Le migliori pratiche per l'ottimizzazione delle prestazioni per i job di AWS Glue Apache Spark

: Le migliori pratiche per l'ottimizzazione delle prestazioni per i job di AWS Glue Apache Spark

Table of Contents

Le migliori pratiche per l'ottimizzazione delle prestazioni per i job di AWS Glue Apache Spark

Roman Myers, Takashi Onikura e Noritaka Sekiyama, Amazon Web Services ()AWS

Dicembre 2023 (cronologia dei documenti)

AWS Glue offre diverse opzioni per ottimizzare le prestazioni. Questa guida definisce gli argomenti chiave per l'ottimizzazione di Apache AWS Glue Spark. Fornisce quindi una strategia di base da seguire per ottimizzarli AWS Glue per i job di Apache Spark. Usa questa guida per imparare a identificare i problemi di prestazioni interpretando le metriche disponibili in. AWS Glue Quindi incorpora strategie per risolvere questi problemi, massimizzando le prestazioni e riducendo al minimo i costi.

Questa guida illustra le seguenti pratiche di ottimizzazione:

- Scalare la capacità del cluster
- Usa la AWS Glue versione più recente
- Riduci la quantità di scansione dei dati
- Parallelizza le attività
- Riduci al minimo i costi di pianificazione
- Ottimizza gli shuffles
- Ottimizza le funzioni definite dall'utente

Argomenti chiave di Apache Spark

Questa sezione spiega i concetti di base di Apache Spark e gli argomenti chiave per l'ottimizzazione delle prestazioni di Apache AWS Glue Spark. È importante comprendere questi concetti e argomenti prima di discutere delle strategie di ottimizzazione del mondo reale.

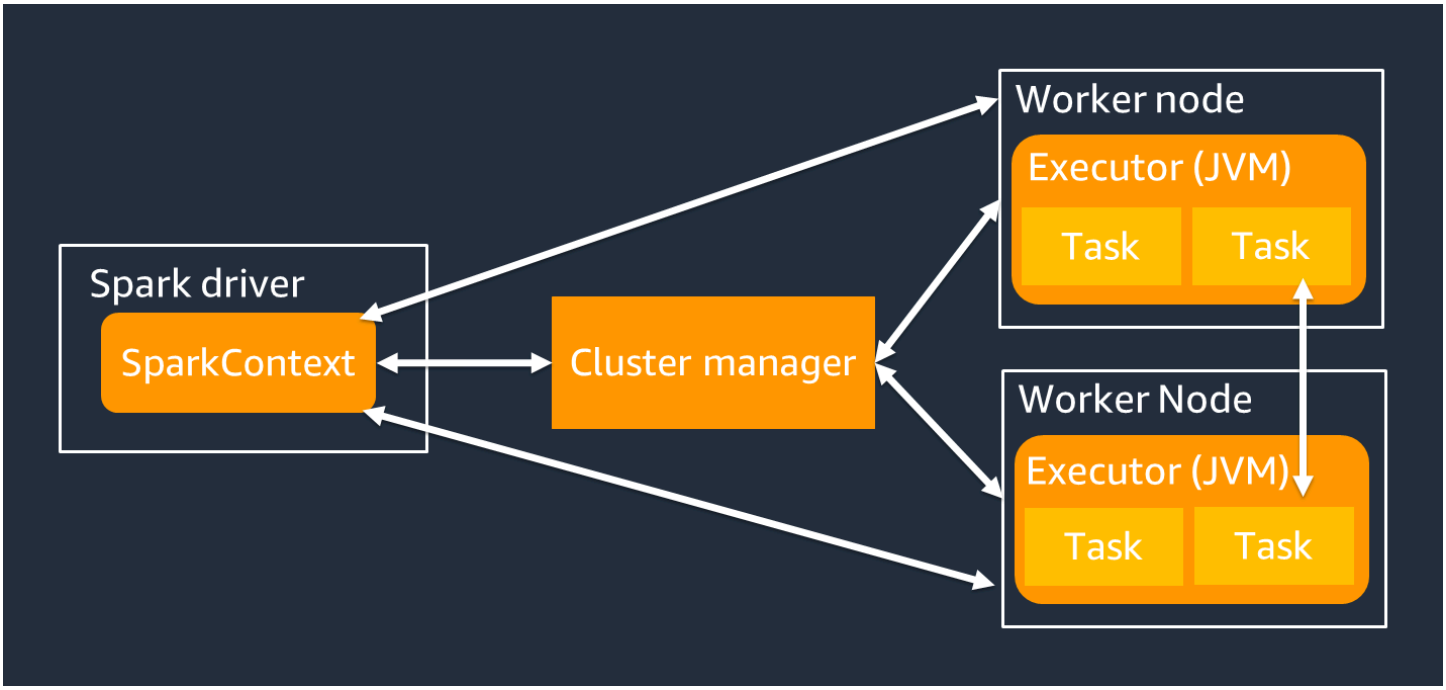
Architettura

Il driver Spark è principalmente responsabile della suddivisione dell'applicazione Spark in attività che possono essere eseguite su singoli lavoratori. Il driver Spark ha le seguenti responsabilità:

- È in esecuzione `main()` nel tuo codice
- Generazione di piani di esecuzione
- Fornitura degli esecutori Spark in collaborazione con il gestore del cluster, che gestisce le risorse del cluster
- Pianificazione delle attività e richiesta di attività per gli esecutori Spark
- Gestione dell'avanzamento e del ripristino delle attività

Si utilizza un `SparkContext` oggetto per interagire con il driver Spark durante l'esecuzione del lavoro.

Un esecutore Spark è un lavoratore che conserva dati ed esegue attività che vengono trasmesse dal driver Spark. Il numero di esecutori Spark aumenterà e diminuirà in base alle dimensioni del cluster.

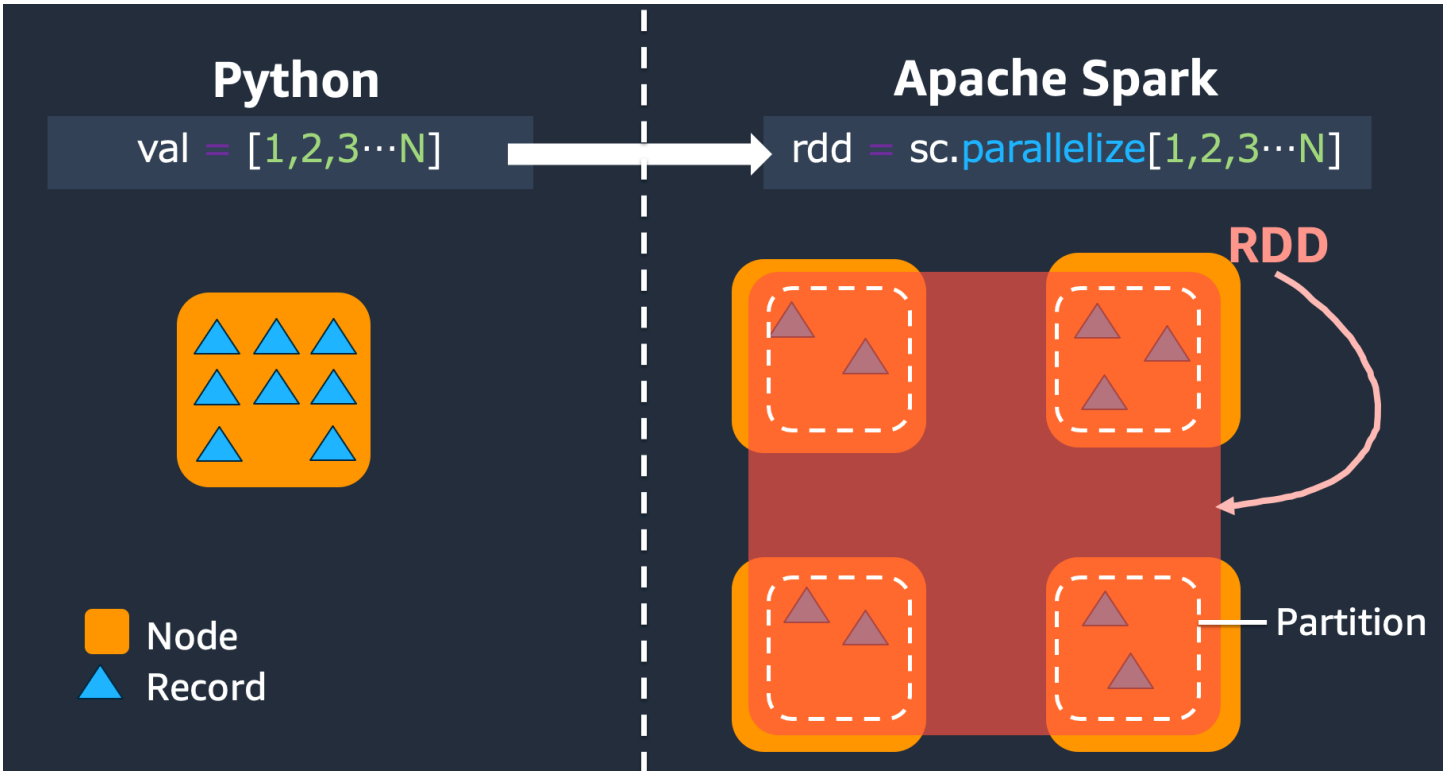


> **Note**
>
> Un esecutore Spark ha più slot in modo che più attività vengano elaborate in parallelo. Per impostazione predefinita, Spark supporta un'attività per ogni core della CPU virtuale (vCPU). Ad esempio, se un executor ha quattro core CPU, può eseguire quattro attività simultanee.

Set di dati distribuito resiliente

Spark svolge il complesso lavoro di archiviazione e tracciamento di set di dati di grandi dimensioni tra gli esecutori Spark. Quando scrivi codice per i job Spark, non devi pensare ai dettagli dello storage. Spark fornisce l'astrazione resiliente del set di dati distribuito (RDD), che è una raccolta di elementi su cui è possibile operare in parallelo e che possono essere partizionati tra gli esecutori Spark del cluster.

La figura seguente mostra la differenza nel modo in cui archiviare i dati in memoria quando uno script Python viene eseguito nel suo ambiente tipico e quando viene eseguito nel framework Spark (). PySpark



- Python: la scrittura `val = [1,2,3...N]` in uno script Python mantiene i dati in memoria sulla singola macchina su cui è in esecuzione il codice.
- PySpark— Spark fornisce la struttura dati RDD per caricare ed elaborare i dati distribuiti nella memoria su più esecutori Spark. Puoi generare un RDD con codice come`rdd = sc.parallelize[1,2,3...N]`, e Spark può distribuire e conservare automaticamente i dati in memoria su più esecutori Spark.

  In molti AWS Glue lavori, usi RDD tramite e Spark. AWS Glue DynamicFramesDataFrames Si tratta di astrazioni che consentono di definire lo schema dei dati in un RDD ed eseguire attività di livello superiore con tali informazioni aggiuntive. Poiché utilizzano RDD internamente, i dati vengono distribuiti in modo trasparente e caricati su più nodi nel seguente codice:

  - DynamicFrame

    ```
    dyf= glueContext.create_dynamic_frame.from_options(
        's3', {"paths": [ "s3://<YourBucket>/<Prefix>/"]},
        format="parquet",
        transformation_ctx="dyf"
    )
    ```

  - DataFrame

```
df = spark.read.format("parquet")
    .load("s3://<YourBucket>/<Prefix>")
```

Un RDD presenta le seguenti funzionalità:

- Gli RDD sono costituiti da dati suddivisi in più parti chiamate partizioni. Ogni esecutore Spark archivia una o più partizioni in memoria e i dati vengono distribuiti su più esecutori.
- Gli RDD sono immutabili, il che significa che non possono essere modificati dopo essere stati creati. Per modificare un DataFrame, puoi usare le trasformazioni, che sono definite nella sezione seguente.
- Gli RDD replicano i dati tra i nodi disponibili, in modo da poter eseguire automaticamente il ripristino in caso di errori dei nodi.

Valutazione pigra

Gli RDD supportano due tipi di operazioni: le trasformazioni, che creano un nuovo set di dati da uno esistente, e le azioni, che restituiscono un valore al programma driver dopo aver eseguito un calcolo sul set di dati.

- Trasformazioni: poiché gli RDD sono immutabili, è possibile modificarli solo utilizzando una trasformazione.

  Ad esempio, `map` è una trasformazione che passa ogni elemento del set di dati attraverso una funzione e restituisce un nuovo RDD che rappresenta i risultati. Notate che il `map` metodo non restituisce un output. Spark memorizza la trasformazione astratta per il futuro, invece di lasciarti interagire con il risultato. Spark non agirà sulle trasformazioni finché non richiederai un'azione.
- Azioni: utilizzando le trasformazioni, costruisci il tuo piano di trasformazione logica. Per avviare il calcolo, si esegue un'azione come`write`,`count`, `show` o. `collect`

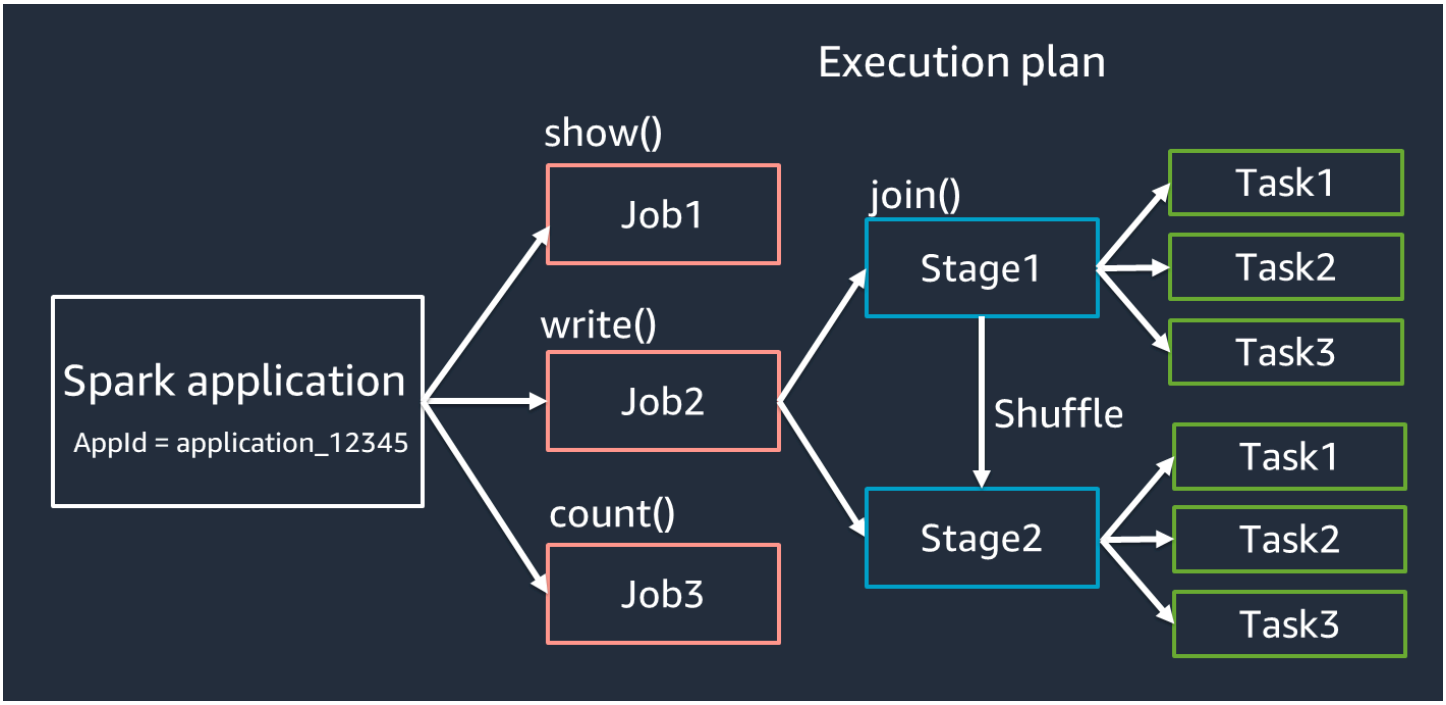
  Tutte le trasformazioni in Spark sono lente, in quanto non calcolano immediatamente i risultati. Spark ricorda invece una serie di trasformazioni applicate ad alcuni set di dati di base, come gli oggetti Amazon Simple Storage Service (Amazon S3). Le trasformazioni vengono calcolate solo quando un'azione richiede la restituzione di un risultato al driver. Questo design consente a Spark di funzionare in modo più efficiente. Ad esempio, si consideri la situazione in cui un set di dati creato tramite la `map` trasformazione viene utilizzato solo da una trasformazione che riduce

sostanzialmente il numero di righe, ad esempio. `reduce` È quindi possibile passare al driver il set di dati più piccolo che ha subito entrambe le trasformazioni, invece di passare il set di dati mappato più grande.

Terminologia delle applicazioni Spark

Questa sezione tratta la terminologia delle applicazioni Spark. Il driver Spark crea un piano di esecuzione e controlla il comportamento delle applicazioni in diverse astrazioni. I seguenti termini sono importanti per lo sviluppo, il debug e l'ottimizzazione delle prestazioni con l'interfaccia utente Spark.

- Applicazione: basata su una sessione Spark (contesto Spark). Identificato da un ID univoco come. `<application_XXX>`
- Offerte di lavoro: in base alle azioni create per un RDD. Un lavoro è costituito da una o più fasi.
- Fasi: basato sugli shuffle creati per un RDD. Una fase è costituita da una o più attività. Lo shuffle è il meccanismo di Spark per ridistribuire i dati in modo che siano raggruppati in modo diverso tra le partizioni RDD. Alcune trasformazioni, ad esempio, richiedono un shuffle. `join()` Lo shuffle viene discusso più dettagliatamente nella pratica di ottimizzazione di Optimize shuffles.
- Attività: un'attività è l'unità minima di elaborazione pianificata da Spark. Le attività vengono create per ogni partizione RDD e il numero di attività è il numero massimo di esecuzioni simultanee nella fase.

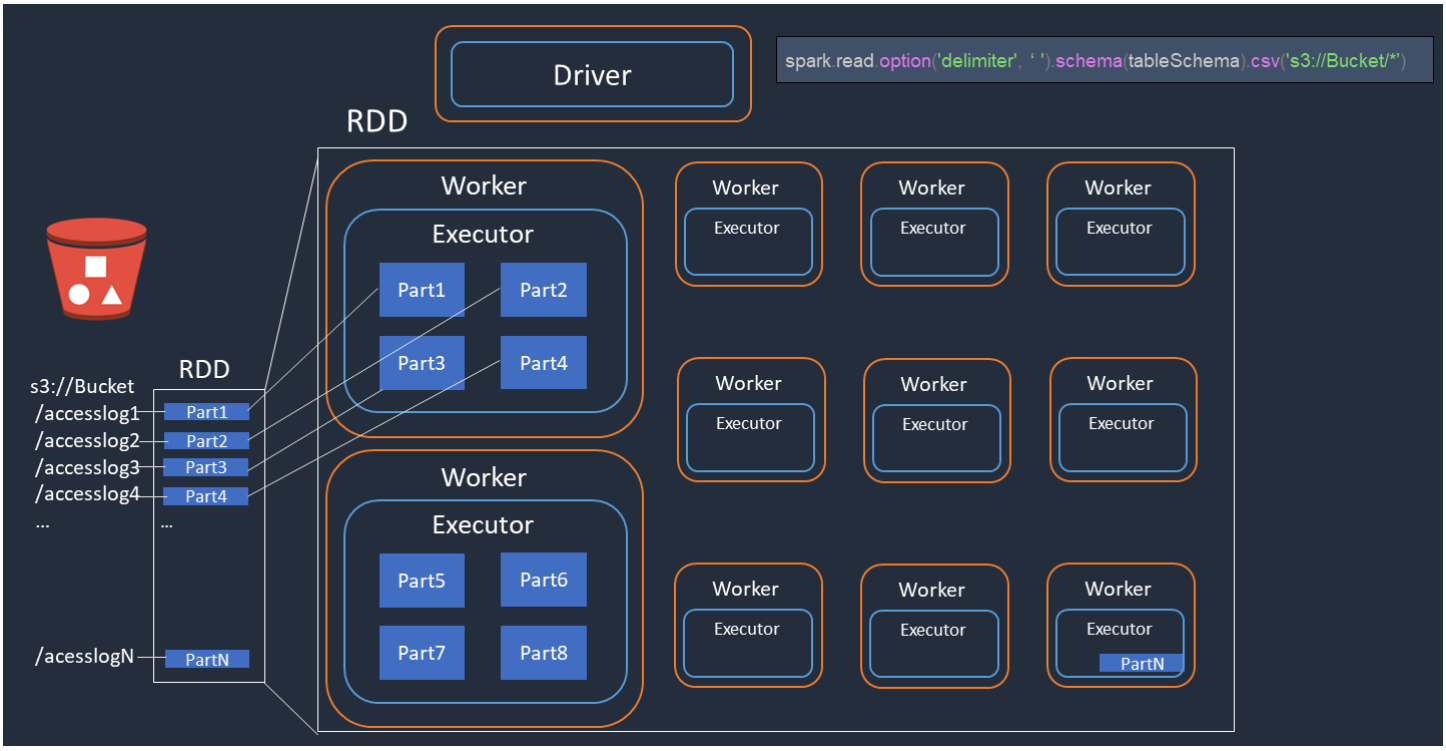


> **Note**
>
> Le attività sono la cosa più importante da considerare quando si ottimizza il parallelismo. Il numero di attività varia in base al numero di RDD

Parallelism

Spark parallelizza le attività per il caricamento e la trasformazione dei dati.

Prendi in considerazione un esempio in cui esegui l'elaborazione distribuita dei file di log di accesso (denominati`accesslog1 ... accesslogN`) su Amazon S3. Il diagramma seguente mostra il flusso di elaborazione distribuito.

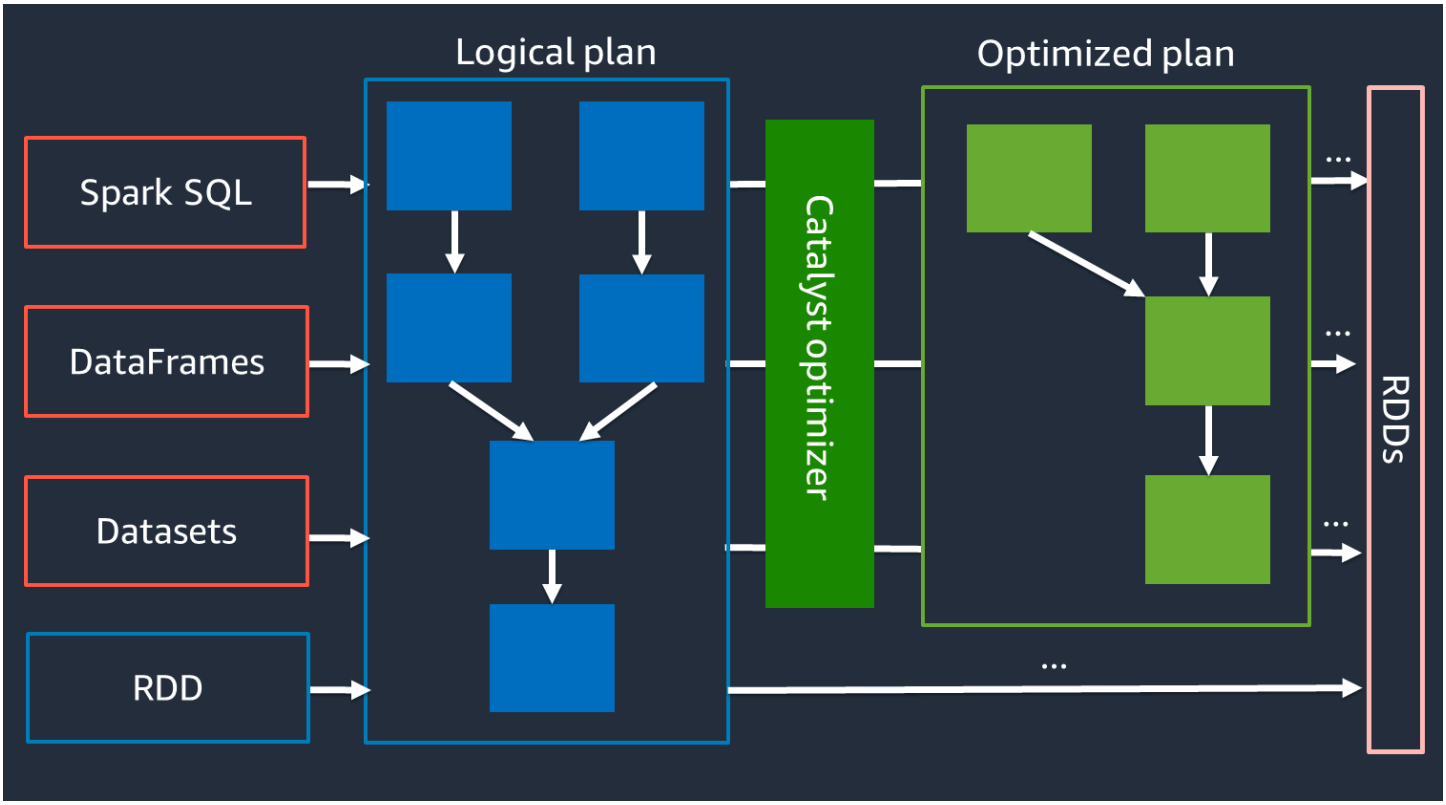


1. Il driver Spark crea un piano di esecuzione per l'elaborazione distribuita su molti esecutori Spark.
2. Il driver Spark assegna le attività a ciascun esecutore in base al piano di esecuzione. Per impostazione predefinita, il driver Spark crea partizioni RDD (ciascuna corrispondente a un'attività Spark) per ogni oggetto S3 (). `Part1 ... N` Quindi il driver Spark assegna le attività a ciascun esecutore.
3. Ogni task Spark scarica l'oggetto S3 assegnato e lo archivia in memoria nella partizione RDD. In questo modo, più esecutori Spark scaricano ed elaborano l'attività assegnata in parallelo.

Per maggiori dettagli sul numero iniziale di partizioni e sull'ottimizzazione, consulta la sezione Parallelizza le attività.

Ottimizzatore Catalyst

Internamente, Spark utilizza un motore chiamato Catalyst Optimizer per ottimizzare i piani di esecuzione. Catalyst dispone di un ottimizzatore di query che puoi usare quando esegui API Spark di alto livello, come Spark SQL e Datasets, come descritto nel diagramma seguente. DataFrame



Poiché l'ottimizzatore Catalyst non funziona direttamente con l'API RDD, le API di alto livello sono generalmente più veloci delle API RDD di basso livello. Per i join complessi, l'ottimizzatore Catalyst può migliorare significativamente le prestazioni ottimizzando il piano di esecuzione dei job. Puoi vedere il piano ottimizzato del tuo job Spark nella scheda SQL dell'interfaccia utente di Spark.

Esecuzione adattiva delle interrogazioni

L'ottimizzatore Catalyst esegue l'ottimizzazione del runtime tramite un processo chiamato Adaptive Query Execution. Adaptive Query Execution utilizza le statistiche di runtime per ottimizzare nuovamente il piano di esecuzione delle query durante l'esecuzione del lavoro. Adaptive Query Execution offre diverse soluzioni ai problemi di prestazioni, tra cui la coalescenza delle partizioni post-shuffle, la conversione dello sort-merge join in broadcast join e l'ottimizzazione dello skew join, come descritto nelle sezioni seguenti.

Adaptive Query Execution è disponibile nella versione 3.0 e successive ed è abilitata per AWS Glue impostazione predefinita nella versione 4.0 (Spark 3.3.0) e versioni successive. AWS Glue Adaptive Query Execution può essere attivata e disattivata utilizzando `spark.conf.set("spark.sql.adaptive.enabled", "true")` il codice.

Partizioni unificate dopo lo shuffle

Questa funzione riduce le partizioni RDD (coalescenza) dopo ogni mescolamento in base alle statistiche di output. `map` Semplifica l'ottimizzazione del numero di partizione shuffle durante l'esecuzione di query. Non è necessario impostare un numero di partizione shuffle per adattarlo al set di dati. Spark può scegliere il numero di partizione shuffle corretto in fase di esecuzione dopo aver ottenuto un numero iniziale sufficientemente grande di partizioni shuffle.

La coalescenza delle partizioni post-shuffle è abilitata quando entrambe le partizioni sono impostate su true. `spark.sql.adaptive.enabled` `spark.sql.adaptive.coalescePartitions.enabled` Per ulteriori informazioni, consulta la documentazione di Apache Spark.

Conversione da sort-merge join a broadcast join

Questa funzione riconosce quando si uniscono due set di dati di dimensioni sostanzialmente diverse e adotta un algoritmo di join più efficiente basato su tali informazioni. Per maggiori dettagli, consulta la documentazione di Apache Spark. Le strategie di unione sono discusse nella sezione Optimize shuffles.

Ottimizzazione delle giunzioni asimmetriche

La distorsione dei dati è uno degli ostacoli più comuni per i lavori Spark. Descrive una situazione in cui i dati vengono distorti su partizioni RDD specifiche (e di conseguenza, attività specifiche), il che ritarda il tempo di elaborazione complessivo dell'applicazione. Ciò può spesso ridurre le prestazioni delle operazioni di join. La funzionalità di ottimizzazione dello skew join gestisce dinamicamente l'inclinazione nei join ordin-merge suddividendo (e replicando se necessario) le attività asimmetriche in attività di dimensioni approssimativamente uguali.

Questa `spark.sql.adaptive.skewJoin.enabled` funzionalità è abilitata quando è impostata su true. Per maggiori dettagli, consulta la documentazione di Apache Spark. La distorsione dei dati viene discussa ulteriormente nella sezione Optimize shuffles.

Analizza i problemi di prestazioni utilizzando l'interfaccia utente di Spark

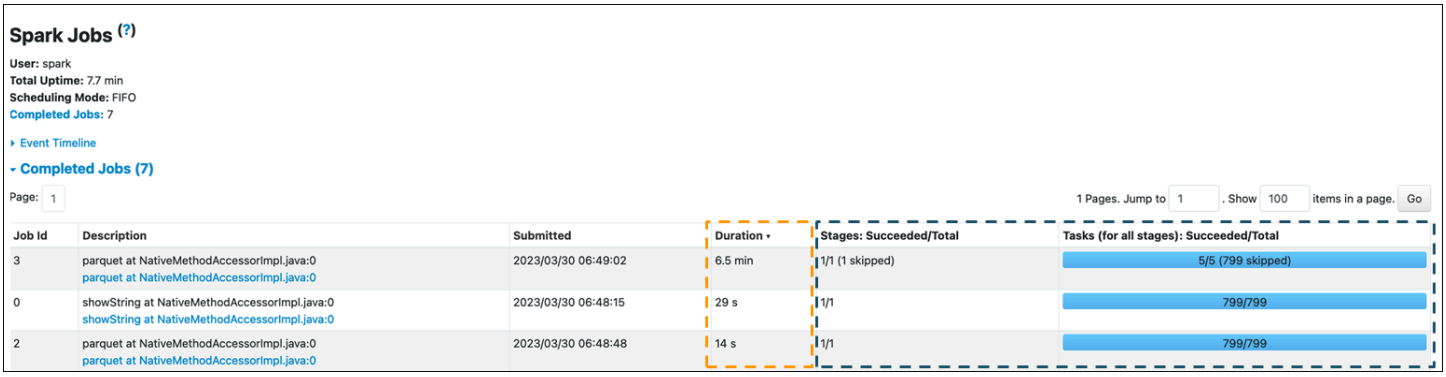
Prima di applicare le migliori pratiche per ottimizzare le prestazioni dei vostri AWS Glue lavori, vi consigliamo vivamente di profilare le prestazioni e identificare gli ostacoli. Questo ti aiuterà a concentrarti sulle cose giuste.

Per un'analisi rapida, le CloudWatch metriche di Amazon forniscono una visualizzazione di base delle metriche relative alle tue offerte di lavoro. L'interfaccia utente Spark offre una visione più approfondita per l'ottimizzazione delle prestazioni. Per utilizzare l'interfaccia utente Spark con AWS Glue, devi abilitare l'interfaccia utente Spark per i tuoi lavori. AWS Glue Dopo aver acquisito dimestichezza con l'interfaccia utente di Spark, segui le strategie per ottimizzare le prestazioni lavorative di Spark per identificare e ridurre l'impatto delle strozzature sulla base delle tue scoperte.

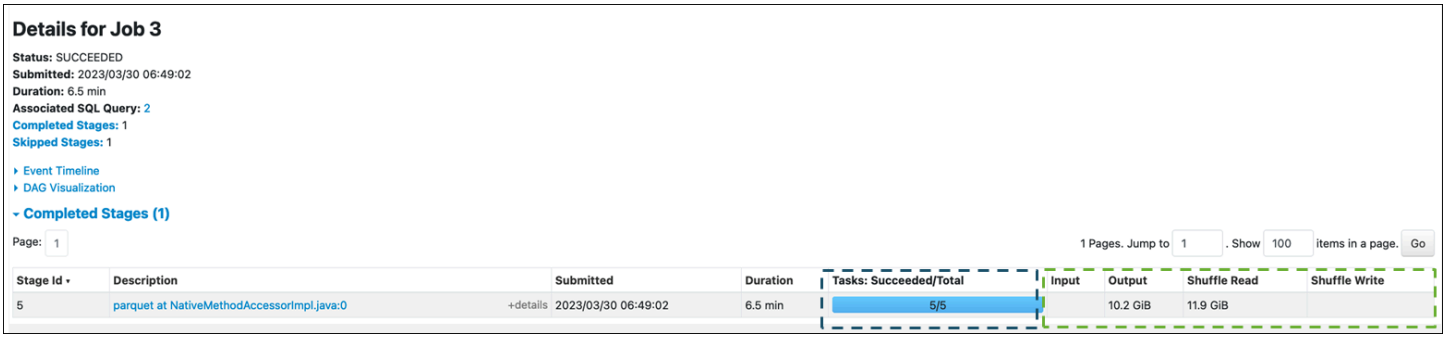
Identifica i punti deboli utilizzando l'interfaccia utente di Spark

Quando apri l'interfaccia utente di Spark, le applicazioni Spark vengono elencate in una tabella. Per impostazione predefinita, il nome dell'app di un AWS Glue lavoro è. `nativespark-<Job Name>-<Job Run ID>` Scegli l'app Spark di destinazione in base al Job Run ID per aprire la scheda Jobs. Le esecuzioni di job incomplete, come le esecuzioni di job in streaming, sono elencate in Mostra applicazioni incomplete.

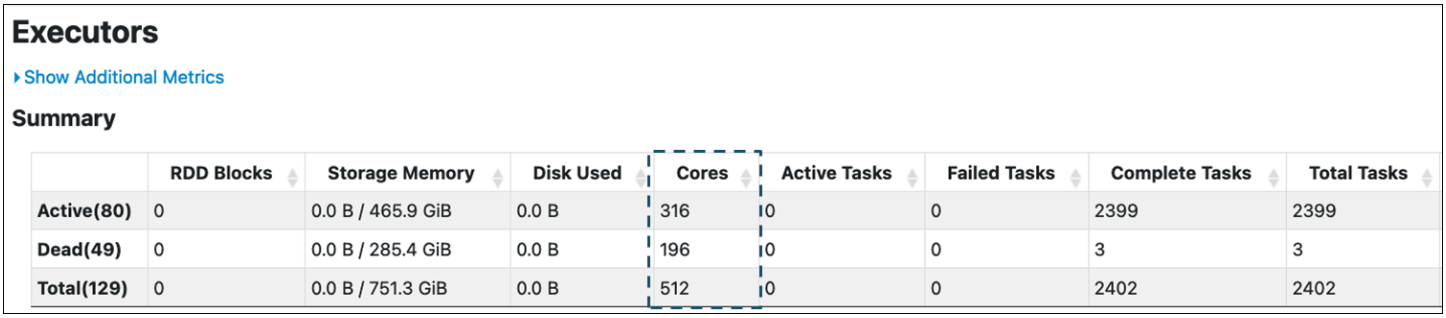
La scheda Lavori mostra un riepilogo di tutti i lavori nell'applicazione Spark. Per determinare eventuali fasi o errori delle attività, controlla il numero totale di attività. Per individuare i punti deboli, ordina scegliendo Durata. Approfondisci i dettagli dei lavori di lunga durata selezionando il link mostrato nella colonna Descrizione.



La pagina Details for Job elenca le fasi. In questa pagina puoi visualizzare informazioni generali come la durata, il numero di attività completate e totali, il numero di input e output e la quantità di letture e scritture casuali.



La scheda Executor mostra in dettaglio la capacità del cluster Spark. Puoi controllare il numero totale di core. Il cluster mostrato nella schermata seguente contiene 316 core attivi e 512 core in totale. Per impostazione predefinita, ogni core può elaborare un task Spark contemporaneamente.



	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks
Active(80)	0	0.0 B / 465.9 GiB	0.0 B	316	0	0	2399	2399
Dead(49)	0	0.0 B / 285.4 GiB	0.0 B	196	0	0	3	3
Total(129)	0	0.0 B / 751.3 GiB	0.0 B	512	0	0	2402	2402

In base al valore 5/5 mostrato nella pagina Details for Job, la fase 5 è la fase più lunga, ma utilizza solo 5 core su 512. Poiché il parallelismo in questa fase è così basso, ma richiede molto tempo, è possibile identificarla come un collo di bottiglia. Per migliorare le prestazioni, è necessario capire perché. Per saperne di più su come riconoscere e ridurre l'impatto dei comuni rallentamenti prestazionali, consulta Strategie per ottimizzare le prestazioni lavorative di Spark.

Strategie per ottimizzare le prestazioni lavorative di Spark

Quando ci si prepara all'ottimizzazione dei parametri, utilizza la seguenti best practice:

- Determinare gli obiettivi di prestazioni prima di iniziare a identificare i problemi.
- Prima di provare di modificare i parametri di ottimizzazione, utilizza i parametri per identificare i problemi.

Per ottenere risultati più coerenti durante l'ottimizzazione di un processo, sviluppa una strategia di base per il lavoro di ottimizzazione.

Strategia di base per l'ottimizzazione delle prestazioni

In genere, l'ottimizzazione delle prestazioni viene eseguita nel seguente flusso di lavoro:

1. Determinazione degli obiettivi delle prestazioni.
2. Misurazione dei parametri.
3. Identificazione dei colli di bottiglia.
4. Riduzione dell'impatto dei colli di bottiglia.
5. Ripeti i passaggi da 2 a 4 fino a raggiungere l'obiettivo prefissato.

Innanzitutto, stabilisci i tuoi obiettivi prestazionali. Ad esempio, uno dei tuoi obiettivi potrebbe essere quello di completare un AWS Glue lavoro entro 3 ore. Dopo aver definito i tuoi obiettivi, misura le metriche delle prestazioni lavorative. Identifica le tendenze nelle metriche e gli ostacoli per raggiungere gli obiettivi. In particolare, identificare i punti deboli è molto importante per la risoluzione dei problemi, il debug e l'ottimizzazione delle prestazioni. Durante l'esecuzione di un'applicazione Spark, Spark registra lo stato e le statistiche di ogni attività nel registro degli eventi Spark.

In AWS Glue, puoi visualizzare le metriche di Spark tramite l'interfaccia utente Web di Spark fornita dal server di cronologia Spark. AWS Glue for Spark jobs può inviare i log degli eventi Spark a una posizione specificata in Amazon S3. AWS Glue fornisce anche un AWS CloudFormation modello di esempio e un Dockerfile per avviare il server di cronologia Spark su un'EC2istanza Amazon o sul computer locale, in modo da poter utilizzare l'interfaccia utente di Spark con i registri degli eventi.

Dopo aver determinato i tuoi obiettivi prestazionali e identificato le metriche per valutarli, puoi iniziare a identificare e correggere i punti deboli utilizzando le strategie illustrate nelle sezioni seguenti.

Pratiche di ottimizzazione per le prestazioni lavorative di Spark

Puoi utilizzare le seguenti strategie per ottimizzare le prestazioni dei job AWS Glue Spark:

- AWS Glue risorse:
  - Scalare la capacità del cluster
  - Usa la versione più recente AWS Glue
- Applicazioni Spark:
  - Ridurre la quantità di scansione dei dati
  - Parallelizza le attività
  - Ottimizza gli shuffles
  - Riduci al minimo i costi di pianificazione
  - Ottimizza le funzioni definite dall'utente

Prima di utilizzare queste strategie, devi avere accesso alle metriche e alla configurazione per il tuo job Spark. Puoi trovare queste informazioni nella documentazione.AWS Glue

Dal punto di vista delle AWS Glue risorse, è possibile ottenere miglioramenti delle prestazioni aggiungendo AWS Glue lavoratori e utilizzando la AWS Glue versione più recente.

Dal punto di vista delle applicazioni Apache Spark, hai accesso a diverse strategie che possono migliorare le prestazioni. Se nel cluster Spark vengono caricati dati non necessari, puoi rimuoverli per ridurre la quantità di dati caricati. Se le risorse del cluster Spark sono sottoutilizzate e l'I/O dei dati è limitato, puoi identificare le attività da parallelizzare. Potresti anche voler ottimizzare le complesse operazioni di trasferimento dei dati, come i join, se richiedono molto tempo. Puoi anche ottimizzare il tuo piano di richieste di lavoro o ridurre la complessità computazionale delle singole attività Spark.

Per applicare in modo efficiente queste strategie, devi identificare quando sono applicabili consultando le tue metriche. Per ulteriori dettagli, consulta ciascuna delle seguenti sezioni. Queste tecniche funzionano non solo per ottimizzare le prestazioni, ma anche per risolvere problemi tipici come gli errori out-of-memory (OOM).

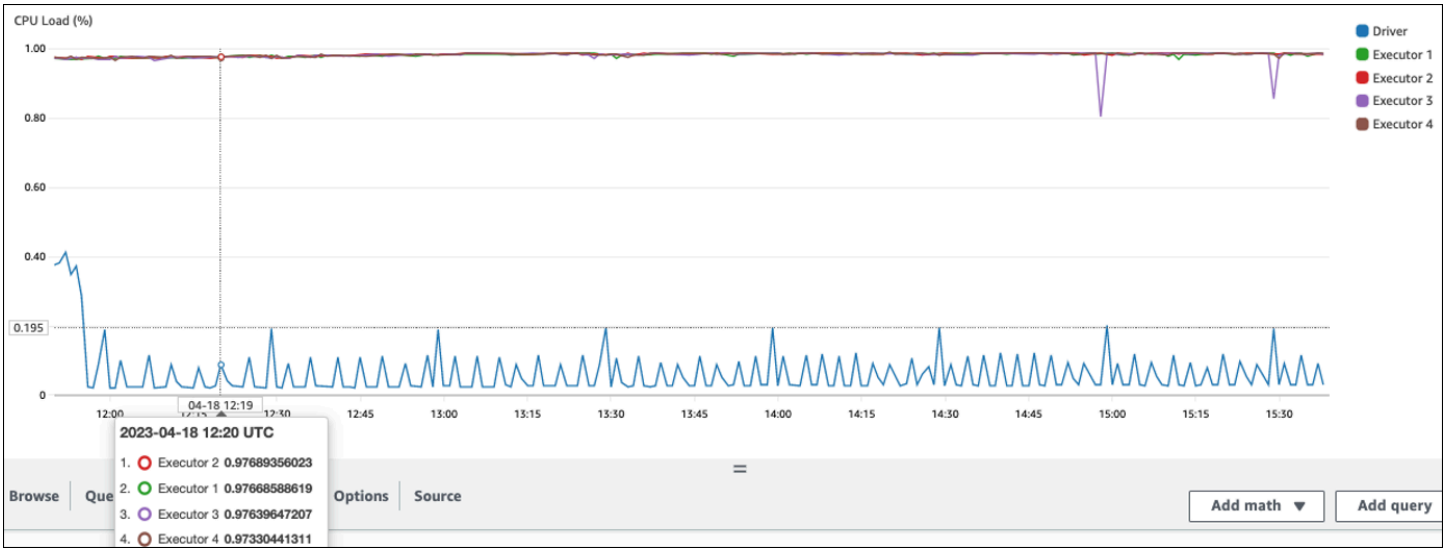
Scalare la capacità del cluster

Se il tuo lavoro richiede troppo tempo, ma gli esecutori consumano risorse sufficienti e Spark sta creando un grande volume di attività relative ai core disponibili, prendi in considerazione la possibilità di scalare la capacità del cluster. Per valutare se ciò è appropriato, utilizza le seguenti metriche.

CloudWatch metriche

- Controllate CPUil carico e l'utilizzo della memoria per determinare se gli esecutori stanno consumando risorse sufficienti.
- Controllate la durata del processo per valutare se il tempo di elaborazione è troppo lungo per soddisfare gli obiettivi prestazionali.

Nell'esempio seguente, quattro executor sono in esecuzione con un CPU carico superiore al 97%, ma l'elaborazione non è stata completata dopo circa tre ore.

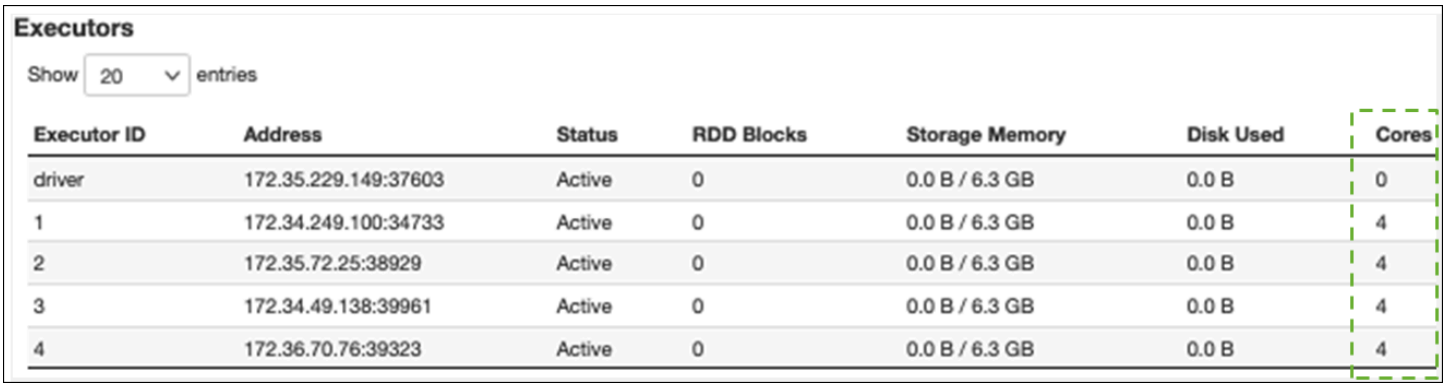


> **Note**
>
> Se CPU il carico è basso, probabilmente non trarrete alcun vantaggio dalla scalabilità della capacità del cluster.

Interfaccia utente di Spark

Nella scheda Job o Stage, puoi vedere il numero di attività per ogni lavoro o fase. Nell'esempio seguente, Spark ha creato `58100` delle attività.

Nella scheda Esecutore, puoi vedere il numero totale di esecutori e attività. Nella schermata seguente, ogni esecutore Spark ha quattro core e può eseguire quattro attività contemporaneamente.



In questo esempio, il numero di attività Spark () `58100`) è molto maggiore delle 16 attività che gli esecutori possono elaborare contemporaneamente (4 esecutori × 4 core).

Se osservi questi sintomi, valuta la possibilità di ridimensionare il cluster. È possibile scalare la capacità del cluster utilizzando le seguenti opzioni:

- Abilita AWS Glue Auto Scaling: Auto Scaling è disponibile per AWS Glue i processi di estrazione, trasformazione e caricamento ETL () e streaming AWS Glue nella versione 3.0 o successiva. AWS Glue aggiunge e rimuove automaticamente i worker dal cluster in base al numero di partizioni in ogni fase o alla velocità con cui vengono generati i microbatch durante l'esecuzione del lavoro.

  Se osservate una situazione in cui il numero di lavoratori non aumenta anche se l'Auto Scaling è abilitato, prendete in considerazione l'aggiunta manuale dei lavoratori. Tuttavia, tieni presente che il ridimensionamento manuale per una fase potrebbe far sì che molti lavoratori rimangano inattivi nelle fasi successive, con costi maggiori senza alcun miglioramento delle prestazioni.

Dopo aver abilitato Auto Scaling, puoi vedere il numero di executor nelle metriche degli executor. CloudWatch Utilizza le seguenti metriche per monitorare la richiesta di esecutori nelle applicazioni Spark:

- `glue.driver.ExecutorAllocationManager.executors.numberAllExecutors`
- `glue.driver.ExecutorAllocationManager.executors.numberMaxNeededExecutors`

Per ulteriori informazioni sui parametri, consulta Monitoraggio AWS Glue tramite i CloudWatch parametri Amazon.

- Scalabilità orizzontale: aumenta il numero di AWS Glue lavoratori: puoi aumentare manualmente il numero di AWS Glue lavoratori. Aggiungi lavoratori solo finché non osservi i lavoratori inattivi. A quel punto, l'aggiunta di più lavoratori aumenterà i costi senza migliorare i risultati. Per ulteriori informazioni, consulta Parallelizzare le attività.
- Scalabilità verticale: utilizza un tipo di worker più grande: puoi modificare manualmente il tipo di istanza dei tuoi AWS Glue lavoratori per utilizzare lavoratori con più core, memoria e storage. I tipi di worker più grandi consentono di scalare verticalmente ed eseguire lavori intensivi di integrazione dei dati, come trasformazioni di dati che richiedono molta memoria, aggregazioni asimmetriche e controlli di rilevamento delle entità che coinvolgono petabyte di dati.

  La scalabilità è utile anche nei casi in cui il driver Spark necessiti di una maggiore capacità, ad esempio perché il piano di richieste di lavoro è piuttosto ampio. Per ulteriori informazioni sui tipi di worker e sulle prestazioni, consulta il post del AWS Big Data Blog Scale your AWS Glue for Apache Spark jobs with new large worker type G.4X e G.8X.

  L'utilizzo di lavoratori di grandi dimensioni può anche ridurre il numero totale di lavoratori necessari, il che aumenta le prestazioni grazie alla riduzione degli spostamenti in operazioni intensive come la partecipazione.

Usa la versione più recente AWS Glue

Ti consigliamo di utilizzare la versione più recente AWS Glue . In ogni versione sono presenti diverse ottimizzazioni e aggiornamenti che potrebbero migliorare automaticamente le prestazioni lavorative. Ad esempio, AWS Glue 4.0 offre le seguenti nuove funzionalità:

- Il nuovo runtime ottimizzato di Apache Spark 3.3.0 — AWS Glue 4.0 si basa sul runtime Apache Spark 3.3.0 e offre miglioramenti prestazionali comparabili a quelli di Spark open source. Il runtime Spark 3.3.0 si basa su molte delle innovazioni di Spark 2.x.

- Connettore Amazon Redshift avanzato: AWS Glue 4.0 e versioni successive forniscono l'integrazione di Amazon Redshift per Apache Spark. L'integrazione si basa su un connettore open source esistente e lo migliora in termini di prestazioni e sicurezza. L'integrazione aiuta le applicazioni a prestazioni fino a 10 volte più veloci. Per ulteriori informazioni, consulta il post di blog sull'integrazione di Amazon Redshift con Apache Spark.
- SIMDesecuzione basata per letture vettoriali con JSON dati CSV e dati: la AWS Glue versione 3.0 e le versioni successive aggiungono lettori ottimizzati che possono velocizzare notevolmente le prestazioni lavorative complessive rispetto ai lettori basati su righe. Per ulteriori informazioni sui CSV dati, consulta Ottimizzare le prestazioni di lettura con un lettore vettoriale. SIMD CSV Per ulteriori informazioni sui JSON dati, consulta Utilizzo del SIMD JSON lettore vettoriale con il formato colonnare Apache Arrow.

Ogni AWS Glue versione includerà aggiornamenti di questo tipo, tra i tanti, tra cui connettori, aggiornamenti di driver e librerie. Per ulteriori informazioni, consulta AWS Glue Versioni e migrazione dei AWS Glue lavori alla AWS Glue versione 4.0.

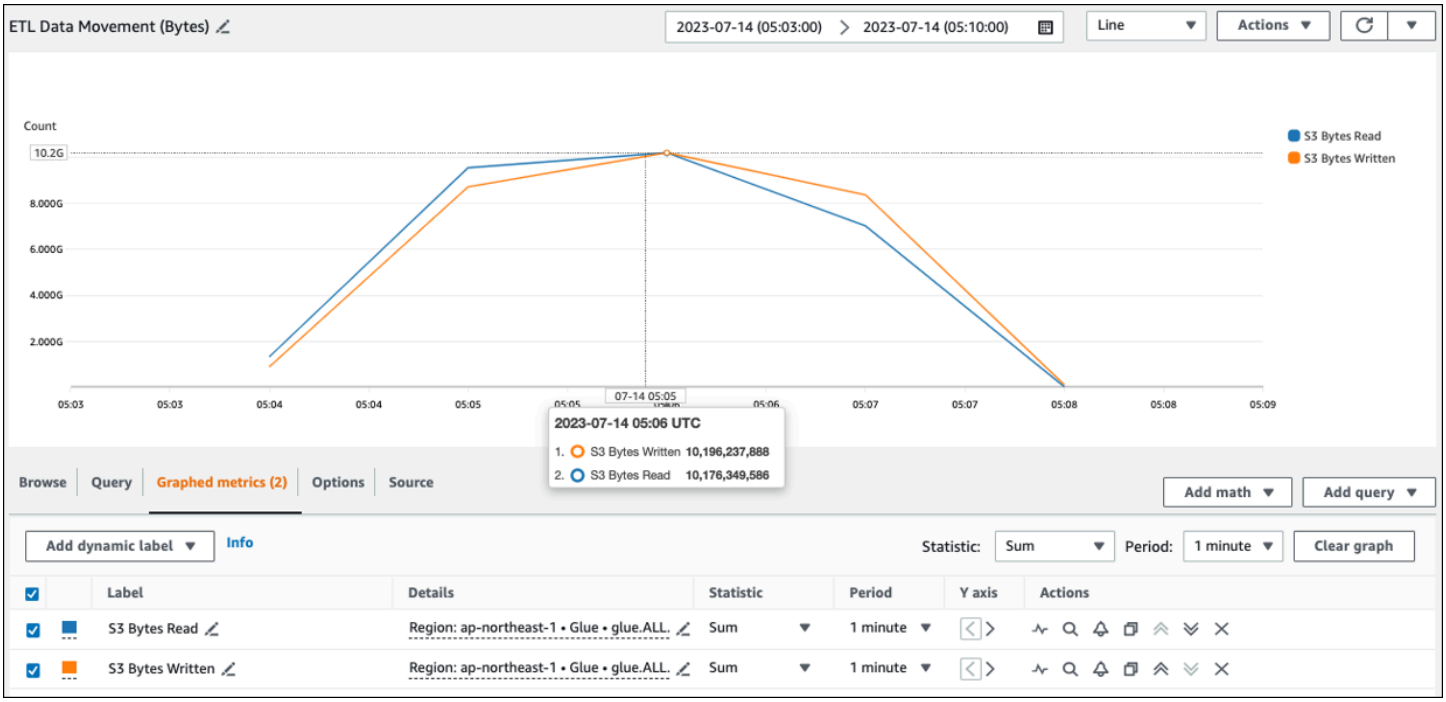
Ridurre la quantità di scansione dei dati

Per iniziare, valuta la possibilità di caricare solo i dati di cui hai bisogno. Puoi migliorare le prestazioni semplicemente riducendo la quantità di dati caricati nel cluster Spark per ogni fonte di dati. Per valutare se questo approccio è appropriato, utilizza le seguenti metriche.

Puoi controllare i byte letti da Amazon S3 CloudWatchnelle metriche e maggiori dettagli nell'interfaccia utente di Spark, come descritto nella sezione Interfaccia utente Spark.

CloudWatch metriche

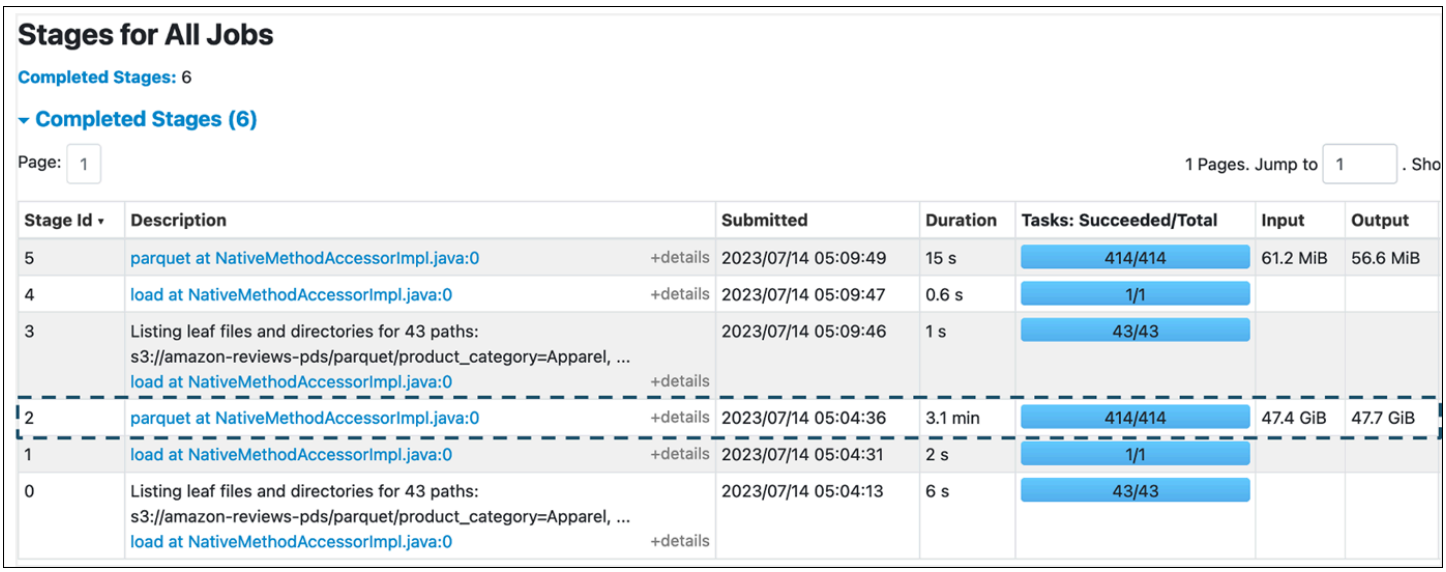
Puoi vedere la dimensione approssimativa di lettura da Amazon S3 ETLin Data Movement (Bytes). Questa metrica mostra il numero di byte letti da Amazon S3 da tutti gli esecutori rispetto al rapporto precedente. Puoi usarlo per monitorare il movimento ETL dei dati da Amazon S3 e confrontare le velocità di lettura con quelle di acquisizione da fonti di dati esterne.



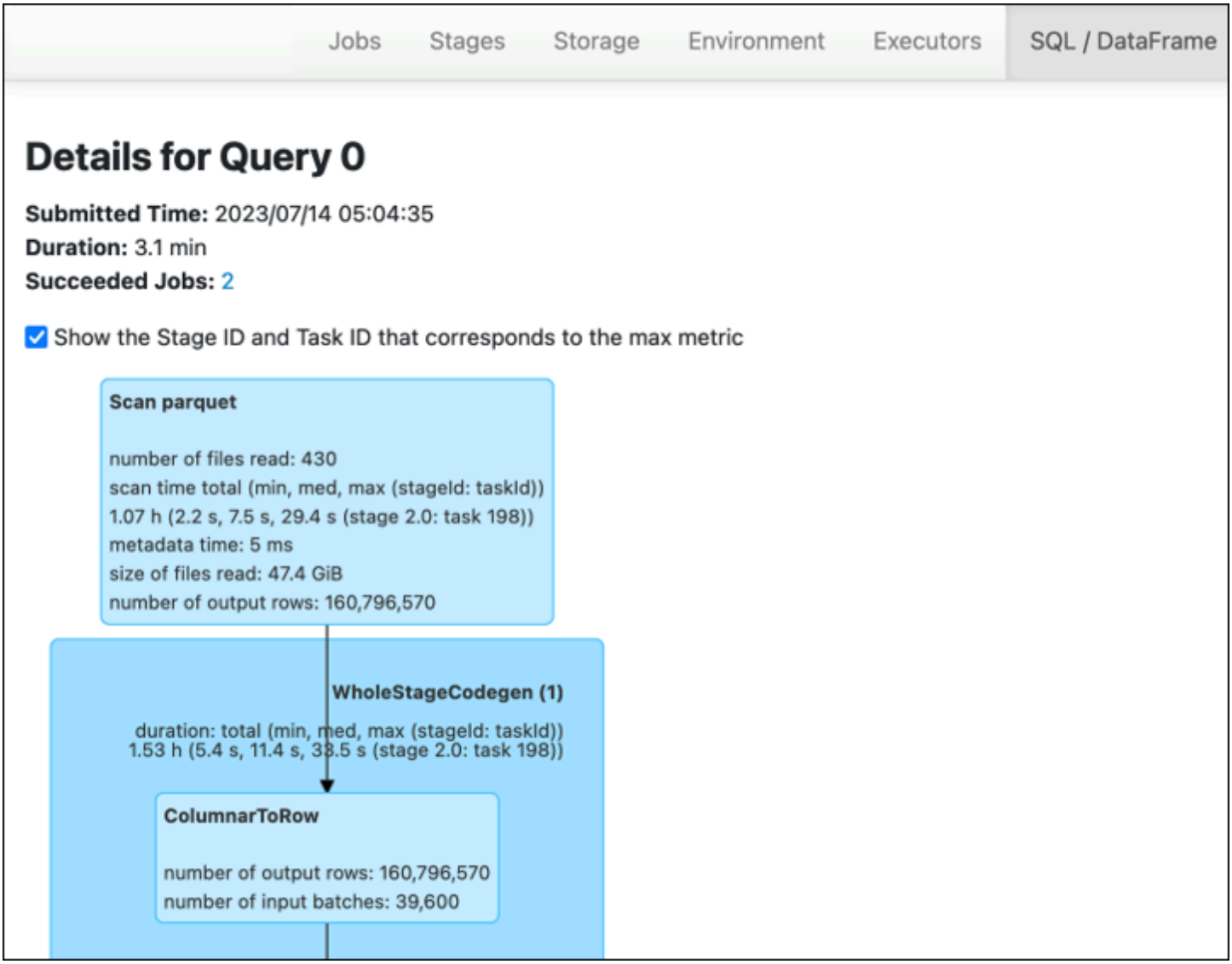
Se osservi un data point S3 Bytes Read più grande del previsto, prendi in considerazione le seguenti soluzioni.

Interfaccia utente di Spark

Nella scheda Stage dell'interfaccia utente AWS Glue di Spark, puoi vedere le dimensioni di input e output. Nell'esempio seguente, lo stage 2 legge 47,4 GiB di input e 47,7 GiB di output, mentre lo stage 5 legge 61,2 MiB di input e 56,6 MiB di output.



Quando utilizzi Spark SQL o DataFrame approcci nel tuo AWS Glue lavoro, la scheda /D mostra ulteriori statistiche su queste fasi. SQL ataFrame In questo caso, la fase 2 mostra il numero di file letti: 430, la dimensione dei file letti: 47,4 GiB e il numero di righe di output: 160.796.570.



Se notate che esiste una differenza sostanziale nelle dimensioni tra i dati che state leggendo e quelli che state utilizzando, provate le seguenti soluzioni.

Amazon S3

Per ridurre la quantità di dati caricati nel tuo lavoro durante la lettura da Amazon S3, considera le dimensioni, la compressione, il formato del file e il layout del file (partizioni) per il tuo set di dati. AWS Glue per Spark i job vengono spesso utilizzati per ETL dati grezzi, ma per un'elaborazione distribuita efficiente è necessario controllare le caratteristiche del formato di origine dei dati.

- Dimensioni del file: consigliamo di mantenere le dimensioni dei file di input e output entro un intervallo moderato (ad esempio, 128 MB). File troppo piccoli e file troppo grandi possono causare problemi.

  Un numero elevato di file di piccole dimensioni causa i seguenti problemi:

  - Elevato carico di I/O di rete su Amazon S3 a causa del sovraccarico necessario per effettuare richieste (`List`ad esempio`Get`, `Head` o) per molti oggetti (rispetto a pochi oggetti che memorizzano la stessa quantità di dati).
  - Intenso carico di I/O e di elaborazione sul driver Spark, che genererà molte partizioni e attività e porterà a un parallelismo eccessivo.

  D'altra parte, se il tipo di file non è divisibile (come gzip) e i file sono troppo grandi, l'applicazione Spark deve attendere il completamento di una singola operazione di lettura dell'intero file.

  Per ridurre l'eccessivo parallelismo che si verifica quando viene creata un'attività di Apache Spark per ogni file di piccole dimensioni, utilizzate il raggruppamento di file per. DynamicFrames Questo approccio riduce le possibilità di un'OOMeccezione dal driver Spark. Per configurare il raggruppamento dei file, impostate i parametri `groupFiles` and`groupSize`. L'esempio di codice seguente utilizza lo script AWS Glue DynamicFrame API in uno ETL script con questi parametri.

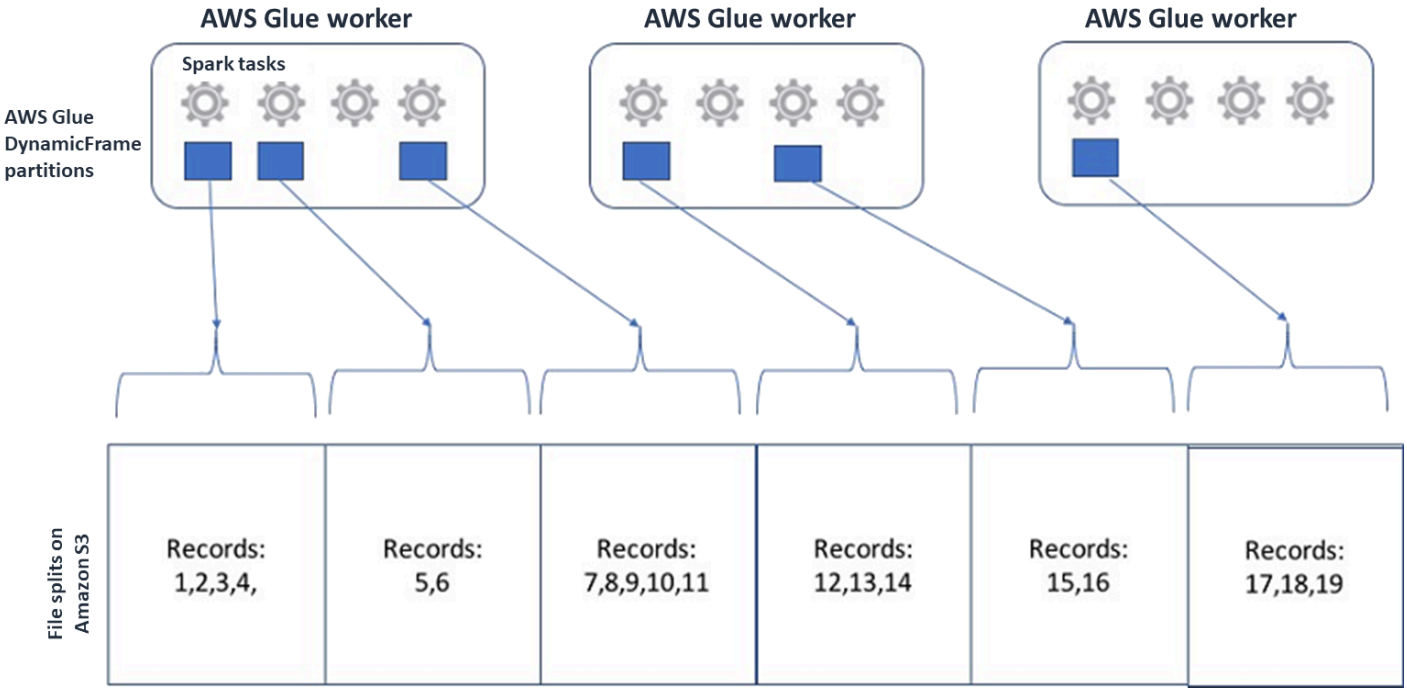
  ```
  dyf = glueContext.create_dynamic_frame_from_options("s3",
      {'paths': ["s3://input-s3-path/"],
      'recurse':True,
      'groupFiles': 'inPartition',
      'groupSize': '1048576'},
      format="json")
  ```

- Compressione: se i tuoi oggetti S3 misurano centinaia di megabyte, valuta la possibilità di comprimerli. Esistono vari formati di compressione, che possono essere generalmente classificati in due tipi:

  - I formati di compressione non divisibili come gzip richiedono che l'intero file venga decompresso da un operatore.
  - I formati di compressione divisibili, come bzip2 o LZO (indicizzato), consentono la decompressione parziale di un file, che può essere parallelizzata.

  Per Spark (e altri motori di elaborazione distribuita comuni), suddividerai il file di dati sorgente in blocchi che il tuo motore può elaborare in parallelo. Queste unità vengono spesso chiamate split. Dopo che i dati sono in un formato divisibile, i AWS Glue lettori ottimizzati possono recuperare le

suddivisioni da un oggetto S3 offrendo la `Range` possibilità di recuperare solo blocchi specifici. `GetObject` API Considerate il diagramma seguente per vedere come funzionerebbe in pratica.



I dati compressi possono velocizzare notevolmente l'applicazione, purché i file abbiano una dimensione ottimale o siano divisibili. Le dimensioni dei dati più piccole riducono i dati scansionati da Amazon S3 e il traffico di rete da Amazon S3 al cluster Spark. D'altra parte, CPU è necessario fare di più per comprimere e decomprimere i dati. La quantità di calcolo richiesta varia in base al rapporto di compressione dell'algoritmo di compressione. Considerate questo compromesso quando scegliete il formato di compressione divisibile.

> **Note**
>
> Sebbene i file gzip non siano generalmente divisibili, puoi comprimere singoli blocchi di parquet con gzip e quei blocchi possono essere parallelizzati.

- Formato di file: utilizza un formato colonnare. Apache Parquet e Apache ORC sono i formati di dati colonnari più diffusi. Parquet e ORC archivia i dati in modo efficiente utilizzando la compressione, la codifica e la compressione basate su colonne in base al tipo di dati. Per ulteriori informazioni sulle codifiche Parquet, consulta Definizioni di codifica Parquet. I file Parquet sono anche divisibili.

I formati colonnari raggruppano i valori per colonna e li memorizzano in blocchi. Quando usi i formati colonnari, puoi saltare i blocchi di dati che corrispondono a colonne che non intendi utilizzare. Le applicazioni Spark possono recuperare solo le colonne di cui hai bisogno. In genere, rapporti di compressione migliori o il salto di blocchi di dati implicano la lettura di un numero inferiore di byte da Amazon S3, con conseguente miglioramento delle prestazioni. Entrambi i formati supportano anche i seguenti approcci pushdown per ridurre l'I/O:

- Projection pushdown: il proiection pushdown è una tecnica per recuperare solo le colonne specificate nell'applicazione. È possibile specificare le colonne nell'applicazione Spark, come mostrato negli esempi seguenti:
  - DataFrame esempio: `df.select("star_rating")`
  - SQLEsempio di Spark: `spark.sql("select start_rating from <table>")`
- Predicate pushdown: il predicate pushdown è una tecnica per l'elaborazione efficiente di clausole e clausole. `WHERE GROUP BY` Entrambi i formati hanno blocchi di dati che rappresentano i valori delle colonne. Ogni blocco contiene le statistiche relative al blocco, ad esempio i valori massimi e minimi. Spark può utilizzare queste statistiche per determinare se il blocco deve essere letto o ignorato a seconda del valore del filtro utilizzato nell'applicazione. Per utilizzare questa funzione, aggiungi altri filtri nelle condizioni, come mostrato negli esempi seguenti:
  - DataFrame esempio: `df.select("star_rating").filter("star_rating < 2")`
  - SQLEsempio di Spark: `spark.sql("select * from <table> where star_rating < 2")`

- Layout dei file: archiviando i dati S3 su oggetti in percorsi diversi in base a come verranno utilizzati i dati, puoi recuperare in modo efficiente i dati pertinenti. Per ulteriori informazioni, consulta Organizzazione degli oggetti utilizzando i prefissi nella documentazione di Amazon S3. AWS Glue supporta la memorizzazione di chiavi e valori nei prefissi Amazon S3 nel formato`key=value`, partizionando i dati in base al percorso Amazon S3. Partizionando i dati, puoi limitare la quantità di dati scansionati da ciascuna applicazione di analisi downstream, migliorando le prestazioni e riducendo i costi. Per ulteriori informazioni, consulta Gestione delle partizioni per l'output in. ETL AWS Glue

  Il partizionamento divide la tabella in diverse parti e mantiene i dati correlati in file raggruppati in base ai valori delle colonne come anno, mese e giorno, come illustrato nell'esempio seguente.

```
# Partitioning by /YYYY/MM/DD
s3://<YourBucket>/year=2023/month=03/day=31/0000.gz
s3://<YourBucket>/year=2023/month=03/day=01/0000.gz
```

```
s3://<YourBucket>/year=2023/month=03/day=02/0000.gz
s3://<YourBucket>/year=2023/month=03/day=03/0000.gz
...
```

È possibile definire le partizioni per il set di dati modellandolo con una tabella in. AWS Glue Data CatalogÈ quindi possibile limitare la quantità di dati scansionati utilizzando la rimozione delle partizioni come segue:

- Per AWS Glue DynamicFrame, set `push_down_predicate` (`ocatalogPartitionPredicate`).

```
dyf = Glue_context.create_dynamic_frame.from_catalog(
    database=src_database_name,
    table_name=src_table_name,
    push_down_predicate = "year='2023' and month ='03'",
)
```

- Per Spark DataFrame, imposta un percorso fisso per eliminare le partizioni.

```
df = spark.read.format("json").load("s3://<YourBucket>/year=2023/month=03/*/*.gz")
```

- Per SparkSQL, puoi impostare la clausola where per eliminare le partizioni dal Data Catalog.

```
df = spark.sql("SELECT * FROM <Table> WHERE year= '2023' and month = '03'")
```

- Per partizionare i dati partitionKeysin base alla data con cui scrivi i dati AWS Glue, devi inserire DynamicFrame o partitionBy() inserire le informazioni sulla data nelle DataFrame colonne nel modo seguente.
  - DynamicFrame

```
glue_context.write_dynamic_frame_from_options(
    frame= dyf, connection_type='s3',format='parquet'
    connection_options= {
        'partitionKeys': ["year", "month", "day"],
        'path': 's3://<YourBucket>/<Prefix>/'
    }
)
```

  - DataFrame

```
df.write.mode('append')\
```

```
    .partitionBy('year','month','day')\
    .parquet('s3://<YourBucket>/<Prefix>/')
```

Ciò può migliorare le prestazioni degli utenti dei dati di output.

Se non avete accesso per modificare la pipeline che crea il set di dati di input, il partizionamento non è un'opzione. Invece, puoi escludere percorsi S3 non necessari utilizzando modelli a glob. Imposta le esclusioni durante la lettura. DynamicFrame Ad esempio, il codice seguente esclude i giorni nei mesi da 01 a 09, nell'anno 2023.

```
dyf = glueContext.create_dynamic_frame.from_catalog(
    database=db,
    table_name=table,
    additional_options = { "exclusions":"[\"**year=2023/month=0[1-9]/**\"]" },
    transformation_ctx='dyf'
)
```

Puoi anche impostare esclusioni nelle proprietà della tabella nel Catalogo dati:

- Chiave: `exclusions`
- Valore: `["**year=2023/month=0[1-9]/**"]`

- Troppe partizioni Amazon S3: evita di partizionare i dati Amazon S3 su colonne che contengono un'ampia gamma di valori, ad esempio una colonna ID con migliaia di valori. Ciò può aumentare notevolmente il numero di partizioni nel bucket, poiché il numero di partizioni possibili è il prodotto di tutti i campi in base ai quali hai partizionato. Troppe partizioni potrebbero causare quanto segue:
  - Maggiore latenza per il recupero dei metadati delle partizioni dal Data Catalog
  - Aumento del numero di file di piccole dimensioni, che richiede più API richieste Amazon S3 (`ListGet`, e) `Head`

  Ad esempio, quando si imposta un tipo di data in `partitionBy` or`partitionKeys`, il partizionamento a livello di data `yyyy/mm/dd` è utile per molti casi d'uso. Tuttavia, `yyyy/mm/dd/<ID>` potrebbe generare così tante partizioni da influire negativamente sulle prestazioni complessive.

  D'altra parte, alcuni casi d'uso, come le applicazioni di elaborazione in tempo reale, richiedono molte partizioni come. `yyyy/mm/dd/hh` Se il tuo caso d'uso richiede partizioni sostanziali, prendi in considerazione l'utilizzo di indici di AWS Glue partizione per ridurre la latenza per il recupero dei metadati delle partizioni dal Data Catalog.

Database e JDBC

Per ridurre la scansione dei dati durante il recupero di informazioni da un database, è possibile specificare un `where` predicato (o una clausola) in una query. SQL I database che non forniscono un'SQLinterfaccia forniranno il proprio meccanismo di interrogazione o filtraggio.

Quando utilizzate le connessioni Java Database Connectivity (JDBC), fornite una query di selezione con la `where` clausola per i seguenti parametri:

- Per DynamicFrame, utilizzate l'sampleQueryopzione. Durante l'utilizzo`create_dynamic_frame.from_catalog`, configura l'`additional_options`argomento come segue.

```
query = "SELECT * FROM <TableName> where id = 'XX' AND"
datasource0 = glueContext.create_dynamic_frame.from_catalog(
  database = db,
  table_name = table,
  additional_options={
      "sampleQuery": query,
      "hashexpression": key,
      "hashpartitions": 10,
      "enablePartitioningForSampleQuery": True
    },
  transformation_ctx = "datasource0"
)
```

  Quando`using create_dynamic_frame.from_options`, configura l'`connection_options`argomento come segue.

```
query = "SELECT * FROM <TableName> where id = 'XX' AND"
datasource0 = glueContext.create_dynamic_frame.from_options(
    connection_type = connection,
    connection_options={
        "url": url,
        "user": user,
        "password": password,
        "dbtable": table,
        "sampleQuery": query,
        "hashexpression": key,
        "hashpartitions": 10,
        "enablePartitioningForSampleQuery": True
```

```
    }
)
```

- Per DataFrame, usa l'opzione di interrogazione.

```
query = "SELECT * FROM <TableName> where id = 'XX'"
jdbcDF = spark.read \
.format('jdbc') \
.option('url', url) \
.option('user', user) \
.option('password', pwd) \
.option('query', query) \
.load()
```

- Per Amazon Redshift, usa la AWS Glue versione 4.0 o successiva per sfruttare il supporto pushdown nel connettore Amazon Redshift Spark.

```
dyf = glueContext.create_dynamic_frame.from_catalog(
    database = "redshift-dc-database-name",
    table_name = "redshift-table-name",
    redshift_tmp_dir = args["temp-s3-dir"],
    additional_options = {"aws_iam_role": "arn:aws:iam::role-account-id:role/rs-role-name"}
)
```

- Per altri database, consulta la documentazione relativa a quel database.

AWS Glue opzioni

- Per evitare una scansione completa per tutte le esecuzioni continue dei lavori ed elaborare solo i dati che non erano presenti durante l'ultima esecuzione del processo, abilita i segnalibri dei lavori.
- Per limitare la quantità di dati di input da elaborare, abilita l'esecuzione limitata con i segnalibri dei lavori. Questo aiuta a ridurre la quantità di dati scansionati per ogni esecuzione di lavoro.

Parallelizza le attività

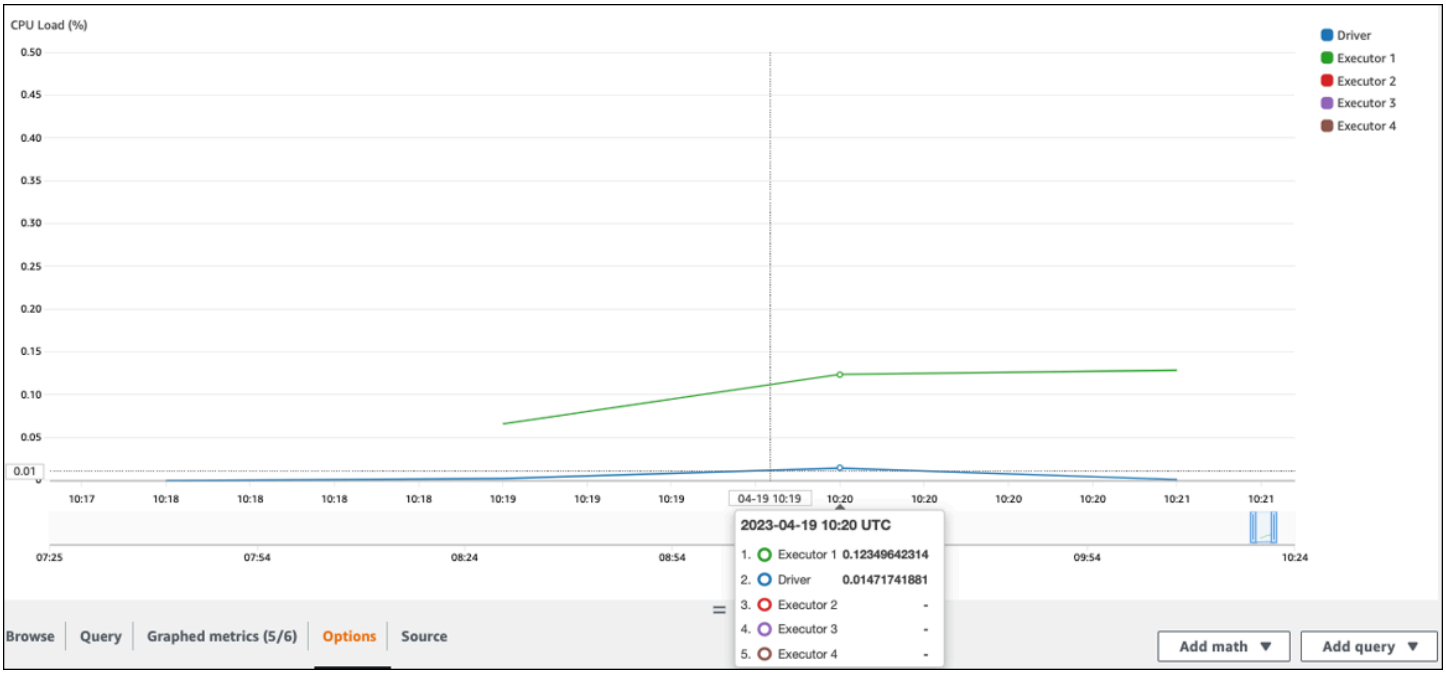
Per ottimizzare le prestazioni, è importante parallelizzare le attività per i caricamenti e le trasformazioni dei dati. Come abbiamo discusso in Argomenti chiave di Apache Spark, il numero di partizioni resilienti di set di dati distribuiti (RDD) è importante, perché determina il grado di

parallelismo. Ogni attività creata da Spark corrisponde a una partizione su base 1:1. RDD Per ottenere le migliori prestazioni, devi capire come viene determinato il numero di RDD partizioni e come tale numero viene ottimizzato.

Se il parallelismo non è sufficiente, i seguenti sintomi verranno registrati nelle CloudWatchmetriche e nell'interfaccia utente di Spark.

CloudWatch metriche

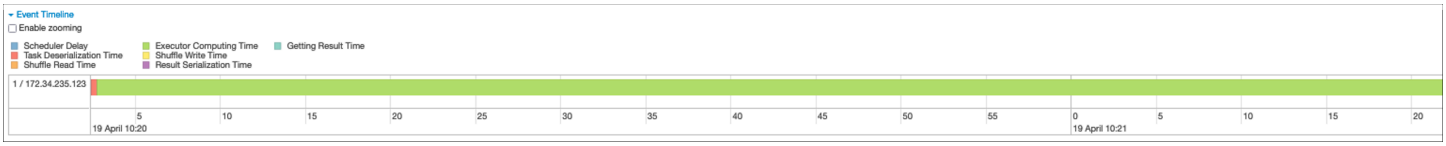
Controlla il CPUcarico e l'utilizzo della memoria. Se alcuni esecutori non eseguono l'elaborazione durante una fase del lavoro, è opportuno migliorare il parallelismo. In questo caso, durante il periodo di tempo visualizzato, l'Executor 1 stava eseguendo un'operazione, ma gli esecutori rimanenti (2, 3 e 4) no. Si può dedurre che a quegli esecutori non sono stati assegnati compiti dal driver Spark.



Interfaccia utente di Spark

Nella scheda Stage dell'interfaccia utente di Spark, puoi vedere il numero di attività in una fase. In questo caso, Spark ha eseguito una sola operazione.

Inoltre, la cronologia degli eventi mostra l'Executor 1 che elabora un'operazione. Ciò significa che il lavoro in questa fase è stato eseguito interamente su un esecutore, mentre gli altri erano inattivi.



Se osservi questi sintomi, prova le seguenti soluzioni per ogni fonte di dati.

Parallelizza il caricamento dei dati da Amazon S3

Per parallelizzare i caricamenti di dati da Amazon S3, controlla innanzitutto il numero predefinito di partizioni. Puoi quindi determinare manualmente il numero di partizioni di destinazione, ma assicurati di evitare di avere troppe partizioni.

Determina il numero predefinito di partizioni

Per Amazon S3, il numero iniziale di RDD partizioni Spark (ognuna delle quali corrisponde a un'attività Spark) è determinato dalle caratteristiche del set di dati Amazon S3 (ad esempio, formato, compressione e dimensione). Quando crei uno AWS Glue DynamicFrame o uno Spark DataFrame da CSV oggetti archiviati in Amazon S3, il numero iniziale RDD di partizioni `NumPartitions` () può essere calcolato approssimativamente come segue:

- Dimensione dell'oggetto **<=** 64 MB: `NumPartitions = Number of Objects`
- Dimensione dell'oggetto **>** 64 MB: `NumPartitions = Total Object Size / 64 MB`
- Indivisibile (gzip): `NumPartitions = Number of Objects`

Come discusso nella sezione Ridurre la quantità di scansione dei dati, Spark divide oggetti S3 di grandi dimensioni in suddivisioni che possono essere elaborate in parallelo. Quando l'oggetto è più grande della dimensione divisa, Spark lo divide e crea una RDD partizione (e un'attività) per ogni divisione. La dimensione suddivisa di Spark si basa sul formato dei dati e sull'ambiente di runtime, ma questa è un'approssimazione iniziale ragionevole. Alcuni oggetti vengono compressi utilizzando formati di compressione non divisibili come gzip, quindi Spark non può dividerli.

Il `NumPartitions` valore può variare in base al formato dei dati, alla compressione, alla AWS Glue versione, al numero di worker e alla configurazione di AWS Glue Spark.

Ad esempio, quando carichi un singolo `csv.gz` oggetto da 10 GB usando Spark DataFrame, il driver Spark creerà solo una RDD partizione (`NumPartitions=1`) perché gzip non è divisibile. Ciò comporta un carico pesante su un particolare esecutore Spark e nessun compito viene assegnato agli esecutori rimanenti, come descritto nella figura seguente.

Controlla il numero effettivo di attività (`NumPartitions`) per lo stage nella scheda Spark Web UI Stage oppure esegui il codice per verificare il `df.rdd.getNumPartitions()` parallelismo.

Quando trovi un file gzip da 10 GB, esamina se il sistema che lo genera è in grado di generarlo in un formato divisibile. Se questa non è un'opzione, potrebbe essere necessario scalare la capacità del cluster per elaborare il file. Per eseguire le trasformazioni in modo efficiente sui dati caricati, sarà necessario ribilanciare le trasformazioni RDD tra i worker del cluster utilizzando la ripartizione.

Determinare manualmente il numero di partizioni desiderato

A seconda delle proprietà dei dati e dell'implementazione di alcune funzionalità da parte di Spark, potresti ritrovarti con un `NumPartitions` valore basso anche se il lavoro sottostante può ancora essere parallelizzato. Se `NumPartitions` è troppo piccolo, esegui `df.repartition(N)` per aumentare il numero di partizioni in modo che l'elaborazione possa essere distribuita su più esecutori Spark.

In questo caso, l'esecuzione `df.repartition(100)` passerà `NumPartitions` da 1 a 100, creando 100 partizioni di dati, ognuna con un'attività che può essere assegnata agli altri esecutori.

L'operazione `repartition(N)` divide equamente tutti i dati (10 GB/100 partizioni = 100 MB/partizione), evitando la distorsione dei dati su determinate partizioni.

> **Note**
>
> Quando si esegue un'operazione di shuffle come quella eseguita, il numero di partizioni `join` viene aumentato o diminuito dinamicamente in base al valore di o. `spark.sql.shuffle.partitions` `spark.default.parallelism` Ciò facilita uno scambio di dati più efficiente tra gli esecutori Spark. Per ulteriori informazioni, consulta la documentazione di Spark.

Il vostro obiettivo nel determinare il numero di partizioni previsto è quello di massimizzare l'uso dei lavoratori assegnati. AWS Glue Il numero di AWS Glue lavoratori e il numero di attività Spark sono correlati dal numero di. vCPUs Spark supporta un task per ogni v CPU core. Nella AWS Glue versione 3.0 o successiva, puoi calcolare un numero target di partizioni utilizzando la seguente formula.

```
# Calculate NumPartitions by WorkerType
```

```
numExecutors = (NumberOfWorkers - 1)
numSlotsPerExecutor =
  4 if WorkerType is G.1X
  8 if WorkerType is G.2X
  16 if WorkerType is G.4X
  32 if WorkerType is G.8X
NumPartitions = numSlotsPerExecutor * numExecutors

# Example: Glue 4.0 / G.1X / 10 Workers
numExecutors = ( 10 - 1 ) = 9  # 1 Worker reserved on Spark Driver
numSlotsPerExecutor       = 4  # G.1X has 4 vCpu core ( Glue 3.0 or later )
NumPartitions = 9  * 4    = 36
```

In questo esempio, ogni worker G.1X fornisce quattro v CPU core a un executor Spark (). `spark.executor.cores = 4` Spark supporta un task per ogni v CPU Core, quindi gli executor G.1X Spark possono eseguire quattro attività contemporaneamente (). `numSlotPerExecutor` Questo numero di partizioni sfrutta appieno il cluster se le attività richiedono lo stesso periodo di tempo. Tuttavia, alcune attività richiederanno più tempo di altre, creando core inattivi. In tal caso, valuta la possibilità di moltiplicare `numPartitions` per 2 o 3 per suddividere e pianificare in modo efficiente le attività più difficili.

Troppe partizioni

Un numero eccessivo di partizioni crea un numero eccessivo di attività. Ciò causa un carico pesante sul driver Spark a causa del sovraccarico legato all'elaborazione distribuita, come le attività di gestione e lo scambio di dati tra gli esecutori Spark.

Se il numero di partizioni del lavoro è notevolmente superiore al numero di partizioni previsto, valuta la possibilità di ridurre il numero di partizioni. È possibile ridurre le partizioni utilizzando le seguenti opzioni:

- Se le dimensioni dei file sono molto piccole, usa AWS Glue groupFiles. È possibile ridurre l'eccessivo parallelismo derivante dall'avvio di un'attività di Apache Spark per elaborare ogni file.
- Si usa per unire le `coalesce(N)` partizioni. Si tratta di un processo a basso costo. Quando si riduce il numero di partizioni, `coalesce(N)` è preferibile rispetto a prima`repartition(N)`, perché `repartition(N)` esegue lo shuffle per distribuire equamente la quantità di record in ogni partizione. Ciò aumenta i costi e il sovraccarico di gestione.
- Usa Spark 3.x Adaptive Query Execution. Come discusso nella sezione Argomenti chiave di Apache Spark, Adaptive Query Execution fornisce una funzione per unire automaticamente il

numero di partizioni. È possibile utilizzare questo approccio quando non è possibile conoscere il numero di partizioni finché non si esegue l'esecuzione.

Parallelizza il caricamento dei dati da JDBC

Il numero di RDD partizioni Spark è determinato dalla configurazione. Nota che per impostazione predefinita viene eseguita una sola operazione per scansionare un intero set di dati di origine tramite una query. `SELECT`

AWS Glue DynamicFrames Sia Spark che Spark DataFrames supportano il caricamento parallelizzato JDBC dei dati su più attività. Questa operazione viene eseguita utilizzando `where` i predicati per suddividere una `SELECT` query in più query. Per parallelizzare le letture daJDBC, configura le seguenti opzioni:

- Per AWS Glue DynamicFrame, set `hashfield` (o and. `hashexpression`) `hashpartition` Per ulteriori informazioni, consulta Leggere da JDBC tabelle in parallelo.

```
connection_mysql8_options = {
    "url": "jdbc:mysql://XXXXXXXXXX.XXXXXXX.us-east-1.rds.amazonaws.com:3306/test",
    "dbtable": "medicare_tb",
    "user": "test",
    "password": "XXXXXXXXX",
    "hashexpression":"id",
    "hashpartitions":"10"
}
datasource0 = glueContext.create_dynamic_frame.from_options(
    'mysql',
    connection_options=connection_mysql8_options,
    transformation_ctx= "datasource0"
)
```

- Per Spark DataFrame, set`numPartitions`, `partitionColumn`lower`Bound`, e`upperBound`. Per saperne di più, consulta JDBC To Other Databases.

```
df = spark.read \
    .format("jdbc") \
    .option("url", "jdbc:mysql://XXXXXXXXXX.XXXXXXX.us-east-1.rds.amazonaws.com:3306/
test") \
    .option("dbtable", "medicare_tb") \
    .option("user", "test") \
```

```
    .option("password", "XXXXXXXXXX") \
    .option("partitionColumn", "id") \
    .option("numPartitions", "10") \
    .option("lowerBound", "0") \
    .option("upperBound", "1141455") \
    .load()

df.write.format("json").save("s3://bucket_name/Tests/sparkjdbc/with_parallel/")
```

Parallelizza il caricamento dei dati da DynamoDB quando usi il connettore ETL

Il numero di RDD partizioni Spark è determinato dal parametro. `dynamodb.splits` Per parallelizzare le letture da Amazon DynamoDB, configura le seguenti opzioni:

- Aumenta il valore di. `dynamodb.splits`
- Ottimizza il parametro seguendo la formula spiegata in Tipi di connessione e opzioni per ETL in AWS Glue for Spark.

Parallelizza il caricamento dei dati da Kinesis Data Streams

Il numero di RDD partizioni Spark è determinato dal numero di shard nel flusso di dati Amazon Kinesis Data Streams di origine. Se hai solo pochi shard nel tuo flusso di dati, ci saranno solo alcune attività Spark. Ciò può comportare un basso parallelismo nei processi a valle. Per parallelizzare le letture da Kinesis Data Streams, configura le seguenti opzioni:

- Aumenta il numero di shard per ottenere un maggiore parallelismo durante il caricamento dei dati da Kinesis Data Streams.
- Se la logica del microbatch è abbastanza complessa, valuta la possibilità di ripartizionare i dati all'inizio del batch, dopo aver eliminato le colonne non necessarie.

Per ulteriori informazioni, consulta Best practice per ottimizzare costi e prestazioni per i lavori di streaming. AWS Glue ETL

Parallelizza le attività dopo il caricamento dei dati

Per parallelizzare le attività dopo il caricamento dei dati, aumenta il numero di RDD partizioni utilizzando le seguenti opzioni:

- Ripartiziona i dati per generare un numero maggiore di partizioni, soprattutto subito dopo il caricamento iniziale se non è possibile parallelizzare il carico stesso.

  Chiama su `repartition()` DynamicFrame o DataFrame, specificando il numero di partizioni. Una buona regola empirica è due o tre volte il numero di core disponibili.

  Tuttavia, quando si scrive una tabella partizionata, ciò può portare a un'esplosione di file (ogni partizione può potenzialmente generare un file in ogni partizione della tabella). Per evitare ciò, puoi ripartizionare il file per colonna. DataFrame Questo utilizza le colonne di partizione della tabella in modo che i dati siano organizzati prima della scrittura. È possibile specificare un numero maggiore di partizioni senza inserire file di piccole dimensioni nelle partizioni della tabella. Tuttavia, fate attenzione a evitare la distorsione dei dati, in cui alcuni valori di partizione finiscono con la maggior parte dei dati e ritardano il completamento dell'operazione.
- In caso di mescolamenti, aumentate il valore. `spark.sql.shuffle.partitions` Questo può anche aiutare a risolvere eventuali problemi di memoria durante lo shuffling.

  Quando hai più di 2.001 partizioni shuffle, Spark utilizza un formato di memoria compressa. Se hai un numero vicino a quello, potresti voler impostare il `spark.sql.shuffle.paritions` valore oltre tale limite per ottenere una rappresentazione più efficiente.

Ottimizza gli shuffles

Alcune operazioni, come `join()` `egroupByKey()`, richiedono che Spark esegua uno shuffle. Lo shuffle è il meccanismo di Spark per ridistribuire i dati in modo che siano raggruppati in modo diverso tra le partizioni. RDD Lo shuffling può aiutare a rimediare ai problemi di prestazioni. Tuttavia, poiché lo shuffling in genere implica la copia dei dati tra gli esecutori Spark, lo shuffle è un'operazione complessa e costosa. Ad esempio, lo shuffle genera i seguenti costi:

- I/O del disco:
  - Genera un gran numero di file intermedi su disco.
- I/O di rete:
  - Richiede molte connessioni di rete (Numero di connessioni =`Mapper × Reducer`).
  - Poiché i record vengono aggregati in nuove RDD partizioni che potrebbero essere ospitate su un altro esecutore Spark, una parte sostanziale del set di dati potrebbe spostarsi tra gli esecutori Spark sulla rete.
- CPUe carico di memoria:

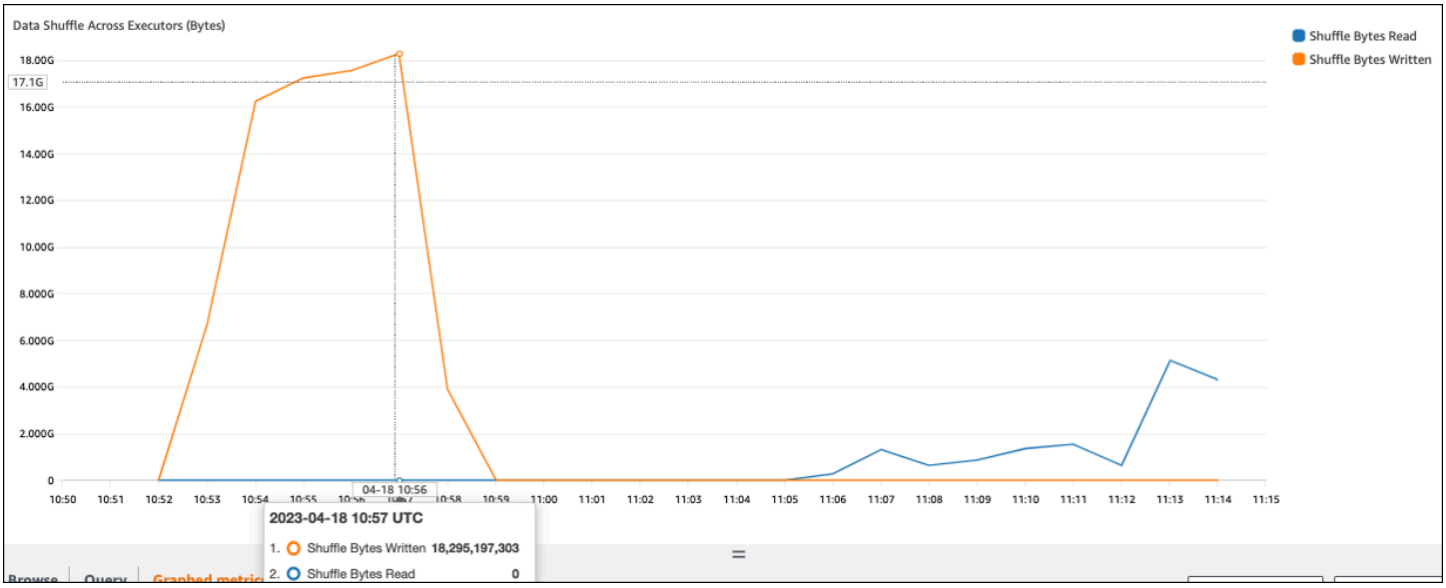
- Ordina i valori e unisce i set di dati. Queste operazioni sono pianificate sull'esecutore, con un carico pesante sull'esecutore.

Shuffle è uno dei fattori più importanti del peggioramento delle prestazioni dell'applicazione Spark. Durante la memorizzazione dei dati intermedi, può esaurire lo spazio sul disco locale dell'esecutore, causando il fallimento del job Spark.

Puoi valutare le tue prestazioni di shuffle nelle metriche e nell' CloudWatch interfaccia utente di Spark.

CloudWatch metriche

Se il valore Shuffle Bytes Written è elevato rispetto a Shuffle Bytes Read, il tuo job Spark potrebbe utilizzare operazioni di shuffle come o. `join()` `groupByKey()`



Interfaccia utente di Spark

Nella scheda Stage dell'interfaccia utente Spark, puoi controllare i valori di Shuffle Read Size/ Records. Puoi vederlo anche nella scheda Executors.

Nella schermata seguente, ogni executor scambia circa 18,6 GB/4020.000 record con il processo shuffle, per una dimensione totale di lettura shuffle di circa 75 GB).

La colonna Shuffle Spill (Disk) mostra che una grande quantità di dati trasferisce memoria su disco, il che potrebbe causare un disco pieno o un problema di prestazioni.

Aggregated Metrics by Executor

Executor ID ▲	Address	Shuffle Read Size / Records	Shuffle Spill (Memory)	Shuffle Spill (Disk)
1	172.35.205.23:46731	18.6 GB / 40210300	98.1 GB	16.8 GB
2	172.35.195.173:46185	18.7 GB / 40246767	117.2 GB	17.3 GB
3	172.36.135.106:35913	18.6 GB / 40253921	101.6 GB	16.6 GB
4	172.34.131.223:46879	18.6 GB / 40190741	99.5 GB	16.4 GB

Se osservate questi sintomi e la fase richiede troppo tempo rispetto agli obiettivi prestazionali prefissati, oppure fallisce con `Out Of Memory` `No space left on device` errori, prendete in considerazione le seguenti soluzioni.

Ottimizza l'unione

L'`join()`operazione, che unisce le tabelle, è l'operazione di shuffle più comunemente utilizzata, ma è spesso un ostacolo alle prestazioni. Poiché l'unione è un'operazione costosa, si consiglia di non utilizzarla a meno che non sia essenziale per i requisiti aziendali. Ricontrolla che stai facendo un uso efficiente della tua pipeline di dati ponendoti le seguenti domande:

- Stai ricalcolando un join che viene eseguito anche in altri lavori che puoi riutilizzare?
- Vi state unendo per risolvere chiavi esterne relative a valori che non vengono utilizzati dai consumatori del vostro output?

Dopo aver verificato che le operazioni di join sono essenziali per i requisiti aziendali, consulta le seguenti opzioni per ottimizzare l'iscrizione in modo da soddisfare i requisiti.

Usa il pushdown prima di partecipare

Filtra le righe e le colonne non necessarie DataFrame prima di eseguire un join. Ciò presenta i seguenti vantaggi:

- Riduce la quantità di dati trasferiti durante lo shuffle
- Riduce la quantità di elaborazione nell'esecutore Spark
- Riduce la quantità di dati da scansionare

```
# Default
df_joined  = df1.join(df2, ["product_id"])

# Use Pushdown
```

```
df1_select =
 df1.select("product_id","product_title","star_rating").filter(col("star_rating")>=4.0)
df2_select = df2.select("product_id","category_id")
df_joined  = df1_select.join(df2_select, ["product_id"])
```

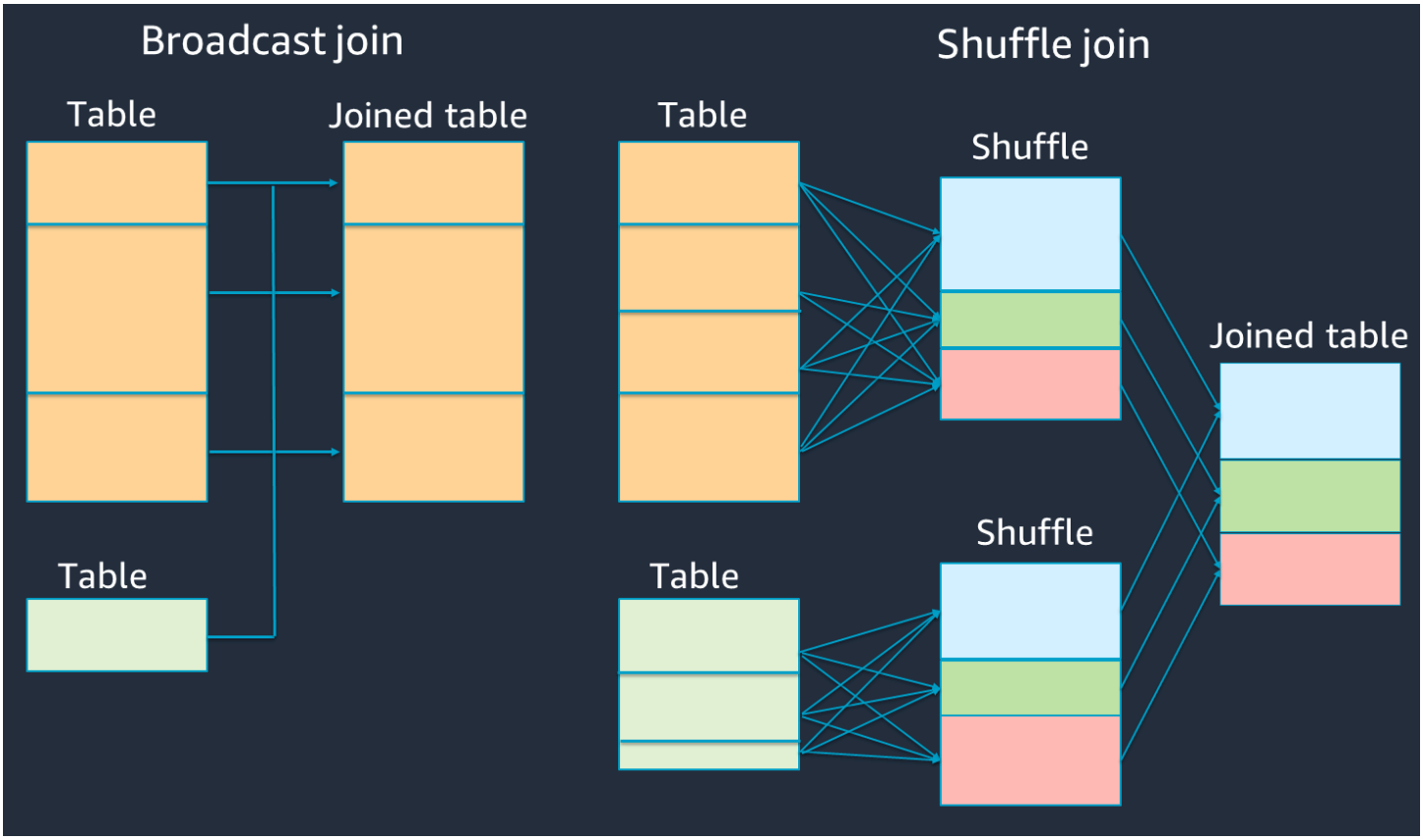
Usa DataFrame Join

Prova a usare uno Spark di alto livello API come Spark SQL e Datasets invece di o join. DataFrame RDD API DynamicFrame Puoi eseguire la conversione in DynamicFrame DataFrame con una chiamata di metodo come. `dyf.toDF()` Come discusso nella sezione Argomenti chiave di Apache Spark, queste operazioni di unione sfruttano internamente l'ottimizzazione delle query mediante l'ottimizzatore Catalyst.

Mescola e trasmetti hash, join e suggerimenti

Spark supporta due tipi di join: shuffle join e broadcast hash join. Un broadcast hash join non richiede lo shuffling e può richiedere meno elaborazione rispetto a uno shuffle join. Tuttavia, è applicabile solo quando si unisce un tavolo piccolo a uno grande. Quando ti unisci a una tabella che può stare nella memoria di un singolo esecutore Spark, prendi in considerazione l'utilizzo di un broadcast hash join.

Il diagramma seguente mostra la struttura e le fasi di alto livello di un broadcast hash join e di uno shuffle join.



I dettagli di ogni join sono i seguenti:

- Shuffle join:
  - Lo shuffle hash join unisce due tabelle senza ordinamento e distribuisce il join tra le due tabelle. È adatto per unire piccole tabelle che possono essere archiviate nella memoria dell'esecutore Spark.
  - Lo sort-merge join distribuisce le due tabelle da unire tramite chiave e le ordina prima di unirle. È adatto per unire tavoli di grandi dimensioni.
- Broadcast hash join:
  - Un broadcast hash join invia il file più piccolo RDD o la tabella a ciascuno dei nodi di lavoro. Quindi esegue una combinazione sul lato della mappa con ogni partizione del più grande o della tabella. RDD

    È adatto per i join quando uno dei tuoi tavoli può stare in memoria RDDs o può essere adattato per adattarsi alla memoria. È utile eseguire un hash join di trasmissione quando possibile, perché non richiede un shuffle. Puoi usare un hint di iscrizione per richiedere un join alla trasmissione da Spark come segue.

```
# DataFrame
from pySpark.sql.functions import broadcast
df_joined= df_big.join(broadcast(df_small), right_df[key] == left_df[key],
 how='inner')

-- SparkSQL
SELECT /*+ BROADCAST(t1) /  FROM t1 INNER JOIN t2 ON t1.key = t2.key;
```

Per maggiori informazioni sui suggerimenti per partecipare, vedi Join hints.

Nella AWS Glue versione 3.0 e successive, puoi sfruttare automaticamente i broadcast hash join abilitando Adaptive Query Execution e parametri aggiuntivi. Adaptive Query Execution converte un sort-merge join in un broadcast hash join quando le statistiche di runtime di entrambi i lati del join sono inferiori alla soglia di adaptive broadcast hash join.

Nella AWS Glue 3.0, è possibile abilitare Adaptive Query Execution impostando. `spark.sql.adaptive.enabled=true` L'esecuzione adattiva delle query è abilitata per impostazione predefinita in AWS Glue 4.0.

È possibile impostare parametri aggiuntivi relativi agli shuffle e agli hash join di trasmissione:

- `spark.sql.adaptive.localShuffleReader.enabled`
- `spark.sql.adaptive.autoBroadcastJoinThreshold`

Per ulteriori informazioni sui parametri correlati, vedere Conversione di sort-merge join in broadcast join.

Nella AWS Glue versione 3.0 e versioni successive, puoi usare altri suggerimenti di join for shuffle per ottimizzare il tuo comportamento.

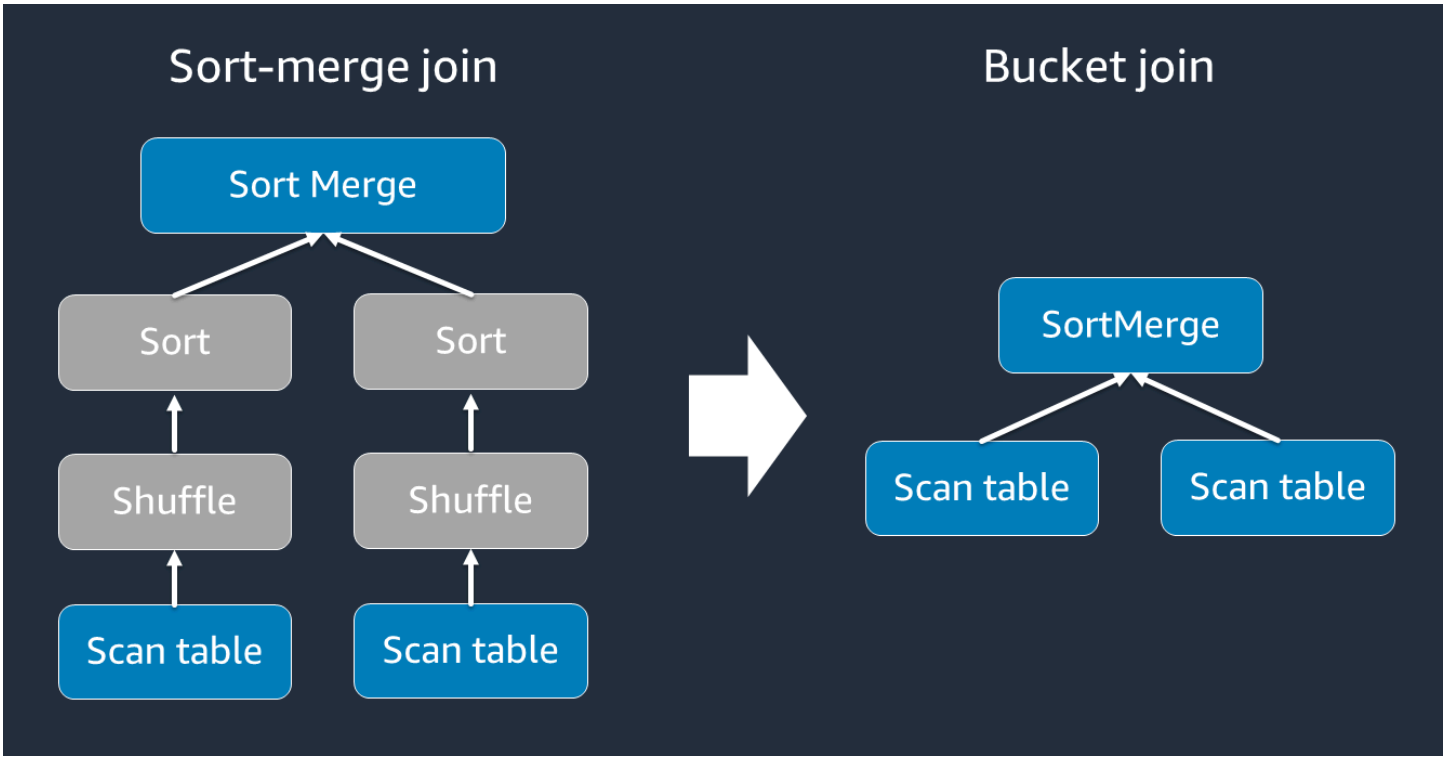
```
-- Join Hints for shuffle sort merge join
SELECT /*+ SHUFFLE_MERGE(t1) /  FROM t1 INNER JOIN t2 ON t1.key = t2.key;
SELECT /*+ MERGEJOIN(t2) /  FROM t1 INNER JOIN t2 ON t1.key = t2.key;
SELECT /*+ MERGE(t1) /  FROM t1 INNER JOIN t2 ON t1.key = t2.key;

-- Join Hints for shuffle hash join
SELECT /*+ SHUFFLE_HASH(t1) /  FROM t1 INNER JOIN t2 ON t1.key = t2.key;
```

```
-- Join Hints for shuffle-and-replicate nested loop join
SELECT /*+ SHUFFLE_REPLICATE_NL(t1) /  FROM t1 INNER JOIN t2 ON t1.key = t2.key;
```

Usa il bucketing

Il sort-merge join richiede due fasi, shuffle e sort, quindi merge. Queste due fasi possono sovraccaricare l'esecutore Spark e causare problemi di prestazioni quando alcuni esecutori si uniscono OOM e altri si ordinano contemporaneamente. In questi casi, potrebbe essere possibile unirsi in modo efficiente utilizzando il bucketing. Bucketing preordinerà e mescolerà i dati immessi nelle chiavi di join, quindi scriverà i dati ordinati in una tabella intermedia. Il costo delle fasi di mescolamento e ordinamento può essere ridotto quando si uniscono tabelle di grandi dimensioni definendo in anticipo le tabelle intermedie ordinate.



Le tabelle bucketed sono utili per quanto segue:

- I dati vengono uniti frequentemente sulla stessa chiave, ad esempio `account_id`
- Caricamento di tabelle cumulative giornaliere, ad esempio tabelle base e delta, che potrebbero essere inserite in una colonna comune

È possibile creare una tabella a blocchi utilizzando il codice seguente.

```
df.write.bucketBy(50, "account_id").sortBy("age").saveAsTable("bucketed_table")
```

Ripartizione delle DataFrames chiavi di unione prima dell'unione

Per ripartizionare le due chiavi DataFrames sulle chiavi di join prima del join, utilizzate le seguenti istruzioni.

```
df1_repartitioned = df1.repartition(N,"join_key")
df2_repartitioned = df2.repartition(N,"join_key")
df_joined  = df1_repartitioned.join(df2_repartitioned,"product_id")
```

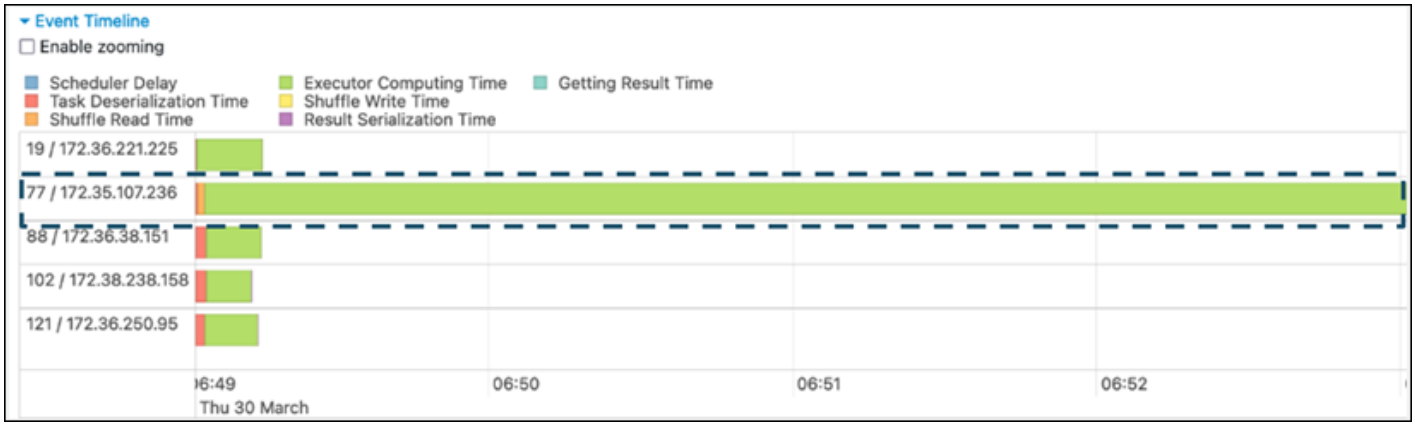
Questo ne partizionerà due (ancora separate) RDDs sulla chiave di unione prima di iniziare l'unione. Se i due record RDDs sono partizionati sulla stessa chiave e con lo stesso codice di partizionamento, è molto probabile che i RDD record che si intende unire vengano collocati insieme sullo stesso worker prima di procedere all'unione. Ciò potrebbe migliorare le prestazioni riducendo l'attività di rete e la distorsione dei dati durante l'unione.

Supera la distorsione dei dati

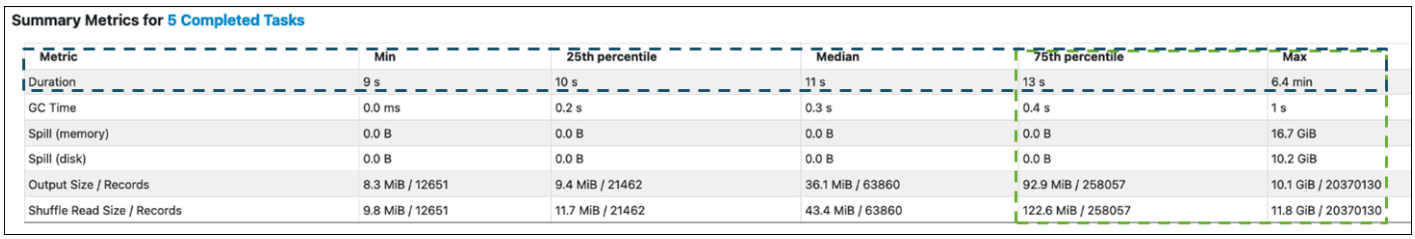
La distorsione dei dati è una delle cause più comuni di difficoltà per i lavori Spark. Si verifica quando i dati non sono distribuiti uniformemente tra le partizioni. RDD Ciò fa sì che le attività relative a quella partizione richiedano molto più tempo rispetto ad altre, ritardando il tempo di elaborazione complessivo dell'applicazione.

Per identificare la distorsione dei dati, valuta le seguenti metriche nell'interfaccia utente di Spark:

- Nella scheda Stage dell'interfaccia utente Spark, esamina la pagina Cronologia degli eventi. Puoi vedere una distribuzione non uniforme delle attività nella schermata seguente. Le attività distribuite in modo non uniforme o che richiedono troppo tempo per essere eseguite possono indicare una distorsione dei dati.



- Un'altra pagina importante è Summary Metrics, che mostra le statistiche per le attività di Spark. La schermata seguente mostra le metriche con percentili per Duration, GC Time, Spill (memoria), Spill (disco) e così via.



Summary Metrics for 5 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	9 s	10 s	11 s	13 s	6.4 min
GC Time	0.0 ms	0.2 s	0.3 s	0.4 s	1 s
Spill (memory)	0.0 B	0.0 B	0.0 B	0.0 B	16.7 GiB
Spill (disk)	0.0 B	0.0 B	0.0 B	0.0 B	10.2 GiB
Output Size / Records	8.3 MiB / 12651	9.4 MiB / 21462	36.1 MiB / 63860	92.9 MiB / 258057	10.1 GiB / 20370130
Shuffle Read Size / Records	9.8 MiB / 12651	11.7 MiB / 21462	43.4 MiB / 63860	122.6 MiB / 258057	11.8 GiB / 20370130

  Quando le attività sono distribuite uniformemente, vedrai numeri simili in tutti i percentili. In caso di distorsione dei dati, in ogni percentile verranno visualizzati valori molto distorti. Nell'esempio, la durata dell'attività è inferiore a 13 secondi in Min, 25° percentile, Mediano e 75° percentile. Sebbene l'attività Max abbia elaborato 100 volte più dati rispetto al 75° percentile, la sua durata di 6,4 minuti è circa 30 volte maggiore. Significa che almeno un'attività (o fino al 25 percento delle attività) ha richiesto molto più tempo rispetto al resto delle attività.

Se i dati sono distorti, prova quanto segue:

- Se usi la AWS Glue versione 3.0, abilita Adaptive Query Execution impostando. `spark.sql.adaptive.enabled=true` Adaptive Query Execution è abilitato per impostazione predefinita nella AWS Glue versione 4.0.

  È inoltre possibile utilizzare Adaptive Query Execution per la distorsione dei dati introdotta dai join impostando i seguenti parametri correlati:

  - `spark.sql.adaptive.skewJoin.skewedPartitionFactor`
  - `spark.sql.adaptive.skewJoin.skewedPartitionThresholdInBytes`

- `spark.sql.adaptive.advisoryPartitionSizeInBytes=128m (128 mebibytes or larger should be good)`
- `spark.sql.adaptive.coalescePartitions.enabled=true (when you want to coalesce partitions)`

Per ulteriori informazioni, consulta la documentazione di Apache Spark.

- Usa chiavi con un ampio intervallo di valori per le chiavi di unione. In uno shuffle join, le partizioni vengono determinate per ogni valore hash di una chiave. Se la cardinalità di una chiave di join è troppo bassa, è più probabile che la funzione hash faccia un cattivo lavoro di distribuzione dei dati tra le partizioni. Pertanto, se la tua applicazione e la tua logica aziendale la supportano, prendi in considerazione l'utilizzo di una chiave di cardinalità più elevata o di una chiave composita.

```
# Use Single Primary Key
df_joined = df1_select.join(df2_select, ["primary_key"])

# Use Composite Key
df_joined = df1_select.join(df2_select, ["primary_key","secondary_key"])
```

Usa la cache

Quando usi metodi ripetitivi DataFrames, evita ulteriori operazioni di shuffle o calcoli utilizzando `df.cache()` o `df.persist()` memorizzando nella cache i risultati del calcolo nella memoria e su disco di ogni esecutore Spark. Spark supporta anche la persistenza RDDs su disco o la replica su più nodi (livello di storage).

Ad esempio, puoi rendere persistente aggiungendo. DataFrames `df.persist()` Quando la cache non è più necessaria, è possibile utilizzarla `unpersist` per eliminare i dati memorizzati nella cache.

```
df = spark.read.parquet("s3://<Bucket>/parquet/product_category=Books/")
df_high_rate = df.filter(col("star_rating")>=4.0)
df_high_rate.persist()

df_joined1 = df_high_rate.join(<Table1>, ["key"])
df_joined2 = df_high_rate.join(<Table2>, ["key"])
df_joined3 = df_high_rate.join(<Table3>, ["key"])
...
df_high_rate.unpersist()
```

Rimuovi le azioni Spark non necessarie

Evita di eseguire azioni non necessarie come `countshow`, o. `collect` Come discusso nella sezione Argomenti chiave di Apache Spark, Spark è pigro. Ogni trasformazione RDD può essere ricalcolata ogni volta che si esegue un'azione su di essa. Quando usi molte azioni Spark, per ogni azione vengono richiamati accessi a più sorgenti, calcoli di attività ed esecuzioni casuali.

Se non hai bisogno `collect()` di altre azioni nel tuo ambiente commerciale, valuta la possibilità di rimuoverle.

> **Note**
>
> Evita il più possibile di usare Spark `collect()` in ambienti commerciali. L'`collect()`azione restituisce tutti i risultati di un calcolo nell'esecutore Spark al driver Spark, il che potrebbe causare la restituzione di un errore da parte del driver Spark. OOM Per evitare un OOM errore, Spark imposta di `spark.driver.maxResultSize = 1GB` default, che limita la dimensione massima dei dati restituiti al driver Spark a 1 GB.

Riduci al minimo i costi di pianificazione

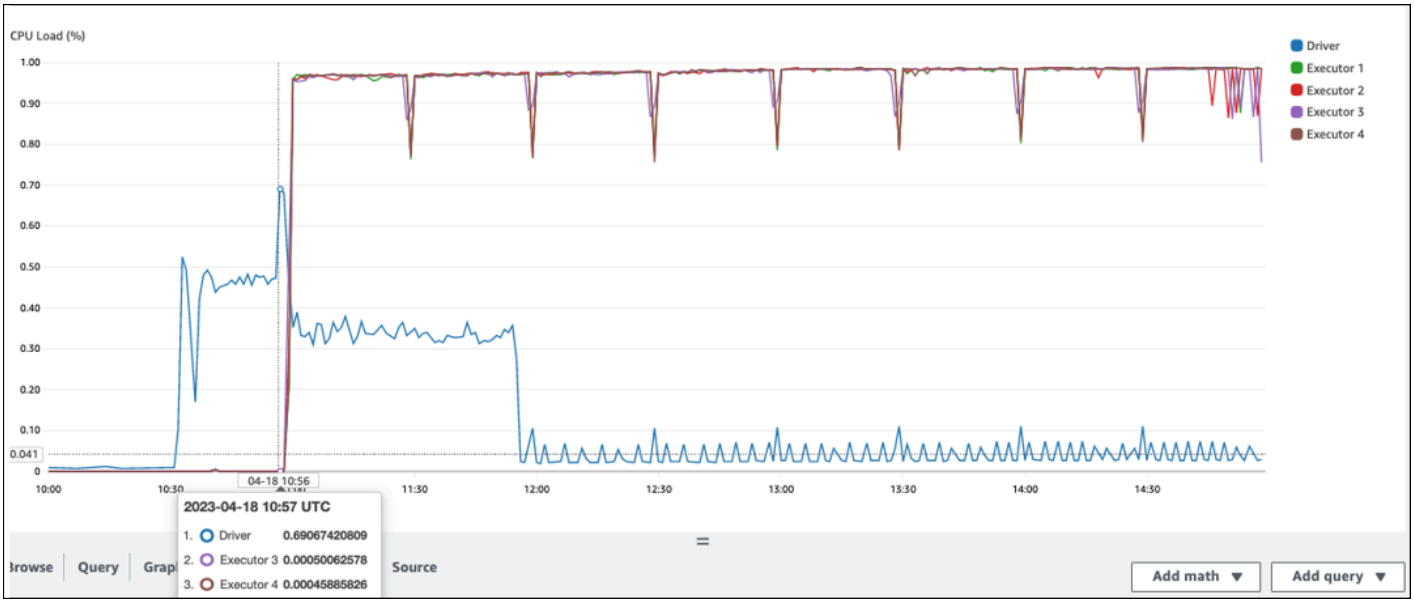
Come discusso Argomenti chiave di Apache Spark, il driver Spark genera il piano di esecuzione. In base a tale piano, le attività vengono assegnate all'esecutore Spark per l'elaborazione distribuita. Tuttavia, il driver Spark può diventare un collo di bottiglia se è presente un gran numero di file di piccole dimensioni o se AWS Glue Data Catalog contiene un gran numero di partizioni. Per identificare un sovraccarico di pianificazione elevato, valuta le seguenti metriche.

CloudWatch metriche

Controlla l'utilizzo CPUdel carico e della memoria per le seguenti situazioni:

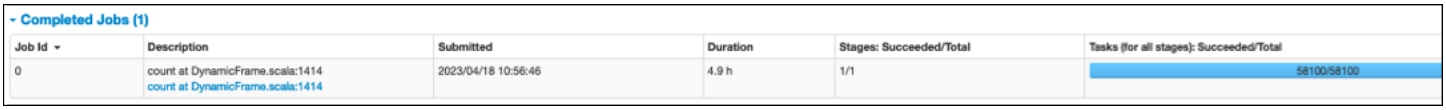
- Il CPUcarico e l'utilizzo della memoria del driver Spark sono registrati come elevati. Normalmente, il driver Spark non elabora i dati, quindi il CPU carico e l'utilizzo della memoria non subiscono picchi. Tuttavia, se l'origine dati di Amazon S3 contiene troppi file di piccole dimensioni, elencare tutti gli oggetti S3 e gestire un gran numero di attività potrebbe causare un utilizzo elevato delle risorse.
- C'è un lungo intervallo prima che l'elaborazione inizi in Spark Executor. Nella seguente schermata di esempio, il CPU carico dell'esecutore Spark è troppo basso fino alle 10:57, anche se il processo

è iniziato alle 10:00. AWS Glue Ciò indica che il driver Spark potrebbe impiegare molto tempo per generare un piano di esecuzione. In questo esempio, recuperare il gran numero di partizioni nel Data Catalog ed elencare il gran numero di file di piccole dimensioni nel driver Spark richiede molto tempo.



Interfaccia utente di Spark

Nella scheda Job dell'interfaccia utente Spark, puoi vedere l'ora di invio. Nell'esempio seguente, il driver Spark ha avviato job0 alle 10:56:46, anche se il lavoro è iniziato alle 10:00:00. AWS Glue



Puoi anche vedere le Attività (per tutte le fasi): Riuscito/Tempo totale nella scheda Job. In questo caso, il numero di attività viene registrato come. `58100` Come spiegato nella sezione Amazon S3 della pagina delle attività di Parallelize, il numero di attività corrisponde approssimativamente al numero di oggetti S3. Ciò significa che ci sono circa 58.100 oggetti in Amazon S3.

Per maggiori dettagli su questo lavoro e sulla tempistica, consulta la scheda Stage. Se riscontri un problema con il driver Spark, prendi in considerazione le seguenti soluzioni:

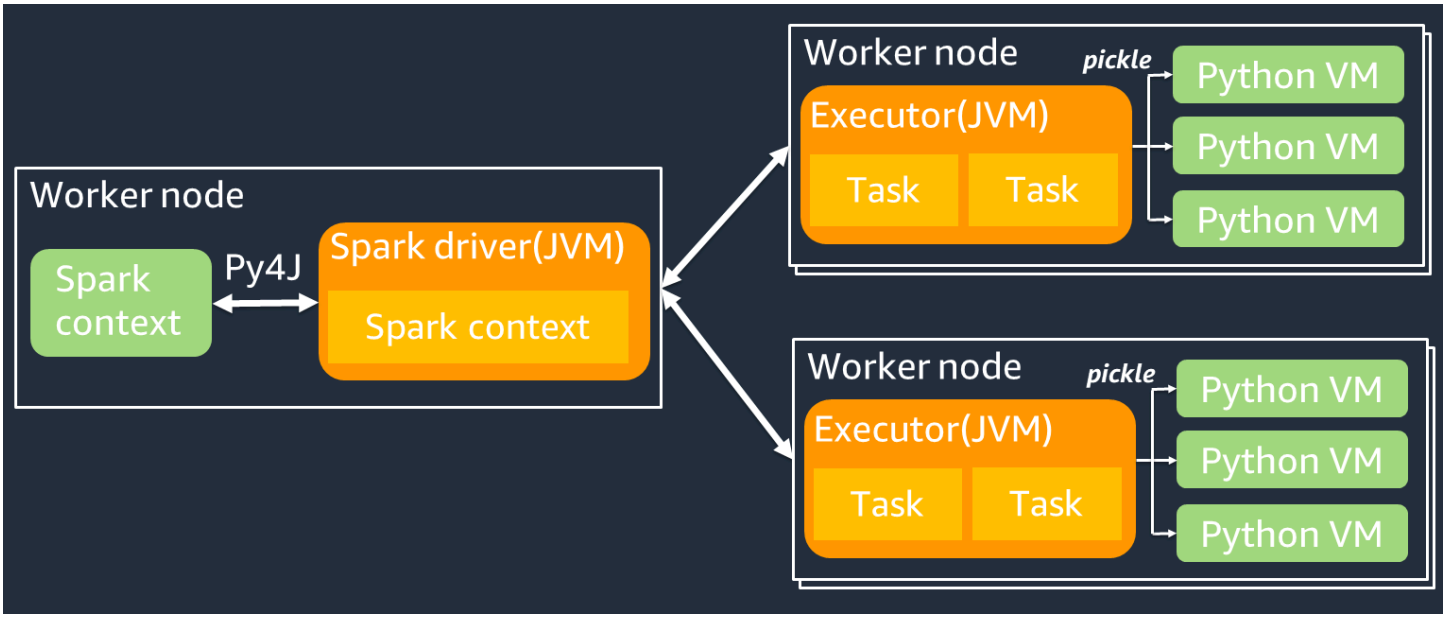
- Quando Amazon S3 ha troppi file, prendi in considerazione le indicazioni sull'eccessivo parallelismo nella sezione Troppe partizioni della pagina delle attività di Parallelize.

- Quando Amazon S3 ha troppe partizioni, prendi in considerazione le indicazioni sul partizionamento eccessivo nella sezione Troppe partizioni Amazon S3 della pagina Riduci la quantità di scansione dei dati. Abilita gli indici di AWS Glue partizione se ci sono molte partizioni per ridurre la latenza per il recupero dei metadati delle partizioni dal Data Catalog. Per ulteriori informazioni, consulta Migliorare le prestazioni delle query utilizzando gli indici di partizione. AWS Glue
- Se JDBC ha troppe partizioni, riduci il valore. `hashpartition`
- Se DynamoDB ha troppe partizioni, riduci il valore. `dynamodb.splits`
- Quando i job di streaming hanno troppe partizioni, riduci il numero di shard.

Ottimizza le funzioni definite dall'utente

Le funzioni definite dall'utente (UDFs) PySpark spesso riducono notevolmente `RDD.map` le prestazioni. Ciò è dovuto al sovraccarico richiesto per rappresentare accuratamente il codice Python nell'implementazione Scala sottostante di Spark.

Il diagramma seguente mostra l'architettura dei job. PySpark Quando si utilizza PySpark, il driver Spark utilizza la libreria Py4j per richiamare i metodi Java da Python. Quando si chiamano Spark SQL o funzioni DataFrame integrate, c'è poca differenza di prestazioni tra Python e Scala perché le funzioni vengono eseguite su ciascun esecutore utilizzando un piano JVM di esecuzione ottimizzato.



Se usi la tua logica Python, ad esempio using`map/ mapPartitions/ udf`, l'attività verrà eseguita in un ambiente di runtime Python. La gestione di due ambienti comporta un costo generale. Inoltre, i dati in memoria devono essere trasformati per essere utilizzati dalle funzioni integrate dell'ambiente di JVM runtime. Pickle è un formato di serializzazione utilizzato di default per lo scambio tra i runtime e JVM Python. Tuttavia, il costo di questo costo di serializzazione e deserializzazione è molto elevato, quindi gli UDFs scritti in Java o Scala sono più veloci di Python. UDFs

Per evitare il sovraccarico di serializzazione e deserializzazione, considerate quanto segue: PySpark

- Usa le SQL funzioni Spark integrate: valuta la possibilità di sostituire la tua funzione o quella della mappa con Spark UDF o con funzioni integrate. SQL DataFrame Quando si eseguono Spark SQL o funzioni DataFrame integrate, c'è poca differenza di prestazioni tra Python e Scala perché le attività vengono gestite su ciascun esecutore. JVM
- UDFsImplementa in Scala o Java: prendi in considerazione l'utilizzo di file scritti in Java o Scala, perché vengono eseguiti su. UDF JVM
- Usa sistemi basati su Apache Arrow UDFs per carichi di lavoro vettoriali: valuta la possibilità di utilizzare quelli basati su Apache Arrow. UDFs Questa funzionalità è nota anche come vettorializzata (Pandas). UDF UDF Apache Arrow è un formato di dati in memoria indipendente dalla lingua che AWS Glue può essere utilizzato per trasferire in modo efficiente i dati tra e processi Python. JVM Questo è attualmente molto vantaggioso per gli utenti di Python che lavorano con Panda o dati. NumPy

  Arrow è un formato colonnare (vettoriale). Il suo utilizzo non è automatico e potrebbe richiedere alcune piccole modifiche alla configurazione o al codice per trarne il massimo vantaggio e garantire la compatibilità. Per maggiori dettagli e limitazioni, consulta Apache Arrow in PySpark.

  L'esempio seguente confronta un incrementale di base UDF in Python standard, come UDF vettorizzato e in Spark. SQL

Python standard UDF

Il tempo di esempio è 3,20 (sec).

Esempio di codice

```
# DataSet
df = spark.range(10000000).selectExpr("id AS a","id AS b")
```

```
# UDF Example
def plus(a,b):
    return a+b
spark.udf.register("plus",plus)

df.selectExpr("count(plus(a,b))").collect()
```

Piano di esecuzione

```
== Physical Plan ==
AdaptiveSparkPlan isFinalPlan=false
+- HashAggregate(keys=[], functions=[count(pythonUDF0#124)])
+- Exchange SinglePartition, ENSURE_REQUIREMENTS, [id=#580]
+- HashAggregate(keys=[], functions=[partial_count(pythonUDF0#124)])
+- Project [pythonUDF0#124]
+- BatchEvalPython [plus(a#116L, b#117L)], [pythonUDF0#124]
+- Project [id#114L AS a#116L, id#114L AS b#117L]
+- Range (0, 10000000, step=1, splits=16)
```

Vettorizzato UDF

Il tempo di esempio è 0,59 (sec).

Vectorized UDF è 5 volte più veloce dell'esempio precedente. UDF VerificaPhysical Plan, come puoi vedereArrowEvalPython, che mostra che questa applicazione è vettorializzata da Apache Arrow. Per abilitare VectorizedUDF, è necessario specificare nel codice. `spark.sql.execution.arrow.pyspark.enabled = true`

Esempio di codice

```
# Vectorized UDF
from pyspark.sql.types import LongType
from pyspark.sql.functions import count, pandas_udf

# Enable Apache Arrow Support
spark.conf.set("spark.sql.execution.arrow.pyspark.enabled", "true")

# DataSet
df = spark.range(10000000).selectExpr("id AS a","id AS b")

# Annotate pandas_udf to use Vectorized UDF
```

```
@pandas_udf(LongType())
def pandas_plus(a,b):
    return a+b
spark.udf.register("pandas_plus",pandas_plus)

df.selectExpr("count(pandas_plus(a,b))").collect()
```

Piano di esecuzione

```
== Physical Plan ==
AdaptiveSparkPlan isFinalPlan=false
+- HashAggregate(keys=[], functions=[count(pythonUDF0#1082L)],
 output=[count(pandas_plus(a, b))#1080L])
+- Exchange SinglePartition, ENSURE_REQUIREMENTS, [id=#5985]
+- HashAggregate(keys=[], functions=[partial_count(pythonUDF0#1082L)],
 output=[count#1084L])
+- Project [pythonUDF0#1082L]
+- ArrowEvalPython [pandas_plus(a#1074L, b#1075L)], [pythonUDF0#1082L], 200
+- Project [id#1072L AS a#1074L, id#1072L AS b#1075L]
+- Range (0, 10000000, step=1, splits=16)
```

Spark SQL

Il tempo di esempio è 0,087 (sec).

Spark SQL è molto più veloce di VectorizedUDF, perché le attività vengono eseguite su ogni executor senza JVM un runtime Python. Se puoi sostituire la tua UDF con una funzione integrata, ti consigliamo di farlo.

Esempio di codice

```
df.createOrReplaceTempView("test")
spark.sql("select count(a+b) from test").collect()
```

Usare i panda per i big data

Se conosci già i panda e vuoi usare Spark per i big data, puoi usare i panda su Spark. API AWS Glue La versione 4.0 e le versioni successive lo supportano. Per iniziare, puoi utilizzare il notebook ufficiale Quickstart: Pandas API su Spark. Per ulteriori informazioni, consulta la documentazione. PySpark

Risorse

- AWS Glue
- Ottimizzazione delle prestazioni (Spark SQL Guide)
- AWS Glue Workshop sull'ottimizzazione

Cronologia dei documenti

La tabella seguente descrive le modifiche significative apportate a questa guida. Per ricevere notifiche sugli aggiornamenti futuri, puoi abbonarti a un feed RSS.

Modifica	Descrizione	Data
Pubblicazione iniziale	—	2 gennaio 2024

AWS Glossario delle linee guida prescrittive

I seguenti sono termini comunemente usati nelle strategie, nelle guide e nei modelli forniti da AWS Prescriptive Guidance. Per suggerire voci, utilizza il link Fornisci feedback alla fine del glossario.

Numeri

7 R

Sette strategie di migrazione comuni per trasferire le applicazioni sul cloud. Queste strategie si basano sulle 5 R identificate da Gartner nel 2011 e sono le seguenti:

- Rifattorizzare/riprogettare: trasferisci un'applicazione e modifica la sua architettura sfruttando appieno le funzionalità native del cloud per migliorare l'agilità, le prestazioni e la scalabilità. Ciò comporta in genere la portabilità del sistema operativo e del database. Esempio: migra il tuo database Oracle locale all'edizione compatibile con Amazon Aurora PostgreSQL.
- Ridefinire la piattaforma (lift and reshape): trasferisci un'applicazione nel cloud e introduci un certo livello di ottimizzazione per sfruttare le funzionalità del cloud. Esempio: migra il tuo database Oracle locale ad Amazon Relational Database Service (Amazon RDS) per Oracle in. Cloud AWS
- Riacquistare (drop and shop): passa a un prodotto diverso, in genere effettuando la transizione da una licenza tradizionale a un modello SaaS. Esempio: migra il tuo sistema di gestione delle relazioni con i clienti (CRM) su Salesforce.com.
- Eseguire il rehosting (lift and shift): trasferisci un'applicazione sul cloud senza apportare modifiche per sfruttare le funzionalità del cloud. Esempio: migra il tuo database Oracle locale a Oracle su un'istanza EC2 in. Cloud AWS
- Trasferire (eseguire il rehosting a livello hypervisor): trasferisci l'infrastruttura sul cloud senza acquistare nuovo hardware, riscrivere le applicazioni o modificare le operazioni esistenti. Esegui la migrazione dei server da una piattaforma locale a un servizio cloud per la stessa piattaforma. Esempio: migra un'applicazione suMicrosoft Hyper-V. AWS
- Riesaminare (mantenere): mantieni le applicazioni nell'ambiente di origine. Queste potrebbero includere applicazioni che richiedono una rifattorizzazione significativa che desideri rimandare a un momento successivo e applicazioni legacy che desideri mantenere, perché non vi è alcuna giustificazione aziendale per effettuarne la migrazione.
- Ritirare: disattiva o rimuovi le applicazioni che non sono più necessarie nell'ambiente di origine.

A

ABAC

Vedi controllo degli accessi basato sugli attributi.

servizi astratti

Vedi servizi gestiti.

ACIDO

Vedi atomicità, consistenza, isolamento, durata.

migrazione attiva-attiva

Un metodo di migrazione del database in cui i database di origine e di destinazione vengono mantenuti sincronizzati (utilizzando uno strumento di replica bidirezionale o operazioni di doppia scrittura) ed entrambi i database gestiscono le transazioni provenienti dalle applicazioni di connessione durante la migrazione. Questo metodo supporta la migrazione in piccoli batch controllati anziché richiedere una conversione una tantum. È più flessibile ma richiede più lavoro rispetto alla migrazione attiva-passiva.

migrazione attiva-passiva

Un metodo di migrazione di database in cui i database di origine e di destinazione vengono mantenuti sincronizzati, ma solo il database di origine gestisce le transazioni provenienti dalle applicazioni di connessione mentre i dati vengono replicati nel database di destinazione. Il database di destinazione non accetta alcuna transazione durante la migrazione.

funzione aggregata

Una funzione SQL che opera su un gruppo di righe e calcola un singolo valore restituito per il gruppo. Esempi di funzioni aggregate includono `SUM` e. `MAX`

Intelligenza artificiale

Vedi intelligenza artificiale.

AIOps

Guarda le operazioni di intelligenza artificiale.

anonimizzazione

Il processo di eliminazione permanente delle informazioni personali in un set di dati. L'anonimizzazione può aiutare a proteggere la privacy personale. I dati anonimi non sono più considerati dati personali.

anti-modello

Una soluzione utilizzata frequentemente per un problema ricorrente in cui la soluzione è controproducente, inefficace o meno efficace di un'alternativa.

controllo delle applicazioni

Un approccio alla sicurezza che consente l'uso solo di applicazioni approvate per proteggere un sistema dal malware.

portfolio di applicazioni

Una raccolta di informazioni dettagliate su ogni applicazione utilizzata da un'organizzazione, compresi i costi di creazione e manutenzione dell'applicazione e il relativo valore aziendale. Queste informazioni sono fondamentali per il processo di scoperta e analisi del portfolio e aiutano a identificare e ad assegnare la priorità alle applicazioni da migrare, modernizzare e ottimizzare.

intelligenza artificiale (IA)

Il campo dell'informatica dedicato all'uso delle tecnologie informatiche per svolgere funzioni cognitive tipicamente associate agli esseri umani, come l'apprendimento, la risoluzione di problemi e il riconoscimento di schemi. Per ulteriori informazioni, consulta la sezione Che cos'è l'intelligenza artificiale?

operazioni di intelligenza artificiale (AIOps)

Il processo di utilizzo delle tecniche di machine learning per risolvere problemi operativi, ridurre gli incidenti operativi e l'intervento umano e aumentare la qualità del servizio. Per ulteriori informazioni su come viene utilizzato AIOps nella strategia di migrazione AWS , consulta la guida all'integrazione delle operazioni.

crittografia asimmetrica

Un algoritmo di crittografia che utilizza una coppia di chiavi, una chiave pubblica per la crittografia e una chiave privata per la decrittografia. Puoi condividere la chiave pubblica perché non viene utilizzata per la decrittografia, ma l'accesso alla chiave privata deve essere altamente limitato.

atomicità, consistenza, isolamento, durabilità (ACID)

Un insieme di proprietà del software che garantiscono la validità dei dati e l'affidabilità operativa di un database, anche in caso di errori, interruzioni di corrente o altri problemi.

Controllo degli accessi basato su attributi (ABAC)

La pratica di creare autorizzazioni dettagliate basate su attributi utente, come reparto, ruolo professionale e nome del team. Per ulteriori informazioni, consulta ABAC for AWS nella documentazione AWS Identity and Access Management (IAM).

fonte di dati autorevole

Una posizione in cui è archiviata la versione principale dei dati, considerata la fonte di informazioni più affidabile. È possibile copiare i dati dalla fonte di dati autorevole in altre posizioni allo scopo di elaborarli o modificarli, ad esempio anonimizzandoli, oscurandoli o pseudonimizzandoli.

Zona di disponibilità

Una posizione distinta all'interno di un edificio Regione AWS che è isolata dai guasti in altre zone di disponibilità e offre una connettività di rete economica e a bassa latenza verso altre zone di disponibilità nella stessa regione.

AWS Cloud Adoption Framework (CAF)AWS

Un framework di linee guida e best practice AWS per aiutare le organizzazioni a sviluppare un piano efficiente ed efficace per passare con successo al cloud. AWS CAF organizza le linee guida in sei aree di interesse chiamate prospettive: business, persone, governance, piattaforma, sicurezza e operazioni. Le prospettive relative ad azienda, persone e governance si concentrano sulle competenze e sui processi aziendali; le prospettive relative alla piattaforma, alla sicurezza e alle operazioni si concentrano sulle competenze e sui processi tecnici. Ad esempio, la prospettiva relativa alle persone si rivolge alle parti interessate che gestiscono le risorse umane (HR), le funzioni del personale e la gestione del personale. In questa prospettiva, AWS CAF fornisce linee guida per lo sviluppo delle persone, la formazione e le comunicazioni per aiutare a preparare l'organizzazione all'adozione del cloud di successo. Per ulteriori informazioni, consulta il sito web di AWS CAF e il white paper AWS CAF.

AWS Workload Qualification Framework (WQF)AWS

Uno strumento che valuta i carichi di lavoro di migrazione dei database, consiglia strategie di migrazione e fornisce stime del lavoro. AWS WQF è incluso in (). AWS Schema Conversion Tool AWS SCT Analizza gli schemi di database e gli oggetti di codice, il codice dell'applicazione, le dipendenze e le caratteristiche delle prestazioni e fornisce report di valutazione.

B

bot difettoso

Un bot che ha lo scopo di disturbare o causare danni a individui o organizzazioni.

BCP

Vedi la pianificazione della continuità operativa.

grafico comportamentale

Una vista unificata, interattiva dei comportamenti delle risorse e delle interazioni nel tempo. Puoi utilizzare un grafico comportamentale con Amazon Detective per esaminare tentativi di accesso non riusciti, chiamate API sospette e azioni simili. Per ulteriori informazioni, consulta Dati in un grafico comportamentale nella documentazione di Detective.

sistema big-endian

Un sistema che memorizza per primo il byte più importante. Vedi anche endianness.

Classificazione binaria

Un processo che prevede un risultato binario (una delle due classi possibili). Ad esempio, il modello di machine learning potrebbe dover prevedere problemi come "Questa e-mail è spam o non è spam?" o "Questo prodotto è un libro o un'auto?"

filtro Bloom

Una struttura di dati probabilistica ed efficiente in termini di memoria che viene utilizzata per verificare se un elemento fa parte di un set.

distribuzioni blu/verdi

Una strategia di implementazione in cui si creano due ambienti separati ma identici. La versione corrente dell'applicazione viene eseguita in un ambiente (blu) e la nuova versione dell'applicazione nell'altro ambiente (verde). Questa strategia consente di ripristinare rapidamente il sistema con un impatto minimo.

bot

Un'applicazione software che esegue attività automatizzate su Internet e simula l'attività o l'interazione umana. Alcuni bot sono utili o utili, come i web crawler che indicizzano le informazioni su Internet. Alcuni altri bot, noti come bot dannosi, hanno lo scopo di disturbare o causare danni a individui o organizzazioni.

botnet

Reti di bot infettate da malware e controllate da un'unica parte, nota come bot herder o bot operator. Le botnet sono il meccanismo più noto per scalare i bot e il loro impatto.

ramo

Un'area contenuta di un repository di codice. Il primo ramo creato in un repository è il ramo principale. È possibile creare un nuovo ramo a partire da un ramo esistente e quindi sviluppare funzionalità o correggere bug al suo interno. Un ramo creato per sviluppare una funzionalità viene comunemente detto ramo di funzionalità. Quando la funzionalità è pronta per il rilascio, il ramo di funzionalità viene ricongiunto al ramo principale. Per ulteriori informazioni, consulta Informazioni sulle filiali (documentazione). GitHub

accesso break-glass

In circostanze eccezionali e tramite una procedura approvata, un mezzo rapido per consentire a un utente di accedere a un sito a Account AWS cui in genere non dispone delle autorizzazioni necessarie. Per ulteriori informazioni, vedere l'indicatore Implementate break-glass procedures nella guida Well-Architected AWS .

strategia brownfield

L'infrastruttura esistente nell'ambiente. Quando si adotta una strategia brownfield per un'architettura di sistema, si progetta l'architettura in base ai vincoli dei sistemi e dell'infrastruttura attuali. Per l'espansione dell'infrastruttura esistente, è possibile combinare strategie brownfield e greenfield.

cache del buffer

L'area di memoria in cui sono archiviati i dati a cui si accede con maggiore frequenza.

capacità di business

Azioni intraprese da un'azienda per generare valore (ad esempio vendite, assistenza clienti o marketing). Le architetture dei microservizi e le decisioni di sviluppo possono essere guidate dalle capacità aziendali. Per ulteriori informazioni, consulta la sezione Organizzazione in base alle funzionalità aziendali del whitepaper Esecuzione di microservizi containerizzati su AWS.

pianificazione della continuità operativa (BCP)

Un piano che affronta il potenziale impatto di un evento che comporta l'interruzione dell'attività, come una migrazione su larga scala, sulle operazioni e consente a un'azienda di riprendere rapidamente le operazioni.

C

CAF

Vedi AWS Cloud Adoption Framework.

implementazione canaria

Il rilascio lento e incrementale di una versione agli utenti finali. Quando sei sicuro, distribuisci la nuova versione e sostituisci la versione corrente nella sua interezza.

CoE

Vedi Cloud Center of Excellence.

CDC

Vedi Change Data Capture.

Change Data Capture (CDC)

Il processo di tracciamento delle modifiche a un'origine dati, ad esempio una tabella di database, e di registrazione dei metadati relativi alla modifica. È possibile utilizzare CDC per vari scopi, ad esempio il controllo o la replica delle modifiche in un sistema di destinazione per mantenere la sincronizzazione.

ingegneria del caos

Introduzione intenzionale di guasti o eventi dirompenti per testare la resilienza di un sistema. Puoi usare AWS Fault Injection Service (AWS FIS) per eseguire esperimenti che stressano i tuoi AWS carichi di lavoro e valutarne la risposta.

CI/CD

Vedi integrazione continua e distribuzione continua.

classificazione

Un processo di categorizzazione che aiuta a generare previsioni. I modelli di ML per problemi di classificazione prevedono un valore discreto. I valori discreti sono sempre distinti l'uno dall'altro. Ad esempio, un modello potrebbe dover valutare se in un'immagine è presente o meno un'auto.

crittografia lato client

Crittografia dei dati a livello locale, prima che il destinatario li AWS servizio riceva.

centro di eccellenza del cloud (CCoE)

Un team multidisciplinare che guida le iniziative di adozione del cloud in tutta l'organizzazione, tra cui lo sviluppo di best practice per il cloud, la mobilitazione delle risorse, la definizione delle tempistiche di migrazione e la guida dell'organizzazione attraverso trasformazioni su larga scala. Per ulteriori informazioni, consulta i post di CCoE sull' Cloud AWS Enterprise Strategy Blog.

cloud computing

La tecnologia cloud generalmente utilizzata per l'archiviazione remota di dati e la gestione dei dispositivi IoT. Il cloud computing è generalmente collegato alla tecnologia di edge computing.

modello operativo cloud

In un'organizzazione IT, il modello operativo utilizzato per creare, maturare e ottimizzare uno o più ambienti cloud. Per ulteriori informazioni, consulta Building your Cloud Operating Model.

fasi di adozione del cloud

Le quattro fasi che le organizzazioni in genere attraversano quando migrano verso Cloud AWS:

- Progetto: esecuzione di alcuni progetti relativi al cloud per scopi di dimostrazione e apprendimento
- Fondamento: effettuare investimenti fondamentali per dimensionare l'adozione del cloud (ad esempio, creazione di una zona di destinazione, definizione di un CCoE, definizione di un modello operativo)
- Migrazione: migrazione di singole applicazioni
- Reinvenzione: ottimizzazione di prodotti e servizi e innovazione nel cloud

Queste fasi sono state definite da Stephen Orban nel post del blog The Journey Toward Cloud-First & the Stages of Adoption on the Enterprise Strategy. Cloud AWS Per informazioni su come si relazionano alla strategia di AWS migrazione, consulta la guida alla preparazione alla migrazione.

CMDB

Vedi database di gestione della configurazione.

repository di codice

Una posizione in cui il codice di origine e altri asset, come documentazione, esempi e script, vengono archiviati e aggiornati attraverso processi di controllo delle versioni. Gli archivi cloud più comuni includono GitHub o AWS CodeCommit. Ogni versione del codice è denominata ramo. In

una struttura a microservizi, ogni repository è dedicato a una singola funzionalità. Una singola pipeline CI/CD può utilizzare più repository.

cache fredda

Una cache del buffer vuota, non ben popolata o contenente dati obsoleti o irrilevanti. Ciò influisce sulle prestazioni perché l'istanza di database deve leggere dalla memoria o dal disco principale, il che richiede più tempo rispetto alla lettura dalla cache del buffer.

dati freddi

Dati a cui si accede raramente e che in genere sono storici. Quando si eseguono interrogazioni di questo tipo di dati, le interrogazioni lente sono in genere accettabili. Lo spostamento di questi dati su livelli o classi di storage meno costosi e con prestazioni inferiori può ridurre i costi.

visione artificiale (CV)

Un campo dell'intelligenza artificiale che utilizza l'apprendimento automatico per analizzare ed estrarre informazioni da formati visivi come immagini e video digitali. Ad esempio, AWS Panorama offre dispositivi che aggiungono CV alle reti di telecamere locali e Amazon SageMaker fornisce algoritmi di elaborazione delle immagini per CV.

deriva della configurazione

Per un carico di lavoro, una modifica della configurazione rispetto allo stato previsto. Potrebbe causare la non conformità del carico di lavoro e in genere è graduale e involontaria.

database di gestione della configurazione (CMDB)

Un repository che archivia e gestisce le informazioni su un database e il relativo ambiente IT, inclusi i componenti hardware e software e le relative configurazioni. In genere si utilizzano i dati di un CMDB nella fase di individuazione e analisi del portafoglio della migrazione.

Pacchetto di conformità

Una raccolta di AWS Config regole e azioni correttive che puoi assemblare per personalizzare i controlli di conformità e sicurezza. È possibile distribuire un pacchetto di conformità come singola entità in una regione Account AWS and o all'interno di un'organizzazione utilizzando un modello YAML. Per ulteriori informazioni, consulta i Conformance Pack nella documentazione. AWS Config

integrazione e distribuzione continua (continuous integration and continuous delivery, CI/CD)

Il processo di automazione delle fasi di origine, creazione, test, gestione temporanea e produzione del processo di rilascio del software. Il processo CI/CD è comunemente descritto come una

pipeline. CI/CD può aiutare ad automatizzare i processi, migliorare la produttività, migliorare la qualità del codice e velocizzare le distribuzioni. Per ulteriori informazioni, consulta Vantaggi della distribuzione continua. CD può anche significare continuous deployment (implementazione continua). Per ulteriori informazioni, consulta Distribuzione continua e implementazione continua a confronto.

CV

Vedi visione artificiale.

D

dati a riposo

Dati stazionari nella rete, ad esempio i dati archiviati.

classificazione dei dati

Un processo per identificare e classificare i dati nella rete in base alla loro criticità e sensibilità. È un componente fondamentale di qualsiasi strategia di gestione dei rischi di sicurezza informatica perché consente di determinare i controlli di protezione e conservazione appropriati per i dati. La classificazione dei dati è un componente del pilastro della sicurezza nel AWS Well-Architected Framework. Per ulteriori informazioni, consulta Classificazione dei dati.

deriva dei dati

Una variazione significativa tra i dati di produzione e i dati utilizzati per addestrare un modello di machine learning o una modifica significativa dei dati di input nel tempo. La deriva dei dati può ridurre la qualità, l'accuratezza e l'equità complessive nelle previsioni dei modelli ML.

dati in transito

Dati che si spostano attivamente attraverso la rete, ad esempio tra le risorse di rete.

rete di dati

Un framework architettonico che fornisce la proprietà distribuita e decentralizzata dei dati con gestione e governance centralizzate.

riduzione al minimo dei dati

Il principio della raccolta e del trattamento dei soli dati strettamente necessari. Praticare la riduzione al minimo dei dati in the Cloud AWS può ridurre i rischi per la privacy, i costi e l'impronta di carbonio delle analisi.

perimetro dei dati

Una serie di barriere preventive nell' AWS ambiente che aiutano a garantire che solo le identità attendibili accedano alle risorse attendibili delle reti previste. Per ulteriori informazioni, consulta Building a data perimeter on. AWS

pre-elaborazione dei dati

Trasformare i dati grezzi in un formato che possa essere facilmente analizzato dal modello di ML. La pre-elaborazione dei dati può comportare la rimozione di determinate colonne o righe e l'eliminazione di valori mancanti, incoerenti o duplicati.

provenienza dei dati

Il processo di tracciamento dell'origine e della cronologia dei dati durante il loro ciclo di vita, ad esempio il modo in cui i dati sono stati generati, trasmessi e archiviati.

soggetto dei dati

Un individuo i cui dati vengono raccolti ed elaborati.

data warehouse

Un sistema di gestione dei dati che supporta la business intelligence, come l'analisi. I data warehouse contengono in genere grandi quantità di dati storici e vengono generalmente utilizzati per interrogazioni e analisi.

linguaggio di definizione del database (DDL)

Istruzioni o comandi per creare o modificare la struttura di tabelle e oggetti in un database.

linguaggio di manipolazione del database (DML)

Istruzioni o comandi per modificare (inserire, aggiornare ed eliminare) informazioni in un database.

DDL

Vedi linguaggio di definizione del database.

deep ensemble

Combinare più modelli di deep learning per la previsione. È possibile utilizzare i deep ensemble per ottenere una previsione più accurata o per stimare l'incertezza nelle previsioni.

deep learning

Un sottocampo del ML che utilizza più livelli di reti neurali artificiali per identificare la mappatura tra i dati di input e le variabili target di interesse.

defense-in-depth

Un approccio alla sicurezza delle informazioni in cui una serie di meccanismi e controlli di sicurezza sono accuratamente stratificati su una rete di computer per proteggere la riservatezza, l'integrità e la disponibilità della rete e dei dati al suo interno. Quando si adotta questa strategia AWS, si aggiungono più controlli a diversi livelli della AWS Organizations struttura per proteggere le risorse. Ad esempio, un defense-in-depth approccio potrebbe combinare l'autenticazione a più fattori, la segmentazione della rete e la crittografia.

amministratore delegato

In AWS Organizations, un servizio compatibile può registrare un account AWS membro per amministrare gli account dell'organizzazione e gestire le autorizzazioni per quel servizio. Questo account è denominato amministratore delegato per quel servizio specifico. Per ulteriori informazioni e un elenco di servizi compatibili, consulta Servizi che funzionano con AWS Organizations nella documentazione di AWS Organizations .

implementazione

Il processo di creazione di un'applicazione, di nuove funzionalità o di correzioni di codice disponibili nell'ambiente di destinazione. L'implementazione prevede l'applicazione di modifiche in una base di codice, seguita dalla creazione e dall'esecuzione di tale base di codice negli ambienti applicativi.

Ambiente di sviluppo

Vedi ambiente.

controllo di rilevamento

Un controllo di sicurezza progettato per rilevare, registrare e avvisare dopo che si è verificato un evento. Questi controlli rappresentano una seconda linea di difesa e avvisano l'utente in caso di eventi di sicurezza che aggirano i controlli preventivi in vigore. Per ulteriori informazioni, consulta Controlli di rilevamento in Implementazione dei controlli di sicurezza in AWS.

mappatura del flusso di valore dello sviluppo (DVSM)

Un processo utilizzato per identificare e dare priorità ai vincoli che influiscono negativamente sulla velocità e sulla qualità nel ciclo di vita dello sviluppo del software. DVSM estende il processo di

mappatura del flusso di valore originariamente progettato per pratiche di produzione snella. Si concentra sulle fasi e sui team necessari per creare e trasferire valore attraverso il processo di sviluppo del software.

gemello digitale

Una rappresentazione virtuale di un sistema reale, ad esempio un edificio, una fabbrica, un'attrezzatura industriale o una linea di produzione. I gemelli digitali supportano la manutenzione predittiva, il monitoraggio remoto e l'ottimizzazione della produzione.

tabella delle dimensioni

In uno schema a stella, una tabella più piccola che contiene gli attributi dei dati quantitativi in una tabella dei fatti. Gli attributi della tabella delle dimensioni sono in genere campi di testo o numeri discreti che si comportano come testo. Questi attributi vengono comunemente utilizzati per il vincolo delle query, il filtraggio e l'etichettatura dei set di risultati.

disastro

Un evento che impedisce a un carico di lavoro o a un sistema di raggiungere gli obiettivi aziendali nella sua sede principale di implementazione. Questi eventi possono essere disastri naturali, guasti tecnici o il risultato di azioni umane, come errori di configurazione involontari o attacchi di malware.

disaster recovery (DR)

La strategia e il processo utilizzati per ridurre al minimo i tempi di inattività e la perdita di dati causati da un disastro. Per ulteriori informazioni, consulta Disaster Recovery of Workloads su AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML

Vedi linguaggio di manipolazione del database.

progettazione basata sul dominio

Un approccio allo sviluppo di un sistema software complesso collegandone i componenti a domini in evoluzione, o obiettivi aziendali principali, perseguiti da ciascun componente. Questo concetto è stato introdotto da Eric Evans nel suo libro,Domain-Driven Design: Tackling Complexity in the Heart of Software (Boston: Addison-Wesley Professional, 2003). Per informazioni su come utilizzare la progettazione basata sul dominio con il modello del fico strangolatore (Strangler Fig), consulta la sezione Modernizzazione incrementale dei servizi Web Microsoft ASP.NET (ASMX) legacy utilizzando container e il Gateway Amazon API.

DOTT.

Vedi disaster recovery.

rilevamento della deriva

Tracciamento delle deviazioni da una configurazione di base. Ad esempio, è possibile AWS CloudFormation utilizzarlo per rilevare deviazioni nelle risorse di sistema oppure AWS Control Tower per rilevare cambiamenti nella landing zone che potrebbero influire sulla conformità ai requisiti di governance.

DVSM

Vedi la mappatura del flusso di valore dello sviluppo.

E

EDA

Vedi analisi esplorativa dei dati.

edge computing

La tecnologia che aumenta la potenza di calcolo per i dispositivi intelligenti all'edge di una rete IoT. Rispetto al cloud computing, l'edge computing può ridurre la latenza di comunicazione e migliorare i tempi di risposta.

crittografia

Un processo di elaborazione che trasforma i dati in chiaro, leggibili dall'uomo, in testo cifrato.

chiave crittografica

Una stringa crittografica di bit randomizzati generata da un algoritmo di crittografia. Le chiavi possono variare di lunghezza e ogni chiave è progettata per essere imprevedibile e univoca.

endianità

L'ordine in cui i byte vengono archiviati nella memoria del computer. I sistemi big-endian memorizzano per primo il byte più importante. I sistemi little-endian memorizzano per primo il byte meno importante.

endpoint

Vedi service endpoint.

servizio endpoint

Un servizio che puoi ospitare in un cloud privato virtuale (VPC) da condividere con altri utenti. Puoi creare un servizio endpoint con AWS PrivateLink e concedere autorizzazioni ad altri Account AWS o a AWS Identity and Access Management (IAM) principali. Questi account o principali possono connettersi al servizio endpoint in privato creando endpoint VPC di interfaccia. Per ulteriori informazioni, consulta Creazione di un servizio endpoint nella documentazione di Amazon Virtual Private Cloud (Amazon VPC).

pianificazione delle risorse aziendali (ERP)

Un sistema che automatizza e gestisce i processi aziendali chiave (come contabilità, MES e gestione dei progetti) per un'azienda.

crittografia envelope

Il processo di crittografia di una chiave di crittografia con un'altra chiave di crittografia. Per ulteriori informazioni, vedete Envelope encryption nella documentazione AWS Key Management Service (AWS KMS).

ambiente

Un'istanza di un'applicazione in esecuzione. Di seguito sono riportati i tipi di ambiente più comuni nel cloud computing:

- ambiente di sviluppo: un'istanza di un'applicazione in esecuzione disponibile solo per il team principale responsabile della manutenzione dell'applicazione. Gli ambienti di sviluppo vengono utilizzati per testare le modifiche prima di promuoverle negli ambienti superiori. Questo tipo di ambiente viene talvolta definito ambiente di test.
- ambienti inferiori: tutti gli ambienti di sviluppo di un'applicazione, ad esempio quelli utilizzati per le build e i test iniziali.
- ambiente di produzione: un'istanza di un'applicazione in esecuzione a cui gli utenti finali possono accedere. In una pipeline CI/CD, l'ambiente di produzione è l'ultimo ambiente di implementazione.
- ambienti superiori: tutti gli ambienti a cui possono accedere utenti diversi dal team di sviluppo principale. Si può trattare di un ambiente di produzione, ambienti di preproduzione e ambienti per i test di accettazione da parte degli utenti.

epica

Nelle metodologie agili, categorie funzionali che aiutano a organizzare e dare priorità al lavoro. Le epiche forniscono una descrizione di alto livello dei requisiti e delle attività di implementazione.

Ad esempio, le epopee della sicurezza AWS CAF includono la gestione delle identità e degli accessi, i controlli investigativi, la sicurezza dell'infrastruttura, la protezione dei dati e la risposta agli incidenti. Per ulteriori informazioni sulle epiche, consulta la strategia di migrazione AWS , consulta la guida all'implementazione del programma.

ERP

Vedi la pianificazione delle risorse aziendali.

analisi esplorativa dei dati (EDA)

Il processo di analisi di un set di dati per comprenderne le caratteristiche principali. Si raccolgono o si aggregano dati e quindi si eseguono indagini iniziali per trovare modelli, rilevare anomalie e verificare ipotesi. L'EDA viene eseguita calcolando statistiche di riepilogo e creando visualizzazioni di dati.

F

tabella dei fatti

Il tavolo centrale in uno schema a stella. Memorizza dati quantitativi sulle operazioni aziendali. In genere, una tabella dei fatti contiene due tipi di colonne: quelle che contengono misure e quelle che contengono una chiave esterna per una tabella di dimensioni.

fallire velocemente

Una filosofia che utilizza test frequenti e incrementali per ridurre il ciclo di vita dello sviluppo. È una parte fondamentale di un approccio agile.

limite di isolamento dei guasti

Nel Cloud AWS, un limite come una zona di disponibilità Regione AWS, un piano di controllo o un piano dati che limita l'effetto di un errore e aiuta a migliorare la resilienza dei carichi di lavoro. Per ulteriori informazioni, consulta AWS Fault Isolation Boundaries.

ramo di funzionalità

Vedi filiale.

caratteristiche

I dati di input che usi per fare una previsione. Ad esempio, in un contesto di produzione, le caratteristiche potrebbero essere immagini acquisite periodicamente dalla linea di produzione.

importanza delle caratteristiche

Quanto è importante una caratteristica per le previsioni di un modello. Di solito viene espresso come punteggio numerico che può essere calcolato con varie tecniche, come Shapley Additive Explanations (SHAP) e gradienti integrati. Per ulteriori informazioni, vedere Interpretabilità del modello di machine learning con:AWS.

trasformazione delle funzionalità

Per ottimizzare i dati per il processo di machine learning, incluso l'arricchimento dei dati con fonti aggiuntive, il dimensionamento dei valori o l'estrazione di più set di informazioni da un singolo campo di dati. Ciò consente al modello di ML di trarre vantaggio dai dati. Ad esempio, se suddividi la data "2021-05-27 00:15:37" in "2021", "maggio", "giovedì" e "15", puoi aiutare l'algoritmo di apprendimento ad apprendere modelli sfumati associati a diversi componenti dei dati.

FGAC

Vedi il controllo granulare degli accessi.

controllo granulare degli accessi (FGAC)

L'uso di più condizioni per consentire o rifiutare una richiesta di accesso.

migrazione flash-cut

Un metodo di migrazione del database che utilizza la replica continua dei dati tramite l'acquisizione dei dati delle modifiche per migrare i dati nel più breve tempo possibile, anziché utilizzare un approccio graduale. L'obiettivo è ridurre al minimo i tempi di inattività.

G

blocco geografico

Vedi restrizioni geografiche.

limitazioni geografiche (blocco geografico)

In Amazon CloudFront, un'opzione per impedire agli utenti di determinati paesi di accedere alle distribuzioni di contenuti. Puoi utilizzare un elenco consentito o un elenco di blocco per specificare i paesi approvati e vietati. Per ulteriori informazioni, consulta Limitare la distribuzione geografica dei contenuti nella CloudFront documentazione.

Flusso di lavoro di GitFlow

Un approccio in cui gli ambienti inferiori e superiori utilizzano rami diversi in un repository di codice di origine. Il flusso di lavoro Gitflow è considerato obsoleto e il flusso di lavoro basato su trunk è l'approccio moderno e preferito.

strategia greenfield

L'assenza di infrastrutture esistenti in un nuovo ambiente. Quando si adotta una strategia greenfield per un'architettura di sistema, è possibile selezionare tutte le nuove tecnologie senza il vincolo della compatibilità con l'infrastruttura esistente, nota anche come brownfield. Per l'espansione dell'infrastruttura esistente, è possibile combinare strategie brownfield e greenfield.

guardrail

Una regola di livello elevato che consente di governare risorse, policy e conformità tra le unità organizzative (OU). I guardrail preventivi applicano le policy per garantire l'allineamento agli standard di conformità. Vengono implementati utilizzando le policy di controllo dei servizi e i limiti delle autorizzazioni IAM. I guardrail di rilevamento rilevano le violazioni delle policy e i problemi di conformità e generano avvisi per porvi rimedio. Sono implementati utilizzando Amazon AWS Config AWS Security Hub GuardDuty AWS Trusted Advisor, Amazon Inspector e controlli personalizzati AWS Lambda .

H

AH

Vedi disponibilità elevata.

migrazione di database eterogenea

Migrazione del database di origine in un database di destinazione che utilizza un motore di database diverso (ad esempio, da Oracle ad Amazon Aurora). La migrazione eterogenea fa in genere parte di uno sforzo di riprogettazione e la conversione dello schema può essere un'attività complessa. AWS offre AWS SCT che aiuta con le conversioni dello schema.

alta disponibilità (HA)

La capacità di un carico di lavoro di funzionare in modo continuo, senza intervento, in caso di sfide o disastri. I sistemi HA sono progettati per il failover automatico, fornire costantemente prestazioni di alta qualità e gestire carichi e guasti diversi con un impatto minimo sulle prestazioni.

modernizzazione storica

Un approccio utilizzato per modernizzare e aggiornare i sistemi di tecnologia operativa (OT) per soddisfare meglio le esigenze dell'industria manifatturiera. Uno storico è un tipo di database utilizzato per raccogliere e archiviare dati da varie fonti in una fabbrica.

migrazione di database omogenea

Migrazione del database di origine in un database di destinazione che condivide lo stesso motore di database (ad esempio, da Microsoft SQL Server ad Amazon RDS per SQL Server). La migrazione omogenea fa in genere parte di un'operazione di rehosting o ridefinizione della piattaforma. Per migrare lo schema è possibile utilizzare le utilità native del database.

dati caldi

Dati a cui si accede frequentemente, ad esempio dati in tempo reale o dati di traduzione recenti. Questi dati richiedono in genere un livello o una classe di storage ad alte prestazioni per fornire risposte rapide alle query.

hotfix

Una soluzione urgente per un problema critico in un ambiente di produzione. A causa della sua urgenza, un hotfix viene in genere creato al di fuori del tipico DevOps flusso di lavoro di rilascio.

periodo di hypercare

Subito dopo la conversione, il periodo di tempo in cui un team di migrazione gestisce e monitora le applicazioni migrate nel cloud per risolvere eventuali problemi. In genere, questo periodo dura da 1 a 4 giorni. Al termine del periodo di hypercare, il team addetto alla migrazione in genere trasferisce la responsabilità delle applicazioni al team addetto alle operazioni cloud.

I

IaC

Considera l'infrastruttura come codice.

Policy basata su identità

Una policy associata a uno o più principi IAM che definisce le relative autorizzazioni all'interno dell' Cloud AWS ambiente.

applicazione inattiva

Un'applicazione che prevede un uso di CPU e memoria medio compreso tra il 5% e il 20% in un periodo di 90 giorni. In un progetto di migrazione, è normale ritirare queste applicazioni o mantenerle on-premise.

IIoT

Vedi Industrial Internet of Things.

infrastruttura immutabile

Un modello che implementa una nuova infrastruttura per i carichi di lavoro di produzione anziché aggiornare, applicare patch o modificare l'infrastruttura esistente. Le infrastrutture immutabili sono intrinsecamente più coerenti, affidabili e prevedibili delle infrastrutture mutabili. Per ulteriori informazioni, consulta la best practice Deploy using immutable infrastructure in Well-Architected AWS Framework.

VPC in ingresso (ingress)

In un'architettura AWS multi-account, un VPC che accetta, ispeziona e indirizza le connessioni di rete dall'esterno di un'applicazione. Nel documento Architettura di riferimento per la sicurezza di AWS si consiglia di configurare l'account di rete con VPC in entrata, in uscita e di ispezione per proteggere l'interfaccia bidirezionale tra l'applicazione e Internet in generale.

migrazione incrementale

Una strategia di conversione in cui si esegue la migrazione dell'applicazione in piccole parti anziché eseguire una conversione singola e completa. Ad esempio, inizialmente potresti spostare solo alcuni microservizi o utenti nel nuovo sistema. Dopo aver verificato che tutto funzioni correttamente, puoi spostare in modo incrementale microservizi o utenti aggiuntivi fino alla disattivazione del sistema legacy. Questa strategia riduce i rischi associati alle migrazioni di grandi dimensioni.

Industria 4.0

Un termine introdotto da Klaus Schwab nel 2016 per riferirsi alla modernizzazione dei processi di produzione attraverso progressi in termini di connettività, dati in tempo reale, automazione, analisi e AI/ML.

infrastruttura

Tutte le risorse e gli asset contenuti nell'ambiente di un'applicazione.

infrastruttura come codice (IaC)

Il processo di provisioning e gestione dell'infrastruttura di un'applicazione tramite un insieme di file di configurazione. Il processo IaC è progettato per aiutarti a centralizzare la gestione dell'infrastruttura, a standardizzare le risorse e a dimensionare rapidamente, in modo che i nuovi ambienti siano ripetibili, affidabili e coerenti.

Internet delle cose industriale (IIoT)

L'uso di sensori e dispositivi connessi a Internet nei settori industriali, come quello manifatturiero, energetico, automobilistico, sanitario, delle scienze della vita e dell'agricoltura. Per ulteriori informazioni, consulta Creazione di una strategia di trasformazione digitale dell'Internet delle cose industriale (IIoT).

VPC di ispezione

In un'architettura AWS multi-account, un VPC centralizzato che gestisce le ispezioni del traffico di rete tra VPC (uguali o diversi Regioni AWS), Internet e reti locali. Nel documento Architettura di riferimento per la sicurezza di AWS si consiglia di configurare l'account di rete con VPC in entrata, in uscita e di ispezione per proteggere l'interfaccia bidirezionale tra l'applicazione e Internet in generale.

Internet of Things (IoT)

La rete di oggetti fisici connessi con sensori o processori incorporati che comunicano con altri dispositivi e sistemi tramite Internet o una rete di comunicazione locale. Per ulteriori informazioni, consulta Cos'è l'IoT?

interpretabilità

Una caratteristica di un modello di machine learning che descrive il grado in cui un essere umano è in grado di comprendere in che modo le previsioni del modello dipendono dai suoi input. Per ulteriori informazioni, consulta la sezione Interpretabilità dei modelli di machine learning con AWS.

IoT

Vedi Internet of Things.

libreria di informazioni IT (ITIL)

Una serie di best practice per offrire servizi IT e allinearli ai requisiti aziendali. ITIL fornisce le basi per ITSM.

gestione dei servizi IT (ITSM)

Attività associate alla progettazione, implementazione, gestione e supporto dei servizi IT per un'organizzazione. Per informazioni sull'integrazione delle operazioni cloud con gli strumenti ITSM, consulta la guida all'integrazione delle operazioni.

ITIL

Vedi la libreria di informazioni IT.

ITSM

Vedi Gestione dei servizi IT.

L

controllo degli accessi basato su etichette (LBAC)

Un'implementazione del controllo di accesso obbligatorio (MAC) in cui agli utenti e ai dati stessi viene assegnato esplicitamente un valore di etichetta di sicurezza. L'intersezione tra l'etichetta di sicurezza utente e l'etichetta di sicurezza dei dati determina quali righe e colonne possono essere visualizzate dall'utente.

zona di destinazione

Una landing zone è un AWS ambiente multi-account ben progettato, scalabile e sicuro. Questo è un punto di partenza dal quale le organizzazioni possono avviare e distribuire rapidamente carichi di lavoro e applicazioni con fiducia nel loro ambiente di sicurezza e infrastruttura. Per ulteriori informazioni sulle zone di destinazione, consulta la sezione Configurazione di un ambiente AWS multi-account sicuro e scalabile.

migrazione su larga scala

Una migrazione di 300 o più server.

BIANCO

Vedi controllo degli accessi basato su etichette.

Privilegio minimo

La best practice di sicurezza per la concessione delle autorizzazioni minime richieste per eseguire un'attività. Per ulteriori informazioni, consulta Applicazione delle autorizzazioni del privilegio minimo nella documentazione di IAM.

eseguire il rehosting (lift and shift)

Vedi 7 R.

sistema little-endian

Un sistema che memorizza per primo il byte meno importante. Vedi anche endianità.

ambienti inferiori

Vedi ambiente.

M

machine learning (ML)

Un tipo di intelligenza artificiale che utilizza algoritmi e tecniche per il riconoscimento e l'apprendimento di schemi. Il machine learning analizza e apprende dai dati registrati, come i dati dell'Internet delle cose (IoT), per generare un modello statistico basato su modelli. Per ulteriori informazioni, consulta la sezione Machine learning.

ramo principale

Vedi filiale.

malware

Software progettato per compromettere la sicurezza o la privacy del computer. Il malware potrebbe interrompere i sistemi informatici, divulgare informazioni sensibili o ottenere accessi non autorizzati. Esempi di malware includono virus, worm, ransomware, trojan horse, spyware e keylogger.

servizi gestiti

AWS servizi per cui AWS gestisce il livello di infrastruttura, il sistema operativo e le piattaforme e si accede agli endpoint per archiviare e recuperare i dati. Amazon Simple Storage Service (Amazon S3) Simple Storage Service (Amazon S3) e Amazon DynamoDB sono esempi di servizi gestiti. Questi sono noti anche come servizi astratti.

sistema di esecuzione della produzione (MES)

Un sistema software per tracciare, monitorare, documentare e controllare i processi di produzione che convertono le materie prime in prodotti finiti in officina.

MAP

Vedi Migration Acceleration Program.

meccanismo

Un processo completo in cui si crea uno strumento, si promuove l'adozione dello strumento e quindi si esaminano i risultati per apportare le modifiche. Un meccanismo è un ciclo che si rafforza e si migliora man mano che funziona. Per ulteriori informazioni, consulta Creazione di meccanismi nel AWS Well-Architected Framework.

account membro

Tutti gli account Account AWS diversi dall'account di gestione che fanno parte di un'organizzazione in. AWS Organizations Un account può essere membro di una sola organizzazione alla volta.

MEH

Vedi sistema di esecuzione della produzione.

Message Queuing Telemetry Transport (MQTT)

Un protocollo di comunicazione machine-to-machine (M2M) leggero, basato sul modello di pubblicazione/sottoscrizione, per dispositivi IoT con risorse limitate.

microservizio

Un piccolo servizio indipendente che comunica tramite API ben definite ed è in genere di proprietà di piccoli team autonomi. Ad esempio, un sistema assicurativo potrebbe includere microservizi che si riferiscono a funzionalità aziendali, come vendite o marketing, o sottodomini, come acquisti, reclami o analisi. I vantaggi dei microservizi includono agilità, dimensionamento flessibile, facilità di implementazione, codice riutilizzabile e resilienza. Per ulteriori informazioni, consulta Integrazione dei microservizi utilizzando servizi serverless. AWS

architettura di microservizi

Un approccio alla creazione di un'applicazione con componenti indipendenti che eseguono ogni processo applicativo come microservizio. Questi microservizi comunicano tramite un'interfaccia ben definita utilizzando API leggere. Ogni microservizio in questa architettura può essere aggiornato, distribuito e dimensionato per soddisfare la richiesta di funzioni specifiche di un'applicazione. Per ulteriori informazioni, vedere Implementazione dei microservizi su. AWS

Programma di accelerazione della migrazione (MAP)

Un AWS programma che fornisce consulenza, supporto, formazione e servizi per aiutare le organizzazioni a costruire una solida base operativa per il passaggio al cloud e per contribuire a compensare il costo iniziale delle migrazioni. MAP include una metodologia di migrazione per eseguire le migrazioni precedenti in modo metodico e un set di strumenti per automatizzare e accelerare gli scenari di migrazione comuni.

migrazione su larga scala

Il processo di trasferimento della maggior parte del portfolio di applicazioni sul cloud avviene a ondate, con più applicazioni trasferite a una velocità maggiore in ogni ondata. Questa fase utilizza le migliori pratiche e le lezioni apprese nelle fasi precedenti per implementare una fabbrica di migrazione di team, strumenti e processi per semplificare la migrazione dei carichi di lavoro attraverso l'automazione e la distribuzione agile. Questa è la terza fase della strategia di migrazione AWS.

fabbrica di migrazione

Team interfunzionali che semplificano la migrazione dei carichi di lavoro attraverso approcci automatizzati e agili. I team di Migration Factory includono in genere operazioni, analisti e proprietari aziendali, ingegneri addetti alla migrazione, sviluppatori e DevOps professionisti che lavorano nell'ambito degli sprint. Tra il 20% e il 50% di un portfolio di applicazioni aziendali è costituito da schemi ripetuti che possono essere ottimizzati con un approccio di fabbrica. Per ulteriori informazioni, consulta la discussione sulle fabbriche di migrazione e la Guida alla fabbrica di migrazione al cloud in questo set di contenuti.

metadati di migrazione

Le informazioni sull'applicazione e sul server necessarie per completare la migrazione. Ogni modello di migrazione richiede un set diverso di metadati di migrazione. Esempi di metadati di migrazione includono la sottorete, il gruppo di sicurezza e l'account di destinazione. AWS

modello di migrazione

Un'attività di migrazione ripetibile che descrive in dettaglio la strategia di migrazione, la destinazione della migrazione e l'applicazione o il servizio di migrazione utilizzati. Esempio: riorganizza la migrazione su Amazon EC2 AWS con Application Migration Service.

Valutazione del portfolio di migrazione (MPA)

Uno strumento online che fornisce informazioni per la convalida del business case per la migrazione a. Cloud AWS MPA offre una valutazione dettagliata del portfolio (dimensionamento

corretto dei server, prezzi, confronto del TCO, analisi dei costi di migrazione) e pianificazione della migrazione (analisi e raccolta dei dati delle applicazioni, raggruppamento delle applicazioni, prioritizzazione delle migrazioni e pianificazione delle ondate). Lo strumento MPA (richiede l'accesso) è disponibile gratuitamente per tutti i AWS consulenti e i consulenti dei partner APN.

valutazione della preparazione alla migrazione (MRA)

Il processo di acquisizione di informazioni sullo stato di preparazione al cloud di un'organizzazione, l'identificazione dei punti di forza e di debolezza e la creazione di un piano d'azione per colmare le lacune identificate, utilizzando il CAF. AWS Per ulteriori informazioni, consulta la guida di preparazione alla migrazione. MRA è la prima fase della strategia di migrazione AWS.

strategia di migrazione

L'approccio utilizzato per migrare un carico di lavoro verso. Cloud AWS Per ulteriori informazioni, consulta la voce 7 R in questo glossario e consulta Mobilita la tua organizzazione per accelerare le migrazioni su larga scala.

ML

Vedi machine learning.

modernizzazione

Trasformazione di un'applicazione obsoleta (legacy o monolitica) e della relativa infrastruttura in un sistema agile, elastico e altamente disponibile nel cloud per ridurre i costi, aumentare l'efficienza e sfruttare le innovazioni. Per ulteriori informazioni, vedere Strategia per la modernizzazione delle applicazioni in. Cloud AWS

valutazione della preparazione alla modernizzazione

Una valutazione che aiuta a determinare la preparazione alla modernizzazione delle applicazioni di un'organizzazione, identifica vantaggi, rischi e dipendenze e determina in che misura l'organizzazione può supportare lo stato futuro di tali applicazioni. Il risultato della valutazione è uno schema dell'architettura di destinazione, una tabella di marcia che descrive in dettaglio le fasi di sviluppo e le tappe fondamentali del processo di modernizzazione e un piano d'azione per colmare le lacune identificate. Per ulteriori informazioni, vedere Valutazione della preparazione alla modernizzazione per le applicazioni in. Cloud AWS

applicazioni monolitiche (monoliti)

Applicazioni eseguite come un unico servizio con processi strettamente collegati. Le applicazioni monolitiche presentano diversi inconvenienti. Se una funzionalità dell'applicazione registra un

picco di domanda, l'intera architettura deve essere dimensionata. L'aggiunta o il miglioramento delle funzionalità di un'applicazione monolitica diventa inoltre più complessa man mano che la base di codice cresce. Per risolvere questi problemi, puoi utilizzare un'architettura di microservizi. Per ulteriori informazioni, consulta la sezione Scomposizione dei monoliti in microservizi.

MAPPA

Vedi Migration Portfolio Assessment.

MQTT

Vedi Message Queuing Telemetry Transport.

classificazione multiclasse

Un processo che aiuta a generare previsioni per più classi (prevedendo uno o più di due risultati). Ad esempio, un modello di machine learning potrebbe chiedere "Questo prodotto è un libro, un'auto o un telefono?" oppure "Quale categoria di prodotti è più interessante per questo cliente?"

infrastruttura mutabile

Un modello che aggiorna e modifica l'infrastruttura esistente per i carichi di lavoro di produzione. Per migliorare la coerenza, l'affidabilità e la prevedibilità, il AWS Well-Architected Framework consiglia l'uso di un'infrastruttura immutabile come best practice.

O

OAC

Vedi Origin Access Control.

QUERCIA

Vedi Origin Access Identity.

OCM

Vedi gestione delle modifiche organizzative.

migrazione offline

Un metodo di migrazione in cui il carico di lavoro di origine viene eliminato durante il processo di migrazione. Questo metodo prevede tempi di inattività prolungati e viene in genere utilizzato per carichi di lavoro piccoli e non critici.

OI

Vedi l'integrazione delle operazioni.

OLA

Vedi accordo a livello operativo.

migrazione online

Un metodo di migrazione in cui il carico di lavoro di origine viene copiato sul sistema di destinazione senza essere messo offline. Le applicazioni connesse al carico di lavoro possono continuare a funzionare durante la migrazione. Questo metodo comporta tempi di inattività pari a zero o comunque minimi e viene in genere utilizzato per carichi di lavoro di produzione critici.

OPC-UA

Vedi Open Process Communications - Unified Architecture.

Comunicazioni a processo aperto - Architettura unificata (OPC-UA)

Un protocollo di comunicazione machine-to-machine (M2M) per l'automazione industriale. OPC-UA fornisce uno standard di interoperabilità con schemi di crittografia, autenticazione e autorizzazione dei dati.

accordo a livello operativo (OLA)

Un accordo che chiarisce quali sono gli impegni reciproci tra i gruppi IT funzionali, a supporto di un accordo sul livello di servizio (SLA).

revisione della prontezza operativa (ORR)

Un elenco di domande e best practice associate che aiutano a comprendere, valutare, prevenire o ridurre la portata degli incidenti e dei possibili guasti. Per ulteriori informazioni, vedere Operational Readiness Reviews (ORR) nel Well-Architected AWS Framework.

tecnologia operativa (OT)

Sistemi hardware e software che interagiscono con l'ambiente fisico per controllare le operazioni, le apparecchiature e le infrastrutture industriali. Nella produzione, l'integrazione di sistemi OT e di tecnologia dell'informazione (IT) è un obiettivo chiave per le trasformazioni dell'Industria 4.0.

integrazione delle operazioni (OI)

Il processo di modernizzazione delle operazioni nel cloud, che prevede la pianificazione, l'automazione e l'integrazione della disponibilità. Per ulteriori informazioni, consulta la guida all'integrazione delle operazioni.

trail organizzativo

Un percorso creato da noi AWS CloudTrail che registra tutti gli eventi di un'organizzazione per tutti Account AWS . AWS Organizations Questo percorso viene creato in ogni Account AWS che fa parte dell'organizzazione e tiene traccia dell'attività in ogni account. Per ulteriori informazioni, consulta Creazione di un percorso per un'organizzazione nella CloudTrail documentazione.

gestione del cambiamento organizzativo (OCM)

Un framework per la gestione di trasformazioni aziendali importanti e che comportano l'interruzione delle attività dal punto di vista delle persone, della cultura e della leadership. OCM aiuta le organizzazioni a prepararsi e passare a nuovi sistemi e strategie accelerando l'adozione del cambiamento, affrontando i problemi di transizione e promuovendo cambiamenti culturali e organizzativi. Nella strategia di AWS migrazione, questo framework si chiama accelerazione delle persone, a causa della velocità di cambiamento richiesta nei progetti di adozione del cloud. Per ulteriori informazioni, consultare la Guida OCM.

controllo dell'accesso all'origine (OAC)

In CloudFront, un'opzione avanzata per limitare l'accesso per proteggere i contenuti di Amazon Simple Storage Service (Amazon S3). OAC supporta tutti i bucket S3 in generale Regioni AWS, la crittografia lato server con AWS KMS (SSE-KMS) e le richieste dinamiche e dirette al bucket S3. `PUT` `DELETE`

identità di accesso origine (OAI)

Nel CloudFront, un'opzione per limitare l'accesso per proteggere i tuoi contenuti Amazon S3. Quando usi OAI, CloudFront crea un principale con cui Amazon S3 può autenticarsi. I principali autenticati possono accedere ai contenuti in un bucket S3 solo tramite una distribuzione specifica. CloudFront Vedi anche OAC, che fornisce un controllo degli accessi più granulare e avanzato.

O

Vedi la revisione della prontezza operativa.

- NON

Vedi la tecnologia operativa.

VPC in uscita (egress)

In un'architettura AWS multi-account, un VPC che gestisce le connessioni di rete avviate dall'interno di un'applicazione. Nel documento Architettura di riferimento per la sicurezza di

AWS si consiglia di configurare l'account di rete con VPC in entrata, in uscita e di ispezione per proteggere l'interfaccia bidirezionale tra l'applicazione e Internet in generale.

P

limite delle autorizzazioni

Una policy di gestione IAM collegata ai principali IAM per impostare le autorizzazioni massime che l'utente o il ruolo possono avere. Per ulteriori informazioni, consulta Limiti delle autorizzazioni nella documentazione di IAM.

informazioni di identificazione personale (PII)

Informazioni che, se visualizzate direttamente o abbinate ad altri dati correlati, possono essere utilizzate per dedurre ragionevolmente l'identità di un individuo. Esempi di informazioni personali includono nomi, indirizzi e informazioni di contatto.

Informazioni che consentono l'identificazione personale degli utenti

Visualizza le informazioni di identificazione personale.

playbook

Una serie di passaggi predefiniti che raccolgono il lavoro associato alle migrazioni, come l'erogazione delle funzioni operative principali nel cloud. Un playbook può assumere la forma di script, runbook automatici o un riepilogo dei processi o dei passaggi necessari per gestire un ambiente modernizzato.

PLC

Vedi controllore logico programmabile.

PLM

Vedi la gestione del ciclo di vita del prodotto.

policy

Un oggetto in grado di definire le autorizzazioni (vedi politica basata sull'identità), specificare le condizioni di accesso (vedi politicabasata sulle risorse) o definire le autorizzazioni massime per tutti gli account di un'organizzazione in (vedi politica di controllo dei servizi). AWS Organizations

persistenza poliglotta

Scelta indipendente della tecnologia di archiviazione di dati di un microservizio in base ai modelli di accesso ai dati e ad altri requisiti. Se i microservizi utilizzano la stessa tecnologia di archiviazione di dati, possono incontrare problemi di implementazione o registrare prestazioni scadenti. I microservizi vengono implementati più facilmente e ottengono prestazioni e scalabilità migliori se utilizzano l'archivio dati più adatto alle loro esigenze. Per ulteriori informazioni, consulta la sezione Abilitazione della persistenza dei dati nei microservizi.

valutazione del portfolio

Un processo di scoperta, analisi e definizione delle priorità del portfolio di applicazioni per pianificare la migrazione. Per ulteriori informazioni, consulta la pagina Valutazione della preparazione alla migrazione.

predicate

Una condizione di interrogazione che restituisce o, in genere, si trova in una clausola`true`. `false` `WHERE`

predicato pushdown

Una tecnica di ottimizzazione delle query del database che filtra i dati della query prima del trasferimento. Ciò riduce la quantità di dati che devono essere recuperati ed elaborati dal database relazionale e migliora le prestazioni delle query.

controllo preventivo

Un controllo di sicurezza progettato per impedire il verificarsi di un evento. Questi controlli sono la prima linea di difesa per impedire accessi non autorizzati o modifiche indesiderate alla rete. Per ulteriori informazioni, consulta Controlli preventivi in Implementazione dei controlli di sicurezza in AWS.

principale

Un'entità in AWS grado di eseguire azioni e accedere alle risorse. Questa entità è in genere un utente root per un Account AWS ruolo IAM o un utente. Per ulteriori informazioni, consulta Principali in Termini e concetti dei ruoli nella documentazione di IAM.

Privacy fin dalla progettazione

Un approccio all'ingegneria dei sistemi che tiene conto della privacy durante l'intero processo di progettazione.

zone ospitate private

Un container che contiene informazioni su come si desidera che Amazon Route 53 risponda alle query DNS per un dominio e i relativi sottodomini all'interno di uno o più VPC. Per ulteriori informazioni, consulta Utilizzo delle zone ospitate private nella documentazione di Route 53.

controllo proattivo

Un controllo di sicurezza progettato per impedire l'implementazione di risorse non conformi. Questi controlli analizzano le risorse prima del loro provisioning. Se la risorsa non è conforme al controllo, non viene fornita. Per ulteriori informazioni, consulta la guida di riferimento sui controlli nella AWS Control Tower documentazione e consulta Controlli proattivi in Implementazione dei controlli di sicurezza su. AWS

gestione del ciclo di vita del prodotto (PLM)

La gestione dei dati e dei processi di un prodotto durante l'intero ciclo di vita, dalla progettazione, sviluppo e lancio, attraverso la crescita e la maturità, fino al declino e alla rimozione.

Ambiente di produzione

Vedi ambiente.

controllore logico programmabile (PLC)

Nella produzione, un computer altamente affidabile e adattabile che monitora le macchine e automatizza i processi di produzione.

pseudonimizzazione

Il processo di sostituzione degli identificatori personali in un set di dati con valori segnaposto. La pseudonimizzazione può aiutare a proteggere la privacy personale. I dati pseudonimizzati sono ancora considerati dati personali.

pubblica/sottoscrivi (pub/sub)

Un pattern che consente comunicazioni asincrone tra microservizi per migliorare la scalabilità e la reattività. Ad esempio, in un MES basato su microservizi, un microservizio può pubblicare messaggi di eventi su un canale a cui altri microservizi possono abbonarsi. Il sistema può aggiungere nuovi microservizi senza modificare il servizio di pubblicazione.

Q

Piano di query

Una serie di passaggi, come le istruzioni, utilizzati per accedere ai dati in un sistema di database relazionale SQL.

regressione del piano di query

Quando un ottimizzatore del servizio di database sceglie un piano non ottimale rispetto a prima di una determinata modifica all'ambiente di database. Questo può essere causato da modifiche a statistiche, vincoli, impostazioni dell'ambiente, associazioni dei parametri di query e aggiornamenti al motore di database.

R

Matrice RACI

Vedi responsabile, responsabile, consultato, informato (RACI).

ransomware

Un software dannoso progettato per bloccare l'accesso a un sistema informatico o ai dati fino a quando non viene effettuato un pagamento.

Matrice RASCI

Vedi responsabile, responsabile, consultato, informato (RACI).

RCAC

Vedi controllo dell'accesso a righe e colonne.

replica di lettura

Una copia di un database utilizzata per scopi di sola lettura. È possibile indirizzare le query alla replica di lettura per ridurre il carico sul database principale.

riprogettare

Vedi 7 Rs.

obiettivo del punto di ripristino (RPO)

Il periodo di tempo massimo accettabile dall'ultimo punto di ripristino dei dati. Ciò determina quella che viene considerata una perdita di dati accettabile tra l'ultimo punto di ripristino e l'interruzione del servizio.

obiettivo del tempo di ripristino (RTO)

Il ritardo massimo accettabile tra l'interruzione del servizio e il ripristino del servizio.

rifattorizzare

Vedi 7 R.

Regione

Una raccolta di AWS risorse in un'area geografica. Ciascuna Regione AWS è isolata e indipendente dalle altre per fornire tolleranza agli errori, stabilità e resilienza. Per ulteriori informazioni, consulta Specificare cosa può usare Regioni AWS il tuo account.

regressione

Una tecnica di ML che prevede un valore numerico. Ad esempio, per risolvere il problema "A che prezzo verrà venduta questa casa?" un modello di ML potrebbe utilizzare un modello di regressione lineare per prevedere il prezzo di vendita di una casa sulla base di dati noti sulla casa (ad esempio, la metratura).

riospitare

Vedi 7 R.

rilascio

In un processo di implementazione, l'atto di promuovere modifiche a un ambiente di produzione.

trasferisco

Vedi 7 Rs.

ripiattaforma

Vedi 7 Rs.

riacquisto

Vedi 7 Rs.

resilienza

La capacità di un'applicazione di resistere o ripristinare le interruzioni. L'elevata disponibilità e il disaster recovery sono considerazioni comuni quando si pianifica la resilienza in. Cloud AWSPer ulteriori informazioni, vedere Cloud AWS Resilience.

policy basata su risorse

Una policy associata a una risorsa, ad esempio un bucket Amazon S3, un endpoint o una chiave di crittografia. Questo tipo di policy specifica a quali principali è consentito l'accesso, le azioni supportate e qualsiasi altra condizione che deve essere soddisfatta.

matrice di assegnazione di responsabilità (RACI)

Una matrice che definisce i ruoli e le responsabilità di tutte le parti coinvolte nelle attività di migrazione e nelle operazioni cloud. Il nome della matrice deriva dai tipi di responsabilità definiti nella matrice: responsabile (R), responsabile (A), consultato (C) e informato (I). Il tipo di supporto (S) è facoltativo. Se includi il supporto, la matrice viene chiamata matrice RASCI e, se la escludi, viene chiamata matrice RACI.

controllo reattivo

Un controllo di sicurezza progettato per favorire la correzione di eventi avversi o deviazioni dalla baseline di sicurezza. Per ulteriori informazioni, consulta Controlli reattivi in Implementazione dei controlli di sicurezza in AWS.

retain

Vedi 7 R.

andare in pensione

Vedi 7 Rs.

rotazione

Processo di aggiornamento periodico di un segreto per rendere più difficile l'accesso alle credenziali da parte di un utente malintenzionato.

controllo dell'accesso a righe e colonne (RCAC)

L'uso di espressioni SQL di base e flessibili con regole di accesso definite. RCAC è costituito da autorizzazioni di riga e maschere di colonna.

RPO

Vedi l'obiettivo del punto di ripristino.

RTO

Vedi l'obiettivo del tempo di ripristino.

runbook

Un insieme di procedure manuali o automatizzate necessarie per eseguire un'attività specifica. In genere sono progettati per semplificare operazioni o procedure ripetitive con tassi di errore elevati.

S

SAML 2.0

Uno standard aperto utilizzato da molti provider di identità (IdPs). Questa funzionalità abilita il single sign-on (SSO) federato, in modo che gli utenti possano accedere AWS Management Console o chiamare le operazioni AWS API senza che tu debba creare un utente in IAM per tutti i membri dell'organizzazione. Per ulteriori informazioni sulla federazione basata su SAML 2.0, consulta Informazioni sulla federazione basata su SAML 2.0 nella documentazione di IAM.

SCADA

Vedi controllo di supervisione e acquisizione dati.

SCP

Vedi la politica di controllo del servizio.

Secret

In AWS Secrets Manager, informazioni riservate o riservate, come una password o le credenziali utente, archiviate in forma crittografata. È costituito dal valore segreto e dai relativi metadati. Il valore segreto può essere binario, una stringa singola o più stringhe. Per ulteriori informazioni, consulta Cosa c'è in un segreto di Secrets Manager? nella documentazione di Secrets Manager.

controllo di sicurezza

Un guardrail tecnico o amministrativo che impedisce, rileva o riduce la capacità di un autore di minacce di sfruttare una vulnerabilità di sicurezza. Esistono quattro tipi principali di controlli di sicurezza: preventivi, investigativi, reattivi e proattivi.

rafforzamento della sicurezza

Il processo di riduzione della superficie di attacco per renderla più resistente agli attacchi. Può includere azioni come la rimozione di risorse che non sono più necessarie, l'implementazione di

best practice di sicurezza che prevedono la concessione del privilegio minimo o la disattivazione di funzionalità non necessarie nei file di configurazione.

sistema di gestione delle informazioni e degli eventi di sicurezza (SIEM)

Strumenti e servizi che combinano sistemi di gestione delle informazioni di sicurezza (SIM) e sistemi di gestione degli eventi di sicurezza (SEM). Un sistema SIEM raccoglie, monitora e analizza i dati da server, reti, dispositivi e altre fonti per rilevare minacce e violazioni della sicurezza e generare avvisi.

automazione della risposta alla sicurezza

Un'azione predefinita e programmata progettata per rispondere o porre rimedio automaticamente a un evento di sicurezza. Queste automazioni fungono da controlli di sicurezza investigativi o reattivi che aiutano a implementare le migliori pratiche di sicurezza. AWS Esempi di azioni di risposta automatizzate includono la modifica di un gruppo di sicurezza VPC, l'applicazione di patch a un'istanza Amazon EC2 o la rotazione delle credenziali.

Crittografia lato server

Crittografia dei dati a destinazione, da parte di chi li riceve. AWS servizio

Policy di controllo dei servizi (SCP)

Una policy che fornisce il controllo centralizzato sulle autorizzazioni per tutti gli account di un'organizzazione in AWS Organizations. Le SCP definiscono i guardrail o fissano i limiti alle azioni che un amministratore può delegare a utenti o ruoli. Puoi utilizzare le SCP come elenchi consentiti o elenchi di rifiuto, per specificare quali servizi o azioni sono consentiti o proibiti. Per ulteriori informazioni, consulta le politiche di controllo del servizio nella AWS Organizations documentazione.

endpoint del servizio

L'URL del punto di ingresso per un AWS servizio. Puoi utilizzare l'endpoint per connetterti a livello di programmazione al servizio di destinazione. Per ulteriori informazioni, consulta Endpoint del AWS servizio nei Riferimenti generali di AWS.

accordo sul livello di servizio (SLA)

Un accordo che chiarisce ciò che un team IT promette di offrire ai propri clienti, ad esempio l'operatività e le prestazioni del servizio.

indicatore del livello di servizio (SLI)

Misurazione di un aspetto prestazionale di un servizio, ad esempio il tasso di errore, la disponibilità o la velocità effettiva.

obiettivo a livello di servizio (SLO)

Una metrica target che rappresenta lo stato di un servizio, misurato da un indicatore del livello di servizio.

Modello di responsabilità condivisa

Un modello che descrive la responsabilità condivisa AWS per la sicurezza e la conformità del cloud. AWS è responsabile della sicurezza del cloud, mentre tu sei responsabile della sicurezza nel cloud. Per ulteriori informazioni, consulta Modello di responsabilità condivisa.

SIEM

Vedi il sistema di gestione delle informazioni e degli eventi sulla sicurezza.

punto di errore singolo (SPOF)

Un guasto in un singolo componente critico di un'applicazione che può disturbare il sistema.

SLAM

Vedi il contratto sul livello di servizio.

SLI

Vedi l'indicatore del livello di servizio.

LENTA

Vedi obiettivo del livello di servizio.

split-and-seed modello

Un modello per dimensionare e accelerare i progetti di modernizzazione. Man mano che vengono definite nuove funzionalità e versioni dei prodotti, il team principale si divide per creare nuovi team di prodotto. Questo aiuta a dimensionare le capacità e i servizi dell'organizzazione, migliora la produttività degli sviluppatori e supporta una rapida innovazione. Per ulteriori informazioni, vedere Approccio graduale alla modernizzazione delle applicazioni in. Cloud AWS

SPOF

Vedi punto di errore singolo.

schema a stella

Una struttura organizzativa di database che utilizza un'unica tabella dei fatti di grandi dimensioni per archiviare i dati transazionali o misurati e utilizza una o più tabelle dimensionali più piccole per memorizzare gli attributi dei dati. Questa struttura è progettata per l'uso in un data warehouse o per scopi di business intelligence.

modello del fico strangolatore

Un approccio alla modernizzazione dei sistemi monolitici mediante la riscrittura e la sostituzione incrementali delle funzionalità del sistema fino alla disattivazione del sistema legacy. Questo modello utilizza l'analogia di una pianta di fico che cresce fino a diventare un albero robusto e alla fine annienta e sostituisce il suo ospite. Il modello è stato introdotto da Martin Fowler come metodo per gestire il rischio durante la riscrittura di sistemi monolitici. Per un esempio di come applicare questo modello, consulta Modernizzazione incrementale dei servizi Web legacy di Microsoft ASP.NET (ASMX) mediante container e Gateway Amazon API.

sottorete

Un intervallo di indirizzi IP nel VPC. Una sottorete deve risiedere in una singola zona di disponibilità.

controllo di supervisione e acquisizione dati (SCADA)

Nella produzione, un sistema che utilizza hardware e software per monitorare gli asset fisici e le operazioni di produzione.

crittografia simmetrica

Un algoritmo di crittografia che utilizza la stessa chiave per crittografare e decrittografare i dati.

test sintetici

Test di un sistema in modo da simulare le interazioni degli utenti per rilevare potenziali problemi o monitorare le prestazioni. Puoi usare Amazon CloudWatch Synthetics per creare questi test.

T

tags

Coppie chiave-valore che fungono da metadati per l'organizzazione delle risorse. AWS Con i tag è possibile a gestire, identificare, organizzare, cercare e filtrare le risorse. Per ulteriori informazioni, consulta Tagging delle risorse AWS.

variabile di destinazione

Il valore che stai cercando di prevedere nel machine learning supervisionato. Questo è indicato anche come variabile di risultato. Ad esempio, in un ambiente di produzione la variabile di destinazione potrebbe essere un difetto del prodotto.

elenco di attività

Uno strumento che viene utilizzato per tenere traccia dei progressi tramite un runbook. Un elenco di attività contiene una panoramica del runbook e un elenco di attività generali da completare. Per ogni attività generale, include la quantità stimata di tempo richiesta, il proprietario e lo stato di avanzamento.

Ambiente di test

Vedi ambiente.

training

Fornire dati da cui trarre ispirazione dal modello di machine learning. I dati di training devono contenere la risposta corretta. L'algoritmo di apprendimento trova nei dati di addestramento i pattern che mappano gli attributi dei dati di input al target (la risposta che si desidera prevedere). Produce un modello di ML che acquisisce questi modelli. Puoi quindi utilizzare il modello di ML per creare previsioni su nuovi dati di cui non si conosce il target.

Transit Gateway

Un hub di transito di rete che è possibile utilizzare per collegare i VPC e le reti on-premise. Per ulteriori informazioni, consulta Cos'è un gateway di transito nella AWS Transit Gateway documentazione.

flusso di lavoro basato su trunk

Un approccio in cui gli sviluppatori creano e testano le funzionalità localmente in un ramo di funzionalità e quindi uniscono tali modifiche al ramo principale. Il ramo principale viene quindi integrato negli ambienti di sviluppo, preproduzione e produzione, in sequenza.

Accesso attendibile

Concessione delle autorizzazioni a un servizio specificato dall'utente per eseguire attività all'interno dell'organizzazione AWS Organizations e nei suoi account per conto dell'utente. Il servizio attendibile crea un ruolo collegato al servizio in ogni account, quando tale ruolo è necessario, per eseguire attività di gestione per conto dell'utente. Per ulteriori informazioni,

consulta Utilizzo AWS Organizations con altri AWS servizi nella AWS Organizations documentazione.

regolazione

Modificare alcuni aspetti del processo di training per migliorare la precisione del modello di ML. Ad esempio, puoi addestrare il modello di ML generando un set di etichette, aggiungendo etichette e quindi ripetendo questi passaggi più volte con impostazioni diverse per ottimizzare il modello.

team da due pizze

Una piccola DevOps squadra che puoi sfamare con due pizze. Un team composto da due persone garantisce la migliore opportunità possibile di collaborazione nello sviluppo del software.

U

incertezza

Un concetto che si riferisce a informazioni imprecise, incomplete o sconosciute che possono minare l'affidabilità dei modelli di machine learning predittivi. Esistono due tipi di incertezza: l'incertezza epistemica, che è causata da dati limitati e incompleti, mentre l'incertezza aleatoria è causata dal rumore e dalla casualità insiti nei dati. Per ulteriori informazioni, consulta la guida Quantificazione dell'incertezza nei sistemi di deep learning.

compiti indifferenziati

Conosciuto anche come sollevamento di carichi pesanti, è un lavoro necessario per creare e far funzionare un'applicazione, ma che non apporta valore diretto all'utente finale né offre vantaggi competitivi. Esempi di attività indifferenziate includono l'approvvigionamento, la manutenzione e la pianificazione della capacità.

ambienti superiori

Vedi ambiente.

V

vacuum

Un'operazione di manutenzione del database che prevede la pulizia dopo aggiornamenti incrementali per recuperare lo spazio di archiviazione e migliorare le prestazioni.

controllo delle versioni

Processi e strumenti che tengono traccia delle modifiche, ad esempio le modifiche al codice di origine in un repository.

Peering VPC

Una connessione tra due VPC che consente di instradare il traffico tramite indirizzi IP privati. Per ulteriori informazioni, consulta Che cos'è il peering VPC? nella documentazione di Amazon VPC.

vulnerabilità

Un difetto software o hardware che compromette la sicurezza del sistema.

W

cache calda

Una cache del buffer che contiene dati correnti e pertinenti a cui si accede frequentemente. L'istanza di database può leggere dalla cache del buffer, il che richiede meno tempo rispetto alla lettura dalla memoria dal disco principale.

dati caldi

Dati a cui si accede raramente. Quando si eseguono interrogazioni di questo tipo di dati, in genere sono accettabili interrogazioni moderatamente lente.

funzione finestra

Una funzione SQL che esegue un calcolo su un gruppo di righe che si riferiscono in qualche modo al record corrente. Le funzioni della finestra sono utili per l'elaborazione di attività, come il calcolo di una media mobile o l'accesso al valore delle righe in base alla posizione relativa della riga corrente.

Carico di lavoro

Una raccolta di risorse e codice che fornisce valore aziendale, ad esempio un'applicazione rivolta ai clienti o un processo back-end.

flusso di lavoro

Gruppi funzionali in un progetto di migrazione responsabili di una serie specifica di attività. Ogni flusso di lavoro è indipendente ma supporta gli altri flussi di lavoro del progetto. Ad esempio,

il flusso di lavoro del portfolio è responsabile della definizione delle priorità delle applicazioni, della pianificazione delle ondate e della raccolta dei metadati di migrazione. Il flusso di lavoro del portfolio fornisce queste risorse al flusso di lavoro di migrazione, che quindi migra i server e le applicazioni.

VERME

Vedi scrivere una volta, leggere molti.

WQF

Vedi AWS Workload Qualification Framework.

scrivi una volta, leggi molte (WORM)

Un modello di storage che scrive i dati una sola volta e ne impedisce l'eliminazione o la modifica. Gli utenti autorizzati possono leggere i dati tutte le volte che è necessario, ma non possono modificarli. Questa infrastruttura di archiviazione dei dati è considerata immutabile.

Z

exploit zero-day

Un attacco, in genere malware, che sfrutta una vulnerabilità zero-day.

vulnerabilità zero-day

Un difetto o una vulnerabilità assoluta in un sistema di produzione. Gli autori delle minacce possono utilizzare questo tipo di vulnerabilità per attaccare il sistema. Gli sviluppatori vengono spesso a conoscenza della vulnerabilità causata dall'attacco.

applicazione zombie

Un'applicazione che prevede un utilizzo CPU e memoria inferiore al 5%. In un progetto di migrazione, è normale ritirare queste applicazioni.

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.