano i valori nell'ultima colonna della tabella di origine come valori per una colonna di partizione nella tabella di destinazione. Tieni presenti queste informazioni quando provi a creare una tabella partizionata da una tabella non partizionata.

Risorse

Per ulteriori informazioni sull'utilizzo di `INSERT INTO` con il partizionamento, consulta le seguenti risorse.

- Per l'inserimento di dati partizionati in una tabella partizionata, consulta Utilizzo di CTAS e INSERT INTO per aggirare il limite di 100 partizioni.
- Per l'inserimento di dati non partizionati in una tabella partizionata, consulta Utilizzo di CTAS e INSERT INTO per ETL e analisi dei dati.

File scritti su Amazon S3

Athena scrive file nelle posizioni dei dati di origine in Amazon S3 come risultato del comando `INSERT`. Ogni operazione `INSERT` crea un nuovo file, anziché aggiungere i dati a un file esistente. Le posizioni dei file dipendono dalla struttura della tabella e dalla query `SELECT`, se presente. Athena genera un file manifest di dati per ogni query `INSERT`. Il manifest tiene traccia dei file scritti dalla query. Viene salvato nella posizione dei risultati di query Athena in Amazon S3. Per ulteriori informazioni, consulta Identificazione dei file di output delle query.

Evita aggiornamenti altamente transazionali

Quando aggiungi righe `INSERT INTO` a una tabella in Amazon S3, Athena non riscrive o modifica i file esistenti. Al contrario, scrive le righe come uno o più file nuovi. Poiché le tabelle con molti file di piccole dimensioni riducono le prestazioni delle query e le operazioni di scrittura `PutObject` e lettura come Amazon S3 `GetObject` comportano costi più elevati, considera le seguenti opzioni quando utilizzi: `INSERT INTO`

- Esegui `INSERT INTO` le operazioni meno frequentemente su batch di righe più grandi.
- Per grandi volumi di ingestione di dati, prendi in considerazione l'utilizzo di un servizio come Amazon Data Firehose.
- Evita di utilizzarlo del tutto. `INSERT INTO` Invece, accumula le righe in file più grandi e caricale direttamente su Amazon S3, dove Athena potrà interrogarle.

Individuazione di file orfani

Se un'`INSERT INTO`istruzione `CTAS` or fallisce, i dati orfani possono essere lasciati nella posizione dei dati e potrebbero essere letti nelle query successive. Per individuare i file orfani per l'ispezione o l'eliminazione, è possibile utilizzare il file manifesto dati fornito da Athena per tenere traccia dell'elenco dei file da scrivere. Per ulteriori informazioni, consulta Identificazione dei file di output delle query e DataManifestLocation.

INSERT INTO...SELECT

Specifica la query da eseguire su una tabella, `source_table`, che determina le righe da inserire in una seconda tabella, `destination_table`. Se la query `SELECT` specifica le colonne in `source_table`, le colonne devono corrispondere esattamente a quelle in `destination_table`.

Per ulteriori informazioni sulle query `SELECT`, consulta SELECT.

Riepilogo

```
INSERT INTO destination_table
SELECT select_query
FROM source_table_or_view
```

Esempi

Seleziona tutte le righe nella tabella `vancouver_pageviews` e inseriscile nella tabella `canada_pageviews`:

```
INSERT INTO canada_pageviews
SELECT *
FROM vancouver_pageviews;
```

Seleziona solo le righe nella tabella `vancouver_pageviews` in cui la colonna `date` ha un valore compreso tra `2019-07-01` e `2019-07-31`, quindi inseriscile in `canada_july_pageviews`:

```
INSERT INTO canada_july_pageviews
SELECT *
FROM vancouver_pageviews
WHERE date
    BETWEEN date '2019-07-01'
        AND '2019-07-31';
```

Seleziona i valori nelle colonne `state` e `city` nella tabella `cities_world` solo dalle righe con il valore `usa` nella colonna `country` e inseriscile nelle colonne `state` e `city` della tabella `cities_usa`:

```
INSERT INTO cities_usa (city,state)
SELECT city,state
FROM cities_world
    WHERE country='usa'
```

INSERT INTO...VALUES

Inserisce le righe in una tabella esistente specificando colonne e valori. Le colonne specificate e i tipi di dati associati devono corrispondere esattamente alle colonne e ai tipi di dati nella tabella di destinazione.

> **Important**
>
> Non è consigliabile inserire righe utilizzando `VALUES` perché Athena genera file per ogni operazione `INSERT`. Questo può causare la creazione di molti file di piccole dimensioni e il peggioramento delle prestazioni delle query della tabella. Per identificare i file creati da una query `INSERT`, esamina il file manifest dei dati. Per ulteriori informazioni, consulta Utilizzo dei risultati delle query, delle query recenti e dei file di output.

Riepilogo

```
INSERT INTO destination_table [(col1,col2,...)]
VALUES (col1value,col2value,...)[,
       (col1value,col2value,...)][,
       ...]
```

Esempi

Negli esempi seguenti, la tabella delle città ha tre colonne: `id`, `city`, `state`, `state_motto`. La colonna `id` è di tipo `INT` e tutte le altre colonne sono di tipo `VARCHAR`.

Inserisci una singola riga nella tabella `cities`, con tutti i valori di colonna specificati:

```
INSERT INTO cities
VALUES (1,'Lansing','MI','Si quaeris peninsulam amoenam circumspice')
```

Inserisci due righe nella tabella `cities`:

```
INSERT INTO cities
VALUES (1,'Lansing','MI','Si quaeris peninsulam amoenam circumspice'),
       (3,'Boise','ID','Esto perpetua')
```

DELETE

Elimina le righe di una tabella Apache Iceberg. `DELETE` è transazionale ed è supportato solo per le tabelle Apache Iceberg.

Riepilogo

Per eliminare le righe di una tabella Iceberg, utilizza la sintassi seguente.

```
DELETE FROM [db_name.]table_name [WHERE predicate]
```

Per ulteriori informazioni, consulta la sezione DELETE della pagina Aggiornamento dati di tabelle Iceberg.

UPDATE

Aggiorna le righe di una tabella Apache Iceberg. UPDATE è transazionale ed è supportato solo per le tabelle Apache Iceberg.

Riepilogo

Per aggiornare le righe di una tabella Iceberg, utilizza la sintassi seguente.

```
UPDATE [db_name.]table_name SET xx=yy[,...] [WHERE predicate]
```

Per ulteriori informazioni, consulta la sezione UPDATE della pagina Aggiornamento dati di tabelle Iceberg.

MERGE INTO

Aggiorna, elimina o inserisce in modo condizionale righe in una tabella Apache Iceberg. Una singola istruzione può combinare operazioni di aggiornamento, eliminazione e inserimento.

> **Note**
>
> L'istruzione MERGE INTO è transazionale ed è supportata solo per le tabelle Apache Iceberg nella versione 3 del motore Athena.

Riepilogo

Per aggiornare, eliminare o inserire in modo condizionale righe di una tabella Iceberg, utilizza la sintassi seguente.

```
MERGE INTO target_table [ [ AS ]  target_alias ]
USING { source_table | query } [ [ AS ] source_alias ]
ON search_condition
```

```
when_clause [...]
```

La clausola *when_clause* è una delle seguenti:

```
WHEN MATCHED [ AND condition ]
    THEN DELETE
```

```
WHEN MATCHED [ AND condition ]
    THEN UPDATE SET ( column = expression [, ...] )
```

```
WHEN NOT MATCHED [ AND condition ]
    THEN INSERT (column_name[, column_name ...]) VALUES (expression, ...)
```

MERGE supporta un numero arbitrario di clausole WHEN con condizioni MATCHED diverse. Le clausole condizionali eseguono l'operazione DELETE, UPDATE o INSERT nella prima clausola WHEN selezionata dallo stato MATCHED e dalla condizione corrispondente.

Per ogni riga di origine, le clausole WHEN vengono elaborate in ordine. Viene eseguita solo la prima clausola WHEN corrispondente. Le clausole successive vengono ignorate. Viene generato un errore utente quando una singola riga della tabella di destinazione corrisponde a più di una riga di origine.

Se una riga di origine non corrisponde ad alcuna clausola WHEN e non è presente alcuna clausola WHEN NOT MATCHED, la riga di origine viene ignorata.

Nelle clausole WHEN che prevedono operazioni UPDATE, le espressioni dei valori delle colonne possono fare riferimento a qualsiasi campo della destinazione o dell'origine. Nel caso di NOT MATCHED, le espressioni INSERT possono fare riferimento a qualsiasi campo dell'origine.

Esempio

L'esempio seguente unisce le righe della seconda tabella alla prima tabella se le righe non esistono nella prima tabella. Tieni presente che le colonne elencate nella clausola VALUES devono essere precedute dall'alias della tabella di origine. Le colonne di destinazione elencate nella clausola INSERT non devono avere questo prefisso.

```
MERGE INTO iceberg_table_sample as ice1
USING iceberg2_table_sample as ice2
ON ice1.col1 = ice2.col1
```

```
WHEN NOT MATCHED
THEN INSERT (col1)
      VALUES (ice2.col1)
```

Per ulteriori esempi `MERGE INTO`, consulta Aggiornamento dati di tabelle Iceberg.

OPTIMIZE

Ottimizza le righe in una tabella Apache Iceberg riscrivendo i file di dati in un layout più ottimizzato in base alle dimensioni e al numero di file di eliminazione associati.

> **Note**
>
> `OPTIMIZE` è transazionale ed è supportato solo per le tabelle Apache Iceberg.

Sintassi

Il riepilogo della sintassi seguente mostra come ottimizzare il layout dei dati per una tabella Iceberg.

```
OPTIMIZE [db_name.]table_name REWRITE DATA USING BIN_PACK
  [WHERE predicate]
```

> **Note**
>
> *Nel predicato della WHERE clausola sono consentite solo le colonne di partizione.* Se si specifica una colonna non di partizione, la query avrà esito negativo.

L'azione di compattazione viene addebitata dalla quantità di dati scansionati durante il processo di riscrittura. L'operazione `REWRITE DATA` utilizza i predicati per selezionare i file che contengono righe corrispondenti. Se una riga del file corrisponde al predicato, il file viene selezionato per l'ottimizzazione. Pertanto, per controllare il numero di file interessati dall'operazione di compattazione, è possibile specificare una clausola `WHERE`.

Configurazione delle proprietà di compattazione

Per controllare la dimensione dei file da selezionare per la compattazione e la dimensione del file risultante dopo la compattazione, è possibile utilizzare i parametri delle proprietà della tabella. Puoi

utilizzare il comando ALTER TABLE SET PROPERTIES per configurare le seguenti proprietà della tabella:

Risorse aggiuntive

Ottimizzazione delle tabelle Iceberg

VACUUM

L'istruzione `VACUUM` esegue la manutenzione delle tabelle Apache Iceberg rimuovendo i file di dati non più necessari.

> **Note**
>
> L'istruzione `VACUUM` è transazionale ed è supportata solo per le tabelle Apache Iceberg nella versione 3 del motore Athena.

Si consiglia di eseguire l'istruzione `VACUUM` sulle tabelle Iceberg per rimuovere i file di dati che non sono più pertinenti e per ridurre le dimensioni dei metadati e il consumo di spazio di archiviazione. Tieni presente che, poiché l'istruzione `VACUUM` effettua chiamate API ad Amazon S3, vengono addebitati costi per le richieste associate ad Amazon S3.

> **Warning**
>
> Se si esegue un'operazione di scadenza delle snapshot, non è più possibile accedere alle snapshot scadute.

Riepilogo

Per rimuovere i file di dati non più necessari per una tabella Iceberg, utilizza la sintassi seguente.

```
VACUUM [database_name.]target_table
```

Per eseguire l'esecuzione `VACUUM` su una tabella il cui nome inizia con un carattere di sottolineatura (ad esempio,`_mytable`), racchiudete il nome della tabella tra segni di spunta rovesciati, come nell'esempio seguente. Se anteponete al nome della tabella un nome di database, non racchiudete

il nome del database tra i backtick. Nota che le virgolette doppie non funzioneranno al posto dei backtick.

Questo comportamento è specifico di. `VACUUM` Le `INSERT INTO` istruzioni `CREATE` and non richiedono il backtick per i nomi di tabella che iniziano con caratteri di sottolineatura.

```
VACUUM `_mytable`
VACUUM my_database.`_mytable`
```

Tieni inoltre presente che `VACUUM` si prevede che i dati Iceberg si trovino in una cartella Amazon S3 anziché in un bucket Amazon S3. Ad esempio, se i dati di Iceberg si trovano in`s3://DOC-EXAMPLE-BUCKET/`anziché in`s3://DOC-EXAMPLE-BUCKET/myicebergfolder/`, l'`VACUUM`istruzione ha esito negativo e viene visualizzato il messaggio di errore GENERIC_INTERNAL_ERROR: Path missing in file system location:. `s3://DOC-EXAMPLE-BUCKET`

Operazioni eseguite

`VACUUM` esegue le seguenti operazioni:

- Rimuove gli snapshot che sono più vecchi del periodo di tempo specificato dalla proprietà della tabella `vacuum_max_snapshot_age_seconds`. Per impostazione predefinita, questa proprietà è impostata su 432.000 secondi (5 giorni).
- Rimuove gli snapshot che non rientrano nel periodo da mantenere e che superano il numero specificato dalla proprietà della tabella `vacuum_min_snapshots_to_keep`. Il valore di default è 1.

  Puoi specificare queste proprietà della tabella nell'istruzione `CREATE TABLE`. Dopo aver creato la tabella, puoi utilizzare l'istruzione ALTER TABLE SET PROPERTIES per aggiornarle.
- Rimuove tutti i metadati e i file di dati che non sono raggiungibili a seguito della rimozione dello snapshot. Puoi configurare il numero di vecchi file di metadati da conservare impostando la proprietà della tabella `vacuum_max_metadata_files_to_keep`. Il valore predefinito è 100.
- Rimuove i file orfani più vecchi del tempo specificato nella proprietà della tabella `vacuum_max_snapshot_age_seconds`. I file orfani sono file nella directory dei dati della tabella che non fanno parte dello stato della tabella.

Per ulteriori informazioni sulla creazione e sulla gestione delle tabelle Apache Iceberg in Athena, consulta le sezioni Creazione di tabelle Iceberg e Gestione di tabelle Iceberg.

Utilizzo di EXPLAIN e EXPLAIN ANALYZE in Athena

L'istruzione `EXPLAIN` mostra il piano di esecuzione logico o distribuito di un'istruzione SQL specificata o convalida l'istruzione SQL. È possibile generare i risultati in formato testo o in formato dati per il rendering in un grafico.

> **Note**
>
> È possibile visualizzare rappresentazioni grafiche di piani logici e distribuiti per le proprie query nella console Athena senza utilizzare la sintassi `EXPLAIN`. Per ulteriori informazioni, consulta Visualizzazione dei piani di esecuzione per query SQL.

L'istruzione `EXPLAIN ANALYZE` mostra sia il piano di esecuzione distribuito di un'istruzione SQL specificata che il costo computazionale di ciascuna operazione in una query SQL. È possibile eseguire l'output dei risultati in formato testo o JSON.

Considerazioni e limitazioni

Le istruzioni `EXPLAIN` e `EXPLAIN ANALYZE` in Athena hanno le seguenti limitazioni.

- Poiché le query `EXPLAIN` non scansionano alcun dato, Athena non addebita alcun costo per loro. Tuttavia, poiché le query `EXPLAIN` effettuano chiamate a AWS Glue per recuperare i metadati della tabella, potresti ricevere addebiti da Glue se le chiamate superano il limite del piano gratuito per Glue.
- Poiché vengono eseguite le query `EXPLAIN ANALYZE`, viene eseguita la scansione dei dati e Athena addebita la quantità di dati scansionati.
- Le informazioni sul filtraggio di righe o celle definite in Lake Formation e le informazioni sulle statistiche delle query non vengono visualizzate nell'output di `EXPLAIN` e `EXPLAIN ANALYZE`.

Sintassi EXPLAIN

```
EXPLAIN [ ( option [, ...]) ] statement
```

Il valore *`opzione`* può essere uno dei seguenti:

```
FORMAT { TEXT | GRAPHVIZ | JSON }
```

```
TYPE { LOGICAL | DISTRIBUTED | VALIDATE | IO }
```

Se l'opzione `FORMAT` non è specificata, l'output avrà il formato di default `TEXT`. Il tipo `IO` fornisce informazioni sulle tabelle e sugli schemi letti dalla query. `IO` è supportato solo nella versione 2 del motore Athena e può essere restituito solo in formato JSON.

Sintassi EXPLAIN ANALYZE

Oltre all'output incluso in `EXPLAIN`, l'output `EXPLAIN ANALYZE` include anche statistiche del runtime per la query specificata, ad esempio l'utilizzo della CPU, l'input del numero di righe e l'output del numero di righe.

```
EXPLAIN ANALYZE [ ( option [, ...]) ] statement
```

Il valore *`opzione`* può essere uno dei seguenti:

```
FORMAT { TEXT | JSON }
```

Se l'opzione `FORMAT` non è specificata, l'output avrà il formato di default `TEXT`. Perché tutte le query per `EXPLAIN ANALYZE` sono `DISTRIBUTED`, l'opzione `TYPE` non è disponibile per `EXPLAIN ANALYZE`.

L'*`istruzione`* può avere uno dei seguenti valori:

```
SELECT
CREATE TABLE AS SELECT
INSERT
UNLOAD
```

Esempi di EXPLAIN

Gli esempi seguenti per l'avanzamento di `EXPLAIN` vanno dai più semplici ai più complessi.

EXPLAIN esempio 1: utilizza l'istruzione EXPLAIN per mostrare un piano di query in formato testo.

Nell'esempio seguente, `EXPLAIN` mostra il piano di esecuzione per una query `SELECT` nei log di Elastic Load Balancing. Il formato viene impostato per impostazione predefinita sull'output di testo.

```
EXPLAIN
```

```
SELECT
   request_timestamp,
   elb_name,
   request_ip
FROM sampledb.elb_logs;
```

Risultati

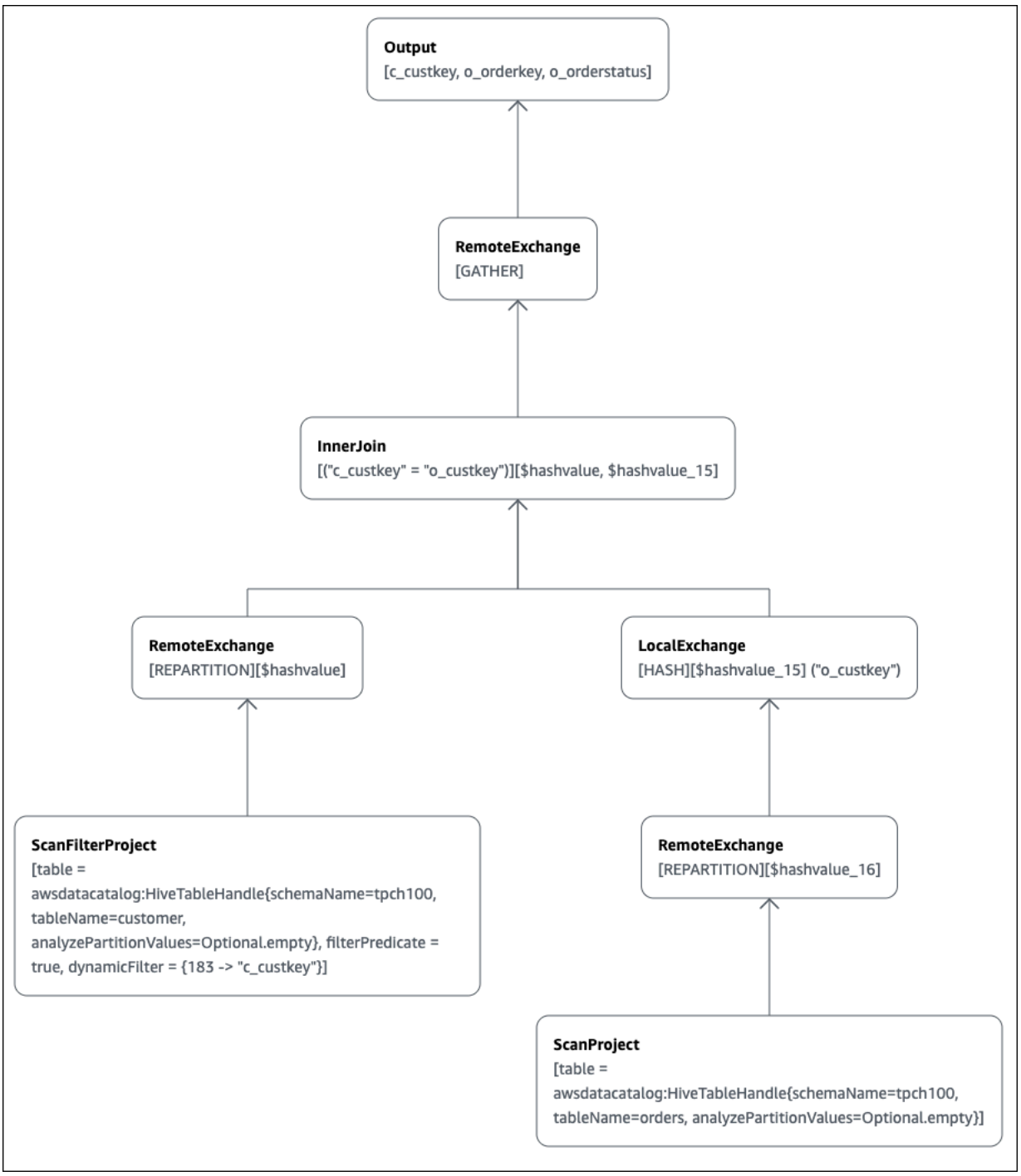
```
- Output[request_timestamp, elb_name, request_ip] => [[request_timestamp, elb_name,
 request_ip]]
    - RemoteExchange[GATHER] => [[request_timestamp, elb_name, request_ip]]
        - TableScan[awsdatacatalog:HiveTableHandle{schemaName=sampledb,
 tableName=elb_logs,
analyzePartitionValues=Optional.empty}] => [[request_timestamp, elb_name, request_ip]]
                LAYOUT: sampledb.elb_logs
                request_ip := request_ip:string:2:REGULAR
                request_timestamp := request_timestamp:string:0:REGULAR
                elb_name := elb_name:string:1:REGULAR
```

EXPLAIN esempio 2: esegui un grafico del piano di query

Puoi usare la console Athena per eseguire un grafico su un piano di query. Inserisci un'istruzione `SELECT` come la seguente nell'editor di query, quindi seleziona Run (Esegui).

```
SELECT
      c.c_custkey,
      o.o_orderkey,
      o.o_orderstatus
   FROM tpch100.customer c
   JOIN tpch100.orders o
       ON c.c_custkey = o.o_custkey
```

La pagina Explain dell'editor di query Athena si apre e mostra un piano distribuito e un piano logico per la query. Il seguente grafico mostra il piano logico per l'esempio.



Important

Attualmente, alcuni filtri di partizione potrebbero non essere visibili nel grafico ad albero degli operatori annidato anche se Athena li applica alla tua query. Per verificare l'effetto di tali filtri, esegui `EXPLAIN` o `EXPLAIN ANALYZE` sulla query e visualizza i risultati.

Per ulteriori informazioni sull'utilizzo delle funzionalità grafiche del piano di query nella console Athena, consulta Visualizzazione dei piani di esecuzione per query SQL.

EXPLAIN esempio 3: utilizza l'istruzione EXPLAIN per verificare la cesura delle partizioni.

Quando si utilizza un predicato di filtro in una chiave partizionata per eseguire una query su una tabella partizionata, il motore di query applica il predicato alla chiave partizionata per ridurre la quantità di dati letti.

Nell'esempio seguente viene utilizzato una query `EXPLAIN` per verificare la cesura delle partizioni per una query `SELECT` su una tabella partizionata. Innanzitutto, un'istruzione `CREATE TABLE` crea la tabella `tpch100.orders_partitioned`. La tabella è partizionata sulla colonna `o_orderdate`.

```
CREATE TABLE `tpch100.orders_partitioned`(
  `o_orderkey` int,
  `o_custkey` int,
  `o_orderstatus` string,
  `o_totalprice` double,
  `o_orderpriority` string,
  `o_clerk` string,
  `o_shippriority` int,
  `o_comment` string)
PARTITIONED BY (
  `o_orderdate` string)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
STORED AS INPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat'
OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION
  's3://DOC-EXAMPLE-BUCKET/<your_directory_path>/'
```

La tabella `tpch100.orders_partitioned` ha diverse partizioni su `o_orderdate`, come mostrato dal comando `SHOW PARTITIONS`.

```
SHOW PARTITIONS tpch100.orders_partitioned;

o_orderdate=1994
o_orderdate=2015
o_orderdate=1998
o_orderdate=1995
```

```
o_orderdate=1993
o_orderdate=1997
o_orderdate=1992
o_orderdate=1996
```

La seguente query EXPLAIN verifica la cesura delle partizioni sull'istruzione SELECTspecificata.

```
EXPLAIN
SELECT
   o_orderkey,
   o_custkey,
   o_orderdate
FROM tpch100.orders_partitioned
WHERE o_orderdate = '1995'
```

Risultati

```
Query Plan
- Output[o_orderkey, o_custkey, o_orderdate] => [[o_orderkey, o_custkey, o_orderdate]]
    - RemoteExchange[GATHER] => [[o_orderkey, o_custkey, o_orderdate]]
        - TableScan[awsdatacatalog:HiveTableHandle{schemaName=tpch100,
 tableName=orders_partitioned,
analyzePartitionValues=Optional.empty}] => [[o_orderkey, o_custkey, o_orderdate]]
                LAYOUT: tpch100.orders_partitioned
                o_orderdate := o_orderdate:string:-1:PARTITION_KEY
                    :: [[1995]]
                o_custkey := o_custkey:int:1:REGULAR
                o_orderkey := o_orderkey:int:0:REGULAR
```

Il testo in grassetto nel risultato mostra che il predicato o_orderdate = '1995' è stato applicato su PARTITION_KEY.

EXPLAIN esempio 4: utilizza una query EXPLAIN per verificare l'ordine e il tipo di join.

La seguente query EXPLAIN verifica l'ordine di join e il tipo di join dell'istruzione SELECT. Utilizzare una query come questa per esaminare l'utilizzo della memoria della query in modo da ridurre le possibilità di ottenere un errore EXCEEDED_LOCAL_MEMORY_LIMIT.

```
EXPLAIN (TYPE DISTRIBUTED)
   SELECT
      c.c_custkey,
```

```
      o.o_orderkey,
      o.o_orderstatus
   FROM tpch100.customer c
   JOIN tpch100.orders o
       ON c.c_custkey = o.o_custkey
   WHERE c.c_custkey = 123
```

Risultati

```
Query Plan
Fragment 0 [SINGLE]
    Output layout: [c_custkey, o_orderkey, o_orderstatus]
    Output partitioning: SINGLE []
    Stage Execution Strategy: UNGROUPED_EXECUTION
    - Output[c_custkey, o_orderkey, o_orderstatus] => [[c_custkey, o_orderkey,
 o_orderstatus]]
        - RemoteSource[1] => [[c_custkey, o_orderstatus, o_orderkey]]

Fragment 1 [SOURCE]
    Output layout: [c_custkey, o_orderstatus, o_orderkey]
    Output partitioning: SINGLE []
    Stage Execution Strategy: UNGROUPED_EXECUTION
    - CrossJoin => [[c_custkey, o_orderstatus, o_orderkey]]
            Distribution: REPLICATED
        - ScanFilter[table = awsdatacatalog:HiveTableHandle{schemaName=tpch100,
tableName=customer, analyzePartitionValues=Optional.empty}, grouped = false,
filterPredicate = ("c_custkey" = 123)] => [[c_custkey]]
                LAYOUT: tpch100.customer
                c_custkey := c_custkey:int:0:REGULAR
        - LocalExchange[SINGLE] () => [[o_orderstatus, o_orderkey]]
            - RemoteSource[2] => [[o_orderstatus, o_orderkey]]

Fragment 2 [SOURCE]
    Output layout: [o_orderstatus, o_orderkey]
    Output partitioning: BROADCAST []
    Stage Execution Strategy: UNGROUPED_EXECUTION
    - ScanFilterProject[table = awsdatacatalog:HiveTableHandle{schemaName=tpch100,
tableName=orders, analyzePartitionValues=Optional.empty}, grouped = false,
filterPredicate = ("o_custkey" = 123)] => [[o_orderstatus, o_orderkey]]
            LAYOUT: tpch100.orders
            o_orderstatus := o_orderstatus:string:2:REGULAR
            o_custkey := o_custkey:int:1:REGULAR
            o_orderkey := o_orderkey:int:0:REGULAR
```

La query di esempio è stata ottimizzata in un cross join per prestazioni migliori. I risultati mostrano che `tpch100.orders` sarà distribuito come tipo di distribuzione `BROADCAST`. Ciò implica che la tabella `tpch100.orders` verrà distribuita a tutti i nodi che eseguono l'operazione di join. Il tipo di distribuzione `BROADCAST` richiederà che tutti i risultati filtrati della tabella `tpch100.orders` siano inseriti nella memoria di ogni nodo che esegue l'operazione di join.

Tuttavia, la tabella `tpch100.customer` è più piccola di `tpch100.orders`. Poiché `tpch100.customer` richiede meno memoria, è possibile riscrivere la query su `BROADCAST` `tpch100.customer` invece che su `tpch100.orders`. In questo modo si riduce la possibilità che la query riceva l'errore `EXCEEDED_LOCAL_MEMORY_LIMIT`. La strategia presuppone i seguenti punti:

- `tpch100.customer.c_custkey` è unico nella tabella `tpch100.customer`.
- Esiste una relazione di one-to-many mappatura tra `tpch100.customer` e`tpch100.orders`.

L'esempio seguente mostra la query riscritta.

```
SELECT
    c.c_custkey,
    o.o_orderkey,
    o.o_orderstatus
FROM tpch100.orders o
JOIN tpch100.customer c -- the filtered results of tpch100.customer are distributed to
 all nodes.
    ON c.c_custkey = o.o_custkey
WHERE c.c_custkey = 123
```

EXPLAIN esempio 5: utilizza una query EXPLAIN per rimuovere i predicati che non hanno effetto

Puoi utilizzare una query `EXPLAIN` per verificare l'efficacia dei predicati di filtraggio. È possibile utilizzare i risultati per rimuovere i predicati che non hanno alcun effetto, come nell'esempio seguente.

```
EXPLAIN
   SELECT
      c.c_name
   FROM tpch100.customer c
   WHERE c.c_custkey = CAST(RANDOM() * 1000 AS INT)
   AND c.c_custkey BETWEEN 1000 AND 2000
   AND c.c_custkey = 1500
```

Risultati

```
Query Plan
- Output[c_name] => [[c_name]]
    - RemoteExchange[GATHER] => [[c_name]]
        - ScanFilterProject[table =
awsdatacatalog:HiveTableHandle{schemaName=tpch100,
tableName=customer, analyzePartitionValues=Optional.empty},
filterPredicate = (("c_custkey" = 1500) AND ("c_custkey" =
CAST(("random"() * 1E3) AS int)))] => [[c_name]]
                LAYOUT: tpch100.customer
                c_custkey := c_custkey:int:0:REGULAR
                c_name := c_name:string:1:REGULAR
```

`filterPredicate` nei risultati mostra che l'ottimizzatore ha unito i tre predicati originali in due predicati e ha cambiato il loro ordine di applicazione.

```
filterPredicate = (("c_custkey" = 1500) AND ("c_custkey" = CAST(("random"() * 1E3) AS
 int)))
```

Poiché i risultati mostrano che il predicato `AND c.c_custkey BETWEEN 1000 AND 2000` non ha alcun effetto, è possibile rimuovere questo predicato senza modificare i risultati della query.

Per informazioni sui termini utilizzati nei risultati delle query `EXPLAIN`, consulta Capire i risultati dell'istruzione EXPLAIN di Athena.

Esempi di EXPLAIN ANALYZE

Gli esempi seguenti mostrano query e output `EXPLAIN ANALYZE` di esempio.

EXPLAIN ANALYZE esempio 1: utilizza EXPLAIN ANALYZE per mostrare il piano di una query e il costo computazionale in formato di testo.

Nell'esempio seguente, vengono `EXPLAIN ANALYZE` illustrati il piano di esecuzione e i costi di calcolo per una `SELECT` query sui CloudFront log. Il formato viene impostato per impostazione predefinita sull'output di testo.

```
EXPLAIN ANALYZE SELECT FROM cloudfront_logs LIMIT 10
```

Risultati

```
 Fragment 1
```

```
     CPU: 24.60ms, Input: 10 rows (1.48kB); per task: std.dev.: 0.00, Output: 10 rows
 (1.48kB)
     Output layout: [date, time, location, bytes, requestip, method, host, uri, status,
 referrer,\
       os, browser, browserversion]
Limit[10] => [[date, time, location, bytes, requestip, method, host, uri, status,
 referrer, os,\
  browser, browserversion]]
             CPU: 1.00ms (0.03%), Output: 10 rows (1.48kB)
             Input avg.: 10.00 rows, Input std.dev.: 0.00%
LocalExchange[SINGLE] () => [[date, time, location, bytes, requestip, method, host,
 uri, status, referrer, os,\
 browser, browserversion]]
                 CPU: 0.00ns (0.00%), Output: 10 rows (1.48kB)
                 Input avg.: 0.63 rows, Input std.dev.: 387.30%
RemoteSource[2] => [[date, time, location, bytes, requestip, method, host, uri, status,
 referrer, os,\
  browser, browserversion]]
                     CPU: 1.00ms (0.03%), Output: 10 rows (1.48kB)
                     Input avg.: 0.63 rows, Input std.dev.: 387.30%

 Fragment 2
     CPU: 3.83s, Input: 998 rows (147.21kB); per task: std.dev.: 0.00, Output: 20 rows
 (2.95kB)
     Output layout: [date, time, location, bytes, requestip, method, host, uri, status,
 referrer, os,\
       browser, browserversion]
LimitPartial[10] => [[date, time, location, bytes, requestip, method, host, uri,
 status, referrer, os,\
  browser, browserversion]]
             CPU: 5.00ms (0.13%), Output: 20 rows (2.95kB)
             Input avg.: 166.33 rows, Input std.dev.: 141.42%
TableScan[awsdatacatalog:HiveTableHandle{schemaName=default, tableName=cloudfront_logs,
\
  analyzePartitionValues=Optional.empty},
grouped = false] => [[date, time, location, bytes, requestip, method, host, uri, st
                 CPU: 3.82s (99.82%), Output: 998 rows (147.21kB)
                 Input avg.: 166.33 rows, Input std.dev.: 141.42%
                 LAYOUT: default.cloudfront_logs
                 date := date:date:0:REGULAR
                 referrer := referrer:string:9:REGULAR
                 os := os:string:10:REGULAR
                 method := method:string:5:REGULAR
                 bytes := bytes:int:3:REGULAR
```

```
                browser := browser:string:11:REGULAR
                host := host:string:6:REGULAR
                requestip := requestip:string:4:REGULAR
                location := location:string:2:REGULAR
                time := time:string:1:REGULAR
                uri := uri:string:7:REGULAR
                browserversion := browserversion:string:12:REGULAR
                status := status:int:8:REGULAR
```

EXPLAIN ANALYZE esempio 2. utilizza EXPLAIN ANALYZE per mostrare un piano di query in formato JSON.

L'esempio seguente mostra il piano di esecuzione e i costi di calcolo per una `SELECT` query sui log. CloudFront L'esempio specifica JSON come formato di output.

```
EXPLAIN ANALYZE (FORMAT JSON) SELECT * FROM cloudfront_logs LIMIT 10
```

Risultati

```
{
    "fragments": [{
        "id": "1",

        "stageStats": {
            "totalCpuTime": "3.31ms",
            "inputRows": "10 rows",
            "inputDataSize": "1514B",
            "stdDevInputRows": "0.00",
            "outputRows": "10 rows",
            "outputDataSize": "1514B"
        },
        "outputLayout": "date, time, location, bytes, requestip, method, host,\
           uri, status, referrer, os, browser, browserversion",

        "logicalPlan": {
            "1": [{
                "name": "Limit",
                "identifier": "[10]",
                "outputs": ["date", "time", "location", "bytes", "requestip", "method",
 "host",\
                  "uri", "status", "referrer", "os", "browser", "browserversion"],
                "details": "",
                "distributedNodeStats": {
```

```
                    "nodeCpuTime": "0.00ns",
                    "nodeOutputRows": 10,
                    "nodeOutputDataSize": "1514B",
                    "operatorInputRowsStats": [{
                        "nodeInputRows": 10.0,
                        "nodeInputRowsStdDev": 0.0
                    }]
                },
                "children": [{
                    "name": "LocalExchange",
                    "identifier": "[SINGLE] ()",
                    "outputs": ["date", "time", "location", "bytes", "requestip",
"method", "host",\
                      "uri", "status", "referrer", "os", "browser", "browserversion"],
                    "details": "",
                    "distributedNodeStats": {
                        "nodeCpuTime": "0.00ns",
                        "nodeOutputRows": 10,
                        "nodeOutputDataSize": "1514B",
                        "operatorInputRowsStats": [{
                            "nodeInputRows": 0.625,
                            "nodeInputRowsStdDev": 387.2983346207417
                        }]
                    },
                    "children": [{
                        "name": "RemoteSource",
                        "identifier": "[2]",
                        "outputs": ["date", "time", "location", "bytes", "requestip",
"method", "host",\
                          "uri", "status", "referrer", "os", "browser",
"browserversion"],
                        "details": "",
                        "distributedNodeStats": {
                            "nodeCpuTime": "0.00ns",
                            "nodeOutputRows": 10,
                            "nodeOutputDataSize": "1514B",
                            "operatorInputRowsStats": [{
                                "nodeInputRows": 0.625,
                                "nodeInputRowsStdDev": 387.2983346207417
                            }]
                        },
                        "children": []
                    }]
                }]
```

```
            }]
        }
    }, {
        "id": "2",

        "stageStats": {
            "totalCpuTime": "1.62s",
            "inputRows": "500 rows",
            "inputDataSize": "75564B",
            "stdDevInputRows": "0.00",
            "outputRows": "10 rows",
            "outputDataSize": "1514B"
        },
        "outputLayout": "date, time, location, bytes, requestip, method, host, uri,
status,\
           referrer, os, browser, browserversion",

        "logicalPlan": {
            "1": [{
                "name": "LimitPartial",
                "identifier": "[10]",
                "outputs": ["date", "time", "location", "bytes", "requestip", "method",
"host", "uri",\
                  "status", "referrer", "os", "browser", "browserversion"],
                "details": "",
                "distributedNodeStats": {
                    "nodeCpuTime": "0.00ns",
                    "nodeOutputRows": 10,
                    "nodeOutputDataSize": "1514B",
                    "operatorInputRowsStats": [{
                        "nodeInputRows": 83.33333333333333,
                        "nodeInputRowsStdDev": 223.60679774997897
                    }]
                },
                "children": [{
                    "name": "TableScan",
                    "identifier": "[awsdatacatalog:HiveTableHandle{schemaName=default,\
                       tableName=cloudfront_logs,
analyzePartitionValues=Optional.empty},\
                       grouped = false]",
                    "outputs": ["date", "time", "location", "bytes", "requestip",
"method", "host", "uri",\
                       "status", "referrer", "os", "browser", "browserversion"],
```

```
                    "details": "LAYOUT: default.cloudfront_logs\ndate :=
 date:date:0:REGULAR\nreferrer :=\
                       referrer: string:9:REGULAR\nos := os:string:10:REGULAR
\nmethod := method:string:5:\
                       REGULAR\nbytes := bytes:int:3:REGULAR\nbrowser :=
 browser:string:11:REGULAR\nhost :=\
                       host:string:6:REGULAR\nrequestip := requestip:string:4:REGULAR
\nlocation :=\
                       location:string:2:REGULAR\ntime := time:string:1: REGULAR
\nuri := uri:string:7:\
                       REGULAR\nbrowserversion := browserversion:string:12:REGULAR
\nstatus :=\
                       status:int:8:REGULAR\n",
                    "distributedNodeStats": {
                        "nodeCpuTime": "1.62s",
                        "nodeOutputRows": 500,
                        "nodeOutputDataSize": "75564B",
                        "operatorInputRowsStats": [{
                            "nodeInputRows": 83.33333333333333,
                            "nodeInputRowsStdDev": 223.60679774997897
                        }]
                    },
                    "children": []
                }]
            }]
        }
    }]
}
```

Risorse aggiuntive

Per ulteriori informazioni consulta le seguenti risorse.

- Capire i risultati dell'istruzione EXPLAIN di Athena
- Visualizzazione dei piani di esecuzione per query SQL
- Visualizzazione di statistiche e dettagli di esecuzione per le query completate
- Documentazione `EXPLAIN` di Trino
- Documentazione `EXPLAIN ANALYZE` di Trino
- Ottimizzazione delle prestazioni delle query federate con EXPLAIN e EXPLAIN ANALYZE in Amazon Athena nel Blog sui Big Data di AWS .

Capire i risultati dell'istruzione EXPLAIN di Athena

Questo argomento fornisce una breve guida ai termini operativi utilizzati nei risultati dell'istruzione `EXPLAIN` di Athena.

Tipi di output dell'istruzione EXPLAIN

Gli output dell'istruzione `EXPLAIN` possono essere uno dei due tipi:

- Piano logico: mostra il piano logico utilizzato dal motore SQL per eseguire un'istruzione. La sintassi per quest'opzione è `EXPLAIN` o `EXPLAIN (TYPE LOGICAL)`.
- Piano distribuito: mostra un piano di esecuzione in un ambiente distribuito. L'output mostra frammenti che stanno elaborando fasi. Ogni frammento del piano viene elaborato da uno o più nodi. I dati possono essere scambiati tra i nodi che elaborano i frammenti. La sintassi per quest'opzione è `EXPLAIN (TYPE DISTRIBUTED)`.

  Nell'output di un piano distribuito, i frammenti (fasi di elaborazione) sono indicati da `Fragment` *`number`* [*`fragment_type`*], dove *`number`* è un numero intero a base zero e *`fragment_type`* specifica come il frammento viene eseguito dai nodi. I tipi di frammento che forniscono informazioni dettagliate sul layout dello scambio di dati sono descritti nella tabella seguente.

  Tipi di frammento del piano distribuito

Tipo di frammento	Descrizione
`SINGLE`	Il frammento viene eseguito su un singolo nodo.
`HASH`	Il frammento viene eseguito su un numero fisso di nodi. I dati di input vengono distribuiti utilizzando una funzione hash.
`ROUND_ROBIN`	Il frammento viene eseguito su un numero fisso di nodi. I dati di input sono distribuiti in modo regolare.
`BROADCAST`	Il frammento viene eseguito su un numero fisso di nodi. I dati di input vengono trasmessi a tutti i nodi.
`SOURCE`	Il frammento viene eseguito su nodi in cui si accede alle divisioni dell'input.

Exchange

I termini relativi a Exchange descrivono il modo in cui i dati vengono scambiati tra i nodi di lavoro. I trasferimenti possono essere locali o remoti.

LocalExchange [*tipo_scambio*]

Trasferisce i dati localmente all'interno dei nodi di lavoro per le diverse fasi di una query. Il valore per *exchange_type* può essere uno dei tipi di scambio logico o distribuito come descritto più avanti in questa sezione.

RemoteExchange *[tipo_scambio]*

Trasferisce i dati tra i nodi di lavoro per le diverse fasi di una query. Il valore per *exchange_type* può essere uno dei tipi di scambio logico o distribuito come descritto più avanti in questa sezione.

Tipi di scambio logico

I seguenti tipi di scambio descrivono le operazioni intraprese durante la fase di scambio di un piano logico.

- **GATHER**: un singolo nodo di lavoro raccoglie l'output da tutti gli altri nodi di lavoro. Ad esempio, l'ultima fase di una query di selezione raccoglie i risultati da tutti i nodi e li scrive su Amazon S3.
- **REPARTITION**: invia i dati di riga a un dipendente specifico in base allo schema di partizionamento necessario per applicare all'operatore successivo.
- **REPLICATE**: copia i dati della riga per tutti i lavoratori.

Tipi di scambio distribuiti

I seguenti tipi di scambio indicano il layout dei dati quando vengono scambiati tra nodi in un piano distribuito.

- **HASH**: lo scambio distribuisce i dati a più destinazioni utilizzando una funzione hash.
- **SINGLE**: Lo scambio distribuisce i dati a una singola destinazione.

Scansione

I seguenti termini descrivono come vengono analizzati i dati durante una query.

TableScan

Analizza i dati di origine di una tabella da Amazon S3 o da un connettore Apache Hive e applica la potatura delle partizioni generata dal predicato del filtro.

ScanFilter

Esegue la scansione dei dati di origine di una tabella da Amazon S3 o da un connettore Hive e applica la cesura delle partizioni generata dal predicato filtro e da predicati filtro aggiuntivi non applicati tramite la cesura delle partizioni.

ScanFilterProject

Per prima cosa esegue la scansione dei dati di origine di una tabella da Amazon S3 o da un connettore Hive e applica la cesura delle partizioni generata dal predicato filtro e da predicati filtro aggiuntivi non applicati tramite la cesura delle partizioni. Quindi, modifica il layout di memoria dei dati di output in una nuova proiezione per migliorare le prestazioni delle fasi successive.

Join

Unisce i dati tra due tabelle. I join possono essere classificati in base al tipo di join e al tipo di distribuzione.

Tipi di join

I tipi di join definiscono il modo in cui si verifica l'operazione di join.

CrossJoin— Produce il prodotto cartesiano delle due tabelle unite.

InnerJoin— Seleziona i record con valori corrispondenti in entrambe le tabelle.

LeftJoin— Seleziona tutti i record dalla tabella di sinistra e i record corrispondenti dalla tabella di destra. Se non sussiste alcuna corrispondenza, il risultato sul lato destro è NULL.

RightJoin— Seleziona tutti i record dalla tabella di destra e i record corrispondenti dalla tabella di sinistra. Se non sussiste alcuna corrispondenza, il risultato sul lato sinistro è NULL.

FullJoin— Seleziona tutti i record in cui è presente una corrispondenza nei record della tabella sinistra o destra. La tabella unita contiene tutti i registri di entrambe le tabelle e riporta NULL per le corrispondenze mancanti su entrambi i lati.

> **Note**
>
> Per motivi di prestazioni, il motore di query può riscrivere una query di join in un tipo di join diverso per produrre gli stessi risultati. Ad esempio, una query inner join con predicato su una tabella può essere riscritta in `CrossJoin`. In questo modo il predicato viene spostato fino alla fase di scansione della tabella in modo che vengano scansionati meno dati.

Tipi di distribuzione join

I tipi di distribuzione definiscono il modo in cui i dati vengono scambiati tra i nodi di lavoro quando viene eseguita l'operazione di join.

Partizionata: sia la tabella sinistra che quella destra sono partizionate in hash-su tutti i nodi (worker). La distribuzione partizionata consuma meno memoria in ogni nodo. La distribuzione partizionata può essere molto più lenta dei join replicati. I join partizionati sono adatti quando si uniscono due tabelle di grandi dimensioni.

Replicata: una tabella è partizionata in hash su tutti i nodi di lavoro e l'altra tabella viene replicata in tutti i nodi di lavoro per eseguire l'operazione di join. La distribuzione replicata può essere molto più veloce dei join partizionati, ma consuma più memoria in ogni nodo di lavoro. Se la tabella replicata è troppo grande, il nodo di lavoro può riscontrare un out-of-memory errore. I join replicati sono adatti quando una delle tabelle unite è piccola.

PREPARE

Crea un'istruzione SQL con il nome `statement_name` da eseguire in un secondo momento. L'istruzione può includere parametri rappresentati da punti interrogativi. Per fornire valori per i parametri ed eseguire l'istruzione preparata, utilizza EXECUTE.

Riepilogo

```
PREPARE statement_name FROM statement
```

Nella tabella seguente vengono descritti questi parametri.

Parametro	Descrizione
`statement_name`	Il nome dell'istruzione da preparare. Il nome deve essere univoco all'interno del gruppo di lavoro.
`statement`	Una query `SELECT`, `CTAS` o `INSERT INTO`.

> **Note**
>
> Il numero massimo di istruzioni preparate in un gruppo di lavoro è 1.000.

Esempi

L'esempio seguente prepara una query di selezione senza parametri.

```
PREPARE my_select1 FROM
SELECT * FROM nation
```

L'esempio seguente prepara una query di selezione che include i parametri. I valori per `productid` e `quantity` verranno forniti dalla clausola `USING` di un'istruzione `EXECUTE`:

```
PREPARE my_select2 FROM
SELECT order FROM orders WHERE productid = ? and quantity < ?
```

L'esempio seguente prepara una query insert.

```
PREPARE my_insert FROM
INSERT INTO cities_usa (city, state)
SELECT city, state
FROM cities_world
WHERE country = ?
```

Risorse aggiuntive

Esecuzione di query con istruzioni preparate

EXECUTE

DEALLOCATE PREPARE

INSERT INTO

EXECUTE

Esegue un'istruzione preparata con il nome `statement_name`. I valori dei parametri per i punti interrogativi nell'istruzione preparata sono definiti nella clausola `USING` in un'elenco separato da virgole. Per creare un'istruzione preparata, utilizza PREPARE.

Riepilogo

```
EXECUTE statement_name [ USING parameter1[, parameter2, ... ] ]
```

Esempi

L'esempio seguente prepara ed esegue una query di selezione senza parametri.

```
PREPARE my_select1 FROM
SELECT name FROM nation
EXECUTE my_select1
```

L'esempio seguente prepara ed esegue una query di selezione con un parametro.

```
PREPARE my_select2 FROM
SELECT * FROM "my_database"."my_table" WHERE year = ?
EXECUTE my_select2 USING 2012
```

Ciò equivale a:

```
SELECT * FROM "my_database"."my_table" WHERE year = 2012
```

L'esempio seguente prepara ed esegue una query di selezione con due parametri.

```
PREPARE my_select3 FROM
SELECT order FROM orders WHERE productid = ? and quantity < ?
EXECUTE my_select3 USING 346078, 12
```

Risorse aggiuntive

Esecuzione di query con istruzioni preparate

PREPARE

INSERT INTO

DEALLOCATE PREPARE

Rimuove l'istruzione preparata con il nome specificato dalle istruzioni preparate nel gruppo di lavoro corrente.

Riepilogo

```
DEALLOCATE PREPARE statement_name
```

Esempi

L'esempio seguente rimuove l'istruzione preparata `my_select1` dal gruppo di lavoro corrente.

```
DEALLOCATE PREPARE my_select1
```

Risorse aggiuntive

Esecuzione di query con istruzioni preparate

PREPARE

UNLOAD

Scrive i risultati delle query da un'istruzione `SELECT` al formato di dati specificato. I formati supportati per `UNLOAD` includono Apache Parquet, ORC, Apache Avro e JSON. CSV è l'unico formato di output supportato dal comando `SELECT` Athena, ma è possibile utilizzare `UNLOAD` il comando, che supporta diversi formati di output, per racchiudere `SELECT` la query e riscriverne l'output in uno dei formati supportati. `UNLOAD`

Sebbene sia possibile utilizzare l'istruzione CTAS per generare dati in formati diversi da CSV, tali istruzioni richiedono anche la creazione di una tabella in Athena. L'istruzione `UNLOAD` è utile quando si desidera generare i risultati di una query `SELECT` in un formato non CSV ma non richiedono la tabella associata. Ad esempio, un'applicazione downstream potrebbe richiedere i risultati di una query `SELECT` in formato JSON, e Parquet o ORC potrebbe fornire un vantaggio in termini di prestazioni rispetto a CSV se si intende utilizzare i risultati della query `SELECT` per ulteriori analisi.

Considerazioni e limitazioni

Quando utilizzi l'istruzione `UNLOAD` in Athena, devi considerare quanto segue:

- Nessun ordine globale dei file – I risultati UNLOAD vengono scritti in più file in parallelo. Se la query SELECT nell'istruzione UNLOAD specifica un ordine, il contenuto di ogni file è in ordine ordinato, ma i file non sono ordinati l'uno rispetto all'altro.
- Dati orfani non eliminati — In caso di guasto, Athena non tenta di eliminare i dati orfani. Questo comportamento è lo stesso per CTAS e per le istruzioni INSERT INTO.
- Partizioni massime: Il numero massimo di partizioni che possono essere utilizzate con UNLOAD è 100.
- File manifest e metadati — Athena genera un file di metadati e un file manifesto di dati per ogni query UNLOAD. Il manifest tiene traccia dei file scritti dalla query. Entrambi i file vengono salvati nella posizione dei risultati della query Athena in Amazon S3. Per ulteriori informazioni, consulta Identificazione dei file di output delle query.
- Crittografia – UNLOAD vengono crittografati in base alla configurazione di crittografia utilizzata per Amazon S3. Per configurare la configurazione di crittografia per crittografare i UNLOAD risultati, puoi utilizzare l'API. EncryptionConfiguration
- Istruzioni preparate – UNLOAD può essere utilizzato con istruzioni preparate. Per informazioni sulle istruzioni preparate in Athena, consulta Utilizzo di query parametrizzate.
- Quote di servizio — UNLOAD utilizza le quote di query DML. Per informazioni sulle quote, consulta Service Quotas (Quote di Servizio).
- Proprietario previsto del bucket: l'impostazione relativa al proprietario previsto del bucket non si applica al percorso del bucket Amazon S3 di destinazione specificato nella query UNLOAD. L'impostazione attesa relativa al proprietario del bucket si applica solo al percorso di output di Amazon S3 specificato per i risultati delle query di Athena. Per ulteriori informazioni, consulta Specificare una posizione dei risultati delle query utilizzando la console Athena.

Sintassi

L'istruzione UNLOAD utilizza la sintassi seguente.

```
UNLOAD (SELECT col_name[, ...] FROM old_table)
TO 's3://DOC-EXAMPLE-BUCKET/my_folder/'
WITH ( property_name = 'expression' [, ...] )
```

Tranne quando si scrive su partizioni, la TO destinazione deve specificare una posizione in Amazon S3 priva di dati. Prima che la query UNLOAD scrive nella posizione specificata, verifica che la posizione del bucket sia vuota. Poiché UNLOAD non scrive i dati nella posizione specificata se la posizione contiene già dati, UNLOAD non sovrascrive i dati esistenti. Per riutilizzare una posizione

bucket come destinazione per UNLOAD, elimina i dati nella posizione del bucket e quindi esegui nuovamente la query.

Tieni presente che quando si UNLOAD scrive su partizioni, questo comportamento è diverso. Se esegui la stessa UNLOAD query più volte con la stessa SELECT istruzione, la stessa TO posizione e le stesse partizioni, ogni UNLOAD query scarica i dati in Amazon S3 nella posizione e nelle partizioni specificate.

Parametri

I valori possibili di *property_name* sono indicati di seguito.

formato = '***file_format***'

Obbligatorio. Specifica il formato di file dell'output. Valori possibili per *file_format* sono ORC, PARQUET, AVRO, JSON o TEXTFILE.

compressione = '***compression_format***'

Facoltativo. Questa opzione è specifica per i formati ORC e Parquet. Per ORC il valore predefinito è zlib e per Parquet il valore predefinito è gzip. Per informazioni sui formati di compressione supportati, consulta Supporto alla compressione Athena.

> **Note**
>
> Questa opzione non è valida per il formato AVRO. Athena utilizza gzip per i formati JSON e TEXTFILE.

compression_level = ***livello_compressione***

Facoltativo. Il livello di compressione da utilizzare per la compressione ZSTD. Questa proprietà si applica solo alla compressione ZSTD. Per ulteriori informazioni, consulta Utilizzo dei livelli di compressione ZSTD in Athena.

field_delimiter = '***delimiter***'

Facoltativo. Specifica un delimitatore di campo a carattere singolo per file in CSV, TSV e altri formati di testo. L'esempio seguente specifica la virgola come separatore decimale.

```
WITH (field_delimiter = ',')
```

Attualmente, i delimitatori di campo multicarattere non sono supportati. Se non si specifica un separatore di campo, viene usato il carattere ottale `\001` (^A).

partitioned_by = ARRAY[***col_name***[,…]]

Facoltativo. Un elenco matrice di colonne in base al quale l'output è partizionato.

> **Note**
>
> Nell'istruzione `SELECT`, assicurati che i nomi delle colonne partizionate siano elencati per ultimi nell'elenco delle colonne.

Esempi

L'esempio seguente scrive l'output di una query `SELECT` per la posizione Amazon S3 `s3://DOC-EXAMPLE-BUCKET/unload_test_1/` utilizzando il formato JSON.

```
UNLOAD (SELECT * FROM old_table)
TO 's3://DOC-EXAMPLE-BUCKET/unload_test_1/'
WITH (format = 'JSON')
```

L'esempio seguente scrive l'output di una query `SELECT` in formato Parquet utilizzando la compressione Snappy.

```
UNLOAD (SELECT * FROM old_table)
TO 's3://DOC-EXAMPLE-BUCKET/'
WITH (format = 'PARQUET',compression = 'SNAPPY')
```

Nell'esempio seguente vengono scritte quattro colonne in formato testo, con l'output partizionato dall'ultima colonna.

```
UNLOAD (SELECT name1, address1, comment1, key1 FROM table1)
TO 's3://DOC-EXAMPLE-BUCKET/ partitioned/'
WITH (format = 'TEXTFILE', partitioned_by = ARRAY['key1'])
```

L'esempio seguente scarica i risultati della query nella posizione specificata utilizzando il formato file Parquet, la compressione ZSTD e il livello di compressione ZSTD 4.

```
UNLOAD (SELECT * FROM old_table)
```

```
TO 's3://DOC-EXAMPLE-BUCKET/'
WITH (format = 'PARQUET', compression = 'ZSTD', compression_level = 4)
```

Risorse aggiuntive

- Semplifica le tue pipeline ETL e ML utilizzando la funzionalità Amazon Athena UNLOAD nel Blog sui Big Data di AWS .

Funzioni in Amazon Athena

Per ulteriori informazioni sulle funzioni tra le versioni del motore Athena, consulta Documentazione di riferimento della versione del motore Athena. Per un elenco dei fusi orari che possono essere utilizzati con l'operatore `AT TIME ZONE`, consulta Fusi orari supportati.

Versione 3 del motore Athena

Le funzioni nella versione 3 del motore Athena sono basate su Trino. Per informazioni su funzioni, operatori ed espressioni di Trino, consulta Funzioni e operatori e le seguenti sezioni specifiche della documentazione di Trino.

- Aggregazione
- Array
- Binary
- Bit per bit
- Color (Colore)
- Confronto
- Condizionale
- Conversione
- Data e ora
- Decimale
- Dati geospaziali
- HyperLogLog
- Indirizzo IP
- JSON
- Lambda

- Logica
- Machine learning
- Mappa
- Math (Matematica)
- Digest quantile
- Espressione regolare
- Sessione
- Set Digest
- Stringa
- Tabella
- Teradata
- T-Digest
- URL
- UUID
- Window

funzione invoker_principal ()

La `invoker_principal` funzione è esclusiva del motore Athena versione 3 e non si trova in Trino.

Restituisce un valore `VARCHAR` che contiene l'ARN del principale (ruolo IAM o Identity Center identity) che ha eseguito la query che chiama la funzione. Ad esempio, se l'invocatore della query utilizza le autorizzazioni di un ruolo IAM per eseguire la query, la funzione restituisce l'ARN del ruolo IAM. Il ruolo che esegue la query deve consentire l'azione. `LakeFormation:GetDataLakePrincipal`

Utilizzo

```
SELECT invoker_principal()
```

La tabella seguente mostra un esempio di risultato.

#	_col0
1	*arn:aws:iam:: 111122223333:role/admin*

Versione 2 del motore Athena

Le funzioni nella versione 2 del motore Athena sono basate su Presto 0.217. Per le funzioni geospaziali della versione 2 del motore Athena, consulta Funzioni geospaziali nella versione 2 del motore Athena.

> **Note**
>
> La documentazione specifica della versione per le funzioni di Presto 0.217 non è più disponibile. Per informazioni sulle funzioni, sugli operatori e sulle espressioni correnti di Presto, consulta la sezione Functions and operators (Funzioni e operatori) nella documentazione di Presto oppure visita i link delle sottocategorie in questa sezione.

- Operatori logici
- Operatori e funzioni di confronto
- Espressioni condizionali
- Funzioni di conversione
- Operatori e funzioni matematiche
- Funzioni bit per bit
- Operatori e funzioni decimali
- Operatori e funzioni di stringa
- Funzioni binarie
- Operatori e funzioni data e ora
- Funzioni di espressioni regolari
- Operatori e funzioni JSON
- Funzioni URL
- Funzioni di aggregazione
- Funzioni finestra
- Funzioni colore
- Operatori e funzioni della matrice
- Operatori e funzioni mappa
- Funzioni ed espressioni Lambda
- Funzioni Teradata

Fusi orari supportati

Puoi utilizzare l'operatore `AT TIME ZONE` in un'istruzione `SELECT timestamp` per specificare il fuso orario per la marca temporale restituita, come nell'esempio seguente:

```
SELECT timestamp '2012-10-31 01:00 UTC' AT TIME ZONE 'America/Los_Angeles' AS la_time;
```

Risultati

```
la_time

2012-10-30 18:00:00.000 America/Los_Angeles
```

L'elenco seguente contiene i fusi orari che possono essere utilizzati con l'operatore `AT TIME ZONE` in Athena. Per ulteriori funzioni ed esempi relativi al fuso orario, consulta Funzioni ed esempi del fuso orario.

```
Africa/Abidjan
Africa/Accra
Africa/Addis_Ababa
Africa/Algiers
Africa/Asmara
Africa/Asmera
Africa/Bamako
Africa/Bangui
Africa/Banjul
Africa/Bissau
Africa/Blantyre
Africa/Brazzaville
Africa/Bujumbura
Africa/Cairo
Africa/Casablanca
Africa/Ceuta
Africa/Conakry
Africa/Dakar
Africa/Dar_es_Salaam
Africa/Djibouti
Africa/Douala
Africa/El_Aaiun
Africa/Freetown
Africa/Gaborone
Africa/Harare
```

```
Africa/Johannesburg
Africa/Juba
Africa/Kampala
Africa/Khartoum
Africa/Kigali
Africa/Kinshasa
Africa/Lagos
Africa/Libreville
Africa/Lome
Africa/Luanda
Africa/Lubumbashi
Africa/Lusaka
Africa/Malabo
Africa/Maputo
Africa/Maseru
Africa/Mbabane
Africa/Mogadishu
Africa/Monrovia
Africa/Nairobi
Africa/Ndjamena
Africa/Niamey
Africa/Nouakchott
Africa/Ouagadougou
Africa/Porto-Novo
Africa/Sao_Tome
Africa/Timbuktu
Africa/Tripoli
Africa/Tunis
Africa/Windhoek
America/Adak
America/Anchorage
America/Anguilla
America/Antigua
America/Araguaina
America/Argentina/Buenos_Aires
America/Argentina/Catamarca
America/Argentina/ComodRivadavia
America/Argentina/Cordoba
America/Argentina/Jujuy
America/Argentina/La_Rioja
America/Argentina/Mendoza
America/Argentina/Rio_Gallegos
America/Argentina/Salta
America/Argentina/San_Juan
```

```
America/Argentina/San_Luis
America/Argentina/Tucuman
America/Argentina/Ushuaia
America/Aruba
America/Asuncion
America/Atikokan
America/Atka
America/Bahia
America/Bahia_Banderas
America/Barbados
America/Belem
America/Belize
America/Blanc-Sablon
America/Boa_Vista
America/Bogota
America/Boise
America/Buenos_Aires
America/Cambridge_Bay
America/Campo_Grande
America/Cancun
America/Caracas
America/Catamarca
America/Cayenne
America/Cayman
America/Chicago
America/Chihuahua
America/Coral_Harbour
America/Cordoba
America/Costa_Rica
America/Creston
America/Cuiaba
America/Curacao
America/Danmarkshavn
America/Dawson
America/Dawson_Creek
America/Denver
America/Detroit
America/Dominica
America/Edmonton
America/Eirunepe
America/El_Salvador
America/Ensenada
America/Fort_Nelson
America/Fort_Wayne
```

```
America/Fortaleza
America/Glace_Bay
America/Godthab
America/Goose_Bay
America/Grand_Turk
America/Grenada
America/Guadeloupe
America/Guatemala
America/Guayaquil
America/Guyana
America/Halifax
America/Havana
America/Hermosillo
America/Indiana/Indianapolis
America/Indiana/Knox
America/Indiana/Marengo
America/Indiana/Petersburg
America/Indiana/Tell_City
America/Indiana/Vevay
America/Indiana/Vincennes
America/Indiana/Winamac
America/Indianapolis
America/Inuvik
America/Iqaluit
America/Jamaica
America/Jujuy
America/Juneau
America/Kentucky/Louisville
America/Kentucky/Monticello
America/Knox_IN
America/Kralendijk
America/La_Paz
America/Lima
America/Los_Angeles
America/Louisville
America/Lower_Princes
America/Maceio
America/Managua
America/Manaus
America/Marigot
America/Martinique
America/Matamoros
America/Mazatlan
America/Mendoza
```

```
America/Menominee
America/Merida
America/Metlakatla
America/Mexico_City
America/Miquelon
America/Moncton
America/Monterrey
America/Montevideo
America/Montreal
America/Montserrat
America/Nassau
America/New_York
America/Nipigon
America/Nome
America/Noronha
America/North_Dakota/Beulah
America/North_Dakota/Center
America/North_Dakota/New_Salem
America/Ojinaga
America/Panama
America/Pangnirtung
America/Paramaribo
America/Phoenix
America/Port-au-Prince
America/Port_of_Spain
America/Porto_Acre
America/Porto_Velho
America/Puerto_Rico
America/Punta_Arenas
America/Rainy_River
America/Rankin_Inlet
America/Recife
America/Regina
America/Resolute
America/Rio_Branco
America/Rosario
America/Santa_Isabel
America/Santarem
America/Santiago
America/Santo_Domingo
America/Sao_Paulo
America/Scoresbysund
America/Shiprock
America/Sitka
```

```
America/St_Barthelemy
America/St_Johns
America/St_Kitts
America/St_Lucia
America/St_Thomas
America/St_Vincent
America/Swift_Current
America/Tegucigalpa
America/Thule
America/Thunder_Bay
America/Tijuana
America/Toronto
America/Tortola
America/Vancouver
America/Virgin
America/Whitehorse
America/Winnipeg
America/Yakutat
America/Yellowknife
Antarctica/Casey
Antarctica/Davis
Antarctica/DumontDUrville
Antarctica/Macquarie
Antarctica/Mawson
Antarctica/McMurdo
Antarctica/Palmer
Antarctica/Rothera
Antarctica/South_Pole
Antarctica/Syowa
Antarctica/Troll
Antarctica/Vostok
Arctic/Longyearbyen
Asia/Aden
Asia/Almaty
Asia/Amman
Asia/Anadyr
Asia/Aqtau
Asia/Aqtobe
Asia/Ashgabat
Asia/Ashkhabad
Asia/Atyrau
Asia/Baghdad
Asia/Bahrain
Asia/Baku
```

```
Asia/Bangkok
Asia/Barnaul
Asia/Beirut
Asia/Bishkek
Asia/Brunei
Asia/Calcutta
Asia/Chita
Asia/Choibalsan
Asia/Chongqing
Asia/Chungking
Asia/Colombo
Asia/Dacca
Asia/Damascus
Asia/Dhaka
Asia/Dili
Asia/Dubai
Asia/Dushanbe
Asia/Gaza
Asia/Harbin
Asia/Hebron
Asia/Ho_Chi_Minh
Asia/Hong_Kong
Asia/Hovd
Asia/Irkutsk
Asia/Istanbul
Asia/Jakarta
Asia/Jayapura
Asia/Jerusalem
Asia/Kabul
Asia/Kamchatka
Asia/Karachi
Asia/Kashgar
Asia/Kathmandu
Asia/Katmandu
Asia/Khandyga
Asia/Kolkata
Asia/Krasnoyarsk
Asia/Kuala_Lumpur
Asia/Kuching
Asia/Kuwait
Asia/Macao
Asia/Macau
Asia/Magadan
Asia/Makassar
```

```
Asia/Manila
Asia/Muscat
Asia/Nicosia
Asia/Novokuznetsk
Asia/Novosibirsk
Asia/Omsk
Asia/Oral
Asia/Phnom_Penh
Asia/Pontianak
Asia/Pyongyang
Asia/Qatar
Asia/Qyzylorda
Asia/Rangoon
Asia/Riyadh
Asia/Saigon
Asia/Sakhalin
Asia/Samarkand
Asia/Seoul
Asia/Shanghai
Asia/Singapore
Asia/Srednekolymsk
Asia/Taipei
Asia/Tashkent
Asia/Tbilisi
Asia/Tehran
Asia/Tel_Aviv
Asia/Thimbu
Asia/Thimphu
Asia/Tokyo
Asia/Tomsk
Asia/Ujung_Pandang
Asia/Ulaanbaatar
Asia/Ulan_Bator
Asia/Urumqi
Asia/Ust-Nera
Asia/Vientiane
Asia/Vladivostok
Asia/Yakutsk
Asia/Yangon
Asia/Yekaterinburg
Asia/Yerevan
Atlantic/Azores
Atlantic/Bermuda
Atlantic/Canary
```

```
Atlantic/Cape_Verde
Atlantic/Faeroe
Atlantic/Faroe
Atlantic/Jan_Mayen
Atlantic/Madeira
Atlantic/Reykjavik
Atlantic/South_Georgia
Atlantic/St_Helena
Atlantic/Stanley
Australia/ACT
Australia/Adelaide
Australia/Brisbane
Australia/Broken_Hill
Australia/Canberra
Australia/Currie
Australia/Darwin
Australia/Eucla
Australia/Hobart
Australia/LHI
Australia/Lindeman
Australia/Lord_Howe
Australia/Melbourne
Australia/NSW
Australia/North
Australia/Perth
Australia/Queensland
Australia/South
Australia/Sydney
Australia/Tasmania
Australia/Victoria
Australia/West
Australia/Yancowinna
Brazil/Acre
Brazil/DeNoronha
Brazil/East
Brazil/West
CET
CST6CDT
Canada/Atlantic
Canada/Central
Canada/Eastern
Canada/Mountain
Canada/Newfoundland
Canada/Pacific
```

```
Canada/Saskatchewan
Canada/Yukon
Chile/Continental
Chile/EasterIsland
Cuba
EET
EST5EDT
Egypt
Eire
Europe/Amsterdam
Europe/Andorra
Europe/Astrakhan
Europe/Athens
Europe/Belfast
Europe/Belgrade
Europe/Berlin
Europe/Bratislava
Europe/Brussels
Europe/Bucharest
Europe/Budapest
Europe/Busingen
Europe/Chisinau
Europe/Copenhagen
Europe/Dublin
Europe/Gibraltar
Europe/Guernsey
Europe/Helsinki
Europe/Isle_of_Man
Europe/Istanbul
Europe/Jersey
Europe/Kaliningrad
Europe/Kiev
Europe/Kirov
Europe/Lisbon
Europe/Ljubljana
Europe/London
Europe/Luxembourg
Europe/Madrid
Europe/Malta
Europe/Mariehamn
Europe/Minsk
Europe/Monaco
Europe/Moscow
Europe/Nicosia
```

```
Europe/Oslo
Europe/Paris
Europe/Podgorica
Europe/Prague
Europe/Riga
Europe/Rome
Europe/Samara
Europe/San_Marino
Europe/Sarajevo
Europe/Simferopol
Europe/Skopje
Europe/Sofia
Europe/Stockholm
Europe/Tallinn
Europe/Tirane
Europe/Tiraspol
Europe/Ulyanovsk
Europe/Uzhgorod
Europe/Vaduz
Europe/Vatican
Europe/Vienna
Europe/Vilnius
Europe/Volgograd
Europe/Warsaw
Europe/Zagreb
Europe/Zaporozhye
Europe/Zurich
GB
GB-Eire
Hongkong
Iceland
Indian/Antananarivo
Indian/Chagos
Indian/Christmas
Indian/Cocos
Indian/Comoro
Indian/Kerguelen
Indian/Mahe
Indian/Maldives
Indian/Mauritius
Indian/Mayotte
Indian/Reunion
Iran
Israel
```

```
Jamaica
Japan
Kwajalein
Libya
MET
MST7MDT
Mexico/BajaNorte
Mexico/BajaSur
Mexico/General
NZ
NZ-CHAT
Navajo
PRC
PST8PDT
Pacific/Apia
Pacific/Auckland
Pacific/Bougainville
Pacific/Chatham
Pacific/Chuuk
Pacific/Easter
Pacific/Efate
Pacific/Enderbury
Pacific/Fakaofo
Pacific/Fiji
Pacific/Funafuti
Pacific/Galapagos
Pacific/Gambier
Pacific/Guadalcanal
Pacific/Guam
Pacific/Honolulu
Pacific/Johnston
Pacific/Kiritimati
Pacific/Kosrae
Pacific/Kwajalein
Pacific/Majuro
Pacific/Marquesas
Pacific/Midway
Pacific/Nauru
Pacific/Niue
Pacific/Norfolk
Pacific/Noumea
Pacific/Pago_Pago
Pacific/Palau
Pacific/Pitcairn
```

```
Pacific/Pohnpei
Pacific/Ponape
Pacific/Port_Moresby
Pacific/Rarotonga
Pacific/Saipan
Pacific/Samoa
Pacific/Tahiti
Pacific/Tarawa
Pacific/Tongatapu
Pacific/Truk
Pacific/Wake
Pacific/Wallis
Pacific/Yap
Poland
Portugal
ROK
Singapore
Turkey
US/Alaska
US/Aleutian
US/Arizona
US/Central
US/East-Indiana
US/Eastern
US/Hawaii
US/Indiana-Starke
US/Michigan
US/Mountain
US/Pacific
US/Pacific-New
US/Samoa
W-SU
WET
```

Funzioni ed esempi del fuso orario

Qui di seguito è possibile trovare ulteriori funzioni ed esempi relativi al fuso orario.

- at_timezone (***timestamp***,***zone (zona)***): Restituisce il valore di *timestamp* nell'ora locale corrispondente per *zone (zona)*.

  Esempio

```
SELECT at_timezone(timestamp '2021-08-22 00:00 UTC', 'Canada/Newfoundland')
```

Risultato

```
2021-08-21 21:30:00.000 Canada/Newfoundland
```

- timezone_hour(***timestamp***): restituisce l'ora dell'offset del fuso orario dal timestamp come `bigint`.

  Esempio

  ```
  SELECT timezone_hour(timestamp '2021-08-22 04:00 UTC' AT TIME ZONE 'Canada/
  Newfoundland')
  ```

  Risultato

  ```
  -2
  ```

- timezone_minute(***timestamp***): Restituisce l'ora dell'offset del fuso orario dal *timestamp* come `bigint`.

  Esempio

  ```
  SELECT timezone_minute(timestamp '2021-08-22 04:00 UTC' AT TIME ZONE 'Canada/
  Newfoundland')
  ```

  Risultato

  ```
  -30
  ```

- with_timezone (***timestamp***,***zona***): Restituisce un timestamp con fuso orario dai valori *timestamp* e *zone (zona)* specificati.

  Esempio

  ```
  SELECT with_timezone(timestamp '2021-08-22 04:00', 'Canada/Newfoundland')
  ```

  Risultato

```
2021-08-22 04:00:00.000 Canada/Newfoundland
```

Istruzioni DDL

Utilizza le seguenti istruzioni DDL direttamente in Athena.

Il motore di query Athena è basato in parte su HiveQL DDL.

Athena non supporta tutte le istruzioni DDL e ci sono alcune differenze tra HiveQL DDL e Athena DDL. Per ulteriori informazioni, consulta gli argomenti di riferimento in questa sezione e in DDL non supportato.

Argomenti

- MSCK REPAIR TABLE
- SHOW COLUMNS
- SHOW CREATE TABLE
- SHOW CREATE VIEW
- SHOW DATABASES
- SHOW PARTITIONS
- SHOW TABLES
- SHOW TBLPROPERTIES
- SHOW VIEWS

DDL non supportato

Le seguenti istruzioni DDL non sono supportate da Athena:

- ALTER INDEX
- ALTER TABLE *table_name* ARCHIVE PARTITION
- ALTER TABLE *table_name* CLUSTERED BY
- ALTER TABLE *table_name* EXCHANGE PARTITION
- ALTER TABLE *table_name* NOT CLUSTERED
- ALTER TABLE *table_name* NOT SKEWED
- ALTER TABLE *table_name* NOT SORTED
- ALTER TABLE *table_name* NOT STORED AS DIRECTORIES
- ALTER TABLE *table_name* partitionSpec CHANGE COLUMNS
- ALTER TABLE *table_name* partitionSpec COMPACT
- ALTER TABLE *table_name* partitionSpec CONCATENATE
- ALTER TABLE *table_name* partitionSpec SET FILEFORMAT
- ALTER TABLE *table_name* SET SERDEPROPERTIES
- ALTER TABLE *table_name* SET SKEWED LOCATION
- ALTER TABLE *table_name* SKEWED BY
- ALTER TABLE *table_name* TOUCH
- ALTER TABLE *table_name* UNARCHIVE PARTITION

- COMMIT
- CREATE INDEX
- CREATE ROLE
- CREATE TABLE *table_name* LIKE *existing_table_name*
- CREATE TEMPORARY MACRO
- DELETE FROM
- DESCRIBE DATABASE
- DFS
- DROP INDEX
- DROP ROLE
- DROP TEMPORARY MACRO
- EXPORT TABLE
- GRANT ROLE
- IMPORT TABLE
- LOCK DATABASE
- LOCK TABLE
- REVOKE ROLE
- ROLLBACK
- SHOW COMPACTIONS
- SHOW CURRENT ROLES
- SHOW GRANT
- SHOW INDEXES
- SHOW LOCKS
- SHOW PRINCIPALS
- SHOW ROLE GRANT
- SHOW ROLES
- SHOW STATS
- SHOW TRANSACTIONS
- START TRANSACTION

- UNLOCK DATABASE
- UNLOCK TABLE

ALTER DATABASE SET DBPROPERTIES

Crea una o più proprietà per un database. I codici `DATABASE` e `SCHEMA` sono utilizzabili in modo intercambiabile, poiché significano la stessa cosa.

Riepilogo

```
ALTER {DATABASE|SCHEMA} database_name
  SET DBPROPERTIES ('property_name'='property_value' [, ...] )
```

Parametri

SET DBPROPERTIES ('property_name'='property_value' [, ...]

Specifica una o più proprietà per il database denominato `property_name` e stabilisce il valore per ciascuna proprietà rispettivamente come `property_value`. Se `property_name` è già presente, il vecchio valore viene sovrascritto con `property_value`.

Esempi

```
ALTER DATABASE jd_datasets
  SET DBPROPERTIES ('creator'='John Doe', 'department'='applied mathematics');
```

```
ALTER SCHEMA jd_datasets
  SET DBPROPERTIES ('creator'='Jane Doe');
```

ALTER TABLE ADD COLUMNS

Aggiunge una o più colonne a una tabella esistente. Quando si utilizza la sintassi `PARTITION` facoltativa, aggiorna i metadati della partizione.

Riepilogo

```
ALTER TABLE table_name
```

```
  [PARTITION
   (partition_col1_name = partition_col1_value
   [,partition_col2_name = partition_col2_value][,...])]
  ADD COLUMNS (col_name data_type)
```

Parametri

PARTITION (partition_col_name = partition_col_value [,...])

Crea una partizione con le combinazioni nome/valore colonna specificate dall'utente. Racchiudere `partition_col_value` tra virgolette solo se il tipo di dati della colonna è una stringa.

ADD COLUMNS (col_name data_type [,col_name data_type,…])

Aggiunge colonne dopo le colonne esistenti, ma prima delle colonne delle partizioni.

Esempi

```
ALTER TABLE events ADD COLUMNS (eventowner string)
```

```
ALTER TABLE events PARTITION (awsregion='us-west-2') ADD COLUMNS (event string)
```

```
ALTER TABLE events PARTITION (awsregion='us-west-2') ADD COLUMNS (eventdescription
 string)
```

Note

- Per visualizzare una nuova colonna della tabella nel pannello di navigazione dell'editor di query Athena dopo l'esecuzione di `ALTER TABLE ADD COLUMNS`, aggiornare manualmente l'elenco di tabelle nell'editor e quindi espandere nuovamente la tabella.
- `ALTER TABLE ADD COLUMNS` non funziona per le colonne con il tipo di dati `date`. Per risolvere questo problema, utilizzare invece il tipo di dati `timestamp`.

ALTER TABLE ADD PARTITION

Crea una o più colonne di partizione per la tabella. Ogni partizione è composta da una o più combinazioni diverse di nome/valore. Viene creata una directory di dati separata per ciascuna combinazione specificata; in alcune circostanze ciò può migliorare le prestazioni delle query. Le

colonne partizionate non sono presenti all'interno della tabella di dati stessa, perciò se si utilizza un nome di colonna uguale a un nome di una colonna nella tabella, si riceverà un messaggio di errore. Per ulteriori informazioni, consulta Partizionamento dei dati in Athena.

In Athena, una tabella e le relative partizioni devono utilizzare gli stessi formati di dati, tuttavia i loro schemi possono differire. Per ulteriori informazioni, consulta Aggiornamenti in tabelle con partizioni.

Per informazioni sulle autorizzazioni a livello di risorsa richieste nelle policy IAM (incluse `glue:CreatePartition`), consulta Autorizzazioni API AWS Glue : documentazione di riferimento su operazioni e risorse e Accesso granulare a database e tabelle in AWS Glue Data Catalog. Per informazioni sulla risoluzione dei problemi relativi alle autorizzazioni quando si utilizza Athena, consulta la sezione Autorizzazioni dell'argomento Risoluzione dei problemi in Athena.

Riepilogo

```
ALTER TABLE table_name ADD [IF NOT EXISTS]
  PARTITION
  (partition_col1_name = partition_col1_value
  [,partition_col2_name = partition_col2_value]
  [,...])
  [LOCATION 'location1']
  [PARTITION
  (partition_colA_name = partition_colA_value
  [,partition_colB_name = partition_colB_value
  [,...])]
  [LOCATION 'location2']
  [,...]
```

Parametri

Quando si aggiunge una partizione, è necessario specificare una o più coppie nome/valore colonna per la partizione e il percorso Amazon S3 in cui risiedono i file di dati per tale partizione.

[IF NOT EXISTS]

Provoca l'errore da sopprimere se una partizione con la stessa definizione è già esistente.

PARTITION (partition_col_name = partition_col_value [,...])

Crea una partizione con le combinazioni nome/valore colonna specificate dall'utente. Racchiudere `partition_col_value` in caratteri di stringa solo se il tipo di dati della colonna è una stringa.

[LOCATION 'location']

Specifica la directory in cui memorizzare la partizione definita dalla dichiarazione precedente. La clausola `LOCATION` è facoltativa quando i dati utilizzano il partizionamento in modalità Hive (`pk1=v1/pk2=v2/pk3=v3`). Con il partizionamento in modalità Hive, l'URI completo di Amazon S3 è creato automaticamente dalla posizione della tabella, dai nomi delle chiavi di partizione e dai valori delle chiavi di partizione. Per ulteriori informazioni, consulta Partizionamento dei dati in Athena.

Considerazioni

Amazon Athena non impone un limite specifico al numero di partizioni che è possibile aggiungere in una singola istruzione DDL. `ALTER TABLE ADD PARTITION` Tuttavia, se devi aggiungere un numero significativo di partizioni, valuta la possibilità di suddividere l'operazione in batch più piccoli per evitare potenziali problemi di prestazioni. L'esempio seguente utilizza comandi successivi per aggiungere partizioni singolarmente e utilizza comandi `IF NOT EXISTS` per evitare l'aggiunta di duplicati.

```
ALTER TABLE table_name ADD IF NOT EXISTS PARTITION (ds='2023-01-01')
ALTER TABLE table_name ADD IF NOT EXISTS PARTITION (ds='2023-01-02')
ALTER TABLE table_name ADD IF NOT EXISTS PARTITION (ds='2023-01-03')
```

Quando lavori con le partizioni in Athena, tieni presente anche i seguenti punti:

- Sebbene Athena supporti l'interrogazione di AWS Glue tabelle con 10 milioni di partizioni, Athena non è in grado di leggere più di 1 milione di partizioni in una singola scansione.
- Per ottimizzare le query e ridurre il numero di partizioni analizzate, prendete in considerazione strategie come l'eliminazione delle partizioni o l'utilizzo degli indici di partizione.
- Se non si utilizza AWS Glue Data Catalog, il numero massimo di partizioni per tabella è 20.000. È possibile richiedere un aumento della quota.

Per ulteriori considerazioni sull'utilizzo delle partizioni in Athena, vedere. Partizionamento dei dati in Athena

Esempi

L'esempio seguente aggiunge una singola partizione a una tabella di dati partizionati in modalità Hive.

```
ALTER TABLE orders ADD
  PARTITION (dt = '2016-05-14', country = 'IN');
```

L'esempio seguente aggiunge più partizioni a una tabella per dati partizionati in stile Hive.

```
ALTER TABLE orders ADD
  PARTITION (dt = '2016-05-31', country = 'IN')
  PARTITION (dt = '2016-06-01', country = 'IN');
```

Quando la tabella non è per dati partizionati in stile Hive, la `LOCATION` clausola è obbligatoria e deve essere l'URI completo di Amazon S3 per il prefisso che contiene i dati della partizione.

```
ALTER TABLE orders ADD
  PARTITION (dt = '2016-05-31', country = 'IN') LOCATION 's3://DOC-EXAMPLE-BUCKET/path/
to/INDIA_31_May_2016/'
  PARTITION (dt = '2016-06-01', country = 'IN') LOCATION 's3://DOC-EXAMPLE-BUCKET/path/
to/INDIA_01_June_2016/';
```

Per ignorare gli errori quando la partizione già esiste, utilizza la clausola `IF NOT EXISTS`, come nell'esempio seguente.

```
ALTER TABLE orders ADD IF NOT EXISTS
  PARTITION (dt = '2016-05-14', country = 'IN');
```

File **`_$folder$`** di zero byte

Se esegui un'istruzione `ALTER TABLE ADD PARTITION` e specifichi erroneamente una partizione già esistente e una posizione Amazon S3 errata, vengono creati in Amazon S3 i file segnaposto di zero byte del formato *`partition_value`*`_$folder$`. È necessario rimuovere questi file manualmente.

Per evitare che ciò accada, utilizza la sintassi `ADD IF NOT EXISTS` nella dichiarazione `ALTER TABLE ADD PARTITION`, come nell'esempio seguente.

```
ALTER TABLE table_name ADD IF NOT EXISTS PARTITION […]
```

ALTER TABLE DROP PARTITION

Rimuove una o più partizioni specificate per la tabella denominata.

Riepilogo

```
ALTER TABLE table_name DROP [IF EXISTS] PARTITION (partition_spec) [, PARTITION
 (partition_spec)]
```

Parametri

[IF EXISTS]

Sopprime il messaggio di errore se la partizione specificata non esiste.

PARTITION (partition_spec)

Ogni `partition_spec` specifica una combinazione nome/valore di colonna nel modulo `partition_col_name = partition_col_value [,...]`.

Esempi

```
ALTER TABLE orders
DROP PARTITION (dt = '2014-05-14', country = 'IN');
```

```
ALTER TABLE orders
DROP PARTITION (dt = '2014-05-14', country = 'IN'), PARTITION (dt = '2014-05-15',
 country = 'IN');
```

Note

L'istruzione `ALTER TABLE DROP PARTITION` non fornisce una singola sintassi per eliminare tutte le partizioni contemporaneamente o supporta i criteri di filtraggio per specificare un intervallo di partizioni da eliminare.

Come soluzione alternativa, puoi utilizzare l' AWS Glue API GetPartitionse BatchDeletePartitionle azioni negli script. L'operazione `GetPartitions` supporta espressioni di filtro complesse come quelle presenti in un'espressione `WHERE` SQL. Dopo aver utilizzato `GetPartitions` per creare un elenco filtrato di partizioni da eliminare, è possibile utilizzare l'operazione `BatchDeletePartition` per eliminare le partizioni in batch di 25.

> **Important**
>
> A causa di un problema noto, quando viene specificata una partizione non valida per la dichiarazione `ALTER TABLE DROP PARTITION`, tutte le partizioni della tabella vengono

eliminate in AWS Glue. Ad esempio, la seguente dichiarazione eliminerà tutte le partizioni della tabella *my_table* anche se la partizione specificata non esiste. Come soluzione alternativa, assicurati di inserire correttamente le informazioni sulla partizione prima di eseguire la dichiarazione `ALTER TABLE DROP PARTITION`.

```
ALTER TABLE my_table DROP IF EXISTS PARTITION(zzz='');
```

ALTER TABLE RENAME PARTITION

Rinomina un valore di partizione.

> **Note**
>
> ALTER TABLE RENAME PARTITIONnon rinomina le colonne delle partizioni. Per modificare il nome di una colonna di partizione, puoi usare la console. AWS Glue Per ulteriori informazioni, consultare Rinominare una colonna di partizione in AWS Glue riportata di seguito in questo documento.

Riepilogo

Per la tabella denominata`table_name`, rinomina il valore di partizione specificato da con il valore specificato `partition_spec` da. `new_partition_spec`

```
ALTER TABLE table_name PARTITION (partition_spec) RENAME TO PARTITION
 (new_partition_spec)
```

Parametri

PARTITION (partition_spec)

Ogni `partition_spec` specifica una combinazione nome/valore di colonna nel modulo `partition_col_name = partition_col_value [,...]`.

Esempi

```
ALTER TABLE orders
```

```
PARTITION (dt = '2014-05-14', country = 'IN') RENAME TO PARTITION (dt = '2014-05-15',
 country = 'IN');
```

Rinominare una colonna di partizione in AWS Glue

Utilizzare la procedura seguente per rinominare i nomi delle colonne di partizione nella console. AWS Glue

Per rinominare una colonna di partizione della tabella nella console AWS Glue

1. Accedere AWS Management Console e aprire la AWS Glue console all'indirizzo https://console.aws.amazon.com/glue/.
2. Nel pannello di navigazione, seleziona Tabelle.
3. Nella pagina Tabelle, utilizza la casella di ricerca Filtra tabelle per trovare la tabella che desideri modificare.
4. Nella colonna Nome, scegli il link della tabella che desideri modificare.
5. Nella pagina dei dettagli della tabella, nella sezione Schema, esegui una delle seguenti operazioni:
   - Per modificare il nome in formato JSON, scegli Modifica schema come JSON.
   - Per modificare direttamente il nome, scegli Modifica schema. Questa procedura sceglie Modifica schema.
6. Seleziona la casella di controllo relativa alla colonna partizionata che desideri rinominare, quindi scegli Modifica.
7. Nella finestra di dialogo Modifica immissione dello schema, in Nome, inserisci il nuovo nome per la colonna della partizione.
8. Scegliete Salva come nuova versione della tabella. Questa azione aggiorna il nome della colonna della partizione e conserva la cronologia dell'evoluzione dello schema senza creare una copia fisica separata dei dati.
9. Per confrontare le versioni della tabella, nella pagina dei dettagli della tabella, scegli Azioni, quindi scegli Confronta versioni.

Risorse aggiuntive

Per ulteriori informazioni sulle partizioni, consulta Partizionamento dei dati in Athena.

ALTER TABLE REPLACE COLUMNS

Rimuove tutte le colonne esistenti da una tabella creata con LazySimpleSerDe e le sostituisce con il set di colonne specificato. Quando si utilizza la sintassi `PARTITION` facoltativa, aggiorna i metadati della partizione. È possibile utilizzare anche `ALTER TABLE REPLACE COLUMNS` per eliminare le colonne specificando solo le colonne che si desidera mantenere.

Riepilogo

```
ALTER TABLE table_name
  [PARTITION
   (partition_col1_name = partition_col1_value
   [,partition_col2_name = partition_col2_value][,...])]
  REPLACE COLUMNS (col_name data_type [, col_name data_type, ...])
```

Parametri

PARTITION (partition_col_name = partition_col_value [,...])

Specifica una partizione con le combinazioni nome/valore colonna specificate dall'utente. Racchiudere `partition_col_value` tra virgolette solo se il tipo di dati della colonna è una stringa.

REPLACE COLUMNS (col_name data_type [,col_name data_type,…])

Sostituisce le colonne esistenti con i nomi delle colonne e i tipi di dati specificati.

Note

- Per visualizzare la modifica nelle colonne della tabella nel pannello di navigazione dell'editor di query Athena dopo l'esecuzione di `ALTER TABLE REPLACE COLUMNS`, potrebbe essere necessario aggiornare manualmente l'elenco di tabelle nell'editor e quindi espandere nuovamente la tabella.
- `ALTER TABLE REPLACE COLUMNS` non funziona per le colonne con il tipo di dati `date`. Per risolvere questo problema, utilizzare invece il tipo di dati `timestamp` nella tabella.
- Anche se si sta sostituendo una sola colonna, la sintassi deve essere `ALTER TABLE` *`table-name`* `REPLACE COLUMNS`, con colonne al plurale. È necessario specificare non solo la colonna che si desidera sostituire, ma anche le colonne che si desidera conservare. In caso contrario, le colonne non specificate verranno eliminate. Questa sintassi e questo comportamento derivano da

Apache Hive DDL. Come riferimento, consulta Aggiungi/sostituisci colonne nella documentazione di Apache.

Esempio

Nell'esempio seguente, la tabellanames_cities, creata utilizzando LazySimpleSerDe, ha tre colonne denominate col1col2, ecol3. Tutte le colonne sono di tipo string. Per visualizzare le colonne nella tabella, il comando seguente utilizza l'istruzione SHOW COLUMNS.

```
SHOW COLUMNS IN names_cities
```

Risultato della query:

```
col1
col2
col3
```

Il comando ALTER TABLE REPLACE COLUMNS di seguito riportato sostituisce i nomi delle colonne con first_name, last_name ecity. I dati di origine sottostanti non sono interessati.

```
ALTER TABLE names_cities
REPLACE COLUMNS (first_name string, last_name string, city string)
```

Per testare il risultato, SHOW COLUMNS viene eseguito nuovamente.

```
SHOW COLUMNS IN names_cities
```

Risultato della query:

```
first_name
last_name
city
```

Un altro modo per mostrare i nuovi nomi di colonna è quello di visualizzare in anteprima la tabella nell'editor di query di Athena o eseguire la propria query SELECT.

ALTER TABLE SET LOCATION

Modifica il percorso per la tabella denominata `table_name` e, facoltativamente, una partizione con `partition_spec`.

Riepilogo

```
ALTER TABLE table_name [ PARTITION (partition_spec) ] SET LOCATION 'new location'
```

Parametri

PARTITION (partition_spec)

Specifica la partizione con i parametri `partition_spec` di cui si desidera modificare il percorso. L'elemento `partition_spec` specifica una combinazione nome/valore di colonna nel modulo `partition_col_name = partition_col_value`.

SET LOCATION "nuovo percorso"

Specifica il nuovo percorso, che deve essere interno ad Amazon S3. Per ulteriori informazioni sulla sintassi, consula Posizione della tabella in Amazon S3.

Esempi

```
ALTER TABLE customers PARTITION (zip='98040', state='WA') SET LOCATION 's3://DOC-
EXAMPLE-BUCKET/custdata/';
```

ALTER TABLE SET TBLPROPERTIES

Aggiunge proprietà di metadati personalizzate o predefinite a una tabella e imposta i valori assegnati. Per visualizzare le proprietà in una tabella, utilizzare il comando SHOW TBLPROPERTIES.

Le tabelle gestite Apache Hive non sono supportate, quindi impostare `'EXTERNAL'='FALSE'` non determina alcun effetto.

Riepilogo

```
ALTER TABLE table_name SET TBLPROPERTIES ('property_name' = 'property_value' [ , ... ])
```

Parametri

SET TBLPROPERTIES ('property_name' = 'property_value' [, ...])

Specifica le proprietà dei metadati da aggiungere come `property_name` e il valore per ognuna di esse come `property value`. Se `property_name` esiste già, il suo valore è impostato sul nuovo `property_value`.

Le seguenti proprietà predefinite della tabella hanno usi speciali.

Proprietà predefinite	Descrizione
`classification`	Indica il tipo di dati per AWS Glue. I valori possibili sono `csv`, `parquet`, `orc`, `avro` o `json`. Le tabelle create per Athena nella CloudTrail console vengono aggiunte `cloudtrail` come valore per la `classification` proprietà. Per ulteriori informazioni, consulta la sezione TBLPROPERTIES di CREATE TABLE.
`has_encrypted_data`	Indica se il set di dati specificato da `LOCATION` è crittografato. Per ulteriori informazioni, consulta la sezione TBLPROPERTIES di CREATE TABLE e Creazione di tabelle basate su set di dati crittografati in Amazon S3.
`orc.compress`	Specifica un formato di compressione per i dati in formato ORC. Per ulteriori informazioni, consulta ORCO SerDe.
`parquet.compression`	Specifica un formato di compressione per i dati in formato Parquet. Per ulteriori informazioni, consulta Parquet SerDe.
`write.compression`	Specifica un formato di compressione per i dati in formato file di testo o JSON. Per i formati Parquet e ORC, utilizzare rispettivamente le proprietà `parquet.compression` e `orc.compress` .
`compression_level`	Specifica un livello di compressione da utilizzare. Questa proprietà si applica solo alla compressione ZSTD. I valori possibili sono compresi tra 1 e 22. Il valore predefinito è 3. Per ulteriori informazioni, consulta Utilizzo dei livelli di compressione ZSTD in Athena.

Proprietà predefinite	Descrizione
`projection.*`	Proprietà personalizzate usate nella proiezione della partizione che consentono ad Athena di sapere quali modelli di partizione aspettarsi quando viene eseguita una query sulla tabella. Per ulteriori informazioni, consulta Proiezione delle partizioni con Amazon Athena.
`skip.header.line.count`	Ignora le intestazioni nei dati quando si definisce una tabella. Per ulteriori informazioni, consulta Ignorare intestazioni.
`storage.location.template`	Specifica un modello di percorso Amazon S3 personalizzato per le partizioni proiettate. Per ulteriori informazioni, consulta Impostazione della proiezione delle partizioni.

Esempi

Nell'esempio seguente viene aggiunto un commento alle proprietà della tabella.

```
ALTER TABLE orders
SET TBLPROPERTIES ('notes'="Please don't drop this table.");
```

L'esempio seguente modifica la tabella `existing_table` per utilizzare il formato file Parquet con compressione ZSTD e livello di compressione ZSTD 4.

```
ALTER TABLE existing_table
SET TBLPROPERTIES ('parquet.compression' = 'ZSTD', 'compression_level' = 4)
```

CREATE DATABASE

Crea un database. L'uso di `DATABASE` e `SCHEMA` è intercambiabile. Significano la stessa cosa.

> **Note**
>
> Per vedere un esempio di creazione di un database o di una tabella e di esecuzione di una query `SELECT` sulla tabella di Athena, consulta Nozioni di base.

Riepilogo

```
CREATE {DATABASE|SCHEMA} [IF NOT EXISTS] database_name
  [COMMENT 'database_comment']
  [LOCATION 'S3_loc']
  [WITH DBPROPERTIES ('property_name' = 'property_value') [, ...]]
```

Parametri

[IF NOT EXISTS]

Determina la rimozione dell'errore se è già esistente un database denominato `database_name`.

[COMMENT database_comment]

Stabilisce il valore dei metadati per la proprietà metadati integrata denominata `comment` e il valore fornito per `database_comment`. In AWS Glue, i `COMMENT` contenuti vengono scritti nel `Description` campo delle proprietà del database.

[LOCATION S3_loc]

Specifica la posizione in cui i file e i metastore di database saranno presenti come `S3_loc`. È necessario che si tratti di una posizione Amazon S3.

[WITH DBPROPERTIES ('property_name' = 'property_value') [, ...]]

Consente di specificare proprietà personalizzate per i metadati relativi alla definizione del database.

Esempi

```
CREATE DATABASE clickstreams;
```

```
CREATE DATABASE IF NOT EXISTS clickstreams
  COMMENT 'Site Foo clickstream data aggregates'
  LOCATION 's3://DOC-EXAMPLE-BUCKET/clickstreams/'
  WITH DBPROPERTIES ('creator'='Jane D.', 'Dept.'='Marketing analytics');
```

Visualizzazione delle proprietà del database

Per visualizzare le proprietà del database per un database creato AWSDataCatalog utilizzandoCREATE DATABASE, è possibile utilizzare il AWS CLI comando `aws glue get-database`, come nell'esempio seguente:

```
aws glue get-database --name <your-database-name>
```

Nell'output JSON, il risultato è simile a quello nell'esempio seguente.

```
{
    "Database": {
        "Name": "<your-database-name>",
        "Description": "<your-database-comment>",
        "LocationUri": "s3://DOC-EXAMPLE-BUCKET",
        "Parameters": {
            "<your-database-property-name>": "<your-database-property-value>"
        },
        "CreateTime": 1603383451.0,
        "CreateTableDefaultPermissions": [
            {
                "Principal": {
                    "DataLakePrincipalIdentifier": "IAM_ALLOWED_PRINCIPALS"
                },
                "Permissions": [
                    "ALL"
                ]
            }
        ]
    }
}
```

Per ulteriori informazioni su AWS CLI, consulta la Guida AWS Command Line Interface per l'utente.

CREATE TABLE

Crea una tabella con il nome e i parametri specificati dall'utente.

> **Note**
>
> Questa pagina contiene informazioni di riferimento riepilogative. Per ulteriori informazioni sulla creazione delle tabelle in Athena e un esempio di istruzione CREATE TABLE, consulta la

sezione Creazione di tabelle in Athena. Per vedere esempio di creazione di un database o di una tabella e di esecuzione di una query SELECT sulla tabella di Athena, consulta Nozioni di base.

Riepilogo

```
CREATE EXTERNAL TABLE [IF NOT EXISTS]
 [db_name.]table_name [(col_name data_type [COMMENT col_comment] [, ...] )]
 [COMMENT table_comment]
 [PARTITIONED BY (col_name data_type [COMMENT col_comment], ...)]
 [CLUSTERED BY (col_name, col_name, ...) INTO num_buckets BUCKETS]
 [ROW FORMAT row_format]
 [STORED AS file_format]
 [WITH SERDEPROPERTIES (...)]
 [LOCATION 's3://DOC-EXAMPLE-BUCKET/[folder]/']
 [TBLPROPERTIES ( ['has_encrypted_data'='true | false',]
 ['classification'='aws_glue_classification',] property_name=property_value [, ...] ) ]
```

Parametri

EXTERNAL

Specifica che la tabella si basa su un file di dati sottostante esistente in Amazon S3 nella posizione specificata dall'utente in LOCATION. Tranne che durante la creazione di tabelle Iceberg, utilizza sempre la parola chiave EXTERNAL. Se utilizzi CREATE TABLE senza la parola chiave EXTERNAL per le tabelle non Iceberg, Athena genera un errore. Quando crei una tabella esterna, i dati a cui fai riferimento devono rispettare il formato predefinito o il formato specificato con le clausole ROW FORMAT, STORED AS e WITH SERDEPROPERTIES.

[IF NOT EXISTS]

Questo parametro verifica se esiste già una tabella con lo stesso nome. In caso affermativo, il parametro restituisce TRUE e Amazon Athena annulla l'azione CREATE TABLE. Poiché l'annullamento avviene prima che Athena richiami il catalogo dati, non emette alcun evento. AWS CloudTrail

[db_name.]table_name

Specifica un nome per la tabella da creare. Il parametro facoltativo db_name specifica il database in cui è presente la tabella. Se omesso, sarà utilizzato il database corrente. Se il nome della

tabella include numeri, racchiudi `table_name` tra virgolette, ad esempio `"table123"`. Se `table_name` inizia con un trattino basso, utilizza l'apice inverso, ad esempio `` `_mytable` ``. I caratteri speciali (a eccezione del trattino basso) non sono supportati.

I nomi di tabella di Athena fanno distinzione tra maiuscole e minuscole; se però utilizzi Apache Spark, dovrai specificare nomi di tabella in minuscolo.

[(col_name data_type [COMMENT col_comment] [, ...])]

Specifica il nome di ogni colonna da creare, insieme al tipo di dati della colonna. Nei nomi di colonna non sono ammessi caratteri speciali diversi dal trattino basso `(_)`. Se `col_name` inizia con un trattino basso, racchiudi il nome della colonna tra apici inversi, ad esempio `` `_mycolumn` ``.

Il valore `data_type` può essere qualunque tra i seguenti:

- `boolean`: i valori validi sono `true` e `false`.
- `tinyint`: un numero intero firmato a 8 bit in formato a due complementi, con un valore minimo pari a -2^7 e un valore massimo pari a 2^7-1.
- `smallint`: un numero intero firmato a 16 bit in formato a due complementi, con un valore minimo pari a -2^15 e un valore massimo pari a 2^15-1.
- `int`: nelle query DDL (Data Definition Language) come`CREATE TABLE`, utilizzare la parola chiave `int` per rappresentare un numero intero. In tutte le altre query, utilizzare la parola chiave `integer`, dove `integer` è rappresentato come valore firmato a 32 bit in formato a due complementi, con un valore minimo pari a -2^31 e un valore massimo pari a 2^31-1. Nel driver JDBC viene restituito `integer` per garantire la compatibilità con le applicazioni di analisi aziendale.
- `bigint`: un numero intero firmato a 64 bit in formato a due complementi, con un valore minimo pari a -2^63 e un valore massimo pari a 2^63-1.
- `double`: un numero a virgola mobile a precisione doppia firmato a 64 bit. L'intervallo va da 4.94065645841246544e-324d a 1.79769313486231570e+308d, positivo o negativo. `double` segue IEEE Standard for Floating-Point Arithmetic (IEEE 754).
- `float`: un numero a virgola mobile a precisione singola a 32 bit. L'intervallo va da 1.40129846432481707e-45 a 3.40282346638528860e+38, positivo o negativo. `float` segue IEEE Standard for Floating-Point Arithmetic (IEEE 754). Equivalente a `real` in Presto. In Athena, usa `float` nelle istruzioni DDL come `CREATE TABLE` e `real` nelle funzioni SQL come`SELECT CAST`. Il AWS Glue crawler restituisce i valori in `float` e Athena traduce `real` e `float` digita internamente (vedi le note di rilascio). 5 giugno 2018

- `decimal [ (precision, scale) ]`, dove *precision* è il numero totale di cifre e *scale* (facoltativo) è il numero di cifre nella parte frazionaria, il valore di default è 0. Ad esempio, è possibile usare il tipo di queste definizioni: `decimal(11,5)`, `decimal(15)`. Il valore massimo per la *precisione* è 38 mentre il valore massimo per la *scala* è 38.

  Per specificare valori decimali come letterali, ad esempio durante la selezione delle righe con un determinato valore decimale nell'espressione di una query DDL, specifica la definizione di tipo `decimal` e indica il valore decimale come letterale (tra virgolette singole) nella query, come in questo esempio: `decimal_value = decimal '0.12'`.
- `char`: lunghezza fissa dei dati dei caratteri specificata tra 1 e 255, ad esempio `char(10)`. Per ulteriori informazioni, consulta la sezione relativa a tipo di dati Hive CHAR.
- `varchar`: lunghezza variabile dei dati dei caratteri specificata tra 1 e 65535, ad esempio `varchar(10)`. Per ulteriori informazioni, consulta la sezione relativa a tipo di dati Hive VARCHAR.
- `string`: una stringa letterale racchiusa tra virgolette singole o doppie.

  > **Note**
  >
  > I tipi di dati non stringa non possono essere diffusi in `string` in Athena, ma possono essere diffusi in `varchar`.

- `binary`: (per i dati in Parquet)
- `date`: un dato in formato ISO, ad esempio *YYYY*-*MM*-*DD*. Ad esempio, `date '2008-09-15'`. Un'eccezione è SerDe OpenCSV, che utilizza il numero di giorni trascorsi dal 1° gennaio 1970. Per ulteriori informazioni, consulta OpenCSV per l'elaborazione di file SerDe CSV.
- `timestamp`: data e ora istantanea in un formato compatibile con `java.sql.Timestamp` fino a una risoluzione massima in millisecondi, come *yyyy*-*MM*-*dd* *HH*:*mm*:*ss*`[.f...]`. Ad esempio, `timestamp '2008-09-15 03:04:05.324'`. Un'eccezione è SerDe OpenCSV, che `TIMESTAMP` utilizza dati in formato numerico UNIX (ad esempio,). `1579059880000` Per ulteriori informazioni, consulta OpenCSV per l'elaborazione di file SerDe CSV.
- `array` < data_type >
- `map` < primitive_type, data_type >
- `struct` < col_name : data_type [COMMENT col_comment] [, ...] >

[COMMENT table_comment]

Crea la proprietà della tabella `comment` e la popola con il parametro `table_comment` specificato.

[PARTITIONED BY (col_name data_type [COMMENT col_comment], ...)]

Crea una tabella partizionata con una o più colonne di partizione in cui sono specificati `col_name`, `data_type` e `col_comment`. Una tabella può avere una o più partizioni, che consistono in una combinazione distinta di nome e valore per la colonna. Viene creata una directory di dati separata per ciascuna combinazione specificata; in alcune circostanze ciò può migliorare le prestazioni delle query. Le colonne partizionate non esistono all'interno dei dati della tabella stessa. Se per `col_name` usi un valore uguale a una colonna della tabella, viene restituito un errore. Per ulteriori informazioni, consulta la pagina relativa al partizionamento di dati.

> **Note**
>
> Dopo aver creato una tabella con partizioni, esegui una successiva query che comprenda la clausola MSCK REPAIR TABLE per aggiornare i metadati della partizione, ad esempio `MSCK REPAIR TABLE cloudfront_logs;`. Per le partizioni che non sono compatibili con Hive, utilizzare ALTER TABLE ADD PARTITION per caricare le partizioni in modo da poter eseguire una query sui dati.

[CLUSTERED BY (col_name, col_name, ...) INTO num_buckets BUCKETS]

Divide, con o senza partizionamento, i dati nelle colonne `col_name` specificate in sottoinsiemi di dati chiamati bucket. Il parametro `num_buckets` specifica il numero di bucket da creare. Il bucketing può migliorare le prestazioni di alcune query su set di dati di grandi dimensioni.

[ROW FORMAT row_format]

Specifica il formato di riga della tabella e i relativi dati di origine sottostanti, se applicabile. In `row_format`, è possibile specificare uno o più delimitatori con la clausola `DELIMITED` o, in alternativa, utilizzare la clausola `SERDE` come descritto di seguito. Se `ROW FORMAT` viene omesso o specificato, viene utilizzato `ROW FORMAT DELIMITED` un file nativo. SerDe

- [DELIMITED FIELDS TERMINATED BY char [ESCAPED BY char]]
- [DELIMITED COLLECTION ITEMS TERMINATED BY char]
- [MAP KEYS TERMINATED BY char]
- [LINES TERMINATED BY char]
- [NULL DEFINED AS char]

Disponibile solo con Hive 0.13 e quando il formato del file STORED AS è `TEXTFILE`.

--OPPURE--

- SERDE 'serde_name' [WITH SERDEPROPERTIES ("property_name" = "property_value", "property_name" = "property_value" [, ...])]

  `serde_name`Indica la modalità SerDe di utilizzo. La `WITH SERDEPROPERTIES` clausola consente di fornire una o più proprietà personalizzate consentite da. SerDe

[STORED AS file_format]

Specifica il formato di file per i dati della tabella. Se omesso, il valore predefinito è `TEXTFILE`. Le opzioni per `file_format` sono:

- SEQUENCEFILE
- TEXTFILE
- RCFILE
- ORC
- PARQUET
- AVRO
- ION
- INPUTFORMAT input_format_classname OUTPUTFORMAT output_format_classname

[POSIZIONE 's3://DOC-EXAMPLE-BUCKET/[folder]/']

Specifica la posizione dei dati sottostanti in Amazon S3 da cui viene creata la tabella. Il percorso della posizione deve essere un nome bucket o un nome bucket e una o più cartelle. Se si utilizzano le partizioni, specificare la radice dei dati partizionati. Per ulteriori informazioni sulla posizione della tabella, consulta Posizione delle tabelle in Amazon S3. Per informazioni sul formato dei dati e le autorizzazioni, consulta la sezione Requisiti per tabelle in Athena e dati in Amazon S3.

Utilizza una barra finale per la cartella o il bucket. Non utilizzare nomi di file o caratteri glob.

Utilizza:

```
s3://DOC-EXAMPLE-BUCKET/
```

```
s3://DOC-EXAMPLE-BUCKET/folder/
```

```
s3://DOC-EXAMPLE-BUCKET/folder/anotherfolder/
```

Non utilizzare:

`s3://DOC-EXAMPLE-BUCKET`

`s3://DOC-EXAMPLE-BUCKET/*`

`s3://DOC-EXAMPLE-BUCKET/mydatafile.dat`

[TBLPROPERTIES (['has_encrypted_data'='true | false',] ['classification'='classification_value',] property_name=property_value [, ...])]

Specifica coppie personalizzate di metadati chiave-valore per la definizione della tabella, oltre a proprietà predefinite per la tabella, ad esempio `"comment"`.

has_encrypted_data: Athena ha una proprietà integrata, `has_encrypted_data`. Impostata su `true` indica che il set di dati sottostanti specificato da `LOCATION` è crittografati. Se omesso e se le impostazioni del gruppo di lavoro non sostituiscono le impostazioni lato client, viene utilizzato `false`. Se omesso o impostato su `false` con dati sottostanti crittografati, la query restituisce un errore. Per ulteriori informazioni, consulta Crittografia a riposo.

classificazione: le tabelle create per Athena nella CloudTrail console vengono aggiunte `cloudtrail` come valore per la `classification` proprietà. Per eseguire processi ETL, è AWS Glue necessario creare una tabella con la `classification` proprietà per indicare il tipo di dati per AWS Glue as`csv`,, `parquet` `orc`o `avro`. `json` Ad esempio, `'classification'='csv'`. Se questa proprietà non viene specificata, i processi ETL non saranno completati con successo. È possibile specificarla in un secondo momento tramite la console AWS Glue , l'API o la CLI. Per ulteriori informazioni, consulta la sezione Authoring Jobs in Utilizzo AWS Glue di job per ETL con AthenaAWS Glue nella AWS Glue Developer Guide.

compression_level: la proprietà `compression_level` specifica il livello di compressione da utilizzare. Questa proprietà si applica solo alla compressione ZSTD. I valori possibili sono compresi tra 1 e 22. Il valore predefinito è 3. Per ulteriori informazioni, consulta Utilizzo dei livelli di compressione ZSTD in Athena.

Per ulteriori informazioni su altre proprietà della tabella, consulta ALTER TABLE SET TBLPROPERTIES.

Esempi

L'`CREATE TABLE`istruzione di esempio seguente crea una tabella basata su dati planetari separati da tabulazioni archiviati in Amazon S3.

```
CREATE EXTERNAL TABLE planet_data (
  planet_name string,
  order_from_sun int,
  au_to_sun float,
  mass float,
  gravity_earth float,
  orbit_years float,
  day_length float
  )
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\t'
STORED AS TEXTFILE
LOCATION 's3://DOC-EXAMPLE-BUCKET/tsv/'
```

Notare i seguenti punti:

- La `ROW FORMAT DELIMITED` clausola indica che i dati sono delimitati da un carattere specifico.
- La `FIELDS TERMINATED BY '\t'` clausola specifica che i campi nei dati TSV sono separati dal carattere di tabulazione ('\ t').
- La `STORED AS TEXTFILE` clausola indica che i dati vengono archiviati come file di testo semplice in Amazon S3.

Per interrogare i dati, puoi usare una semplice `SELECT` dichiarazione come la seguente:

```
SELECT * FROM planet_data
```

Per utilizzare l'esempio per creare la tua tabella TSV in Athena, sostituisci i nomi di tabelle e colonne con i nomi e i tipi di dati della tua tabella e delle tue colonne e aggiorna `LOCATION` la clausola in modo che punti al percorso Amazon S3 in cui sono archiviati i tuoi file TSV.

Per ulteriori informazioni sulla creazione delle tabelle, consulta Creazione di tabelle in Athena.

CREATE TABLE AS

Crea una nuova tabella popolata con i risultati di una query SELECT. Per creare una tabella vuota, utilizza CREATE TABLE. `CREATE TABLE AS` combina una dichiarazione DDL `CREATE TABLE` con una dichiarazione DML `SELECT`, quindi tecnicamente contiene sia DDL sia DML. Tieni presente che, sebbene `CREATE TABLE AS` siano raggruppate qui con altre dichiarazioni DDL, le query CTAS

in Athena vengono trattate come DML ai fini delle Service Quotas. Per informazioni sulle Service Quotas in Athena, consulta Service Quotas (Quote di Servizio).

> **Note**
>
> Per le istruzioni CTAS, l'impostazione prevista per il proprietario del bucket non si applica alla posizione della tabella di destinazione in Amazon S3. L'impostazione prevista per il proprietario del bucket si applica solo al percorso di output di Amazon S3 specificato per i risultati delle query di Athena. Per ulteriori informazioni, consulta Specificare una posizione dei risultati delle query utilizzando la console Athena.

Per ulteriori informazioni su `CREATE TABLE AS` oltre all'ambito di questo argomento di riferimento, consulta Creazione di una tabella dai risultati delle query (CTAS).

Argomenti

- Riepilogo
- Proprietà tabella CTAS
- Esempi

Riepilogo

```
CREATE TABLE table_name
[ WITH ( property_name = expression [, ...] ) ]
AS query
[ WITH [ NO ] DATA ]
```

Dove:

WITH (property_name = expression[, ...])

Un elenco di proprietà tabella CTAS facoltative, alcune delle quali sono specifiche per il formato di storage dei dati. Per informazioni, consulta Proprietà tabella CTAS.

query

Query SELECT utilizzata per creare una nuova tabella.

> **Important**
>
> Se prevedi di creare una query con partizioni, specifica i nomi delle colonne partizionate nelle ultime voci dell'elenco delle colonne nella dichiarazione `SELECT`.

[WITH [NO] DATA]

Se si utilizza `WITH NO DATA`, viene creata una nuova tabella vuota con lo stesso schema della tabella originale.

> **Note**
>
> Per includere le intestazioni di colonna nell'output dei risultati della query, è possibile utilizzare una semplice query `SELECT` anziché una query CTAS. È possibile recuperare i risultati dalla posizione dei risultati della query o scaricarli direttamente utilizzando la console Athena. Per ulteriori informazioni, consulta Utilizzo dei risultati delle query, delle query recenti e dei file di output.

Proprietà tabella CTAS

Ogni tabella CTAS in Athena dispone di un elenco di proprietà tabella CTAS opzionali specificate dall'utente utilizzando `WITH (property_name = expression [, ...] )`. Per ulteriori informazioni sull'uso di questi parametri, consulta Esempi di query CTAS.

`WITH (property_name = expression [, ...], )`

`table_type = ['HIVE', 'ICEBERG']`

Facoltativo. Il valore predefinito è `HIVE`. Specifica il tipo di tabella della tabella risultante

Esempio:

```
WITH (table_type ='ICEBERG')
```

`external_location = [location]`

> **Note**
>
> Poiché le tabelle Iceberg non sono esterne, questa proprietà non si applica ad esse. Per definire la posizione principale di una tabella Iceberg in un'istruzione CTAS, utilizza la proprietà `location` descritta più avanti in questa sezione.

Facoltativo. La posizione nella quale Athena salva la query CTAS in Amazon S3.

Esempio:

```
WITH (external_location ='s3://DOC-EXAMPLE-BUCKET/tables/parquet_table/')
```

Athena non utilizza due volte lo stesso percorso per i risultati della query. Se si specifica il percorso manualmente, verificare che il percorso Amazon S3 non contenga dei dati. Athena non tenta mai di cancellare i tuoi dati. Se si desidera utilizzare di nuovo la stessa posizione, occorre eliminare manualmente i dati, altrimenti la query CTAS fallirà.

Se si esegue una query CTAS che specifica una posizione `external_location` in un gruppo di lavoro che applica una posizione dei risultati della query, la query non riesce e viene mostrato un messaggio di errore. Per visualizzare la posizione dei risultati della query specificata per il gruppo di lavoro, vedere i dettagli del gruppo di lavoro.

Se il gruppo di lavoro sostituisce l'impostazione lato client per la posizione dei risultati della query, Athena crea la tabella nel percorso seguente:

```
s3://DOC-EXAMPLE-BUCKET/tables/query-id/
```

Se non si utilizza la proprietà `external_location` per specificare una posizione e il gruppo di lavoro non sostituisce le impostazioni lato client, Athena utilizza l'impostazione lato client per la posizione dei risultati della query per creare la tabella nel percorso seguente:

```
s3://DOC-EXAMPLE-BUCKET/Unsaved-or-query-name/year/month/date/tables/query-id/
```

`is_external = [boolean]`

Facoltativo. Indica se la tabella è una tabella esterna. Il valore predefinito è true. Per le tabelle Iceberg, questo valore deve essere impostato su false.

Esempio:

```
WITH (is_external = false)
```

location = [location]

Obbligatorio per le tabelle Iceberg. Specifica la posizione principale della tabella Iceberg da creare partendo dai risultati della query.

Esempio:

```
WITH (location ='s3://DOC-EXAMPLE-BUCKET/tables/iceberg_table/')
```

field_delimiter = [delimiter]

Facoltativo e specifiche per formati di storage dei dati basati su testo. Il separatore di campo a carattere singolo per file in CSV, TSV e file di testo. Ad esempio, `WITH (field_delimiter = ',')`. Attualmente, i delimitatori di campo multicanale non sono supportati per le query CTAS. Se non si specifica un delimitatore, per impostazione predefinita viene utilizzato `\001`.

format = [storage_format]

Il formato di archiviazione per i risultati delle query CTAS, ad esempio `ORC`, `PARQUET`, `AVRO`, `JSON`, `ION` o `TEXTFILE`. Per le tabelle Iceberg, i formati consentiti sono `ORC`, `PARQUET` e `AVRO`. Se omesso, per impostazione predefinita viene utilizzato `PARQUET`. Il nome di questo parametro `format` deve essere elencato in lettere minuscole, altrimenti la query CTAS fallirà.

Esempio:

```
WITH (format = 'PARQUET')
```

bucketed_by = ARRAY[column_name[,…], bucket_count = [int]]

> **Note**
>
> Questa proprietà non è valida per le tabelle Iceberg. Per le tabelle Iceberg, utilizza il partizionamento con trasformazione del bucket.

Elenco matrice di bucket per periodizzare i dati. Se omesso, Athena non salva tuoi dati nel bucket in questa query.

bucket_count = [int]

> **Note**
>
> Questa proprietà non è valida per le tabelle Iceberg. Per le tabelle Iceberg, utilizza il partizionamento con trasformazione del bucket.

Il numero di bucket per la periodizzazione dei dati. Se omesso, Athena non conserva i dati nel bucket. Esempio:

```
CREATE TABLE bucketed_table WITH (
  bucketed_by = ARRAY[column_name],
  bucket_count = 30, format = 'PARQUET',
  external_location ='s3://DOC-EXAMPLE-BUCKET/tables/parquet_table/'
) AS
SELECT
  *
FROM
  table_name
```

partitioned_by = ARRAY[col_name[,…]]

> **Note**
>
> Questa proprietà non è valida per le tabelle Iceberg. Per utilizzare le trasformazioni delle partizioni per le tabelle Iceberg, utilizza la proprietà `partitioning` descritta più avanti in questa sezione.

Facoltativo. Un elenco matrice di colonne in base al quale sarà partizionata la tabella CTAS. Verificare che i nomi delle colonne partizionate siano elencate per ultimi nell'elenco delle colonne della dichiarazione `SELECT`.

partitioning = ARRAY[partition_transform, ...]

Facoltativo. Specifica il partizionamento della tabella Iceberg da creare. Iceberg supporta un'ampia varietà di trasformazioni ed evoluzione delle partizioni. La tabella seguente riepiloga le trasformazioni delle partizioni.

Trasformazione	Descrizione
`year(ts)`	Crea una partizione per ogni anno. Il valore della partizione è la differenza espressa con numero intero degli anni compresi tra `ts` e il 1° gennaio 1970.
`month(ts)`	Crea una partizione per ogni mese di ogni anno. Il valore della partizione è la differenza espressa con numero intero dei mesi compresi tra `ts` e il 1° gennaio 1970.
`day(ts)`	Crea una partizione per ogni giorno di ogni anno. Il valore della partizione è la differenza espressa con numero intero dei giorni compresi tra `ts` e il 1° gennaio 1970.
`hour(ts)`	Crea una partizione per ogni ora di ogni giorno. Il valore della partizione è un timestamp con i minuti e i secondi impostati su zero.
`bucket(x, nbuckets)`	Esegue l'hashing dei dati nel numero di bucket specificato. Il valore della partizione è un hash intero di `x`, con un valore compreso tra 0 e `nbuckets - 1` inclusi.
`truncate(s, nchars)`	Imposta il valore della partizione come primi `nchars` caratteri di `s`.

Esempio:

```
WITH (partitioning = ARRAY['month(order_date)',
                           'bucket(account_number, 10)',
                           'country']))
```

`optimize_rewrite_min_data_file_size_bytes = [long]`

Facoltativo. Configurazione specifica dell'ottimizzazione dei dati I file più piccoli del valore specificato sono inclusi per la compattazione. Il valore predefinito è 0,75 volte il valore di `write_target_data_file_size_bytes`. Questa proprietà si applica solo alle tabelle Iceberg. Per ulteriori informazioni, consulta Ottimizzazione delle tabelle Iceberg.

Esempio:

```
WITH (optimize_rewrite_min_data_file_size_bytes = 402653184)
```

optimize_rewrite_max_data_file_size_bytes = [long]

Facoltativo. Configurazione specifica dell'ottimizzazione dei dati I file più grandi del valore specificato sono inclusi per la compattazione. Il valore predefinito è 1,8 volte il valore di `write_target_data_file_size_bytes`. Questa proprietà si applica solo alle tabelle Iceberg. Per ulteriori informazioni, consulta Ottimizzazione delle tabelle Iceberg.

Esempio:

```
WITH (optimize_rewrite_max_data_file_size_bytes = 966367641)
```

optimize_rewrite_data_file_threshold = [int]

Facoltativo. Configurazione specifica dell'ottimizzazione dei dati Se ci sono meno file di dati che richiedono un'ottimizzazione rispetto alla soglia specificata, i file non vengono riscritti. Ciò consente l'accumulo di più file di dati per produrre file più vicini alle dimensioni di destinazione e saltare i calcoli non necessari per risparmiare sui costi. Il predefinito è 5. Questa proprietà si applica solo alle tabelle Iceberg. Per ulteriori informazioni, consulta Ottimizzazione delle tabelle Iceberg.

Esempio:

```
WITH (optimize_rewrite_data_file_threshold = 5)
```

optimize_rewrite_delete_file_threshold = [int]

Facoltativo. Configurazione specifica dell'ottimizzazione dei dati Se a un file di dati sono associati meno file di eliminazione rispetto alla soglia, il file di dati non viene riscritto. Ciò consente l'accumulo di più file di eliminazione per ogni file di dati per risparmiare sui costi. Il valore predefinito è 2. Questa proprietà si applica solo alle tabelle Iceberg. Per ulteriori informazioni, consulta Ottimizzazione delle tabelle Iceberg.

Esempio:

```
WITH (optimize_rewrite_delete_file_threshold = 2)
```

`vacuum_min_snapshots_to_keep = [int]`

Facoltativo. Configurazione specifica per il vacuum. Il numero minimo di snapshot più recenti da mantenere. Il valore di default è 1. Questa proprietà si applica solo alle tabelle Iceberg. Per ulteriori informazioni, consulta VACUUM.

> **Note**
>
> La proprietà `vacuum_min_snapshots_to_keep` richiede la versione 3 del motore Athena.

Esempio:

```
WITH (vacuum_min_snapshots_to_keep = 1)
```

`vacuum_max_snapshot_age_seconds = [long]`

Facoltativo. Configurazione specifica per il vacuum. Un periodo in secondi che rappresenta l'età degli snapshot da mantenere. Il valore predefinito è 432.000 (5 giorni). Questa proprietà si applica solo alle tabelle Iceberg. Per ulteriori informazioni, consulta VACUUM.

> **Note**
>
> La proprietà `vacuum_max_snapshot_age_seconds` richiede la versione 3 del motore Athena.

Esempio:

```
WITH (vacuum_max_snapshot_age_seconds = 432000)
```

`write_compression = [compression_format]`

Il tipo di compressione da utilizzare per qualsiasi formato di archiviazione che consente di specificare la compressione. Il valore `compression_format` specifica la compressione da utilizzare quando i dati vengono scritti nella tabella. È possibile specificare la compressione per i formati file `TEXTFILE`, `JSON`, `PARQUET` e `ORC`.

Ad esempio, se la proprietà `format` specifica `PARQUET` come formato di archiviazione, il valore per `write_compression` specifica il formato di compressione per Parquet. In questo caso, specificare un valore per `write_compression` equivale a specificare un valore per `parquet_compression`.

Analogamente, se la proprietà `format` specifica `ORC` come formato di archiviazione, il valore per `write_compression` specifica il formato di compressione per ORC. In questo caso, specificare un valore per `write_compression` equivale a specificare un valore per `orc_compression`.

Non è possibile specificare più proprietà della tabella dei formati di compressione nella stessa query CTAS. Ad esempio, non è possibile specificare sia `write_compression` che `parquet_compression` nella stessa query. Lo stesso vale per `write_compression` e `orc_compression`. Per informazioni sui tipi di compressione supportati per ciascun formato di file, consultare Supporto della compressione in Athena.

`orc_compression = [compression_format]`

Il tipo di compressione da utilizzare per il formato di file `ORC` quando i dati `ORC` vengono scritti nella tabella. Ad esempio, `WITH (orc_compression = 'ZLIB')`. I blocchi all'interno del file `ORC` (eccetto il Postscript `ORC`) vengono compressi utilizzando la compressione specificata. Se omesso, per impostazione predefinita per `ORC` viene utilizzata la compressione ZLIB.

> **Note**
>
> Si consiglia invece di utilizzare la proprietà `write_compression` al posto di `orc_compression`. Utilizzare la proprietà `format` per specificare il formato di archiviazione come `ORC`, quindi utilizzare la proprietà `write_compression` per specificare il formato di compressione che sarà utilizzato da `ORC`.

`parquet_compression = [compression_format]`

Il tipo di compressione da utilizzare per il formato di file Parquet quando i dati Parquet vengono scritti nella tabella. Ad esempio, `WITH (parquet_compression = 'SNAPPY')`. Questa compressione viene applicata ai blocchi di colonna all'interno dei file Parquet. Se omesso, per impostazione predefinita per Parquet viene utilizzata la compressione GZIP.

> **Note**
>
> Si consiglia invece di utilizzare la proprietà `write_compression` al posto di `parquet_compression`. Utilizzare la proprietà `format` per specificare il formato di archiviazione come `PARQUET`, quindi utilizzare la proprietà `write_compression` per specificare il formato di compressione che sarà utilizzato da `PARQUET`.

`compression_level = [compression_level]`

Il livello di compressione da utilizzare. Questa proprietà si applica solo alla compressione ZSTD. I valori possibili sono compresi tra 1 e 22. Il valore predefinito è 3. Per ulteriori informazioni, consulta Utilizzo dei livelli di compressione ZSTD in Athena.

Esempi

Per esempi di query CTAS, consultare le risorse seguenti.

- Esempi di query CTAS
- Utilizzo di CTAS e INSERT INTO per ETL e analisi dei dati
- Uso di istruzioni CTAS con Amazon Athena per ridurre i costi e migliorare le prestazioni
- Utilizzo di CTAS e INSERT INTO per aggirare il limite di 100 partizioni

CREATE VIEW

Crea una nuova visualizzazione da una query `SELECT` specificata. La visualizzazione è una tabella logica a cui le query future potranno fare riferimento. Le visualizzazioni non contengono e non scrivono dati. Al contrario, la query specificata dalla visualizzazione viene eseguita ogni volta che si fa riferimento alla visualizzazione da un'altra query.

> **Note**
>
> In questo argomento vengono fornite informazioni di riepilogo per riferimento. Per informazioni più dettagliate sull'utilizzo delle visualizzazioni in Athena, consultare Utilizzo delle visualizzazioni. Per informazioni sui limiti di visualizzazione, vedereLimitazioni per le visualizzazioni.

Riepilogo

```
CREATE [ OR REPLACE ] VIEW view_name AS query
```

La clausola opzionale `OR REPLACE` consente di aggiornare la visualizzazione esistente sostituendola. Per ulteriori informazioni, consulta Creazione di visualizzazioni.

Esempi

Per creare una visualizzazione `test` dalla tabella `orders`, utilizza una query simile alla seguente:

```
CREATE VIEW test AS
SELECT
orderkey,
orderstatus,
totalprice / 2 AS half
FROM orders;
```

Per creare una visualizzazione `orders_by_date` dalla tabella `orders`, utilizza la query seguente:

```
CREATE VIEW orders_by_date AS
SELECT orderdate, sum(totalprice) AS price
FROM orders
GROUP BY orderdate;
```

Per aggiornare una visualizzazione esistente, utilizza un esempio simile al seguente:

```
CREATE OR REPLACE VIEW test AS
SELECT orderkey, orderstatus, totalprice / 4 AS quarter
FROM orders;
```

Consulta anche SHOW COLUMNS, SHOW CREATE VIEW, DESCRIBE VIEW e DROP VIEW.

DESCRIBE

Visualizza una o più colonne, tra cui quelle di partizione, per la tabella specificata. Questo comando è utile per esaminare gli attributi di colonne complesse.

Riepilogo

```
DESCRIBE [EXTENDED | FORMATTED] [db_name.]table_name [PARTITION partition_spec]
 [col_name ( [.field_name] | [.'$elem$'] | [.'$key$'] | [.'$value$'] )]
```

> **Important**
>
> La sintassi per questa istruzione è `DESCRIBE table_name`, non `DESCRIBE TABLE table_name`. L'utilizzo di quest'ultima sintassi genera il messaggio di errore FAILED: SemanticException [Error 10001]: Tabella non trovata tabella.

Parametri

[EXTENDED | FORMATTED]

Determina il formato dell'output. L'omissione di questi parametri mostra i nomi delle colonne e i relativi tipi di dati, incluse le colonne di partizione, in formato tabulare. La specificazione `FORMATTED` non solo mostra i nomi delle colonne e i tipi di dati in formato tabulare, ma anche informazioni dettagliate sulla tabella e sull'archiviazione. `EXTENDED` mostra le informazioni sulle colonne e sui tipi di dati in formato tabulare e metadati dettagliati per la tabella in formato serializzato Thrift. Questo formato è meno leggibile ed è utile principalmente per il debug.

[PARTITION partition_spec]

Se incluso, elenca i metadati per la partizione specificata da `partition_spec`, dove `partition_spec` è nel formato `(partition_column = partition_col_value, partition_column = partition_col_value, ...)`.

[col_name ([.field_name] | [.'$elem$'] | [.'key'] | [.'$value$'])*]

Specifica la colonna e gli attributi da esaminare. Puoi specificare `.field_name` per un elemento di una struttura, `'$elem$'` per un elemento array, `'$key$'` per una chiave di mappatura e `'$value$'` per un valore di mappatura. È possibile specificare questi parametri in modo ricorsivo per esplorare ulteriormente la colonna complessa.

Esempi

```
DESCRIBE orders
```

```
DESCRIBE FORMATTED mydatabase.mytable PARTITION (part_col = 100) columnA;
```

La seguente query e il relativo output mostrano le informazioni relative alla colonna e al tipo di dati da una tabella `impressions` basata sui dati di esempio Amazon EMR.

```
DESCRIBE impressions
```

```
requestbegintime          string                                          from
 deserializer
adid                      string                                          from
 deserializer
impressionid              string                                          from
 deserializer
referrer                  string                                          from
 deserializer
useragent                 string                                          from
 deserializer
usercookie                string                                          from
 deserializer
ip                        string                                          from
 deserializer
number                    string                                          from
 deserializer
processid                 string                                          from
 deserializer
browsercokie              string                                          from
 deserializer
requestendtime            string                                          from
 deserializer
timers                    struct<modellookup:string,requesttime:string>  from
 deserializer
threadid                  string                                          from
 deserializer
hostname                  string                                          from
 deserializer
sessionid                 string                                          from
 deserializer
dt                        string

# Partition Information
# col_name                data_type               comment
```

```
dt                      string
```

La query e l'output di esempio riportati di seguito mostrano il risultato per la stessa tabella quando l'opzione FORMATTED viene utilizzata.

```
DESCRIBE FORMATTED impressions
```

```
requestbegintime          string                                              from
 deserializer
adid                      string                                              from
 deserializer
impressionid              string                                              from
 deserializer
referrer                  string                                              from
 deserializer
useragent                 string                                              from
 deserializer
usercookie                string                                              from
 deserializer
ip                        string                                              from
 deserializer
number                    string                                              from
 deserializer
processid                 string                                              from
 deserializer
browsercokie              string                                              from
 deserializer
requestendtime            string                                              from
 deserializer
timers                    struct<modellookup:string,requesttime:string>  from
 deserializer
threadid                  string                                              from
 deserializer
hostname                  string                                              from
 deserializer
sessionid                 string                                              from
 deserializer
dt                        string

# Partition Information
# col_name                data_type                comment

dt                        string
```

```
# Detailed Table Information
Database:                 sampledb
Owner:                    hadoop
CreateTime:               Thu Apr 23 02:55:21 UTC 2020
LastAccessTime:           UNKNOWN
Protect Mode:             None
Retention:                0
Location:                 s3://us-east-1.elasticmapreduce/samples/hive-ads/tables/
impressions
Table Type:               EXTERNAL_TABLE
Table Parameters:
        EXTERNAL                  TRUE
        transient_lastDdlTime     1587610521

# Storage Information
SerDe Library:                        org.openx.data.jsonserde.JsonSerDe
InputFormat:                          org.apache.hadoop.mapred.TextInputFormat
OutputFormat:
 org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat
Compressed:                           No
Num Buckets:                          -1
Bucket Columns:                       []
Sort Columns:                         []
Storage Desc Params:
        paths                                 requestbegintime, adid, impressionid,
 referrer, useragent, usercookie, ip
        serialization.format                  1
```

La query e l'output di esempio riportati di seguito mostrano il risultato per la stessa tabella quando l'opzione EXTENDED viene utilizzata. Le informazioni dettagliate della tabella vengono visualizzate su una singola riga, ma qui sono formattate per la leggibilità.

```
DESCRIBE EXTENDED impressions
```

```
requestbegintime          string                                              from
 deserializer
adid                      string                                              from
 deserializer
impressionid              string                                              from
 deserializer
```

```
referrer                string                                              from
 deserializer
useragent               string                                              from
 deserializer
usercookie              string                                              from
 deserializer
ip                      string                                              from
 deserializer
number                  string                                              from
 deserializer
processid               string                                              from
 deserializer
browsercokie            string                                              from
 deserializer
requestendtime          string                                              from
 deserializer
timers                  struct<modellookup:string,requesttime:string>  from
 deserializer
threadid                string                                              from
 deserializer
hostname                string                                              from
 deserializer
sessionid               string                                              from
 deserializer
dt                      string

# Partition Information
# col_name              data_type               comment

dt                      string

Detailed Table Information      Table(tableName:impressions, dbName:sampledb,
 owner:hadoop, createTime:1587610521,
lastAccessTime:0, retention:0, sd:StorageDescriptor(cols:
[FieldSchema(name:requestbegintime, type:string, comment:null),
FieldSchema(name:adid, type:string, comment:null), FieldSchema(name:impressionid,
 type:string, comment:null),
FieldSchema(name:referrer, type:string, comment:null), FieldSchema(name:useragent,
 type:string, comment:null),
FieldSchema(name:usercookie, type:string, comment:null), FieldSchema(name:ip,
 type:string, comment:null),
FieldSchema(name:number, type:string, comment:null), FieldSchema(name:processid,
 type:string, comment:null),
```

```
FieldSchema(name:browsercokie, type:string, comment:null),
 FieldSchema(name:requestendtime, type:string, comment:null),
FieldSchema(name:timers, type:struct<modellookup:string,requesttime:string>,
 comment:null), FieldSchema(name:threadid,
type:string, comment:null), FieldSchema(name:hostname, type:string, comment:null),
 FieldSchema(name:sessionid,
type:string, comment:null)], location:s3://us-east-1.elasticmapreduce/samples/hive-ads/
tables/impressions,
inputFormat:org.apache.hadoop.mapred.TextInputFormat,
outputFormat:org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat, compressed:false,
 numBuckets:-1,
serdeInfo:SerDeInfo(name:null, serializationLib:org.openx.data.jsonserde.JsonSerDe,
 parameters:{serialization.format=1,
paths=requestbegintime, adid, impressionid, referrer, useragent, usercookie, ip}),
 bucketCols:[], sortCols:[], parameters:{},
skewedInfo:SkewedInfo(skewedColNames:[], skewedColValues:[],
 skewedColValueLocationMaps:{}),
storedAsSubDirectories:false), partitionKeys:[FieldSchema(name:dt, type:string,
 comment:null)],
parameters:{EXTERNAL=TRUE, transient_lastDdlTime=1587610521}, viewOriginalText:null,
 viewExpandedText:null,
tableType:EXTERNAL_TABLE)
```

DESCRIBE VIEW

Visualizza l'elenco delle colonne per la visualizzazione specificata. In questo modo è possibile esaminare gli attributi di una visualizzazione complessa.

Riepilogo

```
DESCRIBE [db_name.]view_name
```

Esempio

```
DESCRIBE orders;
```

Consulta anche SHOW COLUMNS, SHOW CREATE VIEW, SHOW VIEWS e DROP VIEW.

DROP DATABASE

Rimuove il database denominato dal catalogo. Se il database contiene tabelle, è necessario eliminarle prima di eseguire `DROP DATABASE` o utilizzare la clausola `CASCADE`. L'uso di `DATABASE` e `SCHEMA` è intercambiabile. Significano la stessa cosa.

Riepilogo

```
DROP {DATABASE | SCHEMA} [IF EXISTS] database_name [RESTRICT | CASCADE]
```

Parametri

[IF EXISTS]

Determina la rimozione dell'errore se `database_name` non esiste.

[RESTRICT|CASCADE]

Stabilisce come sono considerate le tabelle all'interno del `database_name` durante l'operazione `DROP`. Specificando `RESTRICT`, il database non viene eliminato se contiene tabelle. Questo è il comportamento che segue di default. Specificando `CASCADE`, verranno eliminati il database e tutte le relative tabelle.

Esempi

```
DROP DATABASE clickstreams;
```

```
DROP SCHEMA IF EXISTS clickstreams CASCADE;
```

> **Note**
>
> Quando tenti di eliminare un database il cui nome contiene caratteri speciali (ad esempio, `my-database`), potresti visualizzare un messaggio di errore. Per risolvere questo problema, prova a racchiudere il nome del database tra caratteri di apice retroverso (`). Per informazioni sulla denominazione dei database in Athena, consulta Nomi di tabelle, database e colonne.

DROP TABLE

Rimuove la definizione della tabella dei metadati per la tabella denominata `table_name`. Quando elimini una tabella esterna, i dati sottostanti non vengono modificati.

Riepilogo

```
DROP TABLE [IF EXISTS] table_name
```

Parametri

[IF EXISTS]

Determina la rimozione dell'errore se `table_name` non esiste.

Esempi

```
DROP TABLE fulfilled_orders
```

```
DROP TABLE IF EXISTS fulfilled_orders
```

Quando si utilizza l'editor di query della console Athena per eliminare una tabella con caratteri speciali diversi dal carattere di sottolineatura (_), utilizzare gli apici inversi, come nell'esempio seguente.

```
DROP TABLE `my-athena-database-01.my-athena-table`
```

Quando si utilizza il connettore JDBC per eliminare una tabella con caratteri speciali, i caratteri apice inverso non sono necessari.

```
DROP TABLE my-athena-database-01.my-athena-table
```

DROP VIEW

Elimina una vista esistente. La clausola facoltativa `IF EXISTS` provoca l'errore da sopprimere se la vista non esiste.

Per ulteriori informazioni, consulta Utilizzo delle visualizzazioni.

Riepilogo

```
DROP VIEW [ IF EXISTS ] view_name
```

Esempi

```
DROP VIEW orders_by_date
```

```
DROP VIEW IF EXISTS orders_by_date
```

Consulta anche CREATE VIEW, SHOW COLUMNS, SHOW CREATE VIEW, SHOW VIEWS e DESCRIBE VIEW.

MSCK REPAIR TABLE

Utilizzare il comando `MSCK REPAIR TABLE` per aggiornare i metadati nel catalogo dopo aver aggiunto le partizioni compatibili con Hive.

Il comando `MSCK REPAIR TABLE` esegue la scansione di un file system, ad esempio Amazon S3 per la ricerca di partizioni compatibili con Hive aggiunte al file system dopo la creazione della tabella. `MSCK REPAIR TABLE` confronta le partizioni nei metadati della tabella e le partizioni in S3. Se sono presenti nuove partizioni nella posizione S3 specificata al momento della creazione della tabella, queste vengono aggiunte ai metadati e alla tabella Athena.

Quando si aggiungono partizioni fisiche, i metadati nel catalogo diventano incoerenti con il layout dei dati nel file system e devono essere aggiunte al catalogo informazioni sulle nuove partizioni. Per aggiornare i metadati, esegui `MSCK REPAIR TABLE` in modo da poter interrogare i dati nelle nuove partizioni da Athena.

> **Note**
>
> `MSCK REPAIR TABLE` aggiunge solo partizioni ai metadati; non le rimuove. Per rimuovere le partizioni dai metadati dopo che le partizioni sono state eliminate manualmente in Amazon S3, eseguire il comando `ALTER TABLE` *`table-name`* `DROP PARTITION`. Per ulteriori informazioni, consulta ALTER TABLE DROP PARTITION.

Considerazioni e limitazioni

Quando si utilizza `MSCK REPAIR TABLE`, tenere presenti le informazioni seguenti:

- È possibile che l'aggiunta di tutte le partizioni richieda vari minuti. Se l'operazione scade, acquisirà lo stato "incompleta" laddove al catalogo siano state aggiunte solo alcune partizioni. In tal caso sarà necessario eseguire `MSCK REPAIR TABLE` sulla stessa tabella finché non vengono aggiunte tutte le partizioni. Per ulteriori informazioni, consulta Partizionamento dei dati in Athena.
- Per le partizioni che non sono compatibili con Hive, utilizzare ALTER TABLE ADD PARTITION per caricare le partizioni in modo da poter eseguire una query sui dati.
- Le posizioni delle partizioni da utilizzare con Athena devono utilizzare il protocollo `s3` (ad esempio, `s3://DOC-EXAMPLE-BUCKET/folder/`). In Athena, i percorsi che utilizzano altri protocolli (ad esempio, `s3a://bucket/folder/`) determineranno errori quando le query `MSCK REPAIR TABLE` vengono eseguite sulle tabelle contenenti.
- Poiché `MSCK REPAIR TABLE` analizza sia una cartella che le relative sottocartelle per trovare uno schema di partizioni corrispondente, assicurarsi di conservare i dati per tabelle separate in gerarchie di cartelle separate. Ad esempio, supponiamo di avere dati per la tabella 1 in `s3://DOC-EXAMPLE-BUCKET1` e dati per la tabella 2 in`s3://DOC-EXAMPLE-BUCKET1/table-2-data`. Se entrambe le tabelle sono partizionate per stringa, `MSCK REPAIR TABLE` aggiungerà le partizioni per la tabella 2 alla tabella 1. Per evitare ciò, usa invece strutture di cartelle separate come `s3://DOC-EXAMPLE-BUCKET1` e. `s3://DOC-EXAMPLE-BUCKET2` Questo comportamento è coerente con Amazon EMR e Apache Hive.
- A causa di un problema noto, `MSCK REPAIR TABLE` avrà esito negativo e non verrà inviato alcun messaggio di errore quando i valori delle partizioni contengono i due punti (`:`) (ad esempio, quando il valore della partizione è un timestamp). Come soluzione alternativa, utilizza ALTER TABLE ADD PARTITION.
- `MSCK REPAIR TABLE` non aggiunge nomi di colonne di partizione che iniziano con un carattere di sottolineatura (_). Per ovviare al problema della limitazione, utilizza ALTER TABLE ADD PARTITION.

Riepilogo

```
MSCK REPAIR TABLE table_name
```

Esempi

```
MSCK REPAIR TABLE orders;
```

Risoluzione dei problemi

Dopo l'esecuzione`MSCK REPAIR TABLE`, se Athena non aggiunge le partizioni alla tabella in AWS Glue Data Catalog, verifica quanto segue:

- AWS Glue accesso: assicurati che il ruolo AWS Identity and Access Management (IAM) disponga di una politica che consenta l'azione`glue:BatchCreatePartition`. Per ulteriori informazioni, consultare Allow glue: BatchCreatePartition nella policy IAM riportata di seguito in questo documento.
- Accesso ad Amazon S3: assicurati che il ruolo disponga di una policy con autorizzazioni sufficienti per accedere ad Amazon S3, inclusa l'operazione `s3:DescribeJob`. Per un esempio di quali operazioni Amazon S3 consentire, consulta la policy del bucket di esempio in Accesso tra account in Athena a bucket Amazon S3.
- Maiuscole e minuscole delle chiavi oggetto Amazon S3: assicurati che il percorso Amazon S3 sia in minuscolo invece che in camel case (ad esempio `userid` invece di `userId`) o utilizza `ALTER TABLE ADD PARTITION` per specificare i nomi delle chiavi oggetto. Per ulteriori informazioni, consultare Modificare o ridefinire il percorso Amazon S3 riportata di seguito in questo documento.
- Timeout per query – `MSCK REPAIR TABLE` viene utilizzato al meglio quando si crea una tabella per la prima volta o quando vi è incertezza sulla parità tra i metadati dei dati e delle partizioni. Se utilizzi `MSCK REPAIR TABLE` per aggiungere frequentemente nuove partizioni (ad esempio, su base giornaliera) e si verificano timeout di query, prendere in considerazione l'utilizzo di ALTER TABLE ADD PARTITION.
- Partizioni mancanti dal file system: se elimini manualmente una partizione in Amazon S3 e poi esegui `MSCK REPAIR TABLE`, è possibile che venga visualizzato il messaggio di errore Partizioni mancanti dal file system. Ciò si verifica perché `MSCK REPAIR TABLE` non rimuove le partizioni obsolete dai metadati della tabella. Per rimuovere le partizioni eliminate dai metadati della tabella, eseguire invece ALTER TABLE DROP PARTITION. Nota cheSHOW PARTITIONS elenca in modo simile solo le partizioni nei metadati, non le partizioni nel file system.
- Errore «NullPointerException name is null»

  Se si utilizza l'operazione AWS Glue CreateTableAPI o il AWS CloudFormation `AWS::Glue::Table`modello per creare una tabella da utilizzare in Athena senza specificare la `TableType` proprietà e quindi si esegue una query DDL come `SHOW CREATE TABLE` o`MSCK REPAIR TABLE`, è possibile ricevere il messaggio di errore FAILED: NullPointerException Name is null.

Per risolvere l'errore, specifica un valore per l'TableInput`TableType`attributo come parte della chiamata o del modello AWS Glue`CreateTable` API.AWS CloudFormation I valori possibili per `TableType` includono `EXTERNAL_TABLE` o `VIRTUAL_VIEW`.

Questo requisito si applica solo quando si crea una tabella utilizzando l'operazione AWS Glue `CreateTable` API o il `AWS::Glue::Table` modello. Se si crea una tabella per Athena utilizzando un'istruzione DDL o un crawler AWS Glue , la proprietà `TableType` viene definita automaticamente.

Nelle sezioni seguenti vengono fornite informazioni aggiuntive.

Allow glue: BatchCreatePartition nella policy IAM

Esamina le policy IAM collegate al ruolo che stai utilizzando per eseguire `MSCK REPAIR TABLE`. Quando si utilizza AWS Glue Data Catalog con Athena, la policy IAM deve consentire l'azione`glue:BatchCreatePartition`. Per un esempio di policy IAM che consente l'operazione `glue:BatchCreatePartition`, vedere AWS politica gestita: AmazonAthenaFullAccess.

Modificare o ridefinire il percorso Amazon S3

Se una o più chiavi oggetto nel percorso Amazon S3 sono in maiuscolo anziché in minuscolo, `MSCK REPAIR TABLE` potrebbe non essere in grado di aggiungere le partizioni a AWS Glue Data Catalog. Ad esempio, se il percorso Amazon S3 include il nome della chiave oggetto `userId`, le seguenti partizioni potrebbero non essere aggiunte a AWS Glue Data Catalog:

```
s3://DOC-EXAMPLE-BUCKET/path/userId=1/

s3://DOC-EXAMPLE-BUCKET/path/userId=2/

s3://DOC-EXAMPLE-BUCKET/path/userId=3/
```

Per risolvere il problema, procedi in uno dei seguenti modi:

- Usa le lettere minuscole anziché le maiuscole camel quando crei le tue chiavi oggetto Amazon S3:

```
s3://DOC-EXAMPLE-BUCKET/path/userid=1/

s3://DOC-EXAMPLE-BUCKET/path/userid=2/
```

```
s3://DOC-EXAMPLE-BUCKET/path/userid=3/
```

- Utilizza ALTER TABLE ADD PARTITION per ridefinire il percorso, come nell'esempio seguente:

```
ALTER TABLE table_name ADD [IF NOT EXISTS]
PARTITION (userId=1)
LOCATION 's3://DOC-EXAMPLE-BUCKET/path/userId=1/'
PARTITION (userId=2)
LOCATION 's3://DOC-EXAMPLE-BUCKET/path/userId=2/'
PARTITION (userId=3)
LOCATION 's3://DOC-EXAMPLE-BUCKET/path/userId=3/'
```

Tieni presente che, sebbene i nomi delle chiavi degli oggetti di Amazon S3 possano utilizzare lettere maiuscole, i nomi dei bucket di Amazon S3 devono essere sempre in minuscolo. Per ulteriori informazioni, consulta le Linee guida per la denominazione delle chiavi oggetto e le Regole di denominazione dei bucket nella Guida per l'utente di Amazon S3.

SHOW COLUMNS

Mostra solo i nomi delle colonne per una tabella o una vista specificata. Per ottenere informazioni più dettagliate, interroga AWS Glue Data Catalog invece il. Per informazioni ed esempi, consulta le sezioni seguenti nell'argomento Esecuzione di query AWS Glue Data Catalog:

- Per visualizzare i metadati delle colonne, ad esempio il tipo di dati, consulta Elencare o ricercare colonne per una tabella o una vista specificata.
- Per visualizzare tutte le colonne di ogni tabella in un database specifico in `AwsDataCatalog`, consulta Elencare o ricercare colonne per una tabella o una vista specificata.
- Per visualizzare tutte le colonne per tutte le tabelle in tutti i database in `AwsDataCatalog`, consulta Elencare tutte le colonne di ogni tabella.
- Per visualizzare le colonne che hanno in comune tabelle specifiche di un database, consulta Elencare le colonne che hanno in comune tabelle specifiche.

Riepilogo

```
SHOW COLUMNS {FROM|IN} database_name.table_name
```

```
SHOW COLUMNS {FROM|IN} table_name [{FROM|IN} database_name]
```

Le parole chiave `FROM` e `IN` possono essere utilizzate in modo intercambiabile. Se *`table_name`* o *`database_name`* ha caratteri speciali come trattini, racchiudi il nome tra accenti gravi (ad esempio, `` `my-database`.`my-table` ``). Non circondare *`table_name`* o *`database_name`* con virgolette singole o doppie. Attualmente, l'uso di `LIKE` e le espressioni di corrispondenza dei modelli non sono supportati.

Esempi

Gli esempi equivalenti seguenti mostrano le colonne della tabella `orders` nel database `customers`. I primi due esempi presuppongono che `customers` sia il database corrente.

```
SHOW COLUMNS FROM orders
```

```
SHOW COLUMNS IN orders
```

```
SHOW COLUMNS FROM customers.orders
```

```
SHOW COLUMNS IN customers.orders
```

```
SHOW COLUMNS FROM orders FROM customers
```

```
SHOW COLUMNS IN orders IN customers
```

SHOW CREATE TABLE

Analizza una tabella esistente denominata `table_name` per generare la query che l'ha creata.

Riepilogo

```
SHOW CREATE TABLE [db_name.]table_name
```

Parametri

TABLE [db_name.]table_name

> Il parametro db_name è facoltativo. Se omesso, il contesto rimanda automaticamente al database corrente.

> **Note**
>
> Il nome della tabella è obbligatorio.

Esempi

```
SHOW CREATE TABLE orderclickstoday;
```

```
SHOW CREATE TABLE `salesdata.orderclickstoday`;
```

Risoluzione dei problemi

Se si utilizza l'operazione AWS Glue CreateTableAPI o il AWS CloudFormation `AWS::Glue::Table`modello per creare una tabella da utilizzare in Athena senza specificare la `TableType` proprietà e quindi si esegue una query DDL come `SHOW CREATE TABLE` o`MSCK REPAIR TABLE`, è possibile ricevere il messaggio di errore FAILED: NullPointerException Name is null.

Per risolvere l'errore, specifica un valore per l'TableInput`TableType`attributo come parte della chiamata o del modello AWS Glue`CreateTable` API.AWS CloudFormation I valori possibili per `TableType` includono `EXTERNAL_TABLE` o `VIRTUAL_VIEW`.

Questo requisito si applica solo quando si crea una tabella utilizzando l'operazione AWS Glue `CreateTable` API o il `AWS::Glue::Table` modello. Se si crea una tabella per Athena utilizzando un'istruzione DDL o un crawler AWS Glue , la proprietà `TableType` viene definita automaticamente.

SHOW CREATE VIEW

Mostra l'istruzione SQL che crea la vista specificata.

Riepilogo

```
SHOW CREATE VIEW view_name
```

Esempi

```
SHOW CREATE VIEW orders_by_date
```

Consulta anche CREATE VIEW e DROP VIEW.

SHOW DATABASES

Elenca tutti i database definiti nel metastore. È possibile utilizzare `DATABASES` o `SCHEMAS`. Significano la stessa cosa.

L'equivalente programmatico di `SHOW DATABASES` è l'azione API ListDatabasesAthena. Il metodo equivalente in AWS SDK for Python (Boto3) è list_databases.

Riepilogo

```
SHOW {DATABASES | SCHEMAS} [LIKE 'regular_expression']
```

Parametri

[LIKE '***regular_expression***']

Filtra l'elenco di database mostrando solo quelli che soddisfano la *`regular_expression`* specificata. Per la corrispondenza dei caratteri jolly, è possibile utilizzare la combinazione `.*`, che corrisponde a qualsiasi carattere zero a tempi illimitati.

Esempi

```
SHOW SCHEMAS;
```

```
SHOW DATABASES LIKE '.*analytics';
```

SHOW PARTITIONS

Elenca tutte le partizioni in una tabella Athena in ordine non ordinato.

Riepilogo

```
SHOW PARTITIONS table_name
```

- Per visualizzare le partizioni in una tabella ed elencarle in un ordine specifico, vedere la sezione Elencare le partizioni per una tabella specifica nella pagina Esecuzione di query AWS Glue Data Catalog.

- Per visualizzare il contenuto di una partizione, vedere la sezione Esecuzione di query sui dati della pagina Partizionamento dei dati in Athena.
- `SHOW PARTITIONS`non elenca le partizioni proiettate da Athena ma non registrate nel catalogo. AWS Glue Per informazioni sulla proiezione delle partizioni, consulta Proiezione delle partizioni con Amazon Athena.
- `SHOW PARTITIONS` elenca le partizioni nei metadati, non le partizioni nel file system effettivo. Per aggiornare i metadati dopo aver eliminato manualmente le partizioni in Amazon S3, eseguire ALTER TABLE DROP PARTITION.

Esempi

La query di esempio seguente mostra le partizioni per l'oggetto `flight_delays_csv`, che mostra i dati della tabella di volo del Dipartimento dei Trasporti degli Stati Uniti. Per ulteriori informazioni sulle tabelle `flight_delays_csv` in questo esempio, consulta LazySimpleSerDe per CSV, TSV e file delimitati in modo personalizzato. La tabella è suddivisa per anno.

```
SHOW PARTITIONS flight_delays_csv
```

Risultati

```
year=2007
year=2015
year=1999
year=1993
year=1991
year=2003
year=1996
year=2014
year=2004
year=2011
...
```

La query di esempio seguente mostra le partizioni per l'oggetto `impressions`, che contiene dati di esplorazione Web di esempio. Per ulteriori informazioni sulle tabelle `impressions` in questo esempio, consulta Partizionamento dei dati in Athena. La tabella è partizionata dal comando `dt` (datetime).

```
SHOW PARTITIONS impressions
```

Risultati

```
dt=2009-04-12-16-00
dt=2009-04-13-18-15
dt=2009-04-14-00-20
dt=2009-04-12-13-00
dt=2009-04-13-02-15
dt=2009-04-14-12-05
dt=2009-04-14-06-15
dt=2009-04-12-21-15
dt=2009-04-13-22-15
...
```

Elenco delle partizioni in ordine ordinato

Per ordinare le partizioni nell'elenco dei risultati, utilizza la seguente sintassi `SELECT` invece di `SHOW PARTITIONS`.

```
SELECT * FROM database_name."table_name$partitions" ORDER BY column_name
```

La query seguente mostra l'elenco delle partizioni per l'esempio `flight_delays_csv`, ma in ordine.

```
SELECT * FROM "flight_delays_csv$partitions" ORDER BY year
```

Risultati

```
year
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
...
```

Per ulteriori informazioni, consulta la sezione Elencare le partizioni per una tabella specifica nella pagina Esecuzione di query AWS Glue Data Catalog.

SHOW TABLES

Elenca tutte le tabelle di base e le viste in un database.

Riepilogo

```
SHOW TABLES [IN database_name] ['regular_expression']
```

Parametri

[IN database_name]

Specifica il `database_name` da cui saranno elencate le tabelle. Se omesso, sarà utilizzato automaticamente il database dal contesto corrente.

> **Note**
>
> `SHOW TABLES` potrebbe fallire se `database_name` utilizza un carattere non supportato come un trattino. Come soluzione alternativa, prova a racchiudere il nome del database tra i backtick.

['regular_expression']

Filtra l'elenco di tabelle mostrando solo quelle che soddisfano la `regular_expression` specificata. Per indicare qualsiasi carattere nelle tabelle `AWSDataCatalog` è possibile utilizzare le espressioni jolly `*` o `.*`. Per i database Apache Hive, utilizzare l'espressione jolly `.*`. Per indicare una scelta tra i caratteri, utilizzare il carattere `|`.

Esempi

Example mostra tutte le tabelle nel database **`sampledb`**

```
SHOW TABLES IN sampledb
```

`Results`

```
alb_logs
cloudfront_logs
elb_logs
flights_2016
flights_parquet
view_2016_flights_dfw
```

Example mostra i nomi di tutte le tabelle **`sampledb`** che includono la parola "flights"

```
SHOW TABLES IN sampledb '*flights*'
```

`Results`

```
flights_2016
flights_parquet
view_2016_flights_dfw
```

Example mostra i nomi di tutte le tabelle in **`sampledb`** che terminano nella parola "logs"

```
SHOW TABLES IN sampledb '*logs'
```

`Results`

```
alb_logs
cloudfront_logs
elb_logs
```

SHOW TBLPROPERTIES

Elenca le proprietà della tabella con nome.

Riepilogo

```
SHOW TBLPROPERTIES table_name [('property_name')]
```

Parametri

[('property_name')]

Se incluso, viene elencato solo il valore della proprietà denominata `property_name`.

Esempi

```
SHOW TBLPROPERTIES orders;
```

```
SHOW TBLPROPERTIES orders('comment');
```

SHOW VIEWS

Elenca le viste nel database specificato o, se si omette il nome del database, quelle nel database corrente. Utilizza la clausola `LIKE` facoltativa con un'espressione regolare per limitare l'elenco dei nomi di vista.

Athena restituisce un elenco di valori di tipo `STRING`, in cui ogni valore è un nome della visualizzazione.

Riepilogo

```
SHOW VIEWS [IN database_name] [LIKE 'regular_expression']
```

Parametri

[IN database_name]

Specifica il `database_name` da cui saranno elencate le viste. Se omesso, sarà utilizzato automaticamente il database dal contesto corrente.

[LIKE 'regular_expression']

Filtra l'elenco delle viste mostrando solo quelle che soddisfano la `regular_expression` specificata. È possibile utilizzare solo il carattere jolly `*`, il quale indica qualsiasi carattere, o `|`, il quale indica una scelta tra caratteri.

Esempi

```
SHOW VIEWS;
```

```
SHOW VIEWS IN marketing_analytics LIKE 'orders*'
```

Consulta anche SHOW COLUMNS, SHOW CREATE VIEW, DESCRIBE VIEW e DROP VIEW.

Considerazioni e restrizioni per le query SQL in Amazon Athena

Quando si eseguono query sulle tabelle Athena, tenere presente le seguenti considerazioni e limitazioni:

- Procedure archiviate: le procedure archiviate non sono supportate.
- Numero massimo di partizioni: il numero massimo di partizioni che è possibile creare con le istruzioni `CREATE TABLE AS SELECT` (CTAS) è 100. Per ulteriori informazioni, consulta CREATE TABLE AS. Per una soluzione alternativa, vedere Utilizzo di CTAS e INSERT INTO per aggirare il limite di 100 partizioni.
- Istruzioni non supportate: le seguenti istruzioni non sono supportate:
  - `CREATE TABLE LIKE` non è supportato.
  - `DESCRIBE INPUT` e `DESCRIBE OUTPUT` non sono supportati.
  - L'istruzione `MERGE` è supportata solo per i formati di tabelle transazionali. Per ulteriori informazioni, consulta MERGE INTO.
  - Le istruzioni `UPDATE` non sono supportate.
- Connettori Trino e Presto: i connettori Trino e Presto non sono supportati. Utilizzare Amazon Athena Federated Query per collegare le origini dati. Per ulteriori informazioni, consulta Utilizzo di Amazon Athena Federated Query.
- Timeout su tabelle con molte partizioni – Athena potrebbe verificarsi un timeout quando si esegue una query su una tabella con molte migliaia di partizioni. Questo può accadere quando la tabella ha molte partizioni che non sono di tipo `string`. Quando si utilizza il tipo `string`, Athena elimina le partizioni a livello di metastore. Tuttavia, quando si utilizzano altri tipi di dati, Athena elimina le partizioni a livello di server. Più partizioni hai, più questo processo richiede tempo e più è probabile che le tue query vadano in timeout. Per risolvere questo problema, impostare il tipo di partizione su `string` in modo che Athena elimini le partizioni a livello di metastore. Ciò riduce il sovraccarico e impedisce il timeout delle query.
- Supporto per S3 Glacier: per informazioni sull'esecuzione di query su oggetti Amazon S3 Glacier ripristinati, consulta Esecuzione di query su oggetti Amazon S3 Glacier ripristinati.
- File trattati come nascosti – Athena tratta i file sorgenti che iniziano con un trattino basso (_) o un punto (`.`) come nascosti. Per aggirare questa limitazione, rinominare i file.
- Limitazione della dimensione di righe o colonne — Le dimensioni di una singola riga o delle relative colonne non può superare i 32 megabyte. Questo limite può essere superato quando, ad esempio,

una riga in un file CSV o JSON contiene una singola colonna di 300 megabyte. Il superamento di questo limite può anche produrre il messaggio di errore Riga troppo lunga nel file di testo. Per aggirare questa limitazione, assicurati che la somma dei dati delle colonne in qualsiasi riga sia inferiore a 32 MB.

- Massimo clausola LIMIT: il numero massimo di righe che è possibile specificare per la clausola `LIMIT` è

  9223372036854775807. Quando si utilizza `ORDER BY`, il numero massimo di righe supportate per la clausola LIMIT è 2147483647. Il superamento di questo limite causa il messaggio di errore NOT_SUPPORTED: ORDER BY LIMIT > 2147483647 is not supported (NOT_SUPPORTED: ORDER BY LIMIT > 2147483647 non è supportato).
- information_schema: l'interrogazione `information_schema` è più performante se si dispone di una quantità di metadati da piccola a moderata. AWS Glue Se disponi di una quantità di metadati elevata, possono verificarsi degli errori. Per informazioni sull'interrogazione dei metadati nel database, consulta. `information_schema` AWS Glue Esecuzione di query AWS Glue Data Catalog
- Inizializzazioni dell'array: a causa di una limitazione in Java, non è possibile inizializzare un array in Athena con più di 254 argomenti.
- Colonne di metadati nascoste: le colonne di metadati nascoste Hive o Iceberg, `$bucket`, `$file_modified_time`, `$file_size` e `$partition` non sono supportate per le visualizzazioni. Per informazioni sull'utilizzo della colonna dei metadati `$path` in Athena, consulta Ottenere le posizioni dei file per i dati di origine in Amazon S3 .

Per informazioni sulla lunghezza massima della stringa di query, sulle quote per i timeout delle query e sulle quote per il numero attivo di query DML, consulta la sezione Service Quotas (Quote di Servizio).

Risoluzione dei problemi in Athena

Il team Athena ha raccolto le seguenti informazioni sulla risoluzione dei problemi dei clienti. Sebbene non siano complete, includono consigli su alcuni problemi comuni relativi a prestazioni, timeout e memoria esaurita.

Argomenti

- CREATE TABLE AS SELECT (CTAS)
- Problemi con i file di dati

CREATE TABLE AS SELECT (CTAS)

I dati duplicati si verificano con istruzioni CTAS simultanee

Athena non mantiene la convalida simultanea per CTAS. Assicurarsi che non vi sia nessuna istruzione CTAS duplicata per la stessa posizione nello stesso momento. Anche se un'istruzione CTAS o INSERT INTO ha esito negativo, i dati orfani possono essere lasciati nella posizione dei dati specificata nell'istruzione.

HIVE_TOO_MANY_OPEN_PARTITIONS

Quando si utilizza un'istruzione CTAS per creare una tabella con più di 100 partizioni, potresti ricevere l'errore HIVE_TOO_MANY_OPEN_PARTITIONS: Exceeded limit of 100 open writers for partitions/buckets (HIVE_TOO_MANY_OPEN_PARTITIONS: limite di 100 scrittori aperti per partizioni/bucket superato). Per ovviare questa limitazione, puoi utilizzare un'istruzione CTAS e una serie di istruzioni `INSERT INTO` che creano o inseriscono fino a 100 partizioni ciascuna. Per ulteriori informazioni, consulta Utilizzo di CTAS e INSERT INTO per aggirare il limite di 100 partizioni.

Problemi con i file di dati

Athena non riesce a leggere i file nascosti

Athena considera i file sorgente che iniziano con un trattino basso (_) o un punto (.) come nascosti. Per aggirare questa limitazione, rinominare i file.

Athena legge i file che ho escluso dal crawler AWS Glue

Athena non riconosce i pattern di esclusione specificati in un AWS Glue crawler. Ad esempio, se disponi di un bucket Amazon S3 che contiene i file `.csv` e `.json` ed escludi i file `.json` dal crawler, Athena esegue query su entrambi i gruppi di file. Per evitare ciò, posizionare i file che si desidera escludere in una posizione diversa.

HIVE_BAD_DATA: Errore nell'analisi del valore del campo

Questo errore può verificarsi nei seguenti scenari:

- Il tipo di dati definito nella tabella non corrisponde ai dati di origine oppure un singolo campo contiene diversi tipi di dati. Per le risoluzioni suggerite, consulta La mia query Amazon Athena non ha esito negativo con l'errore "HIVE_BAD_DATA: Errore durante l'analisi del valore del campo per il campo X: Per la stringa di input: "12312845691"" nel Knowledge Center di AWS .
- I valori Null sono presenti in un campo numerico intero. Una soluzione alternativa è creare la colonna con i valori nulli come`string` e quindi utilizzare `CAST` per convertire il campo in una query, fornendo un valore predefinito `0` per i valori nulli. Per ulteriori informazioni, consulta Quando eseguo query sui dati CSV in Athena, ricevo l'errore "HIVE_BAD_DATA: Error parsing field value " for field X: For input string: """ nel Portale del sapere AWS .

HIVE_CANNOT_OPEN_SPLIT: errore durante l'apertura di Hive split s3://DOC-EXAMPLE-BUCKET

Questo errore può verificarsi quando si esegue una query su un prefisso bucket Amazon S3 con un numero elevato di oggetti. Per ulteriori informazioni, vedi Come posso risolvere l'errore «HIVE_CANNOT_OPEN_SPLIT: errore nell'apertura di Hive split s3://DOC-EXAMPLE-BUCKET/: Slow down» in Athena? nel Knowledge Center. AWS

HIVE_CURSOR_ERROR: com.amazonaws.services.s3.model.AmazonS3Exception: The specified key does not exist

Questo errore si verifica in genere quando un file viene rimosso quando una query è in esecuzione. Eseguire di nuovo la query o controllare il flusso di lavoro per verificare se un altro processo o processo sta modificando i file durante l'esecuzione della query.

HIVE_CURSOR_ERROR: Unexpected end of input stream

Questo messaggio indica che il file è danneggiato o vuoto. Verificare l'integrità del file ed eseguire di nuovo la query.

HIVE_FILESYSTEM_ERROR: Dimensione file non corretta ***1234567*** per il file

Questo messaggio viene visualizzato quando un file cambia tra la pianificazione e l'esecuzione della query. Di solito ciò si verifica quando un file su Amazon S3 viene sostituito sul posto (ad esempio, viene eseguito PUT su una chiave in cui esiste già un oggetto). Athena non supporta l'eliminazione o la sostituzione del contenuto di un file durante l'esecuzione di una query. Per evitare questo errore, pianificare i processi che sovrascrivono o eliminano i file quando le query non vengono eseguite o scrivono solo dati su nuovi file o partizioni.

HIVE_UNKNOWN_ERROR: Impossibile creare il formato di input

Questo errore può essere dovuto a problemi di questo tipo:

- Il AWS Glue crawler non è stato in grado di classificare il formato dei dati
- Alcune proprietà di definizione AWS Glue delle tabelle sono vuote
- Athena non supporta il formato dati dei file in Amazon S3

Per ulteriori informazioni, consulta Come faccio a risolvere l'errore "Impossibile creare il formato di input" in Athena? nel Portale del sapere AWS o guarda il video nel Knowledge Center.

Il percorso S3 fornito per salvare i risultati della query non è valido.

Verificare che sia stato specificato un percorso S3 valido per i risultati della query. Per ulteriori informazioni, consulta Specificare una posizione dei risultati delle query nell'argomento Utilizzo dei risultati delle query, delle query recenti e dei file di output.

Tabelle Delta Lake di Linux Foundation

Lo schema della tabella Delta Lake non è sincronizzato

Quando esegui una query su una tabella Delta Lake AWS Glue che contiene uno schema obsoleto, puoi ricevere il seguente messaggio di errore:

```
INVALID_GLUE_SCHEMA: Delta Lake table schema in Glue does not match the most recent
 schema of the
Delta Lake transaction log. Please ensure that you have the correct schema defined in
 Glue.
```

Lo schema può diventare obsoleto se viene modificato AWS Glue dopo essere stato aggiunto ad Athena. Per aggiornare lo schema, completa una delle seguenti operazioni:

- Nel AWS Glue, esegui il crawler.AWS Glue
- In Athena, elimina la tabella e creala di nuovo.
- Aggiungi manualmente le colonne mancanti utilizzando l'istruzione ALTER TABLE ADD COLUMNS in Athena o modificando lo schema della tabella in AWS Glue.

Query federate

Timeout durante la chiamata ListTableMetadata

Una chiamata all'ListTableMetadataAPI può scadere se ci sono molte tabelle nell'origine dati, se l'origine dati è lenta o se la rete è lenta. Per risolvere questo problema, provare a eseguire le seguenti operazioni.

- Controlla il numero di tabelle: se hai più di 1.000 tabelle, prova a ridurne il numero. Per una risposta `ListTableMetadata` più rapida, è consigliabile avere meno di 1000 tabelle per catalogo.
- Controlla la configurazione Lambda: il monitoraggio del comportamento della funzione Lambda è fondamentale. Quando utilizzi cataloghi federati, assicurati di esaminare i log di esecuzione della funzione Lambda. In base ai risultati, regola di conseguenza i valori di memoria e timeout. Per identificare eventuali problemi con i timeout, riesamina la configurazione Lambda. Per ulteriori informazioni, consulta Configurazione timeout (console) di funzione nella Guida per gli sviluppatori di AWS Lambda .

- Controlla i log delle origini dati federate: esamina i log e i messaggi di errore delle origine dati federate per vedere se ci sono problemi o errori. I log possono fornire informazioni preziose sulla causa del timeout.
- Utilizza **`StartQueryExecution`** per recuperare i metadati: se disponi di più di 1.000 tabelle, il recupero dei metadati tramite il connettore federato può richiedere più tempo del previsto. Poiché la natura asincrona di garantisce StartQueryExecutionche Athena esegua la query nel modo più ottimale, prendi in considerazione l'utilizzo `StartQueryExecution` come alternativa a. `ListTableMetadata` AWS CLI Gli esempi seguenti mostrano come `StartQueryExecution` può essere utilizzato anziché per `ListTableMetadata` ottenere tutti i metadati delle tabelle nel catalogo dati.

  Innanzitutto, esegui una query che ottenga tutte le tabelle, come nell'esempio seguente.

```
aws athena start-query-execution --region us-east-1 \
--query-string "SELECT table_name FROM information_schema.tables LIMIT 50" \
--work-group "your-work-group-name"
```

  Quindi, recupera i metadati di una singola tabella, come nell'esempio seguente.

```
aws athena start-query-execution --region us-east-1 \
--query-string "SELECT * FROM information_schema.columns \
WHERE table_name = 'your-table-name' AND \
table_catalog = 'your-catalog-name'" \
--work-group "your-work-group-name"
```

  Il tempo impiegato per ottenere i risultati dipende dal numero di tabelle nel catalogo.

Per ulteriori informazioni sulla risoluzione dei problemi delle query federate, vedere Common_Problems nella aws-athena-query-federation sezione awslabs/ di o consultare la documentazione per i singoli GitHub connettori di origine dati Athena.

Errori correlati a JSON

Errori di dati NULL o errati durante il tentativo di leggere i dati JSON

Gli errori di dati NULL o errati quando provi a leggere i dati JSON possono dipendere da molteplici cause diverse. Per identificare le righe che causano errori durante l'utilizzo di OpenX SerDe, impostate `suignore.malformed.json`. `true` I registri non corretti verranno restituiti come NULL.

Per ulteriori informazioni, consulta Ricevo messaggi di errore quando cerco di leggere dati JSON su Amazon Athena nel Portale del sapere AWS o guarda il video nel Portale del sapere.

HIVE_BAD_DATA: Errore durante l'analisi del valore del campo per il campo 0: impossibile eseguore il casting di java.lang.String su org.openx.data.jsonserde.json.JSONObject

OpenX JSON SerDe genera questo errore quando non riesce a analizzare una colonna in una query Athena. Ciò può accadere se definisci una colonna come `map` o `struct`, ma i dati sottostanti sono in realtà `string`, `int` o altro tipo primitivo.

HIVE_CURSOR_ERROR: Row is not a valid JSON Object - JSONException: Duplicate key

Questo errore si verifica quando si utilizza Athena per interrogare AWS Config risorse che hanno più tag con lo stesso nome in maiuscole e minuscole. La soluzione consiste nell'eseguire `CREATE TABLE` utilizzando `WITH SERDEPROPERTIES 'case.insensitive'='false'` e mappare quindi i nomi. Per ulteriori informazioni su `case.insensitive` e sulla mappatura, consulta Librerie JSON SerDe . Per ulteriori informazioni, vedi Come posso risolvere «HIVE_CURSOR_ERROR: La riga non è un oggetto JSON valido - JsonException: chiave duplicata» durante la lettura di file da Athena? AWS Config AWS nel Knowledge Center.

Messaggi HIVE_CURSOR_ERROR con JSON formattato

Le librerie JSON Hive SerDe e OpenX JSON SerDe prevedono che ogni documento JSON sia su una singola riga di testo senza caratteri di terminazione riga che separano i campi nel registro. Se il testo JSON è in un bel formato di stampa, potresti ricevere un messaggio di errore del tipo HIVE_CURSOR_ERROR: Row is not a valid JSON Object o HIVE_CURSOR_ERROR:: Unexpected JsonParseException end-of-input: il marker di chiusura previsto per OBJECT quando tenti di interrogare la tabella dopo averla creata. Per ulteriori informazioni, consulta i file di dati JSON nella documentazione di SerDe OpenX su. GitHub

Più registri JSON restituiscono un SELECT COUNT di 1

Se usi OpenX JSON SerDe, assicurati che i registri siano separati da un carattere di a capo. Per ulteriori informazioni, consulta La query SELECT COUNT in Amazon Athena restituisce un solo record anche se il file JSON di input ha più record nel AWS Knowledge Center.

Impossibile interrogare una tabella creata da un AWS Glue crawler che utilizza un classificatore JSON personalizzato

Il motore Athena non supporta i classificatori JSON personalizzati. Per risolvere il problema, è necessario creare una nuova tabella senza classificatore personalizzato. Per trasformare il JSON, è possibile utilizzare CTAS o creare una vista. Ad esempio, se si lavora con gli array, è possibile utilizzare l'opzione UNNEST per appiattire il JSON. Un'altra opzione è utilizzare un processo AWS Glue ETL che supporti il classificatore personalizzato, convertire i dati in parquet in Amazon S3 e quindi interrogarli in Athena.

MSCK REPAIR TABLE

Per informazioni sui problemi relativi a MSCK REPAIR TABLE, consulta le sezioni Considerazioni e limitazioni e Risoluzione dei problemi nella pagina MSCK REPAIR TABLE.

Problemi con l'output

Impossibile verificare/creare il bucket di output

Questo errore può verificarsi se il percorso del risultato della query specificato non esiste o se le autorizzazioni corrette non sono presenti. Per ulteriori informazioni, consulta Come posso risolvere l'errore «impossibile verificare/creare un bucket di output» in Amazon Athena? nel Knowledge Center. AWS

Il risultato TIMESTAMP è vuoto

Athena richiede il formato Java TIMESTAMP. Per ulteriori informazioni, consulta Quando eseguo una query su una tabella in Amazon Athena, il risultato TIMESTAMP è vuoto nel Portale del sapere di AWS .

Archivia l'output della query Athena in un formato diverso da CSV

Per impostazione predefinita, Athena restituisce file solo in formato CSV. Per produrre i risultati di una query `SELECT` in un formato diverso, è possibile utilizzare l'istruzione `UNLOAD`. Per ulteriori informazioni, consulta UNLOAD. Per configurare il formato di output, è possibile utilizzare anche una query CTAS che utilizzi la proprietà della tabella `format`. A differenza di `UNLOAD`, la tecnica CTAS richiede la creazione di una tabella. Per ulteriori informazioni, consulta Come posso archiviare l'output di una query Athena in un formato diverso da CSV, ad esempio un formato compresso? nel Knowledge Center. AWS

Il percorso S3 fornito per salvare i risultati della query non è valido

È possibile ricevere questo messaggio di errore se la posizione del bucket di output non si trova nella stessa Regione della Regione in cui si esegue la query. Per evitare ciò, specificare una posizione dei risultati della query nella Regione in cui si esegue la query. Per le fasi, consulta Specificare una posizione dei risultati delle query.

Problemi con il parquet

org.apache.parquet.io. GroupColumnL'IO non può essere trasmesso a org.apache.parquet.io. PrimitiveColumnIO

Questo errore è causato da una mancata corrispondenza dello schema Parquet. Una colonna con un tipo non primitivo (ad esempio `array`) è stato dichiarata come tipo primitivo (ad esempio, `string`) in AWS Glue. Per risolvere questo problema, controllare lo schema dei dati nei file e confrontarlo con lo schema dichiarato in AWS Glue.

Problemi relativi alle statistiche del parquet

Quando leggi i dati di Parquet, potresti ricevere messaggi di errore simili a quelli che seguono:

```
HIVE_CANNOT_OPEN_SPLIT: Index x out of bounds for length y
HIVE_CURSOR_ERROR: Failed to read x bytes
HIVE_CURSOR_ERROR: FailureException at Malformed input: offset=x
HIVE_CURSOR_ERROR: FailureException at java.io.IOException:
can not read class org.apache.parquet.format.PageHeader: Socket is closed by peer.
```

Per risolvere questo problema, utilizzate l'ALTER TABLE SET TBLPROPERTIESistruzione CREATE TABLE or per impostare la SerDe `parquet.ignore.statistics` proprietà Parquet su`true`, come illustrato negli esempi seguenti.

Esempio di CREATE TABLE

```
...
ROW FORMAT SERDE
'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe'
WITH SERDEPROPERTIES ('parquet.ignore.statistics'='true')
STORED AS PARQUET
...
```

Esempio di ALTER TABLE

```
ALTER TABLE ... SET TBLPROPERTIES ('parquet.ignore.statistics'='true')
```

Per ulteriori informazioni su Parquet Hive SerDe, vedere. Parquet SerDe

Problemi di partizionamento

MSCK REPAIR TABLE non rimuove le partizioni obsolete

Se elimini manualmente una partizione in Amazon S3 e poi esegui MSCK REPAIR TABLE, potresti ricevere il messaggio di errore Partizioni mancanti dal file system. Ciò si verifica perché MSCK REPAIR TABLE non rimuove le partizioni obsolete dai metadati della tabella. Utilizza ALTER TABLE DROP PARTITION per rimuovere manualmente le partizioni obsolete. Per ulteriori informazioni, consulta la sezione "Risoluzione dei problemi" dell'argomento MSCK REPAIR TABLE.

Errore MSCK REPAIR TABLE

Quando una grande quantità di partizioni (ad esempio, più di 100.000) è associata a una tabella particolare, `MSCK REPAIR TABLE` può avere esito negativo a causa delle limitazioni della memoria. Per risolvere il limite, usa invece ALTER TABLE ADD PARTITION.

MSCK REPAIR TABLE rileva le partizioni ma non le aggiunge a AWS Glue

Questo problema può verificarsi se un percorso di Amazon S3 è in maiuscolo anziché in minuscolo o una policy IAM non consente l'azione `glue:BatchCreatePartition`. Per ulteriori informazioni, vedere MSCK REPAIR TABLE rileva le partizioni in Athena ma non le aggiunge nel Knowledge Center. AWS Glue Data Catalog AWS

Gli intervalli di proiezione delle partizioni con il formato data di gg-MM-aaaa-HH-mm-ss o aaaa-MM-gg non funzionano

Per funzionare correttamente, il formato della data deve essere impostato su `yyyy-MM-dd HH:00:00`. Per ulteriori informazioni, consulta il post Stack Overflow La proiezione della partizione Athena non funziona come previsto.

PARTITION BY non supporta il tipo BIGINT

Converti il tipo di dati in `string` e riprova.

Nessuna partizione significativa disponibile

Questo messaggio di errore in genere indica che le impostazioni della partizione sono state danneggiate. Per risolvere questo problema, eliminare la tabella e creare una tabella con nuove partizioni.

La proiezione delle partizioni non funziona in combinazione con le partizioni di intervallo

Verificare che l'unità dell'intervallo di tempo projection.*<columnName>*.interval.unit corrisponda al separatore per le partizioni. Ad esempio, se le partizioni sono delimitate da giorni, l'unità di un intervallo di ore non funzionerà.

Errore di proiezione della partizione quando l'intervallo è specificato da un trattino

Specificando la proprietà della tabella `range` con un trattino anziché con una virgola ottinei un errore del tipo INVALID_TABLE_PROPERTY: per la stringa di input: "*number-number*". Assicuratevi che i valori dell'intervallo siano separati da una virgola, non da un trattino. Per ulteriori informazioni, consulta Tipo intero.

HIVE_UNKNOWN_ERROR: Impossibile creare il formato di input

Una o più partizioni Glue sono dichiarate in un formato diverso poiché ogni partizione Glue ha il proprio formato di input specifico in modo indipendente. Controllate come sono definite le vostre partizioni in. AWS Glue

HIVE_PARTITION_SCHEMA_MISMATCH

Se lo schema di una partizione differisce dallo schema della tabella, una query può dare esito negativo con il messaggio di errore HIVE_PARTITION_SCHEMA_MISMATCH. Per ulteriori informazioni, consulta Sincronizzazione dello schema di partizione per evitare "HIVE_PARTITION_SCHEMA_MISMATCH".

SemanticException la tabella non è partizionata ma esistono le specifiche della partizione

Questo errore può verificarsi in presenza di partizioni definite nell'istruzione `CREATE TABLE`. Per ulteriori informazioni, vedi Come posso risolvere l'errore «FAILED: la SemanticException tabella non è partizionata ma le specifiche della partizione esistono» in Athena? AWS nel Knowledge Center.

File **_$folder$** di zero byte

Se esegui un'istruzione `ALTER TABLE ADD PARTITION` e specifichi erroneamente una partizione già esistente e una posizione Amazon S3 errata, vengono creati in Amazon S3 i file segnaposto di zero byte del formato *`partition_value`*`_$folder$`. È necessario rimuovere questi file manualmente.

Per evitare che ciò accada, utilizzare la sintassi `ADD IF NOT EXISTS` nell'istruzione `ALTER TABLE ADD PARTITION`, nel modo seguente:

```
ALTER TABLE table_name ADD IF NOT EXISTS PARTITIION […]
```

Zero registri restituiti dai dati partizionati

Questo problema può verificarsi per una serie di motivi. Per possibili cause e soluzioni, vedi Ho creato una tabella in Amazon Athena con partizioni definite, ma quando eseguo una query sulla tabella, nel AWS Knowledge Center non vengono restituiti record.

Consulta anche HIVE_TOO_MANY_OPEN_PARTITIONS.

Autorizzazioni

Errore di accesso negato durante l'esecuzione di query su Amazon S3

Ciò può verificarsi quando non si dispone dell'autorizzazione per leggere i dati nel bucket o l'autorizzazione a scrivere nel bucket dei risultati oppure il percorso Amazon S3 contiene un endpoint della Regione come `us-east-1.amazonaws.com`. Per ulteriori informazioni, consulta Quando eseguo una query Athena, ottengo un errore "Accesso negato" nel Portale del sapere AWS .

Accesso negato con codice di stato "Errore 403" durante l'esecuzione di query DDL su dati crittografati in Amazon S3

Quando potresti ricevere il messaggio di errore Accesso negato (Servizio: Amazon S3; Codice di stato: 403; Codice di errore: AccessDenied; ID richiesta:) <request_id>se sono vere le seguenti condizioni:

1. Si esegue una query DDL come `ALTER TABLE ADD PARTITION` o `MSCK REPAIR TABLE`.
2. Hai un bucket con crittografia predefinita configurata per utilizzare `SSE-S3`.

3. Il bucket ha anche una policy del bucket come la seguente che forza le richieste `PutObject` per indicare gli header PUT come `"s3:x-amz-server-side-encryption": "true"` e `"s3:x-amz-server-side-encryption": "AES256"`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::<resource-name>/*",
            "Condition": {
                "Null": {
                    "s3:x-amz-server-side-encryption": "true"
                }
            }
        },
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::<resource-name>/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-server-side-encryption": "AES256"
                }
            }
        }
    ]
}
```

In un caso come questo, la soluzione consigliata è rimuovere la policy del bucket come quella sopra, dato che la crittografia predefinita del bucket è già presente.

Accesso negato con codice di stato 403 quando si esegue una query su un bucket Amazon S3 in un altro account

Questo errore può verificarsi quando si tenta di interrogare i log scritti da un altro account Servizio AWS e il secondo account è il proprietario del bucket ma non possiede gli oggetti nel bucket. Per ulteriori informazioni, consulta Ricevo l'eccezione Amazon S3 «accesso negato con codice di stato:

403" in Amazon Athena quando eseguo una query su un bucket in un altro account nel Knowledge Center o guardo il video del AWS Knowledge Center.

Utilizzare le credenziali del ruolo IAM per connettersi al driver Athena JDBC

È possibile recuperare le credenziali temporanee di un ruolo per autenticare la connessione JDBC ad Athena. Le credenziali temporanee hanno una durata massima di 12 ore. Per ulteriori informazioni, vedi Come posso usare le mie credenziali del ruolo IAM o passare a un altro ruolo IAM quando mi connetto ad Athena utilizzando il driver JDBC? nel Knowledge Center. AWS

Problemi sintassi delle query

FALLITO: NullPointerException il nome è nullo

Se si utilizza l'operazione AWS Glue CreateTableAPI o il AWS CloudFormation `AWS::Glue::Table`modello per creare una tabella da utilizzare in Athena senza specificare la `TableType` proprietà e quindi si esegue una query DDL come `SHOW CREATE TABLE` o`MSCK REPAIR TABLE`, è possibile ricevere il messaggio di errore FAILED: NullPointerException Name is null.

Per risolvere l'errore, specifica un valore per l'TableInput`TableType`attributo come parte della chiamata o del modello AWS Glue`CreateTable` API.AWS CloudFormation I valori possibili per `TableType` includono `EXTERNAL_TABLE` o `VIRTUAL_VIEW`.

Questo requisito si applica solo quando si crea una tabella utilizzando l'operazione AWS Glue `CreateTable` API o il `AWS::Glue::Table` modello. Se si crea una tabella per Athena utilizzando un'istruzione DDL o un crawler AWS Glue , la proprietà `TableType` viene definita automaticamente.

Funzione non registrata

Questo errore si verifica quando si tenta di utilizzare una funzione che Athena non supporta. Per un elenco delle funzioni supportate da Athena, consulta Funzioni in Amazon Athena oppure esegui l'istruzione `SHOW FUNCTIONS` nell'editor di query. Puoi anche scrivere la tua funzione definita dall'utente (UDF). Per ulteriori informazioni, consulta Come faccio a risolvere l'errore di sintassi "funzione non registrata" in Athena? nel Portale del sapere AWS .

Eccezioni GENERIC_INTERNAL_ERROR

Le eccezioni `GENERIC_INTERNAL_ERROR` possono avere diverse cause, tra cui le seguenti:

- GENERIC_INTERNAL_ERROR: null (GENERIC_INTERNAL_ERROR: nullo): questa eccezione potrebbe essere visualizzata in una delle condizioni seguenti:
  - È presente una mancata corrispondenza dello schema tra il tipo di dati di una colonna nella definizione di tabella e il tipo di dati effettivo del set di dati.
  - Si sta eseguendo una query `CREATE TABLE AS SELECT` (CTAS) con sintassi imprecisa.
- GENERIC_INTERNAL_ERROR: il generatore principale è nullo: potresti vedere questa eccezione quando esegui una query su una tabella con colonne di tipo di dati `array` e stai utilizzando la libreria OpenCSV. SerDe Il formato OpenCSVSerde non supporta il tipo di dati `array`.
- GENERIC_INTERNAL_ERROR: Value exceeds MAX_INT (GENERIC_INTERNAL_ERROR: il valore supera MAX_INT): questa eccezione può essere visualizzata quando la colonna dei dati di origine è definita con il tipo di dati `INT` e ha un valore numerico superiore a 2.147.483.647.
- GENERIC_INTERNAL_ERROR: Value exceeds MAX_BYTE (GENERIC_INTERNAL_ERROR: il valore supera MAX_BYTE): questa eccezione può essere visualizzata quando la colonna dei dati di origine ha un valore numerico che supera la dimensione consentita per il tipo di dati `BYTE`. Il tipo di dati `BYTE` equivale a `TINYINT`. `TINYINT` è un valore intero firmato a 8 bit in formato a due complementi con un valore minimo pari a -128 e un valore massimo pari a 127.
- GENERIC_INTERNAL_ERROR: Number of partition values does not match number of filters (GENERIC_INTERNAL_ERROR: il numero di valori di partizione non corrisponde al numero di filtri): questa eccezione potrebbe essere visualizzata in caso di partizioni non coerenti su dati Amazon Simple Storage Service (Amazon S3). È possibile che vi siano partizioni non coerenti in una delle seguenti condizioni:
  - Le partizioni su Amazon S3 sono cambiate (ad esempio, sono state aggiunte nuove partizioni).
  - Il numero di colonne di partizione nella tabella non corrisponde a quello dei metadati della partizione.

Per informazioni più dettagliate su ciascuno di questi errori, vedi Come posso risolvere l'errore «GENERIC_INTERNAL_ERROR» quando eseguo una query su una tabella in Amazon Athena? AWS nel Knowledge Center.

Il numero di gruppi corrispondenti non corrisponde al numero di colonne

Questo errore si verifica quando si utilizza Regex SerDe in un'istruzione CREATE TABLE e il numero di gruppi corrispondenti regex non corrisponde al numero di colonne specificate per la tabella. Per ulteriori informazioni, consulta Come posso risolvere l' RegexSerDe errore «il numero di gruppi

corrispondenti non corrisponde al numero di colonne» in Amazon Athena? nel AWS Knowledge Center.

queryString non è riuscito a soddisfare il vincolo: il membro deve avere una lunghezza inferiore o uguale a 262144

La lunghezza massima della stringa di query in Athena (262.144 byte) non è una quota regolabile. AWS Support non puoi aumentare la quota per te, ma puoi risolvere il problema suddividendo le query lunghe in query più piccole. Per ulteriori informazioni, consulta Come faccio ad aumentare la lunghezza massima della stringa di query in Athena? nel Portale del sapere di AWS .

SYNTAX_ERROR: Impossibile risolvere la colonna

Questo errore può verificarsi quando si esegue una query su una tabella creata da un AWS Glue crawler a partire da un file CSV con codifica UTF-8 e dotato di un byte order mark (BOM). AWS Glue non riconosce i BOM e li trasforma in punti interrogativi, che Amazon Athena non riconosce. La soluzione è rimuovere il punto interrogativo in Athena o in AWS Glue.

Troppi argomenti per la chiamata di funzione

Nella versione 3 del motore Athena, le funzioni non possono accettare più di 127 argomenti. Questa limitazione è di progettazione. Se si utilizza una funzione con più di 127 parametri, viene visualizzato un messaggio di errore come il seguente:

TOO_MANY_ARGUMENTS: riga *nnn*:*nn*: Numero di argomenti eccessivo per la chiamata di funzione *function_name*().

Per risolvere questo problema, utilizzare un numero inferiore di parametri per chiamata di funzione.

Problemi di timeout delle query

Se riscontri errori di timeout con le tue query su Athena, controlla i log. CloudTrail Le query possono scadere a causa del throttling delle nostre API Lake AWS Glue Formation. Quando si verificano questi errori, i messaggi di errore corrispondenti possono indicare un problema di timeout delle query anziché un problema di limitazione (della larghezza di banda della rete). Per risolvere il problema, puoi controllare i log prima di contattarci. CloudTrail AWS Support Per ulteriori informazioni, consulta Interrogazione dei log AWS CloudTrail e Registrazione delle chiamate API Amazon Athena con AWS CloudTrail.

Per informazioni sui problemi di timeout delle query con le query federate quando si chiama l'API `ListTableMetadata`, consulta Timeout durante la chiamata ListTableMetadata.

Problemi di limitazione

Se le tue richieste superano i limiti dei servizi dipendenti come Amazon S3 AWS KMS,, AWS Lambda o AWS Glue, è possibile che vengano visualizzati i seguenti messaggi. Per risolvere questi problemi, ridurre il numero di chiamate simultanee provenienti dallo stesso account.

Servizio	Messaggio di errore
AWS Glue	AWSGlueException: Frequenza superata.
AWS KMS	Hai superato la frequenza con cui puoi chiamare KMS. Riduci la frequenza delle tue chiamate.
AWS Lambda	Velocità superata TooManyRequestsException
Amazon S3	Amazons3Exception: riduci la frequenza delle richieste.

Per informazioni sui modi per prevenire la limitazione (della larghezza di banda della rete) di Amazon S3 quando usi Athena, consulta Come previnire la limitazione (della larghezza di banda della rete) di Amazon S3.

Visualizzazioni

Le viste create nella struttura di Apache Hive non funzionano in Athena

A causa delle loro implementazioni fondamentalmente diverse, le viste create nella struttura di Apache Hive non sono compatibili con Athena. Per risolvere il problema, ricrea le viste in Athena.

La vista è obsoleta; deve essere ricreata

È possibile ricevere questo errore se la tabella che si trova sotto una vista è stata modificata o eliminata. La risoluzione consiste nel ricreare la vista. Per ulteriori informazioni, vedi Come posso risolvere l'errore «la vista è obsoleta; deve essere ricreata» in Athena? nel Knowledge Center. AWS

Gruppi di lavoro

Per informazioni sulla risoluzione dei problemi dei gruppi di lavoro, consulta la sezione Risoluzione dei problemi relativi ai gruppi di lavoro.

Risorse aggiuntive

Le pagine seguenti forniscono ulteriori informazioni per la risoluzione dei problemi relativi ad Amazon Athena.

- catalogo degli errori Athena
- Service Quotas (Quote di Servizio)
- Considerazioni e restrizioni per le query SQL in Amazon Athena
- DDL non supportato
- Nomi di tabelle, database e colonne
- Tipi di dati in Amazon Athena
- Formati di SerDe e di dati supportati
- Supporto della compressione in Athena
- Parole chiave riservate
- Risoluzione dei problemi relativi ai gruppi di lavoro

Anche le seguenti AWS risorse possono essere di aiuto:

- Argomenti di Athena nel centro di conoscenza AWS
- Domande su Amazon Athena su re:POST AWS
- Post su Athena nel blog AWS Big Data

La risoluzione dei problemi richiede spesso query iterative e scoperta da parte di un esperto o di una community di aiutanti. Se continui a riscontrare problemi dopo aver provato i suggerimenti in questa pagina, contatta AWS Support (nella sezione, fai clic su Support, Support Center) o fai una domanda su AWS re:post utilizzando il tag Amazon Athena. AWS Management Console

catalogo degli errori Athena

Athena fornisce informazioni di errore standardizzate per aiutarti a comprendere le query non riuscite e intraprendere le operazioni adatte a seguito di un errore di query. La funzione `AthenaError`

include un campo `ErrorCategory` e `ErrorType`. `ErrorCategory` specifica se la causa della query non riuscita è dovuta a errori di sistema, errori dell'utente o altri errori. `ErrorType` fornisce informazioni più granulari sull'origine dell'errore. Combinando i due campi, è possibile comprendere meglio le circostanze e le cause dell'errore specifico verificato.

Categorie di errori

La tabella seguente elenca i valori delle categorie di errori Athena e i relativi significati.

Categorie di errori	Origine
1	SYSTEM
2	UTENTE
3	OTHER

Documentazione di riferimento per i tipi di errori

La tabella seguente elenca i valori dei tipi di errore Athena e i relativi significati.

Tipi di errore	Descrizione
0	Interroga le risorse esauste con questo fattore di scala
1	Interroga le risorse esauste con questo fattore di scala
2	Interroga le risorse esauste con questo fattore di scala
3	Interroga le risorse esauste con questo fattore di scala
4	Interroga le risorse esauste con questo fattore di scala
5	Interroga le risorse esauste con questo fattore di scala
6	Interroga le risorse esauste con questo fattore di scala
7	Interroga le risorse esauste con questo fattore di scala

Tipi di errore	Descrizione
8	Interroga le risorse esauste con questo fattore di scala
100	Errore interno del servizio
200	Errore interno nel motore di query
201	Errore interno nel motore di query
202	Errore interno nel motore di query
203	Errore driver
204	Il metastore ha riscontrato un errore
205	Errore interno nel motore di query
206	Query scaduta
207	Errore interno nel motore di query
208	Errore interno nel motore di query
209	Annullamento della query non riuscito
210	Query scaduta
211	Errore interno nel motore di query
212	Errore interno nel motore di query
213	Errore interno nel motore di query
214	Errore interno nel motore di query
215	Errore interno nel motore di query
216	Errore interno nel motore di query
217	Errore interno nel motore di query

Tipi di errore	Descrizione
218	Errore interno nel motore di query
219	Errore interno nel motore di query
220	Errore interno nel motore di query
221	Errore interno nel motore di query
222	Errore interno nel motore di query
223	Errore interno nel motore di query
224	Errore interno nel motore di query
225	Errore interno nel motore di query
226	Errore interno nel motore di query
227	Errore interno nel motore di query
228	Errore interno nel motore di query
229	Errore interno nel motore di query
230	Errore interno nel motore di query
231	Errore interno nel motore di query
232	Errore interno nel motore di query
233	Errore di Iceberg
234	Errore di Lake Formation
235	Errore interno nel motore di query
236	Errore interno nel motore di query
237	Errore di serializzazione

Tipi di errore	Descrizione
238	Caricamento dei metadati in Amazon S3 non riuscito
239	Errore di persistenza generale
240	Impossibile inviare la query
300	Errore interno del servizio
301	Errore interno del servizio
302	Errore interno del servizio
303	Errore interno del servizio
400	Errore interno del servizio
401	Impossibile scrivere i risultati delle query in Amazon S3
402	Impossibile scrivere i risultati delle query in Amazon S3
1000	Errore dell'utente
1001	Errore di dati
1002	Errore di dati
1003	Attività DDL non riuscita
1004	Errore di schema
1005	Errore di serializzazione
1006	Errore di sintassi
1007	Errore di dati
1008	Query rifiutata
1009	Query non riuscita

Tipi di errore	Descrizione
1010	Errore interno del servizio
1011	Query annullata dall'utente
1012	Errore interno nel motore di query
1013	Errore interno nel motore di query
1014	Query annullata dall'utente
1100	Fornito argomento non valido
1101	Fornita proprietà non valida
1102	Errore interno nel motore di query
1103	Fornita proprietà della tabella non valida
1104	Errore interno nel motore di query
1105	Errore interno nel motore di query
1106	Fornito argomento della funzione non valido
1107	Visualizzazione non valida
1108	Registrazione della funzione non riuscita
1109	Fornito percorso Amazon S3 non trovato
1110	Fornita tabella o visualizzazione non esistente
1200	Query non supportata
1201	Fornito decoder non supportato
1202	Tipo di query non supportato
1300	Errore generale non trovato

Tipi di errore	Descrizione
1301	Entità generale non trovata
1302	File non trovato
1303	Fornita funzione o implementazione della funzione non trovata
1304	Errore interno nel motore di query
1305	Errore interno nel motore di query
1306	Bucket Amazon S3 non trovato
1307	Motore selezionato non trovato
1308	Errore interno nel motore di query
1400	Errori di limitazione
1401	Interrogazione non riuscita a causa della limitazione AWS Glue
1402	Query non riuscita a causa del numero eccessivo di versioni di tabella in AWS Glue
1403	Query non riuscita a causa della limitazione di Amazon S3
1404	Query non riuscita a causa della limitazione di Amazon Athena
1405	Query non riuscita a causa della limitazione di Amazon Athena
1406	Query non riuscita a causa della limitazione di Amazon Athena
1500	Errore di autorizzazione
1501	Errore di autorizzazione Amazon S3
1602	Limite di capacità riservato superato. Capacità insufficiente per eseguire questa query.
1700	Interrogazione non riuscita a causa di un'eccezione interna di Lake Formation
1701	Interrogazione non riuscita a causa di un'eccezione AWS Glue interna

Tipi di errore	Descrizione
9999	Errore interno del servizio

Esempi di codice

Gli esempi riportati in questo argomento utilizzano l'SDK per Java2.x come punto di partenza per la creazione di applicazioni Athena.

> **Note**
>
> Per informazioni sulla programmazione di Athena utilizzando AWS SDK specifici per altri linguaggi, consulta le seguenti risorse:
>
> - AWS Command Line Interface (`athena`)
> - AWS SDK for .NET (`Amazon.Athena.Model`)
> - AWS SDK for C++ (`Aws::Athena::AthenaClient`)
> - AWS SDK for Go (`athena`)
> - AWS SDK for JavaScript v3 () `AthenaClient`
> - AWS SDK for PHP 3.x () `Aws\Athena`
> - AWS SDK for Python (Boto3) (`Athena.Client`)
> - AWS SDK for Ruby v3 () `Aws::Athena::Client`

Per ulteriori informazioni sull'esecuzione degli esempi di codice Java in questa sezione, consulta il readme Java di Amazon Athena nel repository degli esempi di AWS codice su. GitHub Per il riferimento alla programmazione Java per Athena, vedere AthenaClientin. AWS SDK for Java 2.x

- Esempi di codice Java
  - Costanti
  - Creazione di un client per accedere ad Athena
  - Esecuzioni delle query
    - Avvio dell'esecuzione di query
    - Interruzione dell'esecuzione di query

        - Elenco di esecuzioni di query
    - Utilizzo delle query denominate
        - Creazione di una query denominata
        - Eliminazione di una query denominata
        - Elenco di esecuzioni di query

> **Note**
>
> In questi esempi, per le stringhe vengono utilizzate costanti (ad esempio, `ATHENA_SAMPLE_QUERY`) che sono definite in una dichiarazione di classe `ExampleConstants.java`. Sostituisci le costanti con le tue stringhe o le tue costanti definite personali.

Costanti

La classe `ExampleConstants.java` dimostra come eseguire le query su una tabella creata tramite il tutorial Nozioni di base in Athena.

```
package aws.example.athena;

public class ExampleConstants {

    public static final int CLIENT_EXECUTION_TIMEOUT = 100000;
    public static final String ATHENA_OUTPUT_BUCKET = "s3://bucketscott2"; // change
 the Amazon S3 bucket name to match
                                                                          // your
 environment
    // Demonstrates how to query a table with a comma-separated value (CSV) table.
    // For information, see
    // https://docs.aws.amazon.com/athena/latest/ug/work-with-data.html
    public static final String ATHENA_SAMPLE_QUERY = "SELECT * FROM scott2;"; // change
 the Query statement to match
                                                                            // your
 environment
    public static final long SLEEP_AMOUNT_IN_MS = 1000;
    public static final String ATHENA_DEFAULT_DATABASE = "mydatabase"; // change the
 database to match your database
```

```
}
```

Creazione di un client per accedere ad Athena

La classe `AthenaClientFactory.java` illustra come creare e configurare un client Amazon Athena.

```
package aws.example.athena;

import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.athena.AthenaClient;
import software.amazon.awssdk.services.athena.AthenaClientBuilder;

public class AthenaClientFactory {
    private final AthenaClientBuilder builder = AthenaClient.builder()
            .region(Region.US_WEST_2)
            .credentialsProvider(ProfileCredentialsProvider.create());

    public AthenaClient createClient() {
        return builder.build();
    }
}
```

Avvio dell'esecuzione di query

`StartQueryExample` mostra come inviare una query ad Athena per l'esecuzione, attendere che i risultati diventano disponibili e quindi elaborare i risultati.

```
package aws.example.athena;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.athena.AthenaClient;
import software.amazon.awssdk.services.athena.model.QueryExecutionContext;
import software.amazon.awssdk.services.athena.model.ResultConfiguration;
import software.amazon.awssdk.services.athena.model.StartQueryExecutionRequest;
import software.amazon.awssdk.services.athena.model.StartQueryExecutionResponse;
import software.amazon.awssdk.services.athena.model.AthenaException;
import software.amazon.awssdk.services.athena.model.GetQueryExecutionRequest;
import software.amazon.awssdk.services.athena.model.GetQueryExecutionResponse;
import software.amazon.awssdk.services.athena.model.QueryExecutionState;
import software.amazon.awssdk.services.athena.model.GetQueryResultsRequest;
```

```
import software.amazon.awssdk.services.athena.model.GetQueryResultsResponse;
import software.amazon.awssdk.services.athena.model.ColumnInfo;
import software.amazon.awssdk.services.athena.model.Row;
import software.amazon.awssdk.services.athena.model.Datum;
import software.amazon.awssdk.services.athena.paginators.GetQueryResultsIterable;
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 StartQueryExample {

    public static void main(String[] args) throws InterruptedException {
        AthenaClient athenaClient = AthenaClient.builder()
                .region(Region.US_WEST_2)
                .build();

        String queryExecutionId = submitAthenaQuery(athenaClient);
        waitForQueryToComplete(athenaClient, queryExecutionId);
        processResultRows(athenaClient, queryExecutionId);
        athenaClient.close();
    }

    // Submits a sample query to Amazon Athena and returns the execution ID of the
    // query.
    public static String submitAthenaQuery(AthenaClient athenaClient) {
        try {
            // The QueryExecutionContext allows us to set the database.
            QueryExecutionContext queryExecutionContext =
 QueryExecutionContext.builder()
                    .database(ExampleConstants.ATHENA_DEFAULT_DATABASE)
                    .build();

            // The result configuration specifies where the results of the query should
 go.
            ResultConfiguration resultConfiguration = ResultConfiguration.builder()
                    .outputLocation(ExampleConstants.ATHENA_OUTPUT_BUCKET)
                    .build();
```

```
            StartQueryExecutionRequest startQueryExecutionRequest =
StartQueryExecutionRequest.builder()
                    .queryString(ExampleConstants.ATHENA_SAMPLE_QUERY)
                    .queryExecutionContext(queryExecutionContext)
                    .resultConfiguration(resultConfiguration)
                    .build();

            StartQueryExecutionResponse startQueryExecutionResponse = athenaClient
                    .startQueryExecution(startQueryExecutionRequest);
            return startQueryExecutionResponse.queryExecutionId();

        } catch (AthenaException e) {
            e.printStackTrace();
            System.exit(1);
        }
        return "";
    }

    // Wait for an Amazon Athena query to complete, fail or to be cancelled.
    public static void waitForQueryToComplete(AthenaClient athenaClient, String
queryExecutionId)
            throws InterruptedException {
        GetQueryExecutionRequest getQueryExecutionRequest =
GetQueryExecutionRequest.builder()
                .queryExecutionId(queryExecutionId)
                .build();

        GetQueryExecutionResponse getQueryExecutionResponse;
        boolean isQueryStillRunning = true;
        while (isQueryStillRunning) {
            getQueryExecutionResponse =
athenaClient.getQueryExecution(getQueryExecutionRequest);
            String queryState =
getQueryExecutionResponse.queryExecution().status().state().toString();
            if (queryState.equals(QueryExecutionState.FAILED.toString())) {
                throw new RuntimeException(
                        "The Amazon Athena query failed to run with error message: " +
getQueryExecutionResponse
                                .queryExecution().status().stateChangeReason());
            } else if (queryState.equals(QueryExecutionState.CANCELLED.toString())) {
                throw new RuntimeException("The Amazon Athena query was cancelled.");
            } else if (queryState.equals(QueryExecutionState.SUCCEEDED.toString())) {
                isQueryStillRunning = false;
            } else {
```

```
                // Sleep an amount of time before retrying again.
                Thread.sleep(ExampleConstants.SLEEP_AMOUNT_IN_MS);
            }
            System.out.println("The current status is: " + queryState);
        }
    }

    // This code retrieves the results of a query
    public static void processResultRows(AthenaClient athenaClient, String
 queryExecutionId) {
        try {
            // Max Results can be set but if its not set,
            // it will choose the maximum page size.
            GetQueryResultsRequest getQueryResultsRequest =
 GetQueryResultsRequest.builder()
                    .queryExecutionId(queryExecutionId)
                    .build();

            GetQueryResultsIterable getQueryResultsResults = athenaClient
                    .getQueryResultsPaginator(getQueryResultsRequest);
            for (GetQueryResultsResponse result : getQueryResultsResults) {
                List<ColumnInfo> columnInfoList =
 result.resultSet().resultSetMetadata().columnInfo();
                List<Row> results = result.resultSet().rows();
                processRow(results, columnInfoList);
            }

        } catch (AthenaException e) {
            e.printStackTrace();
            System.exit(1);
        }
    }

    private static void processRow(List<Row> row, List<ColumnInfo> columnInfoList) {
        for (Row myRow : row) {
            List<Datum> allData = myRow.data();
            for (Datum data : allData) {
                System.out.println("The value of the column is " +
 data.varCharValue());
            }
        }
    }
}
```

Interruzione dell'esecuzione di query

La classe `StopQueryExecutionExample` esegue un esempio di query, interrompe immediatamente la query e controlla lo stato della query per verificare che sia stata annullata.

```
package aws.example.athena;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.athena.AthenaClient;
import software.amazon.awssdk.services.athena.model.StopQueryExecutionRequest;
import software.amazon.awssdk.services.athena.model.GetQueryExecutionRequest;
import software.amazon.awssdk.services.athena.model.GetQueryExecutionResponse;
import software.amazon.awssdk.services.athena.model.QueryExecutionState;
import software.amazon.awssdk.services.athena.model.AthenaException;
import software.amazon.awssdk.services.athena.model.QueryExecutionContext;
import software.amazon.awssdk.services.athena.model.ResultConfiguration;
import software.amazon.awssdk.services.athena.model.StartQueryExecutionRequest;
import software.amazon.awssdk.services.athena.model.StartQueryExecutionResponse;

/**
 * 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 StopQueryExecutionExample {
    public static void main(String[] args) {
        AthenaClient athenaClient = AthenaClient.builder()
                .region(Region.US_WEST_2)
                .build();

        String sampleQueryExecutionId = submitAthenaQuery(athenaClient);
        stopAthenaQuery(athenaClient, sampleQueryExecutionId);
        athenaClient.close();
    }

    public static void stopAthenaQuery(AthenaClient athenaClient, String
 sampleQueryExecutionId) {
        try {
            StopQueryExecutionRequest stopQueryExecutionRequest =
 StopQueryExecutionRequest.builder()
```

```
                    .queryExecutionId(sampleQueryExecutionId)
                    .build();

            athenaClient.stopQueryExecution(stopQueryExecutionRequest);
            GetQueryExecutionRequest getQueryExecutionRequest =
GetQueryExecutionRequest.builder()
                    .queryExecutionId(sampleQueryExecutionId)
                    .build();

            GetQueryExecutionResponse getQueryExecutionResponse = athenaClient
                    .getQueryExecution(getQueryExecutionRequest);
            if (getQueryExecutionResponse.queryExecution()
                    .status()
                    .state()
                    .equals(QueryExecutionState.CANCELLED)) {

                System.out.println("The Amazon Athena query has been cancelled!");
            }

        } catch (AthenaException e) {
            e.printStackTrace();
            System.exit(1);
        }
    }

    // Submits an example query and returns a query execution Id value
    public static String submitAthenaQuery(AthenaClient athenaClient) {
        try {
            QueryExecutionContext queryExecutionContext =
QueryExecutionContext.builder()
                    .database(ExampleConstants.ATHENA_DEFAULT_DATABASE)
                    .build();

            ResultConfiguration resultConfiguration = ResultConfiguration.builder()
                    .outputLocation(ExampleConstants.ATHENA_OUTPUT_BUCKET)
                    .build();

            StartQueryExecutionRequest startQueryExecutionRequest =
StartQueryExecutionRequest.builder()
                    .queryExecutionContext(queryExecutionContext)
                    .queryString(ExampleConstants.ATHENA_SAMPLE_QUERY)
                    .resultConfiguration(resultConfiguration).build();

            StartQueryExecutionResponse startQueryExecutionResponse = athenaClient
```

```
                    .startQueryExecution(startQueryExecutionRequest);
            return startQueryExecutionResponse.queryExecutionId();

        } catch (AthenaException e) {
            e.printStackTrace();
            System.exit(1);
        }
        return null;

    }
}
```

Elenco di esecuzioni di query

La classe `ListQueryExecutionsExample` mostra come ottenere un elenco di ID di esecuzione di query.

```
package aws.example.athena;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.athena.AthenaClient;
import software.amazon.awssdk.services.athena.model.AthenaException;
import software.amazon.awssdk.services.athena.model.ListQueryExecutionsRequest;
import software.amazon.awssdk.services.athena.model.ListQueryExecutionsResponse;
import software.amazon.awssdk.services.athena.paginators.ListQueryExecutionsIterable;
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 ListQueryExecutionsExample {
    public static void main(String[] args) {
        AthenaClient athenaClient = AthenaClient.builder()
                .region(Region.US_WEST_2)
                .build();

        listQueryIds(athenaClient);
        athenaClient.close();
```

```
    }

    public static void listQueryIds(AthenaClient athenaClient) {
        try {
            ListQueryExecutionsRequest listQueryExecutionsRequest =
 ListQueryExecutionsRequest.builder().build();
            ListQueryExecutionsIterable listQueryExecutionResponses = athenaClient
                    .listQueryExecutionsPaginator(listQueryExecutionsRequest);
            for (ListQueryExecutionsResponse listQueryExecutionResponse :
 listQueryExecutionResponses) {
                List<String> queryExecutionIds =
 listQueryExecutionResponse.queryExecutionIds();
                System.out.println("\n" + queryExecutionIds);
            }

        } catch (AthenaException e) {
            e.printStackTrace();
            System.exit(1);
        }
    }
}
```

Creazione di una query denominata

La classe `CreateNamedQueryExample` mostra come creare una query denominata.

```
package aws.example.athena;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.athena.AthenaClient;
import software.amazon.awssdk.services.athena.model.AthenaException;
import software.amazon.awssdk.services.athena.model.CreateNamedQueryRequest;

/**
 * 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 CreateNamedQueryExample {
```

```
    public static void main(String[] args) {
        final String USAGE = """

                Usage:
                    <name>

                Where:
                    name - the name of the Amazon Athena query.\s
                """;

        if (args.length != 1) {
            System.out.println(USAGE);
            System.exit(1);
        }

        String name = args[0];
        AthenaClient athenaClient = AthenaClient.builder()
                .region(Region.US_WEST_2)
                .build();

        createNamedQuery(athenaClient, name);
        athenaClient.close();
    }

    public static void createNamedQuery(AthenaClient athenaClient, String name) {
        try {
            // Create the named query request.
            CreateNamedQueryRequest createNamedQueryRequest =
CreateNamedQueryRequest.builder()
                    .database(ExampleConstants.ATHENA_DEFAULT_DATABASE)
                    .queryString(ExampleConstants.ATHENA_SAMPLE_QUERY)
                    .description("Sample Description")
                    .name(name)
                    .build();

            athenaClient.createNamedQuery(createNamedQueryRequest);
            System.out.println("Done");

        } catch (AthenaException e) {
            e.printStackTrace();
            System.exit(1);
        }
    }
```

```
}
```

Eliminazione di una query denominata

La classe `DeleteNamedQueryExample` mostra come eliminare una query denominata utilizzando l'ID di query denominata.

```
package aws.example.athena;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.athena.AthenaClient;
import software.amazon.awssdk.services.athena.model.DeleteNamedQueryRequest;
import software.amazon.awssdk.services.athena.model.AthenaException;
import software.amazon.awssdk.services.athena.model.CreateNamedQueryRequest;
import software.amazon.awssdk.services.athena.model.CreateNamedQueryResponse;

/**
 * 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 DeleteNamedQueryExample {
    public static void main(String[] args) {
        final String USAGE = """

                Usage:
                    <name>

                Where:
                    name - the name of the Amazon Athena query.\s
                """;

        if (args.length != 1) {
            System.out.println(USAGE);
            System.exit(1);
        }

        String name = args[0];
        AthenaClient athenaClient = AthenaClient.builder()
                .region(Region.US_WEST_2)
```

```
                .build();

        String sampleNamedQueryId = getNamedQueryId(athenaClient, name);
        deleteQueryName(athenaClient, sampleNamedQueryId);
        athenaClient.close();
    }

    public static void deleteQueryName(AthenaClient athenaClient, String
sampleNamedQueryId) {
        try {
            DeleteNamedQueryRequest deleteNamedQueryRequest =
DeleteNamedQueryRequest.builder()
                    .namedQueryId(sampleNamedQueryId)
                    .build();

            athenaClient.deleteNamedQuery(deleteNamedQueryRequest);

        } catch (AthenaException e) {
            e.printStackTrace();
            System.exit(1);
        }
    }

    public static String getNamedQueryId(AthenaClient athenaClient, String name) {
        try {
            CreateNamedQueryRequest createNamedQueryRequest =
CreateNamedQueryRequest.builder()
                    .database(ExampleConstants.ATHENA_DEFAULT_DATABASE)
                    .queryString(ExampleConstants.ATHENA_SAMPLE_QUERY)
                    .name(name)
                    .description("Sample description")
                    .build();

            CreateNamedQueryResponse createNamedQueryResponse =
athenaClient.createNamedQuery(createNamedQueryRequest);
            return createNamedQueryResponse.namedQueryId();

        } catch (AthenaException e) {
            e.printStackTrace();
            System.exit(1);
        }
        return null;
    }
```

```
}
```

Elenco di query denominate

La classe `ListNamedQueryExample` mostra come ottenere un elenco di ID di query denominate.

```
package aws.example.athena;

import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.athena.AthenaClient;
import software.amazon.awssdk.services.athena.model.AthenaException;
import software.amazon.awssdk.services.athena.model.ListNamedQueriesRequest;
import software.amazon.awssdk.services.athena.model.ListNamedQueriesResponse;
import software.amazon.awssdk.services.athena.paginators.ListNamedQueriesIterable;
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 ListNamedQueryExample {
    public static void main(String[] args) {
        AthenaClient athenaClient = AthenaClient.builder()
                .region(Region.US_WEST_2)
                .build();

        listNamedQueries(athenaClient);
        athenaClient.close();
    }

    public static void listNamedQueries(AthenaClient athenaClient) {
        try {
            ListNamedQueriesRequest listNamedQueriesRequest =
 ListNamedQueriesRequest.builder()
                    .build();

            ListNamedQueriesIterable listNamedQueriesResponses = athenaClient
                    .listNamedQueriesPaginator(listNamedQueriesRequest);
```

```
            for (ListNamedQueriesResponse listNamedQueriesResponse :
 listNamedQueriesResponses) {
                List<String> namedQueryIds = listNamedQueriesResponse.namedQueryIds();
                System.out.println(namedQueryIds);
            }

        } catch (AthenaException e) {
            e.printStackTrace();
            System.exit(1);
        }
    }
}
```

Utilizzo di Apache Spark in Amazon Athena

Amazon Athena facilita l'esecuzione di analisi e l'esplorazione dei dati in modo interattivo mediante Apache Spark senza la necessità di pianificare, configurare o gestire le risorse. Eseguire le applicazioni Apache Spark su Athena significa inviare il codice Spark per l'elaborazione e ricevere direttamente i risultati senza la necessità di configurazioni aggiuntive. Puoi utilizzare l'esperienza semplificata dei notebook nella console Amazon Athena per sviluppare applicazioni Apache Spark utilizzando Python o le API dei notebook Athena. Apache Spark su Amazon Athena è serverless e offre il dimensionamento automatico e on demand per l'elaborazione istantanea, in modo da far fronte ai cambiamenti dei volumi di dati e dei requisiti di elaborazione.

Amazon Athena offre le seguenti funzionalità:

- Utilizzo della console: invia le tue applicazioni Spark dalla console Amazon Athena.
- Scripting: crea ed esegui il debug di applicazioni Apache Spark in Python in modo rapido e interattivo.
- Dimensionamento dinamico: Amazon Athena determina automaticamente le risorse di elaborazione e memoria necessarie per eseguire un processo e dimensiona continuamente tali risorse di conseguenza fino ai massimi specificati. Questo dimensionamento dinamico riduce i costi senza influire sulla velocità.
- Esperienza del notebook: utilizza l'editor notebook Athena per creare, modificare ed eseguire calcoli utilizzando un'interfaccia familiare. I notebook Athena sono compatibili con i notebook Jupyter e contengono un elenco di celle che vengono eseguite in ordine sotto forma di calcoli. Il contenuto delle celle può includere codice, testo, Markdown, matematica, grafici e rich media.

Per ulteriori informazioni, consulta Esegui Spark SQL su Amazon Athena Spark ed Esplora il tuo data lake usando Amazon Athena per Apache Spark nel blog Big Data.AWS

Considerazioni e limitazioni

- Attualmente, Amazon Athena per Apache Spark è disponibile nelle seguenti Regioni AWS:
  - Asia Pacifico (Mumbai)
  - Asia Pacifico (Singapore)
  - Asia Pacifico (Sydney)
  - Asia Pacifico (Tokyo)

  - Europa (Francoforte)
  - Europa (Irlanda)
  - Stati Uniti orientali (Virginia settentrionale)
  - Stati Uniti orientali (Ohio)
  - US West (Oregon)
- AWS Lake Formation non è supportato.
- Le tabelle che utilizzano la proiezione delle partizioni non sono supportate.
- I gruppi di lavoro compatibili con Apache Spark possono utilizzare l'editor di notebook Athena, ma non l'editor di query Athena. Solo i gruppi di lavoro Athena SQL possono utilizzare l'editor di query Athena.
- Le query di visualizzazione su più motori non sono supportate. Athena per Spark non può eseguire query suulle viste create da Athena SQL. Poiché le viste per i due motori sono implementate in modo diverso, non sono compatibili per l'uso tra motori diversi.
- MLLib (libreria di machine learning Apache Spark) e il `pyspark.ml` pacchetto non sono supportati. Per un elenco delle librerie Python supportate, consulta la pagina Elenco delle librerie Python preinstallate.
- Al momento, non `pip install` è supportato nelle sessioni di Athena for Spark.
- È consentita una sola sessione attiva per notebook.
- Quando più utenti utilizzano la console per aprire una sessione esistente in un gruppo di lavoro, accedono allo stesso notebook. Per evitare confusione, apri solo le sessioni create da te.
- I domini di hosting per le applicazioni Apache Spark che potresti utilizzare con Amazon Athena (ad esempio, `analytics-gateway.us-east-1.amazonaws.com`) sono registrati nella Public Suffix List (PSL) di Internet. Se hai bisogno di impostare cookie sensibili nei tuoi domini, ti consigliamo di utilizzare i cookie con un prefisso `__Host-` per proteggere il tuo dominio dai tentativi di falsificazione delle richieste tra siti (CSRF). Per ulteriori informazioni, consulta la pagina Set-Cookie nella documentazione per gli sviluppatori di Mozilla.org.
- Per informazioni sulla risoluzione dei problemi relativi a notebook, sessioni e gruppi di lavoro Spark in Athena, consulta la pagina Risoluzione dei problemi di Athena per Spark.

Nozioni di base su Apache Spark su Amazon Athena

Per iniziare a utilizzare Apache Spark su Amazon Athena, devi prima di tutto creare un gruppo di lavoro abilitato a Spark. Dopo il passaggio al gruppo di lavoro, è possibile creare un notebook o

aprirne uno esistente. Quando apri un notebook in Athena, viene avviata automaticamente una nuova sessione ed è possibile utilizzare il notebook direttamente nell'editor notebook Athena.

> **Note**
>
> Assicurati di creare un gruppo di lavoro abilitato a Spark prima di tentare di creare un notebook.

Creazione di un gruppo di lavoro abilitato a Spark in Athena

Puoi utilizzare i gruppi di lavoro in Athena per raggruppare utenti, team, applicazioni o carichi di lavoro e per tenere traccia dei costi. Per utilizzare Apache Spark in Amazon Athena, crei un gruppo di lavoro Amazon Athena che utilizza un motore Spark.

> **Note**
>
> I gruppi di lavoro compatibili con Apache Spark possono utilizzare l'editor di notebook Athena, ma non l'editor di query Athena. Solo i gruppi di lavoro Athena SQL possono utilizzare l'editor di query Athena.

Creazione di un gruppo di lavoro abilitato a Spark in Athena

1. Apri la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Nel pannello di navigazione, seleziona Workgroups (Gruppi di lavoro).
4. Nella pagina Gruppi di lavoro scegli Crea gruppo di lavoro.
5. In Workgroup name (Nome del gruppo di lavoro), inserisci un nome per il tuo gruppo di lavoro Apache Spark.
6. (Facoltativo) Per Description (Descrizione), inserisci una descrizione per il livello.
7. In Analytics engine (Motore di analisi), scegli Apache Spark.

> **Note**
>
> Dopo avere creato un gruppo di lavoro, il tipo di motore di analisi del gruppo di lavoro non può essere modificato. Ad esempio, un gruppo di lavoro del motore Athena versione 3 non può essere modificato in un gruppo di lavoro PySpark del motore versione 3.

8. Ai fini di questo tutorial, seleziona Turn on example notebook (Attiva il notebook di esempio). Questa funzionalità opzionale aggiunge un notebook di esempio con lo stesso nome `example-notebook-`*`random_string`* al gruppo di lavoro e aggiunge le autorizzazioni AWS Glue relative che il notebook utilizza per creare, mostrare ed eliminare database e tabelle specifici nel tuo account e le autorizzazioni di lettura in Amazon S3 per il set di dati di esempio. Per visualizzare le autorizzazioni aggiunte, scegli View additional permissions details (Visualizza dettagli aggiuntivi sulle autorizzazioni).

> **Note**
>
> L'esecuzione del notebook di esempio potrebbe comportare dei costi aggiuntivi.

9. Per Additional configurations (Configurazioni aggiuntive), esegui le operazioni seguenti:
   - Utilizza l'impostazione Use defaults (Utilizza valori predefiniti). Questa opzione è l'impostazione predefinita e ti aiuta a iniziare a utilizzare il tuo gruppo di lavoro abilitato per Spark. Con questa opzione, Athena crea per tuo conto un ruolo IAM e una posizione dei risultati di calcolo in Amazon S3. Il nome del ruolo IAM e la posizione del bucket S3 da creare sono visualizzati nella casella sotto l'intestazione Additional configurations (Configurazioni aggiuntive).
   - Disabilita l'impostazione Use defaults (Utilizza valori predefiniti), quindi continua con i passaggi nella sezione Specifica di configurazioni personalizzate per il gruppo di lavoro per configurare manualmente il gruppo di lavoro.
10. (Facoltativo) Tags (Tag): utilizza questa opzione per aggiungere tag al gruppo di lavoro. Per ulteriori informazioni, consulta Assegnazione di tag alle risorse Athena.
11. Selezionare Create workgroup (Crea gruppo di lavoro). Un messaggio informa che il gruppo di lavoro è stato creato correttamente e il gruppo di lavoro viene visualizzato nell'elenco dei gruppi di lavoro.

Specifica di configurazioni personalizzate per il gruppo di lavoro

Se desideri specificare un ruolo IAM e una posizione dei risultati del calcolo personalizzati per il notebook, segui i passaggi illustrati in questa sezione. Se hai scelto Use defaults (Utilizza valori predefiniti) per l'opzione Additional configurations (Configurazioni aggiuntive), salta questa sezione e vai direttamente alla sezione Apertura di Notebook explorer e cambio del gruppo di lavoro .

La procedura seguente presuppone che siano stati completati i passaggi da 1 a 9 della procedura Creazione di un gruppo di lavoro abilitato a Spark in Athena della sezione precedente.

Definizione di configurazioni personalizzate per il gruppo di lavoro

1. Se desideri creare o utilizzare un ruolo IAM personalizzato o configurare la crittografia dei notebook, espandi IAM role configuration (Configurazione del ruolo IAM).
   - Per Service Role (Ruolo di servizio), procedi come segue:

- Create a service role (Crea un ruolo di servizio): scegli questa opzione affinché Athena crei un ruolo di servizio per tuo conto. Per visualizzare le autorizzazioni concesse dal ruolo, scegli View permission details (Visualizza i dettagli delle autorizzazioni).
- Choose an existing service role (Scegli un ruolo di servizio esistente): dall'elenco a discesa, scegli un ruolo esistente. Il ruolo scelto deve includere le autorizzazioni nella prima opzione. Per ulteriori informazioni sulle autorizzazioni per i gruppi di lavoro abilitati per i notebook, consulta la pagina Risoluzione dei problemi relativi ai gruppi di lavoro abilitati per Spark.

- In Notebook and calculation code encryption key management (Gestione delle chiavi di crittografia del codice di calcolo), scegli una delle opzioni seguenti:
  - Di proprietà di Amazon Athena: la AWS KMS chiave è di proprietà e gestita da Amazon Athena. Non ti viene addebitato alcun costo aggiuntivo per l'utilizzo di questa chiave.
  - A symmetric key stored in your account, owned and managed by you (Una chiave simmetrica memorizzata nel tuo account, di tua proprietà e gestita da te): per questa opzione, esegui una delle seguenti operazioni:
    - Per utilizzare una chiave esistente, usa la casella di ricerca per scegliere AWS KMS o immettere una chiave ARN.
    - Per creare una chiave nella AWS KMS console, scegli Crea una AWS KMS chiave. Il ruolo di esecuzione deve disporre dell'autorizzazione a utilizzare la chiave che crei.

> **Important**
>
> Quando modifichi la AWS KMS key per un gruppo di lavoro, i notebook gestiti prima dell'aggiornamento continuano a fare riferimento alla vecchia chiave KMS. I notebook gestiti dopo l'aggiornamento utilizzano la nuova chiave KMS. Per aggiornare i vecchi notebook in modo che facciano riferimento alla nuova chiave KMS, esporta e quindi importa ciascuno dei vecchi notebook. Se elimini la vecchia chiave KMS prima di aggiornare i riferimenti dei vecchi notebook alla nuova chiave KMS, i vecchi notebook non saranno più decifrabili e non potranno essere recuperati.
> Questo comportamento si applica anche agli aggiornamenti degli alias, che sono nomi semplici per le chiavi KMS. Quando si aggiorna un alias di chiave KMS in modo che punti a una nuova chiave KMS, i notebook gestiti prima dell'aggiornamento dell'alias fanno ancora riferimento alla vecchia chiave KMS, mentre i notebook gestiti dopo l'aggiornamento dell'alias utilizzano la nuova chiave KMS. Tieni a mente questi aspetti prima di aggiornare le chiavi o gli alias KMS.

2. Se desideri specificare delle impostazioni personalizzate per i risultati del calcolo, espandi Calculation result settings (Impostazioni dei risultati del calcolo) e scegli una delle seguenti opzioni.

   - Create a new S3 bucket (Crea un nuovo bucket S3): questa opzione crea un bucket Amazon S3 nel tuo account per i risultati dei calcoli. Il nome del bucket ha il formato *account_id*-*region*-athena-results-bucket-*alphanumeric_id* e utilizza le impostazioni: ACL disabilitate, accesso pubblico bloccato, controllo delle versioni disabilitato e proprietario del bucket applicato.
   - Choose an existing S3 location (Scegli una posizione S3 esistente): per questa opzione, procedi come segue:
     - Inserisci il percorso S3 di una posizione esistente nella casella di ricerca o scegli Browse S3 (Sfoglia S3) per selezionare un bucket da un elenco.

       > **Note**
       >
       > Quando selezioni una posizione esistente in Amazon S3, non aggiungere una barra (/) alla posizione. In questo modo, il collegamento alla posizione dei risultati del calcolo nella pagina dei dettagli del calcolo punta alla directory errata. In tal caso, modifica la posizione dei risultati del gruppo di lavoro rimuovendo la barra finale.

     - (Facoltativo) Scegli View (Visualizza) per aprire la pagina Buckets (Bucket) della console Amazon S3, dove puoi trovare ulteriori informazioni sul bucket esistente che hai scelto.
     - (Facoltativo) Per Proprietario previsto del bucket, inserisci l'ID dell' AWS account che prevedi sia il proprietario del bucket di posizione di output dei risultati della query. Ti consigliamo di scegliere questa opzione come ulteriore misura di sicurezza. Se l'ID account del proprietario del bucket non corrisponde all'ID specificato, i tentativi di output nel bucket avranno esito negativo. Per informazioni dettagliate, consulta Verifica della proprietà del bucket con condizione del proprietario del bucket nella Guida per l'utente di Amazon S3.
     - (Facoltativo) Seleziona Assign bucket owner full control over query results (Assegna al proprietario del bucket il controllo completo dei risultati delle query) se la posizione dei risultati dei calcoli è di proprietà di un altro account e desideri concedere a tale altro account il controllo completo dei risultati delle tue query.

3. (Facoltativo) Seleziona Encrypt calculation results (Crittografa i risultati dei calcoli), quindi scegli una delle opzioni seguenti:

   - SSE_S3: è una chiave di crittografia lato server gestita da S3.

- SSE_KMS: una chiave fornita dall'utente. Per Scegli una AWS KMS chiave, puoi scegliere una delle seguenti opzioni:
  - Usa chiave AWS proprietaria: utilizza una chiave che AWS possiede e gestisce per te.
  - Scegli una AWS KMS chiave diversa (avanzata): scegli o crea una chiave.
    - Per utilizzare una chiave esistente, usa la casella di ricerca per scegliere AWS KMS o immettere una chiave ARN.
    - Per creare una chiave nella console KMS, scegli Crea una AWS KMS chiave. Dopo aver completato la creazione della chiave nella console KMS, torna alla pagina Crea gruppo di lavoro nella console Athena, quindi utilizza la casella di ricerca Scegli una AWS KMS chiave o inserisci un ARN per scegliere la chiave appena creata.

4. (Facoltativo) Altre impostazioni: espandi questa opzione per abilitare o disabilitare l'opzione Pubblica CloudWatch metriche per il gruppo di lavoro. Questo campo è selezionato per impostazione predefinita. Per ulteriori informazioni, consulta Monitoraggio dei calcoli di Apache Spark con i parametri di CloudWatch.
5. (Facoltativo) Tags (Tag): utilizza questa opzione per aggiungere tag al gruppo di lavoro. Per ulteriori informazioni, consulta Assegnazione di tag alle risorse Athena.
6. Selezionare Create workgroup (Crea gruppo di lavoro). Un messaggio informa che il gruppo di lavoro è stato creato correttamente e il gruppo di lavoro viene visualizzato nell'elenco dei gruppi di lavoro.

Apertura di Notebook explorer e cambio del gruppo di lavoro

Prima di poter utilizzare il gruppo di lavoro abilitato a Spark che hai appena creato, devi passare a tale gruppo di lavoro. Per passare da un gruppo di lavoro abilitato a Spark a un altro, puoi utilizzare l'opzione Workgroup (Gruppo di lavoro) in Notebook explorer o Notebook editor (Editor notebook).

> **Note**
>
> Prima di iniziare, verifica che il tuo browser non blocchi i cookie di terzi. Qualsiasi browser che blocca i cookie di terze parti per impostazione predefinita o abilitata dall'utente impedirà l'avvio di notebook. Per ulteriori informazioni sulla gestione dei cookie, consulta:
>
> - Chrome
> - Firefox

- Safari

Apertura di Notebook explorer e cambio del gruppo di lavoro

1. Nel pannello di navigazione, scegli Notebook explorer.
2. Usa l'opzione Workgroup (Gruppo di lavoro) nell'angolo in alto a destra della console per scegliere il gruppo di lavoro abilitato a Spark che hai creato. Il notebook di esempio è mostrato nell'elenco dei notebook.

   Puoi utilizzare Notebook explorer nei modi seguenti:

   - Scegli il nome con collegamento di un notebook per aprirlo in una nuova sessione.
   - Per rinominare, eliminare o esportare il tuo notebook, utilizza il menu Actions (Operazioni).
   - Per importare un file del notebook, scegli Import file (Importa file).
   - Per creare un notebook, scegli Create notebook (Crea notebook).

Esecuzione del notebook di esempio

Il notebook di esempio richiama i dati da un set di dati di viaggi in taxi a New York City disponibile al pubblico. Il taccuino contiene esempi che mostrano come lavorare con Spark DataFrames, Spark SQL e. AWS Glue Data Catalog

Esecuzione del notebook di esempio

1. In Notebook explorer, scegli il nome collegato del notebook di esempio.

   Questo avvia una sessione del notebook con i parametri predefiniti e apre il notebook nell'editor notebook. Un messaggio informa che è stata avviata una nuova sessione di Apache Spark utilizzando i parametri predefiniti (massimo 20 DPU).
2. Per eseguire le celle in ordine e visualizzare i risultati, premi il pulsante Run (Esegui) una volta per ogni cella del notebook.

   - Scorri verso il basso per vedere i risultati e visualizzare nuove celle.
   - Per le celle che contengono un calcolo, una barra di avanzamento mostra la percentuale di completamento, il tempo trascorso e il tempo rimanente.

- Il notebook di esempio crea un database e una tabella nell'account. La cella finale li rimuove come fase di pulizia.

> **Note**
>
> Se modifichi i nomi di cartelle, tabelle o database nel notebook di esempio, assicurati che tali modifiche si riflettano nei ruoli IAM che utilizzi. In caso contrario, il notebook potrebbe non funzionare a causa di autorizzazioni insufficienti.

Modifica dei dettagli della sessione

Dopo avere avviato una sessione di notebook, puoi modificare dettagli della sessione come, il formato delle tabelle, il timeout di inattività della sessione e il numero massimo di unità di elaborazione dati (DPU) simultanee che intendi utilizzare. Una DPU è una misura relativa della potenza di elaborazione ed è costituita da 4 vCPU di capacità di elaborazione e 16 GB di memoria.

Modifica dei dettagli della sessione

1. Nell'editor notebook, nel menu Session (Sessione) in alto a destra, scegli Edit session (Modifica sessione).
2. Nella finestra di dialogo Modifica dettagli sessione, nella sezione Proprietà Spark, scegli o inserisci i valori per le seguenti opzioni:

   - Formato di tabella aggiuntivo: seleziona Linux Foundation Delta Lake, Apache Hudi, Apache Iceberg oppure Personalizzato.
     - Per le opzioni di tabella Delta, Hudi o Iceberg, le proprietà di tabella richieste per il formato di tabella corrispondente vengono fornite automaticamente nelle opzioni Modifica nella tabella e Modifica in JSON. Per ulteriori informazioni sull'utilizzo di queste tabelle, consulta Utilizzo di formati di tabella non Hive in Amazon Athena per Apache Spark.
     - Per aggiungere o rimuovere proprietà di tabella per i tipi di tabella Personalizzata o di altro tipo, utilizzate le opzioni Modifica nella tabella e Modifica in JSON.
     - Per l'opzione Modifica nella tabella, seleziona Aggiungi proprietà per aggiungere una proprietà o Rimuovi per rimuovere una proprietà. Per immettere i nomi delle proprietà e i relativi valori, utilizzate le caselle Chiave e Valore.

- Per l'opzione Modifica in JSON, utilizzate l'editor di testo JSON per modificare direttamente la configurazione.
  - Per copiare il testo JSON negli appunti, seleziona Copia.
  - Per rimuovere tutto il testo dall'editor JSON, scegli Cancella.
  - Per configurare la disposizione delle linee o scegliere un tema di colore per l'editor JSON, scegli l'icona delle impostazioni (a forma di ingranaggio).
- Attiva la crittografia Spark -: seleziona questa opzione per crittografare i dati scritti su disco e inviati tramite i nodi di rete Spark. Per ulteriori informazioni, consulta Abilitazione della crittografia Apache Spark.

3. Nella sezione Parametri sessione scegli o inserisci i valori per le seguenti opzioni:

- Session idle timeout (Timeout di inattività della sessione): scegli o inserisci un valore compreso tra 1 e 480 minuti. Il valore di default è 20.
- Coordinator size (Dimensione del coordinatore): un coordinatore è un esecutore speciale che orchestra il lavoro di elaborazione e gestisce altri esecutori in una sessione di notebook. Attualmente, 1 DPU è il valore predefinito nonché l'unico possibile.
- Executor size (Dimensioni dell'esecutore): un esecutore è l'unità di calcolo più piccola che una sessione di notebook può richiedere ad Athena. Attualmente, 1 DPU è il valore predefinito nonché l'unico possibile.
- Max concurrent value (Valore simultaneo massimo): il numero massimo di DPU che possono essere eseguite contemporaneamente. Il valore predefinito 20, il valore minimo è 3 e il valore massimo è 60. L'aumento di questo valore non alloca automaticamente risorse aggiuntive, ma Athena tenterà di allocare fino al massimo specificato quando il carico di elaborazione lo richiede e quando le risorse sono disponibili.

4. Selezionare Salva.
5. Alla richiesta Confirm edit (Conferma modifica), scegli Confirm (Conferma).

Athena salva il notebook e avvia una nuova sessione con i parametri specificati. Un banner nell'editor notebook informa che è iniziata una nuova sessione con i parametri modificati.

> **Note**
>
> Athena ricorda le impostazioni della sessione per il notebook. Se modifichi i parametri di una sessione e poi la termini, Athena utilizza i parametri di sessione che hai configurato la volta successiva che avvii una sessione del notebook.

Visualizzazione dei dettagli della sessione e del calcolo

Dopo avere eseguito il notebook, è possibile visualizzare i dettagli della sessione e del calcolo.

Visualizzazione dei dettagli della sessione e del calcolo

1. Dal menu Session (Sessione) in alto a destra, scegli View details (Visualizza dettagli).
   - La scheda Current session (Sessione corrente) mostra informazioni sulla sessione corrente, tra cui l'ID della sessione, l'ora di creazione, lo stato e il gruppo di lavoro.
   - La scheda History (Cronologia) elenca gli ID delle sessioni precedenti. Per visualizzare i dettagli di una sessione precedente, scegli la scheda History (Cronologia), quindi scegli un ID di sessione nell'elenco.
   - La sezione Calculations (Calcoli) mostra un elenco dei calcoli eseguiti durante la sessione.
2. Per visualizzare i dettagli di un calcolo, scegli l'ID del calcolo.
3. Nella pagina Calculation details (Dettagli del calcolo), è possibile eseguire le seguenti operazioni:
   - Per visualizzare il codice per il calcolo, consulta la sezione Code (Codice).
   - Per visualizzare i risultati del calcolo, scegli la scheda Results (Risultati).
   - Per scaricare i risultati visualizzati in formato di testo, scegli Download results (Scarica risultati).
   - Per visualizzare informazioni sui risultati del calcolo in Amazon S3, scegli View in S3 (Visualizza in S3).

Terminazione di una sessione

Terminazione di una sessione del notebook

1. Nell'editor notebook, nel menu Session (Sessione) in alto a destra, scegli Terminate (Termina).
2. Alla richiesta Confirm session termination (Conferma chiusura della sessione), scegli Confirm (Conferma). Il notebook viene salvato e si torna all'editor notebook.

> **Note**
>
> La chiusura di una scheda del notebook nell'editor notebook non interrompe la sessione di un notebook attivo. Se vuoi assicurarti che la sessione venga terminata, utilizza l'opzione Session (Sessione), Terminate (Termina).

Creazione di un notebook

Dopo avere creato un gruppo di lavoro Athena, puoi creare il tuo notebook.

Creazione di un notebook

1. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.
2. Nel pannello di navigazione della console Athena, scegli Notebook explorer o Notebook editor (Editor notebook).
3. Esegui una di queste operazioni:
   - In Notebook explorer, scegli Create notebook (Crea notebook).
   - In Notebook editor (Editor notebook), scegli Create notebook (Crea notebook) oppure seleziona l'icona con il segno più (+) per aggiungere un notebook.
4. Nella finestra di dialogo Create notebook (Crea notebook), per Notebook name (Nome del notebook) inserisci un nome.
5. (Facoltativo) Espandi Proprietà Spark, quindi scegli o inserisci i valori per le seguenti opzioni:
   - Formato di tabella aggiuntivo: seleziona Linux Foundation Delta Lake, Apache Hudi, Apache Iceberg, oppure Personalizzato.
     - Per le opzioni di tabella Delta, Hudi o Iceberg, le proprietà di tabella richieste per il formato di tabella corrispondente vengono fornite automaticamente nelle opzioni Modifica nella tabella e Modifica in JSON. Per ulteriori informazioni sull'utilizzo di queste tabelle, consulta Utilizzo di formati di tabella non Hive in Amazon Athena per Apache Spark.
     - Per aggiungere o rimuovere proprietà di tabella per i tipi di tabella Personalizzata o di altro tipo, utilizzate le opzioni Modifica nella tabella e Modifica in JSON.
     - Per l'opzione Modifica nella tabella, seleziona Aggiungi proprietà per aggiungere una proprietà o Rimuovi per rimuovere una proprietà. Per immettere i nomi delle proprietà e i relativi valori, utilizzate le caselle Chiave e Valore.

- Per l'opzione Modifica in JSON, utilizzate l'editor di testo JSON per modificare direttamente la configurazione.
  - Per copiare il testo JSON negli appunti, seleziona Copia.
  - Per rimuovere tutto il testo dall'editor JSON, scegli Cancella.
  - Per configurare la disposizione delle linee o scegliere un tema di colore per l'editor JSON, scegli l'icona delle impostazioni (a forma di ingranaggio).
- Attiva la crittografia Spark -: seleziona questa opzione per crittografare i dati scritti su disco e inviati tramite i nodi di rete Spark. Per ulteriori informazioni, consulta Abilitazione della crittografia Apache Spark.

6. (Facoltativo) Espandi Session parameters (Parametri della sessione), quindi scegli o inserisci i valori per le seguenti opzioni:
   - Session idle timeout (Timeout di inattività della sessione): scegli o inserisci un valore compreso tra 1 e 480 minuti. Il valore di default è 20.
   - Coordinator size (Dimensione del coordinatore): un coordinatore è un esecutore speciale che orchestra il lavoro di elaborazione e gestisce altri esecutori in una sessione di notebook. Attualmente, 1 DPU è il valore predefinito nonché l'unico possibile. Una DPU (data processing unit, ossia unità di elaborazione dati) è una misura relativa della potenza di elaborazione ed è costituita da 4 vCPU di capacità di elaborazione e 16 GB di memoria.
   - Executor size (Dimensioni dell'esecutore): un esecutore è l'unità di calcolo più piccola che una sessione di notebook può richiedere ad Athena. Attualmente, 1 DPU è il valore predefinito nonché l'unico possibile.
   - Max concurrent value (Valore simultaneo massimo): il numero massimo di DPU che possono essere eseguite contemporaneamente. Il valore predefinito è 20 e il valore massimo è 60. L'aumento di questo valore non alloca automaticamente risorse aggiuntive, ma Athena tenterà di allocare fino al massimo specificato quando il carico di elaborazione lo richiede e quando le risorse sono disponibili.
7. Scegli Crea. Il notebook si apre in una nuova sessione nell'editor di notebook.

Apertura di un notebook creato in precedenza

Apertura di un notebook creato in precedenza

1. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

2. Nel pannello di navigazione della console Athena, scegli Notebook editor (Editor notebook) o Notebook explorer.
3. Esegui una di queste operazioni:
   - In Notebook editor (Editor notebook), scegli un notebook nell'elenco Recent notebooks (Notebook recenti) o Saved notebooks (Notebook salvati). Il notebook si apre in una nuova sessione.
   - In Notebook explorer, scegli il nome di un notebook dall'elenco. Il notebook si apre in una nuova sessione.

Per ulteriori informazioni sulla gestione dei file di notebook, consulta la pagina Gestione dei file di notebook.

Utilizzo dei notebook

Puoi gestire i notebook in Notebook explorer di Athena e modificarli ed eseguirli in sessioni utilizzando l'editor notebook di Athena. Puoi configurare l'utilizzo della DPU per le sessioni dei notebook in base alle tue esigenze.

Quando arresti un notebook, termini anche la sessione associata. Tutti i file vengono salvati, ma le modifiche in corso nelle variabili, nelle funzioni e nelle classi dichiarate vanno perse. Quando riavvii il notebook, Athena ricarica i file del notebook e puoi eseguire nuovamente il codice.

Sessioni e calcoli

Ogni notebook è associato a un singolo kernel Python ed esegue codice Python. Un notebook può avere una o più celle che contengono comandi. Per eseguire le celle in un notebook, innanzitutto è necessario creare una sessione del notebook. Le sessioni tengono traccia delle variabili e dello stato dei notebook.

Eseguire una cella in un notebook significa eseguire un calcolo nella sessione corrente. I calcoli fanno progredire lo stato del notebook e possono eseguire attività come la lettura da Amazon S3 o la scrittura su altri archivi dati. Finché una sessione è in esecuzione, i calcoli utilizzano e modificano lo stato mantenuto per il notebook.

Quando lo stato non ti è più utile, puoi terminare una sessione. Quando termini una sessione, il notebook persiste, ma le variabili e le altre informazioni sullo stato vengono eliminate. Se devi

lavorare su più progetti contemporaneamente, puoi creare una sessione per ogni progetto: le sessioni sono indipendenti l'una dall'altra.

Le sessioni hanno una capacità di elaborazione dedicata, misurata in DPU. Quando crei una sessione, puoi assegnarle un numero di DPU. Sessioni diverse possono avere capacità diverse a seconda dei requisiti dell'attività.

Utilizzo dell'editor notebook Athena

L'editor notebook Athena è un ambiente interattivo per la scrittura e l'esecuzione di codice. Le seguenti sezioni descrivono le funzionalità dell'ambiente.

Modalità di comando e modalità di modifica

L'editor notebook ha un'interfaccia utente modale: una modalità di modifica per inserire il testo in una cella e una modalità di comando per impartire all'editor stesso comandi come copia, incolla o esegui.

Per utilizzare la modalità di modifica e la modalità di comando, è possibile eseguire le seguenti operazioni:

- Per accedere alla modalità di modifica, premi **`ENTER`** o scegli una cella. Quando è in modalità di modifica, la cella ha il margine sinistro verde.
- Per accedere alla modalità di comando, premi **`ESC`** o fai clic all'esterno di una cella. Tieni presente che i comandi in genere si applicano solo alla cella attualmente selezionata, non a tutte le celle. Quando l'editor è in modalità di comando, la cella ha il margine sinistro blu.
- In modalità comando, puoi utilizzare le scorciatoie da tastiera e il menu sopra l'editor, ma non puoi inserire il testo nelle singole celle.
- Per selezionare una cella, scegli la cella.
- Per selezionare tutte le celle, premi **`Ctrl+A`** (Windows) o **`Cmd+A`** (Mac).

Menu dell'editor notebook

Le icone nel menu nella parte superiore dell'editor notebook offrono le seguenti opzioni:

- Save (Salva): salva lo stato corrente del notebook.
- Insert cell below (Inserisci cella sotto): aggiunge una nuova cella (vuota) sotto quella attualmente selezionata.

- Cut selected cells (Taglia celle selezionate): rimuove la cella selezionata dalla posizione corrente e la copia in memoria.
- Copy selected cells (Copia celle selezionate): copia la cella selezionata in memoria.
- Paste cells below (Incolla celle sotto): incolla la cella copiata sotto la cella corrente.
- Move selected cells up (Sposta le celle selezionate in alto): sposta la cella corrente sopra la cella soprastante.
- Move selected cells down (Sposta le celle selezionate in basso): sposta la cella corrente sotto la cella sottostante.
- Run (Esegui): esegue la cella corrente (selezionata). L'output viene visualizzato immediatamente sotto la cella corrente.
- Run all (Esegui tutto): esegue tutte le celle del notebook. L'output per ogni cella viene visualizzato immediatamente sotto la cella.
- Stop (Interrupt the kernel) (Arresta [interrompi il kernel]): arresta il notebook corrente interrompendo il kernel.
- Format option (Opzione di formato): seleziona il formato della cella, che può essere uno dei seguenti:
  - Code (Codice): utilizzato per il codice Python (impostazione predefinita).
  - Markdown: da utilizzare per inserire testo in formato markdown in GitHub stile -style. Per renderizzare il markdown, esegui la cella.
  - Raw NBConvert (NBConvert non elaborato): utilizzato per inserire testo in forma non modificata. Le celle contrassegnate come Raw NBConvert (NBConvert non elaborato) possono essere convertite in un formato diverso come HTML dallo strumento a riga di comando nbconvert di Jupyter.
- Heading (Intestazione): utilizzato per modificare il livello dell'intestazione della cella.
- Command palette (Palette dei comandi): contiene i comandi del notebook Jupyter e le relative scorciatoie da tastiera. Per ulteriori informazioni sulle scorciatoie da tastiera, consulta le sezioni più avanti in questo documento.
- Session (Sessione): utilizza le opzioni di questo menu per visualizzare i dettagli di una sessione, modificare i parametri della sessione o terminare la sessione.

Scorciatoie da tastiera in modalità di comando

Di seguito sono elencate alcune tra le scorciatoie da tastiera più utilizzate nella modalità di comando dell'editor notebook. Queste scorciatoie sono disponibili dopo avere premuto **ESC** per accedere alla

modalità di comando. Per visualizzare l'elenco completo dei comandi disponibili nell'editor, premi **ESC + H**.

Chiave	Azione
1 - 6	Cambia il tipo di cella in markdown e imposta il livello dell'intestazione sul numero digitato
a	Crea una cella sopra la cella corrente
b	Crea una cella sotto la cella corrente
c	Copia la cella corrente in memoria
d d	Elimina la cella corrente
h	Visualizza la schermata di aiuto delle scorciatoie da tastiera
j	Scendi di una cella
k	Sali di una cella
m	Cambia il formato della cella corrente in markdown
r	Cambia il formato della cella corrente in raw (non elaborato)
s	Salva il notebook
v	Incolla il contenuto della memoria sotto la cella corrente
x	Taglia la cella o le celle selezionate
y	Cambia il formato della cella in codice
z	Annulla operazione
Ctrl+Enter	Esegui la cella corrente e accedi alla modalità di comando

Chiave	Azione
Shift+Enter o **Alt+Enter**	Esegui la cella corrente e crea una nuova cella sotto l'output e inserisci la nuova cella in modalità di modifica
Space	Pagina giù
Shift+Space	Pagina su
Shift + L	Attiva la visibilità dei numeri di riga nelle celle

Modifica delle scorciatoie in modalità di comando

L'editor notebook offre la possibilità di personalizzare le scorciatoie da tastiera in modalità di comando.

Modifica delle scorciatoie in modalità di comando

1. Dal menu dell'editor notebook, scegli Command palette (Palette comandi).
2. Dalla palette dei comandi, scegli la voce Edit command mode keyboard shortcuts (Modifica scorciatoie da tastiera in modalità di comando).
3. Utilizza l'interfaccia Edit command mode shortcuts (Modifica scorciatoie in modalità di comando) per mappare o rimappare i comandi che desideri sulla tastiera.

   Per visualizzare le istruzioni per modificare le scorciatoie in modalità di comando, scorri fino alla fine della schermata Edit command mode shortcuts (Modifica scorciatoie in modalità di comando).

Per informazioni sull'uso dei comandi magic in Athena per Apache Spark, consulta Utilizzo dei comandi magici.

Utilizzo dei comandi magici

I comandi magici, o magie, sono comandi speciali che è possibile eseguire in una cella del notebook. Ad esempio, `%env` mostra le variabili di ambiente in una sessione del notebook. Athena supporta le funzioni magiche in IPython 6.0.3.

Questa sezione mostra alcuni comandi magic chiave in Athena per Apache Spark.

- Per visualizzare un elenco dei comandi magici in Athena, esegui il comando **`%lsmagic`** in una cella del notebook.
- Per informazioni sull'uso di magic per creare grafici nei notebook Athena, consulta Magics per la creazione di grafici di dati.
- Per informazioni su ulteriori comandi magici, consulta la pagina Comandi magici incorporati nella documentazione di IPython.

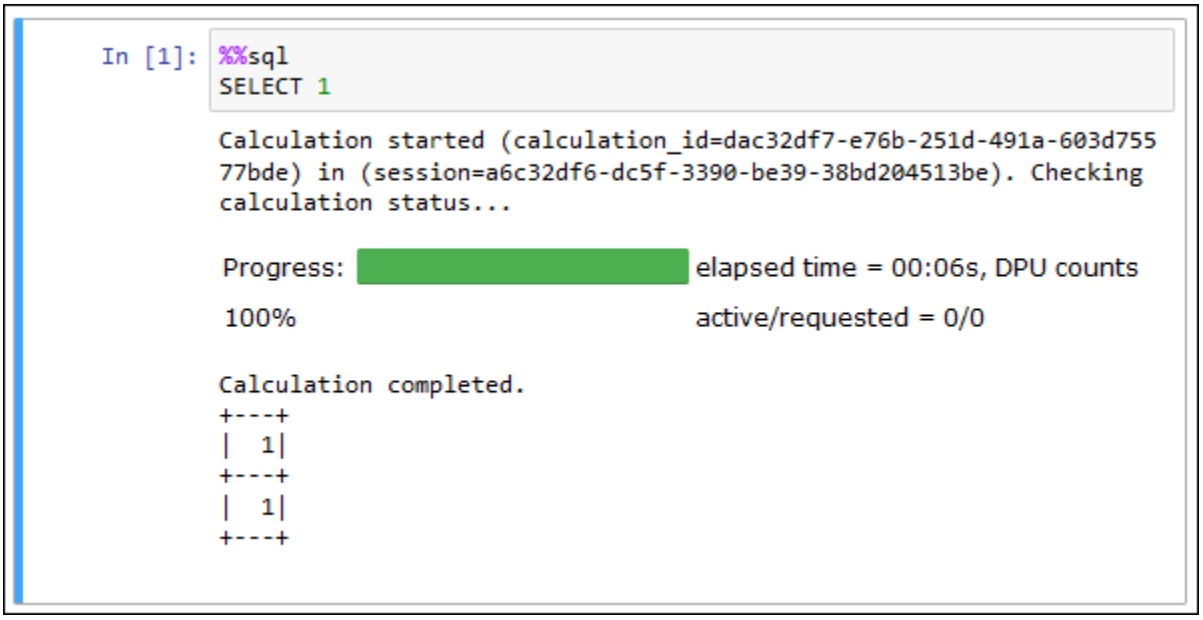
> **Note**
>
> Attualmente, il comando `%pip` fallisce quando viene eseguito. Si tratta di un problema noto.

Magic nelle celle

Le magie scritte su più righe sono precedute da un doppio segno percentuale (%%) e sono chiamate funzioni magiche di cella o magie di cella.

%%sql

Questo magic nelle celle consente di eseguire istruzioni SQL direttamente senza doverle decorare con l'istruzione SQL Spark. Il comando visualizza anche l'output richiamando implicitamente `.show()` nel dataframe restituito.



Il comando `%%sql` tronca automaticamente gli output delle colonne fino a una larghezza di 20 caratteri. Attualmente questa impostazione non è configurabile. Per ovviare a questa limitazione, utilizza la seguente sintassi completa e modifica di conseguenza i parametri del metodo `show`.

```
spark.sql("""YOUR_SQL""").show(n=number, truncate=number, vertical=bool)
```

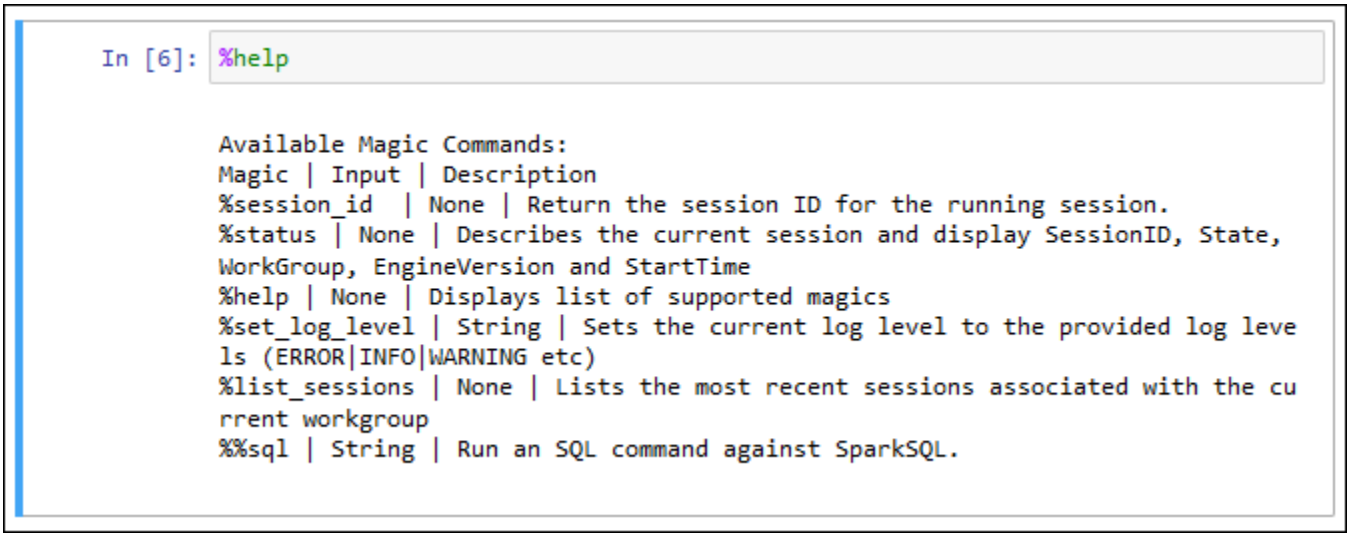
- n `int`, facoltativo. Il numero di righe da mostrare.
- truncate: `bool` o `int`, facoltativo: se `true`, tronca le stringhe più lunghe di 20 caratteri. Se impostato su un numero maggiore di 1, tronca le stringhe lunghe fino alla lunghezza specificata e allinea a destra le celle.
- verticale `bool`, facoltativo. Se `true`, stampa le righe di output in verticale (valore di una riga per colonna).

Magic di riga

Le magie che si trovano su una singola riga sono precedute da un segno percentuale (%) e sono chiamate funzioni magiche di riga o magie di riga.

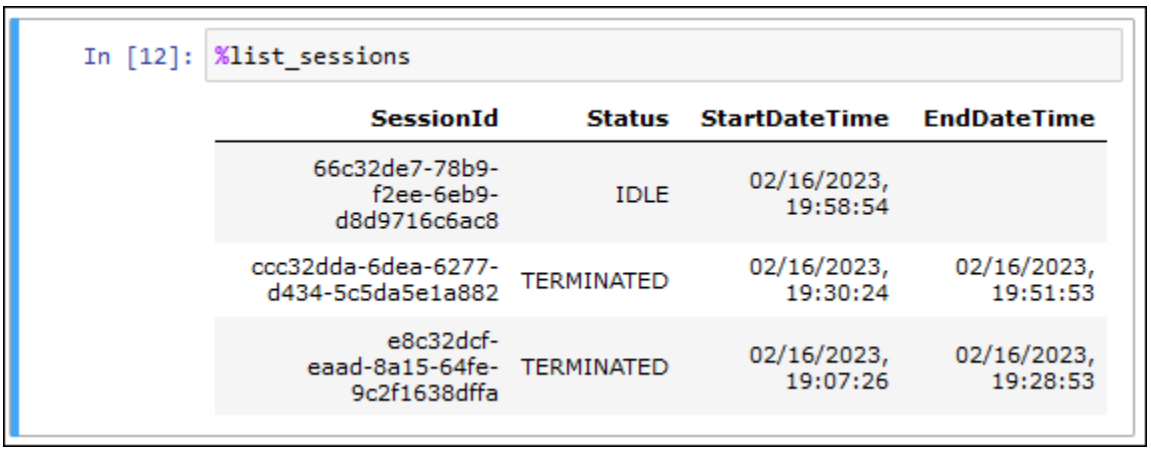
%help

Visualizza le descrizioni dei comandi magic disponibili.



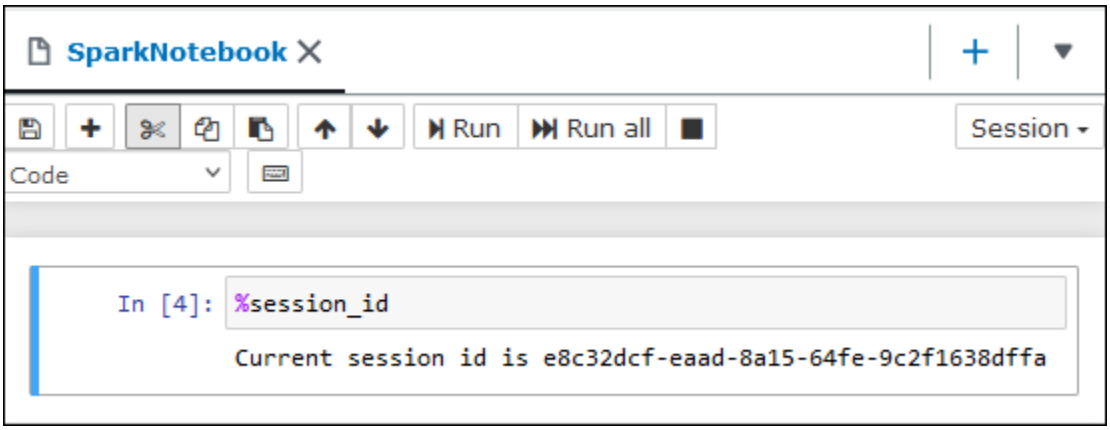
%list_sessions

Elenca le sessioni associate al notebook. Le informazioni per ogni sessione includono l'ID della sessione, lo stato della sessione e la data e l'ora di inizio e fine della sessione.



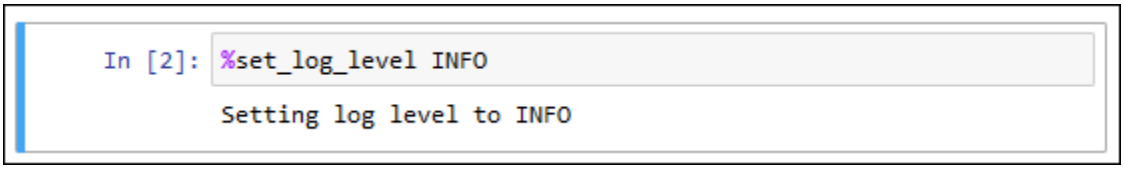
%session_id

Recupera l'ID della sessione corrente.



%set_log_level

Imposta o reimposta il logger per utilizzare il livello di log specificato. I valori possibili sono `DEBUG`, `ERROR`, `FATAL`, `INFO` e `WARN` o `WARNING`. I valori devono essere maiuscoli e non devono essere racchiusi tra virgolette singole o doppie.



%status

Descrive la sessione corrente. L'output include l'ID della sessione, lo stato della sessione, il nome del gruppo di lavoro, la versione del motore PySpark e l'ora di inizio della sessione. Questo comando magic richiede una sessione attiva per recuperare i dettagli della sessione.

Di seguito sono riportati i valori possibili per lo stato:

CREATING: la sessione viene avviata, inclusa l'acquisizione di risorse.

CREATED: la sessione è stata avviata.

IDLE: la sessione è in grado di accettare un calcolo.

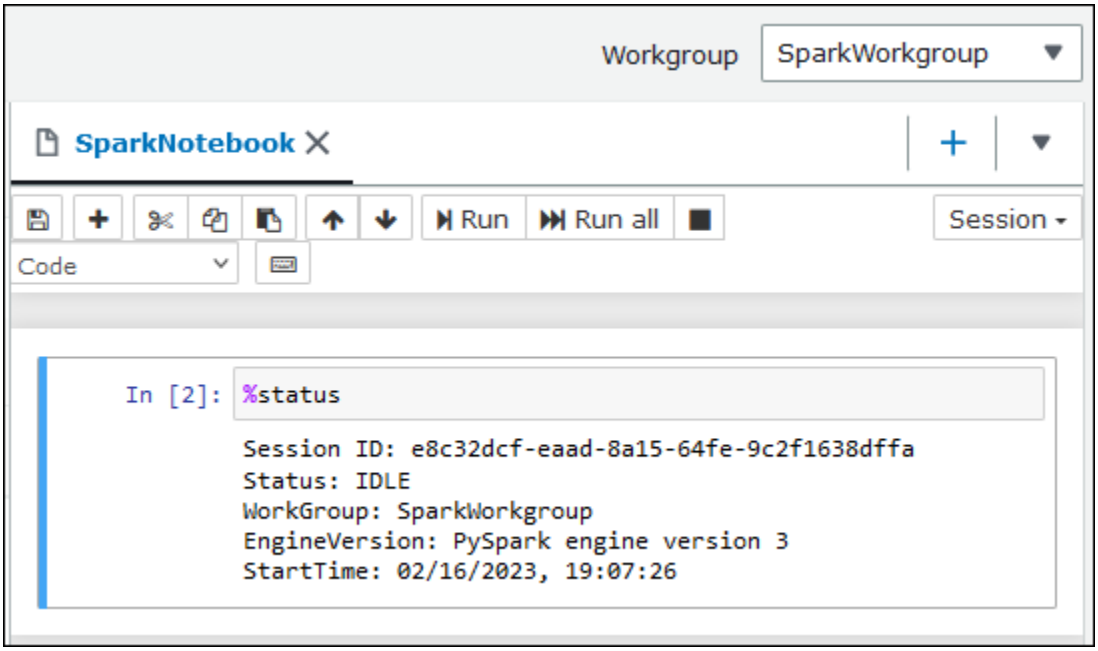
BUSY: la sessione sta elaborando un'altra operazione e non è in grado di accettare un calcolo.

TERMINATING: la sessione è in fase di arresto.

TERMINATED: la sessione e le relative risorse non sono più in esecuzione.

DEGRADED: la sessione non ha coordinatori sani.

FAILED: a causa di un errore, la sessione e le relative risorse non sono più in esecuzione.



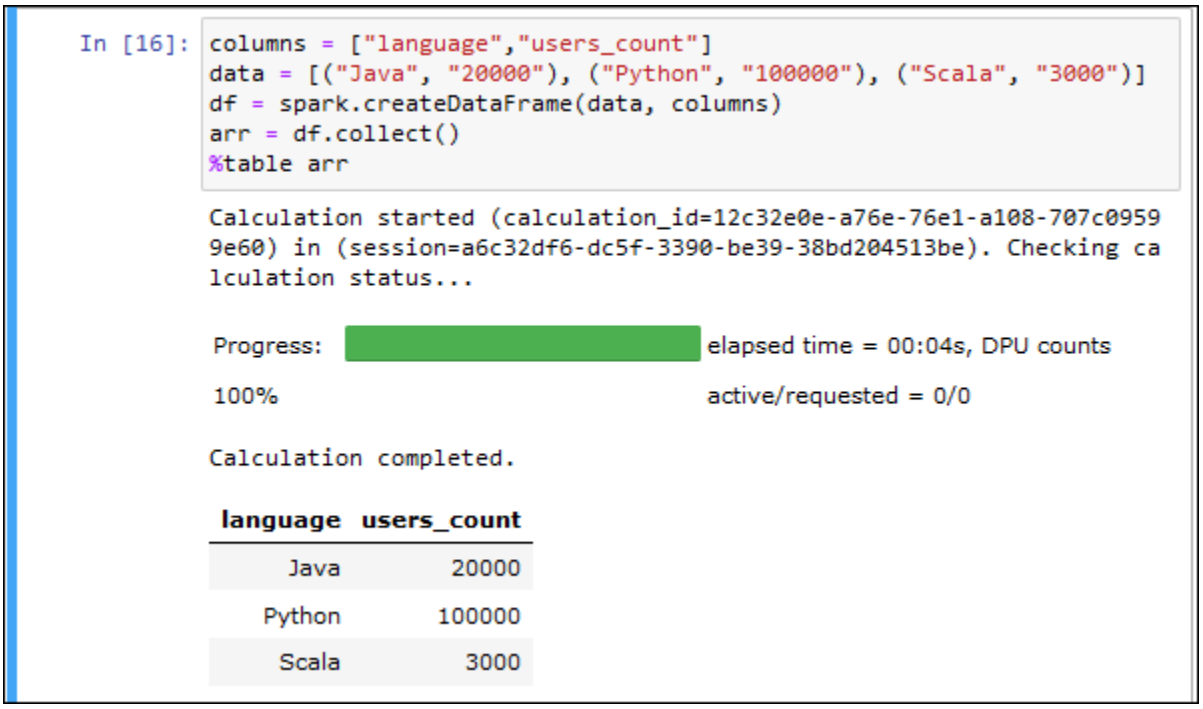
Magics per la creazione di grafici di dati

I line magics di questa sezione sono specializzati nel rendering di dati per particolari tipi di dati o in combinazione con librerie grafiche.

%table

È possibile utilizzare il comando magico `%table` per visualizzare i dati del dataframe in formato tabella.

L'esempio seguente crea un dataframe con due colonne e tre righe di dati, quindi visualizza i dati in formato tabella.



%matplot

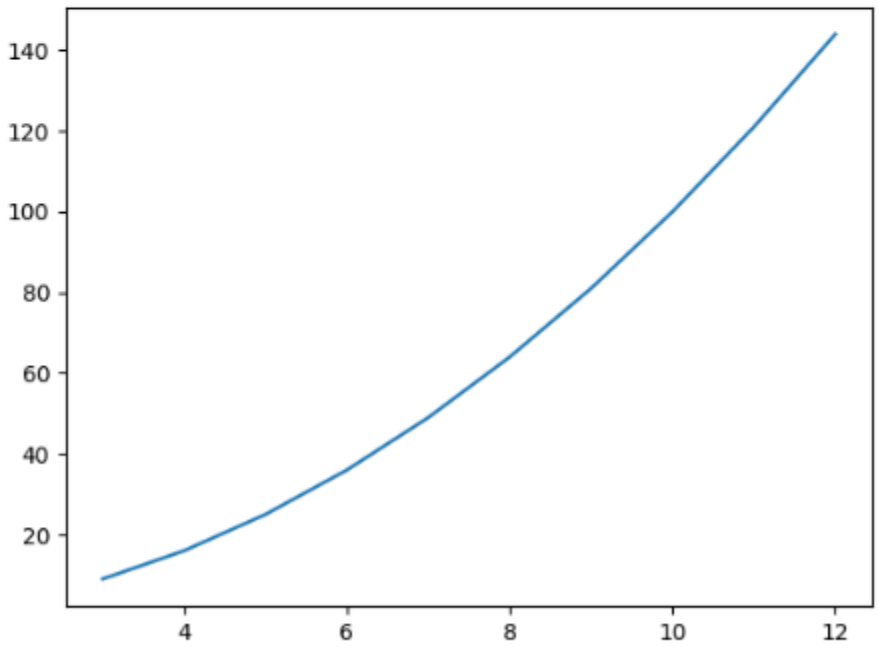
Matplotlib è una libreria completa per la creazione di visualizzazioni statiche, animate e interattive in Python. È possibile utilizzare il comando magic `%matplot` per creare un grafico dopo aver importato la libreria matplotlib in una cella del notebook.

L'esempio seguente importa la libreria matplotlib, crea un insieme di coordinate x e y, quindi utilizza il comando "use the `%matplot` magic" per creare un grafico dei punti.

```
import matplotlib.pyplot as plt
x=[3,4,5,6,7,8,9,10,11,12]
y= [9,16,25,36,49,64,81,100,121,144]
plt.plot(x,y)
%matplot plt
```

In [12]:
```
import matplotlib.pyplot as plt
x=[3,4,5,6,7,8,9,10,11,12]
y= [9,16,25,36,49,64,81,100,121,144]
plt.plot(x,y)
%matplot plt
```

```
Calculation started (calculation_id=5ac32e04-81b6-
9ee7-ce55-539ee2ce383e) in (session=a6c32df6-dc5f-
3390-be39-38bd204513be). Checking calculation stat
us...

Progress:                                elapsed time =

100%                                     00:02s

Calculation completed.
```




```
[<matplotlib.lines.Line2D object at 0x7f6e29e580>]
```

Usare insieme le librerie matplotlib e seaborn

Seaborn è una libreria per creare grafici statistici in Python. Si basa su matplotlib e si integra strettamente con le strutture di dati panda(Python data analysis, "analisi dei dati Python"). Puoi anche usare il comando magic `%matplot` per eseguire il rendering dei dati marittimi.

L'esempio seguente utilizza entrambe le librerie matplotlib e seaborn per creare un semplice grafico a barre.

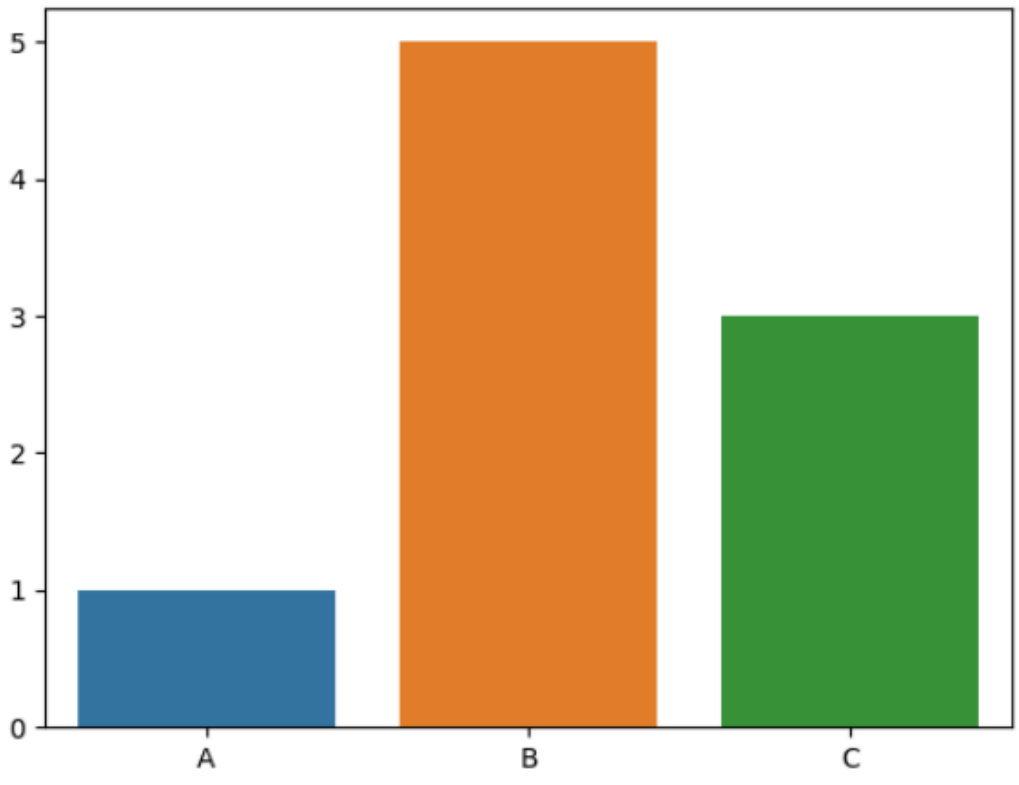
```
import matplotlib.pyplot as plt
import seaborn as sns

x = ['A', 'B', 'C']
y = [1, 5, 3]

sns.barplot(x, y)
%matplot plt
```

In [1]:
```
import matplotlib.pyplot as plt
import seaborn as sns


x = ['A', 'B', 'C']
y = [1, 5, 3]

sns.barplot(x, y)
%matplot plt
```

```
Calculation started (calculation_id=08c32e1b-233b-4a72-6571-1ae7a28a7b78)
in (session=64c32e1a-f45e-1d52-b54b-85202e2a9233). Checking calculation st
atus...
```

Progress: 100% elapsed time = 00:04s

```
Calculation completed.
```



%plotly

Plotly è una libreria grafica open source per Python che puoi usare per creare grafici interattivi. Utilizza il comando magic `%ploty` per eseguire il rendering dei dati grafici.

L'esempio seguente utilizza le librerie StringIO, plotly e pandas sui dati dei prezzi delle azioni per creare un grafico dell'attività azionaria tra febbraio e marzo 2015.

```
from io import StringIO
csvString = """
Date,AAPL.Open,AAPL.High,AAPL.Low,AAPL.Close,AAPL.Volume,AAPL.Adjusted,dn,mavg,up,direction
2015-02-17,127.489998,128.880005,126.919998,127.830002,63152400,122.905254,106.7410523,117.9276
2015-02-18,127.629997,128.779999,127.449997,128.720001,44891700,123.760965,107.842423,118.94033
2015-02-19,128.479996,129.029999,128.330002,128.449997,37362400,123.501363,108.8942449,119.8891
2015-02-20,128.619995,129.5,128.050003,129.5,48948400,124.510914,109.7854494,120.7635001,131.74
2015-02-23,130.020004,133,129.660004,133,70974100,127.876074,110.3725162,121.7201668,133.067817
2015-02-24,132.940002,133.600006,131.169998,132.169998,69228100,127.078049,111.0948689,122.6648
2015-02-25,131.559998,131.600006,128.149994,128.789993,74711700,123.828261,113.2119183,123.6296
2015-02-26,128.789993,130.869995,126.610001,130.419998,91287500,125.395469,114.1652991,124.2823
2015-02-27,130,130.570007,128.240005,128.460007,62014800,123.510987,114.9668484,124.8426669,134
2015-03-02,129.25,130.279999,128.300003,129.089996,48096700,124.116706,115.8770904,125.4036668,
2015-03-03,128.960007,129.520004,128.089996,129.360001,37816300,124.376308,116.9535132,125.9551
2015-03-04,129.100006,129.559998,128.320007,128.539993,31666300,123.587892,118.0874253,126.4730
2015-03-05,128.580002,128.75,125.760002,126.410004,56517100,121.539962,119.1048311,126.848667,1
2015-03-06,128.399994,129.369995,126.260002,126.599998,72842100,121.722637,120.190797,127.22883
2015-03-09,127.959999,129.570007,125.059998,127.139999,88528500,122.241834,121.6289771,127.6311
2015-03-10,126.410004,127.220001,123.800003,124.510002,68856600,119.71316,123.1164763,127.92350
"""
csvStringIO = StringIO(csvString)

from io import StringIO
import plotly.graph_objects as go
import pandas as pd
from datetime import datetime
df = pd.read_csv(csvStringIO)
fig = go.Figure(data=[go.Candlestick(x=df['Date'],
open=df['AAPL.Open'],
high=df['AAPL.High'],
low=df['AAPL.Low'],
close=df['AAPL.Close'])])
%plotly fig
```

Gestione dei file di notebook

Oltre che per creare e aprire notebook, puoi utilizzare Notebook explorer anche per rinominare, eliminare, esportare o importare notebook o visualizzare la cronologia delle sessioni di un notebook.

Ridenominazione di un notebook

1. Termina tutte le sessioni attive per il notebook che desideri rinominare. Le sessioni attive del notebook devono essere terminate prima di poter rinominare il notebook.
2. Apri Notebook explorer.
3. Nell'elenco Notebooks (Notebook), seleziona il pulsante di opzione per il notebook che desideri rinominare.

4. Nel menu Actions (Operazioni), scegli Rename (Rinomina).
5. Alla richiesta Rename notebook (Rinomina notebook), inserisci il nuovo nome, quindi scegli Save (Salva). Il nuovo nome del notebook viene visualizzato nell'elenco dei notebook.

Eliminazione di un notebook

1. Termina tutte le sessioni attive per il notebook che desideri eliminare. Le sessioni attive del notebook devono essere terminate prima di poter eliminare il notebook.
2. Apri Notebook explorer.
3. Nell'elenco Notebooks (Notebook), seleziona il pulsante di opzione per il notebook che desideri eliminare.
4. Nel menu Actions (Operazioni) selezionare Delete (Elimina).
5. Alla richiesta Delete notebook? (Eliminare notebook?), inserisci il nome del notebook, quindi scegli il pulsante Delete (Elimina) per confermare l'eliminazione. Il nome del notebook viene rimosso dall'elenco dei notebook.

Esportazione di un notebook

1. Apri Notebook explorer.
2. Nell'elenco Notebooks (Notebook), seleziona il pulsante di opzione per il notebook che desideri esportare.
3. Nel menu Actions (Operazioni), scegli Export file (Esporta file).

Importazione di un notebook

1. Apri Notebook explorer.
2. Scegli Import file (Importa file).
3. Individua la posizione sul computer locale del file che desideri importare, quindi scegli Open (Apri). Il notebook importato viene visualizzato nell'elenco dei notebook.

Visualizzazione della cronologia delle sessioni di un notebook

1. Apri Notebook explorer.

2. Nell'elenco Notebooks (Notebook), seleziona il pulsante di opzione per il notebook del quale desideri visualizzare la cronologia di sessione.
3. Nel menu Actions (Operazioni), scegli Session history (Cronologia della sessione).
4. Nella scheda History (Cronologia), scegli un Session ID (ID di sessione) per visualizzare le informazioni sulla sessione e i relativi calcoli.

Utilizzo di formati di tabella non Hive in Amazon Athena per Apache Spark

Quando lavori con sessioni e notebook in Athena for Spark, puoi usare le tabelle Delta Lake, Apache Hudi e Apache Iceberg di Linux Foundation, oltre alle tabelle Apache Hive.

Considerazioni e limitazioni

Quando si utilizzano formati di tabella diversi da Apache Hive con Athena per Spark, considerare i seguenti punti:

- Oltre ad Apache Hive, è supportato un solo formato di tabella per notebook. Per utilizzare più formati di tabella in Athena for Spark, crea un notebook separato per ogni formato di tabella. Per informazioni sulla creazione di taccuini in Athena per Spark, consulta Creazione di un notebook.
- I formati di tabella Delta Lake, Hudi e Iceberg sono stati testati su Athena for Spark utilizzandoli come metastore. AWS Glue Potresti essere in grado di utilizzare altri metastore, ma tale utilizzo non è attualmente supportato.
- Per utilizzare i formati di tabella aggiuntivi, sostituisci la proprietà predefinita `spark_catalog`, come indicato nella console Athena e in questa documentazione. Questi cataloghi non Hive possono leggere le tabelle Hive, oltre ai propri formati di tabella.

Versioni di tabella

La seguente tabella mostra le versioni di tabella non Hive supportate in Amazon Athena per Apache Spark.

Formato della tabella	Versione supportata
Apache Iceberg	1.2.1
Apache Hudi	0,13

Formato della tabella	Versione supportata
Linux Foundation Delta Lake	2.0.2

In Athena for Spark, questi file `.jar` in formato tabella e le relative dipendenze vengono caricati nel classpath per i driver e gli executor Spark.

Per un post sul blog AWS Big Data che mostra come lavorare con i formati di tabella Iceberg, Hudi e Delta Lake utilizzando Spark SQL nei notebook Amazon Athena, consulta Usa Amazon Athena con Spark SQL per i tuoi formati di tabelle transazionali open source.

Argomenti

- Apache Iceberg
- Apache Hudi
- Linux Foundation Delta Lake

Apache Iceberg

Apache Iceberg è un formato di tabella aperta per set di dati di grandi dimensioni in Amazon Simple Storage Service (Amazon S3). Ti fornisce prestazioni di query rapide su tabelle di grandi dimensioni, commit atomici, scritture simultanee ed evoluzione delle tabelle compatibili con SQL.

Per utilizzare le tabelle Apache Iceberg in Athena for Spark, configura le seguenti proprietà Spark. Queste proprietà sono configurate automaticamente nella console Athena for Spark quando scegli Apache Iceberg come formato di tabella. Per la procedura, consulta Modifica dei dettagli della sessione o Creazione di un notebook.

```
"spark.sql.catalog.spark_catalog": "org.apache.iceberg.spark.SparkSessionCatalog",
"spark.sql.catalog.spark_catalog.catalog-impl":
 "org.apache.iceberg.aws.glue.GlueCatalog",
"spark.sql.catalog.spark_catalog.io-impl": "org.apache.iceberg.aws.s3.S3FileIO",
"spark.sql.extensions":
 "org.apache.iceberg.spark.extensions.IcebergSparkSessionExtensions"
```

La seguente procedura mostra come utilizzare una tabella Apache Iceberg in un notebook Athena per Spark. Esegui ogni passaggio in una nuova cella del notebook.

Per usare una tabella Apache Iceberg in Athena per Spark

1. Definisci le costanti da usare nel notebook.

```
DB_NAME = "NEW_DB_NAME"
TABLE_NAME = "NEW_TABLE_NAME"
TABLE_S3_LOCATION = "s3://DOC-EXAMPLE-BUCKET"
```

2. DataFrameCrea un Apache Spark.

```
columns = ["language","users_count"]
data = [("Golang", 3000)]
df = spark.createDataFrame(data, columns)
```

3. Crea un database.

```
spark.sql("CREATE DATABASE {} LOCATION '{}'".format(DB_NAME, TABLE_S3_LOCATION))
```

4. Crea una tabella Apache Iceberg vuota.

```
spark.sql("""
CREATE TABLE {}.{} (
language string,
users_count int
) USING ICEBERG
""".format(DB_NAME, TABLE_NAME))
```

5. Inserisci una riga di dati nella tabella.

```
spark.sql("""INSERT INTO {}.{} VALUES ('Golang',
 3000)""".format(DB_NAME, TABLE_NAME))
```

6. Conferma di poter eseguire query sulla nuova tabella.

```
spark.sql("SELECT * FROM {}.{}".format(DB_NAME, TABLE_NAME)).show()
```

Per ulteriori informazioni ed esempi su come lavorare con le tabelle Spark DataFrames e Iceberg, consulta Spark Queries nella documentazione di Apache Iceberg.

Apache Hudi

Apache Hudi è un framework open source per la gestione dei dati che semplifica l'elaborazione incrementale dei dati. Le azioni di inserimento, aggiornamento, annullamento ed eliminazione a livello di record vengono elaborate con maggiore precisione, il che riduce il sovraccarico.

Per utilizzare le tabelle Apache Hudi in Athena per Spark, configura le seguenti proprietà Spark. Queste proprietà sono configurate automaticamente nella console Athena for Spark quando scegli Apache Hudi come formato di tabella. Per la procedura, consulta Modifica dei dettagli della sessione o Creazione di un notebook.

```
"spark.sql.catalog.spark_catalog": "org.apache.spark.sql.hudi.catalog.HoodieCatalog",
"spark.serializer": "org.apache.spark.serializer.KryoSerializer",
"spark.sql.extensions": "org.apache.spark.sql.hudi.HoodieSparkSessionExtension"
```

La seguente procedura mostra come utilizzare una tabella Apache Hudi in un notebook Athena per Spark. Esegui ogni passaggio in una nuova cella del notebook.

Per usare una tabella Apache Hudi in Athena per Spark

1. Definisci le costanti da usare nel notebook.

```
DB_NAME = "NEW_DB_NAME"
TABLE_NAME = "NEW_TABLE_NAME"
TABLE_S3_LOCATION = "s3://DOC-EXAMPLE-BUCKET"
```

2. Crea DataFrameun Apache Spark.

```
columns = ["language","users_count"]
data = [("Golang", 3000)]
df = spark.createDataFrame(data, columns)
```

3. Crea un database.

```
spark.sql("CREATE DATABASE {} LOCATION '{}'".format(DB_NAME, TABLE_S3_LOCATION))
```

4. Crea una tabella Apache Hudi vuota.

```
spark.sql("""
CREATE TABLE {}.{} (
language string,
```

```
users_count int
) USING HUDI
TBLPROPERTIES (
primaryKey = 'language',
type = 'mor'
);
""".format(DB_NAME, TABLE_NAME))
```

5. Inserisci una riga di dati nella tabella.

```
spark.sql("""INSERT INTO {}.{} VALUES ('Golang',
 3000)""".format(DB_NAME,TABLE_NAME))
```

6. Conferma di poter eseguire query sulla nuova tabella.

```
spark.sql("SELECT * FROM {}.{}".format(DB_NAME, TABLE_NAME)).show()
```

Linux Foundation Delta Lake

Linux Foundation Delta Lake è un formato di tabella che puoi utilizzare per la Big data/analisi. Puoi usare Athena per Spark per leggere direttamente le tabelle Delta Lake archiviate in Amazon S3.

Per utilizzare le tabelle Delta Lake in Athena per Spark, configura le seguenti proprietà Spark. Queste proprietà sono configurate automaticamente nella console Athena for Spark quando scegli Delta Lake come formato di tabella. Per la procedura, consulta Modifica dei dettagli della sessione o Creazione di un notebook.

```
"spark.sql.catalog.spark_catalog" : "org.apache.spark.sql.delta.catalog.DeltaCatalog",
"spark.sql.extensions" : "io.delta.sql.DeltaSparkSessionExtension"
```

La seguente procedura mostra come utilizzare una tabella Delta Lake in un notebook Athena per Spark. Esegui ogni passaggio in una nuova cella del notebook.

Per usare una tabella Delta Lake in Athena per Spark

1. Definisci le costanti da usare nel notebook.

```
DB_NAME = "NEW_DB_NAME"
TABLE_NAME = "NEW_TABLE_NAME"
TABLE_S3_LOCATION = "s3://DOC-EXAMPLE-BUCKET"
```

2. Crea un Apache Spark. DataFrame

```
columns = ["language","users_count"]
data = [("Golang", 3000)]
df = spark.createDataFrame(data, columns)
```

3. Crea un database.

```
spark.sql("CREATE DATABASE {} LOCATION '{}'".format(DB_NAME, TABLE_S3_LOCATION))
```

4. Crea una tabella Delta Lake vuota.

```
spark.sql("""
CREATE TABLE {}.{} (
  language string,
  users_count int
) USING DELTA
""".format(DB_NAME, TABLE_NAME))
```

5. Inserisci una riga di dati nella tabella.

```
spark.sql("""INSERT INTO {}.{} VALUES ('Golang',
 3000)""".format(DB_NAME, TABLE_NAME))
```

6. Conferma di poter eseguire query sulla nuova tabella.

```
spark.sql("SELECT * FROM {}.{}".format(DB_NAME, TABLE_NAME)).show()
```

Supporto delle librerie Python in Amazon Athena per Apache Spark

Questa pagina descrive la terminologia utilizzata e la gestione del ciclo di vita seguita per i runtime, le librerie e i pacchetti utilizzati in Amazon Athena per Apache Spark.

Definizioni

- Amazon Athena per Apache Spark è una versione personalizzata di Apache Spark open source. Per visualizzare la versione corrente, esegui il comando `print(f'{spark.version}')` in una cella del notebook.

- Il runtime Athena è l'ambiente in cui viene eseguito il codice. L'ambiente include un interprete Python e librerie. PySpark
- Una libreria o un pacchetto esterno è una libreria JAR o Python di Java o Scala che non fa parte del runtime di Athena ma può essere inclusa nei processi di Athena for Spark. I pacchetti esterni possono essere creati da Amazon o da te.
- Un pacchetto di convenienza è una raccolta di pacchetti esterni selezionati da Athena che puoi scegliere di includere nelle tue applicazioni Spark.
- Un bundle combina il runtime Athena e un pacchetto di convenienza.
- Una libreria utente è una libreria o un pacchetto esterno che aggiungi esplicitamente al tuo processo Athena per Spark.
  - Una libreria utente è un pacchetto esterno che non fa parte di un pacchetto di convenienza. Una libreria utente richiede il caricamento e l'installazione, come quando si scrivono alcuni file `.py`, li si comprime e quindi si aggiunge il file `.zip` all'applicazione.
- Un'applicazione Athena per Spark è un processo o una query che invii ad Athena per Spark.

Gestione del ciclo di vita

Controllo delle versioni e impostazione del runtime come obsoleto

Il componente principale del runtime di Athena è l'interprete Python. Poiché Python è un linguaggio in evoluzione, vengono rilasciate regolarmente nuove versioni e il supporto per le versioni precedenti viene rimosso. Athena sconsiglia di eseguire programmi con versioni di interpreti Python obsolete e consiglia vivamente di utilizzare il runtime Athena più recente, quando possibile.

La pianificazione dell'impostazione del runtime di Athena come obsoleto è la seguente:

1. Dopo che avrà fornito un nuovo runtime, Athena continuerà a supportare il runtime precedente per 6 mesi. Durante questo periodo, Athena applicherà patch di sicurezza e aggiornamenti al runtime precedente.
2. Dopo 6 mesi, Athena terminerà il supporto per il runtime precedente. Athena non applicherà più patch di sicurezza e altri aggiornamenti al runtime precedente. Le applicazioni Spark che utilizzano il runtime precedente non saranno più idonee al supporto tecnico.
3. Dopo 12 mesi, non sarà più possibile aggiornare o modificare le applicazioni Spark in un gruppo di lavoro che utilizza il runtime precedente. Ti consigliamo di aggiornare le applicazioni Spark prima del termine di questo periodo. Al termine del periodo, sarà ancora possibile eseguire i notebook

esistenti, ma tutti i notebook che utilizzano ancora il runtime precedente registreranno un avviso in tal senso.

4. Dopo 18 mesi, non sarà più possibile eseguire i processi nel gruppo di lavoro utilizzando il runtime precedente.

Controllo delle versioni e impostazione dei pacchetti di convenienza come obsoleti

Il contenuto dei pacchetti di convenienza cambia nel tempo. Athena aggiunge, rimuove o aggiorna occasionalmente questi pacchetti di convenienza.

Athena utilizza le seguenti linee guida per i pacchetti di convenienza:

- I pacchetti di convenienza hanno uno schema di controllo delle versioni semplice, come 1, 2, 3.
- Ogni versione del pacchetto di convenienza include versioni specifiche di pacchetti esterni. Dopo che Athena ha creato un pacchetto di convenienza, il set di pacchetti esterni del pacchetto di convenienza e le versioni corrispondenti non cambiano.
- Athena crea una nuova versione del pacchetto di convenienza quando include un nuovo pacchetto esterno, rimuove un pacchetto esterno o aggiorna la versione di uno o più pacchetti esterni.

Athena imposta un pacchetto di convenienza come obsoleto quando imposta come obsoleto il runtime Athena utilizzato dal pacchetto. Athena può impostare i pacchetti come obsoleti prima del tempo per limitare il numero di pacchetti supportati.

La pianificazione dell'impostazione dei pacchetti di convenienza come obsoleti segue la pianificazione di impostazione come obsoleto del runtime di Athena.

Elenco delle librerie Python preinstallate

Le librerie Python preinstallate includono quanto segue.

```
boto3==1.24.31
botocore==1.27.31
certifi==2022.6.15
charset-normalizer==2.1.0
cycler==0.11.0
cython==0.29.30
docutils==0.19
fonttools==4.34.4
idna==3.3
```

```
jmespath==1.0.1
joblib==1.1.0
kiwisolver==1.4.4
matplotlib==3.5.2
mpmath==1.2.1
numpy==1.23.1
packaging==21.3
pandas==1.4.3
patsy==0.5.2
pillow==9.2.0
plotly==5.9.0
pmdarima==1.8.5
pyathena==2.9.6
pyparsing==3.0.9
python-dateutil==2.8.2
pytz==2022.1
requests==2.28.1
s3transfer==0.6.0
scikit-learn==1.1.1
scipy==1.8.1
seaborn==0.11.2
six==1.16.0
statsmodels==0.13.2
sympy==1.10.1
tenacity==8.0.1
threadpoolctl==3.1.0
urllib3==1.26.10
pyarrow==9.0.0
```

Note

- MLLib (libreria di machine learning Apache Spark) e il `pyspark.ml` pacchetto non sono supportati.
- Al momento, non `pip install` è supportato nelle sessioni di Athena for Spark.

Per informazioni sull'importazione di librerie Python in Amazon Athena per Apache Spark, consulta. Importazione di file e librerie Python in Amazon Athena per Apache Spark

Importazione di file e librerie Python in Amazon Athena per Apache Spark

Questo documento fornisce esempi di come importare file e librerie Python in Amazon Athena per Apache Spark.

Considerazioni e limitazioni

- Versione Python: attualmente, Athena for Spark utilizza la versione Python 3.9.16. Nota che i pacchetti Python sono sensibili alle versioni minori di Python.
- Architettura Athena per Spark: Athena per Spark utilizza Amazon Linux 2 su architettura ARM64. Nota che alcune librerie Python non distribuiscono file binari per questa architettura.
- Oggetti binari condivisi (SO) — Poiché il SparkContext addPyFilemetodo non rileva oggetti binari condivisi, non può essere usato in Athena per Spark per aggiungere pacchetti Python che dipendono da oggetti condivisi.
- Resilient Distributed Dataset (RDD): gli RDD non sono supportati.
- DataFrame.foreach — Il metodo .foreach non è supportato. PySpark DataFrame

Esempi

Gli esempi utilizzano le seguenti convenzioni.

- Il segnaposto della posizione di Amazon S3 `s3://DOC-EXAMPLE-BUCKET`. Sostituisci questo valore con la posizione del tuo bucket S3.
- Tutti i blocchi di codice eseguiti da una shell (interprete di comandi) Unix vengono visualizzati come *`directory_name`* $. Ad esempio, il comando `ls` nella directory `/tmp` e il relativo output vengono visualizzati come segue:

```
/tmp $ ls
```

Output

```
file1 file2
```

- Aggiunta di un file a un notebook dopo averlo scritto nella directory temporanea locale
- Importazione di un file da Amazon S3
- Aggiunta di file Python e registrazione di un'UDF
- Importazione di un file .zip Python
- Importazione di due versioni di una libreria Python come moduli separati
- Importazione di un file .zip Python da PyPI

- Importazione di un file .zip Python da PyPI con dipendenze

Importazione di file di testo da utilizzare nei calcoli

Gli esempi di questa sezione mostrano come importare file di testo da utilizzare nei calcoli nei notebook in Athena per Spark.

Aggiunta di un file a un notebook dopo averlo scritto nella directory temporanea locale

L'esempio seguente mostra come scrivere un file in una directory temporanea locale, aggiungerlo a un notebook e testarlo.

```
import os
from pyspark import SparkFiles
tempdir = '/tmp/'
path = os.path.join(tempdir, "test.txt")
with open(path, "w") as testFile:
    _ = testFile.write("5")
sc.addFile(path)

def func(iterator):
    with open(SparkFiles.get("test.txt")) as testFile:
        fileVal = int(testFile.readline())
        return [x * fileVal for x in iterator]

#Test the file
from pyspark.sql.functions import udf
from pyspark.sql.functions import col

udf_with_import = udf(func)
df = spark.createDataFrame([(1, "a"), (2, "b")])
df.withColumn("col", udf_with_import(col('_2'))).show()
```

Output

```
Calculation completed.
+---+---+-------+
| _1| _2|    col|
+---+---+-------+
|  1|  a|[aaaaa]|
|  2|  b|[bbbbb]|
```

```
+---+---+-------+
```

Importazione di un file da Amazon S3

Il seguente esempio mostra come importare un file da Amazon S3 in un notebook e testarlo.

Importazione di un file da Amazon S3 in un notebook

1. Crea un file denominato `test.txt` con una singola riga contenente il valore `5`.
2. Aggiungi il file a un bucket in Amazon S3. Questo esempio utilizza la posizione `s3://DOC-EXAMPLE-BUCKET`.
3. Utilizza il codice seguente per importare il file sul tuo notebook e testarlo.

```
from pyspark import SparkFiles
sc.addFile('s3://DOC-EXAMPLE-BUCKET/test.txt')

def func(iterator):
   with open(SparkFiles.get("test.txt")) as testFile:
       fileVal = int(testFile.readline())
       return [x * fileVal for x in iterator]

#Test the file
from pyspark.sql.functions import udf
from pyspark.sql.functions import col

udf_with_import = udf(func)
df = spark.createDataFrame([(1, "a"), (2, "b")])
df.withColumn("col", udf_with_import(col('_2'))).show()
```

Output

```
Calculation completed.
+---+---+-------+
| _1| _2|    col|
+---+---+-------+
|  1|  a|[aaaaa]|
|  2|  b|[bbbbb]|
+---+---+-------+
```

Aggiunta di file Python

Gli esempi in questa sezione mostrano come aggiungere file e librerie Python ai notebook Spark in Athena.

Aggiunta di file Python e registrazione di un'UDF

La procedura seguente mostra come aggiungere file Python da Amazon S3 al notebook e come registrare un'UDF.

Aggiunta di file Python al notebook e registrazione di un'UDF

1. Utilizzando la tua posizione Amazon S3, crea il file `s3://DOC-EXAMPLE-BUCKET/file1.py` con i seguenti contenuti:

```
def xyz(input):
    return 'xyz  - udf ' + str(input);
```

2. Nella stessa posizione S3, crea il file `s3://DOC-EXAMPLE-BUCKET/file2.py` con i seguenti contenuti:

```
from file1 import xyz
def uvw(input):
    return 'uvw -> ' + xyz(input);
```

3. Nel tuo notebook Athena per Spark, esegui i seguenti comandi.

```
sc.addPyFile('s3://DOC-EXAMPLE-BUCKET/file1.py')
sc.addPyFile('s3://DOC-EXAMPLE-BUCKET/file2.py')

def func(iterator):
    from file2 import uvw
    return [uvw(x) for x in iterator]

from pyspark.sql.functions import udf
from pyspark.sql.functions import col

udf_with_import = udf(func)

df = spark.createDataFrame([(1, "a"), (2, "b")])

df.withColumn("col", udf_with_import(col('_2'))).show(10)
```

Output

```
Calculation started (calculation_id=1ec09e01-3dec-a096-00ea-57289cdb8ce7) in
 (session=c8c09e00-6f20-41e5-98bd-4024913d6cee). Checking calculation status...
Calculation completed.
+---+---+--------------------+
| _1| _2|                col|
+---+---+--------------------+
| 1 |  a|[uvw -> xyz - ud... |
| 2 |  b|[uvw -> xyz - ud... |
+---+---+--------------------+
```

Importazione di un file .zip Python

Puoi utilizzare i metodi `addPyFile` e `import` di Python per importare un file .zip Python nel tuo notebook.

> **Note**
>
> I file `.zip` importati in Athena Spark possono includere solo pacchetti Python. Ad esempio, l'inclusione di pacchetti con file basati su C non è supportata.

Importazione di un file **`.zip`** Python in un notebook

1. Sul tuo computer locale, in una directory del desktop come ad esempio `\tmp`, crea una directory chiamata `moduletest`.
2. Nella directory `moduletest`, creare un file denominato `hello.py` con il seguente contenuto:

```
def hi(input):
    return 'hi ' + str(input);
```

3. Nella stessa directory, aggiungi un file vuoto denominato `__init__.py`.

   Se elenchi i contenuti della directory, ora dovrebbero apparire come segue.

```
/tmp $ ls moduletest
__init__.py       hello.py
```

4. Utilizza il comando `zip` per inserire i due file del modulo in un file chiamato `moduletest.zip`.

```
moduletest $ zip -r9 ../moduletest.zip *
```

5. Carica il file `.zip` nel tuo bucket in Amazon S3.
6. Utilizza il codice seguente per importare il file `.zip` Python nel notebook.

```
sc.addPyFile('s3://DOC-EXAMPLE-BUCKET/moduletest.zip')

from moduletest.hello import hi

from pyspark.sql.functions import udf
from pyspark.sql.functions import col

hi_udf = udf(hi)

df = spark.createDataFrame([(1, "a"), (2, "b")])

df.withColumn("col", hi_udf(col('_2'))).show()
```

Output

```
Calculation started (calculation_id=6ec09e8c-6fe0-4547-5f1b-6b01adb2242c) in
 (session=dcc09e8c-3f80-9cdc-bfc5-7effa1686b76). Checking calculation status...
Calculation completed.
+---+---+----+
| _1| _2| col|
+---+---+----+
|  1|  a|hi a|
|  2|  b|hi b|
+---+---+----+
```

Importazione di due versioni di una libreria Python come moduli separati

I seguenti esempi di codice mostrano come aggiungere e importare due versioni distinte di una libreria Python da una posizione in Amazon S3 come due moduli separati. Il codice aggiunge ogni file della libreria da S3, lo importa e quindi stampa la versione della libreria per verificare l'importazione.

```
sc.addPyFile('s3://DOC-EXAMPLE-BUCKET/python-third-party-libs-test/
simplejson_v3_15.zip')
```

```
sc.addPyFile('s3://DOC-EXAMPLE-BUCKET/python-third-party-libs-test/
simplejson_v3_17_6.zip')

import simplejson_v3_15
print(simplejson_v3_15.__version__)
```

Output

```
3.15.0
```

```
import simplejson_v3_17_6
print(simplejson_v3_17_6.__version__)
```

Output

```
3.17.6
```

Importazione di un file .zip Python da PyPI

Questo esempio utilizza il comando `pip` per scaricare un file .zip in Python del progetto bpabel/piglatin dal Python Package Index (PyPI).

Importazione di un file .zip Python da PyPI

1. Sul desktop locale, utilizza i seguenti comandi per creare una directory chiamata `testpiglatin` e creare un ambiente virtuale.

   ```
   /tmp $ mkdir testpiglatin
   /tmp $ cd testpiglatin
   testpiglatin $ virtualenv .
   ```

   Output

   ```
   created virtual environment CPython3.9.6.final.0-64 in 410ms
   creator CPython3Posix(dest=/private/tmp/testpiglatin, clear=False,
    no_vcs_ignore=False, global=False)
   seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle,
    via=copy, app_data_dir=/Users/user1/Library/Application Support/virtualenv)
   added seed packages: pip==22.0.4, setuptools==62.1.0, wheel==0.37.1
   ```

```
activators
 BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonAct
```

2. Crea una sottodirectory denominata `unpacked` per ospitare il progetto.

```
testpiglatin $ mkdir unpacked
```

3. Utilizza il comando `pip` per installare il progetto nella directory `unpacked`.

```
testpiglatin $ bin/pip install -t $PWD/unpacked piglatin
```

Output

```
Collecting piglatin
Using cached piglatin-1.0.6-py2.py3-none-any.whl (3.1 kB)
Installing collected packages: piglatin
Successfully installed piglatin-1.0.6
```

4. Verifica il contenuto della directory.

```
testpiglatin $ ls
```

Output

```
bin lib pyvenv.cfg unpacked
```

5. Passa alla directory `unpacked` e visualizza il contenuto.

```
testpiglatin $ cd unpacked
unpacked $ ls
```

Output

```
piglatin piglatin-1.0.6.dist-info
```

6. Utilizza il comando `zip` per inserire il contenuto del progetto piglatin in un file denominato `library.zip`.

```
unpacked $ zip -r9 ../library.zip *
```

Output

```
adding: piglatin/ (stored 0%)
adding: piglatin/__init__.py (deflated 56%)
adding: piglatin/__pycache__/ (stored 0%)
adding: piglatin/__pycache__/__init__.cpython-39.pyc (deflated 31%)
adding: piglatin-1.0.6.dist-info/ (stored 0%)
adding: piglatin-1.0.6.dist-info/RECORD (deflated 39%)
adding: piglatin-1.0.6.dist-info/LICENSE (deflated 41%)
adding: piglatin-1.0.6.dist-info/WHEEL (deflated 15%)
adding: piglatin-1.0.6.dist-info/REQUESTED (stored 0%)
adding: piglatin-1.0.6.dist-info/INSTALLER (stored 0%)
adding: piglatin-1.0.6.dist-info/METADATA (deflated 48%)
```

7. (Facoltativo) Utilizza i seguenti comandi per testare l'importazione localmente.

   a. Imposta il percorso Python nella posizione del file `library.zip` e avvia Python.

   ```
   /home $ PYTHONPATH=/tmp/testpiglatin/library.zip
   /home $ python3
   ```

   Output

   ```
   Python 3.9.6 (default, Jun 29 2021, 06:20:32)
   [Clang 12.0.0 (clang-1200.0.32.29)] on darwin
   Type "help", "copyright", "credits" or "license" for more information.
   ```

   b. Importa la libreria ed esegui un comando di test.

   ```
   >>> import piglatin
   >>> piglatin.translate('hello')
   ```

   Output

   ```
   'ello-hay'
   ```

8. Utilizza i comandi seguenti per aggiungere il file `.zip` da Amazon S3, importarlo nel tuo notebook in Athena e testarlo.

```
sc.addPyFile('s3://DOC-EXAMPLE-BUCKET/library.zip')
```

```
import piglatin
piglatin.translate('hello')

from pyspark.sql.functions import udf
from pyspark.sql.functions import col

hi_udf = udf(piglatin.translate)

df = spark.createDataFrame([(1, "hello"), (2, "world")])

df.withColumn("col", hi_udf(col('_2'))).show()
```

Output

```
Calculation started (calculation_id=e2c0a06e-f45d-d96d-9b8c-ff6a58b2a525) in
 (session=82c0a06d-d60e-8c66-5d12-23bcd55a6457). Checking calculation status...
Calculation completed.
+---+-----+--------+
| _1|   _2|     col|
+---+-----+--------+
|  1|hello|ello-hay|
|  2|world|orld-way|
+---+-----+--------+
```

Importazione di un file .zip Python da PyPI con dipendenze

Questo esempio importa da PyPI il pacchetto md2gemini, che converte il testo in formato Markdown nel formato di testo Gemini. Il pacchetto include le seguenti dipendenze:

```
cjkwrap
mistune
wcwidth
```

Importazione da PyPI di un file .zip Python con dipendenze

1. Sul computer locale, utilizza i seguenti comandi per creare una directory chiamata `testmd2gemini` e creare un ambiente virtuale.

```
/tmp $ mkdir testmd2gemini
/tmp $ cd testmd2gemini
```

```
testmd2gemini$ virtualenv .
```

2. Crea una sottodirectory denominata `unpacked` per ospitare il progetto.

```
testmd2gemini $ mkdir unpacked
```

3. Utilizza il comando `pip` per installare il progetto nella directory `unpacked`.

```
/testmd2gemini $ bin/pip install -t $PWD/unpacked md2gemini
```

Output

```
Collecting md2gemini
  Downloading md2gemini-1.9.0-py3-none-any.whl (31 kB)
Collecting wcwidth
  Downloading wcwidth-0.2.5-py2.py3-none-any.whl (30 kB)
Collecting mistune<3,>=2.0.0
  Downloading mistune-2.0.2-py2.py3-none-any.whl (24 kB)
Collecting cjkwrap
  Downloading CJKwrap-2.2-py2.py3-none-any.whl (4.3 kB)
Installing collected packages: wcwidth, mistune, cjkwrap, md2gemini
Successfully installed cjkwrap-2.2 md2gemini-1.9.0 mistune-2.0.2 wcwidth-0.2.5
...
```

4. Passa alla directory `unpacked` e controlla il contenuto.

```
testmd2gemini $ cd unpacked
unpacked $ ls -lah
```

Output

```
total 16
drwxr-xr-x  13 user1  wheel   416B Jun  7 18:43 .
drwxr-xr-x   8 user1  wheel   256B Jun  7 18:44 ..
drwxr-xr-x   9 user1  staff   288B Jun  7 18:43 CJKwrap-2.2.dist-info
drwxr-xr-x   3 user1  staff    96B Jun  7 18:43 __pycache__
drwxr-xr-x   3 user1  staff    96B Jun  7 18:43 bin
-rw-r--r--   1 user1  staff   5.0K Jun  7 18:43 cjkwrap.py
drwxr-xr-x   7 user1  staff   224B Jun  7 18:43 md2gemini
drwxr-xr-x  10 user1  staff   320B Jun  7 18:43 md2gemini-1.9.0.dist-info
drwxr-xr-x  12 user1  staff   384B Jun  7 18:43 mistune
drwxr-xr-x   8 user1  staff   256B Jun  7 18:43 mistune-2.0.2.dist-info
```

```
drwxr-xr-x  16 user1  staff   512B Jun  7 18:43 tests
drwxr-xr-x  10 user1  staff   320B Jun  7 18:43 wcwidth
drwxr-xr-x   9 user1  staff   288B Jun  7 18:43 wcwidth-0.2.5.dist-info
```

5. Utilizza il comando `zip` per inserire il contenuto del progetto md2gemini in un file denominato `md2gemini.zip`.

```
unpacked $ zip -r9 ../md2gemini *
```

Output

```
  adding: CJKwrap-2.2.dist-info/ (stored 0%)
  adding: CJKwrap-2.2.dist-info/RECORD (deflated 37%)
  ....
  adding: wcwidth-0.2.5.dist-info/INSTALLER (stored 0%)
  adding: wcwidth-0.2.5.dist-info/METADATA (deflated 62%)
```

6. (Facoltativo) Utilizza i seguenti comandi per verificare che la libreria funzioni sul computer locale.

   a. Imposta il percorso Python nella posizione del file `md2gemini.zip` e avvia Python.

```
/home $ PYTHONPATH=/tmp/testmd2gemini/md2gemini.zip
/home python3
```

   b. Importa la libreria ed esegui un test.

```
>>> from md2gemini import md2gemini
>>> print(md2gemini('[abc](https://abc.def)'))
```

Output

```
https://abc.def abc
```

7. Utilizza i comandi seguenti per aggiungere il file `.zip` da Amazon S3, importarlo nel tuo notebook in Athena ed eseguire un test non UDF.

```
# (non udf test)
sc.addPyFile('s3://DOC-EXAMPLE-BUCKET/md2gemini.zip')
from md2gemini import md2gemini
print(md2gemini('[abc](https://abc.def)'))
```

Output

```
Calculation started (calculation_id=0ac0a082-6c3f-5a8f-eb6e-f8e9a5f9bc44) in
 (session=36c0a082-5338-3755-9f41-0cc954c55b35). Checking calculation status...
Calculation completed.
=> https://abc.def (https://abc.def/) abc
```

8. Utilizza i comandi seguenti per eseguire un test UDF.

```
# (udf test)

from pyspark.sql.functions import udf
from pyspark.sql.functions import col
from md2gemini import md2gemini


hi_udf = udf(md2gemini)
df = spark.createDataFrame([(1, "[first website](https://abc.def)"), (2, "[second
 website](https://aws.com)")])
df.withColumn("col", hi_udf(col('_2'))).show()
```

Output

```
Calculation started (calculation_id=60c0a082-f04d-41c1-a10d-d5d365ef5157) in
 (session=36c0a082-5338-3755-9f41-0cc954c55b35). Checking calculation status...
Calculation completed.
+---+--------------------+--------------------+
| _1|                  _2|                 col|
+---+--------------------+--------------------+
|  1|[first website](h...|=> https://abc.de...|
|  2|[second website](...|=> https://aws.co...|
+---+--------------------+--------------------+
```

Aggiungere file JAR e configurazione Spark personalizzata

Quando crei o modifichi una sessione in Amazon Athena per Apache Spark, puoi utilizzare le proprietà Spark per specificare file `.jar`, pacchetti o un'altra configurazione personalizzata per la sessione. Per specificare le proprietà di Spark, puoi utilizzare la console Athena, o AWS CLI l'API Athena.

Utilizzo della console Athena per specificare le proprietà Spark

Nella console Athena, puoi specificare le proprietà di Spark quando crei un notebook o modifichi una sessione corrente.

Come aggiungere proprietà nella finestra di dialogo Crea notebook o Modifica dettagli della sessione

1. Espandi le proprietà Spark.
2. Per aggiungere le tue proprietà, utilizza l'opzione Modifica nella tabella o Modifica in JSON.
   - Per l'opzione Modifica nella tabella, seleziona Aggiungi proprietà per aggiungere una proprietà o seleziona Rimuovi per rimuovere una proprietà. Utilizza le caselle Chiave e Valore per inserire i nomi delle proprietà e i relativi valori.
     - Per aggiungere un file `.jar` personalizzato, utilizza la proprietà `spark.jars`.
     - Per specificare un file pacchetto, utilizza la proprietà `spark.jars.packages`.
   - Per inserire e modificare direttamente la configurazione, seleziona l'opzione Modifica in JSON. Nell'editor di testo JSON, puoi eseguire le seguenti attività:
     - Seleziona Copia per copiare il testo JSON negli appunti.
     - Seleziona Cancella per rimuovere tutto il testo dall'editor JSON.
     - Scegli l'icona delle impostazioni (ingranaggio) per configurare la disposizione delle linee o scegli un tema di colore per l'editor JSON.

Note

- Puoi impostare le proprietà in Athena per Spark, il che equivale a impostare le proprietà Spark direttamente su un oggetto SparkConf.
- Avvia tutte le proprietà Spark con il prefisso `spark.`. Le proprietà con altri prefissi vengono ignorate.
- Non tutte le proprietà Spark sono disponibili per la configurazione personalizzata su Athena. Se invii una richiesta `StartSession` con una configurazione limitata, la sessione non viene avviata.
  - Non è possibile utilizzare il prefisso `spark.athena.` perché è riservato.

Utilizzo dell'API AWS CLI o Athena per fornire una configurazione personalizzata

Per utilizzare l'API AWS CLI o Athena per fornire la configurazione della sessione, utilizza l'azione API StartSession o il ocmando CLI start-session. Nella richiesta `StartSession`, utilizza il campo `SparkProperties` dell'oggetto EngineConfiguration per passare le informazioni di configurazione in formato JSON. Questo avvia una sessione con la configurazione specificata. Per la sintassi della richiesta, consulta StartSession nella Guida di riferimento dell'API di Amazon Athena .

Risoluzione degli errori di avvio della sessione

Quando si verifica un errore di configurazione personalizzato durante l'avvio di una sessione, la console Athena for Spark mostra un banner con un messaggio di errore. Per risolvere gli errori di avvio della sessione, puoi controllare la modifica dello stato della sessione o le informazioni di registrazione.

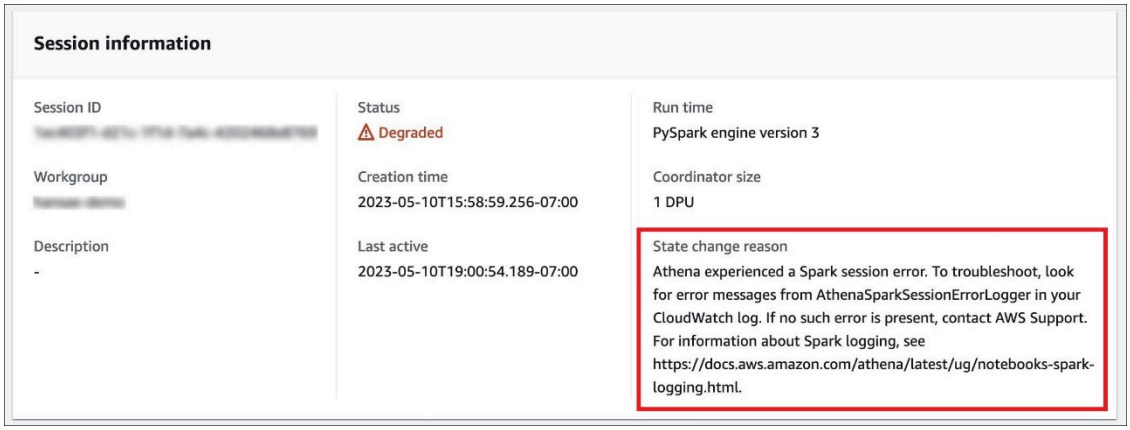
Visualizzazione delle informazioni sulla modifica dello stato della sessione

Puoi ottenere dettagli sulla modifica dello stato della sessione dall'editor di notebook Athena o dall'API Athena.

Per visualizzare le informazioni sullo stato della sessione nella console Athena

1. Nell'editor notebook di Athena, nel menu Sessione in alto a destra, scegli Vedi dettagli.
2. Visualizza la scheda Sessione corrente. La sezione Informazioni sessione mostra informazioni come l'ID della sessione, il gruppo di lavoro, lo stato e il motivo del cambio di stato.

   La seguente schermata d'esempio mostra le informazioni nella sezione Motivo modifica stato della finestra di dialogo Informazioni sessione per un errore di sessione Spark in Athena.



Per visualizzare le informazioni sullo stato della sessione utilizzando l'API Athena

- Nell'API Athena, puoi trovare informazioni sulla modifica dello stato della sessione nel campo `StateChangeReason` dell'oggetto SessionStatus.

> **Note**
>
> Dopo aver interrotto manualmente una sessione o se la sessione si interrompe dopo un timeout di inattività (l'impostazione predefinita è 20 minuti), il valore di StateChangeReason cambia in Sessione terminata a seguito richiesta.

Utilizzo della registrazione per risolvere gli errori di avvio della sessione

Gli errori di configurazione personalizzati che si verificano durante l'avvio di una sessione vengono registrati da Amazon CloudWatch. Nel tuo CloudWatch Logs, cerca i messaggi di errore da `AthenaSparkSessionErrorLogger` per risolvere un avvio non riuscito della sessione.

Per ulteriori informazioni sulla registrazione di log su Spark, consulta Registrazione di log di evento dell'applicazione Spark in Athena.

Per ulteriori informazioni sulla risoluzione dei problemi delle sessioni in Athena per Spark, consulta Risoluzione dei problemi delle sessioni.

Formati di dati e archiviazione supportati

La tabella seguente mostra i formati supportati in modo nativo in Athena per Apache Spark..

Formato dei dati	Lettura	Scrittura	Compressione di scrittura
parquet	sì	sì	nessuno, non compresso, snappy, gzip
orc	sì	sì	nessuno, snappy, zlib, lzo

Formato dei dati	Lettura	Scrittura	Compressione di scrittura
json	sì	sì	bzip2, gzip, deflate
csv	sì	sì	bzip2, gzip, deflate
text	sì	sì	nessuno, bzip2, gzip, deflate
file binario	sì	N/D	N/D

Monitoraggio dei calcoli di Apache Spark con i parametri di CloudWatch

Athena pubblica i parametri relativi al calcolo su Amazon CloudWatch quando viene selezionata l'opzione Publish CloudWatch metrics per il gruppo di lavoro abilitato per Spark. Nella console CloudWatch puoi creare pannelli di controllo personalizzati e impostare soglie e allarmi relativi ai parametri.

Athena pubblica i parametri seguenti nella console CloudWatch sotto lo spazio dei nomi `AmazonAthenaForApacheSpark`:

- `DPUCount`: il numero di DPU utilizzate durante la sessione per eseguire i calcoli.

Questo parametro possiede le seguenti dimensioni:

- `SessionId`: l'ID della sessione in cui vengono inviati i calcoli.
- `WorkGroup`: il nome del gruppo di lavoro.

Visualizzazione dei parametri dei gruppi di lavoro abilitati per Spark nella console Amazon CloudWatch

1. Aprire la console CloudWatch all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Nel pannello di navigazione, seleziona Metrics (Parametri), All metrics (Tutti i parametri).
3. Seleziona lo spazio dei nomi AmazonAthenaForApacheSpark.

Per visualizzare i parametri con CLI

- Completa una delle seguenti operazioni:
  - Per elencare i parametri per i gruppi di lavoro Athena abilitati per Spark, apri un prompt dei comandi e utilizza il comando seguente:

```
aws cloudwatch list-metrics --namespace "AmazonAthenaForApacheSpark"
```

  - Per visualizzare un elenco di tutti i parametri disponibili, usare il comando seguente:

```
aws cloudwatch list-metrics
```

Elenco di parametri e dimensioni di CloudWatch per i calcoli Apache Spark in Athena

Se hai abilitato i parametri di CloudWatch nel gruppo di lavoro Athena abilitato per Spark, Athena invia i parametri seguenti a CloudWatch per ciascun gruppo di lavoro. Il parametro utilizza lo spazio dei nomi `AmazonAthenaForApacheSpark`.

Nome parametro	Descrizione
DPUCount	Il numero di unità elaborazione dati (DPU) utilizzate durante la sessione per eseguire i calcoli. Una DPU è una misura relativa della potenza di elaborazione ed è costituita da 4 vCPU di capacità di elaborazione e 16 GB di memoria.

Questo parametro ha le seguenti dimensioni.

Dimensione	Descrizione
SessionId	L'ID della sessione in cui vengono inviati i calcoli.
Gruppo di lavoro	Il nome del gruppo di lavoro.

L'abilitazione del richiedente paga i bucket Amazon S3 in Athena per Spark

Quando un bucket Amazon S3 è configurato per il pagamento a carico del richiedente, all'account dell'utente che esegue la query vengono addebitati i costi di accesso e trasferimento dei dati associati alla richiesta. Per ulteriori informazioni, consulta la sezione Utilizzo dei bucket con pagamento a carico del richiedente per i trasferimenti e l'utilizzo dello storage nella Guida per l'utente di Amazon S3.

In Athena for Spark, i bucket di pagamento del richiedente sono abilitati per sessione, non per gruppo di lavoro. Ad alto livello, l'abilitazione dei bucket requester pay include i seguenti passaggi:

1. Nella console Amazon S3, abilita i pagamenti del richiedente sulle proprietà del bucket e aggiungi una policy del bucket per specificare l'accesso.
2. Nella console IAM, crea una policy IAM per consentire l'accesso al bucket, quindi collega la policy al ruolo IAM che verrà utilizzato per accedere al bucket a pagamento a carico del richiedente.
3. In Athena for Spark, aggiungi una proprietà di sessione per abilitare la funzionalità di pagamento del richiedente.

1. Abilita il pagamento a carico del richiedente su un bucket Amazon S3 e aggiungi una policy sui bucket

Come abilitare il pagamento a carico del richiedente su un bucket Amazon S3

1. Apri la console Amazon S3 su https://console.aws.amazon.com/s3/.
2. Nell'elenco dei bucket scegli il il link per il bucket per il quale vuoi abilitare il pagamento a carico del richiedente.
3. Nella pagina dei bucket scegli la scheda Proprietà.
4. Scorri verso il basso fino alla sezione Pagamento a carico del richiedente e seleziona Modifica.
5. Nella pagina Modifica i pagamenti a carico del richiedente, seleziona Abilita, quindi seleziona Salva modifiche.
6. Scegliere la scheda Permissions (Autorizzazioni).
7. Seleziona Modifica nella sezione Policy bucket.
8. Nella pagina Modifica policy bucket, applica la policy del bucket che desideri al bucket di origine. La seguente policy di esempio fornisce l'accesso a tutti i principali (`"AWS": "*"`) AWS, ma il

tuo accesso può essere più granulare. Ad esempio, potresti voler specificare solo un ruolo IAM specifico in un altro account.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::account_number-us-east-1-my-s3-requester-pays-
bucket",
                "arn:aws:s3:::account_number-us-east-1-my-s3-requester-pays-bucket/
*"
            ]
        }
    ]
}
```

2. Come creare una policy IAM e collegarvi il ruolo IAM.

Quindi, crea una policy IAM per consentire l'accesso a un bucket. Quindi alleghi la policy al ruolo che verrà utilizzato per accedere al bucket con costi a carico del richiedente.

Creare una policy IAM per il richiedente paga il bucket e allegare la policy a un ruolo

1. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione della console IAM seleziona Policy.
3. Scegli Create Policy (Crea policy).
4. Scegli JSON.
5. In Editor di policy, aggiungi una policy come quella che segue:

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

```
        {
            "Action": [
                "s3:*"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::account_number-us-east-1-my-s3-requester-pays-
bucket",
                "arn:aws:s3:::account_number-us-east-1-my-s3-requester-pays-bucket/
*"
            ]
        }
    ]
}
```

6. Seleziona Successivo.
7. Nella pagina Esamina e crea, immetti un nome per la policy e una descrizione facoltativa, quindi seleziona Crea policy.
8. Nel pannello di navigazione, seleziona Ruoli.
9. Nella pagina Ruoli, trova il ruolo che desideri utilizzare, quindi scegli il link al nome del ruolo.
10. Nella sezione Policy di autorizzazioni, seleziona Aggiungi autorizzazioni, Collega policy.
11. Nella sezione Altre policy di autorizzazione, seleziona la casella di controllo relativa alla policy che hai creato, quindi seleziona Aggiungi autorizzazioni.

3. Aggiungi una proprietà di sessione Athena per Spark

Dopo aver configurato il bucket Amazon S3 e le autorizzazioni associate per i pagamenti dei richiedenti, puoi abilitare la funzionalità in una sessione di Athena for Spark.

Per abilitare i bucket requester pay in una sessione di Athena for Spark

1. Nell'editor notebook, nel menu Session (Sessione) in alto a destra, scegli Edit session (Modifica sessione).
2. Espandi le proprietà di Spark.
3. Scegli Modifica in JSON.
4. Nell'editor di testo JSON, immetti quanto segue:

```
{
```

```
  "spark.hadoop.fs.s3.useRequesterPaysHeader":"true"
}
```

5. Seleziona Salva.

Abilitazione della crittografia Apache Spark

Puoi abilitare la crittografia Apache Spark su Athena. In questo modo si crittografano i dati in transito tra i nodi Spark e si crittografano anche i dati a riposo archiviati localmente da Spark. Per migliorare la sicurezza di questi dati, Athena utilizza la seguente configurazione di crittografia:

```
spark.io.encryption.keySizeBits="256"
spark.io.encryption.keygen.algorithm="HmacSHA384"
```

Per abilitare la crittografia Spark, puoi utilizzare la console Athena, la AWS CLI, o l'API Athena.

Utilizzo della console Athena per abilitare la crittografia Spark

Per creare un nuovo notebook con la crittografia Spark abilitata

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.
3. Completa una delle seguenti operazioni:
   - In Notebook explorer, scegli Create notebook (Crea notebook).
   - In Notebook editor (Editor notebook), scegli Create notebook (Crea notebook) oppure seleziona l'icona con il segno più (+) per aggiungere un notebook.
4. In Nome notebook, inserisci un nome per il notebook.
5. Espandi l'opzione delle Proprietà Spark.
6. Seleziona Attiva la crittografia Spark.
7. Seleziona Crea.

La sessione notebook che crei è crittografata. Usa il nuovo notebook come faresti normalmente. Quando in seguito lancerai nuove sessioni che utilizzano il notebook, anche le nuove sessioni verranno crittografate.

Puoi anche utilizzare la console Athena per abilitare la crittografia Spark per un notebook esistente.

Per abilitare la crittografia per un notebook esistente

1. Apri una nuova sessione per un notebook creato in precedenza.
2. Nell'editor notebook, nel menu Session (Sessione) in alto a destra, scegli Edit session (Modifica sessione).
3. Nella finestra di dialogo Modifica i dettagli della sessione, espandi le proprietà Spark.
4. Seleziona Attiva la crittografia Spark.
5. Seleziona Salva.

La console avvia una nuova sessione con la crittografia abilitata. Anche le sessioni successive create per questo notebook avranno la crittografia abilitata.

Come utilizzare AWS CLI per abilitare la crittografia Spark

Puoi utilizzare AWS CLI per abilitare la crittografia all'avvio di una sessione specificando le proprietà Spark appropriate.

Come utilizzare AWS CLI per abilitare la crittografia Spark

1. Utilizzate un comando come il seguente per creare un oggetto JSON di configurazione del motore che specifichi le proprietà di crittografia Spark.

```
ENGINE_CONFIGURATION_JSON=$(
  cat <<EOF
{
    "CoordinatorDpuSize": 1,
    "MaxConcurrentDpus": 20,
    "DefaultExecutorDpuSize": 1,
    "SparkProperties": {
      "spark.authenticate": "true",
      "spark.io.encryption.enabled": "true",
      "spark.network.crypto.enabled": "true"
    }
}
EOF
)
```

2. In AWS CLI, utilizza il comando `athena start-session` e passa l'oggetto JSON creato all'argomento `--engine-configuration`, come nell'esempio seguente:

```
aws athena start-session \
   --region "region" \
   --work-group "your-work-group" \
   --engine-configuration "$ENGINE_CONFIGURATION_JSON"
```

Utilizzo dell'API Athena per abilitare la crittografia Spark

Per abilitare la crittografia Spark con l'API Athena, usa l'azione StartSession e il relativo parametro `SparkProperties` EngineConfiguration per specificare la configurazione di crittografia nella tua richiesta `StartSession`.

Configurazione dell' AWS Glue accesso tra account diversi in Athena per Spark

In questo argomento viene illustrato come configurare l'account consumer *666666666666* e l'account proprietario *999999999999* per l'accesso multi-account su AWS Glue . Quando gli account sono configurati, l'account consumer può eseguire query da Athena for Spark sui database e sulle tabelle AWS Glue del proprietario.

1. Nel AWS Glue, fornisci l'accesso ai ruoli dei consumatori

Nel AWS Glue, il proprietario crea una politica che fornisce ai ruoli del consumatore l'accesso al catalogo AWS Glue dati del proprietario.

Per aggiungere una AWS Glue politica che consenta a un ruolo di consumatore di accedere al catalogo dati del proprietario

1. Accedi a AWS Management Console utilizzando l'account del proprietario del catalogo.
2. Apri la AWS Glue console all'indirizzo https://console.aws.amazon.com/glue/.
3. Nel riquadro di navigazione, espandi Data Catalog, quindi seleziona Impostazioni catalogo.
4. Nella pagina Impostazioni del catalogo dati, nella sezione Autorizzazioni, aggiungi una policy come quella che segue. Questa policy fornisce i ruoli per l'accesso dell'account consumer *666666666666* al catalogo dati dell'account proprietario *999999999999*.

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

```
  "Statement": [
    {
      "Sid": "Cataloguers",
      "Effect": "Allow",
      "Principal": {
        "AWS": [
          "arn:aws:iam::666666666666:role/Admin",
          "arn:aws:iam::666666666666:role/AWSAthenaSparkExecutionRole"
        ]
      },
      "Action": "glue:*",
      "Resource": [
        "arn:aws:glue:us-west-2:999999999999:catalog",
        "arn:aws:glue:us-west-2:999999999999:database/*",
        "arn:aws:glue:us-west-2:999999999999:table/*"
      ]
    }
  ]
}
```

2. Configurazione dell'account consumer per l'accesso

Nell'account consumer, crea una policy per consentire l'accesso al proprietario AWS Glue Data Catalog, ai database e alle tabelle e associala a un ruolo. Nell'esempio seguente viene utilizzato l'account consumer *666666666666*.

Per creare una AWS Glue politica di accesso al proprietario AWS Glue Data Catalog

1. Accedi con l'account consumer a AWS Management Console.
2. Aprire la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
3. Nel riquadro di navigazione, espandi Gestione accesso, quindi seleziona Policy.
4. Scegli Crea policy.
5. Nella pagina Specifica autorizzazioni, seleziona JSON.
6. Nell'editor delle politiche, inserisci un'istruzione JSON come la seguente che consente AWS Glue azioni sul catalogo di dati dell'account proprietario.

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

```
        {
            "Effect": "Allow",
            "Action": "glue:*",
            "Resource": [
                "arn:aws:glue:us-east-1:999999999999:catalog",
                "arn:aws:glue:us-east-1:999999999999:database/*",
                "arn:aws:glue:us-east-1:999999999999:table/*"
            ]
        }
    ]
}
```

7. Seleziona Successivo.
8. Nella pagina Rivedi e crea, per Nome policy inserisci un nome per la policy.
9. Scegli Crea policy.

Successivamente, utilizzi la console IAM nell'account consumer per allegare la policy che hai appena creato a uno o più ruoli IAM che l'account consumer utilizzerà per accedere al catalogo dati del proprietario.

Per allegare la AWS Glue politica ai ruoli nell'account consumatore

1. Nel pannello di navigazione della console IAM dell'account consumer, seleziona Ruoli.
2. Nella pagina Ruoli, individua il ruolo a cui intendi collegare la policy.
3. Seleziona Aggiungi autorizzazioni, quindi seleziona Collega policy.
4. Trova la policy che hai appena creato.
5. Seleziona la casella di controllo della policy, quindi seleziona Aggiungi autorizzazioni.
6. Ripeti i passaggi per aggiungere la policy agli altri ruoli che intendi utilizzare.

3. Configurazione di una sessione e creazione di una query

In Athena Spark, nell'account del richiedente, utilizzando il ruolo specificato, crea una sessione per testare l'accesso creando un notebook o modificando una sessione corrente. Quando configuri le proprietà della sessione, specifica una delle seguenti opzioni:

- Il separatore del catalogo Glue: con questo approccio, includi l'ID dell'account proprietario nelle tue query. Utilizza questo metodo se intendi utilizzare la sessione per eseguire query sui cataloghi di dati di diversa proprietà.

- L'ID del catalogo Glue: con questo approccio, si interroga direttamente il database. Questo metodo è più comodo se intendi utilizzare la sessione per eseguire query sul catalogo dati di una sola proprietà.

Utilizzo dell'approccio del separatore di AWS Glue catalogo

Quando modificate le proprietà della sessione, aggiungete quanto segue:

```
{
    "spark.hadoop.aws.glue.catalog.separator": "/"
}
```

Quando esegui una query in una cella, utilizza una sintassi simile a quella dell'esempio seguente. Tieni presente che nella clausola `FROM`, l'ID del catalogo e il separatore sono obbligatori prima del nome del database.

```
df = spark.sql('SELECT requestip, uri, method, status FROM `999999999999/
mydatabase`.cloudfront_logs LIMIT 5')
df.show()
```

Utilizzo dell'approccio AWS Glue Catalog ID

Quando modificate le proprietà della sessione, inserite la seguente proprietà. Sostituisci *999999999999* con l'ID dell'account del proprietario.

```
{
    "spark.hadoop.hive.metastore.glue.catalogid": "999999999999"
}
```

Quando esegui una query in una cella, utilizza una sintassi come la seguente: Si noti che nella clausola `FROM`, l'ID del catalogo e il separatore non sono richiesti prima del nome del database.

```
df = spark.sql('SELECT * FROM mydatabase.cloudfront_logs LIMIT 10')
df.show()
```

Risorse aggiuntive

Accesso tra account ai cataloghi dati AWS Glue

Gestione delle autorizzazioni per più account utilizzando entrambi AWS Glue e Lake Formation nella AWS Lake Formation Developer Guide.

Configura l'accesso tra account a un account condiviso AWS Glue Data Catalog utilizzando Amazon Athena AWS in Prescriptive Guidance Patterns.

Service quotas per Amazon Athena per Apache Spark

Le service quotas, anche note come limiti, rappresentano il numero massimo di risorse di servizio o operazioni utilizzabili dall'Account AWS. Per ulteriori informazioni sulle service quotas di altri servizi AWS che puoi utilizzare Amazon Athena per Spark, consulta service quotas AWS nella Riferimenti generali di Amazon Web Services.

> **Note**
>
> I nuovi Account AWS potrebbero avere quote iniziali inferiori che possono aumentare nel tempo. Amazon Athena per Apche Spark monitora l'utilizzo dell'account all'interno di ciascuna Regione AWS, quindi aumenta automaticamente le quote in base all'utilizzo. Se i tuoi requisiti superano i limiti indicati, contatta l'assistenza clienti.

La tabella seguente elenca le service quotas per Amazon Athena per Apache Spark.

Nome	Default	Adattabile	Descrizione
Simultaneità delle DPU per Apache Spark	160	No	Il numero massimo di unità di elaborazione dati (DPU) che puoi utilizzare contemporaneamente per i calcoli di Apache Spark per un singolo account nell'attuale Regione AWS. Una DPU è una misura relativa della potenza di elaborazione ed è costituita da 4 vCPU di capacità di elaborazione e 16 GB di memoria.
Simultaneità DPU della sessione Apache Spark	60	No	Il numero massimo di DPU che puoi utilizzare simultaneamente per un calcolo su Apache Spark all'interno di una sessione.

API dei notebook Athena

L'elenco seguente contiene collegamenti di riferimento alle operazioni dell'API per notebook Athena. Per le strutture dei dati e altre operazioni dell'API Athena, consulta la documentazione di riferimento dell'API Amazon Athena.

- CreateNotebook
- CreatePresignedNotebookUrl
- DeleteNotebook
- ExportNotebook
- GetCalculationExecution
- GetCalculationExecutionCode
- GetCalculationExecutionStatus
- GetNotebookMetadata
- GetSession
- GetSessionStatus
- ImportNotebook
- ListApplicationDPUSizes
- ListCalculationExecutions
- ListExecutors
- ListNotebookMetadata
- ListNotebookSessions
- ListSessions
- StartCalculationExecution
- StartSession
- StopCalculationExecution
- TerminateSession
- UpdateNotebook
- UpdateNotebookMetadata

Problemi noti in Athena per Spark

Questa pagina documenta alcuni dei problemi noti di Athena per Apache Spark.

Eccezione di argomento illegale durante la creazione di una tabella

Sebbene Spark non consenta la creazione di database con una proprietà location vuota, i database in AWS Glue possono avere una `LOCATION` proprietà vuota se vengono creati all'esterno di Spark.

Se crei una tabella e specifichi un AWS Glue database con un `LOCATION` campo vuoto, può verificarsi un'eccezione come la seguente IllegalArgumentException: Impossibile creare un percorso da una stringa vuota.

Ad esempio, il comando seguente genera un'eccezione se il database predefinito in AWS Glue contiene un campo `LOCATION` vuoto:

```
spark.sql("create table testTable (firstName STRING)")
```

Soluzione consigliata AWS Glue A: da utilizzare per aggiungere una posizione al database in uso.

Per aggiungere una posizione a un AWS Glue database

1. Accedere AWS Management Console e aprire la AWS Glue console all'indirizzo https://console.aws.amazon.com/glue/.
2. Nel riquadro di navigazione, scegli Databases (Database).
3. Nell'elenco dei database, scegli il nome del database da modificare.
4. Nella pagina dei dettagli per il database, scegli Edit (Modifica).
5. Nella pagina Update a database (Aggiorna un database), in Location (Posizione), inserisci una posizione Amazon S3.
6. Scegli Update Database (Aggiorna database).

Soluzione consigliata B: utilizza un database AWS Glue diverso con una posizione esistente e valida in Amazon S3. Ad esempio, se hai un database denominato `dbWithLocation`, utilizza il comando `spark.sql("use dbWithLocation")` per passare a quel database.

Soluzione consigliata C: quando utilizzi Spark SQL per creare la tabella, specifica un valore per `location`, come nell'esempio seguente.

```
spark.sql("create table testTable (firstName STRING)
       location 's3://DOC-EXAMPLE-BUCKET/'").
```

Soluzione consigliata D: se hai specificato una posizione quando hai creato la tabella ma il problema persiste, assicurati che il percorso Amazon S3 fornito abbia una barra finale. Ad esempio, il comando seguente genera un'eccezione di argomento illegale:

```
spark.sql("create table testTable (firstName STRING)
       location 's3://DOC-EXAMPLE-BUCKET'")
```

Per correggere questo problema, aggiungi una barra finale alla posizione (ad esempio, `'s3://DOC-EXAMPLE-BUCKET/'`).

Database creato in una posizione del gruppo di lavoro

Se si utilizza un comando come `spark.sql('create database db')` per creare un database e non si specifica una posizione per il database, Athena crea una sottodirectory nella posizione del gruppo di lavoro e la utilizza per il database appena creato.

Problemi con le tabelle gestite da Hive nel database AWS Glue predefinito

Se la `Location` proprietà del database predefinito in non AWS Glue è vuota e specifica una posizione valida in Amazon S3 e utilizzi Athena for Spark per creare una tabella gestita Hive nel database predefinito, AWS Glue i dati vengono scritti nella posizione Amazon S3 specificata nel gruppo di lavoro Athena Spark anziché nella posizione specificata dal database. AWS Glue

Questo problema si verifica a causa del modo in cui Apache Hive gestisce il database predefinito. Apache Hive crea dati di tabella nella posizione principale del magazzino Hive, che può essere diversa dall'effettiva posizione predefinita del database.

Quando usi Athena for Spark per creare una tabella gestita Hive nel database predefinito in AWS Glue, i metadati della AWS Glue tabella possono puntare a due posizioni diverse. Ciò può causare un comportamento imprevisto quando si tenta un'operazione `INSERT` o `DROP TABLE`.

I passaggi per riprodurre il problema sono i seguenti:

1. In Athena for Spark, usi uno dei seguenti metodi per creare o salvare una tabella gestita da Hive:
   - Un'istruzione SQL come `CREATE TABLE $tableName`

- Un PySpark comando del `df.write.mode("overwrite").saveAsTable($tableName)` genere non specifica l'`path`opzione nell'API Dataframe.

A questo punto, la AWS Glue console potrebbe mostrare una posizione errata in Amazon S3 per la tabella.

2. In Athena per Spark, utilizza l'istruzione `DROP TABLE $table_name` per eliminare la tabella che hai creato.
3. Dopo aver eseguito l'istruzione `DROP TABLE`, noterai che i file sottostanti in Amazon S3 sono ancora presenti.

Per risolvere il problema, procedi in uno dei seguenti modi:

Soluzione A: utilizza un AWS Glue database diverso quando crei tabelle gestite da Hive.

Soluzione B: specifica un percorso vuoto per il database predefinito in AWS Glue. Quindi, create le tabelle gestite nel database predefinito.

Incompatibilità dei formati di file CSV e JSON tra Athena for Spark e Athena SQL

A causa di un problema noto con Spark open source, quando crei una tabella in Athena per Spark su dati CSV o JSON, la tabella potrebbe non essere leggibile da Athena SQL e viceversa.

Ad esempio, è possibile creare una tabella in Athena per Spark in uno dei seguenti modi:

- Con la seguente sintassi `USING csv`:

```
spark.sql('''CREATE EXTERNAL TABLE $tableName (
$colName1 $colType1,
$colName2 $colType2,
$colName3 $colType3)
USING csv
PARTITIONED BY ($colName1)
LOCATION $s3_location''')
```

- Con la seguente sintassi DataFrameAPI:

```
df.write.format('csv').saveAsTable($table_name)
```

A causa del problema noto con Spark open source, le query di Athena SQL sulle tabelle risultanti potrebbero non avere esito positivo.

Soluzione consigliata: prova a creare la tabella in Athena per Spark usando la sintassi Apache Hive. Per ulteriori informazioni, consulta CREATE HIVEFORMAT TABLE nella documentazione di Apache Spark.

Risoluzione dei problemi di Athena per Spark

Utilizza le seguenti informazioni per risolvere i problemi che potresti riscontrare quando utilizzi notebook e sessioni su Athena.

Argomenti

Risoluzione dei problemi relativi ai gruppi di lavoro abilitati per Spark

Utilizza le informazioni seguenti per risolvere i problemi relativi ai gruppi di lavoro abilitati per Spark in Athena.

La sessione smette di rispondere quando si utilizza un ruolo IAM esistente

Se non ne hai creato un nuovo `AWSAthenaSparkExecutionRole` per il tuo gruppo di lavoro abilitato a Spark e hai invece aggiornato o scelto un ruolo IAM esistente, la tua sessione potrebbe smettere di rispondere. In questo caso, potrebbe essere necessario aggiungere le seguenti policy di attendibilità e autorizzazioni al ruolo di esecuzione del gruppo di lavoro abilitato a Spark.

Aggiungi la seguente policy di attendibilità. La policy prevede un controllo "confused deputy" per il ruolo esecutivo. Sostituisci i valori per *`111122223333`* e *`workgroup-name`* con l' Account AWS ID e Regione AWS il gruppo di lavoro che stai utilizzando. *`aws-region`*

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "athena.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:athena:aws-
region:111122223333:workgroup/workgroup-name"
                }
            }
        }
    ]
}
```

Aggiungi una policy di autorizzazioni come la seguente policy predefinita per i gruppi di lavoro abilitati al notebook. Modifica le posizioni Account AWS e gli ID segnaposto Amazon S3 in modo che corrispondano a quelli che stai utilizzando. Sostituisci i valori per DOC-EXAMPLE-BUCKET, *aws-region*, *111122223333* e *workgroup-name* con il bucket Amazon S3, la Regione AWS, l'ID dell' Account AWS e il gruppo di lavoro che stai utilizzando.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:ListBucket",
                "s3:DeleteObject",
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
```

```
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "athena:GetWorkGroup",
                "athena:CreatePresignedNotebookUrl",
                "athena:TerminateSession",
                "athena:GetSession",
                "athena:GetSessionStatus",
                "athena:ListSessions",
                "athena:StartCalculationExecution",
                "athena:GetCalculationExecutionCode",
                "athena:StopCalculationExecution",
                "athena:ListCalculationExecutions",
                "athena:GetCalculationExecution",
                "athena:GetCalculationExecutionStatus",
                "athena:ListExecutors",
                "athena:ExportNotebook",
                "athena:UpdateNotebook"
            ],
            "Resource": "arn:aws:athena:aws-region:111122223333:workgroup/workgroup-
name"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:CreateLogGroup",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:aws-region:111122223333:log-group:/aws-athena:*",
                "arn:aws:logs:aws-region:111122223333:log-group:/aws-athena*:log-
stream:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "logs:DescribeLogGroups",
            "Resource": "arn:aws:logs:aws-region:111122223333:log-group:*"
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "cloudwatch:PutMetricData"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cloudwatch:namespace": "AmazonAthenaForApacheSpark"
                }
            }
        }
    ]
}
```

Utilizzo dell'istruzione EXPLAIN di Spark per risolvere i problemi relativi a Spark SQL

Puoi utilizzare l'istruzione `EXPLAIN` di Spark con Spark SQL per risolvere i problemi del codice Spark. I seguenti esempi di codice e output mostrano questo utilizzo.

Example - Istruzione SELECT di Spark

```
spark.sql("select * from select_taxi_table").explain(True)
```

Output

```
Calculation started (calculation_id=20c1ebd0-1ccf-ef14-db35-7c1844876a7e) in
(session=24c1ebcb-57a8-861e-1023-736f5ae55386).
Checking calculation status...

Calculation completed.
== Parsed Logical Plan ==
'Project [*]
+- 'UnresolvedRelation [select_taxi_table], [], false

== Analyzed Logical Plan ==
VendorID: bigint, passenger_count: bigint, count: bigint
Project [VendorID#202L, passenger_count#203L, count#204L]
+- SubqueryAlias spark_catalog.spark_demo_database.select_taxi_table
   +- Relation spark_demo_database.select_taxi_table[VendorID#202L,
       passenger_count#203L,count#204L] csv
```

```
== Optimized Logical Plan ==
Relation spark_demo_database.select_taxi_table[VendorID#202L,
passenger_count#203L,count#204L] csv

== Physical Plan ==
FileScan csv spark_demo_database.select_taxi_table[VendorID#202L,
passenger_count#203L,count#204L]
Batched: false, DataFilters: [], Format: CSV,
Location: InMemoryFileIndex(1 paths)
[s3://DOC-EXAMPLE-BUCKET/select_taxi],
PartitionFilters: [], PushedFilters: [],
ReadSchema: struct<VendorID:bigint,passenger_count:bigint,count:bigint>
```

Example - Dataframe Spark

Il codice di esempio seguente mostra come utilizzare `EXPLAIN` con un dataframe Spark.

```
taxi1_df=taxi_df.groupBy("VendorID", "passenger_count").count()
taxi1_df.explain("extended")
```

Output

```
Calculation started (calculation_id=d2c1ebd1-f9f0-db25-8477-3effc001b309) in
(session=24c1ebcb-57a8-861e-1023-736f5ae55386).
Checking calculation status...

Calculation completed.
== Parsed Logical Plan ==
'Aggregate ['VendorID, 'passenger_count],
['VendorID, 'passenger_count, count(1) AS count#321L]
+- Relation [VendorID#49L,tpep_pickup_datetime#50,tpep_dropoff_datetime#51,
passenger_count#52L,trip_distance#53,RatecodeID#54L,store_and_fwd_flag#55,
PULocationID#56L,DOLocationID#57L,payment_type#58L,fare_amount#59,
extra#60,mta_tax#61,tip_amount#62,tolls_amount#63,improvement_surcharge#64,
total_amount#65,congestion_surcharge#66,airport_fee#67] parquet

== Analyzed Logical Plan ==
VendorID: bigint, passenger_count: bigint, count: bigint
Aggregate [VendorID#49L, passenger_count#52L],
[VendorID#49L, passenger_count#52L, count(1) AS count#321L]
+- Relation [VendorID#49L,tpep_pickup_datetime#50,tpep_dropoff_datetime#51,
passenger_count#52L,trip_distance#53,RatecodeID#54L,store_and_fwd_flag#55,
```

```
PULocationID#56L,DOLocationID#57L,payment_type#58L,fare_amount#59,extra#60,
mta_tax#61,tip_amount#62,tolls_amount#63,improvement_surcharge#64,
total_amount#65,congestion_surcharge#66,airport_fee#67] parquet

== Optimized Logical Plan ==
Aggregate [VendorID#49L, passenger_count#52L],
[VendorID#49L, passenger_count#52L, count(1) AS count#321L]
+- Project [VendorID#49L, passenger_count#52L]
   +- Relation [VendorID#49L,tpep_pickup_datetime#50,tpep_dropoff_datetime#51,
passenger_count#52L,trip_distance#53,RatecodeID#54L,store_and_fwd_flag#55,
PULocationID#56L,DOLocationID#57L,payment_type#58L,fare_amount#59,extra#60,
mta_tax#61,tip_amount#62,tolls_amount#63,improvement_surcharge#64,
total_amount#65,congestion_surcharge#66,airport_fee#67] parquet

== Physical Plan ==
AdaptiveSparkPlan isFinalPlan=false
+- HashAggregate(keys=[VendorID#49L, passenger_count#52L], functions=[count(1)],
output=[VendorID#49L, passenger_count#52L, count#321L])
   +- Exchange hashpartitioning(VendorID#49L, passenger_count#52L, 1000),
      ENSURE_REQUIREMENTS, [id=#531]
      +- HashAggregate(keys=[VendorID#49L, passenger_count#52L],
         functions=[partial_count(1)], output=[VendorID#49L,
         passenger_count#52L, count#326L])
         +- FileScan parquet [VendorID#49L,passenger_count#52L] Batched: true,
            DataFilters: [], Format: Parquet,
            Location: InMemoryFileIndex(1 paths)[s3://DOC-EXAMPLE-BUCKET/
            notebooks/yellow_tripdata_2016-01.parquet], PartitionFilters: [],
            PushedFilters: [],
            ReadSchema: struct<VendorID:bigint,passenger_count:bigint>
```

Registrazione degli eventi dell'applicazione Spark in Athena

L'editor notebook Athena consente la registrazione standard per Jupyter, Spark e Python. È possibile utilizzarli `df.show()` per visualizzare PySpark DataFrame i contenuti o `print("Output")` per visualizzare i valori nell'output della cella. I risultati dei calcoli `stdout`, `stderr` e `results` vengono scritti nella posizione del bucket dei risultati delle query in Amazon S3.

Registrazione degli eventi dell'applicazione Spark su Amazon CloudWatch

Le tue sessioni Athena possono anche scrivere log su Amazon CloudWatch nell'account che stai utilizzando.

Comprensione dei flussi di log e dei gruppi di log

CloudWatch organizza l'attività dei log in flussi di log e gruppi di log.

Flussi di log: un flusso di CloudWatch log è una sequenza di eventi di log che condividono la stessa fonte. Ogni fonte separata di log in CloudWatch Logs costituisce un flusso di log separato.

Gruppi di log: in CloudWatch Logs, un gruppo di log è un gruppo di flussi di log che condividono le stesse impostazioni di conservazione, monitoraggio e controllo degli accessi.

Non vi è alcun limite al numero di flussi di log che possono appartenere a un gruppo di log.

In Athena, quando si avvia una sessione di notebook per la prima volta, Athena crea un gruppo di log in CloudWatch cui viene utilizzato il nome del gruppo di lavoro abilitato per Spark, come nell'esempio seguente.

```
/aws-athena/workgroup-name
```

Questo gruppo di log riceve un flusso di log per ogni esecutore della sessione che produce almeno un log eventi. Un esecutore è l'unità di calcolo più piccola che una sessione di notebook può richiedere ad Athena. In CloudWatch, il nome del flusso di log inizia con l'ID della sessione e l'ID dell'esecutore.

Per ulteriori informazioni sui gruppi di CloudWatch log e sui flussi di log, consulta Working with log groups and log stream nella Amazon CloudWatch Logs User Guide.

Utilizzo di oggetti logger standard in Athena per Spark

In una sessione di Athena for Spark, puoi utilizzare i seguenti due oggetti logger standard globali per scrivere log su Amazon: CloudWatch

- athena_user_logger — Invia i log solo a. CloudWatch Usa questo oggetto quando desideri registrare le informazioni direttamente nelle tue applicazioni Spark, come nell'esempio seguente. CloudWatch

  ```
  athena_user_logger.info("CloudWatch log line.")
  ```

  L'esempio scrive un evento di registro CloudWatch simile al seguente:

  ```
  AthenaForApacheSpark: 2022-01-01 12:00:00,000 INFO builtins: CloudWatch log line.
  ```

- athena_shared_logger — Invia lo stesso registro sia a che a destinazione per scopi di supporto. CloudWatch AWS È possibile utilizzare questo oggetto per condividere i log con i team di AWS assistenza per la risoluzione dei problemi, come nell'esempio seguente.

```
athena_shared_logger.info("Customer debug line.")
var = [...some variable holding customer data...]
athena_shared_logger.info(var)
```

L'esempio registra la `debug` riga e il valore della `var` variabile in CloudWatch Logs e invia una copia di ogni riga a. AWS Support

> **Note**
>
> Per motivi di privacy, il codice di calcolo e i risultati non vengono condivisi con. AWS Assicurati che le tue chiamate a `athena_shared_logger` scrivano solo le informazioni che desideri rendere visibili a AWS Support.

I logger forniti scrivono eventi tramite Apache Log4j ed ereditano i livelli di registrazione di questa interfaccia. I valori possibili a livello di log sono `DEBUG`, `ERROR`, `FATAL`, `INFO`, `WARN` o `WARNING`. Puoi utilizzare la funzione denominata corrispondente sul logger per produrre questi valori.

> **Note**
>
> Non ricollegare i nomi `athena_user_logger` o `athena_shared_logger`. In questo modo non è possibile scrivere sugli oggetti di registrazione CloudWatch per il resto della sessione.

Esempio: registrazione degli eventi del notebook su CloudWatch

La procedura seguente mostra come registrare gli eventi dei notebook Athena su Amazon CloudWatch Logs.

Per registrare gli eventi dei notebook Athena su Amazon Logs CloudWatch

1. Segui le istruzioni riportate in Nozioni di base su Apache Spark su Amazon Athena per creare un gruppo di lavoro compatibile con Spark in Athena con un nome univoco. In questo tutorial si utilizza il nome del gruppo di lavoro `athena-spark-example`.

2. Segui i passaggi illustrati in Creazione di un notebook per creare un notebook e avviare una nuova sessione.
3. Nell'editor notebook Athena, in una nuova cella del notebook, inserisci il seguente comando:

```
athena_user_logger.info("Hello world.")
```

4. Esegui la cella.
5. Recupera l'ID della sessione corrente effettuando una delle seguenti operazioni:
   - Visualizza l'output della cella (ad esempio `... session=72c24e73-2c24-8b22-14bd-443bdcd72de4`).
   - In una nuova cella, esegui il comando magico `%session_id`.
6. Salva l'ID della sessione.
7. Con lo stesso Account AWS che usi per eseguire la sessione del notebook, apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
8. Nel riquadro di navigazione della CloudWatch console, scegli Registra gruppi.
9. Nell'elenco dei gruppi di log, scegli il gruppo di log con il nome del gruppo di lavoro Athena abilitato per Spark, come nell'esempio seguente.

```
/aws-athena/athena-spark-example
```

   La sezione Log streams (Flussi di log) contiene un elenco di uno o più collegamenti ai flussi di log per il gruppo di lavoro. Ogni nome del flusso di log contiene l'ID di sessione, l'ID dell'esecutore e l'UUID univoco separati da caratteri di barra.

   Ad esempio, se l'ID della sessione è `5ac22d11-9fd8-ded7-6542-0412133d3177` e l'ID dell'esecutore è `f8c22d11-9fd8-ab13-8aba-c4100bfba7e2`, il nome del flusso di log è simile al seguente esempio.

```
5ac22d11-9fd8-ded7-6542-0412133d3177/f8c22d11-9fd8-ab13-8aba-c4100bfba7e2/f012d7cb-cefd-40b1-90b9-67358f003d0b
```

10. Scegli il collegamento del flusso di log della tua sessione.
11. Nella pagina Log events (Eventi di log), visualizza la colonna Message (Messaggio).

    Il log eventi per la cella che hai eseguito è simile al seguente:

```
AthenaForApacheSpark: 2022-01-01 12:00:00,000 INFO builtins: Hello world.
```

12. Torna all'editor notebook Athena.
13. In una nuova cella, inserisci il seguente codice. Il codice registra una variabile in CloudWatch:

```
x = 6
athena_user_logger.warn(x)
```

14. Esegui la cella.
15. Torna alla pagina Registra eventi della CloudWatch console per lo stesso flusso di log.
16. Il flusso di log ora contiene una voce di log eventi con un messaggio simile al seguente:

```
AthenaForApacheSpark: 2022-01-01 12:00:00,000 WARN builtins: 6
```

Utilizzo di CloudTrail per risolvere i problemi relativi alle chiamate API dei notebook Athena

Per risolvere i problemi relativi alle chiamate API del notebook, è possibile esaminare i log di Athena CloudTrail per indagare sulle anomalie o scoprire le operazioni avviate dagli utenti. Per ulteriori informazioni sull'utilizzo di CloudTrail con Athena, consulta la pagina Registrazione delle chiamate API Amazon Athena con AWS CloudTrail.

I seguenti esempi mostrano voci di log di CloudTrail per le API dei notebook Athena:

- StartSession
- TerminateSession
- ImportNotebook
- UpdateNotebook
- StartCalculationExecution

StartSession

L'esempio seguente mostra il log CloudTrail per un evento StartSession del notebook.

```
{
    "eventVersion": "1.08",
```

```
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLE_PRINCIPAL_ID:alias",
        "arn": "arn:aws:sts::123456789012:assumed-role/Admin/alias",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLE_PRINCIPAL_ID",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-10-14T16:41:51Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-10-14T17:05:36Z",
    "eventSource": "athena.amazonaws.com",
    "eventName": "StartSession",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.10",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36",
    "requestParameters": {
        "workGroup": "notebook-workgroup",
        "engineConfiguration": {
            "coordinatorDpuSize": 1,
            "maxConcurrentDpus": 20,
            "defaultExecutorDpuSize": 1,
            "additionalConfigs": {
                "NotebookId": "b8f5854b-1042-4b90-9d82-51d3c2fd5c04",
                "NotebookIframeParentUrl": "https://us-east-1.console.aws.amazon.com"
            }
        },
        "notebookVersion": "KeplerJupyter-1.x",
        "sessionIdleTimeoutInMinutes": 20,
        "clientRequestToken": "d646ff46-32d2-42f0-94d1-d060ec3e5d78"
    },
    "responseElements": {
```

```
        "sessionId": "a2c1ebba-ad01-865f-ed2d-a142b7451f7e",
        "state": "CREATED"
    },
    "requestID": "d646ff46-32d2-42f0-94d1-d060ec3e5d78",
    "eventID": "b58ce998-eb89-43e9-8d67-d3d8e30561c9",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "athena.us-east-1.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```

TerminateSession

L'esempio seguente mostra il log CloudTrail per un evento TerminateSession del notebook.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLE_PRINCIPAL_ID:alias",
        "arn": "arn:aws:sts::123456789012:assumed-role/Admin/alias",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLE_PRINCIPAL_ID",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-10-14T16:41:51Z",
                "mfaAuthenticated": "false"
            }
```

```
        }
    },
    "eventTime": "2022-10-14T17:21:03Z",
    "eventSource": "athena.amazonaws.com",
    "eventName": "TerminateSession",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.11",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36",
    "requestParameters": {
        "sessionId": "a2c1ebba-ad01-865f-ed2d-a142b7451f7e"
    },
    "responseElements": {
        "state": "TERMINATING"
    },
    "requestID": "438ea37e-b704-4cb3-9a76-391997cf42ee",
    "eventID": "49026c5a-bf58-4cdb-86ca-978e711ad238",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "athena.us-east-1.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```


ImportNotebook

L'esempio seguente mostra il log CloudTrail per un evento ImportNotebook del notebook. Per motivi di sicurezza, alcuni contenuti sono nascosti.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLE_PRINCIPAL_ID:alias",
        "arn": "arn:aws:sts::123456789012:assumed-role/Admin/alias",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
```

```
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLE_PRINCIPAL_ID",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-10-14T16:41:51Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-10-14T17:08:54Z",
    "eventSource": "athena.amazonaws.com",
    "eventName": "ImportNotebook",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.12",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/106.0.0.0 Safari/537.36",
    "requestParameters": {
        "workGroup": "notebook-workgroup",
        "name": "example-notebook-name",
        "payload": "HIDDEN_FOR_SECURITY_REASONS",
        "type": "IPYNB",
        "contentMD5": "HIDDEN_FOR_SECURITY_REASONS"
    },
    "responseElements": {
        "notebookId": "05f6225d-bdcc-4935-bc25-a8e19434652d"
    },
    "requestID": "813e777f-6dac-41f4-82a7-e99b7b33f319",
    "eventID": "4abec837-143b-4458-9c1f-fa9fb88ab69b",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "athena.us-east-1.amazonaws.com"
    },
```

```
    "sessionCredentialFromConsole": "true"
}
```

UpdateNotebook

L'esempio seguente mostra il log CloudTrail per un evento UpdateNotebook del notebook. Per motivi di sicurezza, alcuni contenuti sono nascosti.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLE_PRINCIPAL_ID:AthenaExecutor-9cc1ebb2-aac5-
b1ca-8247-5d827bd8232f",
        "arn": "arn:aws:sts::123456789012:assumed-role/AWSAthenaSparkExecutionRole-
om0yj71w5l/AthenaExecutor-9cc1ebb2-aac5-b1ca-8247-5d827bd8232f",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLE_PRINCIPAL_ID",
                "arn": "arn:aws:iam::123456789012:role/service-role/
AWSAthenaSparkExecutionRole-om0yj71w5l",
                "accountId": "123456789012",
                "userName": "AWSAthenaSparkExecutionRole-om0yj71w5l"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-10-14T16:48:06Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-10-14T16:52:22Z",
    "eventSource": "athena.amazonaws.com",
    "eventName": "UpdateNotebook",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.13",
    "userAgent": "Boto3/1.24.84 Python/3.8.14 Linux/4.14.225-175.364.amzn2.aarch64
 Botocore/1.27.84",
    "requestParameters": {
        "notebookId": "c87553ff-e740-44b5-884f-a70e575e08b9",
```

```
        "payload": "HIDDEN_FOR_SECURITY_REASONS",
        "type": "IPYNB",
        "contentMD5": "HIDDEN_FOR_SECURITY_REASONS",
        "sessionId": "9cc1ebb2-aac5-b1ca-8247-5d827bd8232f"
    },
    "responseElements": null,
    "requestID": "baaba1d2-f73d-4df1-a82b-71501e7374f1",
    "eventID": "745cdd6f-645d-4250-8831-d0ffd2fe3847",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "athena.us-east-1.amazonaws.com"
    }
}
```

StartCalculationExecution

L'esempio seguente mostra il log CloudTrail per un evento StartCalculationExecution del notebook. Per motivi di sicurezza, alcuni contenuti sono nascosti.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLE_PRINCIPAL_ID:AthenaExecutor-9cc1ebb2-aac5-
b1ca-8247-5d827bd8232f",
        "arn": "arn:aws:sts::123456789012:assumed-role/AWSAthenaSparkExecutionRole-
om0yj71w5l/AthenaExecutor-9cc1ebb2-aac5-b1ca-8247-5d827bd8232f",
        "accountId": "123456789012",
        "accessKeyId": "EXAMPLE_KEY_ID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLE_PRINCIPAL_ID",
                "arn": "arn:aws:iam::123456789012:role/service-role/
AWSAthenaSparkExecutionRole-om0yj71w5l",
                "accountId": "123456789012",
                "userName": "AWSAthenaSparkExecutionRole-om0yj71w5l"
```

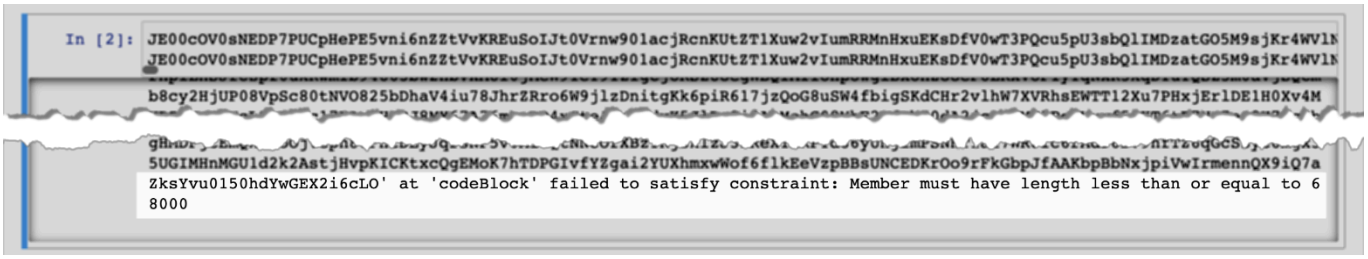
```
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-10-14T16:48:06Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-10-14T16:52:37Z",
    "eventSource": "athena.amazonaws.com",
    "eventName": "StartCalculationExecution",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.14",
    "userAgent": "Boto3/1.24.84 Python/3.8.14 Linux/4.14.225-175.364.amzn2.aarch64
 Botocore/1.27.84",
    "requestParameters": {
        "sessionId": "9cc1ebb2-aac5-b1ca-8247-5d827bd8232f",
        "description": "Calculation started via Jupyter notebook",
        "codeBlock": "HIDDEN_FOR_SECURITY_REASONS",
        "clientRequestToken": "0111cd63-4fd0-4ad8-a738-fd350115fc21"
    },
    "responseElements": {
        "calculationExecutionId": "82c1ebb4-bd08-e4c3-5631-a662fb2ff2c5",
        "state": "CREATING"
    },
    "requestID": "1a107461-3f1b-481e-b8a2-7fbd524e2373",
    "eventID": "b74dbd00-e839-4bd1-a1da-b75fbc70ab9a",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "athena.us-east-1.amazonaws.com"
    }
}
```

Superamento del limite di dimensione del blocco di codice 68k

Athena for Spark ha un limite noto di dimensione del blocco del codice di calcolo di 68000 caratteri. Quando esegui un calcolo con un blocco di codice superiore a questo limite, puoi ricevere il seguente messaggio di errore:

'...' in 'CodeBlock' non è riuscito a soddisfare il vincolo: il member deve avere una lunghezza inferiore o uguale a 68000

L'immagine seguente mostra questo errore nell'editor di notebook della console Athena.



Lo stesso errore può verificarsi quando si utilizza AWS CLI per eseguire un calcolo con un blocco di codice di grandi dimensioni, come nell'esempio seguente.

```
aws athena start-calculation-execution \
    --session-id "{SESSION_ID}" \
    --description "{SESSION_DESCRIPTION}" \
    --code-block "{LARGE_CODE_BLOCK}"
```

Il comando restituisce il seguente messaggio di errore:

{LARGE_CODE_BLOCK} in 'codeBlock' non è riuscito a soddisfare il vincolo: il member deve avere una lunghezza inferiore o uguale a 68000

Soluzione alternativa

Per risolvere il problema, carica il file contenente la query o il codice di calcolo su Amazon S3. Quindi, usa boto3 per leggere il file ed eseguire il codice o il codice SQL.

Gli esempi seguenti presuppongono che tu abbia già caricato il file contenente la tua query SQL o il codice Python su Amazon S3.

Esempio SQL

Il codice di esempio seguente legge il file `large_sql_query.sql` da un bucket Amazon S3 e quindi esegue la query di grandi dimensioni che il file contiene.

```
s3 = boto3.resource('s3')
def read_s3_content(bucket_name, key):
    response = s3.Object(bucket_name, key).get()
    return response['Body'].read()

# SQL
sql = read_s3_content('bucket_name', 'large_sql_query.sql')
df = spark.sql(sql)
```

Esempio di PySpark

Il seguente esempio di codice legge il file `large_py_spark.py` da Amazon S3 e quindi esegue il blocco di codice di grandi dimensioni contenuto nel file.

```
s3 = boto3.resource('s3')

def read_s3_content(bucket_name, key):
    response = s3.Object(bucket_name, key).get()
    return response['Body'].read()

# PySpark
py_spark_code = read_s3_content('bucket_name', 'large_py_spark.py')
exec(py_spark_code)
```

Risoluzione dei problemi delle sessioni

Utilizza le informazioni in questo argomento per risolvere i problemi relativi alle sessioni.

Sessione in stato non integro

Se ricevi il messaggio di errore Session in unhealthy state. Please create a new session (Sessione in stato non integro. Crea una nuova sessione), termina la sessione esistente e creane una nuova.

Impossibile stabilire una connessione al server del notebook

Quando apri un notebook, potresti ricevere il seguente messaggio di errore:

```
A connection to the notebook server could not be established.
The notebook will continue trying to reconnect.
Check your network connection or notebook server configuration.
```

Causa

Quando apre un notebook, Athena crea una sessione e si connette al notebook utilizzando un URL del notebook prefirmato. La connessione al notebook utilizza il protocollo WSS (WebSocketSecure).

Questo errore può verificarsi per i seguenti motivi:

- Un firewall locale (ad esempio un firewall a livello aziendale) sta bloccando il traffico WSS.
- Il software proxy o antivirus sul computer locale sta bloccando la connessione WSS.

Soluzione

Supponiamo di avere una connessione WSS nella Regione `us-east-1` come la seguente:

```
wss://94c2bcdf-66f9-4d17-9da6-7e7338060183.analytics-gateway.us-east-1.amazonaws.com/
api/kernels/33c78c82-b8d2-4631-bd22-1565dc6ec152/channels?session_id=
7f96a3a048ab4917b6376895ea8d7535
```

Per risolvere l'errore, utilizza una delle seguenti strategie.

- Utilizza la sintassi del pattern wild card per consentire l'elenco del traffico WSS sulla porta `443` attraverso e. Regioni AWS Account AWS

  ```
  wss://*amazonaws.com
  ```

- Utilizza la sintassi del pattern wild card per consentire di elencare il traffico WSS sulla porta `443` in una sola porta Regione AWS e tra quelle Account AWS specificate Regione AWS . Nell'esempio seguente viene utilizzato `us-east-1`.

  ```
  wss://*analytics-gateway.us-east-1.amazonaws.com
  ```

Risoluzione dei problemi relativi alle tabelle

Impossibile creare un errore di percorso durante la creazione di una tabella

Messaggio di errore IllegalArgumentException: Impossibile creare un percorso da una stringa vuota.

Causa: Questo errore può verificarsi quando si utilizza Apache Spark in Athena per creare una tabella in un AWS Glue database e il database ha una proprietà vuota. `LOCATION`

Soluzione consigliata: per ulteriori informazioni e soluzioni, consulta la pagina Eccezione di argomento illegale durante la creazione di una tabella.

AccessDeniedException quando si eseguono interrogazioni su tabelle AWS Glue

Messaggio di errore: pyspark.sql.utils. AnalysisException: Impossibile verificare l'esistenza del database predefinito: com.amazonaws.services.glue.model. AccessDeniedException*: User: arn:aws:sts: ::assumed-role/ aws-account-id- AWSAthenaSparkExecutionRole unique-identifier/- AthenaExecutor unique-identifier non è autorizzato a eseguire: glue: on GetDatabase resource: arn:aws:glue: aws-region ::catalog aws-account-idperché nessuna politica basata sull'identità consente il glue: action (Service:; Codice di stato: 400; Codice di errore:; Request ID: request-id; Proxy: null) GetDatabase AWSGlue AccessDeniedException*

Causa: al ruolo di esecuzione per il tuo gruppo di lavoro abilitato a Spark mancano le autorizzazioni per accedere alle AWS Glue risorse.

Soluzione consigliata: per risolvere questo problema, concedi al ruolo di esecuzione l'accesso alle AWS Glue risorse, quindi modifica la policy del bucket di Amazon S3 per concedere l'accesso al ruolo di esecuzione.

La seguente procedura descrive in modo più dettagliato questi passaggi.

Per concedere al ruolo di esecuzione l'accesso alle risorse AWS Glue

1. Aprire la console Athena all'indirizzo https://console.aws.amazon.com/athena/.
2. Se il pannello di navigazione della console non è visibile, scegli il menu di espansione a sinistra.

3. Nel pannello di navigazione della console Athena, scegli Workgroups (Gruppi di lavoro).
4. Nella pagina Workgroups (Gruppi di lavoro), scegli il collegamento del gruppo di lavoro che desideri visualizzare.
5. Nella pagina Overview Details (Dettagli della panoramica) del gruppo di lavoro, scegli il collegamento Role ARN (ARN del ruolo). Il link apre il ruolo di esecuzione Spark nella console IAM.
6. Nella sezione Permissions policies (Policy autorizzazioni), scegli il nome della policy del ruolo con link.
7. Scegli Edit policy (Modifica policy), quindi scegli JSON.
8. Aggiungi AWS Glue l'accesso al ruolo. In genere, si aggiungono le autorizzazioni per le operazioni `glue:GetDatabase` e `glue:GetTable`. Per ulteriori informazioni sulla configurazione dei ruoli IAM, consulta la pagina Adding and removing IAM identity permissions (Aggiunta e rimozione di autorizzazioni per identità IAM) nella Guida per l'utente di IAM.
9. Scegli Review policy (Esamina policy) e quindi Save changes (Salva modifiche).
10. Modifica la policy del bucket Amazon S3 per concedere l'accesso al ruolo di esecuzione. Nota che devi concedere al ruolo l'accesso sia al bucket sia agli oggetti al suo interno. Per i passaggi dettagliati, consulta la pagina Adding a bucket policy using the Amazon S3 console (Aggiunta di una policy del bucket utilizzando la console Amazon S3) nella Guida per l'utente di Amazon Simple Storage Service.

Ottenere supporto

Per ricevere assistenza da AWS, scegli Support, Support Center dal AWS Management Console. Per semplificare l'esperienza, tieni a portata di mano le seguenti informazioni:

- ID della query Athena
- ID sessione
- ID del calcolo

Note di rilascio

Descrive i miglioramenti apportati alle funzioni di Amazon Athena e le correzioni di bug eseguite dalla data di rilascio.

Argomenti

Note di rilascio di Athena per il 2024

26 giugno 2024

Pubblicato il 26/06/2020

La capacità fornita è ora generalmente disponibile nelle regioni Sud America (San Paolo) ed Europa (Spagna). La capacità fornita consente di eseguire query SQL su una capacità di elaborazione completamente gestita e offre funzionalità di gestione dei carichi di lavoro che consentono di stabilire le priorità, controllare e scalare i carichi di lavoro interattivi più importanti. Puoi aggiungere capacità in qualsiasi momento per aumentare il numero di query eseguite contemporaneamente, controllare quali carichi di lavoro utilizzano la capacità e condividere la capacità tra i carichi di lavoro.

Per ulteriori informazioni, consulta Gestione della capacità di elaborazione delle query. Per informazioni sui prezzi, consulta la pagina dei Prezzi di Amazon Athena.

26 aprile 2024

Pubblicato il 26/04/2020

Athena rilascia la versione 3.2.0 del driver JDBC. Per ulteriori informazioni su questa versione del driver, consulta. Note di rilascio di Amazon Athena JDBC 3.x Per scaricare il driver JDBC 3.x, consulta Download del driver JDBC 3.x.

24 aprile 2024

Pubblicato il 2024-04-24

Athena annuncia le correzioni e i miglioramenti seguenti.

- Parquet — Athena ora supporta letture retrocompatibili in Parquet per campi primitivi ripetuti e non annotati che non sono contenuti in un elenco o in un gruppo di mappe. Questa modifica impedisce la restituzione silenziosa di risultati errati e migliora la messaggistica di errore in caso di mancata corrispondenza dello schema.

  Per ulteriori informazioni, consulta Support per letture retrocompatibili per campi primitivi ripetuti non annotati in Parquet su .com. GitHub
- Iceberg OPTIMIZE — Risolto un problema relativo alle `OPTIMIZE` query che causava la perdita di dati quando in una clausola veniva utilizzato un filtro a chiave non di partizione. `WHERE` Per ulteriori informazioni, consulta OPTIMIZE.

16 aprile 2024

Pubblicato il 16/04/2020

Usa la nuova funzionalità passthrough di query federate di Amazon Athena per eseguire intere query direttamente sulla fonte di dati sottostante. Le query passthrough federate ti aiutano a sfruttare le funzioni uniche, il linguaggio di interrogazione e le capacità prestazionali della fonte di dati originale. Ad esempio, è possibile eseguire query Athena su DynamoDB utilizzando il linguaggio PartiQL. Le query passthrough federate sono utili anche quando desideri eseguire `SELECT` query che aggregano, uniscono o richiamano funzioni dell'origine dati che non sono disponibili in Athena. L'utilizzo di query passthrough può ridurre la quantità di dati elaborati da Athena e ridurre i tempi di interrogazione.

Per ulteriori informazioni, consulta Esecuzione di query passthrough federate. Per aggiornare i connettori attualmente in uso alla versione più recente, consulta. Aggiornamento di un connettore origine dati

10 aprile 2024

Pubblicato il 10-04-2020

Athena annuncia la seguenti funzionalità e miglioramenti.

Driver ODBC 1.2.3.1000

Rilascio del driver ODBC 1.2.3.1000 per Athena.

Problemi risolti:

- Problema di connessione al server proxy: quando è stato utilizzato un server proxy senza il certificato principale, il connettore non è riuscito a stabilire una connessione.

Per ulteriori informazioni e per scaricare il driver ODBC 1.x, le note di rilascio e la documentazione, vedere. Driver ODBC 1.x di Athena

Driver JDBC 2.1.5

Rilascio del driver JBDC 2.1.5 per Athena.

Aggiornamenti e miglioramenti:

- AWS Java SDK è stato aggiornato per utilizzare la versione 1.12.687.
- Librerie Jackson aggiornate per utilizzare la versione 2.16.0.
- Librerie Logback aggiornate per utilizzare la versione 1.3.14.

Per ulteriori informazioni e per scaricare il driver JDBC 2.x, le note di rilascio e la documentazione, consulta. Driver JDBC 2.x di Athena

8 aprile 2024

Pubblicato il 2024-04-08

Athena annuncia la versione 2.0.3.0 del driver ODBC. Per ulteriori informazioni, consulta le note di rilascio di 2,03,0. Per scaricare il nuovo driver ODBC v2, consulta Come scaricare il driver ODBC 2.x. Per informazioni sulla connessione, consulta Amazon Athena ODBC 2.x.

15 marzo 2024

Pubblicato il 18/03/2020

Amazon Athena annuncia la disponibilità di Athena SQL nella regione Canada occidentale (Calgary).

Per un elenco completo dei servizi Servizi AWS disponibili in ciascuna regione Regione AWS, consulta AWS Servizi per regione.

15 febbraio 2024

Pubblicato il 15/02/2020

Athena rilascia la versione 3.1.0 del driver JDBC.

La versione 3.1.0 del driver JDBC di Amazon Athena aggiunge il supporto per l'autenticazione integrata di Windows di Microsoft Active Directory Federation Services (AD FS) e l'autenticazione basata su moduli. La versione 3.1.0 include anche altri miglioramenti minori e correzioni di bug.

Per scaricare il driver JDBC v3, vedere. Download del driver JDBC 3.x

31 gennaio 2024

Pubblicato il 31/01/2020

Athena annuncia la seguenti funzionalità e miglioramenti.

- Aggiornamento Hudi — È ora possibile utilizzare Athena SQL per interrogare le tabelle Hudi 0.14.0. Per informazioni sull'utilizzo di Athena SQL per interrogare le tabelle Hudi, vedere. Utilizzo di Athena per eseguire query sui set di dati Apache Hudi

Note di rilascio di Athena per il 2023

14 dicembre 2023

Data di pubblicazione: 14/12/2023

Athena annuncia le correzioni e i miglioramenti seguenti.

Athena rilascia la versione 2.1.3 del driver JDBC. Il driver risolve i seguenti problemi:

- La registrazione è stata migliorata per evitare conflitti con la registrazione delle applicazioni Spring Boot e Gradle.
- Utilizzando il metodo `executeBatch()` di JDBC per inserire i record, il driver inseriva erroneamente solo un record. Poiché Athena non supporta l'esecuzione di query in batch, il driver ora segnala un errore durante l'utilizzo di `executeBatch()`. Per ovviare alla limitazione, è possibile inviare singole query in un ciclo.

Per scaricare il nuovo driver JDBC, le note sul rilascio e la documentazione, consulta Driver JDBC 2.x di Athena.

9 dicembre 2023

Data di pubblicazione: 09/12/2023

Rilasciato il driver ODBC versione 1.2.1.1000 per Athena.

Funzionalità e miglioramenti:

- Supporto RStudio aggiornato: il driver ODBC ora supporta RStudio su macOS.
- Supporto per un unico catalogo e schema: il connettore può ora restituire un unico catalogo e schema. Per ulteriori informazioni, consulta la guida all'installazione e alla configurazione, che è possibile scaricare.

Problemi risolti:

- Istruzioni preparate: quando venivano eseguite istruzioni preparate con un array di parametri utilizzando uno schema a colonne, il connettore restituiva un risultato di query errato.
- Dimensione della colonna: quando era selezionata la colonna di sistema `$file_modified_time`, il connettore restituiva una dimensione di colonna errata.
- SQLPrepare: durante l'associazione dei parametri relativi a `SQLPrepare` nelle query `SELECT`, il connettore restituiva un errore.

Per maggiori informazioni e per scaricare i nuovi driver, le note di rilascio e la documentazione, consulta Driver ODBC 1.x di Athena.

7 dicembre 2023

Data di pubblicazione: 07/12/2023

Athena annuncia il driver ODBC versione 2.0.2.1. Per ulteriori informazioni, consulta le note di rilascio di 2.0.2.1. Per scaricare il nuovo driver ODBC v2, consulta Come scaricare il driver ODBC 2.x. Per informazioni sulla connessione, consulta Amazon Athena ODBC 2.x.

5 dicembre 2023

Data di pubblicazione: 05/12/2023

È ora possibile creare gruppi di lavoro Athena SQL che utilizzano AWS IAM Identity Center la modalità di autenticazione. Questi gruppi di lavoro supportano la funzionalità di propagazione delle identità attendibili di Centro identità IAM. La propagazione affidabile delle identità consente di utilizzare le identità tra servizi di AWS analisi come Amazon Athena e Amazon EMR Studio.

Per ulteriori informazioni, consulta Utilizzo dei gruppi di lavoro Athena abilitati per Centro identità IAM.

28 novembre 2023

Data di pubblicazione: 28/11/2023

Ora è possibile interrogare i dati nella classe di archiviazione Amazon S3 Express One Zone per ottenere rapidamente i risultati delle query. S3 Express One Zone è una classe di archiviazione a singola zona di disponibilità ad alte prestazioni, creata appositamente per fornire un accesso coerente ai dati nell'ordine dei millisecondi per i dati a cui si accede più di frequente e le applicazioni sensibili alla latenza. Per un'esperienza di interrogazione ottimale in Athena, per prima cosa sposta i dati nell'archiviazione S3 Express One Zone e cataloga i dati con AWS Glue Data Catalog.

Per ulteriori informazioni, consulta Interrogazione dei dati di S3 Express One Zone.

27 novembre 2023

Data di pubblicazione: 27/11/2023

Athena annuncia la seguenti funzionalità e miglioramenti.

- Visualizzazioni del catalogo dati di Glue: le viste del catalogo dati di Glue offrono un'unica visualizzazione comune per AWS servizi come Amazon Athena e Amazon Redshift. Nelle viste di Catalogo dati di Glue, le autorizzazioni di accesso sono definite dall'utente che ha creato la vista anziché da quello che la interroga. Queste viste offrono un maggiore controllo degli accessi e una

maggiore sicurezza, contribuiscono a garantire la completezza dei record e possono impedire l'accesso alle tabelle sottostanti.

Per ulteriori informazioni, consulta Usare le viste AWS Glue Data Catalog.

- CloudTrail Supporto Lake: ora puoi utilizzare Amazon Athena per analizzare i dati in AWS CloudTrail Lake. AWS CloudTrail Lake è un data lake gestito CloudTrail che puoi utilizzare per aggregare, archiviare e analizzare i registri delle attività in modo immutabile per controlli, sicurezza e indagini operative. Per interrogare i registri delle attività di CloudTrail Lake da Athena, non è necessario spostare i dati o creare pipeline di elaborazione dati separate. Non sono richieste operazioni ETL.

  Per iniziare, abilita la federazione dei dati in Lake. CloudTrail Quando condividi i metadati CloudTrail del tuo Lake Event Data Store con AWS Glue Data Catalog, CloudTrail crea le AWS Glue Data Catalog risorse necessarie e registra i dati con. AWS Lake Formation In Lake Formation, puoi specificare gli utenti e i ruoli che possono utilizzare Athena per interrogare il tuo datastore di eventi.

  Per ulteriori informazioni, consulta la pagina Enable Lake Query Federation nella Guida per l'utente di AWS CloudTrail .

17 novembre 2023

Data di pubblicazione: 17/11/2023

Athena annuncia la seguenti funzionalità e miglioramenti.

Funzionalità

- Ottimizzatore basato sui costi: Athena annuncia la disponibilità generale dell'ottimizzazione basata sui costi utilizzando le statistiche di. AWS Glue Per ottimizzare le query in Athena SQL, puoi richiedere che Athena raccolga statistiche a livello di tabella o colonna per una delle tue tabelle in AWS Glue. Se tutte le tabelle della query contengono statistiche, Athena le utilizza per esaminare piani di esecuzione alternativi e selezionare quello che è più probabile sia il più veloce.

  Per ulteriori informazioni, consulta Utilizzo dell'ottimizzatore basato sui costi.
- Integrazione con Amazon EMR Studio: ora puoi usare Athena in Amazon EMR Studio senza dover usare direttamente la console Athena. Con l'integrazione di Athena in Amazon EMR, puoi svolgere le seguenti attività:

- Eseguire query SQL Athena
- Visualizzazione dei risultati della query
- Visualizzazione della cronologia delle query
- Visualizzare le query salvate
- Eseguire le query parametrizzate
- Visualizzare database, tabelle e viste per un catalogo di dati

Per ulteriori informazioni, consulta Amazon EMR Studio nell'argomento Servizio AWS integrazioni con Athena.

- Controllo degli accessi nidificati: Athena annuncia il supporto per il controllo degli accessi di Lake Formation per i dati nidificati. In Lake Formation, è possibile definire e applicare filtri di dati su colonne nidificate con tipi di dati `struct`. È possibile utilizzare il filtro dei dati per limitare l'accesso degli utenti alle sottostrutture delle colonne nidificate. Per ulteriori informazioni su come creare un filtro di dati, consulta Creazione di un filtro di dati nella Guida per gli sviluppatori di AWS Lake Formation .
- Metriche di utilizzo della capacità assegnata: Athena annuncia nuove CloudWatch metriche per le prenotazioni di capacità. Puoi utilizzare i nuovi parametri per tenere traccia del numero di DPU che hai assegnato e del numero di DPU utilizzate dalle tue query. Al termine delle query, puoi anche visualizzare il numero di DPU utilizzate dalla query.

  Per ulteriori informazioni, consulta Monitoraggio delle query Athena con metriche CloudWatch .

Miglioramenti

- Modifica del messaggio di errore: il messaggio di errore `Insufficient Lake Formation permissions` ora riporta `Table not found` o `Schema not found`. Questa modifica è stata apportata per impedire a malintenzionati di dedurre l'esistenza di risorse di tabelle o database dal messaggio di errore.

16 novembre 2023

Data pubblicazione: 16/11/2023

Athena ha rilasciato un nuovo driver JDBC che migliora l'esperienza di connessione, esecuzione di quey e visualizzazione dei dati da applicazioni di sviluppo SQL e di business intelligence compatibili.

Il nuovo driver è semplice da aggiornare. Il driver può leggere i risultati delle query direttamente da Amazon S3, in modo da renderli disponibili prima.

Per ulteriori informazioni, consulta Driver JDBC 3.x di Athena.

31 ottobre 2023

Data di pubblicazione: 31/10/2023

Amazon Athena annuncia prenotazioni di 1 ora per la capacità assegnata. A partire da oggi, puoi prenotare e rilasciare la capacità fornita dopo un'ora. Questa modifica semplifica l'ottimizzazione dei costi per i carichi di lavoro la cui domanda cambia nel tempo.

La capacità assegnata offre funzionalità di gestione dei carichi di lavoro che aiutano a stabilire le priorità, controllare e dimensionare i carichi di lavoro interattivi più importanti. Puoi aggiungere capacità in qualsiasi momento per aumentare il numero di query eseguite contemporaneamente, controllare quali carichi di lavoro utilizzano la capacità e condividere la capacità tra i carichi di lavoro.

Per ulteriori informazioni, consulta Gestione della capacità di elaborazione delle query. Per informazioni sui prezzi, consulta la pagina Prezzi di Amazon Athena.

25 ottobre 2023

Data di pubblicazione: 26/10/2023

Athena annuncia le correzioni e i miglioramenti seguenti.

Pacchetto jackson-core: il testo JSON con un valore numerico superiore a 1.000 caratteri ora riporterà un errore. Questa correzione risolve il problema di sicurezza sonatype-2022-6438.

17 ottobre 2023

Data di pubblicazione: 17/10/2023

Athena annuncia il driver ODBC versione 2.0.2.0. Per ulteriori informazioni, consulta le note di rilascio di 2,02,0. Per scaricare il nuovo driver ODBC v2, consulta Come scaricare il driver ODBC 2.x. Per informazioni sulla connessione, consulta Amazon Athena ODBC 2.x.

26 settembre 2023

Data di pubblicazione: 26/09/2023

Athena annuncia la seguenti funzionalità e miglioramenti.

- Supporto di lettura Lake Formation per tabelle Delta Lake. Per informazioni sull'utilizzo delle tabelle Delta Lake con Athena, consulta Esecuzione di query sulle tabelle Delta Lake di Linux Foundation.

23 agosto 2023

Data di pubblicazione: 23/8/2023

Amazon Athena annuncia la disponibilità di Athena SQL nella regione di Israele (Tel Aviv).

Per un elenco completo delle opzioni Servizi AWS disponibili in ciascuna regione Regione AWS, consulta Servizi per regione.AWS

10 agosto 2023

Data di pubblicazione: 10/08/2023

Athena annuncia le correzioni e i miglioramenti seguenti.

Versione del driver ODBC 2.0.1.1

Athena annuncia il driver ODBC versione 2.0.1.1. Per ulteriori informazioni, consulta le note di rilascio di 2,01.1. Per scaricare il nuovo driver ODBC v2, consulta Come scaricare il driver ODBC 2.x. Per informazioni sulla connessione, consulta Amazon Athena ODBC 2.x.

Versione 2.1.1 del driver JDBC

Athena rilascia la versione 2.1.1 del driver JDBC. Il driver risolve i seguenti problemi:

- Un errore che si è verificato quando è stata creata una tabella con una dichiarazione contenente un'espressione regolare.
- Un problema che ha causato l'applicazione errata del parametro di connessione `ApplicationName`.

Per scaricare il nuovo driver JDBC, le note sul rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC.

31 luglio 2023

Data di pubblicazione: 31/07/2023

Amazon Athena annuncia la disponibilità di Athena SQL in ulteriori Regioni AWS.

Questo rilascio amplia la disponibilità di Athena SQL per includere Asia Pacifico (Hyderabad), Asia Pacifico (Melbourne), Europa (Spagna) ed Europa (Zurigo).

Per un elenco completo dei servizi Servizi AWS disponibili in ciascuna regione Regione AWS, consulta AWS Servizi per regione.

27 luglio 2023

Data di pubblicazione: 27/07/2023

Athena rilascia la versione BigQuery 2023.30.1 di Google Connector. Questa versione del connettore riduce i tempi di esecuzione delle query e aggiunge il supporto per l'esecuzione di query su endpoint privati. BigQuery

Per informazioni sul BigQuery connettore Google, consulta. Connettore Google Amazon Athena BigQuery Per informazioni sull'aggiornamento dei connettori esistenti di origine dati, consulta Aggiornamento di un connettore origine dati.

24 luglio 2023

Data di pubblicazione: 24/07/2023

Athena annuncia le correzioni e i miglioramenti seguenti.

- Query con unioni: sono state migliorate le prestazioni di alcune query con le unioni.
- Join con confronti tra tipi: è stato risolto un potenziale errore di query per le dichiarazioni `JOIN` che includevano un confronto tra due tipi diversi.
- Sottoquery su colonne annidate: è stato risolto un problema relativo agli errori delle query quando le sottoquery erano correlate su colonne annidate.
- Visualizzazioni Iceberg: è stato risolto un problema di compatibilità con la precisione di colonne con timestamp nelle viste di Apache Iceberg. Le viste Iceberg con colonne timestamp sono ora leggibili indipendentemente dal fatto che le colonne siano state create sulla versione 2 del motore Athena o sulla versione 3 del motore Athena.

20 luglio 2023

Data di pubblicazione: 20/07/2023

Athena rilascia la versione 2.1.0 del driver JDBC. Il driver include nuovi miglioramenti e ha risolto un problema.

Miglioramenti

Le seguenti librerie di parser JSON Jackson sono state aggiornate:

- jackson-annotations 2.15.2 (in precedenza 2.14.0)
- jackson-core 2.15.2 (in precedenza 2.14.0)
- jackson-databind 2,15.2 (in precedenza 2,14.0)

Problemi risolti

- È stato risolto un problema relativo al passaggio dei parametri dell'array quando veniva utilizzata la libreria sql2o.

Per maggiori informazioni e per scaricare i nuovi driver, le note di rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC.

13 luglio 2023

Data di pubblicazione: 19/09/2023

Athena annuncia la seguenti funzionalità e miglioramenti.

- EXPLAIN ANALYZE: è stato aggiunto all'output di `EXPLAIN ANALYZE` il supporto per la coda, l'analisi, la pianificazione e il tempo di esecuzione.
- EXPLAIN: l'output `EXPLAIN` ora mostra le statistiche quando la query contiene aggregazioni.
- Parquet Hive SerDe: aggiunta la `parquet.ignore.statistics` proprietà per consentire l'ignoranza delle statistiche di elaborazione durante la lettura dei dati di Parquet. Per informazioni, consulta Come ignorare le statistiche Parquet.

Per ulteriori informazioni su `EXPLAIN` e `EXPLAIN ANALYZE`, consulta Utilizzo di EXPLAIN e EXPLAIN ANALYZE in Athena. Per ulteriori informazioni su Parquet Hive SerDe, vedere. Parquet SerDe

3 luglio 2023

Data di pubblicazione: 25/07/2023

A partire dal 3 luglio 2023, Athena ha iniziato a oscurare le stringhe di query dai log. CloudTrail La stringa di query ora ha un valore di `***OMITTED***`. Questa modifica è stata apportata per impedire la divulgazione involontaria di nomi di tabelle o valori di filtro che potrebbero includere informazioni riservate. Se in precedenza facevi affidamento sui CloudTrail log per accedere a stringhe di query complete, ti consigliamo di utilizzare l'`Athena::GetQueryExecution`API e di trasmettere il valore di dal log. `responseElements.queryExecutionId` CloudTrail Per ulteriori informazioni, consulta l'GetQueryExecutionazione nell'Amazon Athena API Reference.

30 giugno 2023

Data di pubblicazione: 30/06/2023

L'editor di query Athena ora supporta suggerimenti di codice typeahead per un'esperienza di creazione di query più rapida. Ora puoi scrivere query SQL con maggiore precisione ed efficienza utilizzando le seguenti funzionalità:

- Durante la digitazione, vengono visualizzati suggerimenti in tempo reale per parole chiave, variabili locali, frammenti ed elementi del catalogo.
- Quando digiti il nome di un database o di tabella seguito da un punto, l'editor visualizza in maniera pratica un elenco di tabelle o colonne tra cui scegliere.
- Quando passi il mouse su un suggerimento di snippet, una sinossi mostra una breve panoramica della sintassi e dell'utilizzo dello snippet.
- Per migliorare la leggibilità del codice, anche le parole chiave e le relative regole di evidenziazione sono state aggiornate per allinearle alla sintassi più recente di Trino e Hive.

Questa caratteristica viene attivata per impostazione predefinita. Puoi abilitare o disabilitare la funzionalità nelle impostazioni delle preferenze dell'editor di codice.

Per provare i suggerimenti del codice typeahead nell'editor di query Athena, visita la console Athena all'indirizzo https://console.aws.amazon.com/athena/.

29 giugno 2023

Data di pubblicazione: 29/06/2023

- Athena annuncia il driver ODBC versione 2.0.1.0. Per ulteriori informazioni, consulta le note di rilascio di 2.0.1.0. Per scaricare il nuovo driver ODBC v2, consulta Come scaricare il driver ODBC 2.x. Per informazioni sulla connessione, consulta Amazon Athena ODBC 2.x.

- Athena e le relative funzionalità ora sono disponibili nella regione del Medio Oriente (EAU). Per un elenco completo dei servizi Servizi AWS disponibili in ciascuna regione Regione AWS, consulta AWS Servizi per regione.

28 giugno 2023

Data di pubblicazione: 28/06/2023

Ora puoi usare Amazon Athena per eseguire query su oggetti ripristinati dalle classi di archiviazione Amazon S3, S3 Glacier Flexible Retrieval (precedentemente Glacier) e S3 Glacier Deep Archive. Questa funzionalità viene configurata in base alla tabella. La funzionalità è supportata solo per le tabelle Apache Hive nella versione 3 del motore Athena.

Per ulteriori informazioni, consulta Esecuzione di query su oggetti Amazon S3 Glacier ripristinati.

12 giugno 2023

Data di pubblicazione: 12/06/2023

Athena annuncia le correzioni e i miglioramenti seguenti.

- Timestamp di Parquet Reader: è stato aggiunto il supporto per la lettura dei timestamp in formato come `bigint` (millis) per Parquet Reader. Questo aggiornamento fornisce lo stesso supporto nella versione 2 del motore Athena.
- EXPLAIN ANALYZE: è stato aggiunto il tempo fisico di lettura dell'input alle statistiche e all'output delle query di `EXPLAIN ANALYZE`. Per informazioni su `EXPLAIN ANALYZE`, consulta Utilizzo di EXPLAIN e EXPLAIN ANALYZE in Athena.
- INSERT: sono state migliorate le prestazioni delle query su tabelle scritte con`INSERT`. Per informazioni su `INSERT`, consulta INSERT INTO.
- Tabelle Delta Lake: è stato corretto un problema con `DROP TABLE` relativo alle tabelle Delta Lake che impediva la loro eliminazione completa se soggette a modifiche simultanee.

8 giugno 2023

Data di pubblicazione: 08/06/2023

Amazon Athena per Apache Spark annuncia le nuove funzionalità seguenti.

- Supporto per librerie e configurazioni Java personalizzate: ora puoi utilizzare i tuoi pacchetti Java e una configurazione personalizzata per le tue sessioni di Apache Spark in Athena. Usa le proprietà Spark per specificare `.jar` file, pacchetti o altre configurazioni personalizzate con la console Athena, o AWS CLI l'API Athena. Per ulteriori informazioni, consulta Aggiungere file JAR e configurazione Spark personalizzata.
- Supporto per le tabelle Apache Hudi, Apache Iceberg e Delta Lake: Athena per Spark ora supporta i formati di tabelle di archiviazione data lake open source di Apache Iceberg, Apache Hudi e Linux Foundation Delta Lake. Per ulteriori informazioni, consulta Utilizzo di formati di tabella non Hive in Amazon Athena per Apache Spark e i singoli argomenti sull'utilizzo delle tabelle Apache Iceberg, Apache Hudi e Linux Foundation Delta Lake in Athena per Spark.
- Supporto per la crittografia per Apache Spark: in Athena per Spark, ora puoi abilitare la crittografia sui dati in transito tra i nodi Spark e sui dati a riposo locali archiviati su disco da Spark. Per abilitare la crittografia Spark, puoi utilizzare la console Athena, o AWS CLI l'API Athena. Per ulteriori informazioni, consulta Abilitazione della crittografia Apache Spark.

Per ulteriori informazioni su Amazon Athena per Apache Spark, consulta Utilizzo di Apache Spark in Amazon Athena.

2 giugno 2023

Data di pubblicazione: 02/06/2023

Ora puoi eliminare le prenotazioni di capacità in Athena e utilizzare i AWS CloudFormation modelli per specificare le prenotazioni di capacità Athena.

- Elimina le prenotazioni della capacità: ora puoi eliminare le prenotazioni della capacità annullate in Athena. Una prenotazione deve essere annullata prima di poter essere cancellata. L'eliminazione di una prenotazione della capacità rimuove immediatamente la prenotazione dal tuo account. La prenotazione eliminata non può più essere referenziata, nemmeno tramite il relativo ARN. Per eliminare una prenotazione, puoi utilizzare la console Athena o l'API Athena. Per ulteriori informazioni, consulta Eliminazione di una prenotazione della capacità la Amazon Athena User Guide e DeleteCapacityReservationl'Amazon Athena API Reference.
- Usa AWS CloudFormation modelli per le prenotazioni di capacità: ora puoi utilizzare i AWS CloudFormation modelli per specificare le prenotazioni di capacità di Athena utilizzando la `AWS::Athena::CapacityReservation` risorsa. Per ulteriori informazioni, vedere AWS: :Athena:: CapacityReservation nella Guida per l'AWS CloudFormation utente.

Per ulteriori informazioni sull'utilizzo delle prenotazioni della capacità per fornire capacità in Athena, consulta Gestione della capacità di elaborazione delle query.

25 maggio 2023

Data di pubblicazione: 25/05/2023

Athena ha rilasciato aggiornamenti dei connettori di origine dati che migliorano le prestazioni delle query federate. Le nuove ottimizzazioni di push-down e il filtraggio dinamico consentono di eseguire più operazioni nel database di origine anziché in Athena. Queste ottimizzazioni riducono il runtime di esecuzione delle query e la quantità di dati scansionati. Questi miglioramenti richiedono la versione 3 del motore Athena.

I seguenti connettori sono stati aggiornati:

- Archiviazione Azure Data Lake
- Azure Synapse
- Cloudera Hive
- Cloudera Impala
- Db2
- DynamoDB
- Google BigQuery
- Hortonworks
- MySQL
- Oracle
- PostgreSQL
- Redshift
- SAP HANA
- Snowflake
- SQL Server
- Teradata

Per informazioni sull'aggiornamento dei connettori di origine dati, consulta Aggiornamento di un connettore origine dati.

18 maggio 2023

Data di pubblicazione: 18/05/2023

Ora puoi utilizzarlo AWS PrivateLink per le connessioni in entrata IPv6 ad Amazon Athena.

Amazon Athena ha ampliato il supporto per le connessioni in entrata tramite endpoint del protocollo Internet versione 6 (IPv6) per includere AWS PrivateLink. A partire da oggi, puoi connetterti ad Athena in modo sicuro e privato utilizzando AWS PrivateLink il tuo Amazon Virtual Private Cloud (Amazon VPC), oltre agli endpoint IPv6 pubblici precedentemente disponibili.

La rapida crescita di Internet sta esaurendo la disponibilità di indirizzi IPv4 (Internet Protocol version 4). IPv6 aumenta di diverse volte il numero di indirizzi disponibili in modo da non dover più gestire spazi di indirizzi sovrapposti nei VPC. Con questa versione, puoi combinare i vantaggi degli indirizzi IPv6 con i vantaggi in termini di sicurezza e prestazioni di AWS PrivateLink.

Per connetterti a livello di codice a un AWS servizio, puoi utilizzare l'SDK o per specificare un endpoint. AWS CLIAWS Per ulteriori informazioni sugli endpoint di servizio e sugli endpoint del servizio Athena, consulta Endpoint di servizio AWS e gli endpoint e le quote di Amazon Athena nella Riferimenti generali di Amazon Web Services.

15 maggio 2023

Data di pubblicazione: 15/05/2023

Athena annuncia il rilascio dei connettori Apache Spark DataSource V2 (DSV2) per DynamoDB, Logs, Metrics e CMDB. CloudWatch CloudWatch AWS Utilizza i nuovi connettori DSV2 per eseguire query su queste origini dati tramite Spark. I connettori DSV2 utilizzano gli stessi parametri dei connettori federati Athena corrispondenti. I connettori DSV2 funzionano direttamente sugli worker di Spark e non richiedono l'implementazione di una funzione Lambda per utilizzarli.

Per ulteriori informazioni, consulta Connettori origine dati di Athena per Apache Spark.

10 maggio 2023

Data di pubblicazione: 10/05/2023

Rilasciato il driver ODBC 1.1.20 per Athena.

Funzionalità e miglioramenti:

- Supporto della sovrascrittura degli endpoint di Lake Formation.

- Il plugin di autenticazione ADFS ha un nuovo parametro per l'impostazione del valore Relying Party (`LoginToRP`).
- AWS aggiornamenti delle librerie.

Correzioni di bug

- Errore di deallocazione delle dichiarazioni preparate quando non è riuscito l'invio del metodo `SQLPrepare()`.
- Errore nell'associazione dei parametri della dichiarazione preparata nel corso della conversione di un tipo C in un tipo SQL.
- Errore di restituzione dei dati quando le query `EXPLAIN` e `EXPLAIN ANALYZE` hanno utilizzato `SQLPrepare()` e `SQLExecute()`.

Per maggiori informazioni e per scaricare i nuovi driver, le note di rilascio e la documentazione, consulta Connessione ad Amazon Athena con ODBC.

8 maggio 2023

Data di pubblicazione: 08/05/2023

Athena annuncia le correzioni e i miglioramenti seguenti.

- Integrazione Hudi aggiornata: Athena ha aggiornato la sua integrazione con Apache Hudi. Ora puoi utilizzare Athena per eseguire query sulle tabelle Hudi 0.12.2 e ora è supportato l'elenco dei metadati Hudi per le tabelle Hudi. Per informazioni, consulta Utilizzo di Athena per eseguire query sui set di dati Apache Hudi e Elenco dei metadati Hudi.
- Correzione conversione del timestamp: è stata corretta la gestione delle conversioni di timestamp in un tipo di dati con precisione inferiore. In precedenza, la versione 3 del motore Athena arrotondava erroneamente il valore al tipo di destinazione anziché troncarlo durante la conversione.

  Gli esempi seguenti illustrano una gestione errata prima della sua correzione.

  Esempio 1: conversione da un timestamp in microsecondi a millisecondi

  Dati campione

```
A, 2020-06-10 15:55:23.383
```

```
B, 2020-06-10 15:55:23.382
C, 2020-06-10 15:55:23.383345
D, 2020-06-10 15:55:23.383945
E, 2020-06-10 15:55:23.383345734
F, 2020-06-10 15:55:23.383945278
```

La seguente query tenta di recuperare i timestamp che corrispondono a un valore specifico.

```
SELECT *
FROM table
WHERE timestamps.col = timestamp'2020-06-10 15:55:23.383'
```

La query ha restituito i seguenti risultati.

```
A, 2020-06-10 15:55:23.383
C, 2020-06-10 15:55:23.383
E, 2020-06-10 15:55:23.383
```

Prima della correzione, Athena non includeva i valori `2020-06-10 15:55:23.383945` o `2020-06-10 15:55:23.383945278` perché erano arrotondati a `2020-06-10 15:55:23.384`.

Esempio 2: conversione da un timestamp a una data

La seguente query ha restituito un risultato errato.

```
SELECT date(timestamp '2020-12-31 23:59:59.999')
```

Risultato

`2021-01-01`

Prima della correzione, Athena ha arrotondato il valore per eccesso, spostando quindi in giorno in avanti. Adesso questi valori vengono troncati anziché essere arrotondati per eccesso.

28 aprile 2023

Data di pubblicazione: 28/04/2023

Ora puoi utilizzare le prenotazioni della capacità su Amazon Athena per eseguire query SQL su capacità di calcolo completamente gestita.

La capacità fornita offre funzionalità di gestione dei carichi di lavoro che aiutano a stabilire le priorità, controllare e dimensionare i carichi di lavoro interattivi più importanti. Puoi aggiungere capacità in qualsiasi momento per aumentare il numero di query eseguite contemporaneamente, controllare quali carichi di lavoro utilizzano la capacità e condividere la capacità tra i carichi di lavoro.

Per ulteriori informazioni, consulta Gestione della capacità di elaborazione delle query. Per informazioni sui prezzi, consulta la pagina dei Prezzi di Amazon Athena.

17 aprile 2023

Data di pubblicazione: 17/04/2023

Athena rilascia la versione 2.0.36 del driver JDBC. Il driver include nuove funzionalità e ha risolto un problema.

Nuove funzionalità

- Ora puoi utilizzare con l'autenticazione AD FS identificatori di relying party personalizzabili.
- Ora puoi aggiungere il nome dell'applicazione che utilizza il connettore alla stringa dell'agente utente.

Problemi risolti

- È stato corretto un errore che si verificava quando `getSchema()` veniva utilizzato per recuperare uno schema inesistente.

Per maggiori informazioni e per scaricare i nuovi driver, le note di rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC.

14 aprile 2023

Data di pubblicazione: 20/06/2023

Athena annuncia le correzioni e i miglioramenti seguenti.

- Quando si esegue la conversione di una stringa in timestamp, è necessario uno spazio tra il giorno e l'ora o il fuso orario. Per ulteriori informazioni, consulta Spazio richiesto tra i valori di data e ora per la conversione da una stringa a un timestamp.

- È stata rimossa una modifica che causava interruzioni nella modalità di gestione della precisione del timestamp. Per mantenere la coerenza tra la versione 2 del motore Athena e la versione 3 del motore Athena, ora la precisione del timestamp è predefinita in millisecondi anziché in microsecondi.
- Athena ora impone in modo coerente l'accesso al bucket di output delle query quando esegue le query. Assicurati che tutti i principali IAM che eseguono l'StartQueryExecutionazione abbiano l'GetBucketLocationautorizzazione S3: sul bucket di output della query.

4 aprile 2023

Data di pubblicazione: 04/04/2023

Ora puoi utilizzare Amazon Athena per creare e interrogare le visualizzazioni su origini dati federati. Utilizza una singola visualizzazione federata per eseguire query su più tabelle o sottoinsiemi di dati esterni. Ciò semplifica il linguaggio SQL richiesto e offre la flessibilità necessaria per nascondere le origini di dati agli utenti finali che devono utilizzare SQL per eseguire query sui dati.

Per ulteriori informazioni, consulta Utilizzo delle visualizzazioni e Esecuzione di query federate.

30 marzo 2023

Data di pubblicazione: 30/03/2023

Amazon Athena annuncia la disponibilità di Amazon Athena per Apache Spark in ulteriori Regioni AWS.

Questa versione amplia la disponibilità di Amazon Athena per Apache Spark per includere Asia Pacifico (Mumbai), Asia Pacifico (Singapore), Asia Pacifico (Sydney) ed Europa (Francoforte).

Per ulteriori informazioni su Amazon Athena per Apache Spark, consulta Utilizzo di Apache Spark in Amazon Athena.

28 marzo 2023

Data di pubblicazione: 28/03/2023

Athena annuncia le correzioni e i miglioramenti seguenti.

- Nelle risposte alle azioni API di Athena `GetQueryExecution` e `BatchGetQueryExecution`, il nuovo campo `subStatementType` mostra il tipo di query eseguita (ad esempio `SELECT`, `INSERT`, `UNLOAD`, `CREATE_TABLE` o`CREATE_TABLE_AS_SELECT`).
- È stato corretto un bug a causa del quale i file manifesto non venivano crittografati correttamente per le operazioni di scrittura di Apache Hive.
- La versione 3 del motore Athena ora gestisce i valori `NaN` e `Infinity` correttamente nella funzione `approx_percentile`. La funzione `approx_percentile` restituisce il percentile approssimativo per un set di dati alla percentuale specificata.

  La versione 2 del motore Athena considera `NaN` erroneamente come un valore maggiore di `Infinity`. La versione 3 del motore Athena ora gestisce `NaN` e `Infinity` in conformità al trattamento di questi valori in altre funzioni analitiche e statistiche. I punti seguenti descrivono in modo più dettagliato il nuovo comportamento.

  - Se `NaN` è presente nel set di dati, Athena restituisce `NaN`.
  - Se `NaN` non è presente, ma `Infinity` è presente, Athena considera `Infinity` come un numero molto grande.
  - Se sono presenti più valori `Infinity`, Athena li considera come uno stesso numero molto grande. Se necessario, Athena emette `Infinity`.
  - Se un singolo set di dati contiene sia `Infinity` sia `-Double.MAX_VALUE`, e un risultato percentile è `-Double.MAX_VALUE`, Athena restituisce `-Infinity`.
  - Se un singolo set di dati contiene sia `Infinity` sia `Double.MAX_VALUE`, e un risultato percentile è `Double.MAX_VALUE`, Athena restituisce `Infinity`.
  - Per escludere `Infinity` e `NaN` da un calcolo, utilizza la funzione `is_finite()`, come nell'esempio seguente.

    ```
    approx_percentile(x, 0.5) FILTER (WHERE is_finite(x))
    ```

27 marzo 2023

Data di pubblicazione: 27/03/2023

Ora puoi specificare un livello minimo di crittografia per i gruppi di lavoro Athena SQL in Amazon Athena. Questa funzionalità assicura che i risultati di tutte le query nel gruppo di lavoro Athena SQL siano crittografati al livello di crittografia specificato o superiore. Puoi scegliere tra diversi livelli

di crittografia per proteggere i dati. Per configurare il livello minimo di crittografia desiderato, puoi utilizzare la console Athena AWS CLI, l'API o l'SDK.

La funzionalità di crittografia minima non è disponibile per i gruppi di lavoro abilitati su Apache Spark. Per ulteriori informazioni, consulta Configurazione della crittografia minima per un gruppo di lavoro.

17 marzo 2023

Data di pubblicazione: 17/03/2023

Athena annuncia le correzioni e i miglioramenti seguenti.

- È stato risolto un problema con il connettore Amazon Athena DynamoDB che causava il fallimento delle query con il messaggio di errore che KeyConditionExpressions doveva contenere solo una condizione per chiave.

  Questo problema si verifica perché la versione 3 del motore Athena è in grado di eseguire il pushdown di più tipi di predicati rispetto alla versione 2 del motore Athena. Nella versione 3 del motore Athena, clausole come `some_column LIKE 'someprefix%` vengono inviate come predicati di filtro che applicano un limite inferiore e superiore a una determinata colonna. La versione 2 del motore Athena non eseguiva il pushdown di questi predicati. Nella versione 3 del motore Athena, quando `some_column` è una colonna chiave di ordinamento, il motore invia il predicato di filtro al connettore DynamoDB. Viene eseguito un ulteriore pushdown del predicato del filtro al servizio DynamoDB. Poiché DynamoDB non supporta più di una condizione di filtro su una chiave di ordinamento, DynamoDB restituisce un errore.

  Per correggere questo problema, aggiorna il connettore DynamoDB di Amazon Athena alla versione 2023.11.1. Per istruzioni sull'aggiornamento del connettore, consulta Aggiornamento di un connettore origine dati.

8 marzo 2023

Data di pubblicazione: 08/03/2023

Athena annuncia le correzioni e i miglioramenti seguenti.

- È stato risolto un problema relativo alle query federate che comportava l'invio in microsecondi anziché in millisecondi di valori del predicato del timestamp.

15 febbraio 2023

Data di pubblicazione: 15/02/2023

Athena annuncia le correzioni e i miglioramenti seguenti.

- Ora puoi utilizzare la crittografia lato client per crittografare i dati nelle operazioni di scrittura di Amazon S3 per Iceberg.
- È stato risolto un problema che riguardava la crittografia lato server nelle operazioni di scrittura in Amazon S3 per Iceberg.

31 gennaio 2023

Data di pubblicazione: 31/01/2023

Ora è possibile utilizzare Amazon Athena per interrogare i dati in Google Cloud Storage. Come Amazon S3, Google Cloud Storage è un servizio gestito che archivia i dati in bucket. Utilizza il connettore Athena per Google Cloud Storage per eseguire query federate interattive sui dati esterni.

Per ulteriori informazioni, consulta Connettore Google Cloud Storage per Amazon Athena.

20 gennaio 2023

Data di pubblicazione: 20/01/2023

Ora puoi vedere la documentazione estesa per il supporto alla compressione Athena. Sono stati aggiunti singoli argomenti per Compressione delle tabelle Hive, Compressione delle tabelle Iceberg e Livelli di compressione ZSTD.

Per ulteriori informazioni, consulta Supporto della compressione in Athena.

3 gennaio 2023

Data di pubblicazione: 03/01/2023

Athena annuncia i seguenti aggiornamenti:

- Comandi aggiuntivi per i metastore Hive: è possibile utilizzare Athena per connettersi al proprio metastore Apache Hive autogestito come catalogo di metadati ed eseguire query sui dati archiviati in Amazon S3. Con questa versione, è possibile utilizzare `CREATE TABLE AS` (CTAS) e 12

comandi Data Definition Language (DDL) aggiuntivi per interagire con i metastore Apache Hive.`INSERT INTO` Puoi gestire gli schemi metastore Hive direttamente da Athena utilizzando questo set esteso di funzionalità SQL.

Per ulteriori informazioni, consulta Utilizzo di un connettore dati Athena per il metastore Hive esterno.

- Driver JDBC versione 2.0.35: Athena rilascia la versione 2.0.35 del driver JDBC. Il driver JDBC 2.0.35 contiene i seguenti aggiornamenti:
  - Il driver ora utilizza le seguenti librerie per il parser Jackson JSON.
    - jackson-annotations 2.14.0 (in precedenza 2.13.2)
    - jackson-core 2.14.0 (in precedenza 2.13.2)
    - jackson-databind 2.14.0 (in precedenza 2.13.2.2)
  - Il supporto per la versione 4.1 di JDBC è stato interrotto.

  Per maggiori informazioni e per scaricare il nuovo driver, le note sul rilascio e la documentazione, consulta la pagina Connessione ad Amazon Athena con JDBC.

Note di rilascio di Athena per il 2022

14 dicembre 2022

Data di pubblicazione: 14/12/2022

Ora puoi utilizzare il connettore Amazon Athena per Kafka per eseguire query SQL sui dati in streaming. Ad esempio, puoi eseguire query analitiche sui dati di streaming in tempo reale in Streaming gestito da Amazon per Apache Kafka (Amazon MSK) e unirli ai dati storici nel tuo data lake in Amazon S3.

Il connettore Amazon Athena per Kafka supporta le query su più motori di streaming. Puoi utilizzare Athena per eseguire query SQL su cluster con provisioning e serverless di Amazon MSK, su implementazioni Kafka autogestite e sullo streaming di dati in Confluent Cloud.

Per ulteriori informazioni, consulta Connettore MSK di Amazon Athena.

2 dicembre 2022

Data di pubblicazione: 02/12/2022

Athena rilascia la versione 2.0.34 del driver JDBC. Il driver JDBC 2.0.34 include le seguenti nuove funzionalità e problemi risolti:

- Supporto per il riutilizzo dei risultati delle query: ora puoi riutilizzare i risultati delle query eseguite in precedenza fino a un limite di tempo specificato invece di fare in modo che Athena ricalcoli i risultati ogni volta che viene eseguita la query. Per ulteriori informazioni, consulta la Guida all'installazione e alla configurazione, disponibile nella pagina dei download di JDBC, e la pagina Riutilizzo dei risultati delle query.
- InstanceMetadata Supporto Ec2: il driver JDBC ora supporta il metodo di InstanceMetadata autenticazione Ec2 utilizzando i profili di istanza IAM.
- Correzione delle eccezioni basate sui caratteri: è stata risolta un'eccezione che si verificava con le query contenenti determinati caratteri linguistici.
- Correzione della vulnerabilità: è stata corretta una vulnerabilità relativa alle dipendenze incluse nel pacchetto con il connettore. AWS

Per maggiori informazioni e per scaricare i nuovi driver, le note di rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC.

30 novembre 2022

Data di pubblicazione: 30/11/2022

Ora puoi creare ed eseguire in modo interattivo applicazioni Apache Spark e notebook compatibili con Jupyter su Athena. Esegui l'analisi dei dati su Athena utilizzando Spark senza dover pianificare, configurare o gestire le risorse. Invia il codice Spark per l'elaborazione e ricevi direttamente i risultati. Utilizza l'esperienza semplificata dei notebook nella console Amazon Athena per sviluppare applicazioni Apache Spark utilizzando Python o API dei notebook Athena.

Apache Spark su Amazon Athena è serverless e offre il dimensionamento automatico e on demand per l'elaborazione istantanea, in modo da far fronte ai cambiamenti dei volumi di dati e dei requisiti di elaborazione.

Per ulteriori informazioni, consulta Utilizzo di Apache Spark in Amazon Athena.

18 novembre 2022

Data di pubblicazione: 18/11/2022

Ora puoi utilizzare il connettore Amazon Athena per IBM Db2 per eseguire query su Db2 da Athena. Ad esempio, puoi eseguire query analitiche su un data warehouse in Db2 e su un data lake in Amazon S3.

Il connettore Amazon Athena Db2 presenta diverse opzioni di configurazione attraverso le variabili di ambiente Lambda. Per informazioni sulle opzioni di configurazione, i parametri, le stringhe di connessione, l'implementazione e le limitazioni, consulta la pagina Connettore IBM Db2 di Amazon Athena.

17 novembre 2022

Data di pubblicazione: 17/11/2022

Il supporto di Apache Iceberg nella versione 3 del motore Athena offre ora le seguenti funzionalità avanzate di transazione ACID:

- Supporto ORC e Avro: crea tabelle Iceberg utilizzando i formati di file basati su righe e colonne Apache Avro e Apache ORC. Il supporto per questi formati si aggiunge al supporto esistente per Parquet.
- MERGE INTO: utilizza il comando `MERGE INTO` per unire i dati su larga scala in modo efficiente. `MERGE INTO` combina le operazioni `INSERT`, `UPDATE` e `DELETE` in un'unica transazione. Ciò riduce il sovraccarico di elaborazione nella pipeline di dati e richiede meno SQL per la scrittura. Per ulteriori informazioni, consulta Aggiornamento dati di tabelle Iceberg e MERGE INTO.
- Supporto CTAS e VIEW: utilizza le istruzioni `CREATE TABLE AS SELECT` (CTAS) e `CREATE VIEW` con le tabelle Iceberg. Per ulteriori informazioni, consulta CREATE TABLE AS e CREATE VIEW.
- Supporto VACUUM: puoi utilizzare l'istruzione `VACUUM` per ottimizzare il tuo data lake eliminando snapshot e dati che non sono più necessari. Puoi utilizzare questa funzionalità per migliorare le prestazioni di lettura e soddisfare i requisiti normativi come il GDPR. Per ulteriori informazioni, consulta Ottimizzazione delle tabelle Iceberg e VACUUM.

Queste nuove funzionalità richiedono la versione 3 del motore Athena e sono disponibili in tutte le Regioni in cui è supportato Athena. Puoi utilizzarli con la console, i driver o l'API Athena.

Per ulteriori informazioni sull'utilizzo di Iceberg in Athena, consulta la pagina Utilizzo di tabelle Apache Iceberg.

14 novembre 2022

Data di pubblicazione: 14/11/2022

Amazon Athena ora supporta gli endpoint IPv6 per le connessioni in entrata che puoi utilizzare per richiamare le funzioni Athena su IPv6. Puoi utilizzare questa funzionalità per soddisfare i requisiti di conformità IPv6. Inoltre, elimina la necessità di apparecchiature di rete aggiuntive per gestire la traduzione degli indirizzi tra IPv4 e IPv6.

Per utilizzare questa funzionalità, configura le tue applicazioni di modo che utilizzino i nuovi endpoint dual-stack di Athena, che supportano sia IPv4 sia IPv6. Gli endpoint dual-stack utilizzano il formato `athena.region.api.aws`. Ad esempio, l'endpoint dual-stack nella Regione Stati Uniti orientali (Virginia settentrionale) è `athena.us-east-1.api.aws`.

Quando effettui una richiesta a un endpoint dual-Stack di Athena, l'endpoint risolve a un indirizzo IPv6 o IPv4 a seconda del protocollo utilizzato dalla rete e dal client. Per connetterti a livello di codice a un AWS servizio, puoi utilizzare o l'SDK per specificare un endpoint. AWS CLIAWS

Per ulteriori informazioni sugli endpoint di servizio, consulta Endpoint di servizio AWS. Per ulteriori informazioni sugli endpoint di servizio di Athena, consulta la pagina Amazon Athena endpoints and quotas (Endpoint e quote di Amazon Athena) nella documentazione di AWS .

Puoi utilizzare i nuovi endpoint dual-stack di Athena per le connessioni in entrata senza costi aggiuntivi. Gli endpoint dual-stack sono disponibili al pubblico in tutte le Regioni AWS.

11 novembre 2022

Data di pubblicazione: 11/11/2022

Athena annuncia le correzioni e i miglioramenti seguenti.

- Controllo granulare degli accessi esteso a Lake Formation: ora è possibile utilizzare le policy di controllo granulare degli accessi di AWS Lake Formation nelle query Athena per i dati archiviati in qualsiasi formato di file o tabella supportato. Puoi utilizzare il controllo granulare degli accessi in Lake Formation per limitare l'accesso ai dati nei risultati delle query usando i filtri di dati per ottenere una sicurezza a livello di colonna, riga e cella. I formati di tabella supportati in Athena includono Apache Iceberg, Apache Hudi e Apache Hive. Il controllo granulare degli accessi esteso è disponibile in tutte le Regioni supportate da Athena. Il supporto esteso per i formati di tabelle e file richiede Versione 3 del motore Athena, che offre nuove funzionalità e prestazioni di query

migliorate, ma non modifica il modo in cui si impostano le policy di controllo granulare degli accessi in Lake Formation.

L'utilizzo di questo controllo granulare degli accessi esteso in Athena comporta le seguenti considerazioni:

- EXPLAIN: le informazioni sul filtraggio di righe o celle definite in Lake Formation e le informazioni sulle statistiche delle query non vengono visualizzate nell'output di `EXPLAIN` e`EXPLAIN ANALYZE` . Per ulteriori informazioni su `EXPLAIN` in Athena, consulta la pagina Utilizzo di EXPLAIN e EXPLAIN ANALYZE in Athena.
- Metastore Hive esterni: le colonne nascoste di Apache Hive non possono essere utilizzate per filtrare in modo dettagliato il controllo degli accessi e le tabelle di sistema Apache Hive nascoste non sono supportate dal controllo granulare degli accessi. Per ulteriori informazioni, consulta Considerazioni e limitazioni nell'argomento Utilizzo di un connettore dati Athena per il metastore Hive esterno.
- Statistiche delle query: il conteggio delle righe di input e output a livello di fase e le informazioni sulla dimensione dei dati non vengono visualizzati nelle statistiche delle query di Athena quando una query ha filtri a livello di riga definiti in Lake Formation. Per informazioni sulla visualizzazione delle statistiche per le query Athena, vedere e. Visualizzazione di statistiche e dettagli di esecuzione per le query completate GetQueryRuntimeStatistics
- Gruppi di lavoro: gli utenti dello stesso gruppo di lavoro Athena possono visualizzare i dati che il controllo granulare degli accessi di Lake Formation ha configurato come accessibili al gruppo di lavoro. Per informazioni sull'utilizzo di Athena per eseguire query sui dati registrati con Lake Formation, consulta la pagina Utilizzo di Athena per eseguire query sui dati registrati con AWS Lake Formation.

Per informazioni sull'utilizzo del controllo granulare degli accessi in Lake Formation, consulta Gestone del controllo granulare degli accessi tramite AWS Lake Formation nel Blog sui Big Data di AWS .

- Athena Federated Query: Athena Federated Query ora conserva la successione di maiuscole e minuscole originale dei nomi dei campo negli oggetti `struct`. In precedenza, i nomi dei campi `struct` venivano automaticamente convertiti in lettere minuscole.

8 novembre 2022

Data di pubblicazione: 08/11/2022

Ora puoi utilizzare la funzione di memorizzazione nella cache per il riutilizzo dei risultati delle query per accelerare le query ripetute in Athena. Una query ripetuta è una query SQL identica a quella inviata di recente che produce gli stessi risultati. Quando è necessario eseguire più query identiche, il riutilizzo dei risultati nella cache può ridurre il tempo necessario per produrre risultati. Il riutilizzo dei risultati nella cache riduce anche i costi riducendo il numero di byte scansionati.

Per ulteriori informazioni, consulta Riutilizzo dei risultati delle query.

13 ottobre 2022

Data di pubblicazione: 13/10/2022

Athena annuncia la versione 3 del motore Athena.

Athena ha aggiornato il proprio motore di query SQL in modo da includere le ultime funzionalità del progetto open source Trino. Oltre a supportare tutte le funzionalità della versione 2 del motore Athena, la versione 3 include oltre 50 nuove funzioni SQL, 30 nuove funzionalità e più di 90 miglioramenti in termini di prestazioni delle query. Con il lancio odierno, Athena introduce inoltre un approccio di integrazione continua per la gestione del software open source volto a migliorare l'allineamento ai progetti Trino e Presto e ottenere così un accesso più rapido ai miglioramenti della community, integrati e ottimizzati all'interno del motore Athena.

Per ulteriori informazioni, consulta Versione 3 del motore Athena.

10 ottobre 2022

Data di pubblicazione: 10/10/2022

Athena rilascia la versione 2.0.33 del driver JDBC. Il driver JDBC 2.0.33 include le seguenti modifiche:

- La nuova versione del driver, la versione di JDBC e le proprietà del nome del plug-in sono state aggiunte alla stringa user-agent nella classe del provider di credenziali.
- I messaggi di errore sono stati corretti e sono state aggiunte le informazioni necessarie.
- Le istruzioni preparate vengono ora deallocate se la connessione viene chiusa o se l'esecuzione dell'istruzione preparata da Athena ha esito negativo.

Per maggiori informazioni e per scaricare i nuovi driver, le note di rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC.

23 settembre 2022

Data di pubblicazione: 26/09/2022

Il connettore Amazon Athena Neptune ora supporta la corrispondenza senza distinzione tra maiuscole e minuscole per i nomi di colonne e tabelle.

- Il connettore dell'origine dei dati Neptune può risolvere i nomi delle colonne nelle tabelle Neptune che utilizzano una combinazione di maiuscole e minuscole, anche se i nomi delle colonne sono indicati in minuscolo nella tabella in AWS Glue. Per abilitare questo comportamento, imposta la variabile di ambiente `enable_caseinsensitivematch` su `true` nella funzione Lambda del connettore Neptune.
- Poiché AWS Glue supporta solo nomi di tabella minuscoli, quando create una AWS Glue tabella per Neptune, specificate AWS Glue il parametro table. `"glabel" = table_name`

Per ulteriori informazioni sul connettore Neptune, consulta Connettore Amazon Athena Neptune.

13 settembre 2022

Data di pubblicazione: 13/09/2022

Athena annuncia le correzioni e i miglioramenti seguenti.

- Metastore Hive esterno: Athena ora restituisce `NULL` invece di generare un'eccezione quando la clausola `WHERE` include una partizione che non esiste in un metastore Hive esterno (EHMS). Il nuovo comportamento corrisponde a quello di AWS Glue Data Catalog.
- Query parametrizzate: ora nelle query parametrizzate è possibile impostare i valori sul tipo di dati `DOUBLE`.
- Apache Iceberg: adesso le operazioni di scrittura nelle tabelle Iceberg hanno esito positivo quando il Blocco oggetti è abilitato in un bucket Amazon S3.

31 agosto 2022

Data di pubblicazione: 31/08/2022

Amazon Athena annuncia la disponibilità di Athena e delle sue funzioni nella Regione Asia Pacific (Giacarta).

Questa versione amplia la disponibilità di Athena per includere le Regioni Asia Pacifico (Hong Kong), Asia Pacific (Giacarta), Asia Pacifico (Mumbai), Asia Pacifico (Osaka-Locale), Asia Pacifico (Seoul), Asia Pacifico (Singapore), Asia Pacifico (Sydney) e Asia Pacifico (Tokyo). Per un elenco completo di Servizi AWS disponibili in queste e in altre Regioni, fai riferimento all'Elenco dei servizi Regione AWS ali.

23 agosto 2022

Pubblicato il 23/8/2022

Il rilascio v2022.32.1 dell'SDK Athena Query Federation include le seguenti modifiche:

- Al connettore dell'origine dei dati Oracle di Amazon Athena è stato aggiunto il supporto per le connessioni basate su SSL alle istanze di Amazon RDS. Il supporto è limitato al protocollo Transport Layer Security (TLS) e all'autenticazione del server da parte del client. Poiché l'autenticazione reciproca non è supportata in Amazon RDS, l'aggiornamento non include il supporto per l'autenticazione reciproca.

  Per ulteriori informazioni, consulta Connettore Amazon Athena per Oracle.

3 agosto 2022

Pubblicato il 3/8/2022

Athena rilascia la versione 2.0.32 del driver JDBC. Il driver JDBC 2.0.32 include le seguenti modifiche:

- La stringa `User-Agent` inviata all'SDK di Athena è stata estesa per contenere la versione del driver, la versione della specifica JDBC e il nome del plug-in di autenticazione.
- È stata risolta l'eccezione `NullPointerException` che si manifestava quando non veniva fornito alcun valore per il parametro `CheckNonProxyHost`.
- È stato risolto un problema di `login_url` analisi nel plug-in di autenticazione. BrowserSaml
- È stato risolto un problema relativo all'host proxy che si verificava quando il parametro `UseProxyforIdp` era impostato su `true`.

Per maggiori informazioni e per scaricare i nuovi driver, le note di rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC.

1 agosto 2022

Pubblicato il 01/08/2022

Athena annuncia miglioramenti all'SDK Athena Query Federation e ai connettori di origine dei dati predefiniti Athena. I miglioramenti includono:

- Analisi della struttura: risolto un problema di analisi di `GlueFieldLexer` in Athena Query Federation SDK che impediva a determinate strutture complicate di visualizzare tutti i dati. Questo problema riguardava i connettori basati sull'SDK Athena Query Federation.
- AWS Glue tabelle: è stato aggiunto un supporto aggiuntivo per i tipi di `decimal` colonna `set` e nelle AWS Glue tabelle.
- Connettore DynamoDB: aggiunta la possibilità di ignorare le maiuscole sui nomi degli attributi di DynamoDB. Per ulteriori informazioni, consulta `disable_projection_and_casing` nella sezione Parametri della pagina Connettore Amazon Athena DynamoDB.

Per ulteriori informazioni, vedere la release v2022.30.2 di Athena Query Federation su. GitHub

21 luglio 2022

Data pubblicazione: 21/07/2022

Ora puoi analizzare ed eseguire il debug delle query utilizzando parametri delle prestazioni e strumenti di analisi interattivi e visivi delle query nella console Athena. I dati sulle prestazioni delle query e i dettagli di esecuzione possono aiutare a identificare i colli di bottiglia nelle query, ispezionare gli operatori e le statistiche per ogni fase di una query, tracciare il volume di dati che scorre tra le fasi e convalidare l'impatto dei predicati. Ora è possibile:

- Accedi al piano di esecuzione distribuito e logico per la query con un solo clic.
- Esplora le operazioni in ogni fase prima che dell'esecuzione.
- Visualizza le prestazioni delle query completate con parametri per il tempo trascorso nelle fasi di creazione della coda, pianificazione ed esecuzione.
- Ottieni informazioni sul numero di righe e sulla quantità di dati di origine elaborati e restituiti dalla query.
- Visualizza i dettagli di esecuzione granulari per le query presentate nel contesto e formattate come un grafico interattivo.

- Usa dettagli di esecuzione precisi a livello di fase per comprendere il flusso di dati attraverso la query.
- Analizza i dati sulle prestazioni delle query a livello di programmazione utilizzando nuove API per ottenere statistiche di runtime delle query, rilasciato oggi.

Per scoprire come utilizzare queste funzionalità nelle tue query, guarda il video tutorial Ottimizza le query di Amazon Athena con nuovi strumenti di analisi delle query sul canale. AWS YouTube

Per la documentazione, consulta Visualizzazione dei piani di esecuzione per query SQL e Visualizzazione di statistiche e dettagli di esecuzione per le query completate.

11 luglio 2022

Pubblicato il 11/07/2022

Ora è possibile eseguire query con parametri direttamente dalla console o dall'API Athena senza preparare in anticipo le istruzioni SQL.

Quando si eseguono query nella console Athena che hanno parametri sotto forma di punti interrogativi, l'interfaccia utente richiede di immettere direttamente i valori per i parametri. Ciò elimina la necessità di modificare i valori letterali nell'editor di query ogni volta che si desidera eseguire la query.

Se si utilizza l'API avanzata per l'esecuzione di query, ora è possibile fornire i parametri di esecuzione e i relativi valori in una singola chiamata.

Per ulteriori informazioni, consulta Utilizzo di query parametrizzate in questa guida per l'utente e il post del blog sui Big Data AWS Utilizzare le query parametrizzate di Amazon Athena per fornire dati come servizio.

8 luglio 2022

Data pubblicazione: 08/07/2022

Athena annuncia le correzioni e i miglioramenti seguenti.

- È stato risolto un problema relativo alla gestione `DATE` della conversione delle colonne per gli SageMaker endpoint (UDF) che causava errori nelle query.

6 giugno 2022

Data pubblicazione: 06/06/2022

Athena rilascia la versione 2.0.31 del driver JDBC. Il driver JDBC 2.0.31 include le seguenti modifiche:

- Problema di dipendenza log4j— Risolve un messaggio di errore Impossibile trovare la classe del driver causato da una dipendenza log4j.

Per maggiori informazioni e per scaricare i nuovi driver, le note di rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC.

25 maggio 2022

Data di pubblicazione: 25/05/2022

Athena annuncia le correzioni e i miglioramenti seguenti.

- Supporto di Iceberg
  - È stato introdotto il supporto per le query tra regioni. Ora puoi interrogare le tabelle Iceberg in un formato diverso da Regione AWS quello Regione AWS che stai utilizzando. L'interrogazione tra regioni non è supportata nelle regioni cinesi.
  - È stato introdotto il supporto per la configurazione della crittografia lato server. Ora puoi utilizzare SSE-S3/SSE-KMS per crittografare i dati dalle operazioni di scrittura di Iceberg in Amazon S3.

  Per ulteriori informazioni sull'utilizzo di Apache Iceberg in Athena, consulta la sezione Utilizzo di tabelle Apache Iceberg.
- Rilasci del driver JDBC 2.0.30

  Il driver JDBC 2.0.30 per Athena presenta i seguenti miglioramenti:
  - Risolve un problema di data race che interessava le istruzioni preparate con parametri.
  - Risolve un problema di avvio dell'applicazione che si è verificato negli ambienti di compilazione Gradle.

  Per scaricare il driver JDBC 2.0.30, le note di rilascio e la documentazione, consulta la sezione Connessione ad Amazon Athena con JDBC.

6 maggio 2022

Data pubblicazione: 06/05/2022

Rilasciati i driver JDBC 2.0.29 e ODBC 1.1.17 per Athena.

Queste modifiche includono:

- È stato aggiornato il processo di avvio del browser del plug-in SAML.

Per maggiori informazioni su queste modifiche e per scaricare i nuovi driver, le note di rilascio e la documentazione, consultare Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

22 aprile 2022

Data pubblicazione: 22/04/2022

Athena annuncia le correzioni e i miglioramenti seguenti.

- È stato risolto un problema nella funzione di indici e filtri delle partizioni con la cache delle partizioni che si verificava quando erano soddisfatte le seguenti condizioni:
  - La `partition_filtering.enabled` chiave è stata impostata su `true` nelle proprietà della AWS Glue tabella.
  - La stessa tabella è stata utilizzata più volte con valori del filtro delle partizioni diversi.

21 aprile 2022

Data pubblicazione: 21/04/2022

Ora puoi usare Amazon Athena per eseguire query federate su nuove fonti di dati, tra cui Google BigQuery, Azure Synapse e Snowflake. I nuovi connettori per origini dati includono:

- Azure Data Lake Storage (ADLS) Gen2
- Azure Synapse
- Cloudera Hive
- Cloudera Impala
- Google BigQuery

- Hortonworks
- Microsoft SQL Server
- Oracle
- SAP HANA (Express Edition)
- Snowflake
- Teradata

Per un elenco completo delle origini dei dati supportate da Athena, consulta la sezione Connettori di origine dati disponibili.

Per semplificare l'esplorazione delle fonti disponibili e la connessione ai dati, ora puoi cercare, ordinare e filtrare i connettori disponibili da una schermata Origini dati (Origini dati) aggiornata nella console Athena.

Per informazioni sulle query di origini federate, consulta Utilizzo di Amazon Athena Federated Query e Esecuzione di query federate.

13 aprile 2022

Data pubblicazione: 13/04/2022

Athena rilascia la versione 2.0.28 del driver JDBC. Il driver JDBC 2.0.28 include le seguenti modifiche:

- Supporto JWT - Ora il driver supporta i token Web JSON (JWT) per l'autenticazione. Per ulteriori informazioni sull'utilizzo di JWT con il driver JDBC, consulta la guida all'installazione e alla configurazione, scaricabile dalla pagina del driver JDBC.
- Librerie Log4j aggiornate - Il driver JDBC ora utilizza le seguenti librerie Log4j:
  - Log4j-API 2.17.1 (in precedenza 2.17.0)
  - Log4j-API 2.17.1 (in precedenza 2.17.0)
  - Log4j-JCL 2.17.2
- Altri miglioramenti - Il nuovo driver include anche i seguenti miglioramenti e correzioni di bug:
  - La funzione relativa alle istruzioni preparate di Athena è ora disponibile tramite JDBC. Per ulteriori informazioni sulle istruzioni preparate, consulta Utilizzo di query parametrizzate.
  - La federazione SAML Athena JDBC è ora funzionale per le regioni Cina.
  - Ulteriori miglioramenti minori.

Per maggiori informazioni e per scaricare i nuovi driver, le note di rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC.

30 marzo 2022

Data pubblicazione: 30/03/2022

Athena annuncia le correzioni e i miglioramenti seguenti.

- Interrogazioni tra regioni: ora puoi usare Athena per interrogare i dati che si trovano in Regioni AWS un bucket Amazon S3 in Asia Pacifico (Hong Kong), Medio Oriente (Bahrein), Africa (Città del Capo) ed Europa (Milano). L'interrogazione tra regioni non è supportata nelle regioni cinesi.
  - Per un elenco delle applicazioni Regioni AWS in cui Athena è disponibile, consulta Endpoint e quote di Amazon Athena.
  - Per informazioni sull'attivazione di un Regione AWS account disabilitato per impostazione predefinita, consulta Enabling a Region.
  - Per ulteriori informazioni relative all'esecuzione di query tra regioni, consulta Esecuzione di query tra regioni.

18 marzo 2022

Data pubblicazione: 18/03/2022

Athena annuncia le correzioni e i miglioramenti seguenti.

- Filtro dinamico: il Filtro dinamico è stato migliorato per le colonne intere applicando efficacemente il filtro a ciascun registro di una tabella corrispondente.
- Iceberg: è stato risolto un problema che causava errori durante la scrittura di file Parquet in Iceberg di dimensioni superiori a 2 GB.
- Output non compresso: le istruzioni CREATE TABLE ora supportano la scrittura di file non compressi. Per scrivere file non compressi, utilizza la sintassi seguente:
  - CREATE TABLE (file di testo o JSON): in `TBLPROPERTIES`, specifica `write.compression = NONE`.
  - CREATE TABLE (Parquet): in `TBLPROPERTIES`, specifica `parquet.compression = UNCOMPRESSED`.
  - CREATE TABLE (ORC): in `TBLPROPERTIES`, specifica `orc.compress = NONE`.

- Compressione: è stato risolto un problema relativo agli inserti per le tabelle di file di testo che creavano file compressi in un formato ma utilizzavano un'altra estensione di file in formato di compressione con metodi di compressione non predefiniti.
- Avro: sono stati risolti i problemi che si verificavano durante la lettura di decimali del tipo fisso dai file Avro.

2 marzo 2022

Data pubblicazione: 02/03/2022

Athena annuncia la seguenti funzionalità e miglioramenti.

- Ora puoi concedere al proprietario del bucket Amazon S3 il controllo di accesso completo ai risultati delle query quando le ACL sono abilitate per il bucket dei risultati della query. Per ulteriori informazioni, consulta Specificare una posizione dei risultati delle query.
- Ora puoi aggiornare le query denominate esistenti. Per ulteriori informazioni, consulta Utilizzo di query salvate.

23 febbraio 2022

Data pubblicazione: 23/02/2022

Athena annuncia le correzioni e i miglioramenti delle prestazioni seguenti.

- Miglioramenti della gestione della memoria per migliorare le prestazioni e ridurre gli errori di memoria.
- Athena ora legge le colonne del timestamp ORC con le informazioni sul fuso orario memorizzate nei piè di pagina e scrive i file ORC con il fuso orario (UTC) nei piè di pagina. Ciò influisce solo sul comportamento della lettura del timestamp ORC se il file ORC da leggere è stato creato in un ambiente con fuso orario non UTC.
- Risolte le stime non corrette delle dimensioni della tabella dei collegamenti simbolici che determinavano piani di query non ottimali.
- Le viste esplose lateralamente ora possono essere interrogate nella console Athena da origini dati di metastore Hive.
- Miglioramento dei messaggi di errore di lettura di Amazon S3 per includere più informazioni dettagliate relative al codice di errore Amazon S3.

- Risolto un problema che causava l'incompatibilità dei file di output in formato ORC con Apache Hive 3.1.
- Risolto un problema che faceva fallire i nomi delle tabelle con virgolette in alcune query DML e DDL.

15 febbraio 2022

Data pubblicazione: 15/02/2022

Amazon Athena ha aumentato la quota di query DML attive in tutte le regioni. AWS Le query attive includono sia query in esecuzione che in coda. Con questa modifica, ora puoi avere più query DML attive rispetto a prima.

Per ulteriori informazioni sulle Service Quotas di Athena, consulta la sezione Service Quotas (Quote di Servizio). Per le quote di query nella regione in cui utilizzi Athena, consulta Endpoint e quote di Amazon Athena nella Riferimenti generali di AWS.

Per monitorare l'utilizzo della quota, puoi utilizzare i parametri di CloudWatch utilizzo. Athena pubblica il parametro `ActiveQueryCount` nello spazio dei nomi `AWS/Usage`. Per ulteriori informazioni, consulta Monitoraggio dei parametri di utilizzo di Athena.

Dopo aver esaminato l'utilizzo, puoi usare la console Service Quotas per richiedere un aumento della quota. Se in precedenza hai richiesto un aumento della quota per il tuo account, la quota richiesta viene comunque applicata se supera la nuova quota di query DML attiva predefinita. In caso contrario, tutti gli account utilizzano il nuovo valore predefinito.

14 febbraio 2022

Data pubblicazione: 14/02/2022

Questa versione aggiunge il `ErrorType` sottocampo all'oggetto di AthenaErrorrisposta nell'azione API GetQueryExecutionAthena.

Se il campo `ErrorCategory` esistente indica l'origine generale di una query non riuscita (sistema, utente o altro), il nuovo campo `ErrorType` fornisce informazioni più granulari relative all'errore verificato. Combina le informazioni di entrambi i campi per ottenere informazioni dettagliate sulle cause dell'errore della query.

Per ulteriori informazioni, consulta catalogo degli errori Athena.

9 febbraio 2022

Data pubblicazione: 09/02/2022

La console Athena precedente non è più disponibile. La nuova console di Athena supporta tutte le funzionalità della console precedente, ma con un'interfaccia moderna e più facile da usare e include nuove funzionalità che migliorano l'esperienza di sviluppo di query, analisi dei dati e gestione dell'utilizzo. Per utilizzare la nuova console Athena, visita il sito https://console.aws.amazon.com/athena/.

8 febbraio 2022

Data pubblicazione: 08/02/2022

Proprietario previsto del bucket: come misura di sicurezza aggiuntiva, ora puoi facoltativamente specificare l' Account AWS ID che prevedi sia il proprietario del bucket di posizione di output dei risultati delle query in Athena. Se l'ID account del proprietario del bucket dei risultati della query non corrisponde all'ID specificato, i tentativi di output nel bucket avranno esito negativo con un errore di autorizzazione di Amazon S3. Puoi regolare questa impostazione a livello di client o gruppo di lavoro.

Per ulteriori informazioni, consulta Specificare una posizione dei risultati delle query.

28 gennaio 2022

Data pubblicazione: 28/01/2022

Athena annuncia i seguenti miglioramenti alle funzionalità del motore:

- Apache Hudi: le query snapshot sulle tabelle Hudi Merge on Read (MoR) possono ora leggere le colonne del timestamp che hanno il tipo di dati `INT64`.
- Query UNION: miglioramento delle prestazioni e riduzione della scansione dei dati per alcune query `UNION` che eseguono la scansione della stessa tabella più volte.
- Query disgiunte: miglioramento delle prestazioni per le query che hanno solo valori disgiunti per ogni colonna di partizione sul filtro.
- Miglioramenti alla proiezione delle partizioni
  - Ora sono consentiti più valori di disgiunzione sulla condizione del filtro per le colonne del tipo `injected`. Per ulteriori informazioni, consulta Tipo iniettato.
  - Miglioramento delle prestazioni per colonne di tipi basati su stringhe come `CHAR` o `VARCHAR` che hanno solo valori disgiunti sul filtro.

13 gennaio 2022

Data pubblicazione: 13/01/2022

Rilasciati i driver JDBC 2.0.27 e ODBC 1.1.15 per Athena.

Il driver JDBC 2.0.27 include le seguenti modifiche:

- Il driver è stato aggiornato per recuperare i cataloghi esterni.
- Il numero di versione esteso del driver è ora incluso nella stringa `user-agent` come parte della chiamata API di Athena.

Il driver ODBC 1.1.15 include le seguenti modifiche:

- Corregge un problema con le seconde chiamate a `SQLParamData()`.

Per maggiori informazioni su queste modifiche e per scaricare i nuovi driver, le note di rilascio e la documentazione, consultare Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

Note di rilascio di Athena per il 2021

26 novembre 2021

Data pubblicazione: 26/11/2021

Athena annuncia l'anteprima pubblica delle transazioni ACID di Athena che aggiungono operazioni di scrittura, eliminazione, aggiornamento e temporali al linguaggio DML (Data Manipulation Language) di Athena SQL. Le transazioni ACID di Athena consentono a più utenti di apportare simultaneamente modifiche affidabili a livello di riga ai dati Amazon S3. Poiché sono basate su formati di tabelle Apache Iceberg, le transazioni ACID di Athena sono compatibili con altri servizi e motori come Amazon EMR e Apache Spark che supportano anche i formati di tabella Iceberg.

Le transazioni ACID di Athena e la sintassi SQL familiare semplificano gli aggiornamenti dei dati aziendali e normativi. Ad esempio, per rispondere a una richiesta di cancellazione dei dati, è possibile eseguire un'operazione SQL `DELETE`. Per apportare correzioni manuali ai registri, è possibile utilizzare una singola istruzione `UPDATE`. Per recuperare i dati eliminati di recente, è possibile emettere query temporali tramite un'istruzione `SELECT`. Le transazioni Athena sono disponibili tramite la console Athena, le operazioni API e i driver ODBC e JDBC.

Per ulteriori informazioni, consulta Utilizzo delle transazioni ACID di Athena.

24 novembre 2021

Data pubblicazione: 24/11/2021

Athena annuncia il supporto per la lettura e la scrittura di dati Parquet e file di testo compressi con ZStandard. Quando scrive dati compressi in ZStandard, Athena utilizza il livello di compressione 3 di ZStandard.

Per ulteriori informazioni sulla compressione dei dati in Athena, consulta Supporto della compressione in Athena.

22 novembre 2021

Data pubblicazione: 22/11/2021

Ora puoi gestire i AWS Step Functions flussi di lavoro dalla console Amazon Athena, semplificando la creazione di pipeline di elaborazione dati scalabili, l'esecuzione di query basate su logiche aziendali personalizzate, l'automazione delle attività amministrative e di avviso e altro ancora.

Step Functions è ora integrato con la console aggiornata di Athena ed è possibile utilizzarlo per visualizzare un diagramma interattivo del flusso di lavoro delle macchine statali che richiamano Athena. Per iniziare, seleziona Workflows (Flussi di lavoro) dal pannello di navigazione sulla sinistra. Se disponi di macchine a stato esistenti con query Athena, seleziona una macchina a stato per visualizzare un diagramma interattivo del flusso di lavoro. Per i nuovi utenti di Step Functions, è possibile iniziare avviando un progetto di esempio dalla console Athena e personalizzandolo in base ai propri casi d'uso.

Per ulteriori informazioni, consulta Creare e orchestrare pipeline ETL utilizzando Amazon Athena oppure AWS Step Functions consulta la documentazione di Step Functions.

18 novembre 2021

Data pubblicazione: 18/11/2021

Athena annuncia nuove funzionalità e nuovi miglioramenti.

- Support spill-to-disk per le query di aggregazione che contengono `DISTINCT` o entrambi, come nell'esempio seguente: `ORDER BY`

```
SELECT array_agg(orderstatus ORDER BY orderstatus)
FROM orders
GROUP BY orderpriority, custkey
```

- Risolti problemi di gestione della memoria per le query che utilizzano `DISTINCT`. Per evitare messaggi di errore come Query exhausted resources at this scale factor (Interroga le risorse esauste con questo fattore di scala) quando si utilizzano le query `DISTINCT`, scegli colonne che hanno una bassa cardinalità per `DISTINCT` o riduci la dimensione dei dati della query.
- Nelle query `SELECT COUNT(*)` che non specificano una colonna specifica, l'uso della memoria e le prestazioni risultano migliorati mantenendo solo il conteggio senza buffer di riga.
- Sono state introdotte le seguenti funzioni stringa.
  - `translate(source, from, to)`: Restituisce la stringa `source` con i caratteri trovati nella stringa `from` sostituita dai caratteri corrispondenti nella stringa `to`. Se la stringa `from` contiene duplicati, sarà utilizzato solo il primo. Se il carattere `source` non esiste nella stringa `from`, il carattere `source` viene copiato senza traduzione. Se l'indice del carattere corrispondente nella stringa `from` è maggiore della lunghezza della stringa `to`, il carattere viene omesso dalla stringa risultante.
  - `concat_ws(string0, array(varchar))`: Restituisce la concatenazione di elementi nell'array che utilizzano `string0` come separatore. Se `string0` è null, allora il valore restituito è null. Tutti i valori nulli nell'array vengono ignorati.
- Risolto un bug in cui le query non riuscivano quando si provava ad accedere a un campo secondario mancante in un `struct`. Le query ora restituiscono un valore nullo per il campo secondario mancante.
- Risolto un problema relativo all'hashing non coerente per il tipo di dati decimale.
- È stato risolto un problema che causava risorse esauste quando c'erano troppe colonne in una partizione.

17 novembre 2021

Data pubblicazione: 17/11/2021

Amazon Athena ora supporta l'indicizzazione delle partizioni per accelerare le query sulle tabelle partizionate nel AWS Glue Data Catalog.

Durante le query su tabelle partizionate, Athena recupera e filtra le partizioni della tabella disponibili nel sottoinsieme pertinente alla query. Quando vengono aggiunti nuovi dati e partizioni, è necessario

più tempo per elaborare le partizioni e il runtime delle query può aumentare. Per ottimizzare l'elaborazione delle partizioni e migliorare le prestazioni delle query su tabelle altamente partizionate, Athena adesso supporta indici di partizioni AWS Glue.

Per ulteriori informazioni, consulta AWS Glue indicizzazione e filtraggio delle partizioni.

16 novembre 2021

Data pubblicazione: 16/11/2021

La nuova e migliorata console Amazon Athena è ora disponibile a livello generale nelle aree AWS commerciali e GovCloud nelle regioni in cui è disponibile Athena. La nuova console di Athena supporta tutte le funzionalità della console precedente, ma con un'interfaccia moderna e più facile da usare e include nuove funzionalità che migliorano l'esperienza di sviluppo di query, analisi dei dati e gestione dell'utilizzo. Ora è possibile:

- Riorganizzare, navigare o chiudere più schede di query da una barra delle schede di query ridisegnata.
- Leggere e modificare le query più facilmente con una formattazione SQL e di testo migliorata.
- Copiare i risultati delle query negli appunti oltre a scaricare l'intero set di risultati.
- Ordina la cronologia delle query, le query salvate e i gruppi di lavoro e scegli quali colonne mostrare o nascondere.
- Utilizzare un'interfaccia semplificata per configurare origini dati e gruppi di lavoro in pochi clic.
- Impostare le preferenze per la visualizzazione dei risultati delle query, della cronologia delle query, del ritorno a capo e altro ancora.
- Aumentare la produttività con scorciatoie da tastiera nuove e migliorate e la documentazione integrata del prodotto.

Con l'annuncio di oggi, la console ridisegnata è quella di default. Per raccontarci la tua esperienza, sceglie Feedback nell'angolo in basso a sinistra della console.

Se lo desideri, puoi utilizzare la console precedente accedendo al tuo account Account AWS, scegliendo Amazon Athena e deselezionando New Athena experience dal pannello di navigazione a sinistra.

12 novembre 2021

Data pubblicazione: 12/11/2021

Ora puoi utilizzare Amazon Athena per eseguire query federate su origini dati situate in un account AWS diverso da quello in uso. Fino ad oggi, l'interrogazione di questi dati richiedeva che l'origine dati e il relativo connettore utilizzassero lo stesso metodo utilizzato dall'utente che Account AWS aveva interrogato i dati.

L'amministratore dei dati può abilitare le query federate tra account condividendo il connettore dati con l'account di un analista di dati. Un analista di dati può aggiungere un connettore dati che un amministratore dei dati ha condiviso con l'utente e il suo account. Le modifiche di configurazione al connettore nell'account di origine si applicano automaticamente al connettore condiviso.

Per informazioni sull'abilitazione di query federate tra account, consulta Abilitazione delle query federate tra account. Per informazioni sulle query di origini federate, consulta Utilizzo di Amazon Athena Federated Query e Esecuzione di query federate.

2 novembre 2021

Data pubblicazione: 02/11/2021

Adesso puoi utilizzare l'istruzione `EXPLAIN ANALYZE` in Athena per visualizzare il piano di esecuzione distribuito e il costo di ogni operazione per le query SQL.

Per ulteriori informazioni, consulta Utilizzo di EXPLAIN e EXPLAIN ANALYZE in Athena.

29 ottobre 2021

Data pubblicazione: 29/10/2021

Athena rilascia i driver JDBC 2.0.25 e ODBC 1.1.13 e annuncia caratteristiche e miglioramenti.

Driver JDBC e ODBC

Rilasciati i driver JDBC 2.0.25 e ODBC 1.1.13 per Athena. Entrambi i driver offrono supporto per l'autenticazione a più fattori SAML del browser, configurabile per funzionare con qualsiasi provider SAML 2.0.

Il driver JDBC 2.0.25 include le seguenti modifiche:

- Supporto per l'autenticazione SAML del browser. Il driver include un plug-in SAML del browser che può essere configurato per funzionare con qualsiasi provider SAML 2.0.

- Support per chiamate AWS Glue API. Puoi utilizzare il parametro `GlueEndpointOverride` per sovrascrivere l'endpoint AWS Glue .
- Modificata la class path `com.simba.athena.amazonaws` in `com.amazonaws`.

Il driver ODBC 1.1.13 include le seguenti modifiche:

- Supporto per l'autenticazione SAML del browser. Il driver include un plug-in SAML del browser che può essere configurato per funzionare con qualsiasi provider SAML 2.0. Per un esempio di come utilizzare il plug-in SAML del browser con il driver ODBC, consulta Configurazione di Single Sign-On tramite ODBC, SAML 2.0 e il provider di identità Okta.
- Ora puoi configurare la durata della sessione di ruolo quando si utilizza ADFS, Azure AD o Browser Azure AD per l'autenticazione.

Per maggiori informazioni su queste e altre modifiche e per scaricare i nuovi driver, le note di rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

Caratteristiche e miglioramenti

Athena annuncia la seguenti funzionalità e miglioramenti.

- È stata introdotta una regola di ottimizzazione per evitare scansioni di tabelle duplicate in determinati casi.

4 ottobre 2021

Data pubblicazione: 04/10/2021

Athena annuncia la seguenti funzionalità e miglioramenti.

- SQL OFFSET: la clausola SQL `OFFSET` è ora supportata nelle istruzioni `SELECT`. Per ulteriori informazioni, consulta SELECT.
- CloudWatch metriche di utilizzo: Athena ora pubblica la metrica `ActiveQueryCount` nel namespace. `AWS/Usage` Per ulteriori informazioni, consulta Monitoraggio dei parametri di utilizzo di Athena.
- Pianificazione query: risolto un bug che in rari casi poteva causare timeout di pianificazione delle query.

16 settembre 2021

Data pubblicazione: 16/09/2021

Athena annuncia le seguenti nuove funzionalità e miglioramenti.

Funzionalità

- È stato aggiunto il supporto per specificare la compressione in file di testo e JSON in CTAS utilizzando la proprietà della tabella `write_compression`. Puoi anche specificare la proprietà `write_compression` in CTAS per i formati Parquet e ORC. Per ulteriori informazioni, consulta Proprietà tabella CTAS.
- Il formato di compressione BZIP2 è ora supportato per la scrittura di file di testo e JSON. Per ulteriori informazioni sui formati di compressione in Athena, consulta Supporto della compressione in Athena.

Miglioramenti

- Risolto un bug in cui le informazioni di identità non venivano inviate alla funzione Lambda UDF.
- Risolto un problema di pushdown del predicato con le condizioni del filtro disgiunto.
- Risolto un problema di hashing per i tipi decimali.
- Risolto un problema di raccolta di statistiche non necessarie.
- Rimosso un messaggio di errore non coerente.
- Prestazioni broadcast join migliorate applicando l'eliminazione dinamica delle partizioni nel nodo worker.
- Per le query federate:
  - Configurazione modificata per ridurre il verificarsi di errori `CONSTRAINT_VIOLATION` nelle query federate.

15 settembre 2021

Data pubblicazione: 15/09/2021

Ora puoi utilizzare una console Amazon Athena riprogettata (anteprima). È stato rilasciato un nuovo driver JDBC per Athena.

Anteprima della console Athena

Ora puoi usare una console Amazon Athena riprogettata (anteprima) Regione AWS ovunque sia disponibile Athena. La nuova console supporta tutte le funzionalità della console esistente, ma da un'interfaccia moderna e più facile da usare.

Per passare alla nuova console, accedi al tuo account Account AWS e scegli Amazon Athena. Dalla barra di navigazione della AWS console, scegli Passa alla nuova console. Per tornare alla console di default, deseleziona New Athena experience (Nuova esperienza Athena) dal pannello di navigazione sul lato sinistro.

Inizia a utilizzare la nuova console oggi stesso. Per raccontarci la tua esperienza, scegli Feedback nell'angolo in basso a sinistra.

Driver JDBC Athena 2.0.24

Athena annuncia la disponibilità del driver JDBC versione 2.0.24 per Athena. Questa versione aggiorna il supporto proxy per tutti i provider di credenziali. Il driver ora supporta l'autenticazione proxy per tutti gli host che non sono supportati dalla proprietà di connessione `NonProxyHosts`.

Per comodità, questa versione include il download del driver JDBC con e senza l' AWS SDK. Questa versione del driver JDBC consente di avere sia l'SDK AWS che il driver JDBC Athena incorporati nel progetto.

Per maggiori informazioni e per scaricare il nuovo driver, le note sul rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC.

31 agosto 2021

Data pubblicazione: 31/08/2021

Athena annuncia i seguenti miglioramenti alle funzioni e correzioni di bug.

- Miglioramenti alla federazione Athena – Athena ha aggiunto il supporto ai tipi di mappe e un migliore supporto per i tipi complessi come parte dell'SDK Athena Query Federation. Questa versione include anche alcuni miglioramenti della memoria e l'ottimizzazione delle prestazioni.
- Nuove categorie di errori – Introdotte le categorie di errori `USER` e `SYSTEM` nei messaggi di errore. Queste categorie consentono di distinguere tra errori che è possibile correggere autonomamente (`USER`) ed errori che possono richiedere assistenza da parte del supporto Athena (`SYSTEM`).

- Messaggistica degli errori di query federate – Aggiornate le categorizzazioni `USER_ERROR` per gli errori correlati alle query federate.
- JOIN: sono stati corretti i bug e i problemi di memoria spill-to-disk correlati per migliorare le prestazioni e ridurre gli errori di memoria nelle operazioni. `JOIN`

12 agosto 2021

Data pubblicazione: 12/08/2021

Rilasciato il driver ODBC 1.1.12 per Athena. Questa versione corregge i problemi relativi a `SQLPrepare()`, `SQLGetInfo()` e `EndpointOverride`.

Per scaricare il nuovo driver, le note sul rilascio e la documentazione, consulta Connessione ad Amazon Athena con ODBC.

6 agosto 2021

Data pubblicazione: 06/08/2021

Amazon Athena annuncia la disponibilità di Athena e delle sue funzioni nella Regione Asia Pacifico (Osaka).

Questa versione amplia la disponibilità di Athena per includere le Regioni Asia Pacifico (Hong Kong), Asia Pacifico (Mumbai), Asia Pacifico (Osaka), Asia Pacifico (Seoul), Asia Pacifico (Singapore), Asia Pacifico (Sydney) e Asia Pacifico (Tokyo). Per un elenco completo dei servizi Servizi AWS disponibili in queste e in altre regioni, consulta l'elenco dei servizi Regione AWS al.

5 agosto 2021

Data pubblicazione: 05/08/2021

Puoi utilizzare l'istruzione `UNLOAD` per scrivere l'output di una query `SELECT` nei formati PARQUET, ORC, AVRO e JSON.

Per ulteriori informazioni, consulta UNLOAD.

30 luglio 2021

Data pubblicazione: 30/07/2021

Athena annuncia i seguenti miglioramenti alle funzioni e correzioni di bug.

- Filtraggio dinamico e cesura delle partizioni – I miglioramenti apportati aumentano le prestazioni e riducono la quantità di dati scansionati in alcune query, come nell'esempio seguente.

  Questo esempio presuppone che `Table_B` sia una tabella non partizionata con dimensioni di file con una dimensione totale massima di 20 MB. Per query come questa, meno dati vengono letti da `Table_A` e la query viene completata più rapidamente.

  ```
  SELECT *
  FROM Table_A
  JOIN Table_B ON Table_A.date = Table_B.date
  WHERE Table_B.column_A = "value"
  ```

- ORDER BY with LIMIT, DISTINCT with LIMIT – Miglioramento delle prestazioni per le query che utilizzano `ORDER BY` o `DISTINCT` seguito da una clausola `LIMIT`.
- File S3 Glacier Deep Archive – Quando Athena esegue una query su una tabella che contiene un mix di file S3 Glacier Deep Archive e file diversi da S3 Glacier Glacier, Athena ora salta i file S3 Glacier Deep Archive. In precedenza, era necessario spostare manualmente questi file dal percorso della query o la query avrebbe avuto esito negativo. Se si desidera utilizzare Athena per interrogare gli oggetti nell'archivio S3 Glacier Deep Archive, è necessario ripristinarli. Per ulteriori informazioni, consulta Ripristino di un oggetto archiviato nella Guida per sviluppatori di Amazon S3.
- Corretto un problema per il quale i file vuoti creati dalla proprietà tabella CTAS `bucketed_by` non venivano crittografati correttamente.

21 luglio 2021

Data pubblicazione: 21/07/2021

Con il rilascio di luglio 2021 di Desktop di Microsoft Power BI, puoi creare report e dashboard utilizzando un connettore di origine dati nativo per Amazon Athena. Il connettore per Amazon Athena è disponibile come connettore standard in Power BI DirectQuery, supporta e consente l'analisi su set di dati di grandi dimensioni e l'aggiornamento dei contenuti tramite Power BI Gateway.

Poiché il connettore utilizza il nome origine dati ODBC (DSN) esistente per connettersi ed eseguire query su Athena, richiede il driver Athena ODBC. Per scaricare l'ultimo driver ODBC, consulta Connessione ad Amazon Athena con ODBC.

Per ulteriori informazioni, consulta Utilizzo del connettore Power BI di Amazon Athena.

16 luglio 2021

Data pubblicazione: 16/07/2021

Amazon Athena ha aggiornato la sua integrazione con Apache Hudi. Hudi è un framework open source per la gestione dei dati che semplifica l'elaborazione incrementale dei dati nei data lake Amazon S3. L'integrazione aggiornata consente di utilizzare Athena per eseguire query sulle tabelle Hudi 0.8.0 gestite tramite Amazon EMR, Apache Spark, Apache Hive o altri servizi compatibili. Inoltre, Athena ora supporta due funzionalità aggiuntive: query snapshot su tabelle Unisci in lettura (MoR) e supporto di lettura su tabelle Bootstrap.

Apache Hudi fornisce l'elaborazione dei dati a livello di registro che può aiutare a semplificare lo sviluppo di pipeline di Change Data Capture (CDC), a rispettare gli aggiornamenti ed eliminazioni dovuti al GDPR e a gestire meglio i dati in streaming da sensori o dispositivi che richiedono l'inserimento dei dati e l'aggiornamento degli eventi. La versione 0.8.0 semplifica la migrazione di tabelle Parquet di grandi dimensioni a Hudi senza copiare i dati, in modo da poterli interrogare e analizzare tramite Athena. Puoi usare il nuovo supporto di Athena per le query snapshot per avere viste quasi in tempo reale degli aggiornamenti delle tabelle di streaming.

Per ulteriori informazioni sull'uso di Hudi con Athena, consulta Utilizzo di Athena per eseguire query sui set di dati Apache Hudi.

8 luglio 2021

Data pubblicazione: 08/07/2021

Rilasciato il driver ODBC 1.1.11 per Athena. Il driver ODBC può ora autenticare la connessione tramite JSON Web Token (JWT). In Linux, il valore predefinito per la proprietà Workgroup è stato impostato su Primary.

Per maggiori informazioni e per scaricare il nuovo driver, le note sul rilascio e la documentazione, consulta Connessione ad Amazon Athena con ODBC.

1° luglio 2021

Data pubblicazione: 01/07/2021

Il 1° luglio 2021 è terminato il trattamento speciale dell'anteprima dei gruppi di lavoro. Anche se i gruppi di lavoro `AmazonAthenaPreviewFunctionality` mantengono il proprio nome, non hanno più uno stato speciale. È possibile continuare a utilizzare i gruppi di lavoro

`AmazonAthenaPreviewFunctionality` per visualizzare, modificare, organizzare ed eseguire query. Tuttavia, le query che utilizzano funzionalità precedentemente in anteprima sono ora soggette ai termini e alle condizioni di fatturazione standard di Athena. Per informazioni sulla fatturazione, consulta Prezzi di Amazon Athena.

23 giugno 2021

Data pubblicazione: 23/06/2021

Rilasciati i driver JDBC 2.0.23 e ODBC 1.1.10 per Athena. Entrambi i driver offrono prestazioni di lettura migliorate e supportano le istruzioni EXPLAIN e le query parametrizzate.

Le istruzioni `EXPLAIN` mostrano il piano di esecuzione logico o distribuito di una query SQL. Le query con parametri consentono di utilizzare la stessa query più volte con valori diversi forniti in fase di esecuzione.

Il rilascio JDBC aggiunge anche il supporto per Active Directory Federation Services 2019 e un'opzione personalizzata di override degli endpoint per AWS STS. La versione ODBC consente di correggere un problema con le credenziali dei profili IAM.

Per maggiori informazioni e per scaricare i nuovi driver, le note sul rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

12 maggio 2021

Data pubblicazione: 12/05/2021

Ora puoi usare Amazon Athena per registrare un AWS Glue catalogo da un account diverso dal tuo. Dopo aver configurato le autorizzazioni IAM richieste per AWS Glue, puoi utilizzare Athena per eseguire query tra account.

Per ulteriori informazioni, consulta Registrazione e AWS Glue Data Catalog da un altro account e Accesso tra account ai cataloghi dati AWS Glue.

10 maggio 2021

Data pubblicazione: 10/05/2021

Rilasciato il driver ODBC versione 1.1.9.1001 per Athena. Questa versione risolve un problema con il tipo di autenticazione `BrowserAzureAD` quando si usa Azure Active Directory (AD).

Per scaricare i nuovi driver, le note sul rilascio e la documentazione, consulta Connessione ad Amazon Athena con ODBC.

5 maggio 2021

Data pubblicazione: 05/05/2021

Ora puoi utilizzare il connettore Amazon Athena Vertica nelle query federate per interrogare le origini dati Vertica da Athena. Ad esempio, è possibile eseguire query analitiche su un data warehouse in Vertica e su un data lake in Amazon S3.

Per distribuire il connettore Athena Vertica, visita AthenaVerticaConnectorla pagina in. AWS Serverless Application Repository

Il connettore Amazon Athena Vertica espone diverse opzioni di configurazione attraverso le variabili di ambiente Lambda. Per informazioni sulle opzioni di configurazione, i parametri, le stringhe di connessione, l'implementazione e le limitazioni, consulta la pagina Connettore Amazon Athena Vertica.

Per informazioni dettagliate sull'utilizzo del connettore Vertica, consulta Esecuzione di query su un'origine dati Vertica in Amazon Athena con l'SDK Athena Federated Query nel blog AWS Big Data.

30 aprile 2021

Data pubblicazione: 30/04/2021

Rilasciati i driver JDBC 2.0.21 e ODBC 1.1.9 per Athena. Entrambe le versioni supportano l'autenticazione SAML con Azure Active Directory (AD) e l'autenticazione SAML con. PingFederate Il rilascio JDBC supporta anche query parametrizzate. Per informazioni sulle query con parametri in Athena, consulta Utilizzo di query parametrizzate.

Per scaricare i nuovi driver, le note sul rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

29 aprile 2021

Data pubblicazione: 29/04/2021

Amazon Athena annuncia la disponibilità della versione 2 del motore Athena nelle Regioni Cina (Pechino) e Cina (Ningxia).

Per ulteriori informazioni sulla versione 2 del motore Athena, consulta Versione 2 del motore Athena.

26 Aprile 2021

Data pubblicazione: 26/04/2021

Le funzioni del valore finestra nella versione 2 del motore Athena ora supportano `IGNORE NULLS` e `RESPECT NULLS`.

Per ulteriori informazioni, consulta Funzioni del valore nella documentazione di Presto.

21 aprile 2021

Data pubblicazione: 21/04/2021

Amazon Athena annuncia la disponibilità della versione 2 del motore Athena nelle Regioni Europa (Milano) e Africa (Città del Capo).

Per ulteriori informazioni sulla versione 2 del motore Athena, consulta Versione 2 del motore Athena.

5 aprile 2021

Data pubblicazione: 05/04/2021

Istruzioni EXPLAIN

Ora puoi usare l'istruzione `EXPLAIN` in Athena per visualizzare il piano di esecuzione per le query SQL.

Per ulteriori informazioni, consulta Utilizzo di EXPLAIN e EXPLAIN ANALYZE in Athena e Capire i risultati dell'istruzione EXPLAIN di Athena.

SageMaker Modelli di Machine Learning nelle query SQL

L'inferenza dei modelli di machine learning con Amazon SageMaker è ora disponibile a livello generale per Amazon Athena. Utilizza i modelli di machine learning nelle query SQL per semplificare attività complesse quali il rilevamento delle anomalie, l'analisi delle coorti dei clienti e le previsioni in serie temporale richiamando una funzione in una query SQL.

Per ulteriori informazioni, consulta Utilizzo di Machine Learning (ML) con Amazon Athena.

Funzioni definite dall'utente (UDF)

Le funzioni definite dall'utente (UDF) sono ora disponibili a livello generale per Athena. Utilizza le UDF per sfruttare funzioni personalizzate che elaborano registri o gruppi di registri in una singola query SQL.

Per ulteriori informazioni, consulta Esecuzione di query con funzioni definite dall'utente.

30 marzo 2021

Data pubblicazione: 30/03/2021

Amazon Athena annuncia la disponibilità della versione 2 del motore Athena nelle Regioni Asia Pacifico (Hong Kong) e Medio Oriente (Bahrein).

Per ulteriori informazioni sulla versione 2 del motore Athena, consulta Versione 2 del motore Athena.

25 marzo 2021

Data pubblicazione: 25/03/2021

Amazon Athena annuncia la disponibilità della versione 2 del motore Athena nelle Regioni Europa (Stoccolma).

Per ulteriori informazioni sulla versione 2 del motore Athena, consulta Versione 2 del motore Athena.

5 marzo 2021

Data pubblicazione: 05/03/2021

Amazon Athena annuncia la disponibilità della versione 2 del motore Athena nelle Regioni Canada (Central), Europa (Francoforte) e Sud America (San Paolo).

Per ulteriori informazioni sulla versione 2 del motore Athena, consulta Versione 2 del motore Athena.

25 febbraio 2021

Data pubblicazione: 25/02/2021

Amazon Athena annuncia la disponibilità generale del motore Athena versione 2 nelle Regioni Asia Pacifico (Seoul), Asia Pacifico (Singapore), Asia Pacifico (Sidney), Europa (Londra) ed Europa (Parigi).

Per ulteriori informazioni sulla versione 2 del motore Athena, consulta Versione 2 del motore Athena.

Note di rilascio di Athena per il 2020

16 dicembre 2020

Data pubblicazione: 16/12/2020

Amazon Athena annuncia la disponibilità di Athena Engine versione 2, Athena Federated Query e in altre regioni. AWS PrivateLink

Versione 2 del motore Athena e Athena Federated Query

Amazon Athena annuncia la disponibilità generale del motore Athena versione 2 e di Athena Federated Query nelle Regioni Asia Pacifico (Mumbai), Asia Pacifico (Tokyo), Europa (Irlanda) e Stati Uniti occidentali (California settentrionale). La versione 2 del motore Athena e le query federate sono già disponibili nelle Regioni Stati Uniti orientali (Virginia settentrionale), Stati Uniti orientali (Ohio) e Stati Uniti occidentali (Oregon).

Per ulteriori informazioni, consulta Versione 2 del motore Athena e Utilizzo di Amazon Athena Federated Query.

AWS PrivateLink

AWS PrivateLink for Athena è ora supportato nella regione Europa (Stoccolma). Per informazioni su AWS PrivateLink Athena, vedere. Connessione ad Amazon Athena utilizzando un endpoint VPC di interfaccia

24 novembre 2020

Data pubblicazione: 24/11/2020

Rilasciati i driver JDBC 2.0.16 e ODBC 1.1.6 per Athena. Queste versioni supportano, a livello di account, l'autenticazione a più fattori (MFA) Okta. Puoi anche usare MFA Okta per configurare l'autenticazione SMS e l'autenticazione Google Authenticator come fattori.

Per scaricare i nuovi driver, le note sul rilascio e la documentazione, consulta Connessione ad Amazon Athena con JDBC e Connessione ad Amazon Athena con ODBC.

11 novembre 2020

Data pubblicazione: 11/11/2020

Amazon Athena annuncia la disponibilità generale della versione 2 del motore Athena e delle query federate nelle Regioni Stati Uniti orientali (Virginia settentrionale), Stati Uniti orientali (Ohio) e Stati Uniti occidentali (Oregon).

Versione 2 del motore Athena

Amazon Athena annuncia la disponibilità generale di una nuova versione del motore per le query, la versione 2 del motore Athena, e delle query federate nelle Regioni Stati Uniti orientali (Virginia settentrionale), Stati Uniti orientali (Ohio) e Stati Uniti occidentali (Oregon).

La versione 2 del motore Athena include miglioramenti delle prestazioni e nuove funzionalità, come il supporto per l'evoluzione dello schema per i dati in formato Parquet, funzioni geospaziali aggiuntive, supporto per la lettura dello schema nidificato per ridurre i costi e migliorare le prestazioni nelle operazioni JOIN e AGGREGATE.

- Per informazioni su miglioramenti, interruzioni delle modifiche e correzioni di bug, consulta Versione 2 del motore Athena.
- Per informazioni su come eseguire l'aggiornamento, consulta Modifica delle versioni del motore Athena.
- Per informazioni su come testare le query, consulta Test delle query in anticipo rispetto all'aggiornamento della versione del motore.

Query SQL federate

Ora puoi utilizzare la query federata di Athena nelle Regioni Stati Uniti orientali (Virginia settentrionale), Stati Uniti orientali (Ohio) e Stati Uniti occidentali (Oregon) senza usare il gruppo di lavoro `AmazonAthenaPreviewFunctionality`.

Utilizza le query SQL federate per eseguire query SQL su origini dati relazionali, non relazionali, oggetti e personalizzate. Con le query federate, puoi inviare una singola query SQL che scansiona dati da molteplici origini eseguite in locale o ospitate nel cloud.

L'esecuzione di analisi sui dati distribuiti tra le applicazioni può essere complessa e richiedere tempo per i seguenti motivi:

- I dati richiesti per l'analisi sono spesso distribuiti tra archivi di dati relazionali, di valore chiave, di documenti, in-memory, di ricerca, di grafi, di oggetti, di serie temporali e registri.
- Per analizzare dati in queste origini, gli analisti costruiscono complesse pipeline per estrarre, trasformare e caricare in un data warehouse in modo che sia possibile eseguire query sui dati.
- Accedere a dati da diverse origini richiede l'apprendimento di nuovi linguaggi di programmazione e di costruzioni di accesso ai dati.

Le query SQL federate in Athena eliminano questa complessità permettendo agli utenti di eseguire query sui dati in loco da qualsiasi luogo risiedano. Gli analisti possono utilizzare i costrutti SQL per unire i dati `JOIN` di più origini dati per un'analisi rapida e archiviare i risultati in Amazon S3 per un uso successivo.

Connettori origine dati

Athena utilizza i connettori Athena Data Source su AWS Lambda per eseguire query federate. I seguenti connettori open source e pre-costruiti sono stati scritti e testati da Athena. Usali per eseguire query SQL in Athena sulle loro origini dati corrispondenti.

- CloudWatch
- Parametri CloudWatch
- DocumentDB
- DynamoDB
- OpenSearch
- HBase
- Neptune
- Redis
- Timestream
- TPC Benchmark DS (TPC-DS)

Connettori origine dati personalizzati

Utilizzando l'SDK Athena Query Federation, gli sviluppatori possono creare connettori per qualsiasi origine dati Athena per consentire l'esecuzione di query SQL sull'origine dati. Athena Query Federation Connector estende i vantaggi dell'interrogazione federata oltre ai connettori forniti. AWS

Poiché i connettori funzionano AWS Lambda, non è necessario gestire l'infrastruttura o pianificare la scalabilità in base ai picchi di domanda.

Fasi successive

- Per ulteriori informazioni sulla funzione delle query federate, consulta Utilizzo di Amazon Athena Federated Query.
- Per iniziare a utilizzare un connettore esistente, consulta Distribuzione di un connettore e connessione a un'origine dati.
- Per informazioni su come creare un connettore di origine dati personalizzato utilizzando l'SDK Athena Query Federation, vedi Example Athena Connector on. GitHub

22 ottobre 2020

Data pubblicazione: 22/10/2020

Ora puoi chiamare Athena con. AWS Step Functions AWS Step Functions può controllarne alcuni Servizi AWS direttamente utilizzando Amazon States Language. È possibile utilizzare Step Functions con Athena per avviare e interrompere l'esecuzione di query, ottenere risultati di query, eseguire query di dati ad-hoc o pianificate e recuperare i risultati da data lake in Amazon S3.

Per ulteriori informazioni, consulta Chiama Athena con Step Functions nella Guida per gli sviluppatori di AWS Step Functions .

29 luglio 2020

Data pubblicazione: 29/07/2020

Rilasciata la versione 2.0.13 del driver JDBC. Questa versione supporta l'utilizzo di più cataloghi dati registrati con Athena, del servizio Okta per l'autenticazione e delle connessioni agli endpoint VPC.

Per scaricare e utilizzare la nuova versione del driver, consulta Connessione ad Amazon Athena con JDBC.

9 luglio 2020

Data pubblicazione: 09/07/2020

Amazon Athena aggiunge il supporto per l'interrogazione di set di dati Hudi compatti e aggiunge la AWS CloudFormation `AWS::Athena::DataCatalog` risorsa per la creazione, l'aggiornamento o l'eliminazione dei cataloghi di dati registrati in Athena.

Esecuzione di query su set di dati Apache Hudi

Apache Hudi è un framework open source per la gestione dei dati che semplifica l'elaborazione incrementale dei dati. Amazon Athena ora supporta l'esecuzione di query sulla visualizzazione ottimizzata per la lettura di un set di dati Apache Hudi nel tuo data lake basato su Amazon S3.

Per ulteriori informazioni, consulta Utilizzo di Athena per eseguire query sui set di dati Apache Hudi.

AWS CloudFormation Risorsa del catalogo dati

Per utilizzare la funzione query federate di Amazon Athena per eseguire query su qualsiasi origine dati, devi prima registrare il tuo catalogo dati in Athena. Ora puoi utilizzare la AWS CloudFormation `AWS::Athena::DataCatalog` risorsa per creare, aggiornare o eliminare i cataloghi di dati registrati in Athena.

Per ulteriori informazioni, consulta AWS::Athena::DataCatalogla Guida per l'AWS CloudFormation utente.

1 giugno 2020

Data pubblicazione: 01/06/2020

Utilizzo di metastore Hive Apache come metacatalogo con Amazon Athena

Ora è possibile connettere Athena a uno o più metastore Hive Apache in aggiunta a AWS Glue Data Catalog con Athena.

Per connettersi a un metastore Hive con hosting autonomo, è necessario un connettore per metastore Athena Hive. Athena offre un connettore per l'implementazione di riferimento che è possibile utilizzare. Il connettore opera nel tuo account come una funzione AWS Lambda .

Per ulteriori informazioni, consulta Utilizzo di un connettore dati Athena per il metastore Hive esterno.

21 maggio 2020

Data pubblicazione: 21/05/2020

Amazon Athena aggiunge il supporto per la proiezione delle partizioni. Utilizzare la proiezione delle partizioni per velocizzare l'elaborazione delle query di tabelle altamente partizionate e automatizzare la gestione delle partizioni. Per ulteriori informazioni, consulta Proiezione delle partizioni con Amazon Athena.

1 Aprile 2020

Data pubblicazione: 01/04/2020

Oltre alla Regione Stati Uniti orientali (Virginia settentrionale), le funzioni query federata, funzioni definite dall'utente (UDF), inferenza del machine learning e metastore Hive esterno di Amazon Athena sono ora disponibili in anteprima nelle Regioni Asia Pacifico (Mumbai), Europa (Irlanda) e Stati Uniti occidentali (Oregon).

11 marzo 2020

Data pubblicazione: 11/03/2020

Amazon Athena ora pubblica eventi EventBridge Amazon per le transizioni di stato delle query. Quando una query passa da uno stato all'altro, ad esempio da In esecuzione a uno stato terminale come Riuscito o Annullato, Athena pubblica un evento di modifica dello stato della query su. EventBridge L'evento contiene informazioni sulla transizione di stato della query. Per ulteriori informazioni, consulta Monitoraggio delle query Athena con eventi Amazon EventBridge.

6 marzo 2020

Data pubblicazione: 06/03/2020

Ora puoi creare e aggiornare gruppi di lavoro Amazon Athena utilizzando la risorsa. AWS CloudFormation `AWS::Athena::WorkGroup` Per ulteriori informazioni, consulta AWS::Athena::WorkGroupla Guida per l'AWS CloudFormation utente.

Note di rilascio di Athena per il 2019

26 novembre 2019

Data pubblicazione: 17/12/2019

Amazon Athena aggiunge il supporto per l'esecuzione di query SQL su origini dati relazionali, non relazionali, oggetti e personalizzate, richiamando modelli di machine learning nelle query SQL, UDF

(User Defined Functions) (anteprima), utilizzando metastore Hive Apache come catalogo di metadati con Amazon Athena (anteprima) e quattro parametri correlati alla query aggiuntivi.

Query SQL federate

Utilizza le query SQL federate per eseguire query SQL su origini dati relazionali, non relazionali, oggetti e personalizzate.

Ora puoi utilizzare la query federata Athena per eseguire la scansione dei dati archiviati in origini dati relazionali, non relazionali, oggetti e personalizzate. Con le query federate, puoi inviare una singola query SQL che scansiona dati da molteplici origini eseguite in locale o ospitate nel cloud.

L'esecuzione di analisi sui dati distribuiti tra le applicazioni può essere complessa e richiedere tempo per i seguenti motivi:

- I dati richiesti per l'analisi sono spesso distribuiti tra archivi di dati relazionali, di valore chiave, di documenti, in-memory, di ricerca, di grafi, di oggetti, di serie temporali e registri.
- Per analizzare dati in queste origini, gli analisti costruiscono complesse pipeline per estrarre, trasformare e caricare in un data warehouse in modo che sia possibile eseguire query sui dati.
- Accedere a dati da diverse origini richiede l'apprendimento di nuovi linguaggi di programmazione e di costruzioni di accesso ai dati.

Le query SQL federate in Athena eliminano questa complessità permettendo agli utenti di eseguire query sui dati in loco da qualsiasi luogo risiedano. Gli analisti possono utilizzare i costrutti SQL per unire i dati `JOIN` di più origini dati per un'analisi rapida e archiviare i risultati in Amazon S3 per un uso successivo.

Connettori origine dati

Athena esegue le query federate utilizzando i connettori Athena Data Source in esecuzione in AWS Lambda. Usa questi connettori di origine dati open source per eseguire query SQL federate in Athena su database relazionali conformi ad Amazon DynamoDB, Apache HBase, Amazon Document DB, Amazon, CloudWatch CloudWatch Amazon Metrics e database relazionali conformi a JDBC come MySQL e PostgreSQL con licenza Apache 2.0.

Connettori origine dati personalizzati

Utilizzando l'SDK Athena Query Federation, gli sviluppatori possono creare connettori per qualsiasi origine dati Athena per consentire l'esecuzione di query SQL sull'origine dati. Athena Query

Federation Connector estende i vantaggi dell'interrogazione federata oltre ai connettori forniti. AWS Poiché i connettori funzionano AWS Lambda, non è necessario gestire l'infrastruttura o pianificare la scalabilità in base ai picchi di domanda.

Disponibilità dell'anteprima

Athena Federated Query è disponibile in anteprima nella Regione Stati Uniti orientali (Virginia settentrionale).

Fasi successive

- Per iniziare l'anteprima, segui le istruzioni riportate nelle Domande frequenti sulle funzionalità di Athena disponibili in anteprima.
- Per ulteriori informazioni sulla funzione delle query federate, consulta Utilizzo di Amazon Athena Federated Query (anteprima).
- Per iniziare a utilizzare un connettore esistente, consulta Distribuzione di un connettore e connessione a un'origine dati.
- Per informazioni su come creare un connettore di origine dati personalizzato utilizzando l'SDK Athena Query Federation, vedi Example Athena Connector on. GitHub

Richiamare modelli di machine learning nelle query SQL

Puoi richiamare i modelli di machine learning per l'inferenza direttamente dalle query Athena. La possibilità di utilizzare i modelli di machine learning nelle query SQL consente di completare attività complesse, quali il rilevamento delle anomalie, l'analisi della coorte dei clienti e le previsioni di vendita, con semplici operazioni come il richiamo di una funzione in una query SQL.

Modelli ML

Puoi utilizzare più di una dozzina di algoritmi di machine learning integrati forniti da Amazon SageMaker, addestrare i tuoi modelli o trovare e abbonarti a pacchetti modello Marketplace AWSe distribuirli su Amazon SageMaker Hosting Services. Non è richiesta alcuna configurazione aggiuntiva. Puoi richiamare questi modelli di ML nelle tue query SQL dalla console Athena, dalle API Athena e attraverso l'anteprima del driver JDBC di Athena.

Disponibilità dell'anteprima

La funzione ML di Athena è disponibile oggi in anteprima nella Regione Stati Uniti orientali (Virginia settentrionale).

Fasi successive

- Per iniziare l'anteprima, segui le istruzioni riportate nelle Domande frequenti sulle funzionalità di Athena disponibili in anteprima.
- Per ulteriori informazioni sulla funzionalità di machine learning, consulta Utilizzo di Machine Learning (ML) con Amazon Athena (anteprima).

Funzioni definite dall'utente (UDF) (anteprima)

Ora puoi scrivere funzioni scalari personalizzate e richiamale nelle query Athena. È possibile scrivere le UDF in Java utilizzando l'SDK Athena Query Federation. Quando una UDF viene utilizzata in una query SQL inviata ad Athena, viene richiamata ed eseguita in AWS Lambda. Le UDF possono essere utilizzate in entrambe le clausole `SELECT` e `FILTER` di una query SQL. Puoi richiamare più funzioni definite dall'utente nella stessa query.

Disponibilità dell'anteprima

La funzione Athena UDF è disponibile oggi in anteprima nella Regione Stati Uniti orientali (Virginia settentrionale).

Fasi successive

- Per iniziare l'anteprima, segui le istruzioni riportate nelle Domande frequenti sulle funzionalità di Athena disponibili in anteprima.
- Per ulteriori informazioni, vedere Esecuzione di query con funzioni definite dall'utente (anteprima).
- Per esempio implementazioni UDF, consulta Amazon Athena UDF Connector on. GitHub
- Per informazioni su come scrivere proprie funzioni utilizzando l'SDK Athena Query Federation, consulta Creazione e distribuzione di una funzione definita dall'utente utilizzando Lambda.

Utilizzo di metastore Hive Apache come metacatalogo con Amazon Athena (anteprima)

Ora è possibile connettere Athena a uno o più metastore Hive Apache in aggiunta a AWS Glue Data Catalog con Athena.

Connettore Metastore

Per connettersi a un metastore Hive con hosting autonomo, è necessario un connettore per metastore Athena Hive. Athena offre un connettore per l'implementazione di riferimento che è possibile utilizzare. Il connettore funziona come una AWS Lambda funzione nel tuo account. Per ulteriori informazioni, consulta Utilizzo del connettore Athena Data per metastore Hive esterno (anteprima).

Disponibilità dell'anteprima

La caratteristica Metastore Hive è disponibile in anteprima nella Regione Stati Uniti orientali (Virginia settentrionale).

Fasi successive

- Per iniziare l'anteprima, segui le istruzioni riportate nelle Domande frequenti sulle funzionalità di Athena disponibili in anteprima.
- Per ulteriori informazioni su questa funzionalità, consulta Utilizzo di Athena Data Connector per metastore Hive esterno (anteprima).

Nuovi parametri relativi alle query

Athena ora pubblica ulteriori parametri di query che possono aiutarti a comprendere le prestazioni di Amazon Athena. Athena pubblica i parametri relativi alle query su Amazon. CloudWatch In questa versione, Athena pubblica i seguenti parametri di query aggiuntivi:

- Tempo di pianificazione query: il tempo impiegato per pianificare la query. Include il tempo impiegato per recuperare le partizioni della tabella dall'origine dati;
- Tempo di permanenza in coda: il tempo in cui la query è rimasta in coda in attesa di risorse.
- Tempo di elaborazione del servizio – Il tempo impiegato per scrivere i risultati dopo che il motore di query ha concluso l'esecuzione.
- Tempo di esecuzione totale – Il tempo impiegato da Athena per eseguire la query.

Per utilizzare queste nuove metriche di query, puoi creare dashboard personalizzate, impostare allarmi e trigger sulle metriche o utilizzare dashboard precompilate CloudWatch direttamente dalla console Athena.

Fasi successive

Per ulteriori informazioni, consulta Monitoraggio delle interrogazioni di Athena con metriche. CloudWatch

12 novembre 2019

Data pubblicazione: 17/12/2019

Amazon Athena è ora disponibile nella Regione Medio Oriente (Bahrein).

8 Novembre 2019

Data pubblicazione: 17/12/2019

Amazon Athena è ora disponibile nella Regione Stati Uniti occidentali (California settentrionale) ed Europa (Parigi).

8 ottobre 2019

Data pubblicazione: 17/12/2019

Amazon Athena ora ti consente di connetterti direttamente ad Athena attraverso un endpoint VPC di interfaccia nel tuo cloud privato virtuale (VPC). Con questa funzione, è possibile inviare le query ad Athena nella massima sicurezza, senza dover richiedere un gateway Internet nel VPC.

Per creare un endpoint VPC di interfaccia per connettersi ad Athena, puoi usare o (). AWS Management Console AWS Command Line Interface AWS CLI Per informazioni sulla creazione di un endpoint di interfaccia, consulta Creazione di un endpoint di interfaccia.

Quando utilizzi un endpoint VPC di interfaccia, la comunicazione tra il tuo VPC e le API Athena è sicura e rimane all'interno della rete. AWS Questa funzione di Athena non prevede costi aggiuntivi. Si applicano le tariffe dell'endpoint VPC di interfaccia.

Per ulteriori informazioni su questa caratteristica, consulta Connessione ad Amazon Athena utilizzando un endpoint VPC di interfaccia.

19 settembre 2019

Data pubblicazione: 17/12/2019

Amazon Athena aggiunge il supporto per l'inserimento di nuovi dati in una tabella esistente utilizzando l'istruzione `INSERT INTO`. Puoi inserire nuove righe in una tabella di destinazione utilizzando un'istruzione di query `SELECT` che viene eseguita su una tabella di origine o in base a un set di valori forniti come parte dell'istruzione di query. I formati di dati supportati comprendono Avro, JSON, ORC, Parquet e file di testo.

Le istruzioni `INSERT INTO` possono anche aiutarti a semplificare il processo di ETL. Ad esempio, puoi utilizzare `INSERT INTO` in una singola query per selezionare i dati da una tabella di origine in formato JSON e trascriverli in una tabella di destinazione in formato Parquet.

Le istruzioni `INSERT INTO` vengono addebitate in base al numero di byte che vengono analizzati nella fase `SELECT`, in modo simile a come Athena addebita le query `SELECT`. Per ulteriori informazioni, consulta Prezzi di Amazon Athena.

Per ulteriori informazioni sull'utilizzo`INSERT INTO`, inclusi i formati supportati, SerDes ed esempi, vedere INSERT INTO nella Guida per l'utente di Athena.

12 settembre 2019

Data pubblicazione: 17/12/2019

Amazon Athena è ora disponibile nella Regione Asia Pacifico (Hong Kong).

16 agosto 2019

Data pubblicazione: 17/12/2019

Amazon Athena aggiunge il supporto per le query di dati nei bucket Amazon S3 con pagamento a carico del richiedente.

Quando un bucket Amazon S3 è configurato per il pagamento a carico del richiedente, il richiedente, non il proprietario del bucket, paga i costi della richiesta e del trasferimento dei dati Amazon S3. In Athena, gli amministratori del gruppo di lavoro possono ora configurare le impostazioni del gruppo di lavoro per consentire ai membri del gruppo di lavoro di eseguire query sui bucket S3 con pagamento a carico del richiedente.

Per informazioni su come configurare l'impostazione per il pagamento a carico del richiedente, consulta Crea un gruppo di lavoro nella Guida per l'utente di Amazon Athena. Per ulteriori informazioni sui bucket con Pagamento a carico del richiedente, consulta Bucket con Pagamento a carico del richiedente nella Guida per gli sviluppatori di Amazon Simple Storage Service.

9 agosto 2019

Data pubblicazione: 17/12/2019

Amazon Athena ora supporta l'applicazione delle policy AWS Lake Formation per il controllo granulare degli accessi a database, tabelle e colonne nuovi o esistenti definiti nel AWS Glue Data Catalog per i dati archiviati in Amazon S3.

Puoi utilizzare questa funzionalità nei seguenti paesi Regioni AWS: Stati Uniti orientali (Ohio), Stati Uniti orientali (Virginia settentrionale), Stati Uniti occidentali (Oregon), Asia Pacifico (Tokyo) ed Europa (Irlanda). Questa caratteristica non prevede costi aggiuntivi.

Per ulteriori informazioni sull'utilizzo di questa caratteristica, consulta Utilizzo di Athena per eseguire query sui dati registrati con AWS Lake Formation. Per ulteriori informazioni su AWS Lake Formation, consulta AWS Lake Formation.

26 giugno 2019

Amazon Athena è ora disponibile nella Regione Europa (Stoccolma). Per un elenco delle regioni e degli endpoint supportati, consultare Regioni AWS ed endpoint.

24 maggio 2019

Data pubblicazione: 24-05-2019

Amazon Athena è ora disponibile nelle regioni AWS GovCloud (Stati Uniti orientali) e AWS GovCloud (Stati Uniti occidentali). Per un elenco delle regioni e degli endpoint supportati, consultare Regioni AWS ed endpoint.

05 marzo 2019

Data pubblicazione: 05/03/2019

Amazon Athena ora è disponibile nella Regione Canada (Centrale). Per un elenco delle regioni e degli endpoint supportati, consultare Regioni AWS ed endpoint. Rilasciata la nuova versione del driver ODBC con il supporto per i gruppi di lavoro di Athena. Per ulteriori informazioni, consulta le note di rilascio del driver ODBC.

Per scaricare e la versione 1.0.5 del driver ODBC, consulta Connessione ad Amazon Athena con ODBC. Per informazioni su questa versione, consulta le note di rilascio del driver ODBC.

Per usare i gruppi di lavoro con il driver ODBC, imposta la nuova proprietà di connessione `Workgroup` nella stringa di connessione mostrata nell'esempio seguente:

```
Driver=Simba Athena ODBC
 Driver;AwsRegion=[Region];S3OutputLocation=[S3Path];AuthenticationType=IAM
 Credentials;UID=[YourAccessKey];PWD=[YourSecretKey];Workgroup=[WorkgroupName]
```

Per ulteriori informazioni, cerca la sezione dedicata ai gruppi di lavoro nella Guida all'installazione e alla configurazione del driver ODBC versione 1.0.5. Non ci sono variazioni alla stringa di connessione del driver ODBC quando si utilizzano i tag nei gruppi di lavoro. Per usare i tag, esegui l'aggiornamento alla versione più recente del driver ODBC, ovvero la versione corrente.

Questa versione del driver consente di usare le operazioni dei gruppi di lavoro dell'API Athena per creare e gestire i gruppi di lavoro e le operazioni dei tag dell'API Athena per aggiungere, elencare o rimuovere tag nei gruppi di lavoro. Prima di iniziare, verifica di disporre delle autorizzazioni a livello di risorsa in IAM per le operazioni su gruppi di lavoro e tag.

Per ulteriori informazioni, consulta:

- Utilizzo dei gruppi di lavoro per l'esecuzione di query e Esempi di policy per i gruppi di lavoro.
- Assegnazione di tag alle risorse Athena e Policy di controllo degli accessi IAM basate su tag.

Se utilizzi il driver JDBC o l' AWS SDK, esegui l'upgrade alla versione più recente del driver e dell'SDK, entrambi i quali includono già il supporto per gruppi di lavoro e tag in Athena. Per ulteriori informazioni, consulta Connessione ad Amazon Athena con JDBC.

22 febbraio 2019

Data pubblicazione: 22/02/2019

Aggiunto il supporto dei tag per i gruppi di lavoro in Amazon Athena. Un tag consiste di una chiave e di un valore che definisci. Quando aggiungi tag a un gruppo di lavoro, gli assegni metadati personalizzati. Puoi aggiungere tag ai gruppi di lavoro per aiutarli a classificarli, utilizzando le migliori pratiche di etichettatura. AWS Puoi usare i tag per limitare l'accesso ai gruppi di lavoro e per tenere traccia dei costi. Ad esempio, crea un gruppo di lavoro per ogni centro di costo. Quindi, aggiungendo tag a questi gruppi di lavoro, puoi tenere traccia della spesa di Athena per ogni centro di costo. Per ulteriori informazioni, consulta la sezione relativa all'utilizzo dei tag per la fatturazione nella Guida per l'utente di AWS Billing and Cost Management .

Puoi lavorare con i tag utilizzando la console Athena o le operazioni API. Per ulteriori informazioni, consulta Assegnazione di tag alle risorse Athena.

Nella console Athena puoi aggiungere uno o più tag a ciascun gruppo di lavoro ed eseguire ricerche in base ai tag. I gruppi di lavoro sono una risorsa controllata da IAM in Athena. In IAM puoi limitare gli utenti che possono aggiungere, rimuovere o elencare i tag nei gruppi di lavoro che hai creato. Puoi anche usare l'operazione API `CreateWorkGroup` con il parametro tag opzionale per aggiungere uno o più tag al gruppo di lavoro. Per aggiungere, rimuovere o elencare i tag, usa `TagResource`, `UntagResource` e `ListTagsForResource`. Per ulteriori informazioni, consulta Utilizzo delle operazioni relative ai tag.

Per consentire agli utenti di aggiungere tag durante la creazione di gruppi di lavoro, assicurati di fornire a ciascun utente le autorizzazioni IAM per le operazioni dell'API `TagResource` e `CreateWorkGroup`. Per maggiori informazioni ed esempi, consulta Policy di controllo degli accessi IAM basate su tag.

Non ci sono variazioni al driver JDBC quando si utilizzano i tag nei gruppi di lavoro. Se crei nuovi gruppi di lavoro e utilizzi il driver JDBC o l' AWS SDK, esegui l'aggiornamento alla versione più recente del driver e dell'SDK. Per informazioni, consulta Connessione ad Amazon Athena con JDBC.

18 febbraio 2019

Data pubblicazione: 18/02/2019

Aggiunta la possibilità di controllare i costi delle query tramite l'esecuzione di query nei gruppi di lavoro. Per informazioni, consulta Uso dei gruppi di lavoro per controllare l'accesso alle query e i costi. È stato migliorato il formato JSON SerDe OpenX utilizzato in Athena, risolto un problema per cui Athena non ignorava gli oggetti trasferiti alla classe di archiviazione e aggiunto esempi per `GLACIER` l'interrogazione dei log di Network Load Balancer.

Sono state apportate le modifiche seguenti:

- Aggiunto il supporto per i gruppi di lavoro. Usa i gruppi di lavoro per separare utenti, team, applicazioni o carichi di lavoro e per impostare i limiti relativi alla quantità di dati che ogni query o l'intero gruppo di lavoro può elaborare. Poiché i gruppi di lavoro fungono da risorse IAM, puoi utilizzare le autorizzazioni a livello di risorsa per controllare l'accesso a un determinato gruppo di lavoro. Puoi anche visualizzare i parametri relativi alle query in Amazon CloudWatch, controllare i costi delle query configurando limiti alla quantità di dati scansionati, creare soglie e attivare azioni, come gli allarmi di Amazon SNS, quando queste soglie vengono violate. Per ulteriori informazioni,

consulta Utilizzo dei gruppi di lavoro per l'esecuzione di query e Controllo dei costi e monitoraggio delle interrogazioni con metriche ed eventi CloudWatch .

I gruppi di lavoro sono una risorsa IAM. Per un elenco completo delle azioni, delle risorse e delle condizioni correlate ai gruppi di lavoro in IAM, consulta Azioni, risorse e chiavi di condizione per Amazon Athena nella Documentazione di riferimento sulle autorizzazioni per i servizi. Prima di creare nuovi gruppi di lavoro, assicurati di usare le policy IAM per i gruppi di lavoro e la AWS politica gestita: AmazonAthenaFullAccess.

Puoi iniziare a utilizzare i gruppi di lavoro nella console con le operazioni API per i gruppi di lavoro o con il driver JDBC. Per una procedura di alto livello, consulta la sezione Configurazione dei gruppi di lavoro. Per scaricare il driver JDBC con il supporto per i gruppi di lavoro, consulta Connessione ad Amazon Athena con JDBC.

Se usi i gruppi di lavoro con il driver JDBC, è devi impostare il nome del gruppo di lavoro nella stringa di connessione utilizzando il parametro di configurazione `Workgroup` come nell'esempio seguente:

```
jdbc:awsathena://AwsRegion=<AWSREGION>;UID=<ACCESSKEY>;
PWD=<SECRETKEY>;S3OutputLocation=s3://DOC-EXAMPLE-BUCKET/<athena-
output>-<AWSREGION>/;
Workgroup=<WORKGROUPNAME>;
```

Non ci sono variazioni nell'esecuzione delle istruzioni SQL o delle chiamate API JDBC al driver. Il driver passa il nome del gruppo di lavoro ad Athena.

Per informazioni sulle differenze introdotte con i gruppi di lavoro, consulta le sezioni API per gruppi di lavoro Athena e Risoluzione dei problemi relativi ai gruppi di lavoro.

- Migliorato il formato JSON SerDe OpenX utilizzato in Athena. I miglioramenti includono, tra l'altro, quanto segue:
  - Supporto della proprietà `ConvertDotsInJsonKeysToUnderscores`. Se impostato su`TRUE`, consente di sostituire i punti nei SerDe nomi delle chiavi con caratteri di sottolineatura. Ad esempio, se il set di dati JSON contiene una chiave denominata `"a.b"`, è possibile usare questa proprietà per definire il nome della colonna come `"a_b"` in Athena. Il valore predefinito è `FALSE`. Per impostazione predefinita, Athena non consente l'uso di punti nei nomi di colonna.
  - Supporto della proprietà `case.insensitive`. Per impostazione predefinita, Athena richiede che tutte le chiavi nel set di dati JSON siano in caratteri minuscoli. `WITH SERDE PROPERTIES ("case.insensitive"= FALSE;)` consente di usare nomi di chiave con distinzione tra

maiuscole e minuscole nei dati. Il valore predefinito è `TRUE`. Se impostato su`TRUE`, SerDe converte tutte le colonne maiuscole in minuscole.

Per ulteriori informazioni, consulta OpenX JSON SerDe.

- Risolto un problema per cui Athena restituiva messaggi di errore `"access denied"` quando elaborava oggetti Amazon S3 archiviati in Glacier tramite policy del ciclo di vita Amazon S3. Come risultato della risoluzione di questo problema, Athena ignora gli oggetti trasmessi alla classe di archiviazione `GLACIER`. Athena non supporta l'esecuzione di query sui dati nella classe di archiviazione `GLACIER`.

  Per ulteriori informazioni, consultare the section called “Requisiti per tabelle in Athena e dati in Amazon S3” e Transizione alla classe di archiviazione GLACIER (archiviazione di oggetti) nella Guida per l'utente di Amazon Simple Storage Service.
- Aggiunti esempi per l'esecuzione di query sui log di accesso di Network Load Balancer che ricevono informazioni sulle richieste Transport Layer Security (TLS). Per ulteriori informazioni, consulta the section called “Network Load Balancer”.

Note di rilascio di Athena per il 2018

20 novembre 2018

Data pubblicazione: 20/11/2018

Rilasciate le nuove versioni dei driver JDBC e ODBC con supporto per l'accesso federato alle API di Athena con AD FS e SAML 2.0 (Security Assertion Markup Language 2.0). Per ulteriori informazioni, consulta le note di rilascio del driver JDBC e note di rilascio del driver ODBC.

Questa versione supporta l'accesso federato ad Athena per Active Directory Federation Service (AD FS 3.0). L'accesso viene stabilito attraverso le versioni dei driver JDBC o ODBC che supportano SAML 2.0. Per ulteriori informazioni sulla configurazione dell'accesso federato all'API Athena, consulta the section called “Abilitazione dell'accesso federato all'API Athena”.

Per scaricare e la versione 2.0.6 del driver JDBC, consulta Connessione ad Amazon Athena con JDBC. Per informazioni su questa versione, consulta le note di rilascio del driver JDBC.

Per scaricare e la versione 1.0.4 del driver ODBC, consulta Connessione ad Amazon Athena con ODBC. Per informazioni su questa versione, consulta le note di rilascio del driver ODBC.

Per ulteriori informazioni sul supporto SAML 2.0 in AWS, consulta About SAML 2.0 Federation nella IAM User Guide.

15 ottobre 2018

Data pubblicazione: 15/10/2018

Se hai effettuato l'aggiornamento a AWS Glue Data Catalog, sono disponibili due nuove funzionalità che forniscono supporto per:

- Crittografia dei metadati del catalogo dati. Se si decide di crittografare i metadati nel catalogo dati, è necessario aggiungere policy specifiche ad Athena. Per ulteriori informazioni, consulta la sezione relativa all'accesso ai metadati crittografati nel AWS Glue Data Catalog.
- Autorizzazioni granulari per accedere alle risorse in. AWS Glue Data CatalogÈ ora possibile definire le policy basate sull'identità (IAM) che limitano o consentono l'accesso a database specifici e alle tabelle dal catalogo dati utilizzato in Athena. Per ulteriori informazioni, consulta Accesso granulare a database e tabelle in AWS Glue Data Catalog.

> **Note**
>
> I dati risiedono nei bucket Amazon S3 e l'accesso a tali dati è controllato dalle Accesso ad Amazon S3 da Athena. Per accedere ai dati di database e di tabelle, continua a utilizzare le policy di controllo di accesso ai bucket Amazon S3 che archiviano i dati.

10 ottobre 2018

Data pubblicazione: 10/10/2018

Athena supporta `CREATE TABLE AS SELECT`, che crea una tabella dal risultato di un'istruzione di query `SELECT`. Per ulteriori informazioni, consulta la sezione relativa alla creazione di una tabella da risultati delle query (CTAS).

Prima di creare query CTAS, è importante apprenderne il comportamento nella documentazione di Athena. che contiene informazioni sul percorso in cui salvare i risultati delle query in Amazon S3, l'elenco dei formati supportati per l'archiviazione dei risultati delle query CTAS, il numero di partizioni che è possibile creare e i formati di compressione supportati. Per ulteriori informazioni, consulta Considerazioni e restrizioni per le query CTAS.

Utilizza le query CTAS per:

- Creare una tabella da risultati di query in un'unica operazione.
- Creare query CTAS nella console di Athena utilizzando degli esempi. Per ulteriori informazioni sulla sintassi, consulta CREATE TABLE AS.
- Trasformare i risultati delle query in altri formati di storage, come Parquet, ORC, AVRO, JSON e TEXTFILE. Per ulteriori informazioni, consulta Considerazioni e restrizioni per le query CTAS e Formati di archiviazione colonnare.

6 settembre 2018

Data pubblicazione: 06/09/2018

Rilasciata la nuova versione del driver ODBC (versione 1.0.3). Per impostazione predefinita, la nuova versione del driver ODBC trasmette i risultati in forma di flussi anziché di paginarli, consentendo agli strumenti di business intelligence di recuperare più rapidamente grandi set di dati. Questa versione include inoltre miglioramenti, correzioni di bug e la documentazione aggiornata per "Utilizzo di SSL con un server proxy". Per ulteriori informazioni, consulta le note di rilascio del driver.

Per scaricare la versione 1.0.3 del driver ODBC e la relativa documentazione, consulta Connessione ad Amazon Athena con ODBC.

La caratteristica dei risultati di streaming è disponibile con questa nuova versione del driver ODBC. È disponibile anche con il driver JDBC. Per informazioni sui risultati di streaming, consulta la Guida all'installazione e alla configurazione dei driver ODBC e cerca. UseResultsetStreaming

La versione 1.0.3 del driver ODBC è una sostituzione drop-in della versione precedente del driver. È consigliabile migrare al driver corrente.

> **Important**
>
> Per utilizzare la versione 1.0.3 del driver ODBC, segui queste istruzioni:
>
> - Mantieni la porta 444 aperta al traffico in uscita.
> - Aggiungi l'operazione di policy `athena:GetQueryResultsStream` all'elenco di policy per Athena. Questa operazione di policy non è esposta direttamente con le API e viene utilizzata solo con i driver JDBC e ODBC, come parte del supporto per i risultati di streaming. Per un esempio di policy, consulta AWS politica gestita: AWSQuicksightAthenaAccess.

23 agosto 2018

Data pubblicazione: 23/08/2018

Aggiunto il supporto per queste caratteristiche del DDL e corretti vari bug, come segue:

- Aggiunto il supporto per i tipi di dati `DATE` e `BINARY`, per i dati in Parquet, per i tipi di dati `TIMESTAMP` e `DATE`, nonché per i dati in Avro.
- Aggiunto il supporto per `INT` e `DOUBLE` nelle query DDL. `INTEGER` è un alias di `INT` ed `DOUBLE PRECISION` è un alias di `DOUBLE`.
- Migliorate le prestazioni delle query `DROP TABLE` e `DROP DATABASE`.
- Rimossa la creazione dell'oggetto `_$folder$` in Amazon S3 quando un bucket di dati è vuoto.
- Risolto un problema in cui `ALTER TABLE ADD PARTITION` generava un errore quando non veniva fornito alcun valore di partizione.
- Risolto un problema in cui `DROP TABLE` ignorava il nome del database durante il controllo delle partizioni dopo il nome qualificato era stato specificato nell'istruzione.

Per ulteriori informazioni sui tipi di dati supportati in Athena, consulta le sezioni Tipi di dati in Amazon Athena.

Per informazioni sulle mappature dei tipi di dati supportati tra i tipi in Athena, il driver JDBC e i tipi di dati Java, consulta la sezione "Tipi di dati" nella Guida per l'installazione e la configurazione del driver JDBC.

16 agosto 2018

Data pubblicazione: 16/08/2018

Rilasciata la versione 2.0.5 del driver JDBC. Per impostazione predefinita, la nuova versione del driver JDBC trasmette i risultati in forma di flussi anziché di paginarli, consentendo agli strumenti di business intelligence di recuperare più rapidamente grandi set di dati. Rispetto alla versione precedente del driver JDBC, sono stati apportati i seguenti miglioramenti delle prestazioni:

- Aumento a circa il doppio delle prestazioni in caso di recupero di meno di 10.000 righe.
- Aumento delle prestazioni di circa 5-6 volte in caso di recupero di più di 10.000 righe.

La caratteristica dei risultati di streaming è disponibile solo con driver JDBC. Non è disponibile con il driver ODBC. Non è possibile utilizzarla con le API Athena. Per informazioni sui risultati di streaming, consulta la Guida all'installazione e alla configurazione dei driver JDBC e cerca. UseResultsetStreaming

Per scaricare la versione 2.0.5 del driver JDBC e la relativa documentazione, consulta Connessione ad Amazon Athena con JDBC.

La versione 2.0.5 del driver JDBC è una sostituzione drop-in della versione precedente del driver (2.0.2). Per assicurarsi di poter utilizzare la versione 2.0.5 del driver JDBC, occorre aggiungere l'operazione di policy `athena:GetQueryResultsStream` all'elenco di policy per Athena. Questa operazione di policy non è esposta direttamente con le API e viene utilizzata solo con il driver JDBC come parte del supporto per i risultati di streaming. Per un esempio di policy, consulta AWS politica gestita: AWSQuicksightAthenaAccess. Per ulteriori informazioni sulla migrazione dalla versione 2.0.2 alla versione 2.0.5 del driver, consulta la guida alla migrazione del driver JDBC.

Se stai eseguendo la migrazione da una versione 1.x a una 2.x del driver, dovrai eseguire la migrazione delle configurazioni esistenti alla nuova configurazione. È altamente consigliabile migrare alla versione corrente del driver. Per ulteriori informazioni, consulta la Guida alla migrazione del driver JDBC.

7 agosto 2018

Data pubblicazione: 07/08/2018

È ora possibile archiviare i flussi di log Amazon Virtual Private Cloud in formato GZIP direttamente in Amazon S3, dove è possibile eseguire query su tali log in Athena. Per ulteriori informazioni, consulta Esecuzione di query sui log di flusso Amazon VPC e la sezione Log di flusso di Amazon VPC ora inoltrabili a S3.

5 giugno 2018

Data pubblicazione: 05/06/2018

Argomenti

- Supporto per le visualizzazioni
- Miglioramenti e aggiornamenti per i messaggi di errore
- Correzioni di bug

Supporto per le visualizzazioni

Aggiunta del supporto per le visualizzazioni. È ora possibile utilizzare CREATE VIEW, DESCRIBE VIEW, DROP VIEW, SHOW CREATE VIEW e SHOW VIEWS in Athena. La query che definisce la visualizzazione viene eseguita ogni volta che si fa riferimento alla visualizzazione nella query. Per ulteriori informazioni, consulta Utilizzo delle visualizzazioni.

Miglioramenti e aggiornamenti per i messaggi di errore

- È stata inclusa una libreria GSON 2.8.0 in CloudTrail SerDe, per risolvere un problema con le stringhe JSON CloudTrail SerDe e abilitare l'analisi delle stringhe JSON.
- Miglioramento della convalida dello schema di partizione in Athena per Parquet e, in alcuni casi, per ORC, consentendo il riordinamento delle colonne. Ciò consente ad Athena di gestire meglio i cambiamenti nell'evoluzione dello schema nel tempo e le tabelle aggiunte dal AWS Glue Crawler. Per ulteriori informazioni, consulta Gestione degli aggiornamenti degli schemi.
- Aggiunto supporto per analisi per `SHOW VIEWS`.
- Realizzati i seguenti miglioramenti ai messaggi di errore più comuni:
  - Ha sostituito un messaggio di errore interno con un messaggio di errore descrittivo quando SerDe non riesce ad analizzare la colonna in una query Athena. In passato, Athena restituiva un errore interno in caso di errori di analisi. Il nuovo messaggio di errore è: "HIVE_BAD_DATA: errore durante l'analisi del valore di campo per il campo 0: java.lang.String non può essere trasmesso a org.openx.data.jsonserde.json.JSONObject".
  - Migliorati i messaggi di errore relativi ad autorizzazioni insufficienti grazie all'aggiunta ulteriori dettagli.

Correzioni di bug

Sono stati corretti i seguenti bug:

- Corretto un problema consentendo la traduzione di `REAL` in di tipi di dati `FLOAT`. Ciò consente di migliorare l'integrazione con il crawler di AWS Glue che restituisce tipi di dati `FLOAT`.
- Corretto un problema per cui Athena non convertiva AVRO `DECIMAL` (un tipo logico) in a un tipo `DECIMAL`.
- Risolto un problema per cui Athena non restituiva risultati per le query su dati Parquet con clausole `WHERE` che facevano riferimento a valori nel tipo di dati `TIMESTAMP`.

17 maggio 2018

Data pubblicazione: 17/05/2018

Aumentate le quote predefinite per le query simultanee da cinque a venti in Athena. È quindi possibile inviare ed eseguire fino a venti query `DDL` e venti query `SELECT` alla volta. Si noti che le quote di simultaneità sono separate per le query `DDL` e `SELECT`.

Le quote di simultaneità in Athena sono definite come il numero di query simultanee che possono essere inviate al servizio. È possibile inviare fino a venti query dello stesso tipo (`DDL` o `SELECT`) alla volta. Se si invia una query che supera la quota delle query simultanee, l'API Athena visualizza un messaggio di errore.

Dopo aver inviato le query ad Athena, Athena le elabora assegnando le risorse in base al carico di servizio globale e alla quantità di richieste in entrata. Monitoriamo e adattiamo il servizio costantemente in modo che le query siano elaborate più rapidamente possibile.

Per informazioni, consulta Service Quotas (Quote di Servizio). Si tratta di una quota regolabile. È possibile utilizzare la Console Service Quotas per richiedere un aumento della quota per le query simultanee.

19 aprile 2018

Data pubblicazione: 19/04/2018

Pubblicata la nuova versione del driver JDBC (versione 2.0.2) con il supporto per restituire i dati `ResultSet` come tipo di dati matrice, miglioramenti e correzioni di bug. Per ulteriori informazioni, consulta le note di rilascio del driver.

Per informazioni su come scaricare la nuova versione 2.0.2 del driver JDBC e la relativa documentazione, consulta Connessione ad Amazon Athena con JDBC.

La versione più recente del driver JDBC è la 2.0.2. Se stai eseguendo la migrazione da una versione 1.x a una 2.x del driver, dovrai eseguire la migrazione delle configurazioni esistenti alla nuova configurazione. È altamente consigliabile migrare al driver corrente.

Per informazioni sulle modifiche apportate nella nuova versione del driver, le differenze tra versioni e alcuni esempi, consulta la guida alla migrazione del driver JDBC.

6 aprile 2018

Data pubblicazione: 06/04/2018

Utilizza la funzione di completamento automatico per digitare query nella console Athena.

15 marzo 2018

Data pubblicazione: 15/03/2018

È stata aggiunta la possibilità di creare automaticamente tabelle Athena per i file di CloudTrail registro direttamente dalla CloudTrail console. Per informazioni, consulta Utilizzo della CloudTrail console per creare una tabella Athena per i log CloudTrail .

2 febbraio 2018

Data pubblicazione: 12/02/2018

Aggiunta del supporto per l'offloading sicuro dei dati intermedi su disco per le query con elevati requisiti di memoria che utilizzano la clausola `GROUP BY`. Questo migliora l'affidabilità di tali query, impedendo gli errori di tipo "Query resource exhausted".

19 gennaio 2018

Data pubblicazione: 19/01/2018

Athena utilizza Presto, un motore di query open source, per l'esecuzione di query.

Con Athena, non ci sono versioni da gestire. Abbiamo eseguito in modo chiaro l'aggiornamento del motore sottostante in Athena a una versione basata sulla versione 0.172 di Presto. Non è richiesta nessuna azione da parte tua.

Con l'aggiornamento, è ora possibile utilizzare funzioni e operatori Presto 0.172, tra cui le espressioni lambda di Presto 0.172 in Athena.

I principali aggiornamenti di questa versione, tra cui le correzioni condivise dalla community, includono:

- Supporto per ignorare le intestazioni. È possibile utilizzare la proprietà `skip.header.line.count` durante la definizione di tabelle per consentire ad Athena di ignorare le intestazioni. Questo è supportato per le query che utilizzano SerDeOpenCSV LazySimpleSerDee non per Grok o Regex. SerDes
- Supporto per il tipo di dati `CHAR(n)` nelle funzioni `STRING`. L'intervallo per `CHAR(n)` è `[1.255]`, mentre l'intervallo per `VARCHAR(n)` è `[1,65535]`.

- Supporto per sottoquery correlate.
- Supporto per espressioni e funzioni Lambda di Presto.
- Migliorate le prestazioni del tipo `DECIMAL` e degli operatori.
- Supporto per aggregazioni filtrate, ad esempio `SELECT sum(col_name) FILTER`, dove `id > 0`.
- Predicati push-down per i tipi di dati `DECIMAL`, `TINYINT`, `SMALLINT` e `REAL`.
- Supporto per un predicati di confronto quantificato: `ALL`, `ANY` e `SOME`.
- Aggiunte funzioni: `arrays_overlap()`, `array_except()`, `levenshtein_distance()`, `codepoint()`, `skewness()`, `kurtosis()` e `typeof()`.
- Aggiunta una variante della funzione `from_unixtime()` che richiede un argomento timezone.
- Aggiunte le funzioni di aggregazione `bitwise_and_agg()` e `bitwise_or_agg()`.
- Aggiunte le funzioni `xxhash64()` e `to_big_endian_64()`.
- Aggiunto il supporto per usare doppie virgolette o barre rovesciate come caratteri di escape utilizzando una barra rovesciata con un subscript di percorso JSON che rimanda alle funzioni `json_extract()` e `json_extract_scalar()`. Questo modifica la semantica di qualsiasi invocazione utilizzando una barra rovesciata, poiché le barre rovesciate erano precedentemente trattate come normali caratteri.

Per un elenco completo di funzioni e operatori, consulta la sezione Query, funzioni e operatori DML in questa guida e la pagina Functions and operators (Funzioni e operatori) nella documentazione di Presto.

Athena non supporta tutte le funzioni di Presto. Per ulteriori informazioni, consulta Limitazioni.

Note di rilascio di Athena per il 2017

13 Novembre 2017

Data pubblicazione: 13/11/2017

Aggiunto supporto per collegare Athena al driver ODBC. Per informazioni, consulta Connessione ad Amazon Athena con ODBC.

1 Novembre 2017

Data pubblicazione: 01/11/2017

Aggiunto supporto per le query su dati geospaziali e per le regioni Asia Pacifico (Seoul), Asia Pacifico (Mumbai) e UE (Londra). Per ulteriori informazioni, consulta Esecuzione di query su dati geospaziali e Regioni AWS ed endpoint.

19 ottobre 2017

Data pubblicazione: 19/10/2017

Aggiunto il supporto per UE (Francoforte). Per un elenco delle regioni supportate, consultare Regioni AWS ed endpoint.

3 ottobre 2017

Data pubblicazione: 03/10/2017

Crea query denominate Athena con. AWS CloudFormation Per ulteriori informazioni, consulta AWS::Athena::NamedQueryla Guida per l'AWS CloudFormation utente.

25 settembre 2017

Data pubblicazione: 25/09/2017

Aggiunta del supporto per la Regione Asia Pacifico (Sydney). Per un elenco delle regioni supportate, consultare Regioni AWS ed endpoint.

14 agosto 2017

Data pubblicazione: 14/08/2017

È stata aggiunta l'integrazione con AWS Glue Data Catalog e una procedura guidata di migrazione per l'aggiornamento dal catalogo di dati gestito da Athena a. AWS Glue Data Catalog Per ulteriori informazioni, consulta Integrazione con AWS Glue.

4 agosto 2017

Data pubblicazione: 04/08/2017

È stato aggiunto il supporto per Grok SerDe, che semplifica la corrispondenza dei pattern per i record in file di testo non strutturati come i log. Per ulteriori informazioni, consulta - Grok SerDe. Aggiunta di

scelte rapide da tastiera per scorrere la cronologia delle query utilizzando la console (CTRL+⇧/⇩ su Windows, CMD+⇧/⇩ su Mac).

22 giugno 2017

Data pubblicazione: 22/06/2017

Aggiunto il supporto per le Regioni Asia Pacifico (Tokyo) e Asia Pacifico (Singapore). Per un elenco delle regioni supportate, consultare Regioni AWS ed endpoint.

8 giugno 2017

Data pubblicazione: 08/06/2017

Aggiunto il supporto per la Regione Europa (Irlanda). Per ulteriori informazioni, consulta Regioni AWS ed endpoint.

19 maggio 2017

Data pubblicazione: 19/05/2017

Aggiunti un'API Amazon Athena e AWS CLI supporto per Athena; driver JDBC aggiornato alla versione 1.1.0; risolti vari problemi.

- Amazon Athena consente la programmazione di applicazioni per Athena. Per ulteriori informazioni, consulta la documentazione di riferimento dell'API di Amazon Athena. Gli AWS SDK più recenti includono il supporto per l'API Athena. Per collegamenti alla documentazione e ai download, consulta la sezione relativa agli SDK in Strumenti per Amazon Web Services.
- AWS CLI Include nuovi comandi per Athena. Per ulteriori informazioni, consulta la documentazione di riferimento dell'API di Amazon Athena.
- È disponibile un nuovo driver JDBC 1.1.0 che supporta le nuove API Athena, nonché le ultime funzionalità e correzioni. Scarica il driver all'indirizzo https://downloads.athena.us-east-1.amazonaws.com/drivers/AthenaJDBC41-1.1.0.jar. Si consiglia di eseguire l'aggiornamento al driver JDBC Athena più recente; tuttavia potrebbe essere ancora possibile utilizzare la versione precedente del driver. Le versioni precedenti non supportano l'API di Athena. Per ulteriori informazioni, consulta Connessione ad Amazon Athena con JDBC.
- Le operazioni specifiche delle istruzioni policy nelle versioni precedenti di Athena sono obsolete. Se si esegue l'aggiornamento alla versione 1.1.0 del driver JDBC e si hanno delle policy IAM gestite dal cliente o inline associate a utenti JDBC, è necessario aggiornare le policy IAM. Al contrario,

le versioni precedenti dei driver JDBC non supportano l'API Athena, perciò è possibile specificare solo operazioni obsolete nelle policy associate a utenti di versioni precedenti di JDBC. Per questo motivo, non è necessario aggiornare policy IAM gestite dal cliente o inline.

- Queste operazioni specifiche di policy erano utilizzate in Athena prima del rilascio dell'API di Athena. Utilizza queste operazioni obsolete nelle policy solo con i driver JDBC precedenti alla versione 1.1.0. Se si ha intenzione di aggiornare il driver JDBC, occorre sostituire le istruzioni di policy che consentono o impediscono operazioni obsolete con le opportune operazioni API come indicato nell'elenco, altrimenti si verificheranno degli errori:

Operazione obsoleta specifica della policy	Operazione API Athena corrispondente
`athena:RunQuery`	`athena:StartQueryExecution`
`athena:CancelQueryExecution`	`athena:StopQueryExecution`
`athena:GetQueryExecutions`	`athena:ListQueryExecutions`

Miglioramenti

- Aumentato il limite di lunghezza della stringa di query a 256 KB.

Correzioni di bug

- Corretto un problema che facevano sembrare i risultati delle query come malformati quando si scorrevano i risultati nella console.
- Risolto un problema in cui una stringa di caratteri `\u0000` nei file di dati Amazon S3 era solita causare errori.
- Corretto un problema che faceva fallire le richieste di eliminazione di una query effettuata attraverso il driver JDBC.
- È stato risolto un problema che causava il AWS CloudTrail SerDe fallimento dei dati di Amazon S3 negli Stati Uniti orientali (Ohio).
- Corretto un problema per cui `DROP TABLE` aveva esito negativo su una tabella partizionata.

4 Aprile 2017

Data pubblicazione: 04/04/2017

Aggiunto il supporto per la crittografia dei dati Amazon S3 e rilasciato un aggiornamento del driver JDBC (versione 1.0.1) con supporto per la crittografia, miglioramenti e correzioni di bug.

Funzionalità

- Aggiunte le seguenti caratteristiche di crittografia:
  - Supporto per l'esecuzione di query sui dati crittografati in Amazon S3.
  - Supporto per la crittografia dei risultati delle query di Athena.
- Una nuova versione del driver supporta nuove caratteristiche di crittografia, aggiunge miglioramenti e correzioni problemi.
- Aggiunta la possibilità di aggiungere, sostituire e modificare colonne utilizzando `ALTER TABLE`. Per ulteriori informazioni, consulta la sezione relativa alla funzione ALTER COLUMN nella documentazione di Hive.
- Aggiunto il supporto per eseguire query sui dati compressi in LZO.

Per ulteriori informazioni, consulta Crittografia a riposo.

Miglioramenti

- Migliori prestazioni di query sul driver JDBC grazie a miglioramenti di dimensioni pagina; vengono ora restituite 1.000 righe anziché 100.
- Aggiunta la possibilità di annullare una query utilizzando l'interfaccia del driver JDBC.
- Aggiunta la possibilità di specificare le opzioni di JDBC nell'URL di connessione JDBC. Consulta Connessione ad Amazon Athena con JDBC per il driver JDBC più recente.
- È stata aggiunta l'impostazione PROXY nel driver, che ora può essere impostata utilizzando ClientConfigurationl' AWS SDK for Java.

Correzioni di bug

Sono stati corretti i seguenti bug:

- Quando venivano inviate più query utilizzando l'interfaccia del driver JDBC, si verificavano errori di throttling.

- Il driver JDBC termina l'operazione quando proietta un tipo di dati decimale.
- Il driver JDBC restituiva tutti i tipi di dati come una stringa, indipendentemente dal modo in cui il tipo di dati era definito nella tabella. Ad esempio, selezionando una colonna definita come tipo di dati `INT` utilizzando `resultSet.GetObject()`, restituiva un tipo di `STRING` anziché `INT`.
- Il driver JDBC verificava le credenziali nel momento in cui veniva stabilita la connessione piuttosto che durante l'esecuzione di una query.
- Le query effettuate attraverso il driver JDBC fallivano quando un schema veniva specificato assieme all'URL.

24 marzo 2017

Data pubblicazione: 24/03/2017

Aggiunto il AWS CloudTrail SerDe, prestazioni migliorate, problemi di partizione risolti.

Funzionalità

- È stato aggiunto il AWS CloudTrail SerDe, che da allora è stato sostituito dal registro JSON Hive SerDe per la lettura. CloudTrail Per informazioni sull' CloudTrail interrogazione dei log, vedere. Interrogazione dei log AWS CloudTrail

Miglioramenti

- Miglioramento delle prestazioni durante la scansione di un numero elevato di partizioni.
- Miglioramento delle prestazioni nell'operazione `MSCK Repair Table`.
- Aggiunta la possibilità di eseguire query sui dati Amazon S3 archiviati in Regioni diverse da quella primaria. In aggiunta alle tariffe standard di Athena, si applicano i costi di trasferimento dei dati interregionali di per Amazon S3.

Correzioni di bug

- Corretto un bug che poteva generare un errore "tabella non trovata" se le partizioni non venivano caricate.
- Corretto un bug per evitare la generazione di un'eccezione con le query `ALTER TABLE ADD PARTITION IF NOT EXISTS`.

- Corretto un bug in `DROP PARTITIONS`.

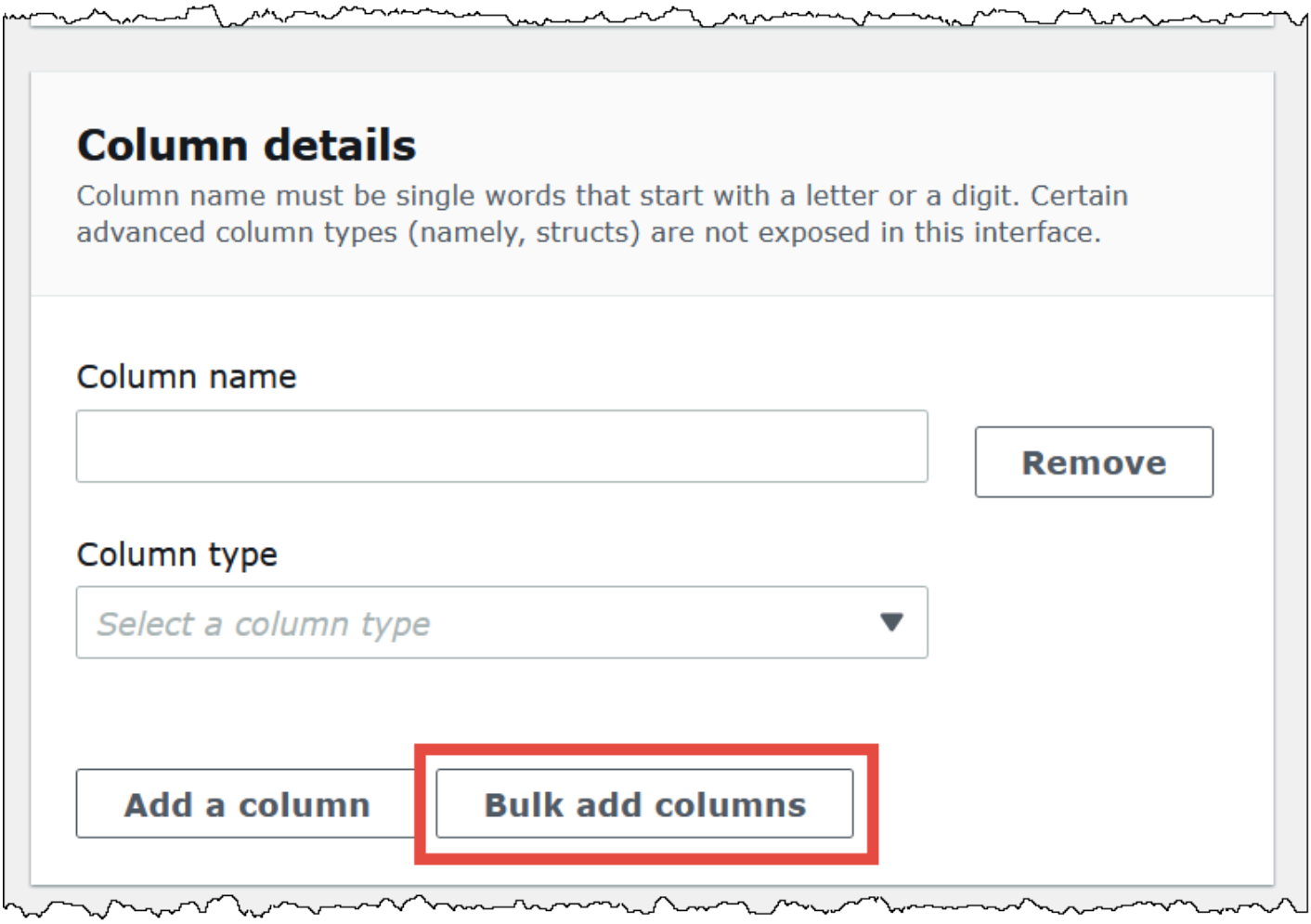
20 febbraio 2017

Data pubblicazione: 20/02/2017

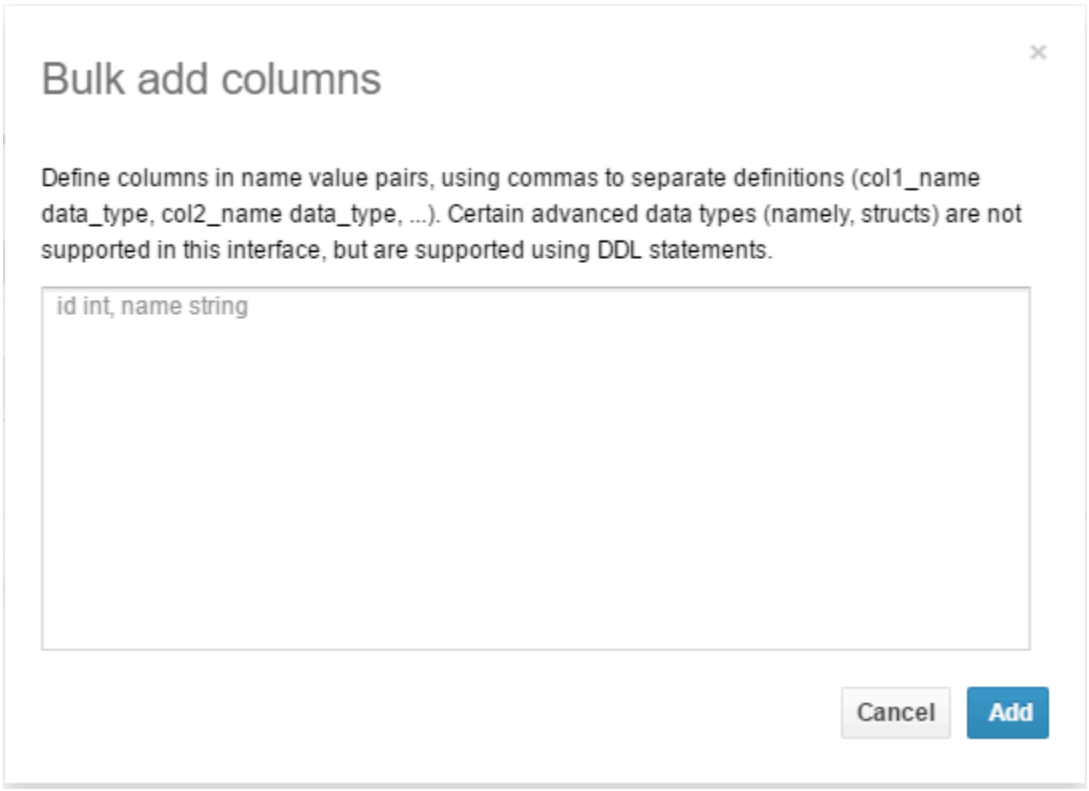
È stato aggiunto il supporto per AvroSerDe SerDe OpenCSV, la regione degli Stati Uniti orientali (Ohio) e la modifica in blocco delle colonne nella procedura guidata della console. Miglioramento delle prestazioni su tabelle Parquet di grandi dimensioni.

Funzionalità

- È stato introdotto il supporto per i nuovi: SerDes
  - Avro SerDe
  - OpenCSV per l'elaborazione di file SerDe CSV
- Lancio nella Regione Stati Uniti orientali (Ohio) (us-east-2). È ora possibile eseguire query in questa regione.
- Ora puoi utilizzare il modulo Create Table From S3 bucket data (Crea tabella dai dati del bucket S3) per definire lo schema della tabella in blocco. Nell'editor di query, scegli Create (Crea), S3 bucket data (Dati del bucket S3) e quindi scegli Bulk add columns (Aggiungi colonne in blocco) nella sezione Column details (Dettagli colonna).



Digita le coppie nome/valore nella casella di testo e seleziona Add (Aggiungi).



Miglioramenti

- Miglioramento delle prestazioni su tabelle Parquet di grandi dimensioni.

Cronologia dei documenti

Ultimo aggiornamento della documentazione: 28 giugno 2024.

Aggiorniamo la documentazione frequentemente per dar spazio ai feedback. Nella seguente tabella sono descritte importanti aggiunte alla documentazione di Amazon Athena. Non tutti gli aggiornamenti sono rappresentati.

Modifica	Descrizione	Data di rilascio
È stata aggiornata la policy gestita `AmazonAthenaFullAccess` .	`glue:GetCatalogImportStatus` Aggiunto alla `BaseGluePermissions` sezione della politica AmazonAthenaFullAccessgestita. L'azione aggiunta consente ad Athena di utilizzare l' AWS Glue API documentata pubblicamente per recuperare lo stato di importazione del catalogo.	18 giugno 2024
È stata aggiornata la policy gestita `AmazonAthenaFullAccess` .	Le `datazone:ListAccountEnvironments` autorizzazioni `datazone:ListDomains` `datazone:ListProjects` , e sono state aggiunte alla policy AmazonAthenaFullAccessgestita. Le azioni aggiunte consentono agli utenti di Athena di lavorare con DataZone domini, progetti e ambienti Amazon. Per ulteriori informazioni, consulta Utilizzo di Amazon DataZone in Athena.	3 gennaio 2024
È stata aggiornata la policy gestita `AmazonAthenaFullAccess` .	Aggiunte `glue:StartColumnStatisticsTaskRun` e `glue:GetColumnStatisticsTaskRuns` autorizzazioni alla policy gestita AmazonAthenaFullAccess. `glue:GetColumnStatisticsTaskRun` Le azioni aggiunte consentono ad Athena di effettuare chiamate AWS Glue per recuperare le statistiche per la funzionalità di ottimizzazione basata sui costi. Per ulteriori informazioni, consulta Utilizzo dell'ottimizzatore basato sui costi.	3 gennaio 2024

Modifica	Descrizione	Data di rilascio
È stata aggiunta la documentazione per i gruppi di lavoro Athena abilitati per Centro identità IAM.	Dopodiché, è possibile creare gruppi di lavoro Athena SQL che utilizzano la modalità di autenticazione Centro identità IAM. Questi gruppi di lavoro supportano l'utilizzo della stessa identità in AWS servizi come Amazon Athena e Amazon EMR Studio. Per ulteriori informazioni, consulta Utilizzo dei gruppi di lavoro Athena abilitati per Centro identità IAM.	5 dicembre 2023
È stata aggiunta documentazione relativa all'interrogazione dei dati di S3 Express One Zone	È possibile utilizzare Athena per interrogare i dati nella classe di archiviazione Amazon S3 Express One Zone. Per ulteriori informazioni, consulta Interrogazione dei dati di S3 Express One Zone.	28 novembre 2023
È stata aggiunta la documentazione per le viste di Catalogo Dati di Glue.	È possibile utilizzare le viste di Catalogo Dati di Glue per fornire un'unica visualizzazione comune di servizi AWS come Amazon Athena e Amazon Redshift. Per ulteriori informazioni, consulta Usare le viste AWS Glue Data Catalog.	27 novembre 2023
È stata aggiunta la documentazione relativa alla funzionalità dell'ottimizzatore basato sui costi.	Puoi utilizzare le statistiche di AWS Glue per ottimizzare le tue query in Athena SQL. Per ulteriori informazioni, consulta Utilizzo dell'ottimizzatore basato sui costi.	17 novembre 2023
Aggiunta la documentazione per il driver JDBC 3.x di Athena	Il driver JDBC 3.x di Athena è in grado di leggere i risultati delle query direttamente da Amazon S3. Il driver JDBC 3.x supporta quasi tutti i metodi di autenticazione supportati dal driver JDBC 2.x. Per ulteriori informazioni, consulta Driver JDBC 3.x di Athena.	16 novembre 2023

Modifica	Descrizione	Data di rilascio
È stata aggiunta la documentazione per l'utilizzo DataZone in Athena.	Puoi utilizzarlo DataZone per semplificare la tua esperienza su servizi di AWS analisi come Athena e Lake AWS Glue Formation. Per ulteriori informazioni, consulta Utilizzo di Amazon DataZone in Athena.	4 ottobre 2023
Documentazione aggiunta per le prenotazioni della capacità.	Ora puoi utilizzare le prenotazioni della capacità su Amazon Athena per eseguire query SQL su capacità di calcolo completamente gestita. Per ulteriori informazioni, consulta Gestione della capacità di elaborazione delle query.	28 aprile 2023
È stata aggiunta la documentazione per eseguire query sulle visualizzazioni federate.	Ora puoi creare ed eseguire query sulle visualizzazioni di origini di dati federate in Athena. Per ulteriori informazioni, consulta Esecuzione di query su visualizzazioni federate.	4 aprile 2023
È stata aggiunta la documentazione sulla prevenzione della limitazione (della larghezza di banda della rete) in Amazon S3.	Per ulteriori informazioni, consulta Come prevenire la limitazione (della larghezza di banda della rete) di Amazon S3.	24 marzo 2023
È stata aggiornata la policy gestita `AmazonAthenaFullAccess` .	Aggiunto `pricing:GetProducts` alla policy AmazonAthenaFullAccessgestita. L'azione aggiunta fornisce l'accesso a AWS Billing and Cost Management. Per ulteriori informazioni, GetProductsconsulta l'AWS Billing and Cost Management API Reference.	25 gennaio 2023

Modifica	Descrizione	Data di rilascio
Documentazione estesa per il supporto alla compressione Athena.	Sono stati aggiunti singoli argomenti per Compressione delle tabelle Hive, Compressione delle tabelle Iceberg e Livelli di compressione ZSTD. Per ulteriori informazioni, consulta Supporto della compressione in Athena.	20 gennaio 2023
È stata aggiunta la documentazione relativa ad Amazon Athena per Apache Spark.	Ora puoi creare ed eseguire in modo interattivo applicazioni Apache Spark e notebook compatibili con Jupyter su Amazon Athena. Per ulteriori informazioni, consulta Utilizzo di Apache Spark in Amazon Athena.	30 novembre 2022
È stata aggiunta la documentazione relativa al connettore Db2 di Amazon Athena.	Puoi utilizzare il connettore Amazon Athena per IBM Db2 per eseguire query su Db2 da Athena. Per ulteriori informazioni, consulta Connettore IBM Db2 di Amazon Athena	18 novembre 2022
È stata aggiunta la documentazione relativa al riutilizzo dei risultati delle query.	Quando esegui nuovamente una query in Athena, ora hai la possibilità di scegliere di riutilizzare l'ultimo risultato della query memorizzato. Ciò può aumentare le prestazioni e ridurre i costi in termini di numero di byte scansionati. Per ulteriori informazioni, consulta Riutilizzo dei risultati delle query.	8 novembre 2022
Documentazione aggiornata per CloudTrail i log.	Il `CREATE TABLE` DDL per l'interrogazione CloudTrail dei log è stato aggiornato per utilizzare SerDe JSON anziché. CloudTrail SerDe Per ulteriori informazioni, consulta Interrogazione dei log AWS CloudTrail.	3 novembre 2022
Aggiunta la documentazione per la versione 3 del motore Athena.	Per ulteriori informazioni sulla versione 3 del motore Athena, consulta Versione 3 del motore Athena.	13 ottobre 2022

Modifica	Descrizione	Data di rilascio
È stato aggiunto un tutorial sulla configurazione di SSO per ODBC utilizzando il plug-in Okta.	Configura il driver ODBC di Amazon Athena e il plug-in SAML basato su browser per aggiungere la funzionalità Single Sign-on (SSO) utilizzando il gestore dell'identità digitale (IdP) Okta. Per ulteriori informazioni, consulta Configurazione di SSO per ODBC utilizzando il plug-in Okta e il gestore dell'identità digitale (IdP) Okta.	23 agosto 2022
Aggiunta della documentazione per la visualizz azione di piani di query e statistic he nella console Athena.	È possibile utilizzare l'editor di query Athena per visualizz are rappresentazioni grafiche di come verranno eseguite le query e i grafici, dettagli e statistiche sull'esecuzione delle query completate. Per ulteriori informazioni, consulta Visualizzazione dei piani di esecuzione per query SQL e Visualizzazione di statistiche e dettagli di esecuzione per le query completate.	21 luglio 2022
Aggiunta la documentazione per eseguire query sulle viste Apache Hive nei metastore esterni di Hive.	È possibile utilizzare Athena per eseguire query sulle viste Apache create in metastore Hive esterni. Alcune funzioni Hive non sono supportate o richiedono una gestione speciale. Per ulteriori informazioni, consulta Utilizzo delle visualizzazioni Hive.	22 aprile 2022
Aggiunta la documentazione per le query salvate.	Puoi utilizzare la funzione Query salvate in Athena per salvare, richiamare, modificare e rinominare le query. Per ulteriori informazioni, consulta Utilizzo di query salvate questa guida e il riferimento UpdateNam edQueryalle API di Amazon Athena.	28 febbraio 2022
Aggiunta la documentazione di anteprima per il supporto Apache Iceberg.	Athena supporta query di lettura, viaggi nel tempo e scrittura per le tabelle Apache Iceberg che utilizzano il formato Apache Parquet per i dati e il catalogo per il loro metastore. AWS Glue Per ulteriori informazioni, consulta Utilizzo di tabelle Apache Iceberg.	26 novembre 2021

Modifica	Descrizione	Data di rilascio
Aggiunta la documentazione per le query federate tra account.	È possibile utilizzare la funzione di query federata tra account per eseguire query sulle origini dati in un altro account. Per informazioni sull'impostazione delle autorizzazioni per abilitare questa funzionalità, consulta la sezione Abilitazione delle query federate tra account.	12 novembre 2021
Aggiunta documentazione per l'istruzione `UNLOAD`di Athena.	Utilizzo dell'istruzione `UNLOAD` per scrivere query i risultati da un'istruzione `SELECT` nei formati Apache Parquet, ORC, Apache Avro e JSON. Per ulteriori informazioni, consulta UNLOAD.	5 agosto 2021
Aggiunta documentazione per la funzione sulle istruzioni `EXPLAIN` di Athena.	Per ulteriori informazioni, consulta Utilizzo di EXPLAIN e EXPLAIN ANALYZE in Athena e Capire i risultati dell'istruzione EXPLAIN di Athena.	5 aprile 2021
Aggiunte pagine sulla risoluzione dei problemi e l'ottimizzazione delle prestazioni in Athena.	Per ulteriori informazioni, consulta Risoluzione dei problemi in Athena e Ottimizzazione delle prestazioni in Athena.	30 dicembre 2020
Aggiunta la documentazione per il controllo delle versioni del motore Athena e per la versione 2 del motore Athena.	Per ulteriori informazioni, consulta Controllo delle versioni del motore di Athena.	11 novembre 2020

Modifica	Descrizione	Data di rilascio
Documentazione aggiornata delle query federate per il rilascio di disponibilità generale.	Per ulteriori informazioni, consulta Utilizzo di Amazon Athena Federated Query e Usare Athena con CalledVia le chiavi di contesto.	11 novembre 2020
Aggiunta la documentazione per l'utilizzo del driver JDBC con Lake Formation per l'accesso federato ad Athena.	Per ulteriori informazioni, consulta Utilizzo di Lake Formation e dei driver Athena JDBC e ODBC per l'accesso federato ad Athena e Tutorial: Configurazione dell'accesso federato per gli utenti Okta in Athena utilizzando Lake Formation e JDBC.	25 settembre 2020
È stata aggiunta la documentazione per il connettore OpenSearch dati Amazon Athena.	Per ulteriori informazioni, consulta Connettore Amazon Athena OpenSearch .	21 luglio 2020
Aggiunta la documentazione per interrogare i set di dati Hudi.	Per ulteriori informazioni, consulta Utilizzo di Athena per eseguire query sui set di dati Apache Hudi.	9 luglio 2020

Modifica	Descrizione	Data di rilascio
Aggiunta la documentazione sull'interrogazione dei registri del server Web Apache e dei registri del server Web IIS archiviati in Amazon S3.	Per ulteriori informazioni, consulta Esecuzione di query sui log Apache archiviati in Amazon S3 e Esecuzione di query sui log Internet Information Server (IIS) archiviati in Amazon S3.	8 luglio 2020
È stata aggiunta la documentazione per la versione generale di Athena Data Connector per metastore Hive esterno.	Per ulteriori informazioni, consulta Utilizzo di un connettore dati Athena per il metastore Hive esterno.	1 giugno 2020
Aggiunta della documentazione per l'assegnazione di tag alle risorse del catalogo dati.	Per ulteriori informazioni, consulta Assegnazione di tag alle risorse Athena.	1 giugno 2020
Aggiunta la documentazione sulla proiezione delle partizioni.	Per ulteriori informazioni, consulta Proiezione delle partizioni con Amazon Athena.	21 maggio 2020
Aggiornati gli esempi di codice Java per Athena.	Per ulteriori informazioni, consulta Esempi di codice.	11 maggio 2020

Modifica	Descrizione	Data di rilascio
È stato aggiunto un argomento sull'interrogazione dei GuardDuty risultati di Amazon.	Per ulteriori informazioni, consulta Interrogazione dei risultati di Amazon GuardDuty .	19 marzo 2020
È stato aggiunto un argomento sull'utilizzo CloudWatch degli eventi per monitorare le transizioni di stato delle query Athena.	Per ulteriori informazioni, consulta Monitoraggio delle query Athena con eventi Amazon EventBridge.	11 marzo 2020
È stato aggiunto un argomento sull'interrogazione dei log di AWS Global Accelerator flusso con Athena.	Per ulteriori informazioni, consulta Interrogazione dei log di AWS Global Accelerator flusso.	6 febbraio 2020

Modifica	Descrizione	Data di rilascio
• Aggiunta la documentazione sull'utilizzo di CTAS con INSERT INTO per aggiungere dati da un'origine non partizionata a una destinazione partizionata. • Aggiunti i collegamenti di download per la versione di anteprima 1.1.0 del driver ODBC per Athena. • Descrizione per il regex `SHOW DATABASES LIKE` corretta. • Sintassi di `partitioned_by` corretta nell'argomento CTA. • Altre correzioni minori.	Gli aggiornamenti della documentazione includono, a titolo esemplificativo, i seguenti argomenti: • Utilizzo di CTAS e INSERT INTO per ETL e analisi dei dati • Connessione ad Amazon Athena con ODBC (Le funzionalità di anteprima 1.1.0 sono ora incluse nel driver ODBC 1.1.2). • SHOW DATABASES • CREATE TABLE AS	4 febbraio 2020

Modifica	Descrizione	Data di rilascio
Aggiunta la documentazione sull'utilizzo di CTAS con INSERT INTO per aggiungere dati dall'origine non partizionata a una destinazione partizionata.	Per ulteriori informazioni, consulta Utilizzo di CTAS e INSERT INTO per aggirare il limite di 100 partizioni.	22 gennaio 2020
Informazioni sulla posizione dei risultati delle query aggiornate.	Athena non crea più una posizione dei risultati della query "predefinita". Per ulteriori informazioni, consulta Specificare una posizione dei risultati delle query.	20 gennaio 2020
È stato aggiunto un argomento sull'interrogazione di. AWS Glue Data Catalog Informazioni aggiornate sulle Service Quotas (in precedenza "limiti di servizio") in Athena.	Per ulteriori informazioni, consulta i seguenti argomenti: • Esecuzione di query AWS Glue Data Catalog • Service Quotas (Quote di Servizio)	17 gennaio 2020

Modifica	Descrizione	Data di rilascio
Argomento corretto su SerDe OpenCSV per indicare che il tipo deve essere specificato `TIMESTAMP` nel formato numerico UNIX.	Per ulteriori informazioni, consulta OpenCSV per l'elaborazione di file SerDe CSV.	15 gennaio 2020
Argomento di sicurezza sulla crittografia aggiornato per indicare che Athena non supporta le chiavi asimmetriche.	Athena supporta solo chiavi simmetriche per la lettura e la scrittura dei dati. Per ulteriori informazioni, consulta Opzioni di crittografia supportate da Amazon S3.	8 gennaio 2020
Sono state aggiunte informazioni sull'accesso tra account ai bucket Amazon S3 crittografati con una chiave personalizzata. AWS KMS	Per ulteriori informazioni, consulta Accesso da più account a un bucket crittografato con una chiave personalizzata AWS KMS.	13 dicembre 2019

Modifica	Descrizione	Data di rilascio
Aggiunta la documentazione per le query federate, i metastore Hive esterni, Machine Learning e le funzioni definite dall'utente. Aggiunta di nuovi parametri CloudWatch.	Per ulteriori informazioni, consulta i seguenti argomenti: • Utilizzo di Amazon Athena Federated Query • Connettori di origine dati disponibili • Utilizzo di un connettore dati Athena per il metastore Hive esterno • Utilizzo di Machine Learning (ML) con Amazon Athena • Esecuzione di query con funzioni definite dall'utente • Elenco di CloudWatch metriche e dimensioni per Athena	26 novembre 2019
Aggiunta della sezione per il nuovo comando INSERT INTO e informazioni aggiornate sulla posizione dei risultati della query per il supporto dei file manifest dei dati.	Per ulteriori informazioni, consulta INSERT INTO e Utilizzo dei risultati delle query, delle query recenti e dei file di output.	18 settembre 2019

Modifica	Descrizione	Data di rilascio
È stata aggiunta una sezione per il supporto degli endpoint VPC dell'interfaccia ()PrivateLink. Aggiornamento dei driver JDBC. Sono state aggiornate le informazioni sui registri di flusso VPC avanzati.	Per ulteriori informazioni, consulta Connessione ad Amazon Athena utilizzando un endpoint VPC di interfaccia, Esecuzione di query sui log di flusso Amazon VPC e Connessione ad Amazon Athena con JDBC.	11 settembre 2019
È stata aggiunta una sezione sull'integrazione con. AWS Lake Formation	Per ulteriori informazioni, consulta Utilizzo di Athena per eseguire query sui dati registrati con AWS Lake Formation.	26 giugno 2019
Aggiornamento della sezione relativa alla sicurezza per coerenza con altri servizi AWS .	Per ulteriori informazioni, consulta Sicurezza di Amazon Athena.	26 giugno 2019
È stata aggiunta una sezione sull'interrogazione dei log. AWS WAF	Per ulteriori informazioni, consulta Interrogazione dei log AWS WAF.	31 maggio 2019

Modifica	Descrizione	Data di rilascio
Rilasciata la nuova versione del driver ODBC con il supporto per i gruppi di lavoro di Athena.	Per scaricare e la versione 1.0.5 del driver ODBC, consulta Connessione ad Amazon Athena con ODBC. Non ci sono variazioni alla stringa di connessione del driver ODBC quando si utilizzano i tag nei gruppi di lavoro. Per usare i tag, esegui l'aggiornamento alla versione più recente del driver ODBC, ovvero la versione corrente. Questa versione del driver consente di usare le operazioni dei gruppi di lavoro dell'API Athena per creare e gestire i gruppi di lavoro e le operazioni dei tag dell'API Athena per aggiungere, elencare o rimuovere tag nei gruppi di lavoro. Prima di iniziare, verifica di disporre delle autorizzazioni a livello di risorsa in IAM per le operazioni su gruppi di lavoro e tag.	5 marzo 2019
Aggiunto il supporto dei tag per i gruppi di lavoro in Amazon Athena.	Un tag consiste di una chiave e di un valore che definisci . Quando aggiungi tag a un gruppo di lavoro, gli assegni metadati personalizzati. Ad esempio, crea un gruppo di lavoro per ogni centro di costo. Quindi, aggiungendo tag a questi gruppi di lavoro, puoi tenere traccia della spesa di Athena per ogni centro di costo. Per ulteriori informazioni, consulta la sezione relativa all'utilizzo dei tag per la fatturazione nella Guida per l'utente di AWS Billing and Cost Management .	22 febbraio 2019

Modifica	Descrizione	Data di rilascio
Migliorato il formato JSON SerDe OpenX utilizzato in Athena.	I miglioramenti includono, tra l'altro, quanto segue: • Supporto della proprietà `ConvertDotsInJsonKeysToUnderscores` . Se impostato su`TRUE`, consente di sostituire i punti nei SerDe nomi delle chiavi con caratteri di sottolineatura. Ad esempio, se il set di dati JSON contiene una chiave denominata `"a.b"`, è possibile usare questa proprietà per definire il nome della colonna come `"a_b"` in Athena. Il valore predefinito è `FALSE`. Per impostazione predefinita, Athena non consente l'uso di punti nei nomi di colonna. • Supporto della proprietà `case.insensitive` . Per impostazione predefinita, Athena richiede che tutte le chiavi nel set di dati JSON siano in caratteri minuscoli. `WITH SERDE PROPERTIES ("case.insensitive"= FALSE;)` consente di usare nomi di chiave con distinzione tra maiuscole e minuscole nei dati. Il valore predefinito è `TRUE`. Se impostato su`TRUE`, SerDe converte tutte le colonne maiuscole in minuscole. Per ulteriori informazioni, consulta OpenX JSON SerDe.	18 febbraio 2019

Modifica	Descrizione	Data di rilascio
Aggiunto il supporto per i gruppi di lavoro.	Usa i gruppi di lavoro per separare utenti, team, applicazioni o carichi di lavoro e per impostare i limiti relativi alla quantità di dati che ogni query o l'intero gruppo di lavoro può elaborare. Poiché i gruppi di lavoro fungono da risorse IAM, puoi utilizzare le autorizzazioni a livello di risorsa per controllare l'accesso a un determinato gruppo di lavoro. Puoi anche visualizzare i parametri relativi alle query in Amazon CloudWatch, controllare i costi delle query configurando limiti alla quantità di dati scansionati, creare soglie e attivare azioni, come gli allarmi di Amazon SNS, quando queste soglie vengono violate. Per ulteriori informazioni, consulta Utilizzo dei gruppi di lavoro per l'esecuzione di query e Controllo dei costi e monitoraggio delle interrogazioni con metriche ed eventi CloudWatch .	18 febbraio 2019
È stato aggiunto il supporto per l'analisi dei log di Network Load Balancer.	Aggiunte query Athena di esempio per analizzare i log di Network Load Balancer. Questi log ricevono informazioni dettagliate sulle richieste Transport Layer Security (TLS) inviate al Network Load Balancer. Puoi utilizzare questi log per analizzare i modelli di traffico e risolvere i problemi che potresti incontrare. Per informazioni, consulta the section called “Network Load Balancer”.	24 gennaio 2019
Rilasciate le nuove versioni dei driver JDBC e ODBC con supporto per l'accesso federato alle API di Athena con AD FS e SAML 2.0 (Security Assertion Markup Language 2.0).	Questa versione dei driver supporta l'accesso federato ad Athena per Active Directory Federation Service (AD FS 3.0). L'accesso viene stabilito attraverso le versioni dei driver JDBC o ODBC che supportano SAML 2.0. Per ulteriori informazioni sulla configurazione dell'accesso federato all'API Athena, consulta the section called “Abilitazione dell'accesso federato all'API Athena”.	10 Novembre 2018

Modifica	Descrizione	Data di rilascio
Aggiunto il supporto per un controllo capillare degli accessi a database e tabelle in Athena. Inoltre, aggiunta di policy in Athena che consentono di crittografare i metadati dei database e delle tabelle nel Catalogo Dati.	È stato aggiunto il supporto per la creazione di policy basate sull'identità (IAM) che forniscono un controllo granulare degli accessi alle risorse in AWS Glue Data Catalog, come database e tabelle utilizzati in Athena. Inoltre, è possibile crittografare i metadati dei database e delle tabelle nel catalogo dati aggiungendo policy specifiche ad Athena. Per informazioni dettagliate, vedi Accesso granulare a database e tabelle in AWS Glue Data Catalog.	15 ottobre 2018
Aggiunto supporto per le istruzioni `CREATE TABLE AS SELECT`. Apportati altri miglioramenti alla documentazione.	Aggiunto supporto per le istruzioni `CREATE TABLE AS SELECT`. Consulta le sezioni Creazione di una tabella dai risultati delle query (CTAS), Considerazioni e restrizioni per le query CTAS e Esempi di query CTAS.	10 ottobre 2018
Rilascio del driver ODBC versione 1.0.3 con il supporto per lo streaming dei risultati invece del recupero in pagine. Apportati altri miglioramenti alla documentazione.	La versione 1.0.3 del driver ODBC supporta lo streaming dei risultati e include anche miglioramenti, correzioni di bug e un aggiornamento della documentazione relativa all'utilizzo di SSL con un server proxy. Per scaricare la versione 1.0.3 del driver ODBC e la relativa documentazione, consulta Connessione ad Amazon Athena con ODBC.	6 settembre 2018

Modifica	Descrizione	Data di rilascio
Rilascio del driver JDBC versione 2.0.5 con il supporto predefinito per lo streaming dei risultati invece del recupero in pagine. Apportati altri miglioramenti alla documentazione.	Rilascio del driver JDBC versione 2.0.5 con il supporto predefinito per lo streaming dei risultati invece del recupero in pagine. Per informazioni, consulta Connessione ad Amazon Athena con JDBC.	16 agosto 2018
Aggiornamento della documentazione per l'esecuzione di query sui log di flusso di Amazon Virtual Private Cloud, che possono essere archiviati direttamente in Amazon S3 in formato GZIP. Aggiornamento degli esempi per l'esecuzione di query sui log ALB.	Aggiornamento della documentazione per l'esecuzione di query sui log di flusso di Amazon Virtual Private Cloud, che possono essere archiviati direttamente in Amazon S3 in formato GZIP. Per informazioni, consulta Esecuzione di query sui log di flusso Amazon VPC. Aggiornamento degli esempi per l'esecuzione di query sui log ALB. Per informazioni, consulta Esecuzione di query nei log di Application Load Balancer.	7 agosto 2018

Modifica	Descrizione	Data di rilascio
Aggiunta del supporto per le visualizzazioni. Aggiunta delle linee guida per le manipolazioni degli schemi per vari formati di storage dei dati.	Aggiunta del supporto per le visualizzazioni. Per informazioni, consulta Utilizzo delle visualizzazioni. Aggiornamento della guida con istruzioni su come gestire gli aggiornamenti dello schema per vari formati di storage dei dati. Per informazioni, consulta Gestione degli aggiornamenti degli schemi.	5 giugno 2018
Aumento dei limiti predefiniti per le query simultanee da cinque a venti.	È possibile inviare ed eseguire fino a venti query `DDL` e venti query `SELECT` alla volta. Per informazioni, consulta Service Quotas (Quote di Servizio).	17 maggio 2018
Aggiunta delle schede di query e della possibilità di configurare il completamento automatico nell'Editor di query.	Aggiunta delle schede di query e della possibilità di configurare il completamento automatico nell'Editor di query. Per informazioni, consulta Nozioni di base.	8 maggio 2018
Rilascio della versione 2.0.2 del driver JDBC.	Rilascio della nuova versione del driver JDBC (versione 2.0.2). Per informazioni, consulta Connessione ad Amazon Athena con JDBC.	19 aprile 2018
Aggiunta del completamento automatico per la digitazione di query nella console Athena.	Aggiunta del completamento automatico per la digitazione di query nella console Athena.	6 aprile 2018

Modifica	Descrizione	Data di rilascio
È stata aggiunta la possibilità di creare tabelle Athena per i file di CloudTrail registro direttamente dalla CloudTrail console.	È stata aggiunta la possibilità di creare automaticamente tabelle Athena per i file di CloudTrail registro direttamente dalla CloudTrail console. Per informazioni, consulta Utilizzo della CloudTrail console per creare una tabella Athena per i log CloudTrail .	15 marzo 2018
Aggiunta del supporto per l'offloading sicuro dei dati intermedi su disco per le query con GROUP BY.	Aggiunta del supporto per l'offloading sicuro dei dati intermedi su disco per le query con elevati requisiti di memoria che utilizzano la clausola `GROUP BY`. Questo migliora l'affidabilità di tali query, impedendo gli errori di tipo "Query resource exhausted". Per ulteriori informazioni,consulta le note di rilascio per 2 febbraio 2018.	2 febbraio 2018
Aggiunta del supporto per Presto versione 0.172.	Aggiornamento del motore sottostante in Amazon Athena a una versione basata su Presto 0.172. Per ulteriori informazioni,consulta le note di rilascio per 19 gennaio 2018.	19 gennaio 2018
Aggiunta del supporto per il driver ODBC.	Aggiunto supporto per collegare Athena al driver ODBC. Per ulteriori informazioni, consulta la sezione relativa alla connessione ad Amazon Athena con ODBC.	13 Novembre 2017
Aggiunto supporto per le Regioni Asia Pacifico (Seoul), Asia Pacifico (Mumbai) ed Europa (Londra). Aggiunta del supporto per eseguire query sui dati geospaziali.	Aggiunto supporto per le query su dati geospaziali e per le Regioni Asia Pacifico (Seoul), Asia Pacifico (Mumbai) ed Europa (Londra). Per ulteriori informazioni, consulta Esecuzione di query sui dati geospaziali e Regioni AWS ed endpoint.	1 Novembre 2017

Modifica	Descrizione	Data di rilascio
Aggiunto il supporto per Europa (Francofo rte).	Aggiunto il supporto per Europa (Francoforte). Per un elenco delle regioni supportate, consulta Regioni AWS ed endpoint.	19 ottobre 2017
È stato aggiunto il supporto per le query denominate Athena con. AWS CloudFormation	È stato aggiunto il supporto per la creazione di query denominate Athena con. AWS CloudFormation Per ulteriori informazioni, consulta AWS::Athena::Named Queryla Guida per l'AWS CloudFormation utente.	3 ottobre 2017
Aggiunta del supporto per la Regione Asia Pacifico (Sydney).	Aggiunta del supporto per la Regione Asia Pacifico (Sydney). Per un elenco delle regioni supportate, consulta Regioni AWS ed endpoint.	25 settembre 2017
È stata aggiunta una sezione a questa guida per l'interrogazione di Servizio AWS log e diversi tipi di dati, tra cui mappe, array, dati annidati e dati contenenti JSON.	Aggiunta di esempi per Interrogazione dei log Servizio AWS e per l'esecuzione di query su diversi tipi di dati in Athena. Per informazioni, consulta Esecuzione di query SQL con Amazon Athena.	5 settembre 2017
È AWS Glue Data Catalog stato aggiunto il supporto per.	È stata aggiunta l'integrazione con AWS Glue Data Catalog e una procedura guidata di migrazione per l'aggiornamento dal catalogo di dati gestito da Athena a. AWS Glue Data Catalog Per ulteriori informazioni, consulta Integrazione con AWS Glue e AWS Glue.	14 agosto 2017

Modifica	Descrizione	Data di rilascio
È stato aggiunto il supporto per Grok. SerDe	È stato aggiunto il supporto per Grok SerDe, che semplifica la corrispondenza dei pattern per i record in file di testo non strutturati come i log. Per ulteriori informazioni, consulta Grok. SerDe Sono state aggiunte le scorciatoie da tastiera per scorrere la cronologia delle query utilizzando la console.	4 agosto 2017
Aggiunto il supporto per la Regione Asia Pacifico (Tokyo).	Aggiunto il supporto per le Regioni Asia Pacifico (Tokyo) e Asia Pacifico (Singapore). Per un elenco delle regioni supportate, consulta Regioni AWS ed endpoint.	22 giugno 2017
Aggiunto il supporto per la Regione Europa (Irlanda).	Aggiunto il supporto per la Regione Europa (Irlanda). Per ulteriori informazioni, consulta Regioni AWS ed endpoint.	8 giugno 2017
Aggiunti un'API e AWS CLI supporto Amazon Athena.	Aggiunti un'API Amazon Athena e AWS CLI supporto per Athena. Aggiornamento del driver JDBC alla versione 1.1.0.	19 maggio 2017
Aggiunto supporto per la crittografia dei dati di Amazon S3.	Aggiunto il supporto per la crittografia dei dati Amazon S3 e rilasciato un aggiornamento del driver JDBC (versione 1.0.1) con supporto per la crittografia, miglioramenti e correzioni di bug. Per ulteriori informazi oni, consulta Crittografia a riposo.	4 Aprile 2017

Modifica	Descrizione	Data di rilascio
Aggiunto il. AWS CloudTrail SerDe	Aggiunta la AWS CloudTrail SerDe, prestazioni migliorate, problemi di partizione risolti. • AWS CloudTrail SerDe È stato sostituito dal registro JSON Hive SerDe per la lettura. CloudTrail Per informazioni sull' CloudTrail interrogazione dei log, vedere. Interrogazione dei log AWS CloudTrail • Miglioramento delle prestazioni durante la scansione di un numero elevato di partizioni. • Miglioramento delle prestazioni nell'operazione `MSCK Repair Table`. • Aggiunta la possibilità di eseguire query sui dati Amazon S3 archiviati in regioni diverse da quella primaria. In aggiunta alle tariffe standard di Athena, si applicano i costi di trasferimento dei dati interregionali di per Amazon S3.	24 marzo 2017
Aggiunto il supporto per gli Stati Uniti orientali (Ohio).	Aggiunto il supporto per Avro SerDe e OpenCSV per l'elaborazione di file SerDe CSV, Stati Uniti orientali (Ohio), aggiunta la modifica in blocco delle colonne nella procedura guidata della console. Miglioramento delle prestazioni su tabelle Parquet di grandi dimensioni.	20 febbraio 2017
	Versione iniziale della Guida per l'utente di Amazon Athena.	Novembre 2016

AWS Glossario

Per la AWS terminologia più recente, consultate il AWS glossario nella sezione Reference.Glossario AWS

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.