aws

Guida per gli sviluppatori

Amazon Timestream

Amazon Timestream: Guida per gli sviluppatori

Table of Contents

A cosa serve Amazon LiveAnalytics Timestream?

Amazon Timestream LiveAnalytics for è un database di serie temporali veloce, scalabile, completamente gestito e creato appositamente che semplifica l'archiviazione e l'analisi di trilioni di punti di dati di serie temporali al giorno. Timestream for LiveAnalytics consente di risparmiare tempo e costi nella gestione del ciclo di vita dei dati delle serie temporali conservando i dati recenti in memoria e spostando i dati storici su un livello di storage a costi ottimizzati in base a politiche definite dall'utente. Il motore LiveAnalytics di query appositamente progettato da Timestream for consente di accedere e analizzare insieme dati recenti e storici, senza doverne specificare la posizione. Amazon Timestream LiveAnalytics for dispone di funzioni integrate di analisi delle serie temporali, che ti aiutano a identificare tendenze e modelli nei tuoi dati quasi in tempo reale. Timestream for LiveAnalytics è serverless e si ridimensiona automaticamente verso l'alto o verso il basso per regolare la capacità e le prestazioni. Poiché non è necessario gestire l'infrastruttura sottostante, è possibile concentrarsi sull'ottimizzazione e sullo sviluppo delle applicazioni.

Timestream for si integra LiveAnalytics anche con i servizi di uso comune per la raccolta, la visualizzazione e l'apprendimento automatico dei dati. Puoi inviare dati ad Amazon Timestream LiveAnalytics per utilizzarli, Amazon Kinesis, AWS IoT Core MSK Amazon e Telegraf open source. Puoi visualizzare i dati utilizzando Amazon QuickSight, Grafana e strumenti di business intelligence tramite. JDBC Puoi anche usare Amazon SageMaker con Timestream LiveAnalytics per l'apprendimento automatico.

Timestream per i vantaggi principali LiveAnalytics

I vantaggi principali di Amazon Timestream LiveAnalytics for sono:

- Senza server con auto-scaling: con Amazon Timestream LiveAnalytics for, non ci sono server da gestire né capacità di provisioning. Man mano che le esigenze dell'applicazione cambiano, Timestream for si ridimensiona automaticamente per adattare la capacità. LiveAnalytics
- Gestione del ciclo di vita dei dati: Amazon Timestream semplifica il complesso processo di gestione del ciclo LiveAnalytics di vita dei dati. Offre lo storage su più livelli, con un archivio di memoria per i dati recenti e un archivio magnetico per i dati storici. Amazon Timestream automatizza il trasferimento dei dati dall'archivio di memoria all'archivio magnetico in base a politiche configurabili dall'utente.
- Accesso semplificato ai dati: con Amazon LiveAnalytics Timestream for, non è più necessario utilizzare strumenti diversi per accedere a dati recenti e storici. Il motore di query specifico di

Amazon Timestream LiveAnalytics for accede e combina in modo trasparente i dati su più livelli di storage senza che sia necessario specificare la posizione dei dati.

- Progettato appositamente per le serie temporali: puoi analizzare rapidamente i dati delle serie temporali utilizzando, con funzioni di serie temporali integrate per il livellamentoSQL, l'approssimazione e l'interpolazione. Timestream for supporta LiveAnalytics anche aggregati avanzati, funzioni di finestra e tipi di dati complessi come matrici e righe.
- Sempre crittografato: Amazon Timestream garantisce che LiveAnalytics i dati delle serie temporali siano sempre crittografati, sia a riposo che in transito. Amazon Timestream LiveAnalytics for consente inoltre di specificare AWS KMS una chiave gestita dal cliente CMK () per crittografare i dati nell'archivio magnetico.
- Alta disponibilità: Amazon Timestream garantisce un'elevata disponibilità delle richieste di scrittura e lettura replicando automaticamente i dati e allocando le risorse su almeno 3 diverse zone di disponibilità all'interno di una singola regione. AWS Per ulteriori informazioni, consulta il contratto sul livello di servizio di Timestream.
- Durabilità: Amazon Timestream garantisce la durabilità dei dati replicando automaticamente i dati della memoria e dell'archivio magnetico in diverse zone di disponibilità all'interno di una singola regione. AWS Tutti i dati vengono scritti su disco prima di confermare che la richiesta di scrittura è completa.

Timestream per i casi d'uso LiveAnalytics

Alcuni esempi di un elenco crescente di casi d'uso di Timestream for includono: LiveAnalytics

- Monitoraggio delle metriche per migliorare le prestazioni e la disponibilità delle applicazioni.
- Archiviazione e analisi della telemetria industriale per semplificare la gestione e la manutenzione delle apparecchiature.
- Monitoraggio dell'interazione dell'utente con un'applicazione nel tempo.
- Archiviazione e analisi dei dati dei sensori IoT.

Guida introduttiva a Timestream per LiveAnalytics

Consigliamo di iniziare leggendo le seguenti sezioni:

- Tutorial- Per creare un database popolato con set di dati di esempio ed eseguire query di esempio.

- Amazon LiveAnalytics Timestream per concetti- Per apprendere i concetti essenziali di Timestream. LiveAnalytics
- Accesso a Timestream per LiveAnalytics- Per sapere come accedere a Timestream per l' LiveAnalytics utilizzo della console, oppure. AWS CLI API
- Quote- Per informazioni sulle quote relative al numero di Timestream per i LiveAnalytics componenti di cui è possibile fornire.

Per informazioni su come iniziare rapidamente a sviluppare applicazioni per Timestream for LiveAnalytics, consulta quanto segue:

- Usando il AWS SDKs
- Riferimento al linguaggio di interrogazione

Come funziona

Le seguenti sezioni forniscono una panoramica dei componenti del servizio Amazon Timestream for Live Analytics e del modo in cui interagiscono.

Dopo aver letto questa introduzione, consulta le Accesso a Timestream per LiveAnalytics sezioni per scoprire come accedere a Timestream for Live Analytics utilizzando la console, oppure. AWS CLI SDKs

Argomenti

- Amazon LiveAnalytics Timestream per concetti
- Architettura
- Scrive
- Storage
- Query
- Interrogazioni pianificate
- Unità di calcolo Timestream () TCU

Amazon LiveAnalytics Timestream per concetti

I dati delle serie temporali sono una sequenza di punti dati registrati in un intervallo di tempo. Questo tipo di dati viene utilizzato per misurare eventi che cambiano nel tempo. Gli esempi includono quanto segue.

- Prezzi delle azioni nel tempo
- Misurazioni della temperatura nel tempo
- CPUutilizzo di un'EC2istanza nel tempo

Con i dati delle serie temporali, ogni punto dati è costituito da un timestamp, uno o più attributi e dall'evento che cambia nel tempo. Questi dati possono essere utilizzati per ricavare informazioni sulle prestazioni e sullo stato di un'applicazione, rilevare anomalie e identificare opportunità di ottimizzazione. Ad esempio, DevOps gli ingegneri potrebbero voler visualizzare i dati che misurano i cambiamenti nelle metriche delle prestazioni dell'infrastruttura. I produttori potrebbero voler tenere traccia dei dati dei sensori IoT che misurano i cambiamenti nelle apparecchiature all'interno di una struttura. I professionisti del marketing online potrebbero voler analizzare i dati clickstream che registrano il modo in cui un utente naviga su un sito Web nel tempo. Poiché i dati delle serie temporali vengono generati da più fonti in volumi estremamente elevati, devono essere raccolti in modo conveniente quasi in tempo reale e richiedono quindi uno storage efficiente che aiuti a organizzare e analizzare i dati.

Di seguito sono riportati i concetti chiave di Timestream for. LiveAnalytics

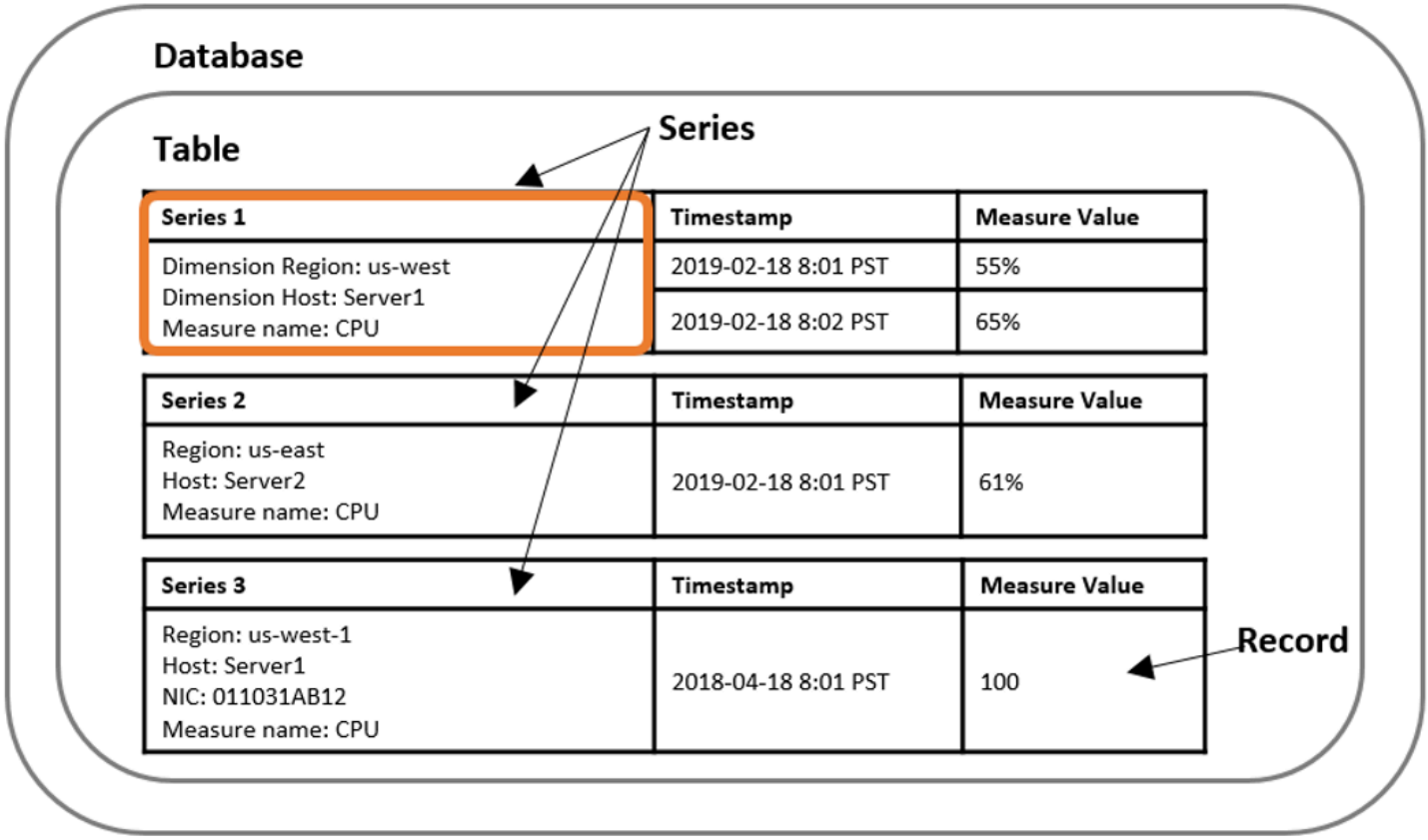
- Serie temporali - Una sequenza di uno o più punti dati (o record) registrati in un intervallo di tempo. Alcuni esempi sono il prezzo di un'azione nel tempo, l'CPUutilizzo della memoria di un'EC2istanza nel tempo e la lettura di temperatura/pressione di un sensore IoT nel tempo.
- Record: un singolo punto dati in una serie temporale.
- Dimensione: un attributo che descrive i metadati di una serie temporale. Una dimensione è composta da un nome di dimensione e un valore di dimensione. Considerare i seguenti esempi:
  - Quando si considera una borsa come dimensione, il nome della dimensione è «borsa» e il valore della dimensione è "» NYSE
  - Quando si considera una AWS regione come dimensione, il nome della dimensione è «regione» e il valore della dimensione è «us-east-1"
  - Per un sensore IoT, il nome della dimensione è «ID dispositivo» e il valore della dimensione è «12345"

- Misura: il valore effettivo misurato dal record. Alcuni esempi sono il prezzo delle azioni, l'CPUutilizzo della memoria e la lettura della temperatura o dell'umidità. Le misure sono costituite da nomi e valori di misura. Considerare i seguenti esempi:
  - Per il prezzo di un'azione, il nome della misura è «prezzo delle azioni» e il valore di misura è il prezzo effettivo delle azioni in un determinato momento.
  - Per CPU l'utilizzo, il nome della misura è "CPUutilizzo» e il valore della misura è l'utilizzo effettivoCPU.

  Le misure possono essere modellate in Timestream come record multimisura o a misura singola. LiveAnalytics Per ulteriori informazioni, consulta Record multimisura vs. record a misura singola.
- Timestamp: indica quando è stata raccolta una misura per un determinato record. Timestream for LiveAnalytics supporta timestamp con granularità di nanosecondi.
- Tabella: un contenitore per un insieme di serie temporali correlate.
- Database: un contenitore di primo livello per le tabelle.

Un riepilogo di Timestream for concepts LiveAnalytics

Un database contiene 0 o più tabelle. Ogni tabella contiene 0 o più serie temporali. Ogni serie temporale è costituita da una sequenza di record su un determinato intervallo di tempo con una granularità specificata. Ogni serie temporale può essere descritta utilizzando i relativi metadati o dimensioni, i dati o le misure e i relativi timestamp.

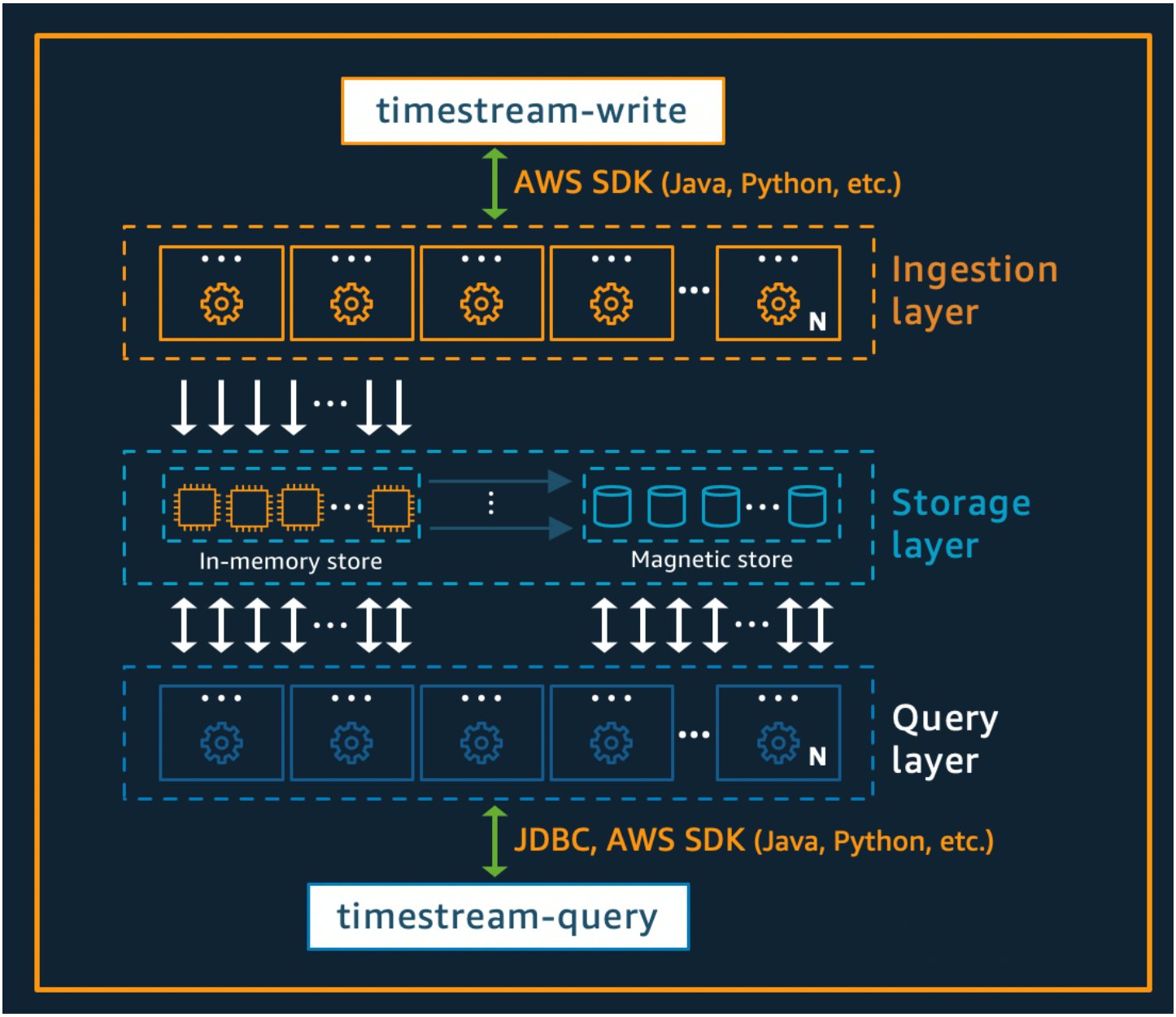


Architettura

Amazon Timestream for Live Analytics è stato progettato da zero per raccogliere, archiviare ed elaborare dati di serie temporali su larga scala. La sua architettura serverless supporta sistemi di acquisizione, archiviazione ed elaborazione delle query di dati completamente disaccoppiati, scalabili in modo indipendente. Questo design semplifica ogni sottosistema, facilitando il raggiungimento di un'affidabilità incrollabile, l'eliminazione dei colli di bottiglia legati alla scalabilità e la riduzione delle possibilità di guasti di sistema correlati. Ciascuno di questi fattori diventa più importante man mano che il sistema cresce.

Argomenti

- Scrivi l'architettura
- Architettura di storage
- Architettura delle interrogazioni
- Architettura cellulare



Scrivi l'architettura

Durante la scrittura di dati di serie temporali, Amazon Timestream for Live Analytics indirizza le scritture per una tabella, una partizione, a un'istanza di storage di memoria con tolleranza ai guasti che elabora scritture di dati ad alto throughput. L'archivio di memoria a sua volta raggiunge la durabilità in un sistema di storage separato che replica i dati su tre zone di disponibilità (). AZs La replica si basa sul quorum in modo tale che la perdita di nodi o di un'intera zona di disponibilità non comprometta la disponibilità di scrittura. Quasi in tempo reale, altri nodi di storage in memoria si sincronizzano con i dati per rispondere alle richieste. Il lettore replica anche i nodi, per garantire un'AZselevata disponibilità di lettura.

Timestream for Live Analytics supporta la scrittura dei dati direttamente nell'archivio magnetico, per applicazioni che generano dati in arrivo tardivo con un throughput inferiore. I dati in arrivo tardivo sono dati con un timestamp precedente all'ora corrente. Analogamente alle scritture ad alta velocità nell'archivio di memoria, i dati scritti nell'archivio magnetico vengono replicati su tre unità AZs e la replica si basa sul quorum.

Indipendentemente dal fatto che i dati vengano scritti nella memoria o nell'archivio magnetico, Timestream for Live Analytics indicizza e partiziona automaticamente i dati prima di scriverli sullo storage. Una singola tabella Timestream for Live Analytics può avere centinaia, migliaia o addirittura milioni di partizioni. Le singole partizioni non comunicano direttamente tra loro e non condividono alcun dato (architettura shared-nothing). Invece, il partizionamento di una tabella viene tracciato tramite un servizio di tracciamento e indicizzazione delle partizioni ad alta disponibilità. Ciò fornisce un'altra separazione delle preoccupazioni progettata specificamente per ridurre al minimo l'effetto dei guasti nel sistema e rendere molto meno probabili i guasti correlati.

Architettura di storage

Quando i dati vengono archiviati in Timestream for Live Analytics, i dati vengono organizzati in ordine temporale e temporale in base agli attributi di contesto scritti con i dati. Avere uno schema di partizionamento che divida lo «spazio» oltre al tempo è importante per scalare in modo massiccio un sistema di serie temporali. Questo perché la maggior parte dei dati delle serie temporali viene scritta all'ora corrente o in prossimità di essa. Di conseguenza, il partizionamento basato solo sul tempo non consente di distribuire bene il traffico di scrittura o di eliminare efficacemente i dati al momento dell'interrogazione. Questo è importante per l'elaborazione di serie temporali su scala estrema e ha consentito a Timestream for Live Analytics di scalare ordini di grandezza superiori rispetto agli altri principali sistemi attualmente disponibili in modalità serverless. Le partizioni risultanti vengono chiamate «tessere» perché rappresentano divisioni di uno spazio bidimensionale (progettate per avere dimensioni simili). Le tabelle Timestream per Live Analytics iniziano come una singola partizione (riquadro), quindi si suddividono nella dimensione spaziale in base alla velocità di trasmissione. Quando i riquadri raggiungono una certa dimensione, vengono suddivisi nella dimensione temporale per ottenere un migliore parallelismo di lettura man mano che la dimensione dei dati aumenta.

Timestream for Live Analytics è progettato per gestire automaticamente il ciclo di vita dei dati delle serie temporali. Timestream for Live Analytics offre due archivi di dati: un archivio in memoria e un archivio magnetico conveniente. Supporta anche la configurazione di policy a livello di tabella per trasferire automaticamente i dati tra gli archivi. Le scritture dei dati ad alta velocità in entrata arrivano nell'archivio di memoria, dove i dati sono ottimizzati per le scritture, nonché le letture eseguite

all'ora corrente per alimentare la dashboard e inviare avvisi di tipo query. Una volta trascorso il periodo di tempo principale necessario per le scritture, gli avvisi e la creazione di dashboard, i dati possono fluire automaticamente dall'archivio di memoria all'archivio magnetico per ottimizzare i costi. Timestream for Live Analytics consente di impostare una politica di conservazione dei dati nell'archivio di memoria per questo scopo. Le scritture dei dati per i dati in arrivo tardivo vengono scritte direttamente nell'archivio magnetico.

Una volta che i dati sono disponibili nell'archivio magnetico (a causa della scadenza del periodo di conservazione dell'archivio di memoria o a causa delle scritture dirette nell'archivio magnetico), vengono riorganizzati in un formato altamente ottimizzato per letture di dati di grandi volumi. L'archivio magnetico ha anche una politica di conservazione dei dati che può essere configurata se esiste una soglia temporale in cui i dati superano la loro utilità. Quando i dati superano l'intervallo di tempo definito per la politica di conservazione degli archivi magnetici, vengono rimossi automaticamente. Pertanto, con Timestream for Live Analytics, a parte alcune configurazioni, la gestione del ciclo di vita dei dati avviene senza problemi dietro le quinte.

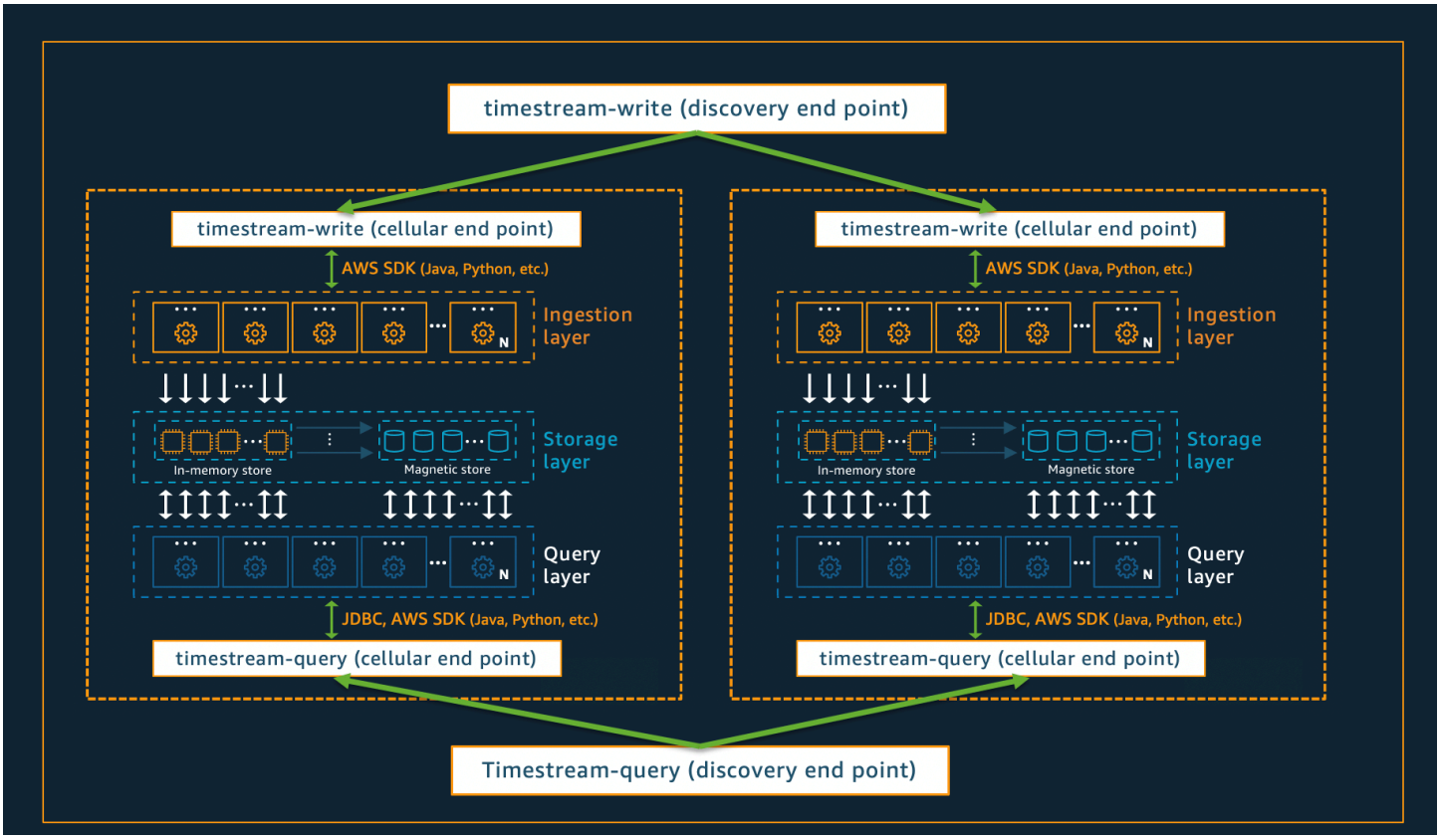
Architettura delle interrogazioni

Le query Timestream for Live Analytics sono espresse in una SQL grammatica con estensioni per il supporto specifico delle serie temporali (tipi di dati e funzioni specifici delle serie temporali), quindi la curva di apprendimento è facile per gli sviluppatori che già conoscono. SQL Le query vengono quindi elaborate da un motore di query adattivo e distribuito che utilizza i metadati del servizio di tracciamento e indicizzazione dei riquadri per accedere e combinare senza problemi i dati tra gli archivi di dati al momento dell'emissione della query. Ciò garantisce un'esperienza molto apprezzata dai clienti, in quanto riduce molte delle complessità di Rube Goldberg in un'astrazione di database semplice e familiare.

Le query vengono eseguite da una flotta di lavoratori dedicata, in cui il numero di lavoratori arruolati per eseguire una determinata query è determinato dalla complessità della query e dalla dimensione dei dati. Le prestazioni per query complesse su set di dati di grandi dimensioni sono ottenute attraverso un parallelismo massiccio, sia sulla flotta di esecuzione delle query che sui parchi di storage del sistema. La capacità di analizzare enormi quantità di dati in modo rapido ed efficiente è uno dei maggiori punti di forza di Timestream for Live Analytics. Una singola query che viene eseguita su terabyte o addirittura petabyte di dati potrebbe avere migliaia di macchine che lavorano su tutte contemporaneamente.

Architettura cellulare

Per garantire che Timestream for Live Analytics possa offrire una scalabilità praticamente infinita per le vostre applicazioni, garantendo contemporaneamente una disponibilità del 99,99%, il sistema è progettato anche utilizzando un'architettura cellulare. Invece di ridimensionare il sistema nel suo insieme, Timestream for Live Analytics si segmenta in più copie più piccole di se stesso, denominate celle. Ciò consente di testare le cellule su vasta scala e impedisce che un problema di sistema in una cella influisca sull'attività di qualsiasi altra cella in una determinata regione. Sebbene Timestream for Live Analytics sia progettato per supportare più celle per regione, considera il seguente scenario fittizio, in cui ci sono 2 celle in una regione.



Nello scenario illustrato sopra, le richieste di inserimento dei dati e di interrogazione vengono prima elaborate dall'endpoint di rilevamento rispettivamente per l'ingestione e l'interrogazione dei dati. Quindi, l'endpoint di rilevamento identifica la cella contenente i dati del cliente e indirizza la richiesta all'endpoint di acquisizione o interrogazione appropriato per quella cella. Quando si utilizzaSDKs, queste attività di gestione degli endpoint vengono gestite in modo trasparente per te.

> **Note**
>
> Quando utilizzi gli VPC endpoint con Timestream for Live Analytics o accedi direttamente alle REST API operazioni per Timestream for Live Analytics, dovrai interagire direttamente con gli endpoint cellulari. Per indicazioni su come eseguire questa operazione, consulta VPCEndpoints per istruzioni su come configurare gli endpoint e VPC Endpoint Discovery Pattern per istruzioni sull'invocazione diretta delle operazioni. REST API

Scrive

È possibile raccogliere dati di serie temporali da dispositivi connessi, sistemi IT e apparecchiature industriali e scriverli in Timestream for Live Analytics. Timestream for Live Analytics consente di scrivere punti dati da una singola serie temporale e/o punti dati da più serie in un'unica richiesta di scrittura quando le serie temporali appartengono alla stessa tabella. Per comodità, Timestream for Live Analytics offre uno schema flessibile che rileva automaticamente i nomi delle colonne e i tipi di dati per le tabelle Timestream for Live Analytics in base ai nomi delle dimensioni e ai tipi di dati dei valori di misura specificati quando richiami le scritture nel database. Puoi anche scrivere batch di dati in Timestream for Live Analytics.

> **Note**
>
> Timestream for Live Analytics supporta l'eventuale semantica di coerenza per le letture. Ciò significa che quando si interrogano i dati immediatamente dopo aver scritto un batch di dati in Timestream for Live Analytics, i risultati della query potrebbero non riflettere i risultati di un'operazione di scrittura completata di recente. I risultati possono includere anche alcuni dati obsoleti. Analogamente, durante la scrittura di dati di serie temporali con una o più nuove dimensioni, una query può restituire un sottoinsieme parziale di colonne per un breve periodo di tempo. Se ripeti queste richieste di query dopo un breve periodo, i risultati dovrebbero restituire i dati più recenti.

È possibile scrivere dati utilizzando AWS SDKsAWS CLI, o tramiteAWS Lambda,AWS IoT Core,Servizio gestito da Amazon per Apache Flink, Amazon KinesisAmazon MSK, eTelegraf open source.

Argomenti

Tipi di dati

Timestream for Live Analytics supporta i seguenti tipi di dati per le scritture.

Tipo di dati	Descrizione
BIGINT	Rappresenta un numero intero con segno a 64 bit.
BOOLEAN	Rappresenta i due valori di verità della logica, vale a dire vero e falso.
DOUBLE	Precisione variabile a 64 bit che implementa IEEE lo standard 754 per l'aritmetica binaria a virgola mobile. **Note** Esistono funzioni del linguaggio di interrogazione e valori doppi che possono essere utilizzati nelle query. `Infinity` `NaN` Ma non puoi scrivere quei valori su Timestream.
VARCHAR	Dati di caratteri a lunghezza variabile con una lunghezza massima opzionale. Il limite massimo è 2 KB.
MULTI	Tipo di dati per record multimisura. Questo tipo di dati include una o più misure di tipo`BIGINT`,`BOOLEAN`, `DOUBLE``VARCHAR`, e`TIMESTAMP` .
TIMESTAMP	Rappresenta un'istanza temporale che utilizza il time in con precision e in nanosecondiUTC, che tiene traccia del tempo trascorso dall'ora

Tipo di dati	Descrizione
	di Unix. Questo tipo di dati è attualmente supportato solo per i record multimisura (ovvero all'interno di valori di misura di tipo). `MULTI` *YYYY*-*MM*-*DD* *hh*:*mm*:*ss*.*sssssssss* Le scritture supportano i timestamp compresi nell'intervallo di. `1970-01-01 00:00:00.000000000` `2262-04-11 23:47:16.854775807`

Nessuna definizione iniziale dello schema

Prima di inviare dati in Amazon Timestream for Live Analytics, devi creare un database e una tabella utilizzando AWS Management Console le operazioni Timestream for Live Analytics o Timestream for Live SDKs Analytics. API Per ulteriori informazioni, consulta Creazione di un database e Creare una tabella. Durante la creazione della tabella, non è necessario definire lo schema in anticipo. Amazon Timestream for Live Analytics rileva automaticamente lo schema in base alle misure e alle dimensioni dei punti dati inviati, quindi non è più necessario modificare lo schema offline per adattarlo ai dati delle serie temporali in rapida evoluzione.

Scrittura di dati (inserti e sconvolgimenti)

L'operazione di scrittura in Amazon Timestream for Live Analytics consente di inserire e modificare dati. Per impostazione predefinita, le scritture in Amazon Timestream for Live Analytics seguono la semantica del primo scrittore vince, in cui i dati vengono archiviati solo come aggiunta e i record duplicati vengono rifiutati. Sebbene il primo scrittore vinca la semantica soddisfi i requisiti di molte applicazioni di serie temporali, ci sono scenari in cui le applicazioni devono aggiornare i record esistenti in modo idempotente e/o scrivere dati con l'ultimo scrittore vince la semantica, in cui il record con la versione più alta viene archiviato nel servizio. Per affrontare questi scenari, Amazon Timestream for Live Analytics offre la possibilità di modificare i dati. Upsert è un'operazione che inserisce un record nel sistema quando il record non esiste o aggiorna il record quando ne esiste uno. Quando il record viene aggiornato, viene aggiornato in modo idempotente.

Non esiste un'operazione a livello di record per l'eliminazione. Ma le tabelle e i database possono essere eliminati.

Scrittura di dati nell'archivio di memoria e nell'archivio magnetico

Amazon Timestream for Live Analytics offre la possibilità di scrivere direttamente i dati nell'archivio di memoria e nell'archivio magnetico. L'archivio di memoria è ottimizzato per le scritture di dati ad alto throughput e l'archivio magnetico è ottimizzato per scritture con throughput inferiore di dati in arrivo tardivo.

I dati in arrivo tardivo sono dati con un timestamp precedente all'ora corrente e al di fuori del periodo di conservazione dell'archivio di memoria. È necessario abilitare esplicitamente la capacità di scrivere dati in arrivo tardivo nell'archivio magnetico abilitando le scritture nell'archivio magnetico per la tabella. Inoltre, `MagneticStoreRejectedDataLocation` viene definito quando viene creata una tabella. Per scrivere nell'archivio magnetico, i chiamanti `WriteRecords` devono disporre delle `S3:PutObject` autorizzazioni per il bucket S3 specificate durante la creazione della tabella. `MagneticStoreRejectedDataLocation` Per ulteriori informazioni, vedere CreateTable, e. WriteRecordsPutObject

Scrittura di dati con record a misura singola e record a più misure

Amazon Timestream for Live Analytics offre la possibilità di scrivere dati utilizzando due tipi di record, vale a dire record a singola misura e record multimisura.

Record a misura singola

I record a misura singola consentono di inviare una singola misura per record. Quando i dati vengono inviati a Timestream for Live Analytics utilizzando questo formato, Timestream for Live Analytics crea una riga di tabella per record. Ciò significa che se un dispositivo emette 4 metriche e ciascuna metrica viene inviata come record a misura singola, Timestream for Live Analytics creerà 4 righe nella tabella per memorizzare questi dati e gli attributi del dispositivo verranno ripetuti per ogni riga. Questo formato è consigliato nei casi in cui si desidera monitorare una singola metrica da un'applicazione o quando l'applicazione non emette più metriche contemporaneamente.

Record multimisura

Con i record con più misure, è possibile memorizzare più misure in una singola riga della tabella, anziché archiviare una misura per riga della tabella. I record multimisura consentono quindi di migrare i dati esistenti dai database relazionali ad Amazon Timestream for Live Analytics con modifiche minime.

Puoi anche raggruppare più dati in una singola richiesta di scrittura rispetto ai record a misura singola. Ciò aumenta la velocità di scrittura e le prestazioni dei dati e riduce anche il costo delle scritture dei dati. Questo perché raggruppare più dati in una richiesta di scrittura consente ad Amazon

Timestream for Live Analytics di identificare più dati ripetibili in una singola richiesta di scrittura (ove applicabile) e di addebitare una sola volta per i dati ripetuti.

Argomenti

Record multimisura

Con i record multimisura, è possibile archiviare i dati delle serie temporali in un formato più compatto nella memoria e nell'archivio magnetico, il che aiuta a ridurre i costi di archiviazione dei dati. Inoltre, l'archiviazione compatta dei dati si presta alla scrittura di query più semplici per il recupero dei dati, migliora le prestazioni delle query e riduce il costo delle query.

Inoltre, i record multimisura supportano anche il tipo di TIMESTAMP dati per l'archiviazione di più di un timestamp in un record di serie temporali. TIMESTAMPgli attributi in un record multimisura supportano timestamp futuri o passati. I record multimisura aiutano quindi a migliorare le prestazioni, i costi e la semplicità delle query e offrono una maggiore flessibilità per l'archiviazione di diversi tipi di misure correlate.

Vantaggi

Di seguito sono riportati i vantaggi dell'utilizzo di record con più misure.

- Prestazioni e costi: i record multimisura consentono di scrivere più misure di serie temporali in un'unica richiesta di scrittura. Ciò aumenta la velocità di scrittura e riduce anche il costo delle scritture. Con i record multimisura, è possibile archiviare i dati in modo più compatto, il che aiuta a ridurre i costi di archiviazione dei dati. L'archiviazione compatta dei record multimisura comporta una riduzione dei dati elaborati dalle query. Questo è progettato per migliorare le prestazioni complessive delle query e contribuire a ridurre i costi delle query.
- Semplicità delle query: con i record multimisura, non è necessario scrivere espressioni di tabella comuni complesse (CTEs) in una query per leggere più misure con lo stesso timestamp. Questo perché le misure vengono memorizzate come colonne in un'unica riga della tabella. I record multimisura consentono quindi di scrivere query più semplici.
- Flessibilità di modellazione dei dati: puoi scrivere timestamp futuri in Timestream for Live Analytics utilizzando il tipo di TIMESTAMP dati e i record multimisura. Un record multimisura può avere più attributi di tipo di TIMESTAMP dati, oltre al campo orario in un record. TIMESTAMPgli attributi,

in un record multimisura, possono avere timestamp futuri o passati e comportarsi come il campo time, tranne per il fatto che Timestream for Live Analytics non indicizza i valori di tipo in un record multimisura. TIMESTAMP

Casi d'uso

È possibile utilizzare record multimisura per qualsiasi applicazione di serie temporali che generi più di una misurazione dallo stesso dispositivo in un dato momento. Di seguito sono riportati alcuni esempi di applicazioni.

- Una piattaforma di streaming video che genera centinaia di metriche contemporaneamente.
- Dispositivi medici che generano misurazioni come i livelli di ossigeno nel sangue, la frequenza cardiaca e il polso.
- Apparecchiature industriali come piattaforme petrolifere che generano sensori metrici, di temperatura e meteorologici.
- Altre applicazioni progettate con uno o più microservizi.

Esempio: monitoraggio delle prestazioni e dello stato di un'applicazione di streaming video

Prendiamo in considerazione un'applicazione di streaming video in esecuzione su 200 EC2 istanze. Vuoi utilizzare Amazon Timestream for Live Analytics per archiviare e analizzare le metriche emesse dall'applicazione, in modo da poter comprendere le prestazioni e lo stato della tua applicazione, identificare rapidamente anomalie, risolvere problemi e scoprire opportunità di ottimizzazione.

Modelleremo questo scenario con record a singola misura e record a più misure, quindi confronteremo o contrapporremo entrambi gli approcci. Per ogni approccio, facciamo le seguenti ipotesi.

- Ogni EC2 istanza emette quattro misure (video_startup_time, rebuffering_ratio, video_playback_failures e average_frame_rate) e quattro dimensioni (device_id, device_type, os_version e region) al secondo.
- Vuoi archiviare 6 ore di dati nell'archivio di memoria e 6 mesi di dati nell'archivio magnetico.
- Per identificare le anomalie, hai impostato 10 query che vengono eseguite ogni minuto per identificare eventuali attività insolite negli ultimi minuti. Hai anche creato una dashboard con otto widget che visualizzano i dati delle ultime 6 ore, in modo da poter monitorare efficacemente la tua applicazione. Questa dashboard è accessibile da cinque utenti alla volta e viene aggiornata automaticamente ogni ora.

Utilizzo di record a misura singola

Modellazione dei dati: con i record a singola misura, creeremo un record per ciascuna delle quattro misure (tempo di avvio del video, rapporto di rebuffering, errori di riproduzione video e frame rate medio). Ogni record avrà le quattro dimensioni (device_id, device_type, os_version e region) e un timestamp.

Scrive: quando scrivi dati in Amazon Timestream for Live Analytics, i record vengono costruiti come segue.

```
public void writeRecords() {
    System.out.println("Writing records");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();

    final Dimension device_id = new
 Dimension().withName("device_id").withValue("12345678");
    final Dimension device_type = new
 Dimension().withName("device_type").withValue("iPhone 11");
    final Dimension os_version = new
 Dimension().withName("os_version").withValue("14.8");
    final Dimension region = new Dimension().withName("region").withValue("us-east-1");

    dimensions.add(device_id);
    dimensions.add(device_type);
    dimensions.add(os_version);
    dimensions.add(region);

    Record videoStartupTime = new Record()
        .withDimensions(dimensions)
        .withMeasureName("video_startup_time")
        .withMeasureValue("200")
        .withMeasureValueType(MeasureValueType.BIGINT)
        .withTime(String.valueOf(time));
    Record rebufferingRatio = new Record()
        .withDimensions(dimensions)
        .withMeasureName("rebuffering_ratio")
        .withMeasureValue("0.5")
        .withMeasureValueType(MeasureValueType.DOUBLE)
        .withTime(String.valueOf(time));
```

```
    Record videoPlaybackFailures = new Record()
        .withDimensions(dimensions)
        .withMeasureName("video_playback_failures")
        .withMeasureValue("0")
        .withMeasureValueType(MeasureValueType.BIGINT)
        .withTime(String.valueOf(time));
    Record averageFrameRate = new Record()
        .withDimensions(dimensions)
        .withMeasureName("average_frame_rate")
        .withMeasureValue("0.5")
        .withMeasureValueType(MeasureValueType.DOUBLE)
        .withTime(String.valueOf(time));

    records.add(videoStartupTime);
    records.add(rebufferingRatio);
    records.add(videoPlaybackFailures);
    records.add(averageFrameRate);

    WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withRecords(records);

    try {
      WriteRecordsResult writeRecordsResult =
amazonTimestreamWrite.writeRecords(writeRecordsRequest);
      System.out.println("WriteRecords Status: " +
writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      System.out.println("RejectedRecords: " + e);
      for (RejectedRecord rejectedRecord : e.getRejectedRecords()) {
        System.out.println("Rejected Index " + rejectedRecord.getRecordIndex() + ": "
            + rejectedRecord.getReason());
      }
      System.out.println("Other records were written successfully. ");
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Quando memorizzi record a misura singola, i dati vengono rappresentati logicamente come segue.

Orario	device_id	tipo_dispositivo	os_version	Regione	measure_name	measure_value::bigint	measure_value::double
2021-09-07 21:48:44.00	12345678	iPhone 11	14,8	us-east-1	ora di avvio del video	200	
2021-09-07 21:48:44.00	12345678	iPhone 11	14,8	us-east-1	rebuffering_ratio		0,5
2021-09-07 21:48:44.00	12345678	iPhone 11	14,8	us-east-1	errori di riproduzione video	0	
2021-09-07 21:48:44.00	12345678	iPhone 11	14,8	us-east-1	frequenza_frame_rate medio		0,85
2021-09-07 21:53:44.00	12345678	iPhone 11	14,8	us-east-1	ora di avvio del video	500	
2021-09-07 21:53:44.00	12345678	iPhone 11	14,8	us-east-1	rebuffering_ratio		1.5
2021-09-07 21:53:44.00	12345678	iPhone 11	14,8	us-east-1	errori di riproduzione video	10	

Orario	device_id	tipo_dispositivo	os_version	Regione	measure_name	measure_value::bigint	measure_value::double
2021-09-07 21:53:44.00	12345678	iPhone 11	14,8	us-east-1	frequenza_frame_rate medio		0.2

Interrogazioni: puoi scrivere una query che recuperi tutti i punti dati con lo stesso timestamp ricevuti negli ultimi 15 minuti come segue.

```
with cte_video_startup_time as ( SELECT time, device_id, device_type, os_version,
 region, measure_value::bigint as video_startup_time FROM table where time >= ago(15m)
 and measure_name=”video_startup_time”),
cte_rebuffering_ratio as ( SELECT time, device_id, device_type, os_version, region,
 measure_value::double as rebuffering_ratio FROM table where time >= ago(15m) and
 measure_name=”rebuffering_ratio”),
cte_video_playback_failures as ( SELECT time, device_id, device_type, os_version,
 region, measure_value::bigint as video_playback_failures FROM table where time >=
 ago(15m) and measure_name=”video_playback_failures”),
cte_average_frame_rate as ( SELECT time, device_id, device_type, os_version, region,
 measure_value::double as average_frame_rate FROM table where time >= ago(15m) and
 measure_name=”average_frame_rate”)
SELECT a.time, a.device_id, a.os_version, a.region, a.video_startup_time,
 b.rebuffering_ratio, c.video_playback_failures, d.average_frame_rate FROM
 cte_video_startup_time a, cte_buffering_ratio b, cte_video_playback_failures c,
 cte_average_frame_rate d WHERE
a.time = b.time AND a.device_id = b.device_id AND a.os_version = b.os_version AND
 a.region=b.region AND
a.time = c.time AND a.device_id = c.device_id AND a.os_version = c.os_version AND
 a.region=c.region AND
a.time = d.time AND a.device_id = d.device_id AND a.os_version = d.os_version AND
 a.region=d.region
```

Costo del carico di lavoro: il costo di questo carico di lavoro è stimato in 373,23 dollari al mese con record a misura singola

Utilizzo di record a più misure

Modellazione dei dati: con i record multimisura, creeremo un record che contiene tutte e quattro le misure (tempo di avvio del video, rapporto di rebuffering, errori di riproduzione video e frame rate medio), tutte e quattro le dimensioni (device_id, device_type, os_version e region) e un timestamp.

Scrive: quando scrivi dati in Amazon Timestream for Live Analytics, i record vengono costruiti come segue.

```
public void writeRecords() {
    System.out.println("Writing records");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();

    final Dimension device_id = new
 Dimension().withName("device_id").withValue("12345678");
    final Dimension device_type = new
 Dimension().withName("device_type").withValue("iPhone 11");
    final Dimension os_version = new
 Dimension().withName("os_version").withValue("14.8");
    final Dimension region = new Dimension().withName("region").withValue("us-east-1");

    dimensions.add(device_id);
    dimensions.add(device_type);
    dimensions.add(os_version);
    dimensions.add(region);

    Record videoMetrics = new Record()
        .withDimensions(dimensions)
        .withMeasureName("video_metrics")
        .withTime(String.valueOf(time));
        .withMeasureValueType(MeasureValueType.MULTI)
        .withMeasureValues(
          new MeasureValue()
         .withName("video_startup_time")
         .withValue("0")
         .withValueType(MeasureValueType.BIGINT),
         new MeasureValue()
  .withName("rebuffering_ratio")
         .withValue("0.5")
```

```
          .withType(MeasureValueType.DOUBLE),
           new MeasureValue()
          .withName("video_playback_failures")
          .withValue("0")
          .withValueType(MeasureValueType.BIGINT),
    new MeasureValue()
           .withName("average_frame_rate")
          .withValue("0.5")
          .withValueType(MeasureValueType.DOUBLE))

     records.add(videoMetrics);

     WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
         .withDatabaseName(DATABASE_NAME)
         .withTableName(TABLE_NAME)
         .withRecords(records);

     try {
       WriteRecordsResult writeRecordsResult =
amazonTimestreamWrite.writeRecords(writeRecordsRequest);
       System.out.println("WriteRecords Status: " +
writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
     } catch (RejectedRecordsException e) {
       System.out.println("RejectedRecords: " + e);
       for (RejectedRecord rejectedRecord : e.getRejectedRecords()) {
         System.out.println("Rejected Index " + rejectedRecord.getRecordIndex() + ": "
             + rejectedRecord.getReason());
       }
       System.out.println("Other records were written successfully. ");
     } catch (Exception e) {
       System.out.println("Error: " + e);
     }
  }
```

Quando memorizzi record multimisura, i dati vengono rappresentati logicamente come segue.

Orario	device_id	tipo_dispositivo	os_version	Regione	measure_name	ora di avvio del video	rebuffering_ratio	fallimenti_riproduzione_video	frequenza_fotogrammi media
2021-09-7 21:48:440	1234567	iPhone 11	14,8	us-east-1	metriche video	200	0,5	0	0,85
2021-09-7 21:53:440	1234567	iPhone 11	14,8	us-east-1	metriche video	500	1.5	10	0.2

Interrogazioni: puoi scrivere una query che recuperi tutti i punti dati con lo stesso timestamp ricevuti negli ultimi 15 minuti come segue.

```
SELECT time, device_id, device_type, os_version, region, video_startup_time,
 rebuffering_ratio, video_playback_failures, average_frame_rate FROM table where time
 >= ago(15m)
```

Costo del carico di lavoro: il costo del carico di lavoro è stimato in 127,43 dollari con record multimisura.

> **Note**
>
> In questo caso, l'utilizzo di record multimisura riduce la spesa mensile complessiva stimata di 2,5 volte, con i costi di scrittura dei dati ridotti di 3,3 volte, i costi di storage ridotti di 3,3 volte e il costo delle query ridotto di 1,2 volte.

Scrittura di dati con un timestamp esistente nel passato o nel futuro

Timestream for Live Analytics offre la possibilità di scrivere dati con un timestamp che si trova al di fuori della finestra di conservazione dell'archivio di memoria attraverso un paio di meccanismi diversi.

- Scritture nell'archivio magnetico: è possibile scrivere i dati in arrivo tardivo direttamente nell'archivio magnetico tramite le scritture nell'archivio magnetico. Per utilizzare le scritture sull'archivio magnetico, è necessario prima abilitare le scritture sull'archivio magnetico per una tabella. È quindi possibile inserire i dati nella tabella utilizzando lo stesso meccanismo utilizzato per scrivere i dati nell'archivio di memoria. Amazon Timestream for Live Analytics scriverà automaticamente i dati nell'archivio magnetico in base al relativo timestamp.

  > **Note**
  >
  > La write-to-read latenza per l'archivio magnetico può arrivare fino a 6 ore, a differenza della scrittura dei dati nell'archivio di memoria, dove la write-to-read latenza è inferiore al secondo.

- TIMESTAMPtipo di dati per misure: puoi utilizzare il tipo di TIMESTAMP dati per archiviare dati passati, presenti o futuri. Un record multimisura può avere più attributi di tipo di TIMESTAMP dati, oltre al campo temporale di un record. TIMESTAMPgli attributi, in un record multimisura, possono avere timestamp futuri o passati e comportarsi come il campo time, tranne per il fatto che Timestream for Live Analytics non indicizza i valori di tipo in un record multimisura. TIMESTAMP

  > **Note**
  >
  > Il tipo di TIMESTAMP dati è supportato solo per i record multimisura.

Eventuale coerenza per le letture

Timestream for Live Analytics supporta l'eventuale semantica di coerenza per le letture. Ciò significa che quando si interrogano i dati immediatamente dopo aver scritto un batch di dati in Timestream for Live Analytics, i risultati della query potrebbero non riflettere i risultati di un'operazione di scrittura completata di recente. Se ripeti queste richieste di query dopo poco tempo, i risultati dovrebbero restituire i dati più recenti.

Il batching scrive con WriteRecords API

Amazon Timestream for Live Analytics ti consente di scrivere punti dati da una singola serie temporale e/o punti dati da più serie in un'unica richiesta di scrittura. Il raggruppamento di più punti dati in un'unica operazione di scrittura è vantaggioso dal punto di vista delle prestazioni e dei costi. Per maggiori Scrive dettagli, consulta la sezione Misurazione e prezzi.

> **Note**
>
> Le tue richieste di scrittura a Timestream for Live Analytics potrebbero essere limitate man mano che Timestream for Live Analytics si adatta alle esigenze di acquisizione dei dati della tua applicazione. Se le applicazioni presentano eccezioni di limitazione, è necessario continuare a inviare dati con la stessa velocità di trasmissione (o superiore) per consentire a Timestream for Live Analytics di adattarsi automaticamente alle esigenze dell'applicazione.

Caricamento in batch

Con il caricamento in batch per Amazon Timestream LiveAnalytics for, puoi importare file archiviati in Amazon S3 in CSV Timestream in batch. Con questa nuova funzionalità, puoi conservare i tuoi dati in Timestream LiveAnalytics senza dover fare affidamento su altri strumenti o scrivere codice personalizzato. Puoi utilizzare il caricamento in batch per completare i dati con tempi di attesa flessibili, ad esempio dati che non sono immediatamente necessari per l'interrogazione o l'analisi.

È possibile creare attività di caricamento in batch utilizzando AWS Management Console AWS CLI, the e. AWS SDKs Per ulteriori informazioni, consulta Utilizzo del caricamento in batch con la console, Utilizzo del caricamento in batch con AWS CLI e Utilizzo del caricamento in batch con AWS SDKs.

Per ulteriori informazioni sul caricamento in batch, vedereUtilizzo del caricamento in batch in Timestream per LiveAnalytics.

Scelta tra WriteRecords API operazione e caricamento in batch

Con l' WriteRecords APIoperazione, puoi scrivere i dati delle serie temporali di streaming in Timestream LiveAnalytics così come vengono generati dal tuo sistema. Utilizzando WriteRecords, puoi importare continuamente un singolo punto dati o lotti di dati più piccoli in tempo reale. Timestream for ti LiveAnalytics offre uno schema flessibile che rileva automaticamente i nomi delle colonne e i tipi di dati per il tuo Timestream per le LiveAnalytics tabelle, in base ai nomi delle dimensioni e ai tipi di dati dei punti dati specificati quando richiami le scritture nel database.

Al contrario, il caricamento in batch consente l'inserimento affidabile di dati di serie temporali in batch dai file di origine (CSVfile) in Timestream for, utilizzando un modello di dati definito dall'utente. LiveAnalytics Alcuni esempi di quando utilizzare il caricamento in batch con un file sorgente sono l'importazione in blocco di dati di serie temporali per la valutazione di Timestream LiveAnalytics tramite un proof of concept, l'importazione in blocco di dati di serie temporali da un dispositivo IoT che è rimasto offline per qualche tempo e la migrazione di dati storici di serie temporali da Amazon S3 a

Timestream for. LiveAnalytics Per informazioni sul caricamento in Utilizzo del caricamento in batch in Timestream per LiveAnalytics batch, consulta.

Entrambe le soluzioni sono sicure, affidabili e performanti.

Da utilizzare WriteRecords quando:

- Streaming di quantità inferiori (meno di 10 MB) di dati per richiesta.
- Inserimento di tabelle esistenti.
- Acquisizione di dati da un flusso di log.
- Esecuzione di analisi in tempo reale.
- Richiede una latenza inferiore.

Utilizza il caricamento in batch quando:

- Inserimento in file di grandi quantità di dati provenienti da Amazon CSV S3. Per ulteriori informazioni sui limiti, consulta Quote.
- Inserimento di nuove tabelle, ad esempio nel caso di una migrazione di dati.
- Arricchimento dei database con dati storici (inserimento in nuove tabelle).
- Hai dati di origine che cambiano lentamente o non cambiano affatto.
- I tempi di attesa sono flessibili perché un'attività di caricamento in batch potrebbe rimanere in sospeso fino a quando le risorse non saranno disponibili, soprattutto se si carica una grande quantità di dati. Il caricamento in batch è adatto per dati che non devono essere immediatamente disponibili per l'interrogazione o l'analisi per aggiungere maggiore chiarezza.

Storage

Timestream for Live Analytics archivia e organizza i dati delle serie temporali per ottimizzare i tempi di elaborazione delle query e ridurre i costi di archiviazione. Offre lo storage dei dati su più livelli e supporta due livelli di archiviazione: un archivio di memoria e un archivio magnetico. L'archivio di memoria è ottimizzato per scritture di dati ad alta velocità e query veloci. point-in-time L'archivio magnetico è ottimizzato per velocità di trasmissione inferiori, scritture di dati in arrivo tardivo, archiviazione di dati a lungo termine e query analitiche veloci.

Timestream for Live Analytics garantisce la durabilità dei dati replicando automaticamente i dati di memoria e archiviazione magnetica in diverse zone di disponibilità all'interno di un'unica soluzione.

Regione AWS Tutti i dati vengono scritti su disco prima di confermare che la richiesta di scrittura è completa.

Timestream for Live Analytics consente di configurare le politiche di conservazione per spostare i dati dall'archivio di memoria all'archivio magnetico. Quando i dati raggiungono il valore configurato, Timestream for Live Analytics sposta automaticamente i dati nell'archivio magnetico. È inoltre possibile impostare un valore di conservazione sull'archivio magnetico. Quando i dati scadono dall'archivio magnetico, vengono eliminati definitivamente.

Ad esempio, si consideri uno scenario in cui si configura l'archivio di memoria per contenere dati per una settimana e l'archivio magnetico per contenere dati per 1 anno. L'età dei dati viene calcolata utilizzando il timestamp associato al punto dati. Quando i dati nell'archivio di memoria risalgono a una settimana, vengono automaticamente spostati nell'archivio magnetico. Vengono quindi mantenuti nell'archivio magnetico per un anno. Quando i dati hanno un anno, vengono eliminati da Timestream for Live Analytics. I valori di conservazione dell'archivio di memoria e dell'archivio magnetico definiscono cumulativamente la quantità di tempo in cui i dati verranno archiviati in Timestream for Live Analytics. Ciò significa che per lo scenario precedente, dal momento dell'arrivo dei dati, i dati vengono archiviati in Timestream for Live Analytics per un periodo totale di 1 anno e 1 settimana.

> **Note**
>
> Quando si aggiorna il periodo di conservazione della memoria o dell'archivio magnetico, la modifica alla conservazione ha effetto da quel momento in poi. Ad esempio, se il periodo di conservazione dell'archivio di memoria è stato impostato su 2 ore e poi modificato su 24 ore aggiornando le politiche di conservazione della tabella, l'archivio di memoria sarà in grado di contenere 24 ore di dati, ma verrà popolato con 24 ore di dati 22 ore dopo la modifica. Timestream for Live Analytics non recupera i dati dall'archivio magnetico per popolare l'archivio di memoria.

Per garantire la sicurezza dei dati delle serie temporali, i dati in Timestream for Live Analytics sono sempre crittografati per impostazione predefinita. Questo vale per i dati in transito e a riposo. Inoltre, Timestream for Live Analytics consente di utilizzare chiavi gestite dai clienti per proteggere i dati nell'archivio magnetico. Per ulteriori informazioni sulle chiavi gestite dai clienti, consulta. AWS KMS keys

Query

Con Timestream for Live Analytics, puoi archiviare e analizzare facilmente metriche DevOps, dati dei sensori per applicazioni IoT e dati di telemetria industriale per la manutenzione delle apparecchiature, oltre a molti altri casi d'uso. Il motore di query adattivo appositamente progettato in Timestream for Live Analytics consente di accedere ai dati su più livelli di archiviazione utilizzando una singola istruzione. SQL Accede e combina in modo trasparente i dati su più livelli di storage senza che sia necessario specificare la posizione dei dati. Puoi utilizzarlo SQL per interrogare i dati in Timestream for Live Analytics per recuperare i dati delle serie temporali da una o più tabelle. È possibile accedere alle informazioni sui metadati per database e tabelle. Timestream for Live Analytics supporta SQL anche funzioni integrate per l'analisi delle serie temporali. Puoi fare riferimento al Riferimento al linguaggio di interrogazione riferimento per ulteriori dettagli.

Timestream for Live Analytics è progettato per avere un'architettura di acquisizione, archiviazione e interrogazione dei dati completamente disaccoppiata, in cui ogni componente può scalare indipendentemente dagli altri componenti (consentendogli di offrire una scalabilità praticamente infinita per le esigenze di un'applicazione). Ciò significa che Timestream for Live Analytics non si «ribalta» quando le applicazioni inviano centinaia di terabyte di dati al giorno o eseguono milioni di query che elaborano piccole o grandi quantità di dati. Man mano che i dati crescono nel tempo, la latenza delle query in Timestream for Live Analytics rimane per lo più invariata. Questo perché l'architettura di query di Timestream for Live Analytics può sfruttare enormi quantità di parallelismo per elaborare volumi di dati più grandi e scalare automaticamente per soddisfare le esigenze di throughput delle query di un'applicazione.

Modello di dati

Timestream supporta due modelli di dati per le query: il modello flat e il modello di serie temporali.

> **Note**
>
> I dati in Timestream vengono archiviati utilizzando il modello flat ed è il modello predefinito per l'interrogazione dei dati. Il modello delle serie temporali è un concetto di query-time e viene utilizzato per l'analisi delle serie temporali.

- Modello piatto
- Modello di serie temporali

Modello piatto

Il modello flat è il modello di dati predefinito di Timestream per le query. Rappresenta i dati delle serie temporali in formato tabulare. I nomi delle dimensioni, l'ora, i nomi delle misure e i valori delle misure vengono visualizzati come colonne. Ogni riga della tabella è un punto dati atomico corrispondente a una misurazione in un momento specifico all'interno di una serie temporale. I database, le tabelle e le colonne di Timestream hanno alcuni vincoli di denominazione. Questi sono descritti in. Limiti del servizio

La tabella seguente mostra un esempio illustrativo di come Timestream memorizza i dati che rappresentano l'CPUutilizzo, l'utilizzo della memoria e l'attività di rete delle EC2 istanze, quando i dati vengono inviati come record a misura singola. In questo caso, le dimensioni sono la regione, la zona di disponibilità, il cloud privato virtuale e l'istanza delle istanze. IDs EC2 Le misure sono l'CPUutilizzo, l'utilizzo della memoria e i dati di rete in entrata per le istanze. EC2 Le colonne region, az, vpc e instance_id contengono i valori delle dimensioni. La colonna time contiene il timestamp di ogni record. La colonna measure_name contiene i nomi delle misure rappresentate da cpu-utilization, memory_utilization e network_bytes_in. Le colonne measure_value: :double contengono le misurazioni emesse come doppie (ad esempio utilizzo e utilizzo della memoria). CPU La colonna measure_value: :bigint contiene le misurazioni emesse come numeri interi, ad esempio i dati di rete in entrata.

Orario	Regione	az	vpc	instance_id	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:00.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef0	utilizzo della cpu_	35,0	null
2019-12-04 19:00:01.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef0	utilizzo della cpu_	38.2	null

Orario	Regione	az	vpc	instance_id	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:02,000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef0	utilizzo della cpu_	45,3	null
2019-12-04 19:00:00.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef0	memory_utilization	54,9	null
2019-12-04 19:00:01.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef0	memory_utilization	42,6	null
2019-12-04 19:00:02,000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef0	memory_utilization	33.3	null
2019-12-04 19:00:00.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef0	network_byte	34.400	null

Orario	Regione	az	vpc	instance_id	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:01.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef0	network_byte	1.500	null
2019-12-04 19:00:02.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef0	network_byte	6.000	null

La tabella seguente mostra un esempio illustrativo di come Timestream memorizza i dati che rappresentano l'CPUutilizzo, l'utilizzo della memoria e l'attività di rete delle EC2 istanze, quando i dati vengono inviati come record multimisura.

Orario	Regione	az	vpc	instance_id	measure_ame	cpu_utilization	memory_ilization	network_bytes
2019-12-04 19:00:00.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcde0	metriche	35,0	54,9	34.400
2019-12-04 19:00:01.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcde0	metriche	38,2	42,6	1.500

Orario	Regione	az	vpc	instance_id	measure_name	cpu_utilization	memory_utilization	network_bytes
2019-12-04 19:00:02.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcde0	metriche	45,3	33,3	6.600

Modello di serie temporali

Il modello delle serie temporali è un costrutto temporale di interrogazione utilizzato per l'analisi delle serie temporali. Rappresenta i dati come una sequenza ordinata di coppie (tempo, valore di misura). Timestream supporta funzioni di serie temporali come l'interpolazione per consentire di colmare le lacune nei dati. Per utilizzare queste funzioni, è necessario convertire i dati nel modello delle serie temporali utilizzando funzioni come create_time_series. Per ulteriori dettagli, fare riferimento a. Riferimento al linguaggio di interrogazione

Utilizzando l'esempio precedente dell'EC2istanza, ecco i dati di CPU utilizzo espressi come serie temporali.

Regione	az	vpc	instance_id	cpu_utilization
us-east-1	us-east-1d	vpc-1a2b3c4d	i-1234567890abcdef0	[{ora: 2019-12-04 19:00:00.000 000000, valore: 35}, {ora: 2019-12-04 19:00:01.000 000000, valore: 38.2}, {ora: 2019-12-04 19:00:02.000 000000, valore: 45.3}]

Interrogazioni pianificate

La funzionalità di interrogazione pianificata di Amazon Timestream for Live Analytics è una soluzione completamente gestita, serverless e scalabile per il calcolo e l'archiviazione di aggregati, rollup e altre forme di dati preelaborati tipicamente utilizzati per alimentare dashboard operativi, report aziendali, analisi ad hoc e altre applicazioni. Le interrogazioni pianificate rendono l'analisi in tempo reale più performante ed economica, in modo da poter ricavare ulteriori informazioni dai dati e continuare a prendere decisioni aziendali migliori.

Per ulteriori informazioni sulle interrogazioni pianificate, vedere. Utilizzo delle interrogazioni pianificate in Timestream per LiveAnalytics

Unità di calcolo Timestream () TCU

Amazon Timestream for Live Analytics misura la capacità di calcolo allocata per le tue esigenze di query nell'unità di calcolo Timestream (). TCU Una TCU comprende 4 e 16 GB di memoria. vCPUs Quando esegui query in Timestream for Live Analytics, il servizio esegue l'allocazione TCUs su richiesta in base alla complessità delle query e alla quantità di dati da elaborare. Il numero di dati consumati da una query TCUs determina il costo associato.

> **Note**
>
> Tutto ciò Account AWS che sarà integrato nel servizio dopo il 29 aprile 2024 verrà utilizzato TCUs per impostazione predefinita per le query di determinazione dei prezzi.

In questo argomento

MaxQuery TCU

Questa impostazione specifica il numero massimo di unità di elaborazione che il servizio utilizzerà in qualsiasi momento per rispondere alle tue richieste. Per eseguire le query, è necessario impostare la capacità minima su 4. TCUs È possibile impostare il numero massimo di TCUs in multipli di 4, ad esempio 4, 8, 16, 32 e così via. Ti vengono addebitate solo le risorse di elaborazione utilizzate per il carico di lavoro. Ad esempio, se imposti il massimo TCUs su 128, ma ne usi costantemente solo 8. TCUs Ti verrà addebitato solo per la durata durante la quale hai utilizzato l'8TCUs. L'impostazione predefinita `MaxQueryTCU` nel tuo account è 200. È possibile regolare `MaxQueryTCU` da 4 a 1000, utilizzando l'UpdateAccountSettingsAPIoperazione AWS Management Console o con AWS SDK o AWS CLI.

Ti consigliamo di impostarlo `MaxQueryTCU` per il tuo account. L'impostazione di un TCU limite massimo consente di controllare i costi limitando il numero di unità di calcolo che il servizio può utilizzare per il carico di lavoro delle query. Ciò consente di prevedere e gestire meglio la spesa per le query.

Fatturazione per TCU

Ciascuno TCU viene fatturato su base oraria con granularità al secondo e per un minimo di 30 secondi. L'unità di utilizzo di queste unità di elaborazione è di -ora. TCU

Quando esegui le query, ti viene fatturato l'importo TCUs utilizzato durante il tempo di esecuzione della query, misurato in ore. TCU Per esempio:

- Il tuo carico di lavoro ne utilizza 20 TCUs per 3 ore. Ti verranno fatturate 60 TCU ore (20 TCUs x 3 ore).
- Il tuo carico di lavoro ne utilizza 10 TCUs per 30 minuti e poi 20 TCUs per i successivi 30 minuti. Ti verranno fatturate 15 TCU ore (10 TCUs x 0,5 ore + 20 TCUs x 0,5 ore).

Il prezzo per TCU ora varia a seconda. Regione AWS Per ulteriori dettagli, consulta i prezzi di Amazon Timestream. Man mano che il carico di lavoro aumenta, il servizio ridimensiona automaticamente la capacità di calcolo fino al TCU limite massimo specificato (`MaxQueryTCU`) per mantenere prestazioni costanti. L'`MaxQueryTCU`impostazione funge da limite per la capacità di elaborazione fino alla quale il servizio può scalare. Questa impostazione consente di controllare il numero di risorse di elaborazione e, di conseguenza, il relativo costo.

Configurazione TCU

Quando effettui l'onboarding del servizio, ognuno Account AWS ha un `MaxQueryTCU` limite predefinito di 200. È possibile aggiornare questo limite come richiesto in qualsiasi momento utilizzando l'UpdateAccountSettingsAPIoperazione AWS Management Console o con AWS SDK o AWS CLI.

Se non sei sicuro dei valori da configurare, monitora la `QueryTCU` metrica relativa al tuo account. Questa metrica è disponibile in Amazon AWS Management Console e in Amazon CloudWatch. Questa metrica fornisce informazioni sulla granularità del numero massimo di TCUs prodotti utilizzati al minuto. In base ai dati storici e alla tua stima delle crescite future, impostala in `MaxQueryTCU` modo da adattarsi ai picchi di utilizzo. Ti consigliamo di avere un margine di crescita di almeno 4-16 TCUs rispetto al picco di utilizzo. Ad esempio, se il tuo picco `QueryTCU` negli ultimi 30 giorni è stato di 128, ti consigliamo di impostare un valore `MaxQueryTCU` compreso tra 132 e 144.

Stima delle unità di calcolo richieste

Le unità di calcolo possono elaborare le interrogazioni contemporaneamente. Per determinare il numero di unità di calcolo richieste, considera le linee guida generali riportate nella tabella seguente:

Query simultanee	TCUs
7	4
14	8
21	12

> **Note**
>
> - Queste sono linee guida generali e il numero effettivo di unità di calcolo richieste dipende da diversi fattori, come:
> - L'effettiva concomitanza delle interrogazioni.
> - Schemi di interrogazione.
> - Il numero di partizioni scansionate.
> - Altre caratteristiche specifiche del carico di lavoro.

- Questa linea guida si riferisce alle query che analizzano i dati dagli ultimi minuti a un'ora e rispettano le migliori pratiche di interrogazione Timestream e le linee guida per la modellazione dei dati.
- Monitora le prestazioni dell'applicazione e la `QueryTCU` metrica per regolare le unità di calcolo, se necessario.

Quando aumentare MaxQuery TCU

È consigliabile prendere in considerazione l'aumento `MaxQueryTCU` nei seguenti scenari:

- Il consumo massimo di query si avvicina o raggiunge il numero massimo di query TCU attualmente configurato. Ti consigliamo di impostare il numero massimo di query TCU almeno 4-16 volte TCUs superiore al picco di consumo.
- Le tue query restituiscono un errore 4xx con il messaggio superato. MaxQuery TCU Se prevedi un aumento pianificato del carico di lavoro, rivedi e modifica di conseguenza il numero massimo di interrogazioni configurato. TCU

Quando diminuire MaxQuery TCU

È consigliabile considerare la possibilità di ridurla `MaxQueryTCU` nei seguenti scenari:

- Il tuo carico di lavoro ha un modello di utilizzo prevedibile e stabile e hai una buona conoscenza dei tuoi requisiti di utilizzo del calcolo. Ridurre il numero massimo di interrogazioni tra 4 e 16 TCU volte al di TCU sopra del picco di consumo può aiutare a prevenire utilizzi e costi involontari. È possibile modificare il valore utilizzando l'operazione. UpdateAccountSettingsAPI
- L'utilizzo massimo del carico di lavoro è diminuito nel tempo, a causa di cambiamenti nell'applicazione o nei modelli di comportamento degli utenti. Ridurre il valore massimo TCU può aiutare a mitigare i costi involontari.

> **Note**
>
> A seconda dell'utilizzo corrente, la riduzione della modifica del TCU limite massimo potrebbe richiedere fino a 24 ore per essere efficace. Ti viene addebitato solo l'importo effettivamente TCUs utilizzato dalle tue query. Un TCU limite massimo di query più elevato non incide sui costi, a meno che non TCUs vengano utilizzati dal carico di lavoro.

Monitoraggio dell'utilizzo tramite metriche CloudWatch

Per monitorare TCU l'utilizzo, Timestream for Live Analytics fornisce la seguente CloudWatch metrica:. `QueryTCU` Questa metrica specifica il numero di unità di calcolo utilizzate in un minuto e viene emessa ogni minuto. Puoi scegliere di monitorare il massimo e il minimo TCUs utilizzati in un minuto. Puoi anche impostare allarmi su questa metrica per tenere traccia dei costi delle query in tempo reale.

Comprensione delle variazioni nell'utilizzo delle unità di calcolo

Il numero di risorse di calcolo necessarie per le query può aumentare o diminuire in base a diversi parametri. Ad esempio, il volume dei dati, i modelli di inserimento dei dati, la latenza delle query, la forma delle query, l'efficienza delle query e le combinazioni di query che utilizzano query analitiche e in tempo reale. Questi parametri possono comportare un aumento o una riduzione delle TCU unità necessarie per il carico di lavoro. In uno stato stazionario in cui questi parametri non cambiano, potresti osservare una diminuzione del numero di unità di calcolo necessarie per il carico di lavoro. Di conseguenza, ciò può ridurre il costo mensile.

Inoltre, se uno qualsiasi di questi parametri del carico di lavoro o dei dati cambia, il numero di unità di elaborazione richieste potrebbe aumentare. Quando Timestream riceve una query, a seconda delle partizioni di dati a cui accede la query, Timestream decide il numero di risorse di elaborazione per rispondere in modo efficiente alla query.

A intervalli periodici, in base ai modelli di inserimento e accesso alle query, Timestream ottimizza il layout dei dati. Timestream esegue l'ottimizzazione raggruppando le partizioni meno accessibili in un'unica partizione o suddividendo una partizione calda in più partizioni per migliorare le prestazioni. Di conseguenza, la capacità di calcolo utilizzata dalla stessa query potrebbe variare leggermente in momenti diversi.

> **Scegli di utilizzare i TCU prezzi per le tue domande**
>
> In qualità di utente esistente, puoi effettuare un opt-in una tantum da utilizzare TCUs per una migliore gestione dei costi e la rimozione del numero minimo di byte misurati per query. È possibile effettuare l'opt-in utilizzando l'operazione or con AWS Management Console o UpdateAccountSettingsAPI. AWS SDK AWS CLI Nell'APIoperazione, imposta il `QueryPricingModel` parametro su. `COMPUTE_UNITS`
> L'adozione del modello di determinazione dei prezzi basato sul calcolo è un cambiamento irreversibile.

Accesso a Timestream per LiveAnalytics

Puoi accedere a Timestream per LiveAnalytics utilizzare la console, o il. CLI API Per informazioni sull'accesso a Timestream per LiveAnalytics, consulta quanto segue:

Argomenti

Registrati per un Account AWS

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

Per iscriverti a un Account AWS

1. Apri la https://portal.aws.amazon.com/billing/registrazione.
2. Segui le istruzioni online.

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

   Quando ti iscrivi a un Account AWS, Utente root dell'account AWSviene creato un. L'utente root dispone dell'accesso a tutte le risorse e tutti i Servizi AWS nell'account. Come best practice di sicurezza, assegna l'accesso amministrativo a un utente e utilizza solo l'utente root per eseguire attività che richiedono l'accesso di un utente root.

AWS ti invia un'email di conferma dopo il completamento della procedura di registrazione. In qualsiasi momento, puoi visualizzare l'attività corrente del tuo account e gestirlo accedendo a https://aws.amazon.com/e scegliendo Il mio account.

Crea un utente con accesso amministrativo

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

Proteggi i tuoi Utente root dell'account AWS

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

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

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

Crea un utente con accesso amministrativo

1. Abilita IAM Identity Center.

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

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

Accesso come utente amministratore

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

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

Assegna l'accesso a ulteriori utenti

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

Segui le istruzioni riportate nella pagina Creazione di un set di autorizzazioni nella Guida per l'utente di AWS IAM Identity Center .

2. Assegna al gruppo prima gli utenti e poi l'accesso con autenticazione unica (Single Sign-On).

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

Fornisci Timestream per l'accesso LiveAnalytics

Le autorizzazioni necessarie per accedere a Timestream LiveAnalytics sono già concesse all'amministratore. Agli altri utenti, è necessario concedere loro l' LiveAnalytics accesso a Timestream utilizzando la seguente politica:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "timestream:*",
        "kms:DescribeKey",
        "kms:CreateGrant",
        "kms:Decrypt",
        "dbqms:CreateFavoriteQuery",
        "dbqms:DescribeFavoriteQueries",
        "dbqms:UpdateFavoriteQuery",
        "dbqms:DeleteFavoriteQueries",
        "dbqms:GetQueryString",
        "dbqms:CreateQueryHistory",
        "dbqms:UpdateQueryHistory",
        "dbqms:DeleteQueryHistory",
        "dbqms:DescribeQueryHistory",
        "s3:ListAllMyBuckets"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Per informazioni sudbqms, consulta Azioni, risorse e chiavi di condizione per il servizio Database Query Metadata. Per informazioni su, kms consulta Azioni, risorse e chiavi di condizione per il servizio di gestione delle AWS chiavi.

Concessione dell'accesso programmatico

Gli utenti necessitano dell'accesso programmatico se desiderano interagire con l' AWS AWS Management Console esterno di. Il modo per concedere l'accesso programmatico dipende dal tipo di utente che accede. AWS

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

Quale utente necessita dell'accesso programmatico?	Per	Come
Identità della forza lavoro (Utenti gestiti in IAM Identity Center)	Utilizza credenziali temporane e per firmare le richieste programmatiche a AWS CLI, AWS SDKs, o. AWS APIs	Segui le istruzioni per l'interfa ccia che desideri utilizzare. • Per la AWS CLI, vedere Configurazione dell'uso AWS IAM Identity Center nella AWS CLI Guida per l'utente.AWS Command Line Interface • Per AWS SDKs gli strumenti e AWS APIs, consulta l'autenticazione di IAM Identity Center nella Guida di riferimento agli strumenti AWS SDKs e agli strumenti.
IAM	Utilizza credenziali temporane e per firmare le richieste programmatiche a AWS CLI, AWS SDKs, o. AWS APIs	Seguendo le istruzioni riportate in Utilizzo delle credenziali temporanee con le

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

Utilizzo della console

Puoi utilizzare la Console di AWS gestione per Timestream Live Analytics per creare, modificare, eliminare, descrivere ed elencare database e tabelle. È inoltre possibile utilizzare la console per eseguire interrogazioni.

Argomenti

- Tutorial
- Creazione di un database
- Creare una tabella
- Esecuzione di una query

- Crea una query pianificata
- Eliminare un'interrogazione pianificata
- Eliminazione di una tabella
- Eliminazione di un database
- Modifica una tabella
- Modifica un database

Tutorial

Questo tutorial mostra come creare un database popolato con set di dati di esempio ed eseguire query di esempio. I set di dati di esempio utilizzati in questo tutorial sono spesso presenti nell'IoT e negli DevOps scenari. Il set di dati IoT contiene dati di serie temporali come la velocità, la posizione e il carico di un camion, per semplificare la gestione della flotta e identificare opportunità di ottimizzazione. Il DevOps set di dati contiene metriche di EC2 istanza come CPU l'utilizzo della rete e della memoria per migliorare le prestazioni e la disponibilità delle applicazioni. Ecco un video tutorial per le istruzioni descritte in questa sezione

Segui questi passaggi per creare un database popolato con i set di dati di esempio ed eseguire query di esempio utilizzando la AWS Console.

1. Apri la console.AWS
2. Nel riquadro di navigazione, scegli Database
3. Fai clic su Crea database.
4. Nella pagina di creazione del database, inserisci quanto segue:
   - Scegli la configurazione: seleziona il database di esempio.
   - Nome: inserisci un nome di database a tua scelta.
   - Scegli set di dati di esempio: seleziona IoT e. DevOps
   - Fai clic su Crea database per creare un database contenente due tabelle, IoT e DevOps popolato con dati di esempio.
5. Nel riquadro di navigazione, scegli Query editor
6. Seleziona Query di esempio dal menu in alto.
7. Fai clic su una delle interrogazioni di esempio. Si tornerà all'editor di query con l'editor popolato con la query di esempio.

8. Fate clic su Esegui per eseguire la query e visualizzare i risultati della query.

Creazione di un database

Segui questi passaggi per creare un database utilizzando la AWS console.

1. Apri la AWS console.
2. Nel riquadro di navigazione, scegli Database
3. Fai clic su Crea database.
4. Nella pagina di creazione del database, inserisci quanto segue.
   - Scegli configurazione: seleziona Database standard.
   - Nome: inserisci un nome di database a tua scelta.
   - Crittografia: scegli una KMS chiave o utilizza l'opzione predefinita, in cui Timestream Live Analytics creerà una KMS chiave nel tuo account se non ne esiste già una.
5. Fai clic su Crea database per creare un database.

Creare una tabella

Segui questi passaggi per creare una tabella utilizzando la AWS console.

1. Apri la AWS console.
2. Nel riquadro di navigazione, scegli Tabelle
3. Fai clic su Crea tabella.
4. Nella pagina di creazione della tabella, inserisci quanto segue.
   - Nome database: selezionare il nome del database creato inCreazione di un database .
   - Nome tabella: immetti un nome di tabella a tua scelta.
   - Conservazione nell'archivio di memoria: specifica per quanto tempo desideri conservare i dati nell'archivio di memoria. L'archivio di memoria elabora i dati in entrata, compresi i dati in arrivo in ritardo (dati con un timestamp precedente all'ora corrente) ed è ottimizzato per query veloci. point-in-time
   - Conservazione dell'archivio magnetico: specifica per quanto tempo desideri conservare i dati nell'archivio magnetico. L'archivio magnetico è pensato per l'archiviazione a lungo termine ed è ottimizzato per query analitiche veloci.

5. Fai clic su Crea tabella.

Esecuzione di una query

Segui questi passaggi per eseguire le query utilizzando la AWS console.

1. Apri la AWS console.
2. Nel riquadro di navigazione, scegli Query editor
3. Nel riquadro sinistro, seleziona il database creato inCreazione di un database .
4. Nel riquadro sinistro, seleziona il database creato inCreare una tabella.
5. Nell'editor di query, è possibile eseguire una query. Per visualizzare le ultime 10 righe della tabella, esegui:

```
SELECT * FROM <database_name>.<table_name> ORDER BY time DESC LIMIT 10
```

6. (Facoltativo) Attiva Enable Insights per ottenere informazioni sull'efficienza delle tue query.

Crea una query pianificata

Segui questi passaggi per creare un'interrogazione pianificata utilizzando la AWS console.

1. Apri la AWS console.
2. Nel riquadro di navigazione, scegli Interrogazioni pianificate.
3. Fai clic su Crea interrogazione pianificata.
4. Nelle sezioni Nome della query e Tabella di destinazione, inserisci quanto segue.
   - Nome: immettere il nome di una query.
   - Nome del database: selezionare il nome del database creato in. Creazione di un database
   - Nome tabella: seleziona il nome della tabella creata in. Creare una tabella
5. Nella sezione Query Statement, immettete un'istruzione di interrogazione valida. Quindi fai clic su Convalida interrogazione.
6. Da Modello di tabella di destinazione, definite il modello per tutti gli attributi non definiti. È possibile utilizzare Visual builder o. JSON

7. Nella sezione Esegui pianificazione, scegli Frequenza fissa o Espressione Chron. Per le espressioni croniche, consulta Schedule Expressions for Scheduled Queries per maggiori dettagli sulle espressioni di pianificazione.
8. Nella sezione SNSargomento, inserisci l'SNSargomento a cui verrà utilizzata la notifica.
9. Nella sezione Report del registro degli errori, inserisci la posizione S3 che verrà utilizzata per segnalare gli errori.

   Scegli un valore per Encryption key type (Tipo di chiave di crittografia).
10. Nella sezione Impostazioni di sicurezza della AWS KMSchiave, scegli il tipo di AWS KMS chiave.

    Inserisci il IAMruolo che Timestream for LiveAnalytics utilizzerà per eseguire la query pianificata. Fai riferimento agli esempi di IAM policy per le query pianificate per i dettagli sulle autorizzazioni richieste e sulla relazione di fiducia per il ruolo.
11. Fai clic su Crea interrogazione pianificata.

Eliminare un'interrogazione pianificata

Segui questi passaggi per eliminare o disabilitare un'interrogazione pianificata utilizzando la AWS Console.

1. Apri la AWS console.
2. Nel riquadro di navigazione, scegli Interrogazioni pianificate
3. Seleziona l'interrogazione pianificata creata inCrea una query pianificata.
4. Seleziona Azioni.
5. Scegli Disabilita o Elimina.
6. Se hai selezionato Elimina, conferma l'azione e seleziona Elimina.

Eliminazione di una tabella

Segui questi passaggi per eliminare un database utilizzando la AWS console.

1. Apri la AWS console.
2. Nel riquadro di navigazione, scegli Tabelle
3. Seleziona la tabella in cui hai creatoCreare una tabella.
4. Fai clic su Delete (Elimina).

5. Digita delete nella casella di conferma.

Eliminazione di un database

Segui questi passaggi per eliminare un database utilizzando la AWS console:

1. Apri la AWS console.
2. Nel riquadro di navigazione, scegli Database
3. Seleziona il database che hai creato in Crea un database.
4. Fai clic su Delete (Elimina).
5. Digita delete nella casella di conferma.

Modifica una tabella

Segui questi passaggi per modificare una tabella utilizzando la AWS console.

1. Apri la AWS console.
2. Nel riquadro di navigazione, scegli Tabelle
3. Seleziona la tabella in cui hai creatoCreare una tabella.
4. Fate clic su Modifica
5. Modifica i dettagli della tabella e salva.

    - Conservazione dell'archivio di memoria: specifica per quanto tempo desideri conservare i dati nell'archivio di memoria. L'archivio di memoria elabora i dati in entrata, compresi i dati in arrivo in ritardo (dati con un timestamp precedente all'ora corrente) ed è ottimizzato per query veloci. point-in-time
    - Conservazione dell'archivio magnetico: specifica per quanto tempo desideri conservare i dati nell'archivio magnetico. L'archivio magnetico è pensato per l'archiviazione a lungo termine ed è ottimizzato per query analitiche veloci.

Modifica un database

Segui questi passaggi per modificare un database utilizzando la AWS console.

1. Apri la AWS console.

2. Nel riquadro di navigazione, scegli Database
3. Seleziona il database che hai creato in Crea un database.
4. Fai clic su Modifica
5. Modifica i dettagli del database e salva.

Accesso ad Amazon Timestream LiveAnalytics per l'utilizzo di AWS CLI

È possibile utilizzare il AWS Command Line Interface (AWS CLI) per controllare più AWS servizi dalla riga di comando e automatizzarli tramite script. È possibile utilizzare il AWS CLI per operazioni ad hoc. Puoi anche usarlo per incorporare Amazon LiveAnalytics Timestream for operations all'interno di script di utilità.

Prima di poterlo utilizzare AWS CLI con Timestream per LiveAnalytics, devi configurare l'accesso programmatico. Per ulteriori informazioni, consulta Concessione dell'accesso programmatico.

Per un elenco completo di tutti i comandi disponibili per Timestream for LiveAnalytics Query API in AWS CLI, consulta il Command Reference.AWS CLI

Per un elenco completo di tutti i comandi disponibili per Timestream for LiveAnalytics Write API in the AWS CLI, vedere il Command Reference.AWS CLI

Argomenti

- Download e configurazione dell' AWS CLI
- Utilizzo di AWS CLI con Timestream per LiveAnalytics

Download e configurazione dell' AWS CLI

AWS CLI Funziona su Windows, macOS o Linux. Per scaricarlo, installarlo e configurarlo, procedi nel seguente modo:

1. Scaricala AWS CLI da http://aws.amazon.com/cli.
2. Segui le istruzioni per l'installazione AWS CLI e la configurazione riportate AWS CLI nella Guida per l'AWS Command Line Interface utente.

Utilizzo di AWS CLI con Timestream per LiveAnalytics

Il formato della riga di comando è costituito da un Amazon Timestream LiveAnalytics per il nome dell'operazione, seguito dai parametri per tale operazione. AWS CLI Supporta una sintassi abbreviata per i valori dei parametri, oltre a. JSON

Usa `help` per elencare tutti i comandi disponibili in Timestream per. LiveAnalytics Per esempio:

```
aws timestream-write help
```

```
aws timestream-query help
```

Puoi anche usare `help` per descrivere un comando specifico e saperne di più sul suo utilizzo:

```
aws timestream-write create-database help
```

Ad esempio, per creare un database:

```
aws timestream-write create-database --database-name myFirstDatabase
```

Per creare una tabella con le scritture magnetiche abilitate:

```
aws timestream-write create-table \
--database-name metricsdb \
--table-name metrics \
--magnetic-store-write-properties "{\"EnableMagneticStoreWrites\": true}"
```

Per scrivere dati utilizzando record a misura singola:

```
aws timestream-write write-records \
--database-name metricsdb \
--table-name metrics \
--common-attributes "{\"Dimensions\":[{\"Name\":\"asset_id\", \"Value\":\"100\"}],
 \"Time\":\"1631051324000\",\"TimeUnit\":\"MILLISECONDS\"}" \
--records "[{\"MeasureName\":\"temperature\", \"MeasureValueType\":\"DOUBLE\",
\"MeasureValue\":\"30\"},{\"MeasureName\":\"windspeed\", \"MeasureValueType\":\"DOUBLE
\",\"MeasureValue\":\"7\"},{\"MeasureName\":\"humidity\", \"MeasureValueType\":\"DOUBLE
\",\"MeasureValue\":\"15\"},{\"MeasureName\":\"brightness\", \"MeasureValueType\":
\"DOUBLE\",\"MeasureValue\":\"17\"}]"
```

Per scrivere dati utilizzando record a più misure:

```
# wide model helper method to create Multi-measure records
function ingest_multi_measure_records {
  epoch=`date +%s`
  epoch+=$i

  # multi-measure records
  aws timestream-write write-records \
  --database-name $src_db_wide \
  --table-name $src_tbl_wide \
  --common-attributes "{\"Dimensions\":[{\"Name\":\"device_id\", \
              \"Value\":\"12345678\"},\
            {\"Name\":\"device_type\", \"Value\":\"iPhone\"}, \
            {\"Name\":\"os_version\", \"Value\":\"14.8\"}, \
            {\"Name\":\"region\", \"Value\":\"us-east-1\"} ], \
            \"Time\":\"$epoch\",\"TimeUnit\":\"MILLISECONDS\"}" \
--records "[{\"MeasureName\":\"video_metrics\", \"MeasureValueType\":\"MULTI\", \
  \"MeasureValues\": \
  [{\"Name\":\"video_startup_time\",\"Value\":\"0\",\"Type\":\"BIGINT\"}, \
  {\"Name\":\"rebuffering_ratio\",\"Value\":\"0.5\",\"Type\":\"DOUBLE\"}, \
  {\"Name\":\"video_playback_failures\",\"Value\":\"0\",\"Type\":\"BIGINT\"}, \
  {\"Name\":\"average_frame_rate\",\"Value\":\"0.5\",\"Type\":\"DOUBLE\"}]}]" \
--endpoint-url $ingest_endpoint \
  --region  $region
}

# create 5 records
for i in {100..105};
  do ingest_multi_measure_records $i;
done
```

Esecuzione di query su una tabella:

```
aws timestream-query query \
--query-string "SELECT time, device_id, device_type, os_version,
region, video_startup_time, rebuffering_ratio, video_playback_failures, \
average_frame_rate \
FROM metricsdb.metrics \
where time >= ago (15m)"
```

Per creare un'interrogazione pianificata:

```
aws timestream-query create-scheduled-query \
  --name scheduled_query_name \
  --query-string "select bin(time, 1m) as time, \
          avg(measure_value::double) as avg_cpu, min(measure_value::double) as min_cpu,
 region \
          from $src_db.$src_tbl where measure_name = 'cpu' \
          and time BETWEEN @scheduled_runtime - (interval '5' minute)  AND
 @scheduled_runtime \
          group by region, bin(time, 1m)" \
  --schedule-configuration "{\"ScheduleExpression\":\"$cron_exp\"}" \
  --notification-configuration "{\"SnsConfiguration\":{\"TopicArn\":\"$sns_topic_arn
\"}}" \
  --scheduled-query-execution-role-arn "arn:aws:iam::452360119086:role/
TimestreamSQExecutionRole" \
  --target-configuration "{\"TimestreamConfiguration\":{\
          \"DatabaseName\": \"$dest_db\",\
          \"TableName\": \"$dest_tbl\",\
          \"TimeColumn\":\"time\",\
          \"DimensionMappings\":[{\
            \"Name\": \"region\", \"DimensionValueType\": \"VARCHAR\"
          }],\
          \"MultiMeasureMappings\":{\
            \"TargetMultiMeasureName\": \"mma_name\",
            \"MultiMeasureAttributeMappings\":[{\
              \"SourceColumn\": \"avg_cpu\", \"MeasureValueType\": \"DOUBLE\",
 \"TargetMultiMeasureAttributeName\": \"target_avg_cpu\"
            },\
            { \
              \"SourceColumn\": \"min_cpu\", \"MeasureValueType\": \"DOUBLE\",
 \"TargetMultiMeasureAttributeName\": \"target_min_cpu\"
            }] \
          }\
          }}" \
  --error-report-configuration "{\"S3Configuration\": {\
        \"BucketName\": \"$s3_err_bucket\",\
        \"ObjectKeyPrefix\": \"scherrors\",\
        \"EncryptionOption\": \"SSE_S3\"\
        }\
      }"
```

Usando il API

Oltre a SDKs, Amazon Timestream LiveAnalytics for fornisce l'accesso REST API diretto tramite il pattern di rilevamento degli endpoint. Il modello di rilevamento degli endpoint è descritto di seguito, insieme ai relativi casi d'uso.

Il modello di scoperta degli endpoint

Poiché Timestream Live Analytics è SDKs progettato per funzionare in modo trasparente con l'architettura del servizio, inclusa la gestione e la mappatura degli endpoint del servizio, si consiglia di utilizzarlo per la maggior parte delle applicazioni. SDKs Tuttavia, ci sono alcuni casi in cui è necessario utilizzare il modello Timestream for endpoint discovery: LiveAnalytics REST API

- Stai usando VPCendpoints () con Timestream per AWS PrivateLink LiveAnalytics
- L'applicazione utilizza un linguaggio di programmazione che non supporta ancora SDK
- È necessario un controllo migliore sull'implementazione lato client

Questa sezione include informazioni su come funziona il pattern di rilevamento degli endpoint, su come implementarlo e note sull'utilizzo. Seleziona uno degli argomenti seguenti per saperne di più.

Argomenti

- Come funziona il modello di scoperta degli endpoint
- Implementazione del modello di rilevamento degli endpoint

Come funziona il modello di scoperta degli endpoint

Timestream è costruito utilizzando un'architettura cellulare per garantire una migliore scalabilità e proprietà di isolamento del traffico. Poiché ogni account cliente è mappato su una cella specifica in una regione, l'applicazione deve utilizzare gli endpoint corretti specifici della cella a cui è stato mappato l'account. Quando si utilizzaSDKs, questa mappatura viene gestita in modo trasparente e non è necessario gestire gli endpoint specifici della cella. Tuttavia, quando si accede direttamente a RESTAPI, sarà necessario gestire e mappare autonomamente gli endpoint corretti. Questo processo, il modello di rilevamento degli endpoint, è descritto di seguito:

1. Il pattern di rilevamento degli endpoint inizia con una chiamata all'`DescribeEndpoints`azione (descritta nella `DescribeEndpoints`sezione).

2. L'endpoint deve essere memorizzato nella cache e riutilizzato per il periodo di tempo specificato dal valore restituito time-to-live (TTL) (the). `CachePeriodInMinutes` È quindi API possibile effettuare chiamate a Timestream Live Analytics per la durata del. TTL
3. Dopo la TTL scadenza, è DescribeEndpoints necessario effettuare una nuova chiamata a per aggiornare l'endpoint (in altre parole, ricominciare dal passaggio 1).

> **Note**
>
> La sintassi, i parametri e altre informazioni sull'utilizzo dell'`DescribeEndpoints`azione sono descritte nella Guida di riferimento. API Nota che l'`DescribeEndpoints`azione è disponibile tramite entrambi SDKs ed è identica per ciascuno.

Per l'implementazione del modello di rilevamento degli endpoint, vedereImplementazione del modello di rilevamento degli endpoint.

Implementazione del modello di rilevamento degli endpoint

Per implementare il modello di rilevamento degli endpoint, scegliete un API (Write o Query), create una DescribeEndpointsrichiesta e utilizzate gli endpoint restituiti per la durata dei TTL valori restituiti. La procedura di implementazione è descritta di seguito.

> **Note**
>
> Assicurati di avere familiarità con le note sull'utilizzo.

Procedura di implementazione

1. Acquisisci l'endpoint per il quale API desideri effettuare chiamate (Write o Query). utilizzando la richiesta. `DescribeEndpoints`
   a. Crea una richiesta `DescribeEndpoints`corrispondente all'oggetto API di interesse (Write o Query) utilizzando uno dei due endpoint descritti di seguito. Non ci sono parametri di input per la richiesta. Assicurati di leggere le note riportate di seguito.

ScriviSDK:

```
ingest.timestream.<region>.amazonaws.com
```

InterrogazioneSDK:

```
query.timestream.<region>.amazonaws.com
```

`us-east-1`Segue un esempio di CLI chiamata per la regione.

```
REGION_ENDPOINT="https://query.timestream.us-east-1.amazonaws.com"
REGION=us-east-1
aws timestream-write describe-endpoints \
--endpoint-url $REGION_ENDPOINT \
--region $REGION
```

> **Note**
>
> L'intestazione HTTP «Host» deve contenere anche l'APIendpoint. La richiesta avrà esito negativo se l'intestazione non è compilata. Questo è un requisito standard per tutte le richieste HTTP /1.1. Se utilizzi una HTTP libreria che supporta 1.1 o versioni successive, la HTTP libreria dovrebbe compilare automaticamente l'intestazione per te.

> **Note**
>
> Sostituto *<region>* con l'identificatore della regione in cui viene effettuata la richiesta, ad es. `us-east-1`

b. Analizza la risposta per estrarre gli endpoint e i TTL valori della cache. La risposta è una matrice di uno o più `Endpoint` oggetti. Ogni `Endpoint` oggetto contiene un indirizzo endpoint (`Address`) e il TTL relativo endpoint (`CachePeriodInMinutes`).

2. Memorizza l'endpoint nella cache per un massimo di quanto specificato. TTL
3. Alla TTL scadenza, recupera un nuovo endpoint ricominciando dalla fase 1 dell'implementazione.

Note d'uso per il modello di rilevamento degli endpoint

- L'DescribeEndpointsazione è l'unica azione riconosciuta dagli endpoint regionali di Timestream Live Analytics.
- La risposta contiene un elenco di endpoint contro cui effettuare chiamate Timestream Live Analytics. API
- In caso di risposta corretta, dovrebbe esserci almeno un endpoint nell'elenco. Se nell'elenco è presente più di un endpoint, ognuno di essi è ugualmente utilizzabile per API le chiamate e il chiamante può scegliere l'endpoint da utilizzare a caso.
- Oltre all'DNSindirizzo dell'endpoint, ogni endpoint nell'elenco specificherà un time to live (TTL) consentito per l'utilizzo dell'endpoint specificato in minuti.
- L'endpoint deve essere memorizzato nella cache e riutilizzato per il periodo di tempo specificato dal valore restituito (in minuti). TTL Dopo la TTL scadenza, è DescribeEndpointsnecessario effettuare una nuova chiamata a per aggiornare l'endpoint da utilizzare, poiché l'endpoint non funzionerà più dopo la scadenza. TTL

Usando il AWS SDKs

Puoi accedere ad Amazon Timestream utilizzando. AWS SDKs Timestream ne supporta due SDKs per lingua; vale a dire, Write SDK e Query. SDK Il Write SDK viene utilizzato per eseguire CRUD operazioni e inserire i dati delle serie temporali in Timestream. La Query SDK viene utilizzata per interrogare i dati delle serie temporali esistenti archiviati in Timestream.

Dopo aver completato i prerequisiti necessari per la tua SDK scelta, puoi iniziare con. Esempi di codice

Argomenti

- Java
- Java v2
- Go
- Python
- Node.js
- .NET

Java

Per iniziare a usare Java 1.0 SDK e Amazon Timestream, completa i prerequisiti descritti di seguito.

Dopo aver completato i prerequisiti necessari per JavaSDK, puoi iniziare con. Esempi di codice

Prerequisiti

Prima di iniziare a usare Java, è necessario effettuare le seguenti operazioni:

1. Segui le istruzioni AWS di configurazione riportate inAccesso a Timestream per LiveAnalytics.
2. Configura un ambiente di sviluppo Java scaricando e installando quanto segue:
   - Kit di sviluppo Java SE 8 (ad esempio Amazon Corretto 8).
   - Java IDE (come Eclipse o IntelliJ).

   Per ulteriori informazioni, vedere Guida introduttiva a AWS SDK for Java
3. Configura AWS le tue credenziali e la regione per lo sviluppo:
   - Imposta le tue credenziali AWS di sicurezza da utilizzare con. AWS SDK for Java
   - Imposta la tua AWS regione per determinare il tuo Timestream predefinito per l'endpoint. LiveAnalytics

Utilizzo di Apache Maven

Puoi usare Apache Maven per configurare e creare progetti. AWS SDK for Java

> **Note**
>
> Per usare Apache Maven, assicurati che Java SDK e il runtime siano 1.8 o superiori.

È possibile configurarla come dipendenza Maven AWS SDK come descritto in Uso con Apache Maven. SDK

È possibile eseguire la compilazione ed eseguire il codice sorgente con il seguente comando:

```
mvn clean compile
mvn exec:java -Dexec.mainClass=<your source code Main class>
```

> **Note**
>
> `<your source code Main class>` è il percorso della classe principale del codice sorgente Java.

Impostazione delle AWS credenziali

È AWS SDK for Javanecessario fornire AWS le credenziali all'applicazione in fase di esecuzione. Gli esempi di codice in questa guida presuppongono che si stia utilizzando un file di AWS credenziali, come descritto nella sezione Configurazione AWS delle credenziali e della regione per lo sviluppo nella Guida per gli AWS SDK for Java sviluppatori.

Di seguito è riportato un esempio di file di AWS credenziali denominato`~/.aws/credentials`, in cui il carattere tilde (~) rappresenta la directory home.

```
[default]
aws_access_key_id = AWS access key ID goes here
aws_secret_access_key = Secret key goes here
```

Java v2

Per iniziare a usare Java 2.0 SDK e Amazon Timestream, completa i prerequisiti descritti di seguito.

Dopo aver completato i prerequisiti necessari per Java 2.0SDK, puoi iniziare con. Esempi di codice

Prerequisiti

Prima di iniziare a usare Java, è necessario effettuare le seguenti operazioni:

1. Segui le istruzioni AWS di configurazione riportate inAccesso a Timestream per LiveAnalytics.
2. È possibile configurarla AWS SDK come dipendenza Maven come descritto in Uso SDK con Apache Maven.
3. Configura un ambiente di sviluppo Java scaricando e installando quanto segue:

   - Kit di sviluppo Java SE 8 (ad esempio Amazon Corretto 8).
   - Java IDE (come Eclipse o IntelliJ).

     Per ulteriori informazioni, vedere Guida introduttiva a AWS SDK for Java

Utilizzo di Apache Maven

Puoi usare Apache Maven per configurare e creare progetti. AWS SDK for Java

> **Note**
>
> Per usare Apache Maven, assicurati che Java SDK e il runtime siano 1.8 o superiori.

È possibile configurarla come dipendenza Maven AWS SDK come descritto in Uso con Apache Maven. SDK Le modifiche richieste al file pom.xml sono descritte qui.

È possibile eseguire compile ed eseguire il codice sorgente con il seguente comando:

```
mvn clean compile
mvn exec:java -Dexec.mainClass=<your source code Main class>
```

> **Note**
>
> `<your source code Main class>` è il percorso della classe principale del codice sorgente Java.

Go

Per iniziare a usare Go SDK e Amazon Timestream, completa i prerequisiti descritti di seguito.

Dopo aver completato i prerequisiti necessari per GoSDK, puoi iniziare con. Esempi di codice

Prerequisiti

1. Scarica GO SDK 1.14.
2. Configura GO SDK.
3. Costruisci il tuo cliente.

Python

Per iniziare a usare Python SDK e Amazon Timestream, completa i prerequisiti descritti di seguito.

Una volta completati i prerequisiti necessari per SDK Python, puoi iniziare con. Esempi di codice

Prerequisiti

Per usare Python, installa e configura Boto3 seguendo le istruzioni qui.

Node.js

Per iniziare a usare Node.js SDK e Amazon Timestream, completa i prerequisiti descritti di seguito.

Dopo aver completato i prerequisiti necessari per Node.jsSDK, puoi iniziare con. Esempi di codice

Prerequisiti

Prima di iniziare con Node.js, è necessario effettuare le seguenti operazioni:

1. Installa Node.js.
2. Installa AWS SDK il modulo JavaScript.

.NET

Per iniziare con. NETSDKe Amazon Timestream, completano i prerequisiti descritti di seguito.

Dopo aver completato i prerequisiti necessari per. NETSDK, puoi iniziare con. Esempi di codice

Prerequisiti

Prima di iniziare con. NET, installa i NuGet pacchetti richiesti e assicurati che la AWSSDK versione.Core sia 3.3.107 o successiva eseguendo i seguenti comandi:

```
dotnet add package AWSSDK.Core
dotnet add package AWSSDK.TimestreamWrite
dotnet add package AWSSDK.TimestreamQuery
```

Nozioni di base

Questa sezione include un tutorial per iniziare a usare Amazon Timestream Live Analytics, oltre a istruzioni per configurare un'applicazione di esempio completamente funzionale. Puoi iniziare con il tutorial o l'applicazione di esempio selezionando uno dei link sottostanti.

Argomenti

- Tutorial
- Applicazione di esempio

Tutorial

Questo tutorial mostra come creare un database popolato con set di dati di esempio ed eseguire query di esempio. I set di dati di esempio utilizzati in questo tutorial sono spesso presenti nell'IoT e negli DevOps scenari. Il set di dati IoT contiene dati di serie temporali come la velocità, la posizione e il carico di un camion, per semplificare la gestione della flotta e identificare opportunità di ottimizzazione. Il set di DevOps dati contiene metriche di EC2 istanza come CPU l'utilizzo della rete e della memoria per migliorare le prestazioni e la disponibilità delle applicazioni. Ecco un video tutorial per le istruzioni descritte in questa sezione.

Segui questi passaggi per creare un database popolato con i set di dati di esempio ed eseguire query di esempio utilizzando la AWS Console:

Utilizzo della console

Segui questi passaggi per creare un database popolato con i set di dati di esempio ed eseguire query di esempio utilizzando la Console: AWS

1. Apri la console.AWS
2. Nel riquadro di navigazione, scegli Database
3. Fai clic su Crea database.
4. Nella pagina di creazione del database, inserisci quanto segue:

   - Scegli la configurazione: seleziona il database di esempio.
   - Nome: inserisci un nome di database a tua scelta.

     > **Note**
     >
     > Dopo aver creato un database con set di dati di esempio, per utilizzare le query di esempio disponibili nella console, è possibile modificare il nome del database a cui si fa riferimento nella query in modo che corrisponda al nome del database immesso qui. Sono disponibili query di esempio per ogni combinazione di set di dati di esempio e tipo di record di serie temporali.

   - Scegli set di dati di esempio: seleziona IoT e DevOps.
   - Scegli il tipo di record delle serie temporali —Seleziona record multimisura.
   - Fai clic su Crea database per creare un database contenente due tabelle popolate con dati di esempio. I nomi delle tabelle per i set di dati di esempio con record multimisura sono

`DevOpsMulti` e. `IoTMulti` I nomi delle tabelle per set di dati di esempio con record a misura singola sono e. `DevOps` `IoT`

5. Nel riquadro di navigazione, scegli Query editor
6. Seleziona Query di esempio dal menu in alto.
7. Fai clic su una delle query di esempio per un set di dati scelto durante la creazione del database di esempio. In questo modo si torna all'editor di query con l'editor popolato con la query di esempio.
8. Modifica il nome del database per la query di esempio.
9. Fate clic su Esegui per eseguire la query e visualizzare i risultati della query.

Usando il SDKs

Timestream Live Analytics fornisce un'applicazione di esempio completamente funzionale che mostra come creare un database e una tabella, popolare la tabella con ~126.000 righe di dati di esempio ed eseguire query di esempio. L'applicazione di esempio è disponibile in GitHubJava, Python, Node.js, Go e. NET.

1. Clona le applicazioni di esempio del GitHub repository Timestream Live Analytics seguendo le istruzioni di. GitHub
2. Configura la connessione AWS SDK ad Amazon Timestream Live Analytics seguendo le istruzioni descritte in. Usando il AWS SDKs
3. Compila ed esegui l'applicazione di esempio utilizzando le istruzioni seguenti:

   - Istruzioni per l'applicazione di esempio Java.
   - Istruzioni per l'applicazione di esempio Java v2.
   - Istruzioni per l'applicazione di esempio Go.
   - Istruzioni per l'applicazione di esempio Python.
   - Istruzioni per l'applicazione di esempio Node.js.
   - Istruzioni per. NETapplicazione di esempio.

Applicazione di esempio

Timestream viene fornito con un'applicazione di esempio completamente funzionale che mostra come creare un database e una tabella, popolare la tabella con ~126.000 righe di dati di esempio

ed eseguire query di esempio. Segui i passaggi seguenti per iniziare a utilizzare l'applicazione di esempio in una delle lingue supportate:

Java

1. Clona il GitHub repository Timestream per applicazioni di LiveAnalytics esempio seguendo le istruzioni di. GitHub
2. Configura il file AWS SDK per connetterti a Timestream LiveAnalytics seguendo le istruzioni descritte in Guida introduttiva a. Java
3. Esegui l'applicazione di esempio Java seguendo le istruzioni qui descritte

Java v2

1. Clona il GitHub repository Timestream per applicazioni di LiveAnalytics esempio seguendo le istruzioni di. GitHub
2. Configura la connessione AWS SDK ad Amazon Timestream LiveAnalytics per seguire le istruzioni descritte in Getting Started with. Java v2
3. Esegui l'applicazione di esempio Java 2.0 seguendo le istruzioni descritte qui

Go

1. Clona il GitHub repository Timestream per applicazioni di LiveAnalytics esempio seguendo le istruzioni di. GitHub
2. Configura la connessione AWS SDK ad Amazon Timestream LiveAnalytics per seguire le istruzioni descritte in Getting Started with. Go
3. Esegui l'applicazione di esempio Go seguendo le istruzioni descritte qui

Python

1. Clona il GitHub repository Timestream per applicazioni di LiveAnalytics esempio seguendo le istruzioni di. GitHub
2. Configura la connessione AWS SDK ad Amazon Timestream LiveAnalytics per seguire le istruzioni descritte in. Python
3. Esegui l'applicazione di esempio Python seguendo le istruzioni qui descritte

Node.js

1. Clona il GitHub repository Timestream per applicazioni di LiveAnalytics esempio seguendo le istruzioni di. GitHub
2. Configura la connessione AWS SDK ad Amazon Timestream LiveAnalytics per seguire le istruzioni descritte in Getting Started with. Node.js
3. Esegui l'applicazione di esempio Node.js seguendo le istruzioni descritte qui

.NET

1. Clona il GitHub repository Timestream per applicazioni di LiveAnalytics esempio seguendo le istruzioni di. GitHub
2. Configura la connessione AWS SDK ad Amazon Timestream LiveAnalytics per seguire le istruzioni descritte in Getting Started with. .NET
3. Esegui il. NETapplicazione di esempio seguendo le istruzioni qui descritte

Esempi di codice

Puoi accedere ad Amazon Timestream utilizzando. AWS SDKs Timestream ne supporta due SDKs per lingua; vale a dire, Write SDK e Query. SDK Il Write SDK viene utilizzato per eseguire CRUD operazioni e inserire i dati delle serie temporali in Timestream. La Query SDK viene utilizzata per interrogare i dati delle serie temporali esistenti archiviati in Timestream. Seleziona un argomento dall'elenco seguente per maggiori dettagli, inclusi esempi di codice per ciascuno dei supporti. SDKs

Argomenti

- Scrivi SDK cliente
- SDKClient di interrogazione
- Crea database
- Descrivi database
- Aggiorna database
- Eliminare il database
- Elenca database
- Create table (Crea tabella)
- Descrivi tabella

- Update Table (Aggiorna tabella)
- Delete Table (Elimina tabella)
- Elencare tabelle
- Scrivere dati (inserti e sconvolgimenti)
- Esegui interrogazione
- Esegui UNLOAD interrogazione
- Annulla interrogazione
- Crea attività di caricamento in batch
- Descrizione dell'attività di caricamento in batch
- Elenca le attività di caricamento in batch
- Riprendi l'operazione di caricamento in batch
- Crea una query pianificata
- Elenca le query pianificate
- Descrivi una query pian
- Esegui interrogazione pianificata
- Aggiorna interrogazione pianificata
- Eliminare una query pianificata

Scrivi SDK cliente

È possibile utilizzare i seguenti frammenti di codice per creare un client Timestream per Write. SDK Il Write SDK viene utilizzato per eseguire CRUD operazioni e inserire i dati delle serie temporali in Timestream.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
    private static AmazonTimestreamWrite buildWriteClient() {
```

```
        final ClientConfiguration clientConfiguration = new ClientConfiguration()
                .withMaxConnections(5000)
                .withRequestTimeout(20 * 1000)
                .withMaxErrorRetry(10);

        return AmazonTimestreamWriteClientBuilder
                .standard()
                .withRegion("us-east-1")
                .withClientConfiguration(clientConfiguration)
                .build();
    }
```

Java v2

```
    private static TimestreamWriteClient buildWriteClient() {
        ApacheHttpClient.Builder httpClientBuilder =
                ApacheHttpClient.builder();
        httpClientBuilder.maxConnections(5000);

        RetryPolicy.Builder retryPolicy =
                RetryPolicy.builder();
        retryPolicy.numRetries(10);

        ClientOverrideConfiguration.Builder overrideConfig =
                ClientOverrideConfiguration.builder();
        overrideConfig.apiCallAttemptTimeout(Duration.ofSeconds(20));
        overrideConfig.retryPolicy(retryPolicy.build());

        return TimestreamWriteClient.builder()
                .httpClientBuilder(httpClientBuilder)
                .overrideConfiguration(overrideConfig.build())
                .region(Region.US_EAST_1)
                .build();
    }
```

Go

```
tr := &http.Transport{
        ResponseHeaderTimeout: 20 * time.Second,
        // Using DefaultTransport values for other parameters: https://golang.org/
pkg/net/http/#RoundTripper
        Proxy: http.ProxyFromEnvironment,
        DialContext: (&net.Dialer{
```

```
            KeepAlive: 30 * time.Second,
            DualStack: true,
            Timeout:   30 * time.Second,
        }).DialContext,
        MaxIdleConns:          100,
        IdleConnTimeout:       90 * time.Second,
        TLSHandshakeTimeout:   10 * time.Second,
        ExpectContinueTimeout: 1 * time.Second,
    }

    // So client makes HTTP/2 requests
    http2.ConfigureTransport(tr)

    sess, err := session.NewSession(&aws.Config{ Region: aws.String("us-east-1"),
 MaxRetries: aws.Int(10), HTTPClient: &http.Client{ Transport: tr }})
    writeSvc := timestreamwrite.New(sess)
```

Python

```
write_client = session.client('timestream-write', config=Config(read_timeout=20,
 max_pool_connections = 5000, retries={'max_attempts': 10}))
```

Node.js

Il seguente frammento viene utilizzato AWS SDK per JavaScript la v3. Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Qui viene mostrata un'ulteriore importazione di comandi. L'`CreateDatabaseCommand`importazione non è richiesta per creare il client.

```
import { TimestreamWriteClient, CreateDatabaseCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });
```

Il seguente frammento utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
var https = require('https');
var agent = new https.Agent({
    maxSockets: 5000
```

```
});
writeClient = new AWS.TimestreamWrite({
        maxRetries: 10,
        httpOptions: {
            timeout: 20000,
            agent: agent
        }
    });
```

.NET

```
var writeClientConfig = new AmazonTimestreamWriteConfig
{
    RegionEndpoint = RegionEndpoint.USEast1,
    Timeout = TimeSpan.FromSeconds(20),
    MaxErrorRetry = 10
};

var writeClient = new AmazonTimestreamWriteClient(writeClientConfig);
```

Ti consigliamo di utilizzare la seguente configurazione.

- Imposta il numero SDK di tentativi su. `10`
- Utilizza `SDK DEFAULT_BACKOFF_STRATEGY`.
- Impostato `RequestTimeout` su `20` secondi.
- Imposta il numero massimo di connessioni `5000` pari o superiore.

SDKClient di interrogazione

È possibile utilizzare i seguenti frammenti di codice per creare un client Timestream per la Query. SDK La Query SDK viene utilizzata per interrogare i dati delle serie temporali esistenti archiviati in Timestream.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
    private static AmazonTimestreamQuery buildQueryClient() {
        AmazonTimestreamQuery client =
AmazonTimestreamQueryClient.builder().withRegion("us-east-1").build();
        return client;
    }
```

Java v2

```
    private static TimestreamQueryClient buildQueryClient() {
        return TimestreamQueryClient.builder()
                .region(Region.US_EAST_1)
                .build();
    }
```

Go

```
sess, err := session.NewSession(&aws.Config{Region: aws.String("us-east-1")})
```

Python

```
query_client = session.client('timestream-query')
```

Node.js

Il seguente frammento viene utilizzato AWS SDK per JavaScript la v3. Per ulteriori informazioni su come installare il client e sull'utilizzo, vedere Timestream Query Client -, per v3.AWS SDK JavaScript

Qui viene mostrata un'ulteriore importazione di comandi. L'`QueryCommand`importazione non è richiesta per creare il client.

```
import { TimestreamQueryClient, QueryCommand } from "@aws-sdk/client-timestream-
query";
const queryClient = new TimestreamQueryClient({ region: "us-east-1" });
```

Il seguente frammento utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
queryClient = new AWS.TimestreamQuery();
```

.NET

```
var queryClientConfig = new AmazonTimestreamQueryConfig
{
    RegionEndpoint = RegionEndpoint.USEast1
};

var queryClient = new AmazonTimestreamQueryClient(queryClientConfig);
```

Crea database

È possibile utilizzare i seguenti frammenti di codice per creare un database.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
  public void createDatabase() {
       System.out.println("Creating database");
       CreateDatabaseRequest request = new CreateDatabaseRequest();
       request.setDatabaseName(DATABASE_NAME);
       try {
           amazonTimestreamWrite.createDatabase(request);
           System.out.println("Database [" + DATABASE_NAME + "] created
successfully");
       } catch (ConflictException e) {
           System.out.println("Database [" + DATABASE_NAME + "] exists. Skipping
database creation");
       }
   }
```

Java v2

```
    public void createDatabase() {
        System.out.println("Creating database");
        CreateDatabaseRequest request =
 CreateDatabaseRequest.builder().databaseName(DATABASE_NAME).build();
        try {
            timestreamWriteClient.createDatabase(request);
            System.out.println("Database [" + DATABASE_NAME + "] created
 successfully");
        } catch (ConflictException e) {
            System.out.println("Database [" + DATABASE_NAME + "] exists. Skipping
 database creation");
        }
    }
```

Go

```
// Create database.
    createDatabaseInput := &timestreamwrite.CreateDatabaseInput{
        DatabaseName: aws.String(*databaseName),
    }

    _, err = writeSvc.CreateDatabase(createDatabaseInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Database successfully created")
    }

    fmt.Println("Describing the database, hit enter to continue")
```

Python

```
    def create_database(self):
        print("Creating Database")
        try:
            self.client.create_database(DatabaseName=Constant.DATABASE_NAME)
            print("Database [%s] created successfully." % Constant.DATABASE_NAME)
        except self.client.exceptions.ConflictException:
```

```
            print("Database [%s] exists. Skipping database creation" %
 Constant.DATABASE_NAME)
        except Exception as err:
            print("Create database failed:", err)
```

Node.js

Il seguente frammento viene utilizzato AWS SDK per JavaScript la v3. Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Vedi anche Class and. CreateDatabaseCommand CreateDatabase

```
import { TimestreamWriteClient, CreateDatabaseCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode"
};

const command = new CreateDatabaseCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Database ${data.Database.DatabaseName} created successfully`);
} catch (error) {
    if (error.code === 'ConflictException') {
        console.log(`Database ${params.DatabaseName} already exists. Skipping
 creation.`);
    } else {
        console.log("Error creating database", error);
    }
}
```

Il seguente frammento utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function createDatabase() {
    console.log("Creating Database");
    const params = {
```

```
        DatabaseName: constants.DATABASE_NAME
    };

    const promise = writeClient.createDatabase(params).promise();

    await promise.then(
        (data) => {
            console.log(`Database ${data.Database.DatabaseName} created
 successfully`);
        },
        (err) => {
            if (err.code === 'ConflictException') {
                console.log(`Database ${params.DatabaseName} already exists.
 Skipping creation.`);
            } else {
                console.log("Error creating database", err);
            }
        }
    );
}
```

.NET

```
        public async Task CreateDatabase()
        {
            Console.WriteLine("Creating Database");

            try
            {
                var createDatabaseRequest = new CreateDatabaseRequest
                {
                    DatabaseName = Constants.DATABASE_NAME
                };
                CreateDatabaseResponse response = await
 writeClient.CreateDatabaseAsync(createDatabaseRequest);
                Console.WriteLine($"Database {Constants.DATABASE_NAME} created");
            }
            catch (ConflictException)
            {
                Console.WriteLine("Database already exists.");
            }
            catch (Exception e)
            {
```

```
                Console.WriteLine("Create database failed:" + e.ToString());
            }

        }
```

Descrivi database

È possibile utilizzare i seguenti frammenti di codice per ottenere informazioni sugli attributi del database appena creato.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub
> Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, vedere.
> Applicazione di esempio

Java

```
   public void describeDatabase() {
       System.out.println("Describing database");
       final DescribeDatabaseRequest describeDatabaseRequest = new
DescribeDatabaseRequest();
       describeDatabaseRequest.setDatabaseName(DATABASE_NAME);
       try {
           DescribeDatabaseResult result =
amazonTimestreamWrite.describeDatabase(describeDatabaseRequest);
           final Database databaseRecord = result.getDatabase();
           final String databaseId = databaseRecord.getArn();
           System.out.println("Database " + DATABASE_NAME + " has id " +
databaseId);
       } catch (final Exception e) {
           System.out.println("Database doesn't exist = " + e);
           throw e;
       }
   }
```

Java v2

```
   public void describeDatabase() {
```

```
        System.out.println("Describing database");
        final DescribeDatabaseRequest describeDatabaseRequest =
 DescribeDatabaseRequest.builder()
                .databaseName(DATABASE_NAME).build();
        try {
            DescribeDatabaseResponse response =
 timestreamWriteClient.describeDatabase(describeDatabaseRequest);
            final Database databaseRecord = response.database();
            final String databaseId = databaseRecord.arn();
            System.out.println("Database " + DATABASE_NAME + " has id " +
 databaseId);
        } catch (final Exception e) {
            System.out.println("Database doesn't exist = " + e);
            throw e;
        }
    }
```

Go

```
describeDatabaseOutput, err := writeSvc.DescribeDatabase(describeDatabaseInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Describe database is successful, below is the output:")
        fmt.Println(describeDatabaseOutput)
    }
```

Python

```
    def describe_database(self):
        print("Describing database")
        try:
            result =
 self.client.describe_database(DatabaseName=Constant.DATABASE_NAME)
            print("Database [%s] has id [%s]" % (Constant.DATABASE_NAME,
 result['Database']['Arn']))
        except self.client.exceptions.ResourceNotFoundException:
            print("Database doesn't exist")
        except Exception as err:
            print("Describe database failed:", err)
```

Node.js

Il seguente frammento viene utilizzato AWS SDK per JavaScript la v3. Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Vedi anche Class and. DescribeDatabaseCommand DescribeDatabase

```
import { TimestreamWriteClient, DescribeDatabaseCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode"
};

const command = new DescribeDatabaseCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Database ${data.Database.DatabaseName} has id
 ${data.Database.Arn}`);
} catch (error) {
    if (error.code === 'ResourceNotFoundException') {
        console.log("Database doesn't exist.");
    } else {
        console.log("Describe database failed.", error);
        throw error;
    }
}
```

Il seguente frammento utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function describeDatabase () {
    console.log("Describing Database");
    const params = {
        DatabaseName: constants.DATABASE_NAME
    };

    const promise = writeClient.describeDatabase(params).promise();
```

```
    await promise.then(
        (data) => {
            console.log(`Database ${data.Database.DatabaseName} has id
 ${data.Database.Arn}`);
        },
        (err) => {
            if (err.code === 'ResourceNotFoundException') {
                console.log("Database doesn't exist.");
            } else {
                console.log("Describe database failed.", err);
                throw err;
            }
        }
    );
}
```

.NET

```
        public async Task DescribeDatabase()
        {
            Console.WriteLine("Describing Database");

            try
            {
                var describeDatabaseRequest = new DescribeDatabaseRequest
                {
                    DatabaseName = Constants.DATABASE_NAME
                };
                DescribeDatabaseResponse response = await
 writeClient.DescribeDatabaseAsync(describeDatabaseRequest);
                Console.WriteLine($"Database {Constants.DATABASE_NAME} has id:
{response.Database.Arn}");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine("Database does not exist.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Describe database failed:" + e.ToString());
            }

        }
```

Aggiorna database

È possibile utilizzare i seguenti frammenti di codice per aggiornare i database.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
   public void updateDatabase(String kmsId) {
       System.out.println("Updating kmsId to " + kmsId);
       UpdateDatabaseRequest request = new UpdateDatabaseRequest();
       request.setDatabaseName(DATABASE_NAME);
       request.setKmsKeyId(kmsId);
       try {
           UpdateDatabaseResult result =
amazonTimestreamWrite.updateDatabase(request);
           System.out.println("Update Database complete");
       } catch (final ValidationException e) {
           System.out.println("Update database failed:");
           e.printStackTrace();
       } catch (final ResourceNotFoundException e) {
           System.out.println("Database " + DATABASE_NAME + " doesn't exist = " +
e);
       } catch (final Exception e) {
           System.out.println("Could not update Database " + DATABASE_NAME + " = "
+ e);
           throw e;
       }
   }
```

Java v2

```
   public void updateDatabase(String kmsKeyId) {

       if (kmsKeyId == null) {
           System.out.println("Skipping UpdateDatabase because KmsKeyId was not
given");
```

```
            return;
        }

        System.out.println("Updating database");

        UpdateDatabaseRequest request = UpdateDatabaseRequest.builder()
                .databaseName(DATABASE_NAME)
                .kmsKeyId(kmsKeyId)
                .build();
        try {
            timestreamWriteClient.updateDatabase(request);
            System.out.println("Database [" + DATABASE_NAME + "] updated
 successfully with kmsKeyId " + kmsKeyId);
        } catch (ResourceNotFoundException e) {
            System.out.println("Database [" + DATABASE_NAME + "] does not exist.
 Skipping UpdateDatabase");
        } catch (Exception e) {
            System.out.println("UpdateDatabase failed: " + e);
        }
    }
```

Go

```
// Update Database.
        updateDatabaseInput := &timestreamwrite.UpdateDatabaseInput {
            DatabaseName: aws.String(*databaseName),
            KmsKeyId: aws.String(*kmsKeyId),
        }

        updateDatabaseOutput, err := writeSvc.UpdateDatabase(updateDatabaseInput)

        if err != nil {
            fmt.Println("Error:")
            fmt.Println(err)
        } else {
            fmt.Println("Update database is successful, below is the output:")
            fmt.Println(updateDatabaseOutput)
        }
```

Python

```
    def update_database(self, kms_id):
        print("Updating database")
```

```
        try:
            result =
 self.client.update_database(DatabaseName=Constant.DATABASE_NAME, KmsKeyId=kms_id)
            print("Database [%s] was updated to use kms [%s] successfully" %
 (Constant.DATABASE_NAME,

 result['Database']['KmsKeyId']))
        except self.client.exceptions.ResourceNotFoundException:
            print("Database doesn't exist")
        except Exception as err:
            print("Update database failed:", err)
```

Node.js

Il seguente frammento viene utilizzato AWS SDK per JavaScript la v3. Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Vedi anche Class and. UpdateDatabaseCommand UpdateDatabase

```
import { TimestreamWriteClient, UpdateDatabaseCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });
let updatedKmsKeyId = "<updatedKmsKeyId>";

const params = {
    DatabaseName: "testDbFromNode",
    KmsKeyId: updatedKmsKeyId
};

const command = new UpdateDatabaseCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Database ${data.Database.DatabaseName} updated kmsKeyId to
 ${updatedKmsKeyId}`);
} catch (error) {
    if (error.code === 'ResourceNotFoundException') {
        console.log("Database doesn't exist.");
    } else {
        console.log("Update database failed.", error);
    }
}
```

Il seguente frammento utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function updateDatabase(updatedKmsKeyId) {

    if (updatedKmsKeyId === undefined) {
        console.log("Skipping UpdateDatabase; KmsKeyId was not given");
        return;
    }
    console.log("Updating Database");
    const params = {
        DatabaseName: constants.DATABASE_NAME,
        KmsKeyId: updatedKmsKeyId
    }

    const promise = writeClient.updateDatabase(params).promise();

    await promise.then(
        (data) => {
            console.log(`Database ${data.Database.DatabaseName} updated kmsKeyId to
 ${updatedKmsKeyId}`);
        },
        (err) => {
            if (err.code === 'ResourceNotFoundException') {
                console.log("Database doesn't exist.");
            } else {
                console.log("Update database failed.", err);
            }
        }
    );
}
```

.NET

```
        public async Task UpdateDatabase(String updatedKmsKeyId)
        {
            Console.WriteLine("Updating Database");

            try
            {
                var updateDatabaseRequest = new UpdateDatabaseRequest
                {
```

```
                    DatabaseName = Constants.DATABASE_NAME,
                    KmsKeyId = updatedKmsKeyId
                };
                UpdateDatabaseResponse response = await
writeClient.UpdateDatabaseAsync(updateDatabaseRequest);
                Console.WriteLine($"Database {Constants.DATABASE_NAME} updated with
KmsKeyId {updatedKmsKeyId}");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine("Database does not exist.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Update database failed: " + e.ToString());
            }

        }

        private void PrintDatabases(List<Database> databases)
        {
            foreach (Database database in databases)
                Console.WriteLine($"Database:{database.DatabaseName}");
        }
```

Eliminare il database

È possibile utilizzare il seguente frammento di codice per eliminare un database.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
    public void deleteDatabase() {
        System.out.println("Deleting database");
```

```
        final DeleteDatabaseRequest deleteDatabaseRequest = new
DeleteDatabaseRequest();
        deleteDatabaseRequest.setDatabaseName(DATABASE_NAME);
        try {
            DeleteDatabaseResult result =
                    amazonTimestreamWrite.deleteDatabase(deleteDatabaseRequest);
            System.out.println("Delete database status: " +
result.getSdkHttpMetadata().getHttpStatusCode());
        } catch (final ResourceNotFoundException e) {
            System.out.println("Database " + DATABASE_NAME + " doesn't exist = " +
e);
            throw e;
        } catch (final Exception e) {
            System.out.println("Could not delete Database " + DATABASE_NAME + " = "
+ e);
            throw e;
        }
    }
```

Java v2

```
    public void deleteDatabase() {
        System.out.println("Deleting database");
        final DeleteDatabaseRequest deleteDatabaseRequest = new
DeleteDatabaseRequest();
        deleteDatabaseRequest.setDatabaseName(DATABASE_NAME);
        try {
            DeleteDatabaseResult result =
                    amazonTimestreamWrite.deleteDatabase(deleteDatabaseRequest);
            System.out.println("Delete database status: " +
result.getSdkHttpMetadata().getHttpStatusCode());
        } catch (final ResourceNotFoundException e) {
            System.out.println("Database " + DATABASE_NAME + " doesn't exist = " +
e);
            throw e;
        } catch (final Exception e) {
            System.out.println("Could not delete Database " + DATABASE_NAME + " = "
+ e);
            throw e;
        }
    }
```

Go

```
deleteDatabaseInput := &timestreamwrite.DeleteDatabaseInput{
        DatabaseName:   aws.String(*databaseName),
    }

    _, err = writeSvc.DeleteDatabase(deleteDatabaseInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Database deleted:", *databaseName)
    }
```

Python

```
    def delete_database(self):
        print("Deleting Database")
        try:
            result =
 self.client.delete_database(DatabaseName=Constant.DATABASE_NAME)
            print("Delete database status [%s]" % result['ResponseMetadata']
['HTTPStatusCode'])
        except self.client.exceptions.ResourceNotFoundException:
            print("database [%s] doesn't exist" % Constant.DATABASE_NAME)
        except Exception as err:
            print("Delete database failed:", err)
```

Node.js

Il seguente frammento viene utilizzato AWS SDK per JavaScript la v3. Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Vedi anche Class and. DeleteDatabaseCommand DeleteDatabase

```
import { TimestreamWriteClient, DeleteDatabaseCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode"
```

```
};

const command = new DeleteDatabaseCommand(params);

try {
    const data = await writeClient.send(command);
    console.log("Deleted database");
} catch (error) {
    if (error.code === 'ResourceNotFoundException') {
        console.log(`Database ${params.DatabaseName} doesn't exists.`);
    } else {
        console.log("Delete database failed.", error);
        throw error;
    }
}
```

Il seguente frammento utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function deleteDatabase() {
    console.log("Deleting Database");
    const params = {
        DatabaseName: constants.DATABASE_NAME
    };

    const promise = writeClient.deleteDatabase(params).promise();

    await promise.then(
        function (data) {
            console.log("Deleted database");
         },
        function(err) {
            if (err.code === 'ResourceNotFoundException') {
                console.log(`Database ${params.DatabaseName} doesn't exists.`);
            } else {
                console.log("Delete database failed.", err);
                throw err;
            }
        }
    );
}
```

.NET

```
        public async Task DeleteDatabase()
        {
            Console.WriteLine("Deleting database");
            try
            {
                var deleteDatabaseRequest = new DeleteDatabaseRequest
                {
                    DatabaseName = Constants.DATABASE_NAME
                };
                DeleteDatabaseResponse response = await
writeClient.DeleteDatabaseAsync(deleteDatabaseRequest);
                Console.WriteLine($"Database {Constants.DATABASE_NAME} delete
request status:{response.HttpStatusCode}");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine($"Database {Constants.DATABASE_NAME} does not
exists");
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception while deleting database:" +
e.ToString());
            }
        }
```

Elenca database

È possibile utilizzare i seguenti frammenti di codice per elencare i database.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
   public void listDatabases() {
       System.out.println("Listing databases");
       ListDatabasesRequest request = new ListDatabasesRequest();
       ListDatabasesResult result = amazonTimestreamWrite.listDatabases(request);
       final List<Database> databases = result.getDatabases();
       printDatabases(databases);

       String nextToken = result.getNextToken();
       while (nextToken != null && !nextToken.isEmpty()) {
           request.setNextToken(nextToken);
           ListDatabasesResult nextResult =
amazonTimestreamWrite.listDatabases(request);
           final List<Database> nextDatabases = nextResult.getDatabases();
           printDatabases(nextDatabases);
           nextToken = nextResult.getNextToken();
       }
   }

   private void printDatabases(List<Database> databases) {
       for (Database db : databases) {
           System.out.println(db.getDatabaseName());
       }
   }
```

Java v2

```
   public void listDatabases() {
       System.out.println("Listing databases");
       ListDatabasesRequest request =
ListDatabasesRequest.builder().maxResults(2).build();
       ListDatabasesIterable listDatabasesIterable =
timestreamWriteClient.listDatabasesPaginator(request);
       for(ListDatabasesResponse listDatabasesResponse : listDatabasesIterable) {
           final List<Database> databases = listDatabasesResponse.databases();
           databases.forEach(database ->
System.out.println(database.databaseName()));
       }
   }
```

Go

```
// List databases.
    listDatabasesMaxResult := int64(15)

    listDatabasesInput := &timestreamwrite.ListDatabasesInput{
        MaxResults: &listDatabasesMaxResult,
    }

    listDatabasesOutput, err := writeSvc.ListDatabases(listDatabasesInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("List databases is successful, below is the output:")
        fmt.Println(listDatabasesOutput)
    }
```

Python

```
    def list_databases(self):
        print("Listing databases")
        try:
            result = self.client.list_databases(MaxResults=5)
            self._print_databases(result['Databases'])
            next_token = result.get('NextToken', None)
            while next_token:
                result = self.client.list_databases(NextToken=next_token,
 MaxResults=5)
                self._print_databases(result['Databases'])
                next_token = result.get('NextToken', None)
        except Exception as err:
            print("List databases failed:", err)
```

Node.js

Il seguente frammento viene utilizzato AWS SDK per JavaScript la v3. Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Vedi anche Class and. ListDatabasesCommand ListDatabases

```
import { TimestreamWriteClient, ListDatabasesCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    MaxResults: 15
};

const command = new ListDatabasesCommand(params);

getDatabasesList(null);

async function getDatabasesList(nextToken) {
    if (nextToken) {
        params.NextToken = nextToken;
    }

    try {
        const data = await writeClient.send(command);

        data.Databases.forEach(function (database) {
            console.log(database.DatabaseName);
        });

        if (data.NextToken) {
            return getDatabasesList(data.NextToken);
        }
    } catch (error) {
        console.log("Error while listing databases", error);
    }
}
```

Il seguente frammento utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function listDatabases() {
    console.log("Listing databases:");
    const databases = await getDatabasesList(null);
    databases.forEach(function(database){
        console.log(database.DatabaseName);
    });
```

```
}

function getDatabasesList(nextToken, databases = []) {
    var params = {
        MaxResults: 15
    };

    if(nextToken) {
        params.NextToken = nextToken;
    }

    return writeClient.listDatabases(params).promise()
        .then(
            (data) => {
                databases.push.apply(databases, data.Databases);
                if (data.NextToken) {
                    return getDatabasesList(data.NextToken, databases);
                } else {
                    return databases;
                }
            },
            (err) => {
                console.log("Error while listing databases", err);
            });
}
```

.NET

```
        public async Task ListDatabases()
        {
            Console.WriteLine("Listing Databases");

            try
            {
                var listDatabasesRequest = new ListDatabasesRequest
                {
                    MaxResults = 5
                };
                ListDatabasesResponse response = await
 writeClient.ListDatabasesAsync(listDatabasesRequest);
                PrintDatabases(response.Databases);
                var nextToken = response.NextToken;
                while (nextToken != null)
```

```
                {
                    listDatabasesRequest.NextToken = nextToken;
                    response = await
writeClient.ListDatabasesAsync(listDatabasesRequest);
                    PrintDatabases(response.Databases);
                    nextToken = response.NextToken;
                }
            }
            catch (Exception e)
            {
                Console.WriteLine("List database failed:" + e.ToString());
            }

        }
```

Create table (Crea tabella)

Argomenti

- Memory Store scrive
- Magnetic Store scrive

Memory Store scrive

È possibile utilizzare il seguente frammento di codice per creare una tabella con le scritture magnetiche disattivate, di conseguenza è possibile scrivere dati solo nella finestra di conservazione dell'archivio di memoria.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
    public void createTable() {
        System.out.println("Creating table");
        CreateTableRequest createTableRequest = new CreateTableRequest();
```

```
        createTableRequest.setDatabaseName(DATABASE_NAME);
        createTableRequest.setTableName(TABLE_NAME);
        final RetentionProperties retentionProperties = new RetentionProperties()
                .withMemoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                .withMagneticStoreRetentionPeriodInDays(CT_TTL_DAYS);
        createTableRequest.setRetentionProperties(retentionProperties);

        try {
            amazonTimestreamWrite.createTable(createTableRequest);
            System.out.println("Table [" + TABLE_NAME + "] successfully created.");
        } catch (ConflictException e) {
            System.out.println("Table [" + TABLE_NAME + "] exists on database [" +
DATABASE_NAME + "] . Skipping database creation");
        }
    }
```

Java v2

```
    public void createTable() {
        System.out.println("Creating table");

        final RetentionProperties retentionProperties =
RetentionProperties.builder()
                .memoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                .magneticStoreRetentionPeriodInDays(CT_TTL_DAYS).build();
        final CreateTableRequest createTableRequest = CreateTableRequest.builder()

.databaseName(DATABASE_NAME).tableName(TABLE_NAME).retentionProperties(retentionProperties)

        try {
            timestreamWriteClient.createTable(createTableRequest);
            System.out.println("Table [" + TABLE_NAME + "] successfully created.");
        } catch (ConflictException e) {
            System.out.println("Table [" + TABLE_NAME + "] exists on database [" +
DATABASE_NAME + "] . Skipping database creation");
        }
    }
```

Go

```
// Create table.
    createTableInput := &timestreamwrite.CreateTableInput{
        DatabaseName: aws.String(*databaseName),
```

```
        TableName:    aws.String(*tableName),
    }
    _, err = writeSvc.CreateTable(createTableInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Create table is successful")
    }
```

Python

```
    def create_table(self):
        print("Creating table")
        retention_properties = {
            'MemoryStoreRetentionPeriodInHours': Constant.HT_TTL_HOURS,
            'MagneticStoreRetentionPeriodInDays': Constant.CT_TTL_DAYS
        }
        try:
            self.client.create_table(DatabaseName=Constant.DATABASE_NAME,
TableName=Constant.TABLE_NAME,
                                     RetentionProperties=retention_properties)
            print("Table [%s] successfully created." % Constant.TABLE_NAME)
        except self.client.exceptions.ConflictException:
            print("Table [%s] exists on database [%s]. Skipping table creation" % (
                Constant.TABLE_NAME, Constant.DATABASE_NAME))
        except Exception as err:
            print("Create table failed:", err)
```

Node.js

Il seguente frammento viene utilizzato AWS SDK per JavaScript la v3. Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Vedi anche Class and. CreateTableCommand CreateTable

```
import { TimestreamWriteClient, CreateTableCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
```

```
    DatabaseName: "testDbFromNode",
    TableName: "testTableFromNode",
    RetentionProperties: {
        MemoryStoreRetentionPeriodInHours: 24,
        MagneticStoreRetentionPeriodInDays: 365
    }
};

const command = new CreateTableCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Table ${data.Table.TableName} created successfully`);
} catch (error) {
    if (error.code === 'ConflictException') {
        console.log(`Table ${params.TableName} already exists on db
 ${params.DatabaseName}. Skipping creation.`);
    } else {
        console.log("Error creating table. ", error);
        throw error;
    }
}
```

Il seguente frammento utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function createTable() {
    console.log("Creating Table");
    const params = {
        DatabaseName: constants.DATABASE_NAME,
        TableName: constants.TABLE_NAME,
        RetentionProperties: {
            MemoryStoreRetentionPeriodInHours: constants.HT_TTL_HOURS,
            MagneticStoreRetentionPeriodInDays: constants.CT_TTL_DAYS
        }
    };

    const promise = writeClient.createTable(params).promise();

    await promise.then(
        (data) => {
            console.log(`Table ${data.Table.TableName} created successfully`);
```

```
        },
        (err) => {
            if (err.code === 'ConflictException') {
                console.log(`Table ${params.TableName} already exists on db
 ${params.DatabaseName}. Skipping creation.`);
            } else {
                console.log("Error creating table. ", err);
                throw err;
            }
        }
    );
}
```

.NET

```
        public async Task CreateTable()
        {
            Console.WriteLine("Creating Table");

            try
            {
                var createTableRequest = new CreateTableRequest
                {
                    DatabaseName = Constants.DATABASE_NAME,
                    TableName = Constants.TABLE_NAME,
                    RetentionProperties = new RetentionProperties
                    {
                        MagneticStoreRetentionPeriodInDays = Constants.CT_TTL_DAYS,
                        MemoryStoreRetentionPeriodInHours = Constants.HT_TTL_HOURS
                    }
                };
                CreateTableResponse response = await
 writeClient.CreateTableAsync(createTableRequest);
                Console.WriteLine($"Table {Constants.TABLE_NAME} created");
            }
            catch (ConflictException)
            {
                Console.WriteLine("Table already exists.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Create table failed:" + e.ToString());
            }
```

```
        }
```

Magnetic Store scrive

È possibile utilizzare il seguente frammento di codice per creare una tabella con le scritture magnetiche abilitate. Con le scritture magnetiche è possibile scrivere dati sia nella finestra di conservazione dell'archivio di memoria che nella finestra di conservazione dell'archivio magnetico.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
   public void createTable(String databaseName, String tableName) {
       System.out.println("Creating table");
       CreateTableRequest createTableRequest = new CreateTableRequest();
       createTableRequest.setDatabaseName(databaseName);
       createTableRequest.setTableName(tableName);
       final RetentionProperties retentionProperties = new RetentionProperties()
               .withMemoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
               .withMagneticStoreRetentionPeriodInDays(CT_TTL_DAYS);
       createTableRequest.setRetentionProperties(retentionProperties);
       // Enable MagneticStoreWrite
       final MagneticStoreWriteProperties magneticStoreWriteProperties = new
MagneticStoreWriteProperties()
               .withEnableMagneticStoreWrites(true);

createTableRequest.setMagneticStoreWriteProperties(magneticStoreWriteProperties);
       try {
           amazonTimestreamWrite.createTable(createTableRequest);
           System.out.println("Table [" + tableName + "] successfully created.");
       } catch (ConflictException e) {
           System.out.println("Table [" + tableName + "] exists on database [" +
databaseName + "] . Skipping table creation");
           //We do not throw exception here, we use the existing table instead
       }
```

```
    }
```

Java v2

```
    public void createTable(String databaseName, String tableName) {
        System.out.println("Creating table");

        // Enable MagneticStoreWrite
        final MagneticStoreWriteProperties magneticStoreWriteProperties =
                MagneticStoreWriteProperties.builder()
                        .enableMagneticStoreWrites(true)
                        .build();

        CreateTableRequest createTableRequest =
                CreateTableRequest.builder()
                        .databaseName(databaseName)
                        .tableName(tableName)
                        .retentionProperties(RetentionProperties.builder()
                                .memoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                                .magneticStoreRetentionPeriodInDays(CT_TTL_DAYS)
                                .build())
                        .magneticStoreWriteProperties(magneticStoreWriteProperties)
                        .build();
        try {
            timestreamWriteClient.createTable(createTableRequest);
            System.out.println("Table [" + tableName + "] successfully created.");
        } catch (ConflictException e) {
            System.out.println("Table [" + tableName + "] exists in database [" +
 databaseName + "] . Skipping table creation");
        }
    }
```

Go

```
// Create table.
    createTableInput := &timestreamwrite.CreateTableInput{
        DatabaseName: aws.String(*databaseName),
        TableName:    aws.String(*tableName),
    // Enable MagneticStoreWrite
        MagneticStoreWriteProperties: &timestreamwrite.MagneticStoreWriteProperties{
            EnableMagneticStoreWrites: aws.Bool(true),
             },
      }
```

```
    _, err = writeSvc.CreateTable(createTableInput)
```

Python

```
    def create_table(self):
        print("Creating table")
        retention_properties = {
            'MemoryStoreRetentionPeriodInHours': Constant.HT_TTL_HOURS,
            'MagneticStoreRetentionPeriodInDays': Constant.CT_TTL_DAYS
        }
        magnetic_store_write_properties = {
            'EnableMagneticStoreWrites': True
        }
        try:
            self.client.create_table(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                                     RetentionProperties=retention_properties,

 MagneticStoreWriteProperties=magnetic_store_write_properties)
            print("Table [%s] successfully created." % Constant.TABLE_NAME)
        except self.client.exceptions.ConflictException:
            print("Table [%s] exists on database [%s]. Skipping table creation" % (
                Constant.TABLE_NAME, Constant.DATABASE_NAME))
        except Exception as err:
            print("Create table failed:", err)
```

Node.js

```
async function createTable() {
    console.log("Creating Table");

    const params = {
        DatabaseName: constants.DATABASE_NAME,
        TableName: constants.TABLE_NAME,
        RetentionProperties: {
            MemoryStoreRetentionPeriodInHours: constants.HT_TTL_HOURS,
            MagneticStoreRetentionPeriodInDays: constants.CT_TTL_DAYS
        },
        MagneticStoreWriteProperties: {
            EnableMagneticStoreWrites: true
        }
    };
```

```
    const promise = writeClient.createTable(params).promise();

    await promise.then(
        (data) => {
            console.log(`Table ${data.Table.TableName} created successfully`);
        },
        (err) => {
            if (err.code === 'ConflictException') {
                console.log(`Table ${params.TableName} already exists on db
 ${params.DatabaseName}. Skipping creation.`);
            } else {
                console.log("Error creating table. ", err);
                throw err;
            }
        }
    );
}
```

.NET

```
        public async Task CreateTable()
        {
            Console.WriteLine("Creating Table");

            try
            {
                var createTableRequest = new CreateTableRequest
                {
                    DatabaseName = Constants.DATABASE_NAME,
                    TableName = Constants.TABLE_NAME,
                    RetentionProperties = new RetentionProperties
                    {
                        MagneticStoreRetentionPeriodInDays = Constants.CT_TTL_DAYS,
                        MemoryStoreRetentionPeriodInHours = Constants.HT_TTL_HOURS
                    },
                    // Enable MagneticStoreWrite
                    MagneticStoreWriteProperties = new MagneticStoreWriteProperties
                    {
                        EnableMagneticStoreWrites = true,
                    }
                };
                CreateTableResponse response = await
 writeClient.CreateTableAsync(createTableRequest);
```

```
                Console.WriteLine($"Table {Constants.TABLE_NAME} created");
            }
            catch (ConflictException)
            {
                Console.WriteLine("Table already exists.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Create table failed:" + e.ToString());
            }

        }
```

Descrivi tabella

È possibile utilizzare i seguenti frammenti di codice per ottenere informazioni sugli attributi della tabella.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
   public void describeTable() {
       System.out.println("Describing table");
       final DescribeTableRequest describeTableRequest = new
DescribeTableRequest();
       describeTableRequest.setDatabaseName(DATABASE_NAME);
       describeTableRequest.setTableName(TABLE_NAME);
       try {
           DescribeTableResult result =
amazonTimestreamWrite.describeTable(describeTableRequest);
           String tableId = result.getTable().getArn();
           System.out.println("Table " + TABLE_NAME + " has id " + tableId);
       } catch (final Exception e) {
           System.out.println("Table " + TABLE_NAME + " doesn't exist = " + e);
           throw e;
```

```
        }
    }
```

Java v2

```
    public void describeTable() {
        System.out.println("Describing table");
        final DescribeTableRequest describeTableRequest =
 DescribeTableRequest.builder()
                .databaseName(DATABASE_NAME).tableName(TABLE_NAME).build();
        try {
            DescribeTableResponse response =
 timestreamWriteClient.describeTable(describeTableRequest);
            String tableId = response.table().arn();
            System.out.println("Table " + TABLE_NAME + " has id " + tableId);
        } catch (final Exception e) {
            System.out.println("Table " + TABLE_NAME + " doesn't exist = " + e);
            throw e;
        }
    }
```

Go

```
// Describe table.
    describeTableInput := &timestreamwrite.DescribeTableInput{
        DatabaseName: aws.String(*databaseName),
        TableName:    aws.String(*tableName),
    }
    describeTableOutput, err := writeSvc.DescribeTable(describeTableInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Describe table is successful, below is the output:")
        fmt.Println(describeTableOutput)
    }
```

Python

```
    def describe_table(self):
        print("Describing table")
        try:
```

```
            result = self.client.describe_table(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME)
            print("Table [%s] has id [%s]" % (Constant.TABLE_NAME, result['Table']
['Arn']))
        except self.client.exceptions.ResourceNotFoundException:
            print("Table doesn't exist")
        except Exception as err:
            print("Describe table failed:", err)
```

Node.js

Il seguente frammento viene utilizzato AWS SDK per JavaScript la v3. Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Vedi anche Class and. DescribeTableCommand DescribeTable

```
import { TimestreamWriteClient, DescribeTableCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode",
    TableName: "testTableFromNode"
};

const command = new DescribeTableCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Table ${data.Table.TableName} has id ${data.Table.Arn}`);
} catch (error) {
    if (error.code === 'ResourceNotFoundException') {
        console.log("Table or Database doesn't exist.");
    } else {
        console.log("Describe table failed.", error);
        throw error;
    }
}
```

Il seguente frammento utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function describeTable() {
    console.log("Describing Table");
    const params = {
        DatabaseName: constants.DATABASE_NAME,
        TableName: constants.TABLE_NAME
    };

    const promise = writeClient.describeTable(params).promise();

    await promise.then(
        (data) => {
            console.log(`Table ${data.Table.TableName} has id ${data.Table.Arn}`);
        },
        (err) => {
            if (err.code === 'ResourceNotFoundException') {
                console.log("Table or Database doesn't exists.");
            } else {
                console.log("Describe table failed.", err);
                throw err;
            }
        }
    );
}
```

.NET

```
        public async Task DescribeTable()
        {
            Console.WriteLine("Describing Table");

            try
            {
                var describeTableRequest = new DescribeTableRequest
                {
                    DatabaseName = Constants.DATABASE_NAME,
                    TableName = Constants.TABLE_NAME
                };
                DescribeTableResponse response = await
 writeClient.DescribeTableAsync(describeTableRequest);
                Console.WriteLine($"Table {Constants.TABLE_NAME} has id:
{response.Table.Arn}");
            }
            catch (ResourceNotFoundException)
```

```
            {
                Console.WriteLine("Table does not exist.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Describe table failed:" + e.ToString());
            }

        }
```

Update Table (Aggiorna tabella)

È possibile utilizzare i seguenti frammenti di codice per aggiornare una tabella.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
    public void updateTable() {
        System.out.println("Updating table");
        UpdateTableRequest updateTableRequest = new UpdateTableRequest();
        updateTableRequest.setDatabaseName(DATABASE_NAME);
        updateTableRequest.setTableName(TABLE_NAME);

        final RetentionProperties retentionProperties = new RetentionProperties()
                .withMemoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                .withMagneticStoreRetentionPeriodInDays(CT_TTL_DAYS);

        updateTableRequest.setRetentionProperties(retentionProperties);

        amazonTimestreamWrite.updateTable(updateTableRequest);
        System.out.println("Table updated");
    }
```

Java v2

```
    public void updateTable() {
        System.out.println("Updating table");

        final RetentionProperties retentionProperties =
 RetentionProperties.builder()
                .memoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                .magneticStoreRetentionPeriodInDays(CT_TTL_DAYS).build();
        final UpdateTableRequest updateTableRequest = UpdateTableRequest.builder()

 .databaseName(DATABASE_NAME).tableName(TABLE_NAME).retentionProperties(retentionProperties)

        timestreamWriteClient.updateTable(updateTableRequest);
        System.out.println("Table updated");
    }
```

Go

```
// Update table.
    magneticStoreRetentionPeriodInDays := int64(7 * 365)
    memoryStoreRetentionPeriodInHours := int64(24)

    updateTableInput := &timestreamwrite.UpdateTableInput{
        DatabaseName: aws.String(*databaseName),
        TableName:    aws.String(*tableName),
        RetentionProperties: &timestreamwrite.RetentionProperties{
            MagneticStoreRetentionPeriodInDays: &magneticStoreRetentionPeriodInDays,
            MemoryStoreRetentionPeriodInHours:  &memoryStoreRetentionPeriodInHours,
        },
    }
    updateTableOutput, err := writeSvc.UpdateTable(updateTableInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    } else {
        fmt.Println("Update table is successful, below is the output:")
        fmt.Println(updateTableOutput)
    }
```

Python

```
    def update_table(self):
        print("Updating table")
        retention_properties = {
            'MemoryStoreRetentionPeriodInHours': Constant.HT_TTL_HOURS,
            'MagneticStoreRetentionPeriodInDays': Constant.CT_TTL_DAYS
        }
        try:
            self.client.update_table(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                                     RetentionProperties=retention_properties)
            print("Table updated.")
        except Exception as err:
            print("Update table failed:", err)
```

Node.js

Il seguente frammento viene utilizzato AWS SDK per JavaScript la v3. Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Vedi anche Class and. UpdateTableCommand UpdateTable

```
import { TimestreamWriteClient, UpdateTableCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode",
    TableName: "testTableFromNode",
    RetentionProperties: {
        MemoryStoreRetentionPeriodInHours: 24,
        MagneticStoreRetentionPeriodInDays: 180
    }
};

const command = new UpdateTableCommand(params);

try {
    const data = await writeClient.send(command);
    console.log("Table updated")
} catch (error) {
```

```
    console.log("Error updating table. ", error);
}
```

Il seguente frammento utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function updateTable() {
    console.log("Updating Table");
    const params = {
        DatabaseName: constants.DATABASE_NAME,
        TableName: constants.TABLE_NAME,
        RetentionProperties: {
            MemoryStoreRetentionPeriodInHours: constants.HT_TTL_HOURS,
            MagneticStoreRetentionPeriodInDays: constants.CT_TTL_DAYS
        }
    };

    const promise = writeClient.updateTable(params).promise();

    await promise.then(
        (data) => {
            console.log("Table updated")
        },
        (err) => {
            console.log("Error updating table. ", err);
            throw err;
        }
    );
}
```

.NET

```
        public async Task UpdateTable()
        {
            Console.WriteLine("Updating Table");

            try
            {
                var updateTableRequest = new UpdateTableRequest
                {
                    DatabaseName = Constants.DATABASE_NAME,
                    TableName = Constants.TABLE_NAME,
```

```
                    RetentionProperties = new RetentionProperties
                    {
                        MagneticStoreRetentionPeriodInDays = Constants.CT_TTL_DAYS,
                        MemoryStoreRetentionPeriodInHours = Constants.HT_TTL_HOURS
                    }
                };
                UpdateTableResponse response = await
writeClient.UpdateTableAsync(updateTableRequest);
                Console.WriteLine($"Table {Constants.TABLE_NAME} updated");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine("Table does not exist.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Update table failed:" + e.ToString());
            }

        }
```

Delete Table (Elimina tabella)

È possibile utilizzare i seguenti frammenti di codice per eliminare una tabella.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub
> Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, vedere.
> Applicazione di esempio

Java

```
    public void deleteTable() {
        System.out.println("Deleting table");
        final DeleteTableRequest deleteTableRequest = new DeleteTableRequest();
        deleteTableRequest.setDatabaseName(DATABASE_NAME);
        deleteTableRequest.setTableName(TABLE_NAME);
        try {
            DeleteTableResult result =
```

```
                    amazonTimestreamWrite.deleteTable(deleteTableRequest);
            System.out.println("Delete table status: " +
 result.getSdkHttpMetadata().getHttpStatusCode());
        } catch (final ResourceNotFoundException e) {
            System.out.println("Table " + TABLE_NAME + " doesn't exist = " + e);
            throw e;
        } catch (final Exception e) {
            System.out.println("Could not delete table " + TABLE_NAME + " = " + e);
            throw e;
        }
    }
```

Java v2

```
    public void deleteTable() {
        System.out.println("Deleting table");
        final DeleteTableRequest deleteTableRequest = DeleteTableRequest.builder()
                .databaseName(DATABASE_NAME).tableName(TABLE_NAME).build();
        try {
            DeleteTableResponse response =
                    timestreamWriteClient.deleteTable(deleteTableRequest);
            System.out.println("Delete table status: " +
 response.sdkHttpResponse().statusCode());
        } catch (final ResourceNotFoundException e) {
            System.out.println("Table " + TABLE_NAME + " doesn't exist = " + e);
            throw e;
        } catch (final Exception e) {
            System.out.println("Could not delete table " + TABLE_NAME + " = " + e);
            throw e;
        }
    }
```

Go

```
deleteTableInput := &timestreamwrite.DeleteTableInput{
        DatabaseName:   aws.String(*databaseName),
        TableName:    aws.String(*tableName),
    }
    _, err = writeSvc.DeleteTable(deleteTableInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
```

```
    } else {
        fmt.Println("Table deleted", *tableName)
    }
```

Python

```
    def delete_table(self):
        print("Deleting Table")
        try:
            result = self.client.delete_table(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME)
            print("Delete table status [%s]" % result['ResponseMetadata']
['HTTPStatusCode'])
        except self.client.exceptions.ResourceNotFoundException:
            print("Table [%s] doesn't exist" % Constant.TABLE_NAME)
        except Exception as err:
            print("Delete table failed:", err)
```

Node.js

Il seguente frammento viene utilizzato AWS SDK per JavaScript la v3. Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Vedi anche Class and. DeleteTableCommand DeleteTable

```
import { TimestreamWriteClient, DeleteTableCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode",
    TableName: "testTableFromNode"
};

const command = new DeleteTableCommand(params);

try {
    const data = await writeClient.send(command);
    console.log("Deleted table");
} catch (error) {
    if (error.code === 'ResourceNotFoundException') {
```

```
        console.log(`Table ${params.TableName} or Database ${params.DatabaseName}
 doesn't exist.`);
    } else {
        console.log("Delete table failed.", error);
        throw error;
    }
}
```

Il seguente frammento utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function deleteTable() {
    console.log("Deleting Table");
    const params = {
        DatabaseName: constants.DATABASE_NAME,
        TableName: constants.TABLE_NAME
    };

    const promise = writeClient.deleteTable(params).promise();

    await promise.then(
        function (data) {
            console.log("Deleted table");
        },
        function(err) {
            if (err.code === 'ResourceNotFoundException') {
                console.log(`Table ${params.TableName} or Database
 ${params.DatabaseName} doesn't exists.`);
            } else {
                console.log("Delete table failed.", err);
                throw err;
            }
        }
    );
}
```

.NET

```
        public async Task DeleteTable()
        {
            Console.WriteLine("Deleting table");
            try
```

```
            {
                var deleteTableRequest = new DeleteTableRequest
                {
                    DatabaseName = Constants.DATABASE_NAME,
                    TableName = Constants.TABLE_NAME
                };
                DeleteTableResponse response = await
writeClient.DeleteTableAsync(deleteTableRequest);
                Console.WriteLine($"Table {Constants.TABLE_NAME} delete request
status: {response.HttpStatusCode}");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine($"Table {Constants.TABLE_NAME} does not exists");
            }
            catch (Exception e)
            {
                Console.WriteLine("Exception while deleting table:" + e.ToString());
            }
        }
```

Elencare tabelle

È possibile utilizzare i seguenti frammenti di codice per elencare le tabelle.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
    public void listTables() {
        System.out.println("Listing tables");
        ListTablesRequest request = new ListTablesRequest();
        request.setDatabaseName(DATABASE_NAME);
        ListTablesResult result = amazonTimestreamWrite.listTables(request);
        printTables(result.getTables());
```

```
        String nextToken = result.getNextToken();
        while (nextToken != null && !nextToken.isEmpty()) {
            request.setNextToken(nextToken);
            ListTablesResult nextResult = amazonTimestreamWrite.listTables(request);

            printTables(nextResult.getTables());
            nextToken = nextResult.getNextToken();
        }
    }

     private void printTables(List<Table> tables) {
        for (Table table : tables) {
            System.out.println(table.getTableName());
        }
    }
```

Java v2

```
    public void listTables() {
        System.out.println("Listing tables");
        ListTablesRequest request =
 ListTablesRequest.builder().databaseName(DATABASE_NAME).maxResults(2).build();
        ListTablesIterable listTablesIterable =
 timestreamWriteClient.listTablesPaginator(request);
        for(ListTablesResponse listTablesResponse : listTablesIterable) {
            final List<Table> tables = listTablesResponse.tables();
            tables.forEach(table -> System.out.println(table.tableName()));
        }
    }
```

Go

```
listTablesMaxResult := int64(15)

    listTablesInput := &timestreamwrite.ListTablesInput{
        DatabaseName: aws.String(*databaseName),
        MaxResults:   &listTablesMaxResult,
    }
    listTablesOutput, err := writeSvc.ListTables(listTablesInput)

    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
```

```
    } else {
        fmt.Println("List tables is successful, below is the output:")
        fmt.Println(listTablesOutput)
    }
```

Python

```
    def list_tables(self):
        print("Listing tables")
        try:
            result = self.client.list_tables(DatabaseName=Constant.DATABASE_NAME,
 MaxResults=5)
            self.__print_tables(result['Tables'])
            next_token = result.get('NextToken', None)
            while next_token:
                result =
 self.client.list_tables(DatabaseName=Constant.DATABASE_NAME,
                                                    NextToken=next_token, MaxResults=5)
                self.__print_tables(result['Tables'])
                next_token = result.get('NextToken', None)
        except Exception as err:
            print("List tables failed:", err)
```

Node.js

Il seguente frammento viene utilizzato AWS SDK per JavaScript la v3. Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Vedi anche Class and. ListTablesCommand ListTables

```
import { TimestreamWriteClient, ListTablesCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-east-1" });

const params = {
    DatabaseName: "testDbFromNode",
    MaxResults: 15
};

const command = new ListTablesCommand(params);

getTablesList(null);
```

```
async function getTablesList(nextToken) {
    if (nextToken) {
        params.NextToken = nextToken;
    }

    try {
        const data = await writeClient.send(command);

        data.Tables.forEach(function (table) {
            console.log(table.TableName);
        });

        if (data.NextToken) {
            return getTablesList(data.NextToken);
        }
    } catch (error) {
        console.log("Error while listing tables", error);
    }
}
```

Il seguente frammento utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function listTables() {
    console.log("Listing tables:");
    const tables = await getTablesList(null);
    tables.forEach(function(table){
        console.log(table.TableName);
    });
}

function getTablesList(nextToken, tables = []) {
    var params = {
        DatabaseName: constants.DATABASE_NAME,
        MaxResults: 15
    };

    if(nextToken) {
        params.NextToken = nextToken;
    }
```

```
    return writeClient.listTables(params).promise()
        .then(
            (data) => {
                tables.push.apply(tables, data.Tables);
                if (data.NextToken) {
                    return getTablesList(data.NextToken, tables);
                } else {
                    return tables;
                }
            },
            (err) => {
                console.log("Error while listing databases", err);
            });
}
```

.NET

```
        public async Task ListTables()
        {
            Console.WriteLine("Listing Tables");

            try
            {
                var listTablesRequest = new ListTablesRequest
                {
                    MaxResults = 5,
                    DatabaseName = Constants.DATABASE_NAME
                };
                ListTablesResponse response = await
 writeClient.ListTablesAsync(listTablesRequest);
                PrintTables(response.Tables);
                string nextToken = response.NextToken;
                while (nextToken != null)
                {
                    listTablesRequest.NextToken = nextToken;
                    response = await writeClient.ListTablesAsync(listTablesRequest);
                    PrintTables(response.Tables);
                    nextToken = response.NextToken;
                }
            }
            catch (Exception e)
            {
                Console.WriteLine("List table failed:" + e.ToString());
```

```
            }

        }

        private void PrintTables(List<Table> tables)
        {
            foreach (Table table in tables)
                Console.WriteLine($"Table: {table.TableName}");
        }
```

Scrivere dati (inserti e sconvolgimenti)

Argomenti

- Scrittura di batch di record
- Scrittura di batch di record con attributi comuni
- Sconvolgimento dei record
- Esempio di attributo multimisura
- Gestione degli errori di scrittura

Scrittura di batch di record

Puoi utilizzare i seguenti frammenti di codice per scrivere dati in una tabella Amazon Timestream. La scrittura di dati in batch aiuta a ottimizzare il costo delle scritture. Per ulteriori informazioni, consulta Calcolo del numero di scritture.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
public void writeRecords() {
  System.out.println("Writing records");
  // Specify repeated values for all records
  List<Record> records = new ArrayList<>();
```

```
    final long time = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = new Dimension().withName("region").withValue("us-
east-1");
    final Dimension az = new Dimension().withName("az").withValue("az1");
    final Dimension hostname = new
 Dimension().withName("hostname").withValue("host1");

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record cpuUtilization = new Record()
        .withDimensions(dimensions)
        .withMeasureName("cpu_utilization")
        .withMeasureValue("13.5")
        .withMeasureValueType(MeasureValueType.DOUBLE)
        .withTime(String.valueOf(time));
    Record memoryUtilization = new Record()
        .withDimensions(dimensions)
        .withMeasureName("memory_utilization")
        .withMeasureValue("40")
        .withMeasureValueType(MeasureValueType.DOUBLE)
        .withTime(String.valueOf(time));

    records.add(cpuUtilization);
    records.add(memoryUtilization);

    WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withRecords(records);

    try {
      WriteRecordsResult writeRecordsResult =
 amazonTimestreamWrite.writeRecords(writeRecordsRequest);
      System.out.println("WriteRecords Status: " +
 writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      System.out.println("RejectedRecords: " + e);
      for (RejectedRecord rejectedRecord : e.getRejectedRecords()) {
        System.out.println("Rejected Index " + rejectedRecord.getRecordIndex() + ":
 "
```

```
            + rejectedRecord.getReason());
      }
      System.out.println("Other records were written successfully. ");
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Java v2

```
  public void writeRecords() {
    System.out.println("Writing records");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = Dimension.builder().name("region").value("us-
east-1").build();
    final Dimension az = Dimension.builder().name("az").value("az1").build();
    final Dimension hostname =
 Dimension.builder().name("hostname").value("host1").build();

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record cpuUtilization = Record.builder()
        .dimensions(dimensions)
        .measureValueType(MeasureValueType.DOUBLE)
        .measureName("cpu_utilization")
        .measureValue("13.5")
        .time(String.valueOf(time)).build();

    Record memoryUtilization = Record.builder()
        .dimensions(dimensions)
        .measureValueType(MeasureValueType.DOUBLE)
        .measureName("memory_utilization")
        .measureValue("40")
        .time(String.valueOf(time)).build();

    records.add(cpuUtilization);
    records.add(memoryUtilization);
```

```
   WriteRecordsRequest writeRecordsRequest = WriteRecordsRequest.builder()
       .databaseName(DATABASE_NAME).tableName(TABLE_NAME).records(records).build();

   try {
     WriteRecordsResponse writeRecordsResponse =
timestreamWriteClient.writeRecords(writeRecordsRequest);
     System.out.println("WriteRecords Status: " +
writeRecordsResponse.sdkHttpResponse().statusCode());
   } catch (RejectedRecordsException e) {
     System.out.println("RejectedRecords: " + e);
     for (RejectedRecord rejectedRecord : e.rejectedRecords()) {
       System.out.println("Rejected Index " + rejectedRecord.recordIndex() + ": "
           + rejectedRecord.reason());
     }
     System.out.println("Other records were written successfully. ");
   } catch (Exception e) {
     System.out.println("Error: " + e);
   }
 }
```

Go

```
now := time.Now()
currentTimeInSeconds := now.Unix()
writeRecordsInput := &timestreamwrite.WriteRecordsInput{
  DatabaseName: aws.String(*databaseName),
  TableName:  aws.String(*tableName),
  Records: []*timestreamwrite.Record{
    &timestreamwrite.Record{
      Dimensions: []*timestreamwrite.Dimension{
        &timestreamwrite.Dimension{
          Name:  aws.String("region"),
          Value: aws.String("us-east-1"),
        },
        &timestreamwrite.Dimension{
          Name:  aws.String("az"),
          Value: aws.String("az1"),
        },
        &timestreamwrite.Dimension{
          Name:  aws.String("hostname"),
          Value: aws.String("host1"),
        },
```

```
      },
      MeasureName:    aws.String("cpu_utilization"),
      MeasureValue:   aws.String("13.5"),
      MeasureValueType: aws.String("DOUBLE"),
      Time:       aws.String(strconv.FormatInt(currentTimeInSeconds, 10)),
      TimeUnit:  aws.String("SECONDS"),
    },
    &timestreamwrite.Record{
      Dimensions: []*timestreamwrite.Dimension{
        &timestreamwrite.Dimension{
          Name:  aws.String("region"),
          Value: aws.String("us-east-1"),
        },
        &timestreamwrite.Dimension{
          Name:  aws.String("az"),
          Value: aws.String("az1"),
        },
        &timestreamwrite.Dimension{
          Name:  aws.String("hostname"),
          Value: aws.String("host1"),
        },
      },
      MeasureName:    aws.String("memory_utilization"),
      MeasureValue:   aws.String("40"),
      MeasureValueType: aws.String("DOUBLE"),
      Time:       aws.String(strconv.FormatInt(currentTimeInSeconds, 10)),
      TimeUnit:  aws.String("SECONDS"),
    },
  },
}

_, err = writeSvc.WriteRecords(writeRecordsInput)

if err != nil {
  fmt.Println("Error:")
  fmt.Println(err)
} else {
  fmt.Println("Write records is successful")
}
```

Python

```
  def write_records(self):
```

```
    print("Writing records")
    current_time = self._current_milli_time()

    dimensions = [
      {'Name': 'region', 'Value': 'us-east-1'},
      {'Name': 'az', 'Value': 'az1'},
      {'Name': 'hostname', 'Value': 'host1'}
    ]

    cpu_utilization = {
      'Dimensions': dimensions,
      'MeasureName': 'cpu_utilization',
      'MeasureValue': '13.5',
      'MeasureValueType': 'DOUBLE',
      'Time': current_time
    }

    memory_utilization = {
      'Dimensions': dimensions,
      'MeasureName': 'memory_utilization',
      'MeasureValue': '40',
      'MeasureValueType': 'DOUBLE',
      'Time': current_time
    }

    records = [cpu_utilization, memory_utilization]

    try:
      result = self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                         Records=records, CommonAttributes={})
      print("WriteRecords Status: [%s]" % result['ResponseMetadata']
['HTTPStatusCode'])
    except self.client.exceptions.RejectedRecordsException as err:
      self._print_rejected_records_exceptions(err)
    except Exception as err:
      print("Error:", err)

  @staticmethod
  def _print_rejected_records_exceptions(err):
    print("RejectedRecords: ", err)
    for rr in err.response["RejectedRecords"]:
      print("Rejected Index " + str(rr["RecordIndex"]) + ": " + rr["Reason"])
      if "ExistingVersion" in rr:
```

```
        print("Rejected record existing version: ", rr["ExistingVersion"])

  @staticmethod
  def _current_milli_time():
    return str(int(round(time.time() * 1000)))
```

Node.js

Il frammento seguente utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function writeRecords() {
  console.log("Writing records");
  const currentTime = Date.now().toString(); // Unix time in milliseconds

  const dimensions = [
    {'Name': 'region', 'Value': 'us-east-1'},
    {'Name': 'az', 'Value': 'az1'},
    {'Name': 'hostname', 'Value': 'host1'}
  ];

  const cpuUtilization = {
    'Dimensions': dimensions,
    'MeasureName': 'cpu_utilization',
    'MeasureValue': '13.5',
    'MeasureValueType': 'DOUBLE',
    'Time': currentTime.toString()
  };

  const memoryUtilization = {
    'Dimensions': dimensions,
    'MeasureName': 'memory_utilization',
    'MeasureValue': '40',
    'MeasureValueType': 'DOUBLE',
    'Time': currentTime.toString()
  };

  const records = [cpuUtilization, memoryUtilization];

  const params = {
    DatabaseName: constants.DATABASE_NAME,
    TableName: constants.TABLE_NAME,
```

```
    Records: records
  };

  const request = writeClient.writeRecords(params);

  await request.promise().then(
    (data) => {
      console.log("Write records successful");
    },
    (err) => {
      console.log("Error writing records:", err);
      if (err.code === 'RejectedRecordsException') {
        const responsePayload =
 JSON.parse(request.response.httpResponse.body.toString());
        console.log("RejectedRecords: ", responsePayload.RejectedRecords);
        console.log("Other records were written successfully. ");
      }
    }
  );
}
```

.NET

```
   public async Task WriteRecords()
   {
     Console.WriteLine("Writing records");

     DateTimeOffset now = DateTimeOffset.UtcNow;
     string currentTimeString = (now.ToUnixTimeMilliseconds()).ToString();

     List<Dimension> dimensions = new List<Dimension>{
       new Dimension { Name = "region", Value = "us-east-1" },
       new Dimension { Name = "az", Value = "az1" },
       new Dimension { Name = "hostname", Value = "host1" }
     };

     var cpuUtilization = new Record
     {
       Dimensions = dimensions,
       MeasureName = "cpu_utilization",
       MeasureValue = "13.6",
       MeasureValueType = MeasureValueType.DOUBLE,
       Time = currentTimeString
```

```
    };

    var memoryUtilization = new Record
    {
      Dimensions = dimensions,
      MeasureName = "memory_utilization",
      MeasureValue = "40",
      MeasureValueType = MeasureValueType.DOUBLE,
      Time = currentTimeString
    };


    List<Record> records = new List<Record> {
      cpuUtilization,
      memoryUtilization
    };

    try
    {
      var writeRecordsRequest = new WriteRecordsRequest
      {
        DatabaseName = Constants.DATABASE_NAME,
        TableName = Constants.TABLE_NAME,
        Records = records
      };
      WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
      Console.WriteLine($"Write records status code:
{response.HttpStatusCode.ToString()}");
    }
    catch (RejectedRecordsException e) {
      Console.WriteLine("RejectedRecordsException:" + e.ToString());
      foreach (RejectedRecord rr in e.RejectedRecords) {
        Console.WriteLine("RecordIndex " + rr.RecordIndex + " : " + rr.Reason);
      }
      Console.WriteLine("Other records were written successfully. ");
    }
    catch (Exception e)
    {
      Console.WriteLine("Write records failure:" + e.ToString());
    }
  }
```

Scrittura di batch di record con attributi comuni

Se i dati delle serie temporali hanno misure e/o dimensioni comuni a molti punti dati, puoi anche utilizzare la seguente versione ottimizzata di writeRecords API per inserire dati in Timestream for. LiveAnalytics L'utilizzo di attributi comuni con il batching può ottimizzare ulteriormente il costo delle scritture, come descritto in. Calcolo del numero di scritture

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
  public void writeRecordsWithCommonAttributes() {
    System.out.println("Writing records with extracting common attributes");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = new Dimension().withName("region").withValue("us-
east-1");
    final Dimension az = new Dimension().withName("az").withValue("az1");
    final Dimension hostname = new
 Dimension().withName("hostname").withValue("host1");

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = new Record()
        .withDimensions(dimensions)
        .withMeasureValueType(MeasureValueType.DOUBLE)
        .withTime(String.valueOf(time));

    Record cpuUtilization = new Record()
        .withMeasureName("cpu_utilization")
        .withMeasureValue("13.5");
    Record memoryUtilization = new Record()
```

```
        .withMeasureName("memory_utilization")
        .withMeasureValue("40");

    records.add(cpuUtilization);
    records.add(memoryUtilization);

    WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withCommonAttributes(commonAttributes);
    writeRecordsRequest.setRecords(records);

    try {
      WriteRecordsResult writeRecordsResult =
 amazonTimestreamWrite.writeRecords(writeRecordsRequest);
      System.out.println("writeRecordsWithCommonAttributes Status: " +
 writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      System.out.println("RejectedRecords: " + e);
      for (RejectedRecord rejectedRecord : e.getRejectedRecords()) {
        System.out.println("Rejected Index " + rejectedRecord.getRecordIndex() + ":
 "
            + rejectedRecord.getReason());
      }
      System.out.println("Other records were written successfully. ");
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Java v2

```
  public void writeRecordsWithCommonAttributes() {
    System.out.println("Writing records with extracting common attributes");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = Dimension.builder().name("region").value("us-
east-1").build();
    final Dimension az = Dimension.builder().name("az").value("az1").build();
```

```
   final Dimension hostname =
Dimension.builder().name("hostname").value("host1").build();

   dimensions.add(region);
   dimensions.add(az);
   dimensions.add(hostname);

   Record commonAttributes = Record.builder()
       .dimensions(dimensions)
       .measureValueType(MeasureValueType.DOUBLE)
       .time(String.valueOf(time)).build();

   Record cpuUtilization = Record.builder()
       .measureName("cpu_utilization")
       .measureValue("13.5").build();
   Record memoryUtilization = Record.builder()
       .measureName("memory_utilization")
       .measureValue("40").build();

   records.add(cpuUtilization);
   records.add(memoryUtilization);

   WriteRecordsRequest writeRecordsRequest = WriteRecordsRequest.builder()
       .databaseName(DATABASE_NAME)
       .tableName(TABLE_NAME)
       .commonAttributes(commonAttributes)
       .records(records).build();

   try {
     WriteRecordsResponse writeRecordsResponse =
timestreamWriteClient.writeRecords(writeRecordsRequest);
     System.out.println("writeRecordsWithCommonAttributes Status: " +
writeRecordsResponse.sdkHttpResponse().statusCode());
   } catch (RejectedRecordsException e) {
     System.out.println("RejectedRecords: " + e);
     for (RejectedRecord rejectedRecord : e.rejectedRecords()) {
       System.out.println("Rejected Index " + rejectedRecord.recordIndex() + ": "
           + rejectedRecord.reason());
     }
     System.out.println("Other records were written successfully. ");
   } catch (Exception e) {
     System.out.println("Error: " + e);
   }
```

```
  }
```

Go

```
now = time.Now()
currentTimeInSeconds = now.Unix()
writeRecordsCommonAttributesInput := &timestreamwrite.WriteRecordsInput{
 DatabaseName: aws.String(*databaseName),
 TableName:  aws.String(*tableName),
 CommonAttributes: &timestreamwrite.Record{
  Dimensions: []*timestreamwrite.Dimension{
   &timestreamwrite.Dimension{
    Name:  aws.String("region"),
    Value: aws.String("us-east-1"),
   },
   &timestreamwrite.Dimension{
    Name:  aws.String("az"),
    Value: aws.String("az1"),
   },
   &timestreamwrite.Dimension{
    Name:  aws.String("hostname"),
    Value: aws.String("host1"),
   },
  },
  MeasureValueType: aws.String("DOUBLE"),
  Time:       aws.String(strconv.FormatInt(currentTimeInSeconds, 10)),
  TimeUnit:     aws.String("SECONDS"),
 },
 Records: []*timestreamwrite.Record{
  &timestreamwrite.Record{
   MeasureName:  aws.String("cpu_utilization"),
   MeasureValue: aws.String("13.5"),
  },
  &timestreamwrite.Record{
   MeasureName:  aws.String("memory_utilization"),
   MeasureValue: aws.String("40"),
  },
 },
}

_, err = writeSvc.WriteRecords(writeRecordsCommonAttributesInput)

if err != nil {
```

```
 fmt.Println("Error:")
 fmt.Println(err)
} else {
 fmt.Println("Ingest records is successful")
}
```

Python

```
  def write_records_with_common_attributes(self):
    print("Writing records extracting common attributes")
    current_time = self._current_milli_time()

    dimensions = [
      {'Name': 'region', 'Value': 'us-east-1'},
      {'Name': 'az', 'Value': 'az1'},
      {'Name': 'hostname', 'Value': 'host1'}
    ]

    common_attributes = {
      'Dimensions': dimensions,
      'MeasureValueType': 'DOUBLE',
      'Time': current_time
    }

    cpu_utilization = {
      'MeasureName': 'cpu_utilization',
      'MeasureValue': '13.5'
    }

    memory_utilization = {
      'MeasureName': 'memory_utilization',
      'MeasureValue': '40'
    }

    records = [cpu_utilization, memory_utilization]

    try:
      result = self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                         Records=records, CommonAttributes=common_attributes)
      print("WriteRecords Status: [%s]" % result['ResponseMetadata']
['HTTPStatusCode'])
    except self.client.exceptions.RejectedRecordsException as err:
```

```
      self._print_rejected_records_exceptions(err)
    except Exception as err:
      print("Error:", err)

  @staticmethod
  def _print_rejected_records_exceptions(err):
    print("RejectedRecords: ", err)
    for rr in err.response["RejectedRecords"]:
      print("Rejected Index " + str(rr["RecordIndex"]) + ": " + rr["Reason"])
      if "ExistingVersion" in rr:
        print("Rejected record existing version: ", rr["ExistingVersion"])

  @staticmethod
  def _current_milli_time():
    return str(int(round(time.time() * 1000)))
```

Node.js

Il frammento seguente utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function writeRecordsWithCommonAttributes() {
  console.log("Writing records with common attributes");
  const currentTime = Date.now().toString(); // Unix time in milliseconds

  const dimensions = [
    {'Name': 'region', 'Value': 'us-east-1'},
    {'Name': 'az', 'Value': 'az1'},
    {'Name': 'hostname', 'Value': 'host1'}
  ];

  const commonAttributes = {
    'Dimensions': dimensions,
    'MeasureValueType': 'DOUBLE',
    'Time': currentTime.toString()
  };

  const cpuUtilization = {
    'MeasureName': 'cpu_utilization',
    'MeasureValue': '13.5'
  };
```

```
  const memoryUtilization = {
    'MeasureName': 'memory_utilization',
    'MeasureValue': '40'
  };

  const records = [cpuUtilization, memoryUtilization];

  const params = {
    DatabaseName: constants.DATABASE_NAME,
    TableName: constants.TABLE_NAME,
    Records: records,
    CommonAttributes: commonAttributes
  };

  const request = writeClient.writeRecords(params);

  await request.promise().then(
    (data) => {
      console.log("Write records successful");
    },
    (err) => {
      console.log("Error writing records:", err);
      if (err.code === 'RejectedRecordsException') {
        const responsePayload =
 JSON.parse(request.response.httpResponse.body.toString());
        console.log("RejectedRecords: ", responsePayload.RejectedRecords);
        console.log("Other records were written successfully. ");
      }
    }
  );
}
```

.NET

```
  public async Task WriteRecordsWithCommonAttributes()
  {
    Console.WriteLine("Writing records with common attributes");

    DateTimeOffset now = DateTimeOffset.UtcNow;
    string currentTimeString = (now.ToUnixTimeMilliseconds()).ToString();

    List<Dimension> dimensions = new List<Dimension>{
      new Dimension { Name = "region", Value = "us-east-1" },
```

```
      new Dimension { Name = "az", Value = "az1" },
      new Dimension { Name = "hostname", Value = "host1" }
    };

    var commonAttributes = new Record
    {
      Dimensions = dimensions,
      MeasureValueType = MeasureValueType.DOUBLE,
      Time = currentTimeString
    };

    var cpuUtilization = new Record
    {
      MeasureName = "cpu_utilization",
      MeasureValue = "13.6"
    };

    var memoryUtilization = new Record
    {
      MeasureName = "memory_utilization",
      MeasureValue = "40"
    };


    List<Record> records = new List<Record>();
    records.Add(cpuUtilization);
    records.Add(memoryUtilization);

    try
    {
      var writeRecordsRequest = new WriteRecordsRequest
      {
        DatabaseName = Constants.DATABASE_NAME,
        TableName = Constants.TABLE_NAME,
        Records = records,
        CommonAttributes = commonAttributes
      };
      WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
      Console.WriteLine($"Write records status code:
{response.HttpStatusCode.ToString()}");
    }
    catch (RejectedRecordsException e) {
      Console.WriteLine("RejectedRecordsException:" + e.ToString());
```

```
      foreach (RejectedRecord rr in e.RejectedRecords) {
        Console.WriteLine("RecordIndex " + rr.RecordIndex + " : " + rr.Reason);
      }
      Console.WriteLine("Other records were written successfully. ");
    }
    catch (Exception e)
    {
      Console.WriteLine("Write records failure:" + e.ToString());
    }
  }
```

Sconvolgimento dei record

Mentre le scritture predefinite in Amazon Timestream seguono la semantica del primo scrittore vince, in cui i dati vengono archiviati solo come aggiunta e i record duplicati vengono rifiutati, alcune applicazioni richiedono la possibilità di scrivere dati in Amazon Timestream utilizzando la semantica last writer wins, in cui il record con la versione più alta viene archiviato nel sistema. Esistono anche applicazioni che richiedono la possibilità di aggiornare i record esistenti. Per affrontare questi scenari, Amazon Timestream offre la possibilità di modificare i dati. Upsert è un'operazione che inserisce un record nel sistema quando il record non esiste o aggiorna il record, quando ne esiste uno.

È possibile alterare i record includendo la definizione `Version` nel record durante l'invio di una richiesta. `WriteRecords` Amazon Timestream memorizzerà il record con il record con il valore più alto. `Version` L'esempio di codice riportato di seguito mostra come modificare i dati:

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
public void writeRecordsWithUpsert() {
  System.out.println("Writing records with upsert");
  // Specify repeated values for all records
  List<Record> records = new ArrayList<>();
  final long time = System.currentTimeMillis();
```

```
    // To achieve upsert (last writer wins) semantic, one example is to use current
 time as the version if you are writing directly from the data source
    long version = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = new Dimension().withName("region").withValue("us-
east-1");
    final Dimension az = new Dimension().withName("az").withValue("az1");
    final Dimension hostname = new
 Dimension().withName("hostname").withValue("host1");

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = new Record()
        .withDimensions(dimensions)
        .withMeasureValueType(MeasureValueType.DOUBLE)
        .withTime(String.valueOf(time))
        .withVersion(version);

    Record cpuUtilization = new Record()
        .withMeasureName("cpu_utilization")
        .withMeasureValue("13.5");
    Record memoryUtilization = new Record()
        .withMeasureName("memory_utilization")
        .withMeasureValue("40");

    records.add(cpuUtilization);
    records.add(memoryUtilization);

    WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withCommonAttributes(commonAttributes);
    writeRecordsRequest.setRecords(records);

    // write records for first time
    try {
      WriteRecordsResult writeRecordsResult =
 amazonTimestreamWrite.writeRecords(writeRecordsRequest);
      System.out.println("WriteRecords Status for first time: " +
 writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
```

```
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }

    // Successfully retry same writeRecordsRequest with same records and versions,
because writeRecords API is idempotent.
    try {
      WriteRecordsResult writeRecordsResult =
amazonTimestreamWrite.writeRecords(writeRecordsRequest);
      System.out.println("WriteRecords Status for retry: " +
writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }

    // upsert with lower version, this would fail because a higher version is
required to update the measure value.
    version -= 1;
    commonAttributes.setVersion(version);

    cpuUtilization.setMeasureValue("14.5");
    memoryUtilization.setMeasureValue("50");

    List<Record> upsertedRecords = new ArrayList<>();
    upsertedRecords.add(cpuUtilization);
    upsertedRecords.add(memoryUtilization);

    WriteRecordsRequest writeRecordsUpsertRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withCommonAttributes(commonAttributes);
    writeRecordsUpsertRequest.setRecords(upsertedRecords);

    try {
      WriteRecordsResult writeRecordsUpsertResult =
amazonTimestreamWrite.writeRecords(writeRecordsUpsertRequest);
      System.out.println("WriteRecords Status for upsert with lower version: " +
writeRecordsUpsertResult.getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      System.out.println("WriteRecords Status for upsert with lower version: ");
      printRejectedRecordsException(e);
```

```
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }

    // upsert with higher version as new data in generated
    version = System.currentTimeMillis();
    commonAttributes.setVersion(version);

    writeRecordsUpsertRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withCommonAttributes(commonAttributes);
    writeRecordsUpsertRequest.setRecords(upsertedRecords);

    try {
      WriteRecordsResult writeRecordsUpsertResult =
 amazonTimestreamWrite.writeRecords(writeRecordsUpsertRequest);
      System.out.println("WriteRecords Status for upsert with higher version: " +
 writeRecordsUpsertResult.getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Java v2

```
  public void writeRecordsWithUpsert() {
    System.out.println("Writing records with upsert");
    // Specify repeated values for all records
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();
    // To achieve upsert (last writer wins) semantic, one example is to use current
 time as the version if you are writing directly from the data source
    long version = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = Dimension.builder().name("region").value("us-
east-1").build();
    final Dimension az = Dimension.builder().name("az").value("az1").build();
    final Dimension hostname =
 Dimension.builder().name("hostname").value("host1").build();
```

```
   dimensions.add(region);
   dimensions.add(az);
   dimensions.add(hostname);

   Record commonAttributes = Record.builder()
       .dimensions(dimensions)
       .measureValueType(MeasureValueType.DOUBLE)
       .time(String.valueOf(time))
       .version(version)
       .build();

   Record cpuUtilization = Record.builder()
       .measureName("cpu_utilization")
       .measureValue("13.5").build();
   Record memoryUtilization = Record.builder()
       .measureName("memory_utilization")
       .measureValue("40").build();

   records.add(cpuUtilization);
   records.add(memoryUtilization);

   WriteRecordsRequest writeRecordsRequest = WriteRecordsRequest.builder()
       .databaseName(DATABASE_NAME)
       .tableName(TABLE_NAME)
       .commonAttributes(commonAttributes)
       .records(records).build();

   // write records for first time
   try {
     WriteRecordsResponse writeRecordsResponse =
timestreamWriteClient.writeRecords(writeRecordsRequest);
     System.out.println("WriteRecords Status for first time: " +
writeRecordsResponse.sdkHttpResponse().statusCode());
   } catch (RejectedRecordsException e) {
     printRejectedRecordsException(e);
   } catch (Exception e) {
     System.out.println("Error: " + e);
   }

   // Successfully retry same writeRecordsRequest with same records and versions,
because writeRecords API is idempotent.
   try {
```

```
     WriteRecordsResponse writeRecordsResponse =
timestreamWriteClient.writeRecords(writeRecordsRequest);
     System.out.println("WriteRecords Status for retry: " +
writeRecordsResponse.sdkHttpResponse().statusCode());
   } catch (RejectedRecordsException e) {
     printRejectedRecordsException(e);
   } catch (Exception e) {
     System.out.println("Error: " + e);
   }

   // upsert with lower version, this would fail because a higher version is
required to update the measure value.
   version -= 1;
   commonAttributes = Record.builder()
       .dimensions(dimensions)
       .measureValueType(MeasureValueType.DOUBLE)
       .time(String.valueOf(time))
       .version(version)
       .build();

   cpuUtilization = Record.builder()
       .measureName("cpu_utilization")
       .measureValue("14.5").build();
   memoryUtilization = Record.builder()
       .measureName("memory_utilization")
       .measureValue("50").build();

   List<Record> upsertedRecords = new ArrayList<>();
   upsertedRecords.add(cpuUtilization);
   upsertedRecords.add(memoryUtilization);

   WriteRecordsRequest writeRecordsUpsertRequest = WriteRecordsRequest.builder()
       .databaseName(DATABASE_NAME)
       .tableName(TABLE_NAME)
       .commonAttributes(commonAttributes)
       .records(upsertedRecords).build();

   try {
     WriteRecordsResponse writeRecordsResponse =
timestreamWriteClient.writeRecords(writeRecordsUpsertRequest);
     System.out.println("WriteRecords Status for upsert with lower version: " +
writeRecordsResponse.sdkHttpResponse().statusCode());
   } catch (RejectedRecordsException e) {
     System.out.println("WriteRecords Status for upsert with lower version: ");
```

```
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }

    // upsert with higher version as new data in generated
    version = System.currentTimeMillis();
    commonAttributes = Record.builder()
        .dimensions(dimensions)
        .measureValueType(MeasureValueType.DOUBLE)
        .time(String.valueOf(time))
        .version(version)
        .build();

    writeRecordsUpsertRequest = WriteRecordsRequest.builder()
        .databaseName(DATABASE_NAME)
        .tableName(TABLE_NAME)
        .commonAttributes(commonAttributes)
        .records(upsertedRecords).build();

    try {
      WriteRecordsResponse writeRecordsUpsertResponse =
 timestreamWriteClient.writeRecords(writeRecordsUpsertRequest);
      System.out.println("WriteRecords Status for upsert with higher version: " +
 writeRecordsUpsertResponse.sdkHttpResponse().statusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }
```

Go

```
// Below code will ingest and upsert cpu_utilization and memory_utilization metric
 for a host on
// region=us-east-1, az=az1, and hostname=host1
fmt.Println("Ingesting records and set version as currentTimeInMills, hit enter to
 continue")
reader.ReadString('\n')

// Get current time in seconds.
now = time.Now()
```

```
currentTimeInSeconds = now.Unix()
// To achieve upsert (last writer wins) semantic, one example is to use current time
 as the version if you are writing directly from the data source
version := time.Now().Round(time.Millisecond).UnixNano() / 1e6   // set version as
 currentTimeInMills

writeRecordsCommonAttributesUpsertInput := &timestreamwrite.WriteRecordsInput{
 DatabaseName: aws.String(*databaseName),
 TableName:  aws.String(*tableName),
 CommonAttributes: &timestreamwrite.Record{
  Dimensions: []*timestreamwrite.Dimension{
   &timestreamwrite.Dimension{
    Name:  aws.String("region"),
    Value: aws.String("us-east-1"),
   },
   &timestreamwrite.Dimension{
    Name:  aws.String("az"),
    Value: aws.String("az1"),
   },
   &timestreamwrite.Dimension{
    Name:  aws.String("hostname"),
    Value: aws.String("host1"),
   },
  },
  MeasureValueType: aws.String("DOUBLE"),
  Time:       aws.String(strconv.FormatInt(currentTimeInSeconds, 10)),
  TimeUnit:  aws.String("SECONDS"),
  Version:      &version,
 },
 Records: []*timestreamwrite.Record{
  &timestreamwrite.Record{
   MeasureName:  aws.String("cpu_utilization"),
   MeasureValue: aws.String("13.5"),
  },
  &timestreamwrite.Record{
   MeasureName:  aws.String("memory_utilization"),
   MeasureValue: aws.String("40"),
  },
 },
}

// write records for first time
_, err = writeSvc.WriteRecords(writeRecordsCommonAttributesUpsertInput)
```

```
if err != nil {
 fmt.Println("Error:")
 fmt.Println(err)
} else {
 fmt.Println("Frist-time write records is successful")
}

fmt.Println("Retry same writeRecordsRequest with same records and versions. Because
 writeRecords API is idempotent, this will success. hit enter to continue")
reader.ReadString('\n')

_, err = writeSvc.WriteRecords(writeRecordsCommonAttributesUpsertInput)

if err != nil {
 fmt.Println("Error:")
 fmt.Println(err)
} else {
 fmt.Println("Retry write records for same request is successful")
}

fmt.Println("Upsert with lower version, this would fail because a higher version is
 required to update the measure value. hit enter to continue")
reader.ReadString('\n')
version -= 1
writeRecordsCommonAttributesUpsertInput.CommonAttributes.Version = &version

updated_cpu_utilization := &timestreamwrite.Record{
 MeasureName:    aws.String("cpu_utilization"),
 MeasureValue:   aws.String("14.5"),
}
updated_memory_utilization := &timestreamwrite.Record{
 MeasureName:    aws.String("memory_utilization"),
 MeasureValue:   aws.String("50"),
}


writeRecordsCommonAttributesUpsertInput.Records = []*timestreamwrite.Record{
 updated_cpu_utilization,
 updated_memory_utilization,
}

_, err = writeSvc.WriteRecords(writeRecordsCommonAttributesUpsertInput)

if err != nil {
```

```
 fmt.Println("Error:")
 fmt.Println(err)
} else {
 fmt.Println("Write records with lower version is successful")
}

fmt.Println("Upsert with higher version as new data in generated, this would
 success. hit enter to continue")
reader.ReadString('\n')

version = time.Now().Round(time.Millisecond).UnixNano() / 1e6  // set version as
 currentTimeInMills
writeRecordsCommonAttributesUpsertInput.CommonAttributes.Version = &version

_, err = writeSvc.WriteRecords(writeRecordsCommonAttributesUpsertInput)

if err != nil {
 fmt.Println("Error:")
 fmt.Println(err)
} else {
 fmt.Println("Write records with higher version is successful")
}
```

Python

```
  def write_records_with_upsert(self):
    print("Writing records with upsert")
    current_time = self._current_milli_time()
    # To achieve upsert (last writer wins) semantic, one example is to use current
 time as the version if you are writing directly from the data source
    version = int(self._current_milli_time())

    dimensions = [
          {'Name': 'region', 'Value': 'us-east-1'},
          {'Name': 'az', 'Value': 'az1'},
          {'Name': 'hostname', 'Value': 'host1'}
        ]

    common_attributes = {
      'Dimensions': dimensions,
      'MeasureValueType': 'DOUBLE',
      'Time': current_time,
      'Version': version
```

```
    }

    cpu_utilization = {
      'MeasureName': 'cpu_utilization',
      'MeasureValue': '13.5'
    }

    memory_utilization = {
      'MeasureName': 'memory_utilization',
      'MeasureValue': '40'
    }

    records = [cpu_utilization, memory_utilization]

    # write records for first time
    try:
      result = self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                         Records=records, CommonAttributes=common_attributes)
      print("WriteRecords Status for first time: [%s]" % result['ResponseMetadata']
['HTTPStatusCode'])
    except self.client.exceptions.RejectedRecordsException as err:
      self._print_rejected_records_exceptions(err)
    except Exception as err:
      print("Error:", err)

    # Successfully retry same writeRecordsRequest with same records and versions,
 because writeRecords API is idempotent.
    try:
      result = self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                         Records=records, CommonAttributes=common_attributes)
      print("WriteRecords Status for retry: [%s]" % result['ResponseMetadata']
['HTTPStatusCode'])
    except self.client.exceptions.RejectedRecordsException as err:
      self._print_rejected_records_exceptions(err)
    except Exception as err:
      print("Error:", err)

    # upsert with lower version, this would fail because a higher version is
 required to update the measure value.
    version -= 1
    common_attributes["Version"] = version
```

```
    cpu_utilization["MeasureValue"] = '14.5'
    memory_utilization["MeasureValue"] = '50'

    upsertedRecords = [cpu_utilization, memory_utilization]

    try:
      upsertedResult =
 self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                           Records=upsertedRecords,
 CommonAttributes=common_attributes)
      print("WriteRecords Status for upsert with lower version: [%s]" %
 upsertedResult['ResponseMetadata']['HTTPStatusCode'])
    except self.client.exceptions.RejectedRecordsException as err:
      self._print_rejected_records_exceptions(err)
    except Exception as err:
      print("Error:", err)


    # upsert with higher version as new data is generated
    version = int(self._current_milli_time())
    common_attributes["Version"] = version

    try:
      upsertedResult =
 self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME,
                           Records=upsertedRecords,
 CommonAttributes=common_attributes)
      print("WriteRecords Upsert Status: [%s]" % upsertedResult['ResponseMetadata']
['HTTPStatusCode'])
    except self.client.exceptions.RejectedRecordsException as err:
      self._print_rejected_records_exceptions(err)
    except Exception as err:
      print("Error:", err)

  @staticmethod
  def _current_milli_time():
    return str(int(round(time.time() * 1000)))
```

Node.js

Il frammento seguente utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function writeRecordsWithUpsert() {
  console.log("Writing records with upsert");
  const currentTime = Date.now().toString(); // Unix time in milliseconds
  // To achieve upsert (last writer wins) semantic, one example is to use current
 time as the version if you are writing directly from the data source
  let version = Date.now();

  const dimensions = [
    {'Name': 'region', 'Value': 'us-east-1'},
    {'Name': 'az', 'Value': 'az1'},
    {'Name': 'hostname', 'Value': 'host1'}
  ];

  const commonAttributes = {
    'Dimensions': dimensions,
    'MeasureValueType': 'DOUBLE',
    'Time': currentTime.toString(),
    'Version': version
  };

  const cpuUtilization = {
    'MeasureName': 'cpu_utilization',
    'MeasureValue': '13.5'
  };

  const memoryUtilization = {
    'MeasureName': 'memory_utilization',
    'MeasureValue': '40'
  };

  const records = [cpuUtilization, memoryUtilization];

  const params = {
    DatabaseName: constants.DATABASE_NAME,
    TableName: constants.TABLE_NAME,
    Records: records,
    CommonAttributes: commonAttributes
```

```
 };

 const request = writeClient.writeRecords(params);

 // write records for first time
 await request.promise().then(
   (data) => {
     console.log("Write records successful for first time.");
   },
   (err) => {
     console.log("Error writing records:", err);
     if (err.code === 'RejectedRecordsException') {
       printRejectedRecordsException(request);
     }
   }
 );

 // Successfully retry same writeRecordsRequest with same records and versions,
because writeRecords API is idempotent.
 await request.promise().then(
   (data) => {
     console.log("Write records successful for retry.");
   },
   (err) => {
     console.log("Error writing records:", err);
     if (err.code === 'RejectedRecordsException') {
       printRejectedRecordsException(request);
     }
   }
 );

 // upsert with lower version, this would fail because a higher version is required
to update the measure value.
 version--;

 const commonAttributesWithLowerVersion = {
   'Dimensions': dimensions,
   'MeasureValueType': 'DOUBLE',
   'Time': currentTime.toString(),
   'Version': version
 };

 const updatedCpuUtilization = {
   'MeasureName': 'cpu_utilization',
```

```
    'MeasureValue': '14.5'
  };

  const updatedMemoryUtilization = {
    'MeasureName': 'memory_utilization',
    'MeasureValue': '50'
  };

  const upsertedRecords = [updatedCpuUtilization, updatedMemoryUtilization];

  const upsertedParamsWithLowerVersion = {
    DatabaseName: constants.DATABASE_NAME,
    TableName: constants.TABLE_NAME,
    Records: upsertedRecords,
    CommonAttributes: commonAttributesWithLowerVersion
  };

  const upsertRequestWithLowerVersion =
writeClient.writeRecords(upsertedParamsWithLowerVersion);

  await upsertRequestWithLowerVersion.promise().then(
    (data) => {
      console.log("Write records for upsert with lower version successful");
    },
    (err) => {
      console.log("Error writing records:", err);
      if (err.code === 'RejectedRecordsException') {
        printRejectedRecordsException(upsertRequestWithLowerVersion);
      }
    }
  );

  // upsert with higher version as new data in generated
  version = Date.now();

  const commonAttributesWithHigherVersion = {
    'Dimensions': dimensions,
    'MeasureValueType': 'DOUBLE',
    'Time': currentTime.toString(),
    'Version': version
  };

  const upsertedParamsWithHigherVerion = {
    DatabaseName: constants.DATABASE_NAME,
```

```
    TableName: constants.TABLE_NAME,
    Records: upsertedRecords,
    CommonAttributes: commonAttributesWithHigherVersion
  };

  const upsertRequestWithHigherVersion =
 writeClient.writeRecords(upsertedParamsWithHigherVerion);

  await upsertRequestWithHigherVersion.promise().then(
    (data) => {
      console.log("Write records upsert successful with higher version");
    },
    (err) => {
      console.log("Error writing records:", err);
      if (err.code === 'RejectedRecordsException') {
        printRejectedRecordsException(upsertedParamsWithHigherVerion);
      }
    }
  );

}
```

.NET

```
  public async Task WriteRecordsWithUpsert()
  {
    Console.WriteLine("Writing records with upsert");

    DateTimeOffset now = DateTimeOffset.UtcNow;
    string currentTimeString = (now.ToUnixTimeMilliseconds()).ToString();
    // To achieve upsert (last writer wins) semantic, one example is to use current
 time as the version if you are writing directly from the data source
    long version = now.ToUnixTimeMilliseconds();

    List<Dimension> dimensions = new List<Dimension>{
      new Dimension { Name = "region", Value = "us-east-1" },
      new Dimension { Name = "az", Value = "az1" },
      new Dimension { Name = "hostname", Value = "host1" }
    };

    var commonAttributes = new Record
    {
      Dimensions = dimensions,
```

```
      MeasureValueType = MeasureValueType.DOUBLE,
      Time = currentTimeString,
      Version = version
    };

    var cpuUtilization = new Record
    {
      MeasureName = "cpu_utilization",
      MeasureValue = "13.6"
    };

    var memoryUtilization = new Record
    {
      MeasureName = "memory_utilization",
      MeasureValue = "40"
    };


    List<Record> records = new List<Record>();
    records.Add(cpuUtilization);
    records.Add(memoryUtilization);

    // write records for first time
    try
    {
      var writeRecordsRequest = new WriteRecordsRequest
      {
        DatabaseName = Constants.DATABASE_NAME,
        TableName = Constants.TABLE_NAME,
        Records = records,
        CommonAttributes = commonAttributes
      };
      WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
      Console.WriteLine($"WriteRecords Status for first time:
{response.HttpStatusCode.ToString()}");
    }
    catch (RejectedRecordsException e) {
      PrintRejectedRecordsException(e);
    }
    catch (Exception e)
    {
      Console.WriteLine("Write records failure:" + e.ToString());
    }
```

```
   // Successfully retry same writeRecordsRequest with same records and versions,
because writeRecords API is idempotent.
   try
   {
     var writeRecordsRequest = new WriteRecordsRequest
     {
       DatabaseName = Constants.DATABASE_NAME,
       TableName = Constants.TABLE_NAME,
       Records = records,
       CommonAttributes = commonAttributes
     };
     WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
     Console.WriteLine($"WriteRecords Status for retry:
{response.HttpStatusCode.ToString()}");
   }
   catch (RejectedRecordsException e) {
     PrintRejectedRecordsException(e);
   }
   catch (Exception e)
   {
     Console.WriteLine("Write records failure:" + e.ToString());
   }

   // upsert with lower version, this would fail because a higher version is
required to update the measure value.
   version--;
   Type recordType = typeof(Record);
   recordType.GetProperty("Version").SetValue(commonAttributes, version);
   recordType.GetProperty("MeasureValue").SetValue(cpuUtilization, "14.6");
   recordType.GetProperty("MeasureValue").SetValue(memoryUtilization, "50");

   List<Record> upsertedRecords = new List<Record> {
     cpuUtilization,
     memoryUtilization
   };

   try
   {
     var writeRecordsUpsertRequest = new WriteRecordsRequest
     {
       DatabaseName = Constants.DATABASE_NAME,
       TableName = Constants.TABLE_NAME,
```

```
        Records = upsertedRecords,
        CommonAttributes = commonAttributes
      };
      WriteRecordsResponse upsertResponse = await
writeClient.WriteRecordsAsync(writeRecordsUpsertRequest);
      Console.WriteLine($"WriteRecords Status for upsert with lower version:
{upsertResponse.HttpStatusCode.ToString()}");
    }
    catch (RejectedRecordsException e) {
      PrintRejectedRecordsException(e);
    }
    catch (Exception e)
    {
      Console.WriteLine("Write records failure:" + e.ToString());
    }

    // upsert with higher version as new data in generated
    now = DateTimeOffset.UtcNow;
    version = now.ToUnixTimeMilliseconds();
    recordType.GetProperty("Version").SetValue(commonAttributes, version);

    try
    {
      var writeRecordsUpsertRequest = new WriteRecordsRequest
      {
        DatabaseName = Constants.DATABASE_NAME,
        TableName = Constants.TABLE_NAME,
        Records = upsertedRecords,
        CommonAttributes = commonAttributes
      };
      WriteRecordsResponse upsertResponse = await
writeClient.WriteRecordsAsync(writeRecordsUpsertRequest);
      Console.WriteLine($"WriteRecords Status for upsert with higher version:
{upsertResponse.HttpStatusCode.ToString()}");
    }
    catch (RejectedRecordsException e) {
      PrintRejectedRecordsException(e);
    }
    catch (Exception e)
    {
      Console.WriteLine("Write records failure:" + e.ToString());
    }
  }
```

Esempio di attributo multimisura

Questo esempio illustra la scrittura di attributi multimisura. Gli attributi multimisura sono utili quando un dispositivo o un'applicazione monitorata emette più metriche o eventi contemporaneamente.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
package com.amazonaws.services.timestream;

import static com.amazonaws.services.timestream.Main.DATABASE_NAME;
import static com.amazonaws.services.timestream.Main.REGION;
import static com.amazonaws.services.timestream.Main.TABLE_NAME;

import java.util.ArrayList;
import java.util.List;

import com.amazonaws.services.timestreamwrite.AmazonTimestreamWrite;
import com.amazonaws.services.timestreamwrite.model.Dimension;
import com.amazonaws.services.timestreamwrite.model.MeasureValue;
import com.amazonaws.services.timestreamwrite.model.MeasureValueType;
import com.amazonaws.services.timestreamwrite.model.Record;
import com.amazonaws.services.timestreamwrite.model.RejectedRecordsException;
import com.amazonaws.services.timestreamwrite.model.WriteRecordsRequest;
import com.amazonaws.services.timestreamwrite.model.WriteRecordsResult;


public class multimeasureAttributeExample {
  AmazonTimestreamWrite timestreamWriteClient;

  public multimeasureAttributeExample(AmazonTimestreamWrite client) {
    this.timestreamWriteClient = client;
  }

  public void writeRecordsMultiMeasureValueSingleRecord() {
    System.out.println("Writing records with multi value attributes");
```

```
    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();
    long version = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = new Dimension().withName("region").withValue(REGION);
    final Dimension az = new Dimension().withName("az").withValue("az1");
    final Dimension hostname = new
Dimension().withName("hostname").withValue("host1");

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = new Record()
        .withDimensions(dimensions)
        .withTime(String.valueOf(time))
        .withVersion(version);

    MeasureValue cpuUtilization = new MeasureValue()
        .withName("cpu_utilization")
        .withType(MeasureValueType.DOUBLE)
        .withValue("13.5");
    MeasureValue memoryUtilization = new MeasureValue()
        .withName("memory_utilization")
        .withType(MeasureValueType.DOUBLE)
        .withValue("40");
    Record computationalResources = new Record()
        .withMeasureName("cpu_memory")
        .withMeasureValues(cpuUtilization, memoryUtilization)
        .withMeasureValueType(MeasureValueType.MULTI);

    records.add(computationalResources);

    WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withCommonAttributes(commonAttributes)
        .withRecords(records);

    // write records for first time
    try {
```

```
      WriteRecordsResult writeRecordResult =
timestreamWriteClient.writeRecords(writeRecordsRequest);
      System.out.println(
          "WriteRecords Status for multi value attributes: " + writeRecordResult
              .getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }

  public void writeRecordsMultiMeasureValueMultipleRecords() {
    System.out.println(
        "Writing records with multi value attributes mixture type");

    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();
    long version = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region = new Dimension().withName("region").withValue(REGION);
    final Dimension az = new Dimension().withName("az").withValue("az1");
    final Dimension hostname = new
Dimension().withName("hostname").withValue("host1");

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = new Record()
        .withDimensions(dimensions)
        .withTime(String.valueOf(time))
        .withVersion(version);

    MeasureValue cpuUtilization = new MeasureValue()
        .withName("cpu_utilization")
        .withType(MeasureValueType.DOUBLE)
        .withValue("13");
    MeasureValue memoryUtilization =new MeasureValue()
        .withName("memory_utilization")
        .withType(MeasureValueType.DOUBLE)
        .withValue("40");
    MeasureValue activeCores = new MeasureValue()
```

```
        .withName("active_cores")
        .withType(MeasureValueType.BIGINT)
        .withValue("4");


    Record computationalResources = new Record()
        .withMeasureName("computational_utilization")
        .withMeasureValues(cpuUtilization, memoryUtilization, activeCores)
        .withMeasureValueType(MeasureValueType.MULTI);

    records.add(computationalResources);

    WriteRecordsRequest writeRecordsRequest = new WriteRecordsRequest()
        .withDatabaseName(DATABASE_NAME)
        .withTableName(TABLE_NAME)
        .withCommonAttributes(commonAttributes)
        .withRecords(records);

    // write records for first time
    try {
      WriteRecordsResult writeRecordResult =
 timestreamWriteClient.writeRecords(writeRecordsRequest);
      System.out.println(
          "WriteRecords Status for multi value attributes: " + writeRecordResult
              .getSdkHttpMetadata().getHttpStatusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }

  private void printRejectedRecordsException(RejectedRecordsException e) {
    System.out.println("RejectedRecords: " + e);
    e.getRejectedRecords().forEach(System.out::println);
  }
}
```

Java v2

```
package com.amazonaws.services.timestream;

import java.util.ArrayList;
```

```
import java.util.List;

import software.amazon.awssdk.services.timestreamwrite.TimestreamWriteClient;
import software.amazon.awssdk.services.timestreamwrite.model.Dimension;
import software.amazon.awssdk.services.timestreamwrite.model.MeasureValue;
import software.amazon.awssdk.services.timestreamwrite.model.MeasureValueType;
import software.amazon.awssdk.services.timestreamwrite.model.Record;
import
 software.amazon.awssdk.services.timestreamwrite.model.RejectedRecordsException;
import software.amazon.awssdk.services.timestreamwrite.model.WriteRecordsRequest;
import software.amazon.awssdk.services.timestreamwrite.model.WriteRecordsResponse;

import static com.amazonaws.services.timestream.Main.DATABASE_NAME;
import static com.amazonaws.services.timestream.Main.TABLE_NAME;


public class multimeasureAttributeExample {

  TimestreamWriteClient timestreamWriteClient;

  public multimeasureAttributeExample(TimestreamWriteClient client) {
    this.timestreamWriteClient = client;
  }

  public void writeRecordsMultiMeasureValueSingleRecord() {
    System.out.println("Writing records with multi value attributes");

    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();
    long version = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region =
        Dimension.builder().name("region").value("us-east-1").build();
    final Dimension az = Dimension.builder().name("az").value("az1").build();
    final Dimension hostname =
        Dimension.builder().name("hostname").value("host1").build();

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = Record.builder()
        .dimensions(dimensions)
```

```
        .time(String.valueOf(time))
        .version(version)
        .build();

    MeasureValue cpuUtilization = MeasureValue.builder()
        .name("cpu_utilization")
        .type(MeasureValueType.DOUBLE)
        .value("13.5").build();
    MeasureValue memoryUtilization = MeasureValue.builder()
        .name("memory_utilization")
        .type(MeasureValueType.DOUBLE)
        .value("40").build();
    Record computationalResources = Record
        .builder()
        .measureName("cpu_memory")
        .measureValues(cpuUtilization, memoryUtilization)
        .measureValueType(MeasureValueType.MULTI)
        .build();

    records.add(computationalResources);

    WriteRecordsRequest writeRecordsRequest = WriteRecordsRequest.builder()
        .databaseName(DATABASE_NAME)
        .tableName(TABLE_NAME)
        .commonAttributes(commonAttributes)
        .records(records).build();

    // write records for first time
    try {
      WriteRecordsResponse writeRecordsResponse =
timestreamWriteClient.writeRecords(writeRecordsRequest);
      System.out.println(
          "WriteRecords Status for multi value attributes: " + writeRecordsResponse
              .sdkHttpResponse()
              .statusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }

  public void writeRecordsMultiMeasureValueMultipleRecords() {
    System.out.println(
```

```
        "Writing records with multi value attributes mixture type");

    List<Record> records = new ArrayList<>();
    final long time = System.currentTimeMillis();
    long version = System.currentTimeMillis();

    List<Dimension> dimensions = new ArrayList<>();
    final Dimension region =
        Dimension.builder().name("region").value("us-east-1").build();
    final Dimension az = Dimension.builder().name("az").value("az1").build();
    final Dimension hostname =
        Dimension.builder().name("hostname").value("host1").build();

    dimensions.add(region);
    dimensions.add(az);
    dimensions.add(hostname);

    Record commonAttributes = Record.builder()
        .dimensions(dimensions)
        .time(String.valueOf(time))
        .version(version)
        .build();

    MeasureValue cpuUtilization = MeasureValue.builder()
        .name("cpu_utilization")
        .type(MeasureValueType.DOUBLE)
        .value("13.5").build();
    MeasureValue memoryUtilization = MeasureValue.builder()
        .name("memory_utilization")
        .type(MeasureValueType.DOUBLE)
        .value("40").build();
    MeasureValue activeCores = MeasureValue.builder()
        .name("active_cores")
        .type(MeasureValueType.BIGINT)
        .value("4").build();


    Record computationalResources = Record
        .builder()
        .measureName("computational_utilization")
        .measureValues(cpuUtilization, memoryUtilization, activeCores)
        .measureValueType(MeasureValueType.MULTI)
        .build();
```

```
    records.add(computationalResources);

    WriteRecordsRequest writeRecordsRequest = WriteRecordsRequest.builder()
        .databaseName(DATABASE_NAME)
        .tableName(TABLE_NAME)
        .commonAttributes(commonAttributes)
        .records(records).build();

    // write records for first time
    try {
      WriteRecordsResponse writeRecordsResponse =
 timestreamWriteClient.writeRecords(writeRecordsRequest);
      System.out.println(
          "WriteRecords Status for multi value attributes: " + writeRecordsResponse
              .sdkHttpResponse()
              .statusCode());
    } catch (RejectedRecordsException e) {
      printRejectedRecordsException(e);
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
  }

  private void printRejectedRecordsException(RejectedRecordsException e) {
    System.out.println("RejectedRecords: " + e);
    e.rejectedRecords().forEach(System.out::println);
  }
}
```

Go

```
  now := time.Now()
  currentTimeInSeconds := now.Unix()
  writeRecordsInput := &timestreamwrite.WriteRecordsInput{
    DatabaseName: aws.String(*databaseName),
    TableName:  aws.String(*tableName),
    Records: []*timestreamwrite.Record{
    &timestreamwrite.Record{
      Dimensions: []*timestreamwrite.Dimension{
      &timestreamwrite.Dimension{
        Name:  aws.String("region"),
        Value: aws.String("us-east-1"),
      },
```

```
      &timestreamwrite.Dimension{
        Name:  aws.String("az"),
        Value: aws.String("az1"),
      },
      &timestreamwrite.Dimension{
        Name:  aws.String("hostname"),
        Value: aws.String("host1"),
      },
      },
      MeasureName:  aws.String("metrics"),
      MeasureValueType: aws.String("MULTI"),
      Time:     aws.String(strconv.FormatInt(currentTimeInSeconds, 10)),
      TimeUnit:  aws.String("SECONDS"),
      MeasureValues: []*timestreamwrite.MeasureValue{
      &timestreamwrite.MeasureValue{
        Name:  aws.String("cpu_utilization"),
        Value: aws.String("13.5"),
        Type:  aws.String("DOUBLE"),
      },
      &timestreamwrite.MeasureValue{
        Name:  aws.String("memory_utilization"),
        Value: aws.String("40"),
        Type:  aws.String("DOUBLE"),
      },
      },
    },
    },
  }

  _, err = writeSvc.WriteRecords(writeRecordsInput)

  if err != nil {
    fmt.Println("Error:")
    fmt.Println(err)
  } else {
    fmt.Println("Write records is successful")
  }
```

Python

```
import time
import boto3
import psutil
```

```
import os

from botocore.config import Config

DATABASE_NAME = os.environ['DATABASE_NAME']
TABLE_NAME = os.environ['TABLE_NAME']

COUNTRY = "UK"
CITY = "London"
HOSTNAME = "MyHostname" # You can make it dynamic using socket.gethostname()

INTERVAL = 1 # Seconds

def prepare_common_attributes():
  common_attributes = {
    'Dimensions': [
      {'Name': 'country', 'Value': COUNTRY},
      {'Name': 'city', 'Value': CITY},
      {'Name': 'hostname', 'Value': HOSTNAME}
    ],
    'MeasureName': 'utilization',
    'MeasureValueType': 'MULTI'
  }
  return common_attributes


def prepare_record(current_time):
  record = {
    'Time': str(current_time),
    'MeasureValues': []
  }
  return record


def prepare_measure(measure_name, measure_value):
  measure = {
    'Name': measure_name,
    'Value': str(measure_value),
    'Type': 'DOUBLE'
  }
  return measure


def write_records(records, common_attributes):
```

```
  try:
    result = write_client.write_records(DatabaseName=DATABASE_NAME,
                                        TableName=TABLE_NAME,
                                        CommonAttributes=common_attributes,
                                        Records=records)
    status = result['ResponseMetadata']['HTTPStatusCode']
    print("Processed %d records. WriteRecords HTTPStatusCode: %s" %
        (len(records), status))
  except Exception as err:
    print("Error:", err)


if __name__ == '__main__':

  print("writing data to database {} table {}".format(
    DATABASE_NAME, TABLE_NAME))

  session = boto3.Session()
  write_client = session.client('timestream-write', config=Config(
    read_timeout=20, max_pool_connections=5000, retries={'max_attempts': 10}))
  query_client = session.client('timestream-query') # Not used

  common_attributes = prepare_common_attributes()

  records = []

  while True:

    current_time = int(time.time() * 1000)
    cpu_utilization = psutil.cpu_percent()
    memory_utilization = psutil.virtual_memory().percent
    swap_utilization = psutil.swap_memory().percent
    disk_utilization = psutil.disk_usage('/').percent

    record = prepare_record(current_time)
    record['MeasureValues'].append(prepare_measure('cpu', cpu_utilization))
    record['MeasureValues'].append(prepare_measure('memory', memory_utilization))
    record['MeasureValues'].append(prepare_measure('swap', swap_utilization))
    record['MeasureValues'].append(prepare_measure('disk', disk_utilization))

    records.append(record)

    print("records {} - cpu {} - memory {} - swap {} - disk {}".format(
      len(records), cpu_utilization, memory_utilization,
```

```
      swap_utilization, disk_utilization))

    if len(records) == 100:
      write_records(records, common_attributes)
      records = []

    time.sleep(INTERVAL)
```

Node.js

Il frammento seguente utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function writeRecords() {
  console.log("Writing records");
  const currentTime = Date.now().toString(); // Unix time in milliseconds

  const dimensions = [
  {'Name': 'region', 'Value': 'us-east-1'},
  {'Name': 'az', 'Value': 'az1'},
  {'Name': 'hostname', 'Value': 'host1'}
  ];

  const record = {
  'Dimensions': dimensions,
  'MeasureName': 'metrics',
  'MeasureValues': [
    {
      'Name': 'cpu_utilization',
      'Value': '40',
      'Type': 'DOUBLE',
    },
    {
      'Name': 'memory_utilization',
      'Value': '13.5',
      'Type': 'DOUBLE',
    },
    ],
    'MeasureValueType': 'MULTI',
    'Time': currentTime.toString()
  }
```

```
    const records = [record];

    const params = {
    DatabaseName: 'DatabaseName',
    TableName: 'TableName',
    Records: records
    };

    const response = await writeClient.writeRecords(params);

    console.log(response);
  }
```

.NET

```
using System;
using System.IO;
using System.Collections.Generic;
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using System.Threading.Tasks;

namespace TimestreamDotNetSample
{
  static class MultiMeasureValueConstants
  {
    public const string MultiMeasureValueSampleDb = "multiMeasureValueSampleDb";
    public const string MultiMeasureValueSampleTable =
 "multiMeasureValueSampleTable";
  }

  public class MultiValueAttributesExample
  {
    private readonly AmazonTimestreamWriteClient writeClient;

    public MultiValueAttributesExample(AmazonTimestreamWriteClient writeClient)
    {
      this.writeClient = writeClient;
    }

    public async Task WriteRecordsMultiMeasureValueSingleRecord()
    {
      Console.WriteLine("Writing records with multi value attributes");
```

```
DateTimeOffset now = DateTimeOffset.UtcNow;
string currentTimeString = (now.ToUnixTimeMilliseconds()).ToString();

List<Dimension> dimensions = new List<Dimension>{
  new Dimension { Name = "region", Value = "us-east-1" },
  new Dimension { Name = "az", Value = "az1" },
  new Dimension { Name = "hostname", Value = "host1" }
};

var commonAttributes = new Record
{
  Dimensions = dimensions,
  Time = currentTimeString
};

var cpuUtilization = new MeasureValue
{
  Name = "cpu_utilization",
  Value = "13.6",
  Type = "DOUBLE"
};

var memoryUtilization = new MeasureValue
{
  Name = "memory_utilization",
  Value = "40",
  Type = "DOUBLE"
};

var computationalRecord = new Record
{
  MeasureName = "cpu_memory",
  MeasureValues = new List<MeasureValue> {cpuUtilization, memoryUtilization},
  MeasureValueType = "MULTI"
};


List<Record> records = new List<Record>();
records.Add(computationalRecord);

try
{
  var writeRecordsRequest = new WriteRecordsRequest
```

```
        {
          DatabaseName = MultiMeasureValueConstants.MultiMeasureValueSampleDb,
          TableName = MultiMeasureValueConstants.MultiMeasureValueSampleTable,
          Records = records,
          CommonAttributes = commonAttributes
        };
        WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
        Console.WriteLine($"Write records status code:
{response.HttpStatusCode.ToString()}");
      }
      catch (Exception e)
      {
        Console.WriteLine("Write records failure:" + e.ToString());
      }
    }

    public async Task WriteRecordsMultiMeasureValueMultipleRecords()
    {
      Console.WriteLine("Writing records with multi value attributes mixture type");

      DateTimeOffset now = DateTimeOffset.UtcNow;
      string currentTimeString = (now.ToUnixTimeMilliseconds()).ToString();

      List<Dimension> dimensions = new List<Dimension>{
        new Dimension { Name = "region", Value = "us-east-1" },
        new Dimension { Name = "az", Value = "az1" },
        new Dimension { Name = "hostname", Value = "host1" }
      };

      var commonAttributes = new Record
      {
        Dimensions = dimensions,
        Time = currentTimeString
      };

      var cpuUtilization = new MeasureValue
      {
        Name = "cpu_utilization",
        Value = "13.6",
        Type = "DOUBLE"
      };

      var memoryUtilization = new MeasureValue
```

```
      {
        Name = "memory_utilization",
        Value = "40",
        Type = "DOUBLE"
      };

      var activeCores = new MeasureValue
      {
        Name = "active_cores",
        Value = "4",
        Type = "BIGINT"
      };

      var computationalRecord = new Record
      {
        MeasureName = "computational_utilization",
        MeasureValues = new List<MeasureValue> {cpuUtilization, memoryUtilization,
activeCores},
        MeasureValueType = "MULTI"
      };

      var aliveRecord = new Record
      {
        MeasureName = "is_healthy",
        MeasureValue = "true",
        MeasureValueType = "BOOLEAN"
      };

      List<Record> records = new List<Record>();
      records.Add(computationalRecord);
      records.Add(aliveRecord);

      try
      {
        var writeRecordsRequest = new WriteRecordsRequest
        {
          DatabaseName = MultiMeasureValueConstants.MultiMeasureValueSampleDb,
          TableName = MultiMeasureValueConstants.MultiMeasureValueSampleTable,
          Records = records,
          CommonAttributes = commonAttributes
        };
        WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
```

```
        Console.WriteLine($"Write records status code:
 {response.HttpStatusCode.ToString()}");
      }
      catch (Exception e)
      {
        Console.WriteLine("Write records failure:" + e.ToString());
      }
    }
  }
}
```

Gestione degli errori di scrittura

Le scritture in Amazon Timestream possono avere esito negativo per uno o più dei seguenti motivi:

- Esistono record con timestamp che non rientrano nella durata di conservazione dell'archivio di memoria.
- Esistono record contenenti dimensioni e/o misure che superano i limiti definiti da Timestream.
- Amazon Timestream ha rilevato record duplicati. I record vengono contrassegnati come duplicati quando sono presenti più record con le stesse dimensioni, timestamp e nomi di misure ma:
  - I valori di misura sono diversi.
  - La versione non è presente nella richiesta oppure il valore della versione nel nuovo record è uguale o inferiore al valore esistente. Se Amazon Timestream rifiuta i dati per questo motivo, `ExistingVersion` il campo in conterrà la versione corrente `RejectedRecords` del record archiviata in Amazon Timestream. Per forzare un aggiornamento, puoi inviare nuovamente la richiesta con una versione del record impostata su un valore maggiore di. `ExistingVersion`

Per ulteriori informazioni sugli errori e sui record rifiutati, vedere Errori e RejectedRecord.

Se la tua applicazione riceve un messaggio `RejectedRecordsException` quando tenta di scrivere record su Timestream, puoi analizzare i record rifiutati per saperne di più sugli errori di scrittura, come mostrato di seguito.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
 try {
   WriteRecordsResult writeRecordsResult =
amazonTimestreamWrite.writeRecords(writeRecordsRequest);
   System.out.println("WriteRecords Status: " +
writeRecordsResult.getSdkHttpMetadata().getHttpStatusCode());
 } catch (RejectedRecordsException e) {
   System.out.println("RejectedRecords: " + e);
   for (RejectedRecord rejectedRecord : e.getRejectedRecords()) {
     System.out.println("Rejected Index " + rejectedRecord.getRecordIndex() + ": "
         + rejectedRecord.getReason());
   }
   System.out.println("Other records were written successfully. ");
 } catch (Exception e) {
   System.out.println("Error: " + e);
 }
```

Java v2

```
   try {
     WriteRecordsResponse writeRecordsResponse =
timestreamWriteClient.writeRecords(writeRecordsRequest);
     System.out.println("writeRecordsWithCommonAttributes Status: " +
writeRecordsResponse.sdkHttpResponse().statusCode());
   } catch (RejectedRecordsException e) {
     System.out.println("RejectedRecords: " + e);
     for (RejectedRecord rejectedRecord : e.rejectedRecords()) {
       System.out.println("Rejected Index " + rejectedRecord.recordIndex() + ": "
           + rejectedRecord.reason());
     }
     System.out.println("Other records were written successfully. ");
   } catch (Exception e) {
     System.out.println("Error: " + e);
```

```
    }
```

Go

```
_, err = writeSvc.WriteRecords(writeRecordsInput)

if err != nil {
  fmt.Println("Error:")
  fmt.Println(err)
} else {
  fmt.Println("Write records is successful")
}
```

Python

```
try:
  result = self.client.write_records(DatabaseName=Constant.DATABASE_NAME,
 TableName=Constant.TABLE_NAME, Records=records, CommonAttributes=common_attributes)
  print("WriteRecords Status: [%s]" % result['ResponseMetadata']['HTTPStatusCode'])
except self.client.exceptions.RejectedRecordsException as err:
  print("RejectedRecords: ", err)
  for rr in err.response["RejectedRecords"]:
    print("Rejected Index " + str(rr["RecordIndex"]) + ": " + rr["Reason"])
  print("Other records were written successfully. ")
except Exception as err:
  print("Error:", err)
```

Node.js

Il frammento seguente utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
await request.promise().then(
    (data) => {
      console.log("Write records successful");
    },
    (err) => {
      console.log("Error writing records:", err);
      if (err.code === 'RejectedRecordsException') {
        const responsePayload =
 JSON.parse(request.response.httpResponse.body.toString());
```

```
        console.log("RejectedRecords: ", responsePayload.RejectedRecords);
        console.log("Other records were written successfully. ");
      }
    }
  );
```

.NET

```
 try
 {
   var writeRecordsRequest = new WriteRecordsRequest
   {
     DatabaseName = Constants.DATABASE_NAME,
     TableName = Constants.TABLE_NAME,
     Records = records,
     CommonAttributes = commonAttributes
   };
   WriteRecordsResponse response = await
writeClient.WriteRecordsAsync(writeRecordsRequest);
   Console.WriteLine($"Write records status code:
{response.HttpStatusCode.ToString()}");
 }
 catch (RejectedRecordsException e) {
   Console.WriteLine("RejectedRecordsException:" + e.ToString());
   foreach (RejectedRecord rr in e.RejectedRecords) {
     Console.WriteLine("RecordIndex " + rr.RecordIndex + " : " + rr.Reason);
   }
   Console.WriteLine("Other records were written successfully. ");
 }
 catch (Exception e)
 {
   Console.WriteLine("Write records failure:" + e.ToString());
 }
```

Esegui interrogazione

Argomenti

- Impaginazione dei risultati
- Analisi dei set di risultati
- Accesso allo stato della query

Impaginazione dei risultati

Quando si esegue una query, Timestream restituisce il set di risultati in modo impaginato per ottimizzare la reattività delle applicazioni. Il frammento di codice seguente mostra come è possibile impaginare il set di risultati. È necessario scorrere tutte le pagine del set di risultati finché non viene visualizzato un valore nullo. I token di paginazione scadono 3 ore dopo essere stati emessi da Timestream per. LiveAnalytics

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
    private void runQuery(String queryString) {
        try {
            QueryRequest queryRequest = new QueryRequest();
            queryRequest.setQueryString(queryString);
            QueryResult queryResult = queryClient.query(queryRequest);
            while (true) {
                parseQueryResult(queryResult);
                if (queryResult.getNextToken() == null) {
                    break;
                }
                queryRequest.setNextToken(queryResult.getNextToken());
                queryResult = queryClient.query(queryRequest);
            }
        } catch (Exception e) {
            // Some queries might fail with 500 if the result of a sequence function
has more than 10000 entries
            e.printStackTrace();
        }
    }
```

Java v2

```
    private void runQuery(String queryString) {
        try {
```

```
            QueryRequest queryRequest =
 QueryRequest.builder().queryString(queryString).build();
            final QueryIterable queryResponseIterator =
 timestreamQueryClient.queryPaginator(queryRequest);
            for(QueryResponse queryResponse : queryResponseIterator) {
                parseQueryResult(queryResponse);
            }
        } catch (Exception e) {
            // Some queries might fail with 500 if the result of a sequence function
 has more than 10000 entries
            e.printStackTrace();
        }
    }
```

Go

```
func runQuery(queryPtr *string, querySvc *timestreamquery.TimestreamQuery, f
 *os.File) {
    queryInput := &timestreamquery.QueryInput{
        QueryString: aws.String(*queryPtr),
    }
    fmt.Println("QueryInput:")
    fmt.Println(queryInput)
    // execute the query
    err := querySvc.QueryPages(queryInput,
        func(page *timestreamquery.QueryOutput, lastPage bool) bool {
            // process query response
            queryStatus := page.QueryStatus
            fmt.Println("Current query status:", queryStatus)
            // query response metadata
            // includes column names and types
            metadata := page.ColumnInfo
            // fmt.Println("Metadata:")
            fmt.Println(metadata)
            header := ""
            for i := 0; i < len(metadata); i++ {
                header += *metadata[i].Name
                if i != len(metadata)-1 {
                    header += ", "
                }
            }
            write(f, header)
```

```
            // query response data
            fmt.Println("Data:")
            // process rows
            rows := page.Rows
            for i := 0; i < len(rows); i++ {
                data := rows[i].Data
                value := processRowType(data, metadata)
                fmt.Println(value)
                write(f, value)
            }
            fmt.Println("Number of rows:", len(page.Rows))
            return true
        })
    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    }
}
```

Python

```
    def run_query(self, query_string):
        try:
            page_iterator = self.paginator.paginate(QueryString=query_string)
            for page in page_iterator:
                self._parse_query_result(page)
        except Exception as err:
            print("Exception while running query:", err)
```

Node.js

Il frammento seguente utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function getAllRows(query, nextToken) {
    const params = {
        QueryString: query
    };

    if (nextToken) {
        params.NextToken = nextToken;
```

```
    }

    await queryClient.query(params).promise()
        .then(
            (response) => {
                parseQueryResult(response);
                if (response.NextToken) {
                    getAllRows(query, response.NextToken);
                }
            },
            (err) => {
                console.error("Error while querying:", err);
            });
}
```

.NET

```
        private async Task RunQueryAsync(string queryString)
        {
            try
            {
                QueryRequest queryRequest = new QueryRequest();
                queryRequest.QueryString = queryString;
                QueryResponse queryResponse = await
 queryClient.QueryAsync(queryRequest);
                while (true)
                {
                    ParseQueryResult(queryResponse);
                    if (queryResponse.NextToken == null)
                    {
                        break;
                    }
                    queryRequest.NextToken = queryResponse.NextToken;
                    queryResponse = await queryClient.QueryAsync(queryRequest);
                }
            } catch(Exception e)
            {
                // Some queries might fail with 500 if the result of a sequence
 function has more than 10000 entries
                Console.WriteLine(e.ToString());
            }
        }
```

Analisi dei set di risultati

È possibile utilizzare i seguenti frammenti di codice per estrarre dati dal set di risultati. I risultati delle query sono accessibili per un massimo di 24 ore dopo il completamento di una query.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
   private static final DateTimeFormatter TIMESTAMP_FORMATTER =
DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss.SSSSSSSSS");
   private static final DateTimeFormatter DATE_FORMATTER =
DateTimeFormatter.ofPattern("yyyy-MM-dd");
   private static final DateTimeFormatter TIME_FORMATTER =
DateTimeFormatter.ofPattern("HH:mm:ss.SSSSSSSSS");

   private static final long ONE_GB_IN_BYTES = 1073741824L;

   private void parseQueryResult(QueryResult response) {
       final QueryStatus currentStatusOfQuery = queryResult.getQueryStatus();

       System.out.println("Query progress so far: " +
currentStatusOfQuery.getProgressPercentage() + "%");

       double bytesScannedSoFar = ((double)
currentStatusOfQuery.getCumulativeBytesScanned() / ONE_GB_IN_BYTES);
       System.out.println("Bytes scanned so far: " + bytesScannedSoFar + " GB");

       double bytesMeteredSoFar = ((double)
currentStatusOfQuery.getCumulativeBytesMetered() / ONE_GB_IN_BYTES);
       System.out.println("Bytes metered so far: " + bytesMeteredSoFar + " GB");

       List<ColumnInfo> columnInfo = response.getColumnInfo();
       List<Row> rows = response.getRows();

       System.out.println("Metadata: " + columnInfo);
       System.out.println("Data: ");
```

```
        // iterate every row
        for (Row row : rows) {
            System.out.println(parseRow(columnInfo, row));
        }
    }

    private String parseRow(List<ColumnInfo> columnInfo, Row row) {
        List<Datum> data = row.getData();
        List<String> rowOutput = new ArrayList<>();
        // iterate every column per row
        for (int j = 0; j < data.size(); j++) {
            ColumnInfo info = columnInfo.get(j);
            Datum datum = data.get(j);
            rowOutput.add(parseDatum(info, datum));
        }
        return String.format("{%s}",
rowOutput.stream().map(Object::toString).collect(Collectors.joining(",")));
    }

    private String parseDatum(ColumnInfo info, Datum datum) {
        if (datum.isNullValue() != null && datum.isNullValue()) {
            return info.getName() + "=" + "NULL";
        }
        Type columnType = info.getType();
        // If the column is of TimeSeries Type
        if (columnType.getTimeSeriesMeasureValueColumnInfo() != null) {
            return parseTimeSeries(info, datum);
        }
        // If the column is of Array Type
        else if (columnType.getArrayColumnInfo() != null) {
            List<Datum> arrayValues = datum.getArrayValue();
            return info.getName() + "=" +
parseArray(info.getType().getArrayColumnInfo(), arrayValues);
        }
        // If the column is of Row Type
        else if (columnType.getRowColumnInfo() != null) {
            List<ColumnInfo> rowColumnInfo = info.getType().getRowColumnInfo();
            Row rowValues = datum.getRowValue();
            return parseRow(rowColumnInfo, rowValues);
        }
        // If the column is of Scalar Type
        else {
            return parseScalarType(info, datum);
```

```
        }
    }

    private String parseTimeSeries(ColumnInfo info, Datum datum) {
        List<String> timeSeriesOutput = new ArrayList<>();
        for (TimeSeriesDataPoint dataPoint : datum.getTimeSeriesValue()) {
            timeSeriesOutput.add("{time=" + dataPoint.getTime() + ", value=" +
                    parseDatum(info.getType().getTimeSeriesMeasureValueColumnInfo(),
dataPoint.getValue()) + "}");
        }
        return String.format("[%s]",
timeSeriesOutput.stream().map(Object::toString).collect(Collectors.joining(",")));
    }

    private String parseScalarType(ColumnInfo info, Datum datum) {
        switch (ScalarType.fromValue(info.getType().getScalarType())) {
            case VARCHAR:
                return parseColumnName(info) + datum.getScalarValue();
            case BIGINT:
                Long longValue = Long.valueOf(datum.getScalarValue());
                return parseColumnName(info) + longValue;
            case INTEGER:
                Integer intValue = Integer.valueOf(datum.getScalarValue());
                return parseColumnName(info) + intValue;
            case BOOLEAN:
                Boolean booleanValue = Boolean.valueOf(datum.getScalarValue());
                return parseColumnName(info) + booleanValue;
            case DOUBLE:
                Double doubleValue = Double.valueOf(datum.getScalarValue());
                return parseColumnName(info) + doubleValue;
            case TIMESTAMP:
                return parseColumnName(info) +
LocalDateTime.parse(datum.getScalarValue(), TIMESTAMP_FORMATTER);
            case DATE:
                return parseColumnName(info) +
LocalDate.parse(datum.getScalarValue(), DATE_FORMATTER);
            case TIME:
                return parseColumnName(info) +
LocalTime.parse(datum.getScalarValue(), TIME_FORMATTER);
            case INTERVAL_DAY_TO_SECOND:
            case INTERVAL_YEAR_TO_MONTH:
                return parseColumnName(info) + datum.getScalarValue();
            case UNKNOWN:
                return parseColumnName(info) + datum.getScalarValue();
```

```
            default:
                throw new IllegalArgumentException("Given type is not valid: " +
info.getType().getScalarType());
        }
    }

    private String parseColumnName(ColumnInfo info) {
        return info.getName() == null ? "" : info.getName() + "=";
    }

    private String parseArray(ColumnInfo arrayColumnInfo, List<Datum> arrayValues) {
        List<String> arrayOutput = new ArrayList<>();
        for (Datum datum : arrayValues) {
            arrayOutput.add(parseDatum(arrayColumnInfo, datum));
        }
        return String.format("[%s]",
arrayOutput.stream().map(Object::toString).collect(Collectors.joining(",")));
    }
```

Java v2

```
    private static final long ONE_GB_IN_BYTES = 1073741824L;

    private void parseQueryResult(QueryResponse response) {
        final QueryStatus currentStatusOfQuery = response.queryStatus();

        System.out.println("Query progress so far: " +
currentStatusOfQuery.progressPercentage() + "%");

        double bytesScannedSoFar = ((double)
currentStatusOfQuery.cumulativeBytesScanned() / ONE_GB_IN_BYTES);
        System.out.println("Bytes scanned so far: " + bytesScannedSoFar + " GB");

        double bytesMeteredSoFar = ((double)
currentStatusOfQuery.cumulativeBytesMetered() / ONE_GB_IN_BYTES);
        System.out.println("Bytes metered so far: " + bytesMeteredSoFar + " GB");

        List<ColumnInfo> columnInfo = response.columnInfo();
        List<Row> rows = response.rows();

        System.out.println("Metadata: " + columnInfo);
        System.out.println("Data: ");
```

```
        // iterate every row
        for (Row row : rows) {
            System.out.println(parseRow(columnInfo, row));
        }
    }

    private String parseRow(List<ColumnInfo> columnInfo, Row row) {
        List<Datum> data = row.data();
        List<String> rowOutput = new ArrayList<>();
        // iterate every column per row
        for (int j = 0; j < data.size(); j++) {
            ColumnInfo info = columnInfo.get(j);
            Datum datum = data.get(j);
            rowOutput.add(parseDatum(info, datum));
        }
        return String.format("{%s}",
rowOutput.stream().map(Object::toString).collect(Collectors.joining(",")));
    }

    private String parseDatum(ColumnInfo info, Datum datum) {
        if (datum.nullValue() != null && datum.nullValue()) {
            return info.name() + "=" + "NULL";
        }
        Type columnType = info.type();
        // If the column is of TimeSeries Type
        if (columnType.timeSeriesMeasureValueColumnInfo() != null) {
            return parseTimeSeries(info, datum);
        }
        // If the column is of Array Type
        else if (columnType.arrayColumnInfo() != null) {
            List<Datum> arrayValues = datum.arrayValue();
            return info.name() + "=" + parseArray(info.type().arrayColumnInfo(),
arrayValues);
        }
        // If the column is of Row Type
        else if (columnType.rowColumnInfo() != null &&
columnType.rowColumnInfo().size() > 0) {
            List<ColumnInfo> rowColumnInfo = info.type().rowColumnInfo();
            Row rowValues = datum.rowValue();
            return parseRow(rowColumnInfo, rowValues);
        }
        // If the column is of Scalar Type
        else {
            return parseScalarType(info, datum);
```

```
        }
    }

    private String parseTimeSeries(ColumnInfo info, Datum datum) {
        List<String> timeSeriesOutput = new ArrayList<>();
        for (TimeSeriesDataPoint dataPoint : datum.timeSeriesValue()) {
            timeSeriesOutput.add("{time=" + dataPoint.time() + ", value=" +
                    parseDatum(info.type().timeSeriesMeasureValueColumnInfo(),
 dataPoint.value()) + "}");
        }
        return String.format("[%s]",
 timeSeriesOutput.stream().map(Object::toString).collect(Collectors.joining(",")));
    }

    private String parseScalarType(ColumnInfo info, Datum datum) {
        return parseColumnName(info) + datum.scalarValue();
    }

    private String parseColumnName(ColumnInfo info) {
        return info.name() == null ? "" : info.name() + "=";
    }

    private String parseArray(ColumnInfo arrayColumnInfo, List<Datum> arrayValues) {
        List<String> arrayOutput = new ArrayList<>();
        for (Datum datum : arrayValues) {
            arrayOutput.add(parseDatum(arrayColumnInfo, datum));
        }
        return String.format("[%s]",
 arrayOutput.stream().map(Object::toString).collect(Collectors.joining(",")));
    }
```

Go

```
func processScalarType(data *timestreamquery.Datum) string {
    return *data.ScalarValue
}

func processTimeSeriesType(data []*timestreamquery.TimeSeriesDataPoint, columnInfo
 *timestreamquery.ColumnInfo) string {
    value := ""
    for k := 0; k < len(data); k++ {
        time := data[k].Time
        value += *time + ":"
```

```
        if columnInfo.Type.ScalarType != nil {
            value += processScalarType(data[k].Value)
        } else if columnInfo.Type.ArrayColumnInfo != nil {
            value += processArrayType(data[k].Value.ArrayValue,
 columnInfo.Type.ArrayColumnInfo)
        } else if columnInfo.Type.RowColumnInfo != nil {
            value += processRowType(data[k].Value.RowValue.Data,
 columnInfo.Type.RowColumnInfo)
        } else {
            fail("Bad data type")
        }
        if k != len(data)-1 {
            value += ", "
        }
    }
    return value
}

func processArrayType(datumList []*timestreamquery.Datum, columnInfo
 *timestreamquery.ColumnInfo) string {
    value := ""
    for k := 0; k < len(datumList); k++ {
        if columnInfo.Type.ScalarType != nil {
            value += processScalarType(datumList[k])
        } else if columnInfo.Type.TimeSeriesMeasureValueColumnInfo != nil {
            value += processTimeSeriesType(datumList[k].TimeSeriesValue,
 columnInfo.Type.TimeSeriesMeasureValueColumnInfo)
        } else if columnInfo.Type.ArrayColumnInfo != nil {
            value += "["
            value += processArrayType(datumList[k].ArrayValue,
 columnInfo.Type.ArrayColumnInfo)
            value += "]"
        } else if columnInfo.Type.RowColumnInfo != nil {
            value += "["
            value += processRowType(datumList[k].RowValue.Data,
 columnInfo.Type.RowColumnInfo)
            value += "]"
        } else {
            fail("Bad column type")
        }

        if k != len(datumList)-1 {
            value += ", "
        }
```

```
    }
    return value
}

func processRowType(data []*timestreamquery.Datum, metadata
 []*timestreamquery.ColumnInfo) string {
    value := ""
    for j := 0; j < len(data); j++ {
        if metadata[j].Type.ScalarType != nil {
            // process simple data types
            value += processScalarType(data[j])
        } else if metadata[j].Type.TimeSeriesMeasureValueColumnInfo != nil {
            // fmt.Println("Timeseries measure value column info")
            // fmt.Println(metadata[j].Type.TimeSeriesMeasureValueColumnInfo.Type)
            datapointList := data[j].TimeSeriesValue
            value += "["
            value += processTimeSeriesType(datapointList,
 metadata[j].Type.TimeSeriesMeasureValueColumnInfo)
            value += "]"
        } else if metadata[j].Type.ArrayColumnInfo != nil {
            columnInfo := metadata[j].Type.ArrayColumnInfo
            // fmt.Println("Array column info")
            // fmt.Println(columnInfo)
            datumList := data[j].ArrayValue
            value += "["
            value += processArrayType(datumList, columnInfo)
            value += "]"
        } else if metadata[j].Type.RowColumnInfo != nil {
            columnInfo := metadata[j].Type.RowColumnInfo
            datumList := data[j].RowValue.Data
            value += "["
            value += processRowType(datumList, columnInfo)
            value += "]"
        } else {
            panic("Bad column type")
        }
        // comma seperated column values
        if j != len(data)-1 {
            value += ", "
        }
    }
    return value
}
```

Python

```
   def _parse_query_result(self, query_result):
       query_status = query_result["QueryStatus"]

       progress_percentage = query_status["ProgressPercentage"]
       print(f"Query progress so far: {progress_percentage}%")

       bytes_scanned = float(query_status["CumulativeBytesScanned"]) /
ONE_GB_IN_BYTES
       print(f"Data scanned so far: {bytes_scanned} GB")

       bytes_metered = float(query_status["CumulativeBytesMetered"]) /
ONE_GB_IN_BYTES
       print(f"Data metered so far: {bytes_metered} GB")

       column_info = query_result['ColumnInfo']

       print("Metadata: %s" % column_info)
       print("Data: ")
       for row in query_result['Rows']:
           print(self._parse_row(column_info, row))

   def _parse_row(self, column_info, row):
       data = row['Data']
       row_output = []
       for j in range(len(data)):
           info = column_info[j]
           datum = data[j]
           row_output.append(self._parse_datum(info, datum))

       return "{%s}" % str(row_output)

   def _parse_datum(self, info, datum):
       if datum.get('NullValue', False):
           return "%s=NULL" % info['Name'],

       column_type = info['Type']

       # If the column is of TimeSeries Type
       if 'TimeSeriesMeasureValueColumnInfo' in column_type:
           return self._parse_time_series(info, datum)

       # If the column is of Array Type
```

```
        elif 'ArrayColumnInfo' in column_type:
            array_values = datum['ArrayValue']
            return "%s=%s" % (info['Name'], self._parse_array(info['Type']
['ArrayColumnInfo'], array_values))

        # If the column is of Row Type
        elif 'RowColumnInfo' in column_type:
            row_column_info = info['Type']['RowColumnInfo']
            row_values = datum['RowValue']
            return self._parse_row(row_column_info, row_values)

        # If the column is of Scalar Type
        else:
            return self._parse_column_name(info) + datum['ScalarValue']

    def _parse_time_series(self, info, datum):
        time_series_output = []
        for data_point in datum['TimeSeriesValue']:
            time_series_output.append("{time=%s, value=%s}"
                                      % (data_point['Time'],
                                         self._parse_datum(info['Type']
['TimeSeriesMeasureValueColumnInfo'],
                                                           data_point['Value'])))
        return "[%s]" % str(time_series_output)

    def _parse_array(self, array_column_info, array_values):
        array_output = []
        for datum in array_values:
            array_output.append(self._parse_datum(array_column_info, datum))

        return "[%s]" % str(array_output)

    @staticmethod
    def _parse_column_name(info):
        if 'Name' in info:
            return info['Name'] + "="
        else:
            return ""
```

Node.js

Il frammento seguente utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
function parseQueryResult(response) {
    const queryStatus = response.QueryStatus;
    console.log("Current query status: " + JSON.stringify(queryStatus));

    const columnInfo = response.ColumnInfo;
    const rows = response.Rows;

    console.log("Metadata: " + JSON.stringify(columnInfo));
    console.log("Data: ");

    rows.forEach(function (row) {
        console.log(parseRow(columnInfo, row));
    });
}

function parseRow(columnInfo, row) {
    const data = row.Data;
    const rowOutput = [];

    var i;
    for ( i = 0; i < data.length; i++ ) {
        info = columnInfo[i];
        datum = data[i];
        rowOutput.push(parseDatum(info, datum));
    }

    return `{${rowOutput.join(", ")}}`
}

function parseDatum(info, datum) {
    if (datum.NullValue != null && datum.NullValue === true) {
        return `${info.Name}=NULL`;
    }

    const columnType = info.Type;

    // If the column is of TimeSeries Type
```

```
    if (columnType.TimeSeriesMeasureValueColumnInfo != null) {
        return parseTimeSeries(info, datum);
    }
    // If the column is of Array Type
    else if (columnType.ArrayColumnInfo != null) {
        const arrayValues = datum.ArrayValue;
        return `${info.Name}=${parseArray(info.Type.ArrayColumnInfo, arrayValues)}`;
    }
    // If the column is of Row Type
    else if (columnType.RowColumnInfo != null) {
        const rowColumnInfo = info.Type.RowColumnInfo;
        const rowValues = datum.RowValue;
        return parseRow(rowColumnInfo, rowValues);
    }
    // If the column is of Scalar Type
    else {
        return parseScalarType(info, datum);
    }
}

function parseTimeSeries(info, datum) {
    const timeSeriesOutput = [];
    datum.TimeSeriesValue.forEach(function (dataPoint) {
        timeSeriesOutput.push(`{time=${dataPoint.Time}, value=
${parseDatum(info.Type.TimeSeriesMeasureValueColumnInfo, dataPoint.Value)}}`)
    });

    return `[${timeSeriesOutput.join(", ")}]`
}

function parseScalarType(info, datum) {
    return parseColumnName(info) + datum.ScalarValue;
}

function parseColumnName(info) {
    return info.Name == null ? "" : `${info.Name}=`;
}

function parseArray(arrayColumnInfo, arrayValues) {
    const arrayOutput = [];
    arrayValues.forEach(function (datum) {
        arrayOutput.push(parseDatum(arrayColumnInfo, datum));
    });
    return `[${arrayOutput.join(", ")}]`
```

```
}
```

.NET

```
        private void ParseQueryResult(QueryResponse response)
        {
            List<ColumnInfo> columnInfo = response.ColumnInfo;
            var options = new JsonSerializerOptions
            {
                IgnoreNullValues = true
            };
            List<String> columnInfoStrings = columnInfo.ConvertAll(x =>
JsonSerializer.Serialize(x, options));
            List<Row> rows = response.Rows;

            QueryStatus queryStatus = response.QueryStatus;
            Console.WriteLine("Current Query status:" +
JsonSerializer.Serialize(queryStatus, options));

            Console.WriteLine("Metadata:" + string.Join(",", columnInfoStrings));
            Console.WriteLine("Data:");

            foreach (Row row in rows)
            {
                Console.WriteLine(ParseRow(columnInfo, row));
            }
        }

        private string ParseRow(List<ColumnInfo> columnInfo, Row row)
        {
            List<Datum> data = row.Data;
            List<string> rowOutput = new List<string>();
            for (int j = 0; j < data.Count; j++)
            {
                ColumnInfo info = columnInfo[j];
                Datum datum = data[j];
                rowOutput.Add(ParseDatum(info, datum));
            }
            return $"{{{string.Join(",", rowOutput)}}}";
        }

        private string ParseDatum(ColumnInfo info, Datum datum)
        {
```

```
            if (datum.NullValue)
            {
                return $"{info.Name}=NULL";
            }

            Amazon.TimestreamQuery.Model.Type columnType = info.Type;
            if (columnType.TimeSeriesMeasureValueColumnInfo != null)
            {
                return ParseTimeSeries(info, datum);
            }
            else if (columnType.ArrayColumnInfo != null)
            {
                List<Datum> arrayValues = datum.ArrayValue;
                return $"{info.Name}={ParseArray(info.Type.ArrayColumnInfo,
arrayValues)}";
            }
            else if (columnType.RowColumnInfo != null &&
columnType.RowColumnInfo.Count > 0)
            {
                List<ColumnInfo> rowColumnInfo = info.Type.RowColumnInfo;
                Row rowValue = datum.RowValue;
                return ParseRow(rowColumnInfo, rowValue);
            }
            else
            {
                return ParseScalarType(info, datum);
            }
        }

        private string ParseTimeSeries(ColumnInfo info, Datum datum)
        {
            var timeseriesString = datum.TimeSeriesValue
                .Select(value => $"{{time={value.Time},
value={ParseDatum(info.Type.TimeSeriesMeasureValueColumnInfo, value.Value)}}}")
                .Aggregate((current, next) => current + "," + next);

            return $"[{timeseriesString}]";
        }

        private string ParseScalarType(ColumnInfo info, Datum datum)
        {
            return ParseColumnName(info) + datum.ScalarValue;
        }
```

```
        private string ParseColumnName(ColumnInfo info)
        {
            return info.Name == null ? "" : (info.Name + "=");
        }

        private string ParseArray(ColumnInfo arrayColumnInfo, List<Datum>
arrayValues)
        {
            return $"[{arrayValues.Select(value => ParseDatum(arrayColumnInfo,
value)).Aggregate((current, next) => current + "," + next)}]";
        }
```

Accesso allo stato della query

È possibile accedere allo stato della query tramite`QueryResponse`, che contiene informazioni sullo stato di avanzamento di una query, sui byte analizzati da una query e sui byte misurati da una query. I `bytesScanned` valori `bytesMetered` and sono cumulativi e vengono aggiornati continuamente durante la paginazione dei risultati delle query. È possibile utilizzare queste informazioni per comprendere i byte analizzati da una singola query e utilizzarle anche per prendere determinate decisioni. Ad esempio, supponendo che il prezzo della query sia di 0,01 USD per GB scansionato, potresti voler annullare le query che superano i 25 USD per query o GB. X Il frammento di codice riportato di seguito mostra come è possibile eseguire questa operazione.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
   private static final long ONE_GB_IN_BYTES = 1073741824L;
   private static final double QUERY_COST_PER_GB_IN_DOLLARS = 0.01; // Assuming the
price of query is $0.01 per GB

   public void cancelQueryBasedOnQueryStatus() {
       System.out.println("Starting query: " + SELECT_ALL_QUERY);
       QueryRequest queryRequest = new QueryRequest();
       queryRequest.setQueryString(SELECT_ALL_QUERY);
```

```
        QueryResult queryResult = queryClient.query(queryRequest);

        while (true) {
            final QueryStatus currentStatusOfQuery = queryResult.getQueryStatus();
            System.out.println("Query progress so far: " +
currentStatusOfQuery.getProgressPercentage() + "%");
            double bytesMeteredSoFar = ((double)
currentStatusOfQuery.getCumulativeBytesMetered() / ONE_GB_IN_BYTES);
            System.out.println("Bytes metered so far: " + bytesMeteredSoFar + "
GB");
            // Cancel query if its costing more than 1 cent
            if (bytesMeteredSoFar * QUERY_COST_PER_GB_IN_DOLLARS > 0.01) {
                cancelQuery(queryResult);
                break;
            }

            if (queryResult.getNextToken() == null) {
                break;
            }
            queryRequest.setNextToken(queryResult.getNextToken());
            queryResult = queryClient.query(queryRequest);
        }
    }
```

Java v2

```
    private static final long ONE_GB_IN_BYTES = 1073741824L;
    private static final double QUERY_COST_PER_GB_IN_DOLLARS = 0.01; // Assuming the
price of query is $0.01 per GB

    public void cancelQueryBasedOnQueryStatus() {
        System.out.println("Starting query: " + SELECT_ALL_QUERY);
        QueryRequest queryRequest =
QueryRequest.builder().queryString(SELECT_ALL_QUERY).build();

        final QueryIterable queryResponseIterator =
timestreamQueryClient.queryPaginator(queryRequest);
        for(QueryResponse queryResponse : queryResponseIterator) {
            final QueryStatus currentStatusOfQuery = queryResponse.queryStatus();
            System.out.println("Query progress so far: " +
currentStatusOfQuery.progressPercentage() + "%");
            double bytesMeteredSoFar = ((double)
currentStatusOfQuery.cumulativeBytesMetered() / ONE_GB_IN_BYTES);
```

```
            System.out.println("Bytes metered so far: " + bytesMeteredSoFar + "GB");
            // Cancel query if its costing more than 1 cent
            if (bytesMeteredSoFar * QUERY_COST_PER_GB_IN_DOLLARS > 0.01) {
                cancelQuery(queryResponse);
                break;
            }
        }
    }
```

Go

```
const OneGbInBytes = 1073741824
// Assuming the price of query is $0.01 per GB
const QueryCostPerGbInDollars = 0.01

func cancelQueryBasedOnQueryStatus(queryPtr *string, querySvc
 *timestreamquery.TimestreamQuery, f *os.File) {
    queryInput := &timestreamquery.QueryInput{
        QueryString: aws.String(*queryPtr),
    }
    fmt.Println("QueryInput:")
    fmt.Println(queryInput)
    // execute the query
    err := querySvc.QueryPages(queryInput,
        func(page *timestreamquery.QueryOutput, lastPage bool) bool {
            // process query response
            queryStatus := page.QueryStatus
            fmt.Println("Current query status:", queryStatus)
            bytes_metered := float64(*queryStatus.CumulativeBytesMetered) /
 float64(ONE_GB_IN_BYTES)
            if bytes_metered * QUERY_COST_PER_GB_IN_DOLLARS > 0.01 {
                cancelQuery(page, querySvc)
                return true
            }
            // query response metadata
            // includes column names and types
            metadata := page.ColumnInfo
            // fmt.Println("Metadata:")
            fmt.Println(metadata)
            header := ""
            for i := 0; i < len(metadata); i++ {
                header += *metadata[i].Name
                if i != len(metadata)-1 {
```

```
                    header += ", "
                }
            }
            write(f, header)

            // query response data
            fmt.Println("Data:")
            // process rows
            rows := page.Rows
            for i := 0; i < len(rows); i++ {
                data := rows[i].Data
                value := processRowType(data, metadata)
                fmt.Println(value)
                write(f, value)
            }
            fmt.Println("Number of rows:", len(page.Rows))
            return true
        })
    if err != nil {
        fmt.Println("Error:")
        fmt.Println(err)
    }
}
```

Python

```
ONE_GB_IN_BYTES = 1073741824
# Assuming the price of query is $0.01 per GB
QUERY_COST_PER_GB_IN_DOLLARS = 0.01

    def cancel_query_based_on_query_status(self):
        try:
            print("Starting query: " + self.SELECT_ALL)
            page_iterator = self.paginator.paginate(QueryString=self.SELECT_ALL)
            for page in page_iterator:
                query_status = page["QueryStatus"]
                progress_percentage = query_status["ProgressPercentage"]
                print("Query progress so far: " + str(progress_percentage) + "%")
                bytes_metered = query_status["CumulativeBytesMetered"] /
 self.ONE_GB_IN_BYTES
                print("Bytes Metered so far: " + str(bytes_metered) + " GB")
                if bytes_metered * self.QUERY_COST_PER_GB_IN_DOLLARS > 0.01:
                    self.cancel_query_for(page)
```

```
                    break
        except Exception as err:
            print("Exception while running query:", err)
            traceback.print_exc(file=sys.stderr)
```

Node.js

Il frammento seguente utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
function parseQueryResult(response) {
    const queryStatus = response.QueryStatus;
    console.log("Current query status: " + JSON.stringify(queryStatus));

    const columnInfo = response.ColumnInfo;
    const rows = response.Rows;

    console.log("Metadata: " + JSON.stringify(columnInfo));
    console.log("Data: ");

    rows.forEach(function (row) {
        console.log(parseRow(columnInfo, row));
    });
}

function parseRow(columnInfo, row) {
    const data = row.Data;
    const rowOutput = [];

    var i;
    for ( i = 0; i < data.length; i++ ) {
        info = columnInfo[i];
        datum = data[i];
        rowOutput.push(parseDatum(info, datum));
    }

    return `{${rowOutput.join(", ")}}`
}

function parseDatum(info, datum) {
    if (datum.NullValue != null && datum.NullValue === true) {
        return `${info.Name}=NULL`;
```

```
    }

    const columnType = info.Type;

    // If the column is of TimeSeries Type
    if (columnType.TimeSeriesMeasureValueColumnInfo != null) {
        return parseTimeSeries(info, datum);
    }
    // If the column is of Array Type
    else if (columnType.ArrayColumnInfo != null) {
        const arrayValues = datum.ArrayValue;
        return `${info.Name}=${parseArray(info.Type.ArrayColumnInfo, arrayValues)}`;
    }
    // If the column is of Row Type
    else if (columnType.RowColumnInfo != null) {
        const rowColumnInfo = info.Type.RowColumnInfo;
        const rowValues = datum.RowValue;
        return parseRow(rowColumnInfo, rowValues);
    }
    // If the column is of Scalar Type
    else {
        return parseScalarType(info, datum);
    }
}

function parseTimeSeries(info, datum) {
    const timeSeriesOutput = [];
    datum.TimeSeriesValue.forEach(function (dataPoint) {
        timeSeriesOutput.push(`{time=${dataPoint.Time}, value=
${parseDatum(info.Type.TimeSeriesMeasureValueColumnInfo, dataPoint.Value)}}`)
    });

    return `[${timeSeriesOutput.join(", ")}]`
}

function parseScalarType(info, datum) {
    return parseColumnName(info) + datum.ScalarValue;
}

function parseColumnName(info) {
    return info.Name == null ? "" : `${info.Name}=`;
}

function parseArray(arrayColumnInfo, arrayValues) {
```

```
    const arrayOutput = [];
    arrayValues.forEach(function (datum) {
        arrayOutput.push(parseDatum(arrayColumnInfo, datum));
    });
    return `[${arrayOutput.join(", ")}]`
}
```

.NET

```
private static readonly long ONE_GB_IN_BYTES = 1073741824L;
private static readonly double QUERY_COST_PER_GB_IN_DOLLARS = 0.01; // Assuming the
 price of query is $0.01 per GB

private async Task CancelQueryBasedOnQueryStatus(string queryString)
{
    try
    {
        QueryRequest queryRequest = new QueryRequest();
        queryRequest.QueryString = queryString;
        QueryResponse queryResponse = await queryClient.QueryAsync(queryRequest);
        while (true)
        {
            QueryStatus queryStatus = queryResponse.QueryStatus;
            double bytesMeteredSoFar = ((double)
 queryStatus.CumulativeBytesMetered / ONE_GB_IN_BYTES);
            // Cancel query if its costing more than 1 cent
            if (bytesMeteredSoFar * QUERY_COST_PER_GB_IN_DOLLARS > 0.01)
            {
                await CancelQuery(queryResponse);
                break;
            }

            ParseQueryResult(queryResponse);
            if (queryResponse.NextToken == null)
            {
                break;
            }
            queryRequest.NextToken = queryResponse.NextToken;
            queryResponse = await queryClient.QueryAsync(queryRequest);
       }
    } catch(Exception e)
    {
```

```
        // Some queries might fail with 500 if the result of a sequence function has
 more than 10000 entries
        Console.WriteLine(e.ToString());
    }
}
```

Per ulteriori dettagli su come annullare una query, consulta. Annulla interrogazione

Esegui UNLOAD interrogazione

I seguenti esempi di codice richiamano una UNLOAD query. Per informazioni su UNLOAD, consulta Utilizzo UNLOAD per esportare i risultati delle query in S3 da Timestream per LiveAnalytics. Per esempi di UNLOAD interrogazioni, vedereEsempio di utilizzo di from Timestream per UNLOAD LiveAnalytics.

Argomenti

- Crea ed esegui un'UNLOADinterrogazione
- Analizza la UNLOAD risposta e ottieni il conteggio delle righe, il collegamento manifesto e il collegamento ai metadati
- Leggi e analizza il contenuto del manifesto
- Leggi e analizza il contenuto dei metadati

Crea ed esegui un'UNLOADinterrogazione

Java

```
// When you have a SELECT like below

String QUERY_1 = "SELECT user_id, ip_address, event, session_id, measure_name, time,
 query, quantity, product_id, channel FROM "
        + DATABASE_NAME + "." + UNLOAD_TABLE_NAME
        + " WHERE time BETWEEN ago(2d) AND now()";

// You can construct UNLOAD query as follows
UnloadQuery unloadQuery = UnloadQuery.builder()
        .selectQuery(QUERY_1)
        .bucketName("timestream-sample-<region>-<accountId>")
        .resultsPrefix("without_partition")
        .format(CSV)
```

```
        .compression(UnloadQuery.Compression.GZIP)
        .build();
QueryResult unloadResult = runQuery(unloadQuery.getUnloadQuery());

// Run UNLOAD query (Similar to how you run SELECT query)
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.pagination
    private QueryResult runQuery(String queryString) {
        QueryResult queryResult = null;
        try {
            QueryRequest queryRequest = new QueryRequest();
            queryRequest.setQueryString(queryString);
            queryResult = queryClient.query(queryRequest);
            while (true) {
                parseQueryResult(queryResult);
                if (queryResult.getNextToken() == null) {
                    break;
                }
                queryRequest.setNextToken(queryResult.getNextToken());
                queryResult = queryClient.query(queryRequest);
            }
        } catch (Exception e) {
            // Some queries might fail with 500 if the result of a sequence function
 has more than 10000 entries
            e.printStackTrace();
        }
        return queryResult;
    }

// Utility that helps to construct UNLOAD query

@Builder
static class UnloadQuery {
    private String selectQuery;
    private String bucketName;
    private String resultsPrefix;
    private Format format;
    private Compression compression;
    private EncryptionType encryptionType;
    private List<String> partitionColumns;
    private String kmsKey;
    private Character csvFieldDelimiter;
    private Character csvEscapeCharacter;
```

```
   public String getUnloadQuery() {
       String destination = constructDestination();
       String withClause = constructOptionalParameters();
       return String.format("UNLOAD (%s) TO '%s' %s", selectQuery, destination,
withClause);
   }

   private String constructDestination() {
       return "s3://" + this.bucketName + "/" + this.resultsPrefix + "/";
   }

   private String constructOptionalParameters() {
       boolean isOptionalParametersPresent = Objects.nonNull(format)
               || Objects.nonNull(compression)
               || Objects.nonNull(encryptionType)
               || Objects.nonNull(partitionColumns)
               || Objects.nonNull(kmsKey)
               || Objects.nonNull(csvFieldDelimiter)
               || Objects.nonNull(csvEscapeCharacter);

       String withClause = "";
       if (isOptionalParametersPresent) {
           StringJoiner optionalParameters = new StringJoiner(",");
           if (Objects.nonNull(format)) {
               optionalParameters.add("format = '" + format + "'");
           }
           if (Objects.nonNull(compression)) {
               optionalParameters.add("compression = '" + compression + "'");
           }
           if (Objects.nonNull(encryptionType)) {
               optionalParameters.add("encryption = '" + encryptionType + "'");
           }
           if (Objects.nonNull(kmsKey)) {
               optionalParameters.add("kms_key = '" + kmsKey + "'");
           }
           if (Objects.nonNull(csvFieldDelimiter)) {
               optionalParameters.add("field_delimiter = '" + csvFieldDelimiter +
"'");
           }
           if (Objects.nonNull(csvEscapeCharacter)) {
               optionalParameters.add("escaped_by = '" + csvEscapeCharacter + "'");
           }
           if (Objects.nonNull(partitionColumns) && !partitionColumns.isEmpty()) {
               final StringJoiner partitionedByList = new StringJoiner(",");
```

```
                partitionColumns.forEach(column -> partitionedByList.add("'" +
 column + "'"));
                optionalParameters.add(String.format("partitioned_by = ARRAY[%s]",
 partitionedByList));
            }
            withClause = String.format("WITH (%s)", optionalParameters);
        }
        return withClause;
    }

    public enum Format {
        CSV, PARQUET
    }

    public enum Compression {
        GZIP, NONE
    }

    public enum EncryptionType {
        SSE_S3, SSE_KMS
    }

    @Override
    public String toString() {
        return getUnloadQuery();
    }
}
```

Java v2

```
// When you have a SELECT like below

String QUERY_1 = "SELECT user_id, ip_address, event, session_id, measure_name, time,
 query, quantity, product_id, channel FROM "
        + DATABASE_NAME + "." + UNLOAD_TABLE_NAME
        + " WHERE time BETWEEN ago(2d) AND now()";

//You can construct UNLOAD query as follows
UnloadQuery unloadQuery = UnloadQuery.builder()
        .selectQuery(QUERY_1)
        .bucketName("timestream-sample-<region>-<accountId>")
        .resultsPrefix("without_partition")
        .format(CSV)
```

```
        .compression(UnloadQuery.Compression.GZIP)
        .build();

QueryResponse unloadResponse = runQuery(unloadQuery.getUnloadQuery());


// Run UNLOAD query (Similar to how you run SELECT query)
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.pagination
private QueryResponse runQuery(String queryString) {
   QueryResponse finalResponse = null;
    try {
        QueryRequest queryRequest =
 QueryRequest.builder().queryString(queryString).build();
        final QueryIterable queryResponseIterator =
 timestreamQueryClient.queryPaginator(queryRequest);
        for(QueryResponse queryResponse : queryResponseIterator) {
            parseQueryResult(queryResponse);
           finalResponse = queryResponse;
        }
    } catch (Exception e) {
        // Some queries might fail with 500 if the result of a sequence function has
 more than 10000 entries
        e.printStackTrace();
    }
   return finalResponse;
}

// Utility that helps to construct UNLOAD query
@Builder
static class UnloadQuery {
    private String selectQuery;
    private String bucketName;
    private String resultsPrefix;
    private Format format;
    private Compression compression;
    private EncryptionType encryptionType;
    private List<String> partitionColumns;
    private String kmsKey;
    private Character csvFieldDelimiter;
    private Character csvEscapeCharacter;

    public String getUnloadQuery() {
        String destination = constructDestination();
```

```
        String withClause = constructOptionalParameters();
        return String.format("UNLOAD (%s) TO '%s' %s", selectQuery, destination,
withClause);
    }

    private String constructDestination() {
        return "s3://" + this.bucketName + "/" + this.resultsPrefix + "/";
    }

    private String constructOptionalParameters() {
        boolean isOptionalParametersPresent = Objects.nonNull(format)
                || Objects.nonNull(compression)
                || Objects.nonNull(encryptionType)
                || Objects.nonNull(partitionColumns)
                || Objects.nonNull(kmsKey)
                || Objects.nonNull(csvFieldDelimiter)
                || Objects.nonNull(csvEscapeCharacter);

        String withClause = "";
        if (isOptionalParametersPresent) {
            StringJoiner optionalParameters = new StringJoiner(",");
            if (Objects.nonNull(format)) {
                optionalParameters.add("format = '" + format + "'");
            }
            if (Objects.nonNull(compression)) {
                optionalParameters.add("compression = '" + compression + "'");
            }
            if (Objects.nonNull(encryptionType)) {
                optionalParameters.add("encryption = '" + encryptionType + "'");
            }
            if (Objects.nonNull(kmsKey)) {
                optionalParameters.add("kms_key = '" + kmsKey + "'");
            }
            if (Objects.nonNull(csvFieldDelimiter)) {
                optionalParameters.add("field_delimiter = '" + csvFieldDelimiter +
"'");
            }
            if (Objects.nonNull(csvEscapeCharacter)) {
                optionalParameters.add("escaped_by = '" + csvEscapeCharacter + "'");
            }
            if (Objects.nonNull(partitionColumns) && !partitionColumns.isEmpty()) {
                final StringJoiner partitionedByList = new StringJoiner(",");
                partitionColumns.forEach(column -> partitionedByList.add("'" +
column + "'"));
```

```
                optionalParameters.add(String.format("partitioned_by = ARRAY[%s]",
 partitionedByList));
            }
            withClause = String.format("WITH (%s)", optionalParameters);
        }
        return withClause;
    }

    public enum Format {
        CSV, PARQUET
    }

    public enum Compression {
        GZIP, NONE
    }

    public enum EncryptionType {
        SSE_S3, SSE_KMS
    }

    @Override
    public String toString() {
        return getUnloadQuery();
    }
}
```

Go

```
// When you have a SELECT like below
var Query = "SELECT user_id, ip_address, event, session_id, measure_name, time,
 query, quantity, product_id, channel FROM "
+ *databaseName + "." + *tableName + " WHERE time BETWEEN ago(2d) AND now()"

// You can construct UNLOAD query as follows
var unloadQuery = UnloadQuery{
    Query: "SELECT user_id, ip_address, session_id, measure_name, time, query,
 quantity, product_id, channel, event FROM " + *databaseName + "." + *tableName +
    " WHERE time BETWEEN ago(2d) AND now()",
    Partitioned_by: []string{},
    Compression: "GZIP",
    Format: "CSV",
    S3Location: bucketName,
    ResultPrefix: "without_partition",
```

```
}


// Run UNLOAD query (Similar to how you run SELECT query)
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.pagination

queryInput := &timestreamquery.QueryInput{
    QueryString: build_query(unloadQuery),
}

err := querySvc.QueryPages(queryInput,
    func(page *timestreamquery.QueryOutput, lastPage bool) bool {
        if (lastPage) {
            var response = parseQueryResult(page)
            var unloadFiles = getManifestAndMetadataFiles(s3Svc, response)
            displayColumns(unloadFiles, unloadQuery.Partitioned_by)
            displayResults(s3Svc, unloadFiles)
        }
        return true
    })

if err != nil {
    fmt.Println("Error:")
    fmt.Println(err)
}

// Utility that helps to construct UNLOAD query
type UnloadQuery struct {
    Query string
    Partitioned_by []string
    Format string
    S3Location string
    ResultPrefix string
    Compression string
}

func build_query(unload_query UnloadQuery) *string {
    var query_results_s3_path = "'s3://" + unload_query.S3Location + "/" +
 unload_query.ResultPrefix + "/'"
    var query = "UNLOAD(" + unload_query.Query + ") TO " + query_results_s3_path + "
 WITH ( "
    if (len(unload_query.Partitioned_by) > 0) {
        query = query + "partitioned_by=ARRAY["
```

```
        for i, column := range unload_query.Partitioned_by {
            if i == 0 {
                query = query + "'" + column + "'"
            } else {
                query = query + ",'" + column + "'"
            }
        }
        query = query + "],"
    }
    query = query + " format='" + unload_query.Format + "', "
    query = query + "  compression='" + unload_query.Compression + "')"
    fmt.Println(query)
    return aws.String(query)
}
```

Python

```
# When you have a SELECT like below
QUERY_1 = "SELECT user_id, ip_address, event, session_id, measure_name, time, query,
 quantity, product_id, channel FROM "
        + database_name + "." + table_name + " WHERE time BETWEEN ago(2d) AND now()"
# You can construct UNLOAD query as follows
UNLOAD_QUERY_1 = UnloadQuery(QUERY_1, "timestream-sample-<region>-<accountId>",
 "without_partition", "CSV", "GZIP", "")

# Run UNLOAD query (Similar to how you run SELECT query)
# https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.pagination
def run_query(self, query_string):
    try:
        page_iterator = self.paginator.paginate(QueryString=UNLOAD_QUERY_1)
    except Exception as err:
        print("Exception while running query:", err)

# Utility that helps to construct UNLOAD query
class UnloadQuery:
    def __init__(self, query, s3_bucket_location, results_prefix, format,
 compression , partition_by):
        self.query = query
        self.s3_bucket_location = s3_bucket_location
        self.results_prefix = results_prefix
        self.format = format
        self.compression = compression
```

```
        self.partition_by = partition_by

    def build_query(self):
        query_results_s3_path = "'s3://" + self.s3_bucket_location + "/" +
 self.results_prefix + "/'"
        unload_query = "UNLOAD("
        unload_query = unload_query + self.query
        unload_query = unload_query + ") "
        unload_query = unload_query + " TO " + query_results_s3_path
        unload_query = unload_query + " WITH ( "

        if(len(self.partition_by) > 0) :
            unload_query = unload_query + " partitioned_by = ARRAY" +
 str(self.partition_by) + ","

        unload_query = unload_query + " format='" + self.format  + "', "
        unload_query = unload_query + "  compression='" + self.compression + "')"

        return unload_query
```

Node.js

```
// When you have a SELECT like below
QUERY_1 = "SELECT user_id, ip_address, event, session_id, measure_name, time, query,
 quantity, product_id, channel FROM "
        + database_name + "." + table_name + " WHERE time BETWEEN ago(2d) AND now()"
// You can construct UNLOAD query as follows
UNLOAD_QUERY_1 = new UnloadQuery(QUERY_1, "timestream-sample-<region>-<accountId>",
 "without_partition", "CSV", "GZIP", "")


// Run UNLOAD query (Similar to how you run SELECT query)
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.pagination

async runQuery(query = UNLOAD_QUERY_1, nextToken) {
    const params = new QueryCommand({
        QueryString: query
    });

    if (nextToken) {
        params.NextToken = nextToken;
    }
```

```
    await queryClient.send(params).then(
            (response) => {
                if (response.NextToken) {
                    runQuery(queryClient, query, response.NextToken);
                } else {
                    await parseAndDisplayResults(response);
                }
            },
            (err) => {
                console.error("Error while querying:", err);
            });
}


class UnloadQuery {
    constructor(query, s3_bucket_location, results_prefix, format, compression ,
 partition_by) {
        this.query = query;
        this.s3_bucket_location = s3_bucket_location
        this.results_prefix = results_prefix
        this.format = format
        this.compression = compression
        this.partition_by = partition_by
    }

    buildQuery() {
        const query_results_s3_path = "'s3://" + this.s3_bucket_location + "/" +
 this.results_prefix + "/'"
        let unload_query = "UNLOAD("
        unload_query = unload_query + this.query
        unload_query = unload_query + ") "
        unload_query = unload_query + " TO " + query_results_s3_path
        unload_query = unload_query + " WITH ( "

        if(this.partition_by.length > 0) {
            let partitionBy = ""
            this.partition_by.forEach((str, i) => {
                partitionBy = partitionBy + (i ? ",'" : "'") + str + "'"
            })
            unload_query = unload_query + " partitioned_by = ARRAY[" + partitionBy +
 "],"
        }
        unload_query = unload_query + " format='" + this.format  + "', "
```

```
        unload_query = unload_query + "  compression='" + this.compression + "')"

        return unload_query
    }
}
```

Analizza la UNLOAD risposta e ottieni il conteggio delle righe, il collegamento manifesto e il collegamento ai metadati

Java

```
// Parsing UNLOAD query response is similar to how you parse SELECT query response:
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.parsing

// But unlike SELECT, UNLOAD only has 1 row * 3 columns outputed
// (rows, metadataFile, manifestFile) => (BIGINT, VARCHAR, VARCHAR)

public UnloadResponse parseResult(QueryResult queryResult) {
    Map<String, String> outputMap = new HashMap<>();
    for (int i = 0; i < queryResult.getColumnInfo().size(); i++) {
        outputMap.put(queryResult.getColumnInfo().get(i).getName(),
                queryResult.getRows().get(0).getData().get(i).getScalarValue());

    }
    return new UnloadResponse(outputMap);
}

@Getter
class UnloadResponse {
    private final String metadataFile;
    private final String manifestFile;
    private final int rows;

    public UnloadResponse(Map<String, String> unloadResponse) {
        this.metadataFile = unloadResponse.get("metadataFile");
        this.manifestFile = unloadResponse.get("manifestFile");
        this.rows = Integer.parseInt(unloadResponse.get("rows"));
    }
}
```

Java v2

```
// Parsing UNLOAD query response is similar to how you parse SELECT query response:
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.parsing

// But unlike SELECT, UNLOAD only has 1 row * 3 columns outputed
// (rows, metadataFile, manifestFile) => (BIGINT, VARCHAR, VARCHAR)

public UnloadResponse parseResult(QueryResponse queryResponse) {
    Map<String, String> outputMap = new HashMap<>();
    for (int i = 0; i < queryResponse.columnInfo().size(); i++) {
        outputMap.put(queryResponse.columnInfo().get(i).name(),
                queryResponse.rows().get(0).data().get(i).scalarValue());

    }
    return new UnloadResponse(outputMap);
}

@Getter
class UnloadResponse {
    private final String metadataFile;
    private final String manifestFile;
    private final int rows;

    public UnloadResponse(Map<String, String> unloadResponse) {
        this.metadataFile = unloadResponse.get("metadataFile");
        this.manifestFile = unloadResponse.get("manifestFile");
        this.rows = Integer.parseInt(unloadResponse.get("rows"));
    }
}
```

Go

```
// Parsing UNLOAD query response is similar to how you parse SELECT query response:
// https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.parsing

// But unlike SELECT, UNLOAD only has 1 row * 3 columns outputed
// (rows, metadataFile, manifestFile) => (BIGINT, VARCHAR, VARCHAR)

func parseQueryResult(queryOutput *timestreamquery.QueryOutput) map[string]string {
    var columnInfo = queryOutput.ColumnInfo;
```

```
    fmt.Println("ColumnInfo", columnInfo)
    fmt.Println("QueryId", queryOutput.QueryId)
    fmt.Println("QueryStatus", queryOutput.QueryStatus)
    return parseResponse(columnInfo, queryOutput.Rows[0])
}

func parseResponse(columnInfo []*timestreamquery.ColumnInfo, row
 *timestreamquery.Row) map[string]string {
    var datum = row.Data
    response := make(map[string]string)
    for i, column := range columnInfo {
        response[*column.Name] = *datum[i].ScalarValue
    }
    return response
}
```

Python

```
# Parsing UNLOAD query response is similar to how you parse SELECT query response:
# https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.parsing

# But unlike SELECT, UNLOAD only has 1 row * 3 columns outputed
# (rows, metadataFile, manifestFile) => (BIGINT, VARCHAR, VARCHAR)

for page in page_iterator:
  last_page = page
response = self._parse_unload_query_result(last_page)

def _parse_unload_query_result(self, query_result):
    column_info = query_result['ColumnInfo']

    print("ColumnInfo: %s" % column_info)
    print("QueryId: %s" % query_result['QueryId'])
    print("QueryStatus:%s" % query_result['QueryStatus'])
    return self.parse_unload_response(column_info, query_result['Rows'][0])

def parse_unload_response(self, column_info, row):
    response = {}
    data = row['Data']
    for i, column in enumerate(column_info):
        response[column['Name']] = data[i]['ScalarValue']
   print("Rows: %s" % response['rows'])
```

```
   print("Metadata File location: %s" % response['metadataFile'])
   print("Manifest File location: %s" % response['manifestFile'])
   return response
```

Node.js

```
# Parsing UNLOAD query response is similar to how you parse SELECT query response:
# https://docs.aws.amazon.com/timestream/latest/developerguide/code-samples.run-
query.html#code-samples.run-query.parsing

# But unlike SELECT, UNLOAD only has 1 row * 3 columns outputed
# (rows, metadataFile, manifestFile) => (BIGINT, VARCHAR, VARCHAR)

async parseAndDisplayResults(data, query) {
    const columnInfo = data['ColumnInfo'];
    console.log("ColumnInfo:", columnInfo)
    console.log("QueryId: %s", data['QueryId'])
    console.log("QueryStatus:", data['QueryStatus'])
    await this.parseResponse(columnInfo, data['Rows'][0], query)
}

async parseResponse(columnInfo, row, query) {
    let response = {}
    const data = row['Data']
    columnInfo.forEach((column, i) => {
        response[column['Name']] = data[i]['ScalarValue']
    })

   console.log("Manifest file", response['manifestFile']);
   console.log("Metadata file", response['metadataFile']);

   return response
}
```

Leggi e analizza il contenuto del manifesto

Java

```
// Read and parse manifest content
public UnloadManifest getUnloadManifest(UnloadResponse unloadResponse) throws
 IOException {
   AmazonS3URI s3URI = new AmazonS3URI(unloadResponse.getManifestFile());
```

```
    S3Object s3Object = s3Client.getObject(s3URI.getBucket(), s3URI.getKey());
    String manifestFileContent = new
 String(IOUtils.toByteArray(s3Object.getObjectContent()), StandardCharsets.UTF_8);
    return new Gson().fromJson(manifestFileContent, UnloadManifest.class);
}

class UnloadManifest {
    @Getter
    public class FileMetadata {
        long content_length_in_bytes;
        long row_count;
    }

    @Getter
    public class ResultFile {
        String url;
        FileMetadata file_metadata;
    }

    @Getter
    public class QueryMetadata {
        long total_content_length_in_bytes;
        long total_row_count;
        String result_format;
        String result_version;
    }

    @Getter
    public class Author {
        String name;
        String manifest_file_version;
    }

    @Getter
    private List<ResultFile> result_files;
    @Getter
    private QueryMetadata query_metadata;
    @Getter
    private Author author;
}
```

Java v2

```
// Read and parse manifest content
public UnloadManifest getUnloadManifest(UnloadResponse unloadResponse) throws
 URISyntaxException {
    // Space needs to encoded to use S3 parseUri function
    S3Uri s3Uri =
 s3Utilities.parseUri(URI.create(unloadResponse.getManifestFile().replace(" ",
 "%20")));
    ResponseBytes<GetObjectResponse> objectBytes =
 s3Client.getObjectAsBytes(GetObjectRequest.builder()
            .bucket(s3Uri.bucket().orElseThrow(() -> new
 URISyntaxException(unloadResponse.getManifestFile(), "Invalid S3 URI")))
            .key(s3Uri.key().orElseThrow(() -> new
 URISyntaxException(unloadResponse.getManifestFile(), "Invalid S3 URI")))
            .build());
    String manifestFileContent = new String(objectBytes.asByteArray(),
 StandardCharsets.UTF_8);
    return new Gson().fromJson(manifestFileContent, UnloadManifest.class);
}

class UnloadManifest {
    @Getter
    public class FileMetadata {
        long content_length_in_bytes;
        long row_count;
    }

    @Getter
    public class ResultFile {
        String url;
        FileMetadata file_metadata;
    }

    @Getter
    public class QueryMetadata {
        long total_content_length_in_bytes;
        long total_row_count;
        String result_format;
        String result_version;
    }

    @Getter
    public class Author {
```

```
        String name;
        String manifest_file_version;
    }

    @Getter
    private List<ResultFile> result_files;
    @Getter
    private QueryMetadata query_metadata;
    @Getter
    private Author author;
}
```

Go

```
// Read and parse manifest content

func getManifestFile(s3Svc *s3.S3, response map[string]string) Manifest {
    var manifestBuf = getObject(s3Svc, response["manifestFile"])
    var manifest Manifest
    json.Unmarshal(manifestBuf.Bytes(), &manifest)
    return manifest
}

func getObject(s3Svc *s3.S3, s3Uri string) *bytes.Buffer {
    u,_ := url.Parse(s3Uri)
    getObjectInput := &s3.GetObjectInput{
        Key:    aws.String(u.Path),
        Bucket: aws.String(u.Host),
    }
    getObjectOutput, err := s3Svc.GetObject(getObjectInput)
    if err != nil {
        fmt.Println("Error: %s\n", err.Error())
    }
    buf := new(bytes.Buffer)
    buf.ReadFrom(getObjectOutput.Body)
    return buf
}

// Unload's Manifest structure

type Manifest struct {
    Author interface{}
    Query_metadata map[string]any
```

```
    Result_files  []struct {
        File_metadata interface{}
        Url string
    }
}}
```

Python

```
def __get_manifest_file(self, response):
    manifest = self.get_object(response['manifestFile']).read().decode('utf-8')
    parsed_manifest = json.loads(manifest)
    print("Manifest contents: \n%s" % parsed_manifest)

def get_object(self, uri):
    try:
        bucket, key = uri.replace("s3://", "").split("/", 1)
        s3_client = boto3.client('s3', region_name=<region>)
        response = s3_client.get_object(Bucket=bucket, Key=key)
        return response['Body']
    except Exception as err:
        print("Failed to get the object for URI:", uri)
        raise err
```

Node.js

```
// Read and parse manifest content

async getManifestFile(response) {
    let manifest;
    await this.getS3Object(response['manifestFile']).then(
        (data) => {
            manifest = JSON.parse(data);
        }
    );
    return manifest;
}

async getS3Object(uri) {
    const {bucketName, key} = this.getBucketAndKey(uri);
    const params = new GetObjectCommand({
        Bucket: bucketName,
        Key: key
    })
```

```
    const response = await this.s3Client.send(params);
    return await response.Body.transformToString();
}

getBucketAndKey(uri) {
    const [bucketName] = uri.replace("s3://", "").split("/", 1);
    const key = uri.replace("s3://", "").split('/').slice(1).join('/');
    return {bucketName, key};
}
```

Leggi e analizza il contenuto dei metadati

Java

```
// Read and parse metadata content
public UnloadMetadata getUnloadMetadata(UnloadResponse unloadResponse) throws
 IOException {
   AmazonS3URI s3URI = new AmazonS3URI(unloadResponse.getMetadataFile());
   S3Object s3Object = s3Client.getObject(s3URI.getBucket(), s3URI.getKey());
    String metadataFileContent = new
 String(IOUtils.toByteArray(s3Object.getObjectContent()), StandardCharsets.UTF_8);
    final Gson gson = new GsonBuilder()
            .setFieldNamingPolicy(FieldNamingPolicy.UPPER_CAMEL_CASE)
            .create();
    return gson.fromJson(metadataFileContent, UnloadMetadata.class);
}

class UnloadMetadata {
    @JsonProperty("ColumnInfo")
    List<ColumnInfo> columnInfo;
    @JsonProperty("Author")
    Author author;

    @Data
    public class Author {
        @JsonProperty("Name")
        String name;
        @JsonProperty("MetadataFileVersion")
        String metadataFileVersion;
    }
}
```

Java v2

```
// Read and parse metadata content

public UnloadMetadata getUnloadMetadata(UnloadResponse unloadResponse) throws
 URISyntaxException {
   // Space needs to encoded to use S3 parseUri function
    S3Uri s3Uri =
 s3Utilities.parseUri(URI.create(unloadResponse.getMetadataFile().replace(" ",
 "%20")));
    ResponseBytes<GetObjectResponse> objectBytes =
 s3Client.getObjectAsBytes(GetObjectRequest.builder()
            .bucket(s3Uri.bucket().orElseThrow(() -> new
 URISyntaxException(unloadResponse.getMetadataFile(), "Invalid S3 URI")))
            .key(s3Uri.key().orElseThrow(() -> new
 URISyntaxException(unloadResponse.getMetadataFile(), "Invalid S3 URI")))
            .build());
    String metadataFileContent = new String(objectBytes.asByteArray(),
 StandardCharsets.UTF_8);
    final Gson gson = new GsonBuilder()
            .setFieldNamingPolicy(FieldNamingPolicy.UPPER_CAMEL_CASE)
            .create();
    return gson.fromJson(metadataFileContent, UnloadMetadata.class);
}

class UnloadMetadata {
    @JsonProperty("ColumnInfo")
    List<ColumnInfo> columnInfo;
    @JsonProperty("Author")
    Author author;

    @Data
    public class Author {
        @JsonProperty("Name")
        String name;
        @JsonProperty("MetadataFileVersion")
        String metadataFileVersion;
    }
}
```

Go

```
// Read and parse metadata content
```

```
func getMetadataFile(s3Svc *s3.S3, response map[string]string) Metadata {
    var metadataBuf = getObject(s3Svc, response["metadataFile"])
    var metadata Metadata
    json.Unmarshal(metadataBuf.Bytes(), &metadata)
    return metadata
}

func getObject(s3Svc *s3.S3, s3Uri string) *bytes.Buffer {
    u,_ := url.Parse(s3Uri)
    getObjectInput := &s3.GetObjectInput{
        Key:    aws.String(u.Path),
        Bucket: aws.String(u.Host),
    }
    getObjectOutput, err := s3Svc.GetObject(getObjectInput)
    if err != nil {
        fmt.Println("Error: %s\n", err.Error())
    }
    buf := new(bytes.Buffer)
    buf.ReadFrom(getObjectOutput.Body)
    return buf
}

// Unload's Metadata structure

type Metadata struct {
    Author interface{}
    ColumnInfo []struct {
        Name string
        Type map[string]string
    }
}
```

Python

```
def __get_metadata_file(self, response):
    metadata = self.get_object(response['metadataFile']).read().decode('utf-8')
    parsed_metadata = json.loads(metadata)
   print("Metadata contents: \n%s" % parsed_metadata)

def get_object(self, uri):
    try:
        bucket, key = uri.replace("s3://", "").split("/", 1)
```

```
        s3_client = boto3.client('s3', region_name=<region>)
        response = s3_client.get_object(Bucket=bucket, Key=key)
        return response['Body']
    except Exception as err:
        print("Failed to get the object for URI:", uri)
        raise err
```

Node.js

```
// Read and parse metadata content
async getMetadataFile(response) {
    let metadata;
    await this.getS3Object(response['metadataFile']).then(
        (data) => {
            metadata = JSON.parse(data);
        }
    );
    return metadata;
}

async getS3Object(uri) {
    const {bucketName, key} = this.getBucketAndKey(uri);
    const params = new GetObjectCommand({
        Bucket: bucketName,
        Key: key
    })
    const response = await this.s3Client.send(params);
    return await response.Body.transformToString();
}

getBucketAndKey(uri) {
    const [bucketName] = uri.replace("s3://", "").split("/", 1);
    const key = uri.replace("s3://", "").split('/').slice(1).join('/');
    return {bucketName, key};
}
```

Annulla interrogazione

È possibile utilizzare i seguenti frammenti di codice per annullare una query.

> **Note**
>
> Questi frammenti di codice si basano su applicazioni di esempio complete su. GitHub Per ulteriori informazioni su come iniziare a utilizzare le applicazioni di esempio, consulta. Applicazione di esempio

Java

```
   public void cancelQuery() {
       System.out.println("Starting query: " + SELECT_ALL_QUERY);
       QueryRequest queryRequest = new QueryRequest();
       queryRequest.setQueryString(SELECT_ALL_QUERY);
       QueryResult queryResult = queryClient.query(queryRequest);

       System.out.println("Cancelling the query: " + SELECT_ALL_QUERY);
       final CancelQueryRequest cancelQueryRequest = new CancelQueryRequest();
       cancelQueryRequest.setQueryId(queryResult.getQueryId());
       try {
           queryClient.cancelQuery(cancelQueryRequest);
           System.out.println("Query has been successfully cancelled");
       } catch (Exception e) {
           System.out.println("Could not cancel the query: " + SELECT_ALL_QUERY + "
= " + e);
       }
   }
```

Java v2

```
   public void cancelQuery() {
       System.out.println("Starting query: " + SELECT_ALL_QUERY);
       QueryRequest queryRequest =
QueryRequest.builder().queryString(SELECT_ALL_QUERY).build();
       QueryResponse queryResponse = timestreamQueryClient.query(queryRequest);

       System.out.println("Cancelling the query: " + SELECT_ALL_QUERY);
       final CancelQueryRequest cancelQueryRequest = CancelQueryRequest.builder()
               .queryId(queryResponse.queryId()).build();
       try {
           timestreamQueryClient.cancelQuery(cancelQueryRequest);
           System.out.println("Query has been successfully cancelled");
       } catch (Exception e) {
```

```
            System.out.println("Could not cancel the query: " + SELECT_ALL_QUERY + "
 = " + e);
        }
    }
```

Go

```
cancelQueryInput := &timestreamquery.CancelQueryInput{
      QueryId: aws.String(*queryOutput.QueryId),
  }

  fmt.Println("Submitting cancellation for the query")
  fmt.Println(cancelQueryInput)

  // submit the query
  cancelQueryOutput, err := querySvc.CancelQuery(cancelQueryInput)

  if err != nil {
      fmt.Println("Error:")
      fmt.Println(err)
  } else {
    fmt.Println("Query has been cancelled successfully")
    fmt.Println(cancelQueryOutput)
  }
```

Python

```
    def cancel_query(self):
        print("Starting query: " + self.SELECT_ALL)
        result = self.client.query(QueryString=self.SELECT_ALL)
        print("Cancelling query: " + self.SELECT_ALL)
        try:
            self.client.cancel_query(QueryId=result['QueryId'])
            print("Query has been successfully cancelled")
        except Exception as err:
            print("Cancelling query failed:", err)
```

Node.js

Il frammento seguente utilizza lo stile AWS SDK for JavaScript V2. Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function tryCancelQuery() {
    const params = {
        QueryString: SELECT_ALL_QUERY
    };
    console.log(`Running query: ${SELECT_ALL_QUERY}`);

    await queryClient.query(params).promise()
        .then(
            async (response) => {
                await cancelQuery(response.QueryId);
            },
            (err) => {
                console.error("Error while executing select all query:", err);
            });
}

async function cancelQuery(queryId) {
    const cancelParams = {
        QueryId: queryId
    };
    console.log(`Sending cancellation for query: ${SELECT_ALL_QUERY}`);
    await queryClient.cancelQuery(cancelParams).promise()
        .then(
            (response) => {
                console.log("Query has been cancelled successfully");
            },
            (err) => {
                console.error("Error while cancelling select all:", err);
            });
}
```

.NET

```
        public async Task CancelQuery()
        {
            Console.WriteLine("Starting query: " + SELECT_ALL_QUERY);
            QueryRequest queryRequest = new QueryRequest();
            queryRequest.QueryString = SELECT_ALL_QUERY;
            QueryResponse queryResponse = await
 queryClient.QueryAsync(queryRequest);

            Console.WriteLine("Cancelling query: " + SELECT_ALL_QUERY);
            CancelQueryRequest cancelQueryRequest = new CancelQueryRequest();
```

```
            cancelQueryRequest.QueryId = queryResponse.QueryId;

            try
            {
                await queryClient.CancelQueryAsync(cancelQueryRequest);
                Console.WriteLine("Query has been successfully cancelled.");
            } catch(Exception e)
            {
                Console.WriteLine("Could not cancel the query: " + SELECT_ALL_QUERY
 + " = " + e);
            }
        }
```

Crea attività di caricamento in batch

È possibile utilizzare i seguenti frammenti di codice per creare attività di caricamento in batch.

Java

```
package com.example.tryit;

import java.util.Arrays;

import
 software.amazon.awssdk.services.timestreamwrite.model.CreateBatchLoadTaskRequest;
import
 software.amazon.awssdk.services.timestreamwrite.model.CreateBatchLoadTaskResponse;
import software.amazon.awssdk.services.timestreamwrite.model.DataModel;
import software.amazon.awssdk.services.timestreamwrite.model.DataModelConfiguration;
import
 software.amazon.awssdk.services.timestreamwrite.model.DataSourceConfiguration;
import
 software.amazon.awssdk.services.timestreamwrite.model.DataSourceS3Configuration;
import software.amazon.awssdk.services.timestreamwrite.model.DimensionMapping;
import
 software.amazon.awssdk.services.timestreamwrite.model.MultiMeasureAttributeMapping;
import software.amazon.awssdk.services.timestreamwrite.model.MultiMeasureMappings;
import software.amazon.awssdk.services.timestreamwrite.model.ReportConfiguration;
import software.amazon.awssdk.services.timestreamwrite.model.ReportS3Configuration;
import software.amazon.awssdk.services.timestreamwrite.model.ScalarMeasureValueType;
import software.amazon.awssdk.services.timestreamwrite.model.TimeUnit;
import software.amazon.awssdk.services.timestreamwrite.TimestreamWriteClient;
```

```
public class BatchLoadExample {
    public static final String DATABASE_NAME = <database name>;
    public static final String TABLE_NAME = <table name>;
    public static final String INPUT_BUCKET = <S3 location>;
    public static final String INPUT_OBJECT_KEY_PREFIX = <CSV filename>;
    public static final String REPORT_BUCKET = <S3 location>;
    public static final long HT_TTL_HOURS = 24L;
    public static final long CT_TTL_DAYS = 7L;

    TimestreamWriteClient amazonTimestreamWrite;

    public BatchLoadExample(TimestreamWriteClient client) {
        this.amazonTimestreamWrite = client;
    }

    public String createBatchLoadTask() {
        System.out.println("Creating batch load task");

        CreateBatchLoadTaskRequest request = CreateBatchLoadTaskRequest.builder()
                .dataModelConfiguration(DataModelConfiguration.builder()
                        .dataModel(DataModel.builder()
                                .timeColumn("timestamp")
                                .timeUnit(TimeUnit.SECONDS)
                                .dimensionMappings(Arrays.asList(
                                        DimensionMapping.builder()
                                                .sourceColumn("vehicle")
                                                .build(),
                                        DimensionMapping.builder()
                                                .sourceColumn("registration")
                                                .destinationColumn("license")
                                                .build()))
                                .multiMeasureMappings(MultiMeasureMappings.builder()
                                        .targetMultiMeasureName("mva_measure_name")

 .multiMeasureAttributeMappings(Arrays.asList(

 MultiMeasureAttributeMapping.builder()
                                                        .sourceColumn("wgt")

 .targetMultiMeasureAttributeName("weight")

 .measureValueType(ScalarMeasureValueType.DOUBLE)
                                                        .build(),
```

```
MultiMeasureAttributeMapping.builder()
                                                        .sourceColumn("spd")

.targetMultiMeasureAttributeName("speed")

.measureValueType(ScalarMeasureValueType.DOUBLE)
                                                        .build(),

MultiMeasureAttributeMapping.builder()
                                                        .sourceColumn("fuel")

.measureValueType(ScalarMeasureValueType.DOUBLE)
                                                        .build(),

MultiMeasureAttributeMapping.builder()
                                                        .sourceColumn("miles")

.measureValueType(ScalarMeasureValueType.DOUBLE)
                                                        .build()))
                                        .build())
                                .build())
                        .build())
                .dataSourceConfiguration(DataSourceConfiguration.builder()
                        .dataSourceS3Configuration(
                                DataSourceS3Configuration.builder()
                                        .bucketName(INPUT_BUCKET)
                                        .objectKeyPrefix(INPUT_OBJECT_KEY_PREFIX)
                                        .build())
                        .dataFormat("CSV")
                        .build())
                .reportConfiguration(ReportConfiguration.builder()
                        .reportS3Configuration(ReportS3Configuration.builder()
                                .bucketName(REPORT_BUCKET)
                                .build())
                        .build())
                .targetDatabaseName(DATABASE_NAME)
                .targetTableName(TABLE_NAME)
                .build();
        try {
            final CreateBatchLoadTaskResponse createBatchLoadTaskResponse =
amazonTimestreamWrite.createBatchLoadTask(request);
            String taskId = createBatchLoadTaskResponse.taskId();
            System.out.println("Successfully created batch load task: " + taskId);
```

```
            return taskId;
        } catch (Exception e) {
            System.out.println("Failed to create batch load task: " + e);
            throw e;
        }
    }
}
```

Go

```
package main

import (
 "fmt"
 "context"
 "log"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/timestreamwrite"
 "github.com/aws/aws-sdk-go-v2/service/timestreamwrite/types"
)

func main() {
 customResolver := aws.EndpointResolverWithOptionsFunc(func(service, region string,
 options ...interface{})(aws.Endpoint, error) {
  if service == timestreamwrite.ServiceID &&  region == "us-west-2" {
      return aws.Endpoint{
          PartitionID:   "aws",
          URL:           <URL>,
          SigningRegion: "us-west-2",
      }, nil
  }
  return aws.Endpoint{}, &  aws.EndpointNotFoundError{}
 })

 cfg, err := config.LoadDefaultConfig(context.TODO(),
 config.WithEndpointResolverWithOptions(customResolver), config.WithRegion("us-
west-2"))

 if err != nil {
    log.Fatalf("failed to load configuration, %v", err)
 }
```

```
client := timestreamwrite.NewFromConfig(cfg)

response, err := client.CreateBatchLoadTask(context.TODO(), &
timestreamwrite.CreateBatchLoadTaskInput{
           TargetDatabaseName: aws.String("BatchLoadExampleDatabase"),
           TargetTableName: aws.String("BatchLoadExampleTable"),
           RecordVersion: aws.Int64(1),
           DataModelConfiguration: &  types.DataModelConfiguration{
               DataModel: &  types.DataModel{
                   TimeColumn: aws.String("timestamp"),
                   TimeUnit: types.TimeUnitMilliseconds,
                   DimensionMappings: []types.DimensionMapping{
                       {
                           SourceColumn: aws.String("registration"),
                           DestinationColumn: aws.String("license"),
                       },
                   },
                   MultiMeasureMappings: &  types.MultiMeasureMappings{
                       TargetMultiMeasureName: aws.String("mva_measure_name"),
                       MultiMeasureAttributeMappings:
[]types.MultiMeasureAttributeMapping{
                           {
                               SourceColumn: aws.String("wgt"),
                               TargetMultiMeasureAttributeName:
aws.String("weight"),
                               MeasureValueType:
types.ScalarMeasureValueTypeDouble,
                           },
                           {
                               SourceColumn: aws.String("spd"),
                               TargetMultiMeasureAttributeName:
aws.String("speed"),
                               MeasureValueType:
types.ScalarMeasureValueTypeDouble,
                           },
                           {
                               SourceColumn: aws.String("fuel_consumption"),
                               TargetMultiMeasureAttributeName: aws.String("fuel"),
                               MeasureValueType:
types.ScalarMeasureValueTypeDouble,
                           },
                       },
                   },
               },
```

```
            },
           DataSourceConfiguration: &  types.DataSourceConfiguration{
                DataSourceS3Configuration: &  types.DataSourceS3Configuration{
                    BucketName: aws.String("test-batch-load-west-2"),
                    ObjectKeyPrefix: aws.String("sample.csv"),
                },
                DataFormat: types.BatchLoadDataFormatCsv,
            },
           ReportConfiguration: & types.ReportConfiguration{
                ReportS3Configuration: & types.ReportS3Configuration{
                    BucketName: aws.String("test-batch-load-report-west-2"),
                    EncryptionOption: types.S3EncryptionOptionSseS3,
                },
            },
 })

 fmt.Println(aws.ToString(response.TaskId))
}
```

Python

```
import boto3
from botocore.config import Config

INGEST_ENDPOINT = "<URL>"
REGION = "us-west-2"
HT_TTL_HOURS = 24
CT_TTL_DAYS = 7
DATABASE_NAME = "<database name>"
TABLE_NAME = "<table name>"
INPUT_BUCKET_NAME = "<S3 location>"
INPUT_OBJECT_KEY_PREFIX = "<CSV file name>"
REPORT_BUCKET_NAME = "<S3 location>"


def create_batch_load_task(client, database_name, table_name, input_bucket_name,
 input_object_key_prefix, report_bucket_name):
    try:
        result = client.create_batch_load_task(TargetDatabaseName=database_name,
 TargetTableName=table_name,
                                               DataModelConfiguration={"DataModel":
 {
                                                   "TimeColumn": "timestamp",
```

```
                                                    "TimeUnit": "SECONDS",
                                                    "DimensionMappings": [
                                                        {
                                                            "SourceColumn": "vehicle"
                                                        },
                                                        {
                                                            "SourceColumn":
"registration",
                                                            "DestinationColumn":
"license"
                                                        }
                                                    ],
                                                    "MultiMeasureMappings": {
                                                        "TargetMultiMeasureName":
"metrics",

"MultiMeasureAttributeMappings": [
                                                            {
                                                                "SourceColumn":
"wgt",
                                                                "MeasureValueType":
"DOUBLE"
                                                            },
                                                            {
                                                                "SourceColumn":
"spd",
                                                                "MeasureValueType":
"DOUBLE"
                                                            },
                                                            {
                                                                "SourceColumn":
"fuel_consumption",

"TargetMultiMeasureAttributeName": "fuel",
                                                                "MeasureValueType":
"DOUBLE"
                                                            },
                                                            {
                                                                "SourceColumn":
"miles",
                                                                "MeasureValueType":
"DOUBLE"
                                                            }
                                                        ]}
```

```
                                                    }
                                                    },
                                                    DataSourceConfiguration={
                                                        "DataSourceS3Configuration": {
                                                            "BucketName":
 input_bucket_name,
                                                            "ObjectKeyPrefix":
 input_object_key_prefix
                                                        },
                                                        "DataFormat": "CSV"
                                                    },
                                                    ReportConfiguration={
                                                        "ReportS3Configuration": {
                                                            "BucketName":
 report_bucket_name,
                                                            "EncryptionOption": "SSE_S3"
                                                        }
                                                    }
                                                    )

        task_id = result["TaskId"]
        print("Successfully created batch load task: ", task_id)
        return task_id
    except Exception as err:
        print("Create batch load task job failed:", err)
        return None


if __name__ == '__main__':
    session = boto3.Session()

    write_client = session.client('timestream-write',
                                  endpoint_url=INGEST_ENDPOINT, region_name=REGION,
                                  config=Config(read_timeout=20,
 max_pool_connections=5000, retries={'max_attempts': 10}))

    task_id = create_batch_load_task(write_client, DATABASE_NAME, TABLE_NAME,
                                     INPUT_BUCKET_NAME, INPUT_OBJECT_KEY_PREFIX,
 REPORT_BUCKET_NAME)
```

Node.js

Il seguente frammento viene utilizzato per la v3. AWS SDK JavaScript Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Per i API dettagli, consulta Class and. CreateBatchLoadCommand CreateBatchLoadTask

```
import { TimestreamWriteClient, CreateBatchLoadTaskCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "us-west-2", endpoint:
 "https://gamma-ingest-cell3.timestream.us-west-2.amazonaws.com" });

const params = {
    TargetDatabaseName: "BatchLoadExampleDatabase",
    TargetTableName: "BatchLoadExampleTable",
 RecordVersion: 1,
 DataModelConfiguration: {
  DataModel: {
   TimeColumn: "timestamp",
            TimeUnit: "MILLISECONDS",
            DimensionMappings: [
                {
                    SourceColumn: "registration",
                    DestinationColumn: "license"
                }
            ],
            MultiMeasureMappings: {
                TargetMultiMeasureName: "mva_measure_name",
                MultiMeasureAttributeMappings: [
                    {
                        SourceColumn: "wgt",
                        TargetMultiMeasureAttributeName: "weight",
                        MeasureValueType: "DOUBLE"
                    },
                    {
                        SourceColumn: "spd",
                        TargetMultiMeasureAttributeName: "speed",
                        MeasureValueType: "DOUBLE"
                    },
                    {
                        SourceColumn: "fuel_consumption",
                        TargetMultiMeasureAttributeName: "fuel",
```

```
                        MeasureValueType: "DOUBLE"
                    }
                ]
            }
        }
    },
 DataSourceConfiguration: {
        DataSourceS3Configuration: {
            BucketName: "test-batch-load-west-2",
            ObjectKeyPrefix: "sample.csv"
        },
        DataFormat: "CSV"
    },
    ReportConfiguration: {
        ReportS3Configuration: {
            BucketName: "test-batch-load-report-west-2",
            EncryptionOption: "SSE_S3"
        }
    }
};

const command = new CreateBatchLoadTaskCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Created batch load task ` + data.TaskId);
} catch (error) {
 console.log("Error creating table. ", error);
 throw error;
}
```

.NET

```
using System;
using System.IO;
using System.Collections.Generic;
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using System.Threading.Tasks;

namespace TimestreamDotNetSample
{
    public class CreateBatchLoadTaskExample
```

```
    {
        public const string DATABASE_NAME = "<database name>";
        public const string TABLE_NAME = "<table name>";
        public const string INPUT_BUCKET = "<input bucket name>";
        public const string INPUT_OBJECT_KEY_PREFIX = "<CSV file name>";
        public const string REPORT_BUCKET = "<report bucket name>";
        public const long HT_TTL_HOURS = 24L;
        public const long CT_TTL_DAYS = 7L;
        private readonly AmazonTimestreamWriteClient writeClient;

        public CreateBatchLoadTaskExample(AmazonTimestreamWriteClient writeClient)
        {
            this.writeClient = writeClient;
        }

        public async Task CreateBatchLoadTask()
        {
            try
            {
                var createBatchLoadTaskRequest = new CreateBatchLoadTaskRequest
                {
                    DataModelConfiguration = new DataModelConfiguration
                    {
                        DataModel = new DataModel
                        {
                            TimeColumn = "timestamp",
                            TimeUnit = TimeUnit.SECONDS,
                            DimensionMappings = new List<DimensionMapping>()
                            {
                                new()
                                {
                                        SourceColumn = "vehicle"
                                },
                                new()
                                {
                                        SourceColumn = "registration",
                                        DestinationColumn = "license"
                                }
                            },
                            MultiMeasureMappings = new MultiMeasureMappings
                            {
                                TargetMultiMeasureName = "mva_measure_name",
                                MultiMeasureAttributeMappings = new
List<MultiMeasureAttributeMapping>()
```

```
                                {
                                        new()
                                        {
                                                SourceColumn = "wgt",
                                                TargetMultiMeasureAttributeName =
"weight",
                                                MeasureValueType =
ScalarMeasureValueType.DOUBLE
                                        },
                                        new()
                                        {
                                                SourceColumn = "spd",
                                                TargetMultiMeasureAttributeName =
"speed",
                                                MeasureValueType =
ScalarMeasureValueType.DOUBLE
                                        },
                                        new()
                                        {
                                                SourceColumn = "fuel",
                                                TargetMultiMeasureAttributeName =
"fuel",
                                                MeasureValueType =
ScalarMeasureValueType.DOUBLE
                                        },
                                        new()
                                        {
                                                SourceColumn = "miles",
                                                TargetMultiMeasureAttributeName =
"miles",
                                                MeasureValueType =
ScalarMeasureValueType.DOUBLE
                                        }
                                }
                            }
                        }
                    },
                    DataSourceConfiguration = new DataSourceConfiguration
                    {
                        DataSourceS3Configuration = new DataSourceS3Configuration
                        {
                            BucketName = INPUT_BUCKET,
                            ObjectKeyPrefix = INPUT_OBJECT_KEY_PREFIX
                        },
```

```
                        DataFormat = "CSV"
                    },
                    ReportConfiguration = new ReportConfiguration
                    {
                        ReportS3Configuration = new ReportS3Configuration
                        {
                            BucketName = REPORT_BUCKET
                        }
                    },
                    TargetDatabaseName = DATABASE_NAME,
                    TargetTableName = TABLE_NAME
                };

                CreateBatchLoadTaskResponse response = await
 writeClient.CreateBatchLoadTaskAsync(createBatchLoadTaskRequest);
                Console.WriteLine($"Task created: " + response.TaskId);
            }
            catch (Exception e)
            {
                Console.WriteLine("Create batch load task failed:" + e.ToString());
            }
        }
    }
}
```

```
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using Amazon;
using Amazon.TimestreamQuery;
using System.Threading.Tasks;
using System;
using CommandLine;
static class Constants
{

}
namespace TimestreamDotNetSample
{
    class MainClass
    {
        public class Options
        {
```

```
        }
        public static void Main(string[] args)
        {
            Parser.Default.ParseArguments<Options>(args)
                .WithParsed<Options>(o => {
                    MainAsync().GetAwaiter().GetResult();
                });
        }

        static async Task MainAsync()
        {
            var writeClientConfig = new AmazonTimestreamWriteConfig
            {
                ServiceURL =  "<service URL>",
                Timeout = TimeSpan.FromSeconds(20),
                MaxErrorRetry = 10
            };

            var writeClient = new AmazonTimestreamWriteClient(writeClientConfig);
            var example = new CreateBatchLoadTaskExample(writeClient);
            await example.CreateBatchLoadTask();
        }
    }
}
```

Descrizione dell'attività di caricamento in batch

È possibile utilizzare i seguenti frammenti di codice per descrivere le attività di caricamento in batch.

Java

```
   public void describeBatchLoadTask(String taskId) {
           final DescribeBatchLoadTaskResponse batchLoadTaskResponse =
 amazonTimestreamWrite

 .describeBatchLoadTask(DescribeBatchLoadTaskRequest.builder()
                                          .taskId(taskId)
                                          .build());

           System.out.println("Task id: " +
 batchLoadTaskResponse.batchLoadTaskDescription().taskId());
```

```
            System.out.println("Status: " +
 batchLoadTaskResponse.batchLoadTaskDescription().taskStatusAsString());
            System.out.println("Records processed: "
                               +
 batchLoadTaskResponse.batchLoadTaskDescription().progressReport().recordsProcessed());
    }
```

Go

```
package main

import (
 "fmt"
 "context"
 "log"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/timestreamwrite"
)

func main() {
 customResolver := aws.EndpointResolverWithOptionsFunc(func(service, region string,
 options ...interface{}) (aws.Endpoint, error) {
  if service == timestreamwrite.ServiceID && region == "us-west-2" {
      return aws.Endpoint{
          PartitionID:   "aws",
          URL:           <URL>,
          SigningRegion: "us-west-2",
      }, nil
  }
  return aws.Endpoint{}, &aws.EndpointNotFoundError{}
 })

 cfg, err := config.LoadDefaultConfig(context.TODO(),
 config.WithEndpointResolverWithOptions(customResolver), config.WithRegion("us-
west-2"))

 if err != nil {
    log.Fatalf("failed to load configuration, %v", err)
 }

 client := timestreamwrite.NewFromConfig(cfg)
```

```
 response, err := client.DescribeBatchLoadTask(context.TODO(),
 &timestreamwrite.DescribeBatchLoadTaskInput{
  TaskId: aws.String("<TaskId>"),
 })

 fmt.Println(aws.ToString(response.BatchLoadTaskDescription.TaskId))
}
```

Python

```
import boto3
from botocore.config import Config

INGEST_ENDPOINT="<url>"
REGION="us-west-2"
HT_TTL_HOURS = 24
CT_TTL_DAYS = 7
TASK_ID = "<task id>"

def describe_batch_load_task(client, task_id):
    try:
        result = client.describe_batch_load_task(TaskId=task_id)
        print("Successfully described batch load task: ", result)
    except Exception as err:
        print("Describe batch load task job failed:", err)


if __name__ == '__main__':
    session = boto3.Session()

    write_client = session.client('timestream-write', \
        endpoint_url=INGEST_ENDPOINT, region_name=REGION, \
        config=Config(read_timeout=20, max_pool_connections = 5000,
 retries={'max_attempts': 10}))

    describe_batch_load_task(write_client, TASK_ID)
```

Node.js

Il seguente frammento di codice viene utilizzato per la versione 3. AWS SDK JavaScript Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Per i API dettagli, consulta Class and. DescribeBatchLoadCommand DescribeBatchLoadTask

```
import { TimestreamWriteClient, DescribeBatchLoadTaskCommand } from "@aws-sdk/
client-timestream-write";
const writeClient = new TimestreamWriteClient({ region: "<region>", endpoint:
 "<endpoint>" });

const params = {
    TaskId: "<TaskId>"
};

const command = new DescribeBatchLoadTaskCommand(params);

try {
    const data = await writeClient.send(command);
    console.log(`Batch load task has id ` + data.BatchLoadTaskDescription.TaskId);
} catch (error) {
    if (error.code === 'ResourceNotFoundException') {
        console.log("Batch load task doesn't exist.");
    } else {
        console.log("Describe batch load task failed.", error);
        throw error;
    }
}
```

.NET

```
using System;
using System.IO;
using System.Collections.Generic;
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using System.Threading.Tasks;

namespace TimestreamDotNetSample
{
    public class DescribeBatchLoadTaskExample
    {
        private readonly AmazonTimestreamWriteClient writeClient;

        public DescribeBatchLoadTaskExample(AmazonTimestreamWriteClient writeClient)
        {
            this.writeClient = writeClient;
```

```
        }

        public async Task DescribeBatchLoadTask(String taskId)
        {
            try
            {
                var describeBatchLoadTaskRequest = new DescribeBatchLoadTaskRequest
                {
                    TaskId = taskId
                };
                DescribeBatchLoadTaskResponse response = await
 writeClient.DescribeBatchLoadTaskAsync(describeBatchLoadTaskRequest);
                Console.WriteLine($"Task has id:
{response.BatchLoadTaskDescription.TaskId}");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine("Batch load task does not exist.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Describe batch load task failed:" +
 e.ToString());
            }
        }
    }
}
```

```
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using Amazon;
using Amazon.TimestreamQuery;
using System.Threading.Tasks;
using System;
using CommandLine;
static class Constants
{

}
namespace TimestreamDotNetSample
{
    class MainClass
    {
```

```
        public class Options
        {

        }
        public static void Main(string[] args)
        {
            Parser.Default.ParseArguments<Options>(args)
                .WithParsed<Options>(o => {
                    MainAsync().GetAwaiter().GetResult();
                });
        }

        static async Task MainAsync()
        {
            var writeClientConfig = new AmazonTimestreamWriteConfig
            {
                ServiceURL =  "<service URL>",
                Timeout = TimeSpan.FromSeconds(20),
                MaxErrorRetry = 10
            };

            var writeClient = new AmazonTimestreamWriteClient(writeClientConfig);
            var example = new DescribeBatchLoadTaskExample(writeClient);
            await example.DescribeBatchLoadTask("<batch load task id>");
        }
    }
}
```

Elenca le attività di caricamento in batch

È possibile utilizzare i seguenti frammenti di codice per elencare le attività di caricamento in batch.

Java

```
    public void listBatchLoadTasks() {
            final ListBatchLoadTasksResponse listBatchLoadTasksResponse =
 amazonTimestreamWrite
                            .listBatchLoadTasks(ListBatchLoadTasksRequest.builder()
                                    .maxResults(15)
                                    .build());
```

```
            for (BatchLoadTask batchLoadTask :
 listBatchLoadTasksResponse.batchLoadTasks()) {
                    System.out.println(batchLoadTask.taskId());
            }
    }
```

Go

```
package main

import (
 "fmt"
 "context"
 "log"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/timestreamwrite"
)

func main() {
 customResolver := aws.EndpointResolverWithOptionsFunc(func(service, region string,
 options ...interface{}) (aws.Endpoint, error) {
  if service == timestreamwrite.ServiceID && region == "us-west-2" {
      return aws.Endpoint{
          PartitionID:   "aws",
          URL:           <URL>,
          SigningRegion: "us-west-2",
      }, nil
  }
  return aws.Endpoint{}, &aws.EndpointNotFoundError{}
 })

 cfg, err := config.LoadDefaultConfig(context.TODO(),
 config.WithEndpointResolverWithOptions(customResolver), config.WithRegion("us-
west-2"))

 if err != nil {
    log.Fatalf("failed to load configuration, %v", err)
 }

 client := timestreamwrite.NewFromConfig(cfg)
 listBatchLoadTasksMaxResult := int32(15)
```

```
 response, err := client.ListBatchLoadTasks(context.TODO(),
 &timestreamwrite.ListBatchLoadTasksInput{
  MaxResults: &listBatchLoadTasksMaxResult,
 })

 for i, task := range response.BatchLoadTasks {
  fmt.Println(i, aws.ToString(task.TaskId))
 }
}
```

Python

```
import boto3
from botocore.config import Config

INGEST_ENDPOINT = "<url>"
REGION = "us-west-2"
HT_TTL_HOURS = 24
CT_TTL_DAYS = 7


def print_batch_load_tasks(batch_load_tasks):
    for batch_load_task in batch_load_tasks:
        print(batch_load_task['TaskId'])


def list_batch_load_tasks(client):
    print("\nListing batch load tasks")
    try:
        response = client.list_batch_load_tasks(MaxResults=10)
        print_batch_load_tasks(response['BatchLoadTasks'])
        next_token = response.get('NextToken', None)
        while next_token:
            response = client.list_batch_load_tasks(
                NextToken=next_token, MaxResults=10)
            print_batch_load_tasks(response['BatchLoadTasks'])
            next_token = response.get('NextToken', None)
    except Exception as err:
        print("List batch load tasks failed:", err)
        raise err


if __name__ == '__main__':
```

```
    session = boto3.Session()

    write_client = session.client('timestream-write',
                                  endpoint_url=INGEST_ENDPOINT, region_name=REGION,
                                  config=Config(read_timeout=20,
max_pool_connections=5000, retries={'max_attempts': 10}))

    list_batch_load_tasks(write_client)
```

Node.js

Il seguente frammento viene utilizzato per la v3. AWS SDK JavaScript Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Per i API dettagli, consulta Class and. DescribeBatchLoadCommand DescribeBatchLoadTask

```
import { TimestreamWriteClient, ListBatchLoadTasksCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "<region>", endpoint:
 "<endpoint>" });

const params = {
    MaxResults: <15>
};

const command = new ListBatchLoadTasksCommand(params);

getBatchLoadTasksList(null);

async function getBatchLoadTasksList(nextToken) {
    if (nextToken) {
        params.NextToken = nextToken;
    }

    try {
        const data = await writeClient.send(command);

        data.BatchLoadTasks.forEach(function (task) {
            console.log(task.TaskId);
        });

        if (data.NextToken) {
```

```
            return getBatchLoadTasksList(data.NextToken);
        }
    } catch (error) {
        console.log("Error while listing batch load tasks", error);
    }
}
```

.NET

```
using System;
using System.IO;
using System.Collections.Generic;
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using System.Threading.Tasks;

namespace TimestreamDotNetSample
{
    public class ListBatchLoadTasksExample
    {
        private readonly AmazonTimestreamWriteClient writeClient;

        public ListBatchLoadTasksExample(AmazonTimestreamWriteClient writeClient)
        {
            this.writeClient = writeClient;
        }

        public async Task ListBatchLoadTasks()
        {
            Console.WriteLine("Listing batch load tasks");

            try
            {
                var listBatchLoadTasksRequest = new ListBatchLoadTasksRequest
                {
                    MaxResults = 15
                };

                ListBatchLoadTasksResponse response = await
 writeClient.ListBatchLoadTasksAsync(listBatchLoadTasksRequest);

                PrintBatchLoadTasks(response.BatchLoadTasks);
                var nextToken = response.NextToken;
```

```
                while (nextToken != null)
                {
                    listBatchLoadTasksRequest.NextToken = nextToken;
                    response = await
 writeClient.ListBatchLoadTasksAsync(listBatchLoadTasksRequest);
                    PrintBatchLoadTasks(response.BatchLoadTasks);
                    nextToken = response.NextToken;
                }
            }
            catch (Exception e)
            {
                Console.WriteLine("List batch load tasks failed:" + e.ToString());
            }
        }

        private void PrintBatchLoadTasks(List<BatchLoadTask> tasks)
        {
            foreach (BatchLoadTask task in tasks)
                Console.WriteLine($"Task:{task.TaskId}");
        }
    }
}
```

```
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using Amazon;
using Amazon.TimestreamQuery;
using System.Threading.Tasks;
using System;
using CommandLine;
static class Constants
{

}
namespace TimestreamDotNetSample
{
    class MainClass
    {
        public class Options
        {

        }
```

```
        public static void Main(string[] args)
        {
            Parser.Default.ParseArguments<Options>(args)
                .WithParsed<Options>(o => {
                    MainAsync().GetAwaiter().GetResult();
                });
        }

        static async Task MainAsync()
        {
            var writeClientConfig = new AmazonTimestreamWriteConfig
            {
                ServiceURL =  "<service URL>",
                Timeout = TimeSpan.FromSeconds(20),
                MaxErrorRetry = 10
            };

            var writeClient = new AmazonTimestreamWriteClient(writeClientConfig);
            var example = new ListBatchLoadTasksExample(writeClient);
            await example.ListBatchLoadTasks();
        }
    }
}
```

Riprendi l'operazione di caricamento in batch

È possibile utilizzare i seguenti frammenti di codice per riprendere le attività di caricamento in batch.

Java

```
    public void resumeBatchLoadTask(String taskId) {
            try {
                    amazonTimestreamWrite

 .resumeBatchLoadTask(ResumeBatchLoadTaskRequest.builder()
                                                    .taskId(taskId)
                                                    .build());

                    System.out.println("Successfully resumed batch load task.");
            } catch (ValidationException validationException) {
                    System.out.println(validationException.getMessage());
            }
```

```
    }
```

Go

```
package main

import (
 "fmt"
 "context"
 "log"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/timestreamwrite"
)

func main() {
 customResolver := aws.EndpointResolverWithOptionsFunc(func(service, region string,
 options ...interface{}) (aws.Endpoint, error) {
  if service == timestreamwrite.ServiceID && region == "us-west-2" {
      return aws.Endpoint{
          PartitionID:   "aws",
          URL:           <URL>,
          SigningRegion: "us-west-2",
      }, nil
  }
  return aws.Endpoint{}, &aws.EndpointNotFoundError{}
 })

 cfg, err := config.LoadDefaultConfig(context.TODO(),
 config.WithEndpointResolverWithOptions(customResolver), config.WithRegion("us-
west-2"))

 if err != nil {
    log.Fatalf("failed to load configuration, %v", err)
 }

 client := timestreamwrite.NewFromConfig(cfg)

 response, err := client.ResumeBatchLoadTask(context.TODO(),
 &timestreamwrite.ResumeBatchLoadTaskInput{
  TaskId: aws.String("TaskId"),
 })
```

```
 if err != nil {
  fmt.Println("Error:")
  fmt.Println(err)
 } else {
  fmt.Println("Resume batch load task is successful")
  fmt.Println(response)
 }
}
```

Python

```
import boto3
from botocore.config import Config

INGEST_ENDPOINT="<url>"
REGION="us-west-2"
HT_TTL_HOURS = 24
CT_TTL_DAYS = 7
TASK_ID = "<TaskId>"

def resume_batch_load_task(client, task_id):
    try:
        result = client.resume_batch_load_task(TaskId=task_id)
        print("Successfully resumed batch load task: ", result)
    except Exception as err:
        print("Resume batch load task failed:", err)


if __name__ == '__main__':
    session = boto3.Session()

    write_client = session.client('timestream-write', \
        endpoint_url=INGEST_ENDPOINT, region_name=REGION, \
        config=Config(read_timeout=20, max_pool_connections = 5000,
 retries={'max_attempts': 10}))

    resume_batch_load_task(write_client, TASK_ID)
```

Node.js

Il seguente frammento viene utilizzato per la v3. AWS SDK JavaScript Per ulteriori informazioni su come installare il client e sull'utilizzo, consulta Timestream Write Client - for v3. AWS SDK JavaScript

Per i API dettagli, consulta Class and. CreateBatchLoadCommand CreateBatchLoadTask

```
import { TimestreamWriteClient, ResumeBatchLoadTaskCommand } from "@aws-sdk/client-
timestream-write";
const writeClient = new TimestreamWriteClient({ region: "<region>", endpoint:
 "<endpoint>" });

const params = {
    TaskId: "<TaskId>"
};

const command = new ResumeBatchLoadTaskCommand(params);

try {
    const data = await writeClient.send(command);
    console.log("Resumed batch load task");
} catch (error) {
    console.log("Resume batch load task failed.", error);
 throw error;
}
```

.NET

```
using System;
using System.IO;
using System.Collections.Generic;
using Amazon.TimestreamWrite;
using Amazon.TimestreamWrite.Model;
using System.Threading.Tasks;

namespace TimestreamDotNetSample
{
    public class ResumeBatchLoadTaskExample
    {
        private readonly AmazonTimestreamWriteClient writeClient;

        public ResumeBatchLoadTaskExample(AmazonTimestreamWriteClient writeClient)
        {
            this.writeClient = writeClient;
        }

        public async Task ResumeBatchLoadTask(String taskId)
        {
```

```
            try
            {
                var resumeBatchLoadTaskRequest = new ResumeBatchLoadTaskRequest
                {
                    TaskId = taskId
                };
                ResumeBatchLoadTaskResponse response = await
 writeClient.ResumeBatchLoadTaskAsync(resumeBatchLoadTaskRequest);
                Console.WriteLine("Successfully resumed batch load task.");
            }
            catch (ResourceNotFoundException)
            {
                Console.WriteLine("Batch load task does not exist.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Resume batch load task failed: " + e.ToString());
            }
        }
    }
}
```

Crea una query pianificata

È possibile utilizzare i seguenti frammenti di codice per creare un'interrogazione pianificata con mappatura multipisura.

Java

```
public static String DATABASE_NAME = "devops_sample_application";
public static String TABLE_NAME = "host_metrics_sample_application";
public static String HOSTNAME = "host-24Gju";
public static String SQ_NAME = "daily-sample";
public static String SCHEDULE_EXPRESSION = "cron(0/2 * * * ? *)";

// Find the average, p90, p95, and p99 CPU utilization for a specific EC2 host over
 the past 2 hours.
public static String QUERY = "SELECT region, az, hostname, BIN(time, 15s) AS
 binned_timestamp, " +
"ROUND(AVG(cpu_utilization), 2) AS avg_cpu_utilization, " +
"ROUND(APPROX_PERCENTILE(cpu_utilization, 0.9), 2) AS p90_cpu_utilization, " +
"ROUND(APPROX_PERCENTILE(cpu_utilization, 0.95), 2) AS p95_cpu_utilization, " +
```

```
"ROUND(APPROX_PERCENTILE(cpu_utilization, 0.99), 2) AS p99_cpu_utilization " +
"FROM " +  DATABASE_NAME + "." +  TABLE_NAME + " " +
"WHERE measure_name = 'metrics' " +
"AND hostname = '" + HOSTNAME + "' " +
"AND time > ago(2h) " +
"GROUP BY region, hostname, az, BIN(time, 15s) " +
"ORDER BY binned_timestamp ASC " +
"LIMIT 5";


public String createScheduledQuery(String topic_arn,
    String role_arn,
    String database_name,
    String table_name) {
    System.out.println("Creating Scheduled Query");

    List<Pair<String, MeasureValueType>> sourceColToMeasureValueTypes =
 Arrays.asList(
        Pair.of("avg_cpu_utilization", DOUBLE),
        Pair.of("p90_cpu_utilization", DOUBLE),
        Pair.of("p95_cpu_utilization", DOUBLE),
        Pair.of("p99_cpu_utilization", DOUBLE));

    CreateScheduledQueryRequest createScheduledQueryRequest = new
 CreateScheduledQueryRequest()
            .withName(SQ_NAME)
            .withQueryString(QUERY)
            .withScheduleConfiguration(new ScheduleConfiguration()
                    .withScheduleExpression(SCHEDULE_EXPRESSION))
            .withNotificationConfiguration(new NotificationConfiguration()
                    .withSnsConfiguration(new SnsConfiguration()
                            .withTopicArn(topic_arn)))
            .withTargetConfiguration(new
 TargetConfiguration().withTimestreamConfiguration(new TimestreamConfiguration()
                    .withDatabaseName(database_name)
                    .withTableName(table_name)
                    .withTimeColumn("binned_timestamp")
                    .withDimensionMappings(Arrays.asList(
                            new DimensionMapping()
                                    .withName("region")
                                    .withDimensionValueType("VARCHAR"),
                            new DimensionMapping()
                                    .withName("az")
                                    .withDimensionValueType("VARCHAR"),
```

```
                            new DimensionMapping()
                                    .withName("hostname")
                                    .withDimensionValueType("VARCHAR")
                    ))
                    .withMultiMeasureMappings(new MultiMeasureMappings()
                        .withTargetMultiMeasureName("multi-metrics")
                        .withMultiMeasureAttributeMappings(
                            sourceColToMeasureValueTypes.stream()
                            .map(pair -> new MultiMeasureAttributeMapping()
                                .withMeasureValueType(pair.getValue().name())
                                .withSourceColumn(pair.getKey()))
                            .collect(Collectors.toList()))))
            .withErrorReportConfiguration(new ErrorReportConfiguration()
                    .withS3Configuration(new S3Configuration()

 .withBucketName(timestreamDependencyHelper.getS3ErrorReportBucketName())))
            .withScheduledQueryExecutionRoleArn(role_arn);

    try {
        final CreateScheduledQueryResult createScheduledQueryResult =
 queryClient.createScheduledQuery(createScheduledQueryRequest);
        final String scheduledQueryArn = createScheduledQueryResult.getArn();
        System.out.println("Successfully created scheduled query : " +
 scheduledQueryArn);
        return scheduledQueryArn;
    }
    catch (Exception e) {
        System.out.println("Scheduled Query creation failed: " + e);
        throw e;
    }
}
```

Java v2

```
public static String DATABASE_NAME = "testJavaV2DB";
public static String TABLE_NAME = "testJavaV2Table";
public static String HOSTNAME = "host-24Gju";
public static String SQ_NAME = "daily-sample";
public static String SCHEDULE_EXPRESSION = "cron(0/2 * * * ? *)";

// Find the average, p90, p95, and p99 CPU utilization for a specific EC2 host over
 the past 2 hours.
```

```
public static String VALID_QUERY = "SELECT region, az, hostname, BIN(time, 15s) AS
 binned_timestamp, " +
"ROUND(AVG(cpu_utilization), 2) AS avg_cpu_utilization, " +
"ROUND(APPROX_PERCENTILE(cpu_utilization, 0.9), 2) AS p90_cpu_utilization, " +
"ROUND(APPROX_PERCENTILE(cpu_utilization, 0.95), 2) AS p95_cpu_utilization, " +
"ROUND(APPROX_PERCENTILE(cpu_utilization, 0.99), 2) AS p99_cpu_utilization " +
"FROM " +  DATABASE_NAME + "." +  TABLE_NAME + " " +
"WHERE measure_name = 'metrics' " +
"AND hostname = '" + HOSTNAME + "' " +
"AND time > ago(2h) " +
"GROUP BY region, hostname, az, BIN(time, 15s) " +
"ORDER BY binned_timestamp ASC " +
"LIMIT 5";


private String createScheduledQueryHelper(String topicArn, String roleArn,
        String s3ErrorReportBucketName, String query,
        TargetConfiguration targetConfiguration) {
    System.out.println("Creating Scheduled Query");

    CreateScheduledQueryRequest createScheduledQueryRequest =
 CreateScheduledQueryRequest.builder()
            .name(SQ_NAME)
            .queryString(query)
            .scheduleConfiguration(ScheduleConfiguration.builder()
                    .scheduleExpression(SCHEDULE_EXPRESSION)
                    .build())
            .notificationConfiguration(NotificationConfiguration.builder()
                    .snsConfiguration(SnsConfiguration.builder()
                            .topicArn(topicArn)
                            .build())
                    .build())
            .targetConfiguration(targetConfiguration)
            .errorReportConfiguration(ErrorReportConfiguration.builder()
                    .s3Configuration(S3Configuration.builder()
                            .bucketName(s3ErrorReportBucketName)
                            .objectKeyPrefix(SCHEDULED_QUERY_EXAMPLE)
                            .build())
                    .build())
            .scheduledQueryExecutionRoleArn(roleArn)
            .build();

    try {
```

```
        final CreateScheduledQueryResponse response =
 queryClient.createScheduledQuery(createScheduledQueryRequest);
        final String scheduledQueryArn = response.arn();
        System.out.println("Successfully created scheduled query : " +
 scheduledQueryArn);
        return scheduledQueryArn;
    }
    catch (Exception e) {
        System.out.println("Scheduled Query creation failed: " + e);
        throw e;
    }
}

public String createScheduledQuery(String topicArn, String roleArn,
        String databaseName, String tableName, String s3ErrorReportBucketName) {
    List<Pair<String, MeasureValueType>> sourceColToMeasureValueTypes =
 Arrays.asList(
            Pair.of("avg_cpu_utilization", DOUBLE),
            Pair.of("p90_cpu_utilization", DOUBLE),
            Pair.of("p95_cpu_utilization", DOUBLE),
            Pair.of("p99_cpu_utilization", DOUBLE));

    TargetConfiguration targetConfiguration = TargetConfiguration.builder()
            .timestreamConfiguration(TimestreamConfiguration.builder()
            .databaseName(databaseName)
            .tableName(tableName)
            .timeColumn("binned_timestamp")
            .dimensionMappings(Arrays.asList(
                    DimensionMapping.builder()
                            .name("region")
                            .dimensionValueType("VARCHAR")
                            .build(),
                    DimensionMapping.builder()
                            .name("az")
                            .dimensionValueType("VARCHAR")
                            .build(),
                    DimensionMapping.builder()
                            .name("hostname")
                            .dimensionValueType("VARCHAR")
                            .build()
            ))
            .multiMeasureMappings(MultiMeasureMappings.builder()
                    .targetMultiMeasureName("multi-metrics")
                    .multiMeasureAttributeMappings(
```

```
                            sourceColToMeasureValueTypes.stream()
                                    .map(pair ->
 MultiMeasureAttributeMapping.builder()

 .measureValueType(pair.getValue().name())
                                            .sourceColumn(pair.getKey())
                                            .build())
                                    .collect(Collectors.toList()))
                    .build())
            .build())
            .build();

    return createScheduledQueryHelper(topicArn, roleArn, s3ErrorReportBucketName,
 VALID_QUERY, targetConfiguration);
}}
```

Go

```
SQ_ERROR_CONFIGURATION_S3_BUCKET_NAME_PREFIX = "sq-error-configuration-sample-s3-
bucket-"
HOSTNAME            = "host-24Gju"
SQ_NAME             = "daily-sample"
SCHEDULE_EXPRESSION = "cron(0/1 * * * ? *)"
QUERY               = "SELECT region, az, hostname, BIN(time, 15s) AS
 binned_timestamp, " +
    "ROUND(AVG(cpu_utilization), 2) AS avg_cpu_utilization, " +
    "ROUND(APPROX_PERCENTILE(cpu_utilization, 0.9), 2) AS p90_cpu_utilization, " +
    "ROUND(APPROX_PERCENTILE(cpu_utilization, 0.95), 2) AS p95_cpu_utilization, " +
    "ROUND(APPROX_PERCENTILE(cpu_utilization, 0.99), 2) AS p99_cpu_utilization " +
    "FROM %s.%s " +
    "WHERE measure_name = 'metrics' " +
    "AND hostname = '" + HOSTNAME + "' " +
    "AND time > ago(2h) " +
    "GROUP BY region, hostname, az, BIN(time, 15s) " +
    "ORDER BY binned_timestamp ASC " +
    "LIMIT 5"
s3BucketName = utils.SQ_ERROR_CONFIGURATION_S3_BUCKET_NAME_PREFIX +
 generateRandomStringWithSize(5)

func generateRandomStringWithSize(size int) string {
     rand.Seed(time.Now().UnixNano())
     alphaNumericList := []rune("abcdefghijklmnopqrstuvwxyz0123456789")
     randomPrefix := make([]rune, size)
```

```
     for i := range randomPrefix {
         randomPrefix[i] = alphaNumericList[rand.Intn(len(alphaNumericList))]
     }
     return string(randomPrefix)
 }

func (timestreamBuilder TimestreamBuilder) createScheduledQuery(topicArn string,
 roleArn string, s3ErrorReportBucketName string,
query string, targetConfiguration timestreamquery.TargetConfiguration) (string,
 error) {

createScheduledQueryInput := &timestreamquery.CreateScheduledQueryInput{
    Name:        aws.String(SQ_NAME),
    QueryString: aws.String(query),
    ScheduleConfiguration: &timestreamquery.ScheduleConfiguration{
        ScheduleExpression: aws.String(SCHEDULE_EXPRESSION),
    },
    NotificationConfiguration: &timestreamquery.NotificationConfiguration{
        SnsConfiguration: &timestreamquery.SnsConfiguration{
            TopicArn: aws.String(topicArn),
        },
    },
    TargetConfiguration: &targetConfiguration,
    ErrorReportConfiguration: &timestreamquery.ErrorReportConfiguration{
        S3Configuration: &timestreamquery.S3Configuration{
            BucketName: aws.String(s3ErrorReportBucketName),
        },
    },
    ScheduledQueryExecutionRoleArn: aws.String(roleArn),
}

createScheduledQueryOutput, err :=
 timestreamBuilder.QuerySvc.CreateScheduledQuery(createScheduledQueryInput)

if err != nil {
    fmt.Printf("Error: %s", err.Error())
} else {
    fmt.Println("createScheduledQueryResult is successful")
    return *createScheduledQueryOutput.Arn, nil
}
return "", err
}
```

```
func (timestreamBuilder TimestreamBuilder) CreateValidScheduledQuery(topicArn
string, roleArn string, s3ErrorReportBucketName string,
    sqDatabaseName string, sqTableName string, databaseName string, tableName
string) (string, error) {

    targetConfiguration := timestreamquery.TargetConfiguration{
        TimestreamConfiguration: &timestreamquery.TimestreamConfiguration{
            DatabaseName: aws.String(sqDatabaseName),
            TableName:    aws.String(sqTableName),
            TimeColumn:   aws.String("binned_timestamp"),
            DimensionMappings: []*timestreamquery.DimensionMapping{
                {
                    Name:               aws.String("region"),
                    DimensionValueType: aws.String("VARCHAR"),
                },
                {
                    Name:               aws.String("az"),
                    DimensionValueType: aws.String("VARCHAR"),
                },
                {
                    Name:               aws.String("hostname"),
                    DimensionValueType: aws.String("VARCHAR"),
                },
            },
            MultiMeasureMappings: &timestreamquery.MultiMeasureMappings{
                TargetMultiMeasureName: aws.String("multi-metrics"),
                MultiMeasureAttributeMappings:
[]*timestreamquery.MultiMeasureAttributeMapping{
                    {
                        SourceColumn:     aws.String("avg_cpu_utilization"),
                        MeasureValueType:
aws.String(timestreamquery.MeasureValueTypeDouble),
                    },
                    {
                        SourceColumn:     aws.String("p90_cpu_utilization"),
                        MeasureValueType:
aws.String(timestreamquery.MeasureValueTypeDouble),
                    },
                    {
                        SourceColumn:     aws.String("p95_cpu_utilization"),
                        MeasureValueType:
aws.String(timestreamquery.MeasureValueTypeDouble),
                    },
                    {
```

```
                        SourceColumn:     aws.String("p99_cpu_utilization"),
                        MeasureValueType:
 aws.String(timestreamquery.MeasureValueTypeDouble),
                    },
                },
            },
        },
    }
    return timestreamBuilder.createScheduledQuery(topicArn, roleArn,
 s3ErrorReportBucketName,
        fmt.Sprintf(QUERY, databaseName, tableName), targetConfiguration)
 }
```

Python

```
HOSTNAME = "host-24Gju"
SQ_NAME = "daily-sample"
ERROR_BUCKET_NAME = "scheduledquerysamplerrorbucket" +
 ''.join([choice(ascii_lowercase) for _ in range(5)])
QUERY = \
    "SELECT region, az, hostname, BIN(time, 15s) AS binned_timestamp, " \
    "    ROUND(AVG(cpu_utilization), 2) AS avg_cpu_utilization, " \
    "    ROUND(APPROX_PERCENTILE(cpu_utilization, 0.9), 2) AS p90_cpu_utilization, "
 \
    "    ROUND(APPROX_PERCENTILE(cpu_utilization, 0.95), 2) AS p95_cpu_utilization,
 " \
    "    ROUND(APPROX_PERCENTILE(cpu_utilization, 0.99), 2) AS p99_cpu_utilization "
 \
    "FROM " + database_name + "." + table_name + " " \
    "WHERE measure_name = 'metrics' " \
    "AND hostname = '" + self.HOSTNAME + "' " \
    "AND time > ago(2h) " \
    "GROUP BY region, hostname, az, BIN(time, 15s) " \
    "ORDER BY binned_timestamp ASC " \
    "LIMIT 5"

def create_scheduled_query_helper(self, topic_arn, role_arn, query,
 target_configuration):
    print("\nCreating Scheduled Query")
    schedule_configuration = {
        'ScheduleExpression': 'cron(0/2 * * * ? *)'
    }
    notification_configuration = {
```

```
        'SnsConfiguration': {
            'TopicArn': topic_arn
        }
    }
    error_report_configuration = {
        'S3Configuration': {
            'BucketName': ERROR_BUCKET_NAME
        }
    }

    try:
        create_scheduled_query_response = \
            query_client.create_scheduled_query(Name=self.SQ_NAME,
                 QueryString=query,
                 ScheduleConfiguration=schedule_configuration,
                 NotificationConfiguration=notification_configuration,
                 TargetConfiguration=target_configuration,
                 ScheduledQueryExecutionRoleArn=role_arn,
                 ErrorReportConfiguration=error_report_configuration
                 )
        print("Successfully created scheduled query : ",
 create_scheduled_query_response['Arn'])
        return create_scheduled_query_response['Arn']
    except Exception as err:
        print("Scheduled Query creation failed:", err)
        raise err

def create_valid_scheduled_query(self, topic_arn, role_arn):
    target_configuration = {
        'TimestreamConfiguration': {
            'DatabaseName': self.sq_database_name,
            'TableName': self.sq_table_name,
            'TimeColumn': 'binned_timestamp',
            'DimensionMappings': [
                {'Name': 'region', 'DimensionValueType': 'VARCHAR'},
                {'Name': 'az', 'DimensionValueType': 'VARCHAR'},
                {'Name': 'hostname', 'DimensionValueType': 'VARCHAR'}
            ],
            'MultiMeasureMappings': {
                'TargetMultiMeasureName': 'target_name',
                'MultiMeasureAttributeMappings': [
                    {'SourceColumn': 'avg_cpu_utilization', 'MeasureValueType':
 'DOUBLE',
                     'TargetMultiMeasureAttributeName': 'avg_cpu_utilization'},
```

```
                    {'SourceColumn': 'p90_cpu_utilization', 'MeasureValueType':
 'DOUBLE',
                     'TargetMultiMeasureAttributeName': 'p90_cpu_utilization'},
                    {'SourceColumn': 'p95_cpu_utilization', 'MeasureValueType':
 'DOUBLE',
                     'TargetMultiMeasureAttributeName': 'p95_cpu_utilization'},
                    {'SourceColumn': 'p99_cpu_utilization', 'MeasureValueType':
 'DOUBLE',
                     'TargetMultiMeasureAttributeName': 'p99_cpu_utilization'},
                ]
            }
        }
    }

    return self.create_scheduled_query_helper(topic_arn, role_arn, QUERY,
 target_configuration)
```

Node.js

Il seguente frammento utilizza lo stile for V2. AWS SDK JavaScript Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
const DATABASE_NAME = 'devops_sample_application';
const TABLE_NAME = 'host_metrics_sample_application';
const SQ_DATABASE_NAME = 'sq_result_database';
const SQ_TABLE_NAME = 'sq_result_table';
const HOSTNAME = "host-24Gju";
const SQ_NAME = "daily-sample";
const SCHEDULE_EXPRESSION = "cron(0/1 * * * ? *)";

// Find the average, p90, p95, and p99 CPU utilization for a specific EC2 host over
 the past 2 hours.
const VALID_QUERY = "SELECT region, az, hostname, BIN(time, 15s) AS
 binned_timestamp, " +
    " ROUND(AVG(cpu_utilization), 2) AS avg_cpu_utilization, " +
    " ROUND(APPROX_PERCENTILE(cpu_utilization, 0.9), 2) AS p90_cpu_utilization, " +
    " ROUND(APPROX_PERCENTILE(cpu_utilization, 0.95), 2) AS p95_cpu_utilization, " +
    " ROUND(APPROX_PERCENTILE(cpu_utilization, 0.99), 2) AS p99_cpu_utilization " +
    "FROM " + DATABASE_NAME + "." + TABLE_NAME + " " +
    "WHERE measure_name = 'metrics' " +
    " AND hostname = '" + HOSTNAME + "' " +
    " AND time > ago(2h) " +
```

```
    "GROUP BY region, hostname, az, BIN(time, 15s) " +
    "ORDER BY binned_timestamp ASC " +
    "LIMIT 5";

async function createScheduledQuery(topicArn, roleArn, s3ErrorReportBucketName) {
    console.log("Creating Valid Scheduled Query");
    const DimensionMappingList = [{
            'Name': 'region',
            'DimensionValueType': 'VARCHAR'
        },
        {
            'Name': 'az',
            'DimensionValueType': 'VARCHAR'
        },
        {
            'Name': 'hostname',
            'DimensionValueType': 'VARCHAR'
        }
    ];

    const MultiMeasureMappings = {
        TargetMultiMeasureName: "multi-metrics",
        MultiMeasureAttributeMappings: [{
                'SourceColumn': 'avg_cpu_utilization',
                'MeasureValueType': 'DOUBLE'
            },
            {
                'SourceColumn': 'p90_cpu_utilization',
                'MeasureValueType': 'DOUBLE'
            },
            {
                'SourceColumn': 'p95_cpu_utilization',
                'MeasureValueType': 'DOUBLE'
            },
            {
                'SourceColumn': 'p99_cpu_utilization',
                'MeasureValueType': 'DOUBLE'
            },
        ]
    }

    const timestreamConfiguration = {
        DatabaseName: SQ_DATABASE_NAME,
        TableName: SQ_TABLE_NAME,
```

```
        TimeColumn: "binned_timestamp",
        DimensionMappings: DimensionMappingList,
        MultiMeasureMappings: MultiMeasureMappings
    }

    const createScheduledQueryRequest = {
        Name: SQ_NAME,
        QueryString: VALID_QUERY,
        ScheduleConfiguration: {
            ScheduleExpression: SCHEDULE_EXPRESSION
        },
        NotificationConfiguration: {
            SnsConfiguration: {
                TopicArn: topicArn
            }
        },
        TargetConfiguration: {
            TimestreamConfiguration: timestreamConfiguration
        },
        ScheduledQueryExecutionRoleArn: roleArn,
        ErrorReportConfiguration: {
            S3Configuration: {
                BucketName: s3ErrorReportBucketName
            }
        }
    };
    try {
        const data = await
 queryClient.createScheduledQuery(createScheduledQueryRequest).promise();
        console.log("Successfully created scheduled query: " + data.Arn);
        return data.Arn;
    } catch (err) {
        console.log("Scheduled Query creation failed: ", err);
        throw err;
    }
}
```

.NET

```
public const string Hostname = "host-24Gju";
public const string SqName = "timestream-sample";
public const string SqDatabaseName = "sq_result_database";
public const string SqTableName = "sq_result_table";
```

```
public const string ErrorConfigurationS3BucketNamePrefix = "error-configuration-
sample-s3-bucket-";
public const string ScheduleExpression = "cron(0/2 * * * ? *)";

// Find the average, p90, p95, and p99 CPU utilization for a specific EC2 host over
 the past 2 hours.
public const string ValidQuery = "SELECT region, az, hostname, BIN(time, 15s) AS
 binned_timestamp, " +
      "ROUND(AVG(cpu_utilization), 2) AS avg_cpu_utilization, " +
      "ROUND(APPROX_PERCENTILE(cpu_utilization, 0.9), 2) AS p90_cpu_utilization, " +
      "ROUND(APPROX_PERCENTILE(cpu_utilization, 0.95), 2) AS p95_cpu_utilization, "
 +
      "ROUND(APPROX_PERCENTILE(cpu_utilization, 0.99), 2) AS p99_cpu_utilization " +
      "FROM " + Constants.DATABASE_NAME + "." + Constants.TABLE_NAME + " " +
      "WHERE measure_name = 'metrics' " +
      "AND hostname = '" + Hostname + "' " +
      "AND time > ago(2h) " +
      "GROUP BY region, hostname, az, BIN(time, 15s) " +
      "ORDER BY binned_timestamp ASC " +
      "LIMIT 5";

private async Task<String> CreateValidScheduledQuery(string topicArn, string
 roleArn,
             string databaseName, string tableName, string s3ErrorReportBucketName)
 {
     List<MultiMeasureAttributeMapping> sourceColToMeasureValueTypes =
         new List<MultiMeasureAttributeMapping>()
         {
             new()
             {
                 SourceColumn = "avg_cpu_utilization",
                 MeasureValueType = MeasureValueType.DOUBLE.Value
             },
             new()
             {
                 SourceColumn = "p90_cpu_utilization",
                 MeasureValueType = MeasureValueType.DOUBLE.Value
             },
             new()
             {
                 SourceColumn = "p95_cpu_utilization",
                 MeasureValueType = MeasureValueType.DOUBLE.Value
             },
```

```
            new()
            {
                SourceColumn = "p99_cpu_utilization",
                MeasureValueType = MeasureValueType.DOUBLE.Value
            }
        };

    TargetConfiguration targetConfiguration = new TargetConfiguration()
    {
        TimestreamConfiguration = new TimestreamConfiguration()
        {
            DatabaseName = databaseName,
            TableName = tableName,
            TimeColumn = "binned_timestamp",
            DimensionMappings = new List<DimensionMapping>()
            {
                new()
                {
                    Name = "region",
                    DimensionValueType = "VARCHAR"
                },
                new()
                {
                    Name = "az",
                    DimensionValueType = "VARCHAR"
                },
                new()
                {
                    Name = "hostname",
                    DimensionValueType = "VARCHAR"
                }
            },
            MultiMeasureMappings = new MultiMeasureMappings()
            {
                TargetMultiMeasureName = "multi-metrics",
                MultiMeasureAttributeMappings = sourceColToMeasureValueTypes
            }
        }
    };
    return await CreateScheduledQuery(topicArn, roleArn, s3ErrorReportBucketName,
        ScheduledQueryConstants.ValidQuery, targetConfiguration);
 }

private async Task<String> CreateScheduledQuery(string topicArn, string roleArn,
```

```
            string s3ErrorReportBucketName, string query, TargetConfiguration
targetConfiguration)
{
    try
    {
        Console.WriteLine("Creating Scheduled Query");
        CreateScheduledQueryResponse response = await
_amazonTimestreamQuery.CreateScheduledQueryAsync(
            new CreateScheduledQueryRequest()
            {
                Name = ScheduledQueryConstants.SqName,
                QueryString = query,
                ScheduleConfiguration = new ScheduleConfiguration()
                {
                    ScheduleExpression = ScheduledQueryConstants.ScheduleExpression
                },
                NotificationConfiguration = new NotificationConfiguration()
                {
                    SnsConfiguration = new SnsConfiguration()
                    {
                        TopicArn = topicArn
                    }
                },
                TargetConfiguration = targetConfiguration,
                ErrorReportConfiguration = new ErrorReportConfiguration()
                {
                    S3Configuration = new S3Configuration()
                    {
                        BucketName = s3ErrorReportBucketName
                    }
                },
                ScheduledQueryExecutionRoleArn = roleArn
            });
        Console.WriteLine($"Successfully created scheduled query :
{response.Arn}");
        return response.Arn;
    }
    catch (Exception e)
    {
        Console.WriteLine($"Scheduled Query creation failed: {e}");
        throw;
    }
}
```

Elenca le query pianificate

È possibile utilizzare i seguenti frammenti di codice per elencare le query pianificate.

Java

```
public void listScheduledQueries() {
    System.out.println("Listing Scheduled Query");
    try {
        String nextToken = null;
        List<String> scheduledQueries = new ArrayList<>();

        do {
            ListScheduledQueriesResult listScheduledQueriesResult =
                    queryClient.listScheduledQueries(new
 ListScheduledQueriesRequest()
                            .withNextToken(nextToken).withMaxResults(10));
            List<ScheduledQuery> scheduledQueryList =
 listScheduledQueriesResult.getScheduledQueries();

            printScheduledQuery(scheduledQueryList);
            nextToken = listScheduledQueriesResult.getNextToken();
        } while (nextToken != null);
    }
    catch (Exception e) {
        System.out.println("List Scheduled Query failed: " + e);
        throw e;
    }
}

public void printScheduledQuery(List<ScheduledQuery> scheduledQueryList) {
    for (ScheduledQuery scheduledQuery: scheduledQueryList) {
        System.out.println(scheduledQuery.getArn());
    }
}
```

Java v2

```
public void listScheduledQueries() {
    System.out.println("Listing Scheduled Query");
    try {
        String nextToken = null;
```

```
        do {
            ListScheduledQueriesResponse listScheduledQueriesResult =

 queryClient.listScheduledQueries(ListScheduledQueriesRequest.builder()
                            .nextToken(nextToken).maxResults(10)
                            .build());
            List<ScheduledQuery> scheduledQueryList =
 listScheduledQueriesResult.scheduledQueries();

            printScheduledQuery(scheduledQueryList);
            nextToken = listScheduledQueriesResult.nextToken();
        } while (nextToken != null);
    }
    catch (Exception e) {
        System.out.println("List Scheduled Query failed: " + e);
        throw e;
    }
}

public void printScheduledQuery(List<ScheduledQuery> scheduledQueryList) {
    for (ScheduledQuery scheduledQuery: scheduledQueryList) {
        System.out.println(scheduledQuery.arn());
    }
}
```

Go

```
func (timestreamBuilder TimestreamBuilder) ListScheduledQueries()
 ([]*timestreamquery.ScheduledQuery, error) {

     var nextToken *string = nil
     var scheduledQueries []*timestreamquery.ScheduledQuery
     for ok := true; ok; ok = nextToken != nil {
         listScheduledQueriesInput := &timestreamquery.ListScheduledQueriesInput{
             MaxResults: aws.Int64(15),
         }
         if nextToken != nil {
             listScheduledQueriesInput.NextToken = aws.String(*nextToken)
         }

         listScheduledQueriesOutput, err :=
 timestreamBuilder.QuerySvc.ListScheduledQueries(listScheduledQueriesInput)
         if err != nil {
```

```
            fmt.Printf("Error: %s", err.Error())
            return nil, err
        }
        scheduledQueries = append(scheduledQueries,
 listScheduledQueriesOutput.ScheduledQueries...)
        nextToken = listScheduledQueriesOutput.NextToken
    }
    return scheduledQueries, nil
}
```

Python

```
def list_scheduled_queries(self):
    print("\nListing Scheduled Queries")
    try:
        response = self.query_client.list_scheduled_queries(MaxResults=10)
        self.print_scheduled_queries(response['ScheduledQueries'])
        next_token = response.get('NextToken', None)
        while next_token:
            response =
 self.query_client.list_scheduled_queries(NextToken=next_token, MaxResults=10)
            self.print_scheduled_queries(response['ScheduledQueries'])
            next_token = response.get('NextToken', None)
    except Exception as err:
        print("List scheduled queries failed:", err)
        raise err

@staticmethod
def print_scheduled_queries(scheduled_queries):
    for scheduled_query in scheduled_queries:
        print(scheduled_query['Arn'])
```

Node.js

Il seguente frammento utilizza lo stile for V2. AWS SDK JavaScript Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function listScheduledQueries() {
    console.log("Listing Scheduled Query");
    try {
        var nextToken = null;
        do {
```

```
            var params = {
                MaxResults: 10,
                NextToken: nextToken
            }
            var data = await queryClient.listScheduledQueries(params).promise();
            var scheduledQueryList = data.ScheduledQueries;
            printScheduledQuery(scheduledQueryList);
            nextToken = data.NextToken;
        }
        while (nextToken != null);
    }  catch (err) {
        console.log("List Scheduled Query failed: ", err);
        throw err;
    }
}

async function printScheduledQuery(scheduledQueryList) {
    scheduledQueryList.forEach(element => console.log(element.Arn));
}
```

.NET

```
private async Task ListScheduledQueries()
{
    try
    {
        Console.WriteLine("Listing Scheduled Query");
        string nextToken;
        do
        {
            ListScheduledQueriesResponse response =
                await _amazonTimestreamQuery.ListScheduledQueriesAsync(new
ListScheduledQueriesRequest());
            foreach (var scheduledQuery in response.ScheduledQueries)
            {
                Console.WriteLine($"{scheduledQuery.Arn}");
            }

            nextToken = response.NextToken;
        } while (nextToken != null);
    }
    catch (Exception e)
    {
```

```
         Console.WriteLine($"List Scheduled Query failed: {e}");
         throw;
     }
 }
```

Descrivi una query pian

È possibile utilizzare i seguenti frammenti di codice per descrivere una query pianificata.

Java

```
public void describeScheduledQueries(String scheduledQueryArn) {
    System.out.println("Describing Scheduled Query");
    try {
        DescribeScheduledQueryResult describeScheduledQueryResult =
 queryClient.describeScheduledQuery(new
 DescribeScheduledQueryRequest().withScheduledQueryArn(scheduledQueryArn));
        System.out.println(describeScheduledQueryResult);
    }
    catch (ResourceNotFoundException e) {
        System.out.println("Scheduled Query doesn't exist");
        throw e;
    }
    catch (Exception e) {
        System.out.println("Describe Scheduled Query failed: " + e);
        throw e;
    }
}
```

Java v2

```
public void describeScheduledQueries(String scheduledQueryArn) {
    System.out.println("Describing Scheduled Query");
    try {
        DescribeScheduledQueryResponse describeScheduledQueryResult =

 queryClient.describeScheduledQuery(DescribeScheduledQueryRequest.builder()
                        .scheduledQueryArn(scheduledQueryArn)
                        .build());
        System.out.println(describeScheduledQueryResult);
    }
    catch (ResourceNotFoundException e) {
```

```
        System.out.println("Scheduled Query doesn't exist");
        throw e;
    }
    catch (Exception e) {
        System.out.println("Describe Scheduled Query failed: " + e);
        throw e;
    }
}
```

Go

```
func (timestreamBuilder TimestreamBuilder) DescribeScheduledQuery(scheduledQueryArn
 string) error {

    describeScheduledQueryInput := &timestreamquery.DescribeScheduledQueryInput{
        ScheduledQueryArn: aws.String(scheduledQueryArn),
    }
    describeScheduledQueryOutput, err :=
 timestreamBuilder.QuerySvc.DescribeScheduledQuery(describeScheduledQueryInput)

    if err != nil {
        if aerr, ok := err.(awserr.Error); ok {
            switch aerr.Code() {
            case timestreamquery.ErrCodeResourceNotFoundException:
                fmt.Println(timestreamquery.ErrCodeResourceNotFoundException,
 aerr.Error())
            default:
                fmt.Printf("Error: %s", err.Error())
            }
        } else {
            fmt.Printf("Error: %s", aerr.Error())
        }
        return err
    } else {
        fmt.Println("DescribeScheduledQuery is successful, below is the output:")
        fmt.Println(describeScheduledQueryOutput.ScheduledQuery)
        return nil
    }
}
```

Python

```
def describe_scheduled_query(self, scheduled_query_arn):
```

```
    print("\nDescribing Scheduled Query")
    try:
        response =
 self.query_client.describe_scheduled_query(ScheduledQueryArn=scheduled_query_arn)
        if 'ScheduledQuery' in response:
            response = response['ScheduledQuery']
            for key in response:
                print("{} :{}".format(key, response[key]))
    except self.query_client.exceptions.ResourceNotFoundException as err:
        print("Scheduled Query doesn't exist")
        raise err
    except Exception as err:
        print("Scheduled Query describe failed:", err)
        raise err
```

Node.js

Il seguente frammento utilizza lo stile for V2. AWS SDK JavaScript Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function describeScheduledQuery(scheduledQueryArn) {
     console.log("Describing Scheduled Query");
     var params = {
         ScheduledQueryArn: scheduledQueryArn
     }
     try {
         const data = await queryClient.describeScheduledQuery(params).promise();
         console.log(data.ScheduledQuery);
     } catch (err) {
         console.log("Describe Scheduled Query failed: ", err);
         throw err;
     }
 }
```

.NET

```
private async Task DescribeScheduledQuery(string scheduledQueryArn)
 {
     try
     {
         Console.WriteLine("Describing Scheduled Query");
```

```
        DescribeScheduledQueryResponse response = await
_amazonTimestreamQuery.DescribeScheduledQueryAsync(
            new DescribeScheduledQueryRequest()
            {
                ScheduledQueryArn = scheduledQueryArn
            });

Console.WriteLine($"{JsonConvert.SerializeObject(response.ScheduledQuery)}");
    }
    catch (ResourceNotFoundException e)
    {
        Console.WriteLine($"Scheduled Query doesn't exist: {e}");
        throw;
    }
    catch (Exception e)
    {
        Console.WriteLine($"Describe Scheduled Query failed: {e}");
        throw;
    }
}
```

Esegui interrogazione pianificata

È possibile utilizzare i seguenti frammenti di codice per eseguire una query pianificata.

Java

```
public void executeScheduledQueries(String scheduledQueryArn, Date invocationTime) {
    System.out.println("Executing Scheduled Query");
    try {
        ExecuteScheduledQueryResult executeScheduledQueryResult =
 queryClient.executeScheduledQuery(new ExecuteScheduledQueryRequest()
                .withScheduledQueryArn(scheduledQueryArn)
                .withInvocationTime(invocationTime)
        );

    }
    catch (ResourceNotFoundException e) {
        System.out.println("Scheduled Query doesn't exist");
        throw e;
    }
    catch (Exception e) {
```

```
        System.out.println("Execution Scheduled Query failed: " + e);
        throw e;
    }
}
```

Java v2

```
public void executeScheduledQuery(String scheduledQueryArn) {
    System.out.println("Executing Scheduled Query");
    try {
        ExecuteScheduledQueryResponse executeScheduledQueryResult =
 queryClient.executeScheduledQuery(ExecuteScheduledQueryRequest.builder()
                .scheduledQueryArn(scheduledQueryArn)
                .invocationTime(Instant.now())
                .build()
        );

        System.out.println("Execute ScheduledQuery response code: " +
 executeScheduledQueryResult.sdkHttpResponse().statusCode());

    }
    catch (ResourceNotFoundException e) {
        System.out.println("Scheduled Query doesn't exist");
        throw e;
    }
    catch (Exception e) {
        System.out.println("Execution Scheduled Query failed: " + e);
        throw e;
    }
}
```

Go

```
func (timestreamBuilder TimestreamBuilder) ExecuteScheduledQuery(scheduledQueryArn
 string, invocationTime time.Time) error {

     executeScheduledQueryInput := &timestreamquery.ExecuteScheduledQueryInput{
         ScheduledQueryArn: aws.String(scheduledQueryArn),
         InvocationTime:    aws.Time(invocationTime),
     }
     executeScheduledQueryOutput, err :=
 timestreamBuilder.QuerySvc.ExecuteScheduledQuery(executeScheduledQueryInput)
```

```
    if err != nil {
        if aerr, ok := err.(awserr.Error); ok {
            switch aerr.Code() {
            case timestreamquery.ErrCodeResourceNotFoundException:
                fmt.Println(timestreamquery.ErrCodeResourceNotFoundException,
aerr.Error())
            default:
                fmt.Printf("Error: %s", aerr.Error())
            }
        } else {
            fmt.Printf("Error: %s", err.Error())
        }
        return err
    } else {
        fmt.Println("ExecuteScheduledQuery is successful, below is the output:")
        fmt.Println(executeScheduledQueryOutput.GoString())
        return nil
    }
}
```

Python

```
def execute_scheduled_query(self, scheduled_query_arn, invocation_time):
    print("\nExecuting Scheduled Query")
    try:

 self.query_client.execute_scheduled_query(ScheduledQueryArn=scheduled_query_arn,
 InvocationTime=invocation_time)
        print("Successfully started executing scheduled query")
    except self.query_client.exceptions.ResourceNotFoundException as err:
        print("Scheduled Query doesn't exist")
        raise err
    except Exception as err:
        print("Scheduled Query execution failed:", err)
        raise err
```

Node.js

Il seguente frammento utilizza lo stile for V2. AWS SDK JavaScript Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function executeScheduledQuery(scheduledQueryArn, invocationTime) {
     console.log("Executing Scheduled Query");
     var params = {
         ScheduledQueryArn: scheduledQueryArn,
         InvocationTime: invocationTime
     }
     try {
         await queryClient.executeScheduledQuery(params).promise();
     } catch (err) {
         console.log("Execute Scheduled Query failed: ", err);
         throw err;
     }
 }
```

.NET

```
private async Task ExecuteScheduledQuery(string scheduledQueryArn, DateTime
 invocationTime)
 {
     try
     {
         Console.WriteLine("Running Scheduled Query");
         await _amazonTimestreamQuery.ExecuteScheduledQueryAsync(new
 ExecuteScheduledQueryRequest()
         {
             ScheduledQueryArn = scheduledQueryArn,
             InvocationTime = invocationTime
         });
         Console.WriteLine("Successfully started manual run of scheduled query");
     }
     catch (ResourceNotFoundException e)
     {
         Console.WriteLine($"Scheduled Query doesn't exist: {e}");
         throw;
     }
     catch (Exception e)
     {
         Console.WriteLine($"Execute Scheduled Query failed: {e}");
         throw;
     }
 }
```

Aggiorna interrogazione pianificata

È possibile utilizzare i seguenti frammenti di codice per aggiornare una query pianificata.

Java

```
public void updateScheduledQueries(String scheduledQueryArn) {
    System.out.println("Updating Scheduled Query");
    try {
        queryClient.updateScheduledQuery(new UpdateScheduledQueryRequest()
                .withScheduledQueryArn(scheduledQueryArn)
                .withState(ScheduledQueryState.DISABLED));
        System.out.println("Successfully update scheduled query state");
    }
    catch (ResourceNotFoundException e) {
        System.out.println("Scheduled Query doesn't exist");
        throw e;
    }
    catch (Exception e) {
        System.out.println("Execution Scheduled Query failed: " + e);
        throw e;
    }
}
```

Java v2

```
public void updateScheduledQuery(String scheduledQueryArn, ScheduledQueryState
 state) {
    System.out.println("Updating Scheduled Query");
    try {
        queryClient.updateScheduledQuery(UpdateScheduledQueryRequest.builder()
                .scheduledQueryArn(scheduledQueryArn)
                .state(state)
                .build());
        System.out.println("Successfully update scheduled query state");
    }
    catch (ResourceNotFoundException e) {
        System.out.println("Scheduled Query doesn't exist");
        throw e;
    }
    catch (Exception e) {
        System.out.println("Execution Scheduled Query failed: " + e);
        throw e;
```

```
    }
}
```

Go

```
func (timestreamBuilder TimestreamBuilder) UpdateScheduledQuery(scheduledQueryArn
 string) error {

    updateScheduledQueryInput := &timestreamquery.UpdateScheduledQueryInput{
        ScheduledQueryArn: aws.String(scheduledQueryArn),
        State:             aws.String(timestreamquery.ScheduledQueryStateDisabled),
    }
    _, err :=
 timestreamBuilder.QuerySvc.UpdateScheduledQuery(updateScheduledQueryInput)

    if err != nil {
        if aerr, ok := err.(awserr.Error); ok {
            switch aerr.Code() {
            case timestreamquery.ErrCodeResourceNotFoundException:
                fmt.Println(timestreamquery.ErrCodeResourceNotFoundException,
 aerr.Error())
            default:
                fmt.Printf("Error: %s", aerr.Error())
            }
        } else {
            fmt.Printf("Error: %s", err.Error())
        }
        return err
    } else {
        fmt.Println("UpdateScheduledQuery is successful")
        return nil
    }
}
```

Python

```
def update_scheduled_query(self, scheduled_query_arn, state):
    print("\nUpdating Scheduled Query")
    try:

 self.query_client.update_scheduled_query(ScheduledQueryArn=scheduled_query_arn,
                                                 State=state)
        print("Successfully update scheduled query state to", state)
```

```
    except self.query_client.exceptions.ResourceNotFoundException as err:
        print("Scheduled Query doesn't exist")
        raise err
    except Exception as err:
        print("Scheduled Query deletion failed:", err)
        raise err
```

Node.js

Il seguente frammento utilizza lo stile for V2. AWS SDK JavaScript Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function updateScheduledQueries(scheduledQueryArn) {
     console.log("Updating Scheduled Query");
     var params = {
         ScheduledQueryArn: scheduledQueryArn,
         State: "DISABLED"
     }
     try {
         await queryClient.updateScheduledQuery(params).promise();
         console.log("Successfully update scheduled query state");
     } catch (err) {
         console.log("Update Scheduled Query failed: ", err);
         throw err;
     }
 }
```

.NET

```
private async Task UpdateScheduledQuery(string scheduledQueryArn,
 ScheduledQueryState state)
 {
     try
     {
         Console.WriteLine("Updating Scheduled Query");
         await _amazonTimestreamQuery.UpdateScheduledQueryAsync(new
 UpdateScheduledQueryRequest()
         {
             ScheduledQueryArn = scheduledQueryArn,
             State = state
         });
         Console.WriteLine("Successfully update scheduled query state");
```

```
    }
    catch (ResourceNotFoundException e)
    {
        Console.WriteLine($"Scheduled Query doesn't exist: {e}");
        throw;
    }
    catch (Exception e)
    {
        Console.WriteLine($"Update Scheduled Query failed: {e}");
        throw;
    }
}
```

Eliminare una query pianificata

È possibile utilizzare i seguenti frammenti di codice per eliminare una query pianificata.

Java

```
public void deleteScheduledQuery(String scheduledQueryArn) {
    System.out.println("Deleting Scheduled Query");

    try {
        queryClient.deleteScheduledQuery(new
 DeleteScheduledQueryRequest().withScheduledQueryArn(scheduledQueryArn));
        System.out.println("Successfully deleted scheduled query");
    }
    catch (Exception e) {
        System.out.println("Scheduled Query deletion failed: " + e);
    }
}
```

Java v2

```
public void deleteScheduledQuery(String scheduledQueryArn) {
    System.out.println("Deleting Scheduled Query");

    try {
        queryClient.deleteScheduledQuery(DeleteScheduledQueryRequest.builder()
                .scheduledQueryArn(scheduledQueryArn).build());
        System.out.println("Successfully deleted scheduled query");
    }
```

```
    catch (Exception e) {
        System.out.println("Scheduled Query deletion failed: " + e);
    }
}
```

Go

```
func (timestreamBuilder TimestreamBuilder) DeleteScheduledQuery(scheduledQueryArn
 string) error {

    deleteScheduledQueryInput := &timestreamquery.DeleteScheduledQueryInput{
        ScheduledQueryArn: aws.String(scheduledQueryArn),
    }
    _, err :=
 timestreamBuilder.QuerySvc.DeleteScheduledQuery(deleteScheduledQueryInput)

    if err != nil {
        fmt.Println("Error:")
        if aerr, ok := err.(awserr.Error); ok {
            switch aerr.Code() {
            case timestreamquery.ErrCodeResourceNotFoundException:
                fmt.Println(timestreamquery.ErrCodeResourceNotFoundException,
 aerr.Error())
            default:
                fmt.Printf("Error: %s", aerr.Error())
            }
        } else {
            fmt.Printf("Error: %s", err.Error())
        }
        return err
    } else {
        fmt.Println("DeleteScheduledQuery is successful")
        return nil
    }
}
```

Python

```
def delete_scheduled_query(self, scheduled_query_arn):
    print("\nDeleting Scheduled Query")
    try:

 self.query_client.delete_scheduled_query(ScheduledQueryArn=scheduled_query_arn)
```

```
        print("Successfully deleted scheduled query :", scheduled_query_arn)
    except Exception as err:
        print("Scheduled Query deletion failed:", err)
        raise err
```

Node.js

Il seguente frammento utilizza lo stile for V2. AWS SDK JavaScript Si basa sull'applicazione di esempio disponibile in Node.js, esempio Amazon Timestream LiveAnalytics per l'applicazione su. GitHub

```
async function deleteScheduleQuery(scheduledQueryArn) {
     console.log("Deleting Scheduled Query");
     const params = {
         ScheduledQueryArn: scheduledQueryArn
     }
     try {
         await queryClient.deleteScheduledQuery(params).promise();
         console.log("Successfully deleted scheduled query");
     } catch (err) {
         console.log("Scheduled Query deletion failed: ", err);
     }
 }
```

.NET

```
private async Task DeleteScheduledQuery(string scheduledQueryArn)
 {
     try
     {
         Console.WriteLine("Deleting Scheduled Query");
         await _amazonTimestreamQuery.DeleteScheduledQueryAsync(new
 DeleteScheduledQueryRequest()
         {
             ScheduledQueryArn = scheduledQueryArn
         });
         Console.WriteLine($"Successfully deleted scheduled query :
 {scheduledQueryArn}");
     }
     catch (Exception e)
     {
         Console.WriteLine($"Scheduled Query deletion failed: {e}");
```

```
            throw;
        }
    }
```

Utilizzo del caricamento in batch in Timestream per LiveAnalytics

Con il caricamento in batch per Amazon Timestream LiveAnalytics for, puoi importare file archiviati in Amazon S3 in CSV Timestream in batch. Con questa nuova funzionalità, puoi conservare i tuoi dati in Timestream LiveAnalytics senza dover fare affidamento su altri strumenti o scrivere codice personalizzato. Puoi utilizzare il caricamento in batch per completare i dati con tempi di attesa flessibili, ad esempio dati che non sono immediatamente necessari per l'interrogazione o l'analisi.

È possibile creare attività di caricamento in batch utilizzando AWS Management Console AWS CLI, the e. AWS SDKs Per ulteriori informazioni, consulta Utilizzo del caricamento in batch con la console, Utilizzo del caricamento in batch con AWS CLI e Utilizzo del caricamento in batch con AWS SDKs.

Oltre al caricamento in batch, è possibile scrivere più record contemporaneamente con l' WriteRecords APIoperazione. Per indicazioni su quale usare, consultaScelta tra WriteRecords API operazione e caricamento in batch.

Argomenti

- Concetti di caricamento in batch in Timestream
- Prerequisiti per il caricamento in batch
- Procedure ottimali per il caricamento in batch
- Preparazione di un file di dati di caricamento in batch
- Mappature dei modelli di dati per il caricamento in batch
- Utilizzo del caricamento in batch con la console
- Utilizzo del caricamento in batch con AWS CLI
- Utilizzo del caricamento in batch con AWS SDKs
- Utilizzo dei report sugli errori di caricamento in batch

Concetti di caricamento in batch in Timestream

Esamina i seguenti concetti per comprendere meglio la funzionalità di caricamento in batch.

Attività di caricamento in batch: l'attività che definisce i dati di origine e la destinazione in Amazon Timestream. Quando crei l'attività di caricamento in batch, specifichi una configurazione aggiuntiva come il modello di dati. È possibile creare attività di caricamento in batch tramite AWS Management Console AWS CLI, the e AWS SDKs.

Destinazione di importazione: il database e la tabella di destinazione in Timestream. Per informazioni sulla creazione di database e tabelle, consulta Creazione di un database e. Creare una tabella

Origine dati: il CSV file sorgente archiviato in un bucket S3. Per informazioni sulla preparazione del file di dati, consulta. Preparazione di un file di dati di caricamento in batch Per informazioni sui prezzi di S3, consulta i prezzi di Amazon S3.

Rapporto sugli errori di caricamento in batch: un rapporto che memorizza le informazioni sugli errori di un'operazione di caricamento in batch. La posizione S3 per i report sugli errori di caricamento in batch viene definita come parte di un'attività di caricamento in batch. Per informazioni sulle informazioni contenute nei report, vedereUtilizzo dei report sugli errori di caricamento in batch.

Mappatura del modello di dati: una mappatura del carico in batch per tempo, dimensioni e misure che parte da un'origine dati in una posizione S3 a un Timestream di destinazione per la tabella. LiveAnalytics Per ulteriori informazioni, consulta Mappature dei modelli di dati per il caricamento in batch.

Prerequisiti per il caricamento in batch

Questo è un elenco di prerequisiti per l'utilizzo del caricamento in batch. Per le best practice, consulta Procedure ottimali per il caricamento in batch.

- I dati di origine del caricamento in batch vengono archiviati in Amazon S3 in CSV formato con intestazioni.
- Per ogni bucket di origine Amazon S3, devi disporre delle seguenti autorizzazioni in una policy allegata:

```
"s3:GetObject",
"s3:GetBucketAcl"
"s3:ListBucket"
```

  Analogamente, per ogni bucket di output di Amazon S3 in cui vengono scritti i report, è necessario disporre delle seguenti autorizzazioni in una policy allegata:

```
"s3:PutObject",
```

```
"s3:GetBucketAcl"
```

Per esempio:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject",
        "s3:GetBucketAcl"
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::inputs-source-bucket-name-A”
        "arn:aws:s3:::inputs-source-bucket-name-B”
      ],
      "Effect": "Allow"
    },
    {
      "Action": [
        "s3:PutObject",
        "s3:GetBucketAcl"
      ],
      "Resource": [
        "arn:aws:s3:::reports-output-bucket-name”
      ]
      "Effect": "Allow"
    }
  ]
}
```

- Timestream for li LiveAnalytics analizza CSV mappando le informazioni fornite nel modello di dati alle intestazioni. CSV I dati devono avere una colonna che rappresenti il timestamp, almeno una colonna di dimensione e almeno una colonna di misura.
- I bucket S3 utilizzati con il caricamento in batch devono trovarsi nella stessa regione e provenire dallo stesso account della LiveAnalytics tabella Timestream for utilizzata nel caricamento in batch.
- La `timestamp` colonna deve essere un tipo di dati lungo che rappresenta il tempo trascorso dall'epoca Unix. Ad esempio, il timestamp `2021-03-25T08:45:21Z` sarebbe rappresentato come. `1616661921` Timestream supporta secondi, millisecondi, microsecondi e nanosecondi per la precisione del timestamp. Quando si utilizza il linguaggio di interrogazione, è possibile eseguire

la conversione tra formati con funzioni come. `to_unixtime` Per ulteriori informazioni, consulta Funzioni data/ora.

- Timestream supporta il tipo di dati stringa per i valori delle dimensioni. Supporta tipi di dati long, double, string e booleani per le colonne di misura.

Per i limiti e le quote di carico in batch, vedere. Caricamento in batch

Procedure ottimali per il caricamento in batch

Il carico in batch funziona al meglio (produttività elevata) se si rispettano le seguenti condizioni e raccomandazioni:

1. CSVi file inviati per l'ingestione sono di piccole dimensioni, in particolare con una dimensione compresa tra 100 MB e 1 GB, per migliorare il parallelismo e la velocità di ingestione.
2. Evita di importare contemporaneamente dati nella stessa tabella (ad esempio utilizzando l' WriteRecords APIoperazione o una query pianificata) quando è in corso il caricamento del batch. Ciò potrebbe causare rallentamenti e l'operazione di caricamento in batch non riuscirà.
3. Non aggiungere, modificare o rimuovere file dal bucket S3 utilizzato nel caricamento in batch mentre l'attività di caricamento in batch è in esecuzione.
4. Non eliminate o revocate le autorizzazioni dalle tabelle o dall'origine e non segnalate i bucket S3 con attività di caricamento batch pianificate o in corso.
5. Quando si importano dati con un set di valori di dimensione ad alta cardinalità, segui le indicazioni riportate in. Consigli per il partizionamento di record multimisura
6. Assicurati di verificare la correttezza dei dati inviando un file di piccole dimensioni. Tutti i dati inviati al caricamento in batch ti verranno addebitati indipendentemente dalla correttezza. Per ulteriori informazioni sui prezzi, consulta i prezzi di Amazon Timestream.
7. Non riprendere un'attività di caricamento in batch a meno che `ActiveMagneticStorePartitions` il numero non sia inferiore a 250. Il processo potrebbe essere rallentato e fallire. L'invio di più lavori contemporaneamente per lo stesso database dovrebbe ridurre il numero.

Di seguito sono riportate le best practice relative alla console:

1. Utilizza il generatore solo per una modellazione dei dati più semplice che utilizza un solo nome di misura per i record di più misure.

2. Per una modellazione dei dati più complessa, usa. JSON Ad esempio, utilizzare JSON quando si utilizzano più nomi di misure quando si utilizzano record con più misure.

Per ulteriori informazioni su Timestream relative alle LiveAnalytics best practice, consulta. Best practice

Preparazione di un file di dati di caricamento in batch

Un file di dati di origine ha valori separati da delimitatori. Il termine più specifico, valori separati da virgole () viene utilizzato genericamente. CSV I separatori di colonna validi includono virgole e pipe. I record sono separati da nuove righe. I file devono essere archiviati in Amazon S3. Quando crei una nuova attività di caricamento in batch, la posizione dei dati di origine viene specificata da un ARN per il file. Un file contiene intestazioni. Una colonna rappresenta il timestamp. Almeno un'altra colonna rappresenta una misura.

I bucket S3 utilizzati con il caricamento in batch devono trovarsi nella stessa area del Timestream per la LiveAnalytics tabella utilizzata nel caricamento in batch. Non aggiungere o rimuovere file dal bucket S3 utilizzato nel caricamento in batch dopo l'invio dell'attività di caricamento in batch. Per informazioni sull'utilizzo dei bucket S3, consulta Guida introduttiva ad Amazon S3.

> **Note**
>
> CSVi file generati da alcune applicazioni come Excel potrebbero contenere un segno di ordine dei byte (BOM) che è in conflitto con la codifica prevista. Timestream per le attività di caricamento LiveAnalytics in batch che fanno riferimento a un CSV file con un errore quando vengono BOM elaborate a livello di codice. Per evitare ciò, puoi rimuovere ilBOM, che è un carattere invisibile.
> Ad esempio, è possibile salvare il file da un'applicazione come Notepad++ che consente di specificare una nuova codifica. È inoltre possibile utilizzare un'opzione programmatica che legge la prima riga, rimuove il carattere dalla riga e scrive il nuovo valore sulla prima riga del file.
> Quando si salva da Excel, sono disponibili diverse CSV opzioni. Il salvataggio con un'CSVopzione diversa potrebbe prevenire il problema descritto. Ma dovresti controllare il risultato perché un cambiamento nella codifica può influire su alcuni caratteri.

CSVparametri di formato

Utilizzate i caratteri di escape quando rappresentate un valore altrimenti riservato dai parametri di formato. Ad esempio, se il carattere delle virgolette è composto da virgolette doppie, per rappresentare una virgoletta doppia nei dati, posiziona il carattere di escape prima delle virgolette doppie.

Per informazioni su quando specificarli durante la creazione di un'operazione di caricamento in batch, vedereCreare un'attività di caricamento in batch.

Parametro	Opzioni
Separatore di colonne	(Virgola (',') | Pipe ('|') | Punto e virgola (';') | Tab ('/t') | Spazio vuoto (''))
Personaggio di fuga	nessuno
Cita il personaggio	Console: (Virgolette doppie («) | Virgolette singole ('))
Valore nullo	Spazio vuoto ('')
Taglia lo spazio bianco	Console: (No | Sì)

Mappature dei modelli di dati per il caricamento in batch

Di seguito viene illustrato lo schema per le mappature dei modelli di dati e viene fornito un esempio.

Schema di mappatura del modello di dati

La sintassi della `CreateBatchLoadTask` richiesta e un `BatchLoadTaskDescription` oggetto restituito da una chiamata per `DescribeBatchLoadTask` includere un `DataModelConfiguration` oggetto che include il `DataModel` caricamento in batch. `DataModel`Definisce le mappature dai dati di origine archiviati in CSV formato in una posizione S3 a un Timestream di destinazione per database e tabella. LiveAnalytics

Il `TimeColumn` campo indica la posizione dei dati di origine per il valore da mappare alla colonna della tabella di destinazione in Timestream for. `time` LiveAnalytics `TimeUnit`Speciifica

l'unità per`TimeColumn`, e può essere una delle`MILLISECONDS`,, `SECONDS` o. `MICROSECONDS` `NANOSECONDS` Sono disponibili anche mappature per dimensioni e misure. Le mappature delle dimensioni sono composte da colonne di origine e campi di destinazione.

Per ulteriori informazioni, vedere. DimensionMapping Le mappature per le misure hanno due opzioni, `MixedMeasureMappings` e. `MultiMeasureMappings`

Per riassumere, a `DataModel` contiene le mappature da un'origine dati in una posizione S3 a un Timestream di destinazione per la tabella per quanto segue. LiveAnalytics

- Orario
- Dimensioni
- Misure

Se possibile, ti consigliamo di mappare i dati di misura su record multimisura in Timestream for. LiveAnalytics Per informazioni sui vantaggi dei record multimisura, consulta. Record multimisura

Se più misure nei dati di origine sono memorizzate in una riga, puoi mappare tali misure su record di più misure in Timestream per utilizzarle. LiveAnalytics `MultiMeasureMappings` Se ci sono valori che devono essere mappati a un record a misura singola, puoi usare. `MixedMeasureMappings`

`MixedMeasureMappings`ed `MultiMeasureMappings` entrambi includono`MultiMeasureAttributeMappings`. I record a più misure sono supportati indipendentemente dal fatto che siano necessari record a misura singola.

Se in Timestream for sono necessari solo record di destinazione a più misure LiveAnalytics, è possibile definire le mappature delle misure nella seguente struttura.

```
CreateBatchLoadTask
    MeasureNameColumn
    MultiMeasureMappings
        TargetMultiMeasureName
        MultiMeasureAttributeMappings array
```

> **Note**
>
> Si consiglia di utilizzarlo quando possibile. `MultiMeasureMappings`

Se sono necessari record di destinazione a misura singola in Timestream for LiveAnalytics, puoi definire le mappature delle misure nella seguente struttura.

```
CreateBatchLoadTask
    MeasureNameColumn
    MixedMeasureMappings array
        MixedMeasureMapping
            MeasureName
            MeasureValueType
            SourceColumn
            TargetMeasureName
            MultiMeasureAttributeMappings array
```

Quando si utilizza`MultiMeasureMappings`, l'array è sempre necessario. `MultiMeasureAttributeMappings` Quando si utilizza l'`MixedMeasureMappings`array, se si `MeasureValueType` tratta `MULTI` di un dato elemento`MixedMeasureMapping`, `MultiMeasureAttributeMappings` è necessario a tal fine`MixedMeasureMapping`. Altrimenti, `MeasureValueType` indica il tipo di misura per il record a misura singola.

In entrambi i casi, è `MultiMeasureAttributeMapping` disponibile una serie di opzioni. Le mappature su record multimisura in ciascuno `MultiMeasureAttributeMapping` di essi vengono definite come segue:

`SourceColumn`

La colonna dei dati di origine che si trova in Amazon S3.

`TargetMultiMeasureAttributeName`

Il nome del nome multimisura di destinazione nella tabella di destinazione. Questo input è obbligatorio quando non `MeasureNameColumn` viene fornito. Se `MeasureNameColumn` fornito, il valore di quella colonna viene utilizzato come nome multimisura.

`MeasureValueType`

Uno di`DOUBLE`, `BIGINT` `BOOLEAN``VARCHAR`, o`TIMESTAMP`.

Mappature di modelli di dati con esempio **`MultiMeasureMappings`**

Questo esempio dimostra la mappatura su record multimisura, l'approccio preferito, che memorizza ogni valore di misura in una colonna dedicata. È possibile scaricare un esempio da sampleCSV. CSV

L'esempio ha i seguenti titoli da mappare a una colonna di destinazione in un Timestream for table. LiveAnalytics

- `time`
- `measure_name`
- `region`
- `location`
- `hostname`
- `memory_utilization`
- `cpu_utilization`

Identifica le `measure_name` colonne `time` e nel file. CSV In questo caso vengono mappate direttamente al Timestream per le colonne della LiveAnalytics tabella con lo stesso nome.

- `time`mappe per `time`
- `measure_name`mappa a `measure_name` (o al valore scelto)

Quando si utilizza ilAPI, si specifica `time` nel `TimeColumn` campo e un valore di unità di tempo supportato, ad esempio `MILLISECONDS` nel `TimeUnit` campo. Questi corrispondono al nome della colonna di origine e all'ora del timestamp immessi nella console. È possibile raggruppare o partizionare i record utilizzando la `measure_name` chiave definita. `MeasureNameColumn`

Nell'esempio, `regionlocation`, e `hostname` sono dimensioni. Le dimensioni sono mappate in una serie di `DimensionMapping` oggetti.

Per le misure, il valore `TargetMultiMeasureAttributeName` diventerà una colonna nel Timestream for table. LiveAnalytics È possibile mantenere lo stesso nome come in questo esempio. Oppure puoi specificarne uno nuovo. `MeasureValueType`è uno dei `DOUBLE`,`BIGINT`,`BOOLEAN`,`VARCHAR`, o`TIMESTAMP`.

```
{
  "TimeColumn": "time",
  "TimeUnit": "MILLISECONDS",
  "DimensionMappings": [
    {
      "SourceColumn": "region",
      "DestinationColumn": "region"
```

```
    },
    {
      "SourceColumn": "location",
      "DestinationColumn": "location"
    },
    {
      "SourceColumn": "hostname",
      "DestinationColumn": "hostname"
    }
  ],
  "MeasureNameColumn": "measure_name",
  "MultiMeasureMappings": {
    "MultiMeasureAttributeMappings": [
      {
        "SourceColumn": "memory_utilization",
        "TargetMultiMeasureAttributeName": "memory_utilization",
        "MeasureValueType": "DOUBLE"
      },
      {
        "SourceColumn": "cpu_utilization",
        "TargetMultiMeasureAttributeName": "cpu_utilization",
        "MeasureValueType": "DOUBLE"
      }
    ]
  }
}
```

Visual builder (7) Info

Reset all mappings

Search by column name

Source column name	Target table column name	Timestream attribute type	Data type
time	time	TIMESTAMP	TIMESTAMP
measure_name	measure_name	MEASURE_NAME	-
region	region	DIMENSION	VARCHAR
location	location	DIMENSION	VARCHAR
hostname	hostname	DIMENSION	VARCHAR
memory_utilization	memory_utilization	MULTI	DOUBLE
cpu_utilization	cpu_utilization	MULTI	DOUBLE

mappature di modelli di dati con esempio **`MixedMeasureMappings`**

Ti consigliamo di utilizzare questo approccio solo quando devi mappare record a misura singola in Timestream for. LiveAnalytics

Utilizzo del caricamento in batch con la console

Di seguito sono riportati i passaggi per utilizzare il caricamento in batch con AWS Management Console. È possibile scaricare un esempio CSV da sample CSV.

Argomenti

Accedi al caricamento in batch

Segui questi passaggi per accedere al caricamento in batch utilizzando AWS Management Console.

1. Apri la console Amazon Timestream.
2. Nel riquadro di navigazione, scegli Strumenti di gestione, quindi scegli Attività di caricamento in Batch.
3. Da qui, puoi visualizzare l'elenco delle attività di caricamento in batch e approfondire una determinata attività per maggiori dettagli. È inoltre possibile creare e riprendere le attività.

Creare un'attività di caricamento in batch

Segui questi passaggi per creare un'attività di caricamento in batch utilizzando AWS Management Console.

1. Apri la console Amazon Timestream.
2. Nel riquadro di navigazione, scegli Strumenti di gestione, quindi scegli Attività di caricamento in Batch.
3. Scegli Crea attività di caricamento in batch.
4. In Destinazione di importazione, scegli quanto segue.

- Database di destinazione: seleziona il nome del database creato inCreazione di un database .
- Tabella di destinazione: seleziona il nome della tabella creata inCreare una tabella.

Se necessario, puoi aggiungere una tabella da questo pannello con il pulsante Crea nuova tabella.

5. Dalla posizione Data source S3 in Data source, seleziona il bucket S3 in cui sono archiviati i dati di origine. Usa il pulsante Sfoglia S3 per visualizzare le risorse S3 a cui l'AWSaccount attivo ha accesso o inserisci la posizione S3. URL L'origine dati deve trovarsi nella stessa regione.
6. Nelle impostazioni del formato del file (sezione espandibile), è possibile utilizzare le impostazioni predefinite per analizzare i dati di input. Puoi anche scegliere Impostazioni avanzate. Da lì puoi scegliere i parametri di CSV formato e selezionare i parametri per analizzare i dati di input. Per informazioni su questi parametri, vedereCSVparametri di formato.
7. Da Configura la mappatura del modello di dati, configura il modello di dati. Per ulteriori indicazioni sul modello di dati, vedere Mappature dei modelli di dati per il caricamento in batch

   - Da Mappatura del modello di dati, scegli Mappatura dell'input di configurazione e scegli una delle seguenti opzioni.
     - Visual Builder: per mappare visivamente i dati, scegli o. TargetMultiMeasureNameMeasureNameColumn Quindi, da Visual Builder, mappa le colonne.

       Visual Builder rileva e carica automaticamente le intestazioni delle colonne di origine dal file di origine dati quando viene selezionato un singolo CSV file come origine dati. Scegli l'attributo e il tipo di dati per creare la mappatura.

       Per informazioni sull'utilizzo del Visual Builder, consulta. Utilizzando il visual builder
     - JSONeditor: un JSON editor in formato libero per configurare il modello di dati. Scegli questa opzione se conosci Timestream for LiveAnalytics e desideri creare mappature avanzate di modelli di dati.
     - JSONfile da S3: seleziona un file di JSON modello che hai archiviato in S3. Scegli questa opzione se hai già configurato un modello di dati e desideri riutilizzarlo per caricamenti batch aggiuntivi.
8. Dalla posizione S3 dei registri di errore nel rapporto del registro degli errori, seleziona la posizione S3 che verrà utilizzata per segnalare gli errori. Per informazioni su come utilizzare questo rapporto, consulta. Utilizzo dei report sugli errori di caricamento in batch

9. Per Tipo di chiave di crittografia, scegli una delle seguenti opzioni.
   - Chiave gestita da Amazon S3 (SSE-S3): una chiave di crittografia che Amazon S3 crea, gestisce e utilizza per te.
   - AWS KMS key (SSE-KMS) — Una chiave di crittografia protetta da (). AWS Key Management Service AWS KMS
10. Scegli Next (Successivo).
11. Nella pagina Rivedi e crea, rivedi le impostazioni e modificale se necessario.

> **Note**
>
> Non è possibile modificare le impostazioni dell'attività di caricamento in batch dopo la creazione dell'attività. I tempi di completamento delle attività variano in base alla quantità di dati importati.

12. Scegli Crea attività di caricamento in batch.

Riprendi un'attività di caricamento in batch

Quando si seleziona un'operazione di caricamento in batch con lo stato «Progresso interrotto» che è ancora ripristinabile, viene richiesto di riprendere l'operazione. C'è anche un banner con il pulsante Riprendi attività quando visualizzi i dettagli di tali attività. Le attività che è possibile riprendere hanno una data di scadenza. Dopo la scadenza di tale data, le attività non possono essere riprese.

Utilizzando il visual builder

Puoi utilizzare il visual builder per mappare le colonne di dati di origine di uno o più CSV file archiviati in un bucket S3 alle colonne di destinazione in un Timestream per tabella. LiveAnalytics

> **Note**
>
> Il tuo ruolo avrà bisogno dell'autorizzazione per il file. `SelectObjectContent` In caso contrario, dovrai aggiungere ed eliminare le colonne manualmente.

Modalità di caricamento automatico delle colonne di origine

Timestream for LiveAnalytics può scansionare automaticamente il CSV file sorgente per i nomi delle colonne se si specifica un solo bucket. Quando non ci sono mappature esistenti, puoi scegliere Importa colonne di origine.

1. Con l'opzione Visual Builder selezionata dalle impostazioni di input della configurazione Mapping, imposta l'input temporale Timestamp. `Milliseconds`è l'impostazione predefinita.
2. Fate clic sul pulsante Carica colonne di origine per importare le intestazioni di colonna presenti nel file di dati di origine. La tabella verrà compilata con i nomi delle intestazioni delle colonne di origine provenienti dal file di origine dati.
3. Scegli il nome della colonna della tabella Target, il tipo di attributo Timestream e il tipo di dati per ogni colonna di origine.

   Per informazioni dettagliate su queste colonne e sui possibili valori, consulta. Mappatura dei campi
4. Usa la drag-to-fill funzione per impostare il valore per più colonne contemporaneamente.

Aggiungi manualmente le colonne sorgente

Se utilizzi un bucket o un CSV prefisso e non uno soloCSV, puoi aggiungere ed eliminare le mappature delle colonne dall'editor visivo con i pulsanti Aggiungi mappatura delle colonne ed Elimina mappatura delle colonne. C'è anche un pulsante per ripristinare le mappature.

Mappatura dei campi

- Nome della colonna di origine: il nome di una colonna nel file di origine che rappresenta una misura da importare. Timestream for LiveAnalytics può compilare questo valore automaticamente quando si utilizzano le colonne di origine di importazione.
- Nome della colonna della tabella di destinazione: input opzionale che indica il nome della colonna per la misura nella tabella di destinazione.
- Tipo di attributo Timestream: il tipo di attributo dei dati nella colonna di origine specificata, ad esempio. `DIMENSION`
  - TIMESTAMP— Speciifica quando è stata raccolta una misura.
  - MULTI— Sono rappresentate più misure.
  - DIMENSION— Metadati delle serie temporali.
  - MEASURE_ NAME — Per i record a misura singola, questo è il nome della misura.

- Tipo di dati: il tipo di colonna Timestream, ad esempio. `BOOLEAN`
  - BIGINT— Un numero intero a 64 bit.
  - BOOLEAN— I due valori di verità della logica: vero e falso.
  - DOUBLE— Numero a precisione variabile a 64 bit.
  - TIMESTAMP— Un'istanza temporale che utilizza il tempo di precisione in UTC nanosecondi e tiene traccia del tempo dall'epoca di Unix.

Utilizzo del caricamento in batch con AWS CLI

Installazione

Per iniziare a utilizzare il caricamento in batch, procedi nel seguente modo.

1. Installa il AWS CLI utilizzando le istruzioni all'indirizzoAccesso ad Amazon Timestream LiveAnalytics per l'utilizzo di AWS CLI.
2. Esegui il comando seguente per verificare che i CLI comandi Timestream siano stati aggiornati. Verifica che create-batch-load-task sia nell'elenco.

   ```
   aws timestream-write help
   ```
3. Prepara una fonte di dati seguendo le istruzioni riportate inPreparazione di un file di dati di caricamento in batch.
4. Crea un database e una tabella seguendo le istruzioni riportate inAccesso ad Amazon Timestream LiveAnalytics per l'utilizzo di AWS CLI.
5. Crea un bucket S3 per l'output dei report. Il bucket deve trovarsi nella stessa regione. Per ulteriori informazioni sui bucket, consulta Creazione, configurazione e utilizzo dei bucket Amazon S3.
6. Crea un'attività di caricamento in batch. Per le fasi, consulta Crea un'attività di caricamento in batch.
7. Conferma lo stato dell'attività. Per le fasi, consulta Descrivi l'operazione di caricamento batch.

Crea un'attività di caricamento in batch

È possibile creare un'attività di caricamento in batch con il `create-batch-load-task` comando. Quando si crea un'operazione di caricamento in batch utilizzando ilCLI, è possibile utilizzare un JSON parametro che consente di aggregare i parametri in un unico JSON frammento. `cli-input-`

`json` È inoltre possibile suddividere questi dettagli utilizzando diversi altri parametri`data-model-configuration`, tra cui, `data-source-configuration` `report-configuration``target-database-name`, e. `target-table-name`

Per un esempio, consulta Crea un esempio di attività di caricamento in batch

Descrivi l'operazione di caricamento batch

È possibile recuperare la descrizione dell'attività di caricamento in batch come segue.

```
aws timestream-write describe-batch-load-task --task-id <value>
```

Di seguito è riportata una risposta di esempio:

```
{
    "BatchLoadTaskDescription": {
        "TaskId": "<TaskId>",
        "DataSourceConfiguration": {
            "DataSourceS3Configuration": {
                "BucketName": "test-batch-load-west-2",
                "ObjectKeyPrefix": "sample.csv"
            },
            "CsvConfiguration": {},
            "DataFormat": "CSV"
        },
        "ProgressReport": {
            "RecordsProcessed": 2,
            "RecordsIngested": 0,
            "FileParseFailures": 0,
            "RecordIngestionFailures": 2,
            "FileFailures": 0,
            "BytesIngested": 119
        },
        "ReportConfiguration": {
            "ReportS3Configuration": {
                "BucketName": "test-batch-load-west-2",
                "ObjectKeyPrefix": "<ObjectKeyPrefix>",
                "EncryptionOption": "SSE_S3"
            }
        },
        "DataModelConfiguration": {
            "DataModel": {
                "TimeColumn": "timestamp",
```

```
                "TimeUnit": "SECONDS",
                "DimensionMappings": [
                    {
                        "SourceColumn": "vehicle",
                        "DestinationColumn": "vehicle"
                    },
                    {
                        "SourceColumn": "registration",
                        "DestinationColumn": "license"
                    }
                ],
                "MultiMeasureMappings": {
                    "TargetMultiMeasureName": "test",
                    "MultiMeasureAttributeMappings": [
                        {
                            "SourceColumn": "wgt",
                            "TargetMultiMeasureAttributeName": "weight",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "SourceColumn": "spd",
                            "TargetMultiMeasureAttributeName": "speed",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "SourceColumn": "fuel",
                            "TargetMultiMeasureAttributeName": "fuel",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "SourceColumn": "miles",
                            "TargetMultiMeasureAttributeName": "miles",
                            "MeasureValueType": "DOUBLE"
                        }
                    ]
                }
            }
        },
        "TargetDatabaseName": "BatchLoadExampleDatabase",
        "TargetTableName": "BatchLoadExampleTable",
        "TaskStatus": "FAILED",
        "RecordVersion": 1,
        "CreationTime": 1677167593.266,
        "LastUpdatedTime": 1677167602.38
```

```
    }
}
```

Elenca le attività di caricamento in batch

È possibile elencare le attività di caricamento in batch come segue.

```
aws timestream-write list-batch-load-tasks
```

L'output viene visualizzato come segue.

```
{
    "BatchLoadTasks": [
        {
            "TaskId": "<TaskId>",
            "TaskStatus": "FAILED",
            "DatabaseName": "BatchLoadExampleDatabase",
            "TableName": "BatchLoadExampleTable",
            "CreationTime": 1677167593.266,
            "LastUpdatedTime": 1677167602.38
        }
    ]
}
```

Riprendi l'operazione di caricamento in batch

È possibile riprendere un'attività di caricamento in batch come segue.

```
aws timestream-write resume-batch-load-task --task-id <value>
```

Una risposta può indicare un successo o contenere informazioni sull'errore.

Crea un esempio di attività di caricamento in batch

Example

1. Crea un flusso temporale per il LiveAnalytics database denominato `BatchLoad` e una tabella denominata. `BatchLoadTest` Verifica e, se necessario, regola i valori di `MemoryStoreRetentionPeriodInHours` e. `MagneticStoreRetentionPeriodInDays`

```
aws timestream-write create-database --database-name BatchLoad \

aws timestream-write create-table --database-name BatchLoad \
--table-name BatchLoadTest \
--retention-properties "{\"MemoryStoreRetentionPeriodInHours\": 12,
 \"MagneticStoreRetentionPeriodInDays\": 100}"
```

2. Utilizzando la console, crea un bucket S3 e copia il `sample.csv` file in quella posizione. Puoi scaricare un esempio da sampleCSV. CSV
3. Utilizzando la console, create un bucket S3 per Timestream per LiveAnalytics scrivere un rapporto se l'attività di caricamento in batch viene completata con errori.
4. Crea un'attività di caricamento in batch. Assicurati di sostituire *$INPUT_BUCKET* e *$REPORT_BUCKET* con i bucket che hai creato nei passaggi precedenti.

```
aws timestream-write create-batch-load-task \
--data-model-configuration "{\
            \"DataModel\": {\
              \"TimeColumn\": \"timestamp\",\
              \"TimeUnit\": \"SECONDS\",\
              \"DimensionMappings\": [\
                {\
                  \"SourceColumn\": \"vehicle\"\
                },\
                {\
                  \"SourceColumn\": \"registration\",\
                  \"DestinationColumn\": \"license\"\
                }\
              ],
              \"MultiMeasureMappings\": {\
                \"TargetMultiMeasureName\": \"mva_measure_name\",\
                \"MultiMeasureAttributeMappings\": [\
                  {\
                    \"SourceColumn\": \"wgt\",\
                    \"TargetMultiMeasureAttributeName\": \"weight\",\
                    \"MeasureValueType\": \"DOUBLE\"\
                  },\
                  {\
                    \"SourceColumn\": \"spd\",\
                    \"TargetMultiMeasureAttributeName\": \"speed\",\
                    \"MeasureValueType\": \"DOUBLE\"\
                  },\
```

```
                  {\
                    \"SourceColumn\": \"fuel_consumption\",\
                    \"TargetMultiMeasureAttributeName\": \"fuel\",\
                    \"MeasureValueType\": \"DOUBLE\"\
                  },\
                  {\
                    \"SourceColumn\": \"miles\",\
                    \"MeasureValueType\": \"BIGINT\"\
                  }\
                ]\
              }\
            }\
          }" \
--data-source-configuration "{
            \"DataSourceS3Configuration\": {\
              \"BucketName\": \"$INPUT_BUCKET\",\
              \"ObjectKeyPrefix\": \"$INPUT_OBJECT_KEY_PREFIX\"
            },\
            \"DataFormat\": \"CSV\"\
          }" \
--report-configuration "{\
            \"ReportS3Configuration\": {\
              \"BucketName\": \"$REPORT_BUCKET\",\
              \"EncryptionOption\": \"SSE_S3\"\
            }\
          }" \
--target-database-name BatchLoad \
--target-table-name BatchLoadTest
```

Il comando precedente restituisce l'output seguente.

```
{
    "TaskId": "TaskId "
}
```

5. Controlla lo stato di avanzamento dell'operazione. Assicurati di sostituire *$TASK_ID* con l'id dell'attività restituito nel passaggio precedente.

```
aws timestream-write describe-batch-load-task --task-id $TASK_ID
```

Output di esempio


```
{
    "BatchLoadTaskDescription": {
        "ProgressReport": {
            "BytesIngested": 1024,
            "RecordsIngested": 2,
            "FileFailures": 0,
            "RecordIngestionFailures": 0,
            "RecordsProcessed": 2,
            "FileParseFailures": 0
        },
        "DataModelConfiguration": {
            "DataModel": {
                "DimensionMappings": [
                    {
                        "SourceColumn": "vehicle",
                        "DestinationColumn": "vehicle"
                    },
                    {
                        "SourceColumn": "registration",
                        "DestinationColumn": "license"
                    }
                ],
                "TimeUnit": "SECONDS",
                "TimeColumn": "timestamp",
                "MultiMeasureMappings": {
                    "MultiMeasureAttributeMappings": [
                        {
                            "TargetMultiMeasureAttributeName": "weight",
                            "SourceColumn": "wgt",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "TargetMultiMeasureAttributeName": "speed",
                            "SourceColumn": "spd",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "TargetMultiMeasureAttributeName": "fuel",
                            "SourceColumn": "fuel_consumption",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
```

```
                            "TargetMultiMeasureAttributeName": "miles",
                            "SourceColumn": "miles",
                            "MeasureValueType": "DOUBLE"
                        }
                    ],
                    "TargetMultiMeasureName": "mva_measure_name"
                }
            }
        },
        "TargetDatabaseName": "BatchLoad",
        "CreationTime": 1672960381.735,
        "TaskStatus": "SUCCEEDED",
        "RecordVersion": 1,
        "TaskId": "TaskId ",
        "TargetTableName": "BatchLoadTest",
        "ReportConfiguration": {
            "ReportS3Configuration": {
                "EncryptionOption": "SSE_S3",
                "ObjectKeyPrefix": "ObjectKeyPrefix ",
                "BucketName": "test-report-bucket"
            }
        },
        "DataSourceConfiguration": {
            "DataSourceS3Configuration": {
                "ObjectKeyPrefix": "sample.csv",
                "BucketName": "test-input-bucket"
            },
            "DataFormat": "CSV",
            "CsvConfiguration": {}
        },
        "LastUpdatedTime": 1672960387.334
    }
}
```

Utilizzo del caricamento in batch con AWS SDKs

Per esempi su come creare, descrivere ed elencare le attività di caricamento in batch con AWS SDKsCrea attività di caricamento in batch, vedereDescrizione dell'attività di caricamento in batch,Elenca le attività di caricamento in batch, eRiprendi l'operazione di caricamento in batch.

Utilizzo dei report sugli errori di caricamento in batch

Le attività di caricamento in batch hanno uno dei seguenti valori di stato:

- `CREATED`(Creato): l'attività viene creata.
- `IN_PROGRESS`(In corso): l'attività è in corso.
- `FAILED`(Non riuscito): l'attività è stata completata. Ma sono stati rilevati uno o più errori.
- `SUCCEEDED`(Completata): l'attività è stata completata senza errori.
- `PROGRESS_STOPPED`(Progresso interrotto): l'attività è stata interrotta ma non completata. È possibile tentare di riprendere l'attività.
- `PENDING_RESUME`(Curriculum in sospeso): l'attività è in attesa di ripresa.

In caso di errori, viene creato un report di registro degli errori nel bucket S3 definito a tale scopo. Gli errori sono classificati come taskErrors o fileErrors in matrici separate. Di seguito è riportato un esempio di segnalazione degli errori.

```
{
    "taskId": "9367BE28418C5EF902676482220B631C",
    "taskErrors": [],
    "fileErrors": [
        {
            "fileName": "example.csv",
            "errors": [
                {
                    "reason": "The record timestamp is outside the time range of the
 data ingestion window.",
                    "lineRanges": [
                        [
                            2,
                            3
                        ]
                    ]
                }
            ]
        }
    ]
}
```

Utilizzo delle interrogazioni pianificate in Timestream per LiveAnalytics

La funzionalità di interrogazione pianificata di Amazon Timestream LiveAnalytics for è una soluzione completamente gestita, serverless e scalabile per il calcolo e l'archiviazione di aggregati, rollup e altre forme di dati preelaborati tipicamente utilizzati per dashboard operativi, report aziendali, analisi ad hoc e altre applicazioni. Le interrogazioni pianificate rendono l'analisi in tempo reale più performante ed economica, in modo da poter ricavare ulteriori informazioni dai dati e continuare a prendere decisioni aziendali migliori.

Con le query pianificate, definisci le query di analisi in tempo reale che calcolano aggregati, rollup e altre operazioni sui dati, mentre Amazon Timestream esegue LiveAnalytics periodicamente e automaticamente queste query e scrive in modo affidabile i risultati delle query in una tabella separata. I dati vengono generalmente calcolati e aggiornati in queste tabelle entro pochi minuti.

È quindi possibile indirizzare i dashboard e i report in modo da interrogare le tabelle che contengono dati aggregati anziché interrogare le tabelle di origine, notevolmente più grandi. Ciò porta a miglioramenti in termini di prestazioni e costi che possono superare ordini di grandezza. Questo perché le tabelle con dati aggregati contengono molti meno dati rispetto alle tabelle di origine, quindi offrono query più veloci e un'archiviazione dei dati più economica.

Inoltre, le tabelle con interrogazioni pianificate offrono tutte le funzionalità esistenti di un Timestream for table. LiveAnalytics Ad esempio, puoi interrogare le tabelle utilizzando. SQL È possibile visualizzare i dati memorizzati nelle tabelle utilizzando Grafana. Puoi anche inserire dati nella tabella utilizzando Amazon Kinesis, AmazonMSK, AWS IoT Core e Telegraf. Puoi configurare le politiche di conservazione dei dati su queste tabelle per la gestione automatica del ciclo di vita dei dati.

Poiché la conservazione dei dati delle tabelle che contengono dati aggregati è completamente disaccoppiata da quella delle tabelle di origine, puoi anche scegliere di ridurre la conservazione dei dati delle tabelle di origine e conservare i dati aggregati per una durata molto più lunga, a una frazione del costo di archiviazione dei dati. Le query pianificate rendono l'analisi in tempo reale più veloce, economica e quindi più accessibile a molti più clienti, in modo che possano monitorare le loro applicazioni e prendere decisioni aziendali migliori basate sui dati.

Argomenti

- Esempio: utilizzo di analisi in tempo reale per rilevare pagamenti fraudolenti e prendere decisioni aziendali migliori
- Concetti di interrogazione pian
- Pianifica le espressioni per le interrogazioni pianificate
- Mappature dei modelli di dati per le interrogazioni pianificate
- Messaggi di notifica delle interrogazioni pianificate
- Rapporti sugli errori delle query pianificate
- Modelli ed esempi di interrogazioni pianificate

Vantaggi delle interrogazioni pianificate

Di seguito sono riportati i vantaggi delle interrogazioni pianificate:

- Facilità operativa: le query pianificate sono senza server e completamente gestite.
- Prestazioni e costi: poiché le query pianificate precalcolano gli aggregati, i rollup o altre operazioni di analisi in tempo reale dei dati e archiviano i risultati in una tabella, le query che accedono alle tabelle popolate da query pianificate contengono meno dati rispetto alle tabelle di origine. Pertanto, le query eseguite su queste tabelle sono più veloci ed economiche. Le tabelle popolate da calcoli pianificati contengono meno dati rispetto alle tabelle di origine e quindi aiutano a ridurre i costi di archiviazione. È inoltre possibile conservare questi dati per un periodo più lungo nell'archivio di memoria a una frazione del costo di conservazione dei dati di origine nell'archivio di memoria.
- Interoperabilità: le tabelle popolate da query pianificate offrono tutte le funzionalità esistenti di Timestream per le LiveAnalytics tabelle e possono essere utilizzate con tutti i servizi e gli strumenti compatibili con Timestream for. LiveAnalytics Vedi Lavorare con altri servizi per i dettagli.

Casi d'uso delle interrogazioni pianificate

È possibile utilizzare le query pianificate per i report aziendali che riepilogano l'attività dell'utente finale delle applicazioni, in modo da poter addestrare modelli di machine learning per la personalizzazione. Puoi anche utilizzare le query pianificate per gli allarmi che rilevano anomalie, intrusioni di rete o attività fraudolente, in modo da poter intraprendere azioni correttive immediate.

Inoltre, puoi utilizzare le query pianificate per una governance dei dati più efficace. Puoi farlo concedendo l'accesso alla tabella di origine esclusivamente alle query pianificate e fornendo agli

sviluppatori l'accesso solo alle tabelle popolate dalle query pianificate. Ciò riduce al minimo l'impatto delle query involontarie e di lunga durata.

Esempio: utilizzo di analisi in tempo reale per rilevare pagamenti fraudolenti e prendere decisioni aziendali migliori

Prendiamo in considerazione un sistema di pagamento che elabori le transazioni inviate da più point-of-sale terminali distribuiti nelle principali città metropolitane degli Stati Uniti. Vuoi utilizzare Amazon Timestream LiveAnalytics per archiviare e analizzare i dati delle transazioni, in modo da poter rilevare transazioni fraudolente ed eseguire query di analisi in tempo reale. Queste domande possono aiutarti a rispondere a domande aziendali come identificare i point-of-sale terminali più trafficati e meno utilizzati all'ora, l'ora più trafficata del giorno per ogni città e la città con il maggior numero di transazioni all'ora.

Il sistema elabora circa 100.000 transazioni al minuto. Ogni transazione memorizzata in Amazon Timestream è di 100 LiveAnalytics byte. Hai configurato 10 query che vengono eseguite ogni minuto per rilevare vari tipi di pagamenti fraudolenti. Hai anche creato 25 query che aggregano e suddividono i dati in varie dimensioni per aiutarti a rispondere alle tue domande aziendali. Ognuna di queste query elabora i dati dell'ultima ora.

Hai creato una dashboard per visualizzare i dati generati da queste query. La dashboard contiene 25 widget, viene aggiornata ogni ora e in genere è accessibile da 10 utenti in un dato momento. Infine, l'archivio di memoria è configurato con un periodo di conservazione dei dati di 2 ore e l'archivio magnetico è configurato per avere un periodo di conservazione dei dati di 6 mesi.

In questo caso, puoi utilizzare query di analisi in tempo reale che ricalcolano i dati ogni volta che si accede e si aggiorna la dashboard, oppure utilizzare tabelle derivate per la dashboard. Il costo delle query per i dashboard basati su query di analisi in tempo reale sarà di 120,70 dollari al mese. Al contrario, il costo delle query di dashboard basate su tabelle derivate sarà di 12,27 dollari al mese (consulta Amazon Timestream per i prezzi). LiveAnalytics In questo caso, l'utilizzo di tabelle derivate riduce il costo delle query di circa 10 volte.

Concetti di interrogazione pian

Stringa di interrogazione: questa è l'interrogazione il cui risultato viene precalcolato e memorizzato in un altro Timestream per tabella. LiveAnalytics È possibile definire un'interrogazione pianificata utilizzando l'intera SQL area di Timestream for LiveAnalytics, che offre la flessibilità di scrivere query con espressioni di tabella comuni, query annidate, funzioni di finestra o qualsiasi tipo di funzione aggregata e scalare supportata da Timestream per il linguaggio di query. LiveAnalytics

Espressione di pianificazione: consente di specificare quando vengono eseguite le istanze di query pianificate. È possibile specificare le espressioni utilizzando un'espressione cron (ad esempio, esegui alle 8:00 UTC ogni giorno) o un'espressione di frequenza (come esegui ogni 10 minuti).

Configurazione della destinazione: consente di specificare come mappare il risultato di una query pianificata nella tabella di destinazione in cui verranno archiviati i risultati di questa query pianificata.

Configurazione delle notifiche: Timestream for esegue LiveAnalytics automaticamente le istanze di una query pianificata in base all'espressione di pianificazione. Riceverai una notifica per ogni interrogazione di questo tipo eseguita su un SNS argomento che configuri quando crei un'interrogazione pianificata. Questa notifica specifica se l'istanza è stata eseguita correttamente o se si sono verificati errori. Inoltre, fornisce informazioni come i byte misurati, i dati scritti nella tabella di destinazione, l'ora di chiamata successiva e così via.

Di seguito è riportato un esempio di questo tipo di messaggio di notifica.

```
{
    "type":"AUTO_TRIGGER_SUCCESS",
    "arn":"arn:aws:timestream:us-east-1:123456789012:scheduled-query/
 PT1mPerMinutePerRegionMeasureCount-9376096f7309",
    "nextInvocationEpochSecond":1637302500,
    "scheduledQueryRunSummary":
    {
        "invocationEpochSecond":1637302440,
        "triggerTimeMillis":1637302445697,
        "runStatus":"AUTO_TRIGGER_SUCCESS",
        "executionStats":
        {
            "executionTimeInMillis":21669,
            "dataWrites":36864,
            "bytesMetered":13547036820,
            "recordsIngested":1200,
            "queryResultRows":1200
        }
    }
}
```

In questo messaggio di notifica, `bytesMetered` sono i byte scansionati dalla query nella tabella di origine e dataWrites i byte scritti nella tabella di destinazione.

> **Note**
>
> Se utilizzi queste notifiche a livello di codice, tieni presente che in futuro potrebbero essere aggiunti nuovi campi al messaggio di notifica.

Posizione del rapporto di errore: le query pianificate vengono eseguite in modo asincrono e memorizzano i dati nella tabella di destinazione. Se un'istanza rileva errori (ad esempio dati non validi che non è stato possibile archiviare), i record in cui si sono verificati errori vengono scritti in un report di errore nella posizione specificata al momento della creazione di una query pianificata. Devi specificare il bucket S3 e il prefisso per la posizione. Timestream for LiveAnalytics aggiunge il nome della query pianificata e l'ora di invocazione a questo prefisso per aiutarti a identificare gli errori associati a un'istanza specifica di una query pianificata.

Etichettatura: è possibile specificare facoltativamente tag da associare a una query pianificata. Per maggiori dettagli, consulta Tagging Timestream for Resources. LiveAnalytics

Esempio

Nell'esempio seguente, si calcola un aggregato semplice utilizzando una query pianificata:

```
SELECT region, bin(time, 1m) as minute,
    SUM(CASE WHEN measure_name = 'metrics' THEN 20 ELSE 5 END) as numDataPoints
FROM raw_data.devops
WHERE time BETWEEN @scheduled_runtime - 10m AND @scheduled_runtime + 1m
GROUP BY bin(time, 1m), region
```

`@scheduled_runtime parameter`- In questo esempio, noterai che la query accetta uno speciale parametro denominato. `@scheduled_runtime` Si tratta di un parametro speciale (di tipo Timestamp) che il servizio imposta quando richiama un'istanza specifica di una query pianificata in modo da poter controllare in modo deterministico l'intervallo di tempo per il quale un'istanza specifica di una query pianificata analizza i dati nella tabella di origine. Puoi utilizzarlo `@scheduled_runtime` nella tua query in qualsiasi posizione in cui è previsto un tipo di timestamp.

Consideriamo un esempio in cui si imposta un'espressione di pianificazione: cron (0/5 * * *? *) dove la query pianificata verrà eseguita al minuto 0, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55 di ogni ora. Per l'istanza attivata il 2021-12-01 00:05:00, il parametro @scheduled_runtime viene inizializzato su questo valore, in modo che l'istanza in questo momento operi su dati compresi tra 2021-11-30 23:55:00 e 2021-12-01 00:06:00.

Istanze con intervalli di tempo sovrapposti: come vedrai in questo esempio, due istanze successive di una query pianificata possono sovrapporsi nei rispettivi intervalli di tempo. Questo è qualcosa che puoi controllare in base ai tuoi requisiti, ai predicati temporali specificati e all'espressione di pianificazione. In questo caso, questa sovrapposizione consente a questi calcoli di aggiornare gli aggregati sulla base di tutti i dati il cui arrivo è stato leggermente ritardato, fino a 10 minuti in questo esempio. L'esecuzione della query attivata alle 00:00:00 del 2021-12-01 coprirà l'intervallo di tempo 2021-11-30 23:50:00 al 2021-12-30 00:01:00 e l'esecuzione della query attivata al 2021-12-01 00:05:00 coprirà l'intervallo dal 2021-11-30 23:55:00 al 01/12/2021 00:06:00.

Per garantire la correttezza e assicurarsi che gli aggregati archiviati nella tabella di destinazione corrispondano agli aggregati calcolati dalla tabella di origine, Timestream for LiveAnalytics assicura che il calcolo al 01/12/2021 00:05:00 verrà eseguito solo dopo il completamento del calcolo del 01/12/2021 00:00. I risultati di questi ultimi calcoli possono aggiornare qualsiasi aggregato precedentemente materializzato se viene generato un valore più recente. Internamente, Timestream for LiveAnalytics utilizza versioni record in cui ai record generati dalle ultime istanze di una query pianificata verrà assegnato un numero di versione più elevato. Pertanto, gli aggregati calcolati dalla chiamata alle 00:05:00 del 01/12/2021 possono aggiornare gli aggregati calcolati dalla chiamata alle 00:00:00del 01/12/2021, supponendo che i dati più recenti siano disponibili nella tabella di origine.

Trigger automatici e trigger manuali: dopo la creazione di una query pianificata, Timestream for eseguirà automaticamente le istanze in base alla pianificazione specificata. LiveAnalytics Tali trigger automatici sono gestiti interamente dal servizio.

Tuttavia, potrebbero esserci scenari in cui potresti voler avviare manualmente alcune istanze di una query pianificata. Ad esempio, se un'istanza specifica ha avuto esito negativo nell'esecuzione di una query, se sono arrivati dati o aggiornamenti in ritardo nella tabella di origine dopo l'esecuzione automatica della pianificazione o se si desidera aggiornare la tabella di destinazione per intervalli di tempo non coperti dalle esecuzioni automatiche di query (ad esempio, per gli intervalli di tempo prima della creazione di una query pianificata).

È possibile utilizzare il ExecuteScheduledQuery API per avviare manualmente un'istanza specifica di una query pianificata passando il InvocationTime parametro, che è un valore utilizzato per il parametro @scheduled_runtime. Di seguito sono riportate alcune considerazioni importanti relative all'utilizzo di: ExecuteScheduledQuery API

- Se state attivando più di queste chiamate, dovete assicurarvi che queste invocazioni non generino risultati in intervalli di tempo sovrapposti. Se non riesci a garantire che gli intervalli di tempo non si sovrappongano, assicurati che queste esecuzioni di query vengano avviate in sequenza una dopo l'altra. Se avviate contemporaneamente più esecuzioni di query che si sovrappongono nei

rispettivi intervalli di tempo, nei report sugli errori relativi a queste esecuzioni di query è possibile che si verifichino conflitti di versione.

- È possibile avviare le chiamate con qualsiasi valore di timestamp per @scheduled_runtime. Quindi è tua responsabilità impostare i valori in modo appropriato in modo che gli intervalli di tempo appropriati vengano aggiornati nella tabella di destinazione corrispondente agli intervalli in cui i dati sono stati aggiornati nella tabella di origine.

Pianifica le espressioni per le interrogazioni pianificate

Puoi creare query pianificate in base a una pianificazione automatica utilizzando Amazon LiveAnalytics Timestream per query pianificate che utilizzano espressioni cron o rate. Tutte le query pianificate utilizzano il fuso UTC orario e la precisione minima possibile per le pianificazioni è di 1 minuto.

Due modi per specificare le espressioni di pianificazione sono cron e rate. Le espressioni cron offrono un controllo più preciso della pianificazione, mentre le espressioni rate sono più semplici da esprimere ma non dispongono di un controllo granulare.

Ad esempio, con un'espressione cron, puoi definire una query pianificata che viene attivata a un'ora specifica in un determinato giorno di ogni settimana o mese, o un minuto specifico ogni ora solo dal lunedì al venerdì e così via. Al contrario, le espressioni rate avviano una query pianificata a una frequenza regolare, ad esempio una volta ogni minuto, ora o giorno, a partire dall'ora esatta in cui viene creata la query pianificata.

Espressione cron

- Sintassi

```
cron(fields)
```

  Le espressioni Cron hanno sei campi obbligatori separati da uno spazio vuoto.

Campo	Valori	Caratteri jolly
Minuti	0-59	, - * /
Ore	0-23	, - * /

Campo	Valori	Caratteri jolly
D ay-of-month	1-31	, - * ? / L W
Mese	1-12 o JAN - DEC	, - * /
D ay-of-week	1-7 o SUN - SAT	, - * ? L #
Anno	1970-2199	, - * /

Caratteri jolly

- Il carattere jolly *, * (virgola) include valori aggiuntivi. Nel campo Mese,JAN,FEB, MAR includerebbe gennaio, febbraio e marzo.
- Il carattere jolly *-* (trattino) specifica gli intervalli. Nel campo Day (Giorno), 1-15 include i giorni dall'1 al 15 del mese specificato.
- Il carattere jolly *** (asterisco) include tutti i valori del campo. Nel campo Ore, *** includerebbe ogni ora. Non è possibile utilizzare *** in entrambi i Day-of-week campi Day-of-month e. Se lo usi in uno, devi usare*? * nell'altra.
- Il carattere jolly */* (barra in avanti) specifica gli incrementi. Nel campo Minuti, è possibile inserire 1/10 per specificare ogni decimo minuto, a partire dal primo minuto dell'ora (ad esempio, l'undicesimo, il ventunesimo e il 31° minuto e così via).
- Il*? * (punto interrogativo) il carattere jolly specifica l'uno o l'altro. Nel Day-of-month campo puoi inserire *7* e se non ti interessa in che giorno della settimana è il 7, puoi inserire*? * nel campo. Day-of-week
- Il carattere jolly *L* nei Day-of-week campi Day-of-month or specifica l'ultimo giorno del mese o della settimana.
- Il carattere jolly W nel campo specifica un giorno della settimana Day-of-month. Nel Day-of-month campo, 3W specifica il giorno della settimana più vicino al terzo giorno del mese.
- Il carattere jolly *#* nel Day-of-week campo specifica una determinata istanza del giorno della settimana specificato all'interno di un mese. Ad esempio, 3#2 sarebbe il secondo martedì del mese: il 3 fa riferimento a martedì perché è il terzo giorno di ogni settimana e il 2 fa riferimento al secondo giorno di questo tipo in un mese.

> **Note**
>
> Se si utilizza un carattere '#', è possibile definire solo un'espressione nel campo. day-of-week Ad esempio, "3#1,6#3" non è valido perché viene interpretato come due espressioni.

Limitazioni

- Non è possibile specificare i Day-of-week campi Day-of-month and nella stessa espressione cron. Se specifichi un valore (o un *) in uno dei campi, devi usare un *? * (punto interrogativo) nell'altro.
- Le espressioni Cron che indicano frequenze più rapide di 1 minuto non sono supportate.

Examples (Esempi)

Minuti	Ore	Giorno del mese	Mese	Giorno della settimana	Anno	Significato
0	10	*	*	?	*	Corri alle 10:00 (UTC) tutti i giorni.
15	12	*	*	?	*	Corri alle 12:15 (UTC) ogni giorno.
0	18	?	*	MON-FRI	*	Corri alle 18:00 (UTC) dal lunedì al venerdì.

Minuti	Ore	Giorno del mese	Mese	Giorno della settimana	Anno	Significato
0	8	1	*	?	*	Esegui alle 8:00 (UTC) ogni primo giorno del mese.
0/15	*	*	*	?	*	Corri ogni 15 minuti.
0/10	*	*	*	MON-FRI	*	Corri ogni 10 minuti dal lunedì al venerdì.
0/5	8-17	?	*	MON-FRI	*	Corri ogni 5 minuti dal lunedì al venerdì tra le 8:00 e le 17:55 ()UTC.

Espressioni della frequenza

- Un espressione rate inizia quando crei la regola di evento pianificato e successivamente la esegui nella relativa pianificazione definita. Le espressioni rate hanno due campi obbligatori. I campi sono separati da uno spazio vuoto.

  Sintassi

  ```
  rate(value unit)
  ```

  - `value`: Un numero positivo.

- `unit`: L'unità di tempo. Sono necessarie unità diverse per i valori di 1 (ad esempio, minuto) e i valori superiori a 1 (ad esempio, minuti). Valori validi: minuto | minuti | ora | ore | giorno | giorni

Mappature dei modelli di dati per le interrogazioni pianificate

Timestream for LiveAnalytics supporta la modellazione flessibile dei dati nelle sue tabelle e questa stessa flessibilità si applica ai risultati delle query pianificate che vengono materializzate in un altro Timestream per tabella. LiveAnalytics Con le interrogazioni pianificate, è possibile eseguire query su qualsiasi tabella, indipendentemente dal fatto che contenga dati in record a più misure o record di singola misura, e scrivere i risultati della query utilizzando record a più o a misura singola.

È possibile utilizzare il TargetConfiguration nella specifica di un'interrogazione pianificata per mappare i risultati della query alle colonne appropriate nella tabella derivata di destinazione. Le sezioni seguenti descrivono i diversi modi di TargetConfiguration specificarlo per ottenere modelli di dati diversi nella tabella derivata. In particolare, vedrai:

- Come scrivere su record multimisura quando il risultato della query non ha un nome di misura e si specifica il nome della TargetConfiguration misura di destinazione in.
- Come si utilizza il nome della misura nei risultati della query per scrivere record con più misure.
- Come definire un modello per scrivere più record con diversi attributi multimisura.
- Come definire un modello per scrivere su record di singola misura nella tabella derivata.
- In che modo è possibile interrogare record di singola misura e/o record multimisura in un'interrogazione pianificata e far sì che i risultati vengano materializzati in un record a misura singola o in un record a più misure, il che consente di scegliere la flessibilità dei modelli di dati.

Esempio: nome della misura di destinazione per i record multimisura

In questo esempio, si vedrà che la query sta leggendo i dati da una tabella con dati a più misure e sta scrivendo i risultati in un'altra tabella utilizzando record multimisura. Il risultato della query pianificata non ha una colonna con il nome della misura naturale. Qui, si specifica il nome della misura nella tabella derivata utilizzando la TargetMultiMeasureName proprietà in TargetConfiguration. TimestreamConfiguration.

```
{
    "Name" : "CustomMultiMeasureName",
    "QueryString" : "SELECT region, bin(time, 1h) as hour, AVG(memory_cached)
 as avg_mem_cached_1h, MIN(memory_free) as min_mem_free_1h, MAX(memory_used) as
```

```
max_mem_used_1h, SUM(disk_io_writes) as sum_1h, AVG(disk_used) as avg_disk_used_1h,
AVG(disk_free) as avg_disk_free_1h, MAX(cpu_user) as max_cpu_user_1h, MIN(cpu_idle) as
min_cpu_idle_1h, MAX(cpu_system) as max_cpu_system_1h FROM raw_data.devops_multi WHERE
time BETWEEN bin(@scheduled_runtime, 1h) - 14h AND bin(@scheduled_runtime, 1h) - 2h
AND measure_name = 'metrics' GROUP BY region, bin(time, 1h)",
   "ScheduleConfiguration" : {
       "ScheduleExpression" : "cron(0 0/1 * * ? *)"
   },
   "NotificationConfiguration" : {
       "SnsConfiguration" : {
           "TopicArn" : "******"
       }
   },
   "ScheduledQueryExecutionRoleArn": "******",
   "TargetConfiguration": {
       "TimestreamConfiguration": {
           "DatabaseName" : "derived",
           "TableName" : "dashboard_metrics_1h_agg_1",
           "TimeColumn" : "hour",
           "DimensionMappings" : [
               {
                   "Name": "region",
                   "DimensionValueType" : "VARCHAR"
               }
           ],
           "MultiMeasureMappings" : {
               "TargetMultiMeasureName": "dashboard-metrics",
               "MultiMeasureAttributeMappings" : [
                   {
                       "SourceColumn" : "avg_mem_cached_1h",
                       "MeasureValueType" : "DOUBLE",
                       "TargetMultiMeasureAttributeName" : "avgMemCached"
                   },
                   {
                       "SourceColumn" : "min_mem_free_1h",
                       "MeasureValueType" : "DOUBLE"
                   },
                   {
                       "SourceColumn" : "max_mem_used_1h",
                       "MeasureValueType" : "DOUBLE"
                   },
                   {
                       "SourceColumn" : "sum_1h",
                       "MeasureValueType" : "DOUBLE",
```

```
                        "TargetMultiMeasureAttributeName" : "totalDiskWrites"
                    },
                    {
                        "SourceColumn" : "avg_disk_used_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "avg_disk_free_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "max_cpu_user_1h",
                        "MeasureValueType" : "DOUBLE",
                        "TargetMultiMeasureAttributeName" : "CpuUserP100"
                    },
                    {
                        "SourceColumn" : "min_cpu_idle_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "max_cpu_system_1h",
                        "MeasureValueType" : "DOUBLE",
                        "TargetMultiMeasureAttributeName" : "CpuSystemP100"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    }
}
```

La mappatura in questo esempio crea un record multimisura con il nome della misura dashboard-metrics e i nomi degli attributi, min_mem_free_1h, max_mem_used_1h, avg_disk_used_1h avgMemCached, avg_disk_free_1h, P100, min_cpu_idle_1h, P100. totalDiskWrites CpuUser CpuSystem Si noti l'uso opzionale di TargetMultiMeasureAttributeName per rinominare le colonne di output della query con un nome di attributo diverso utilizzato per la materializzazione dei risultati.

Di seguito è riportato lo schema per la tabella di destinazione una volta materializzata questa query pianificata. Come si può vedere dal Timestream per il tipo di LiveAnalytics attributo nel risultato seguente, i risultati vengono materializzati in un record multimisura con un nome a misura singola`dashboard-metrics`, come mostrato nello schema di misura.

Colonna	Tipo	Timestream per il tipo di attributo LiveAnalytics
Regione	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
time	timestamp	TIMESTAMP
CpuSystemP100	double	MULTI
avgMemCached	double	MULTI
min_cpu_idle_1h	double	MULTI
avg_disk_free_1h	double	MULTI
avg_disk_used_1h	double	MULTI
totalDiskWrites	double	MULTI
max_mem_used_1 h	double	MULTI
min_mem_free_1h	double	MULTI
CpuUserP100	double	MULTI

Di seguito sono riportate le misure corrispondenti ottenute con una SHOW MEASURES query.

measure_name	data_type	Dimensioni
metriche del pannello di controllo	multi	[{'dimension_name': 'regione', 'data_type': 'varchar'}]

Esempio: utilizzo del nome della misura da una query pianificata in record multimisura

In questo esempio, vedrete una query che legge da una tabella con record a misura singola e materializza i risultati in record a più misure. In questo caso, il risultato della query pianificata ha una colonna i cui valori possono essere utilizzati come nomi di misure nella tabella di destinazione in cui vengono materializzati i risultati dell'interrogazione pianificata. È quindi possibile specificare il nome della misura per il record multimisura nella tabella derivata utilizzando la MeasureNameColumn proprietà in. TargetConfiguration TimestreamConfiguration.

```
{
    "Name" : "UsingMeasureNameFromQueryResult",
    "QueryString" : "SELECT region, bin(time, 1h) as hour, measure_name, AVG(CASE WHEN
 measure_name IN ('memory_cached', 'disk_used', 'disk_free') THEN measure_value::double
 ELSE NULL END) as avg_1h, MIN(CASE WHEN measure_name IN ('memory_free', 'cpu_idle')
 THEN measure_value::double ELSE NULL END) as min_1h, SUM(CASE WHEN measure_name
 IN ('disk_io_writes') THEN measure_value::double ELSE NULL END) as sum_1h,
 MAX(CASE WHEN measure_name IN ('memory_used', 'cpu_user', 'cpu_system') THEN
 measure_value::double ELSE NULL END) as max_1h FROM raw_data.devops WHERE time
 BETWEEN bin(@scheduled_runtime, 1h) - 14h AND bin(@scheduled_runtime, 1h) - 2h AND
 measure_name IN ('memory_free', 'memory_used', 'memory_cached', 'disk_io_writes',
 'disk_used', 'disk_free', 'cpu_user', 'cpu_system', 'cpu_idle') GROUP BY region,
 measure_name, bin(time, 1h)",
    "ScheduleConfiguration" : {
        "ScheduleExpression" : "cron(0 0/1 * * ? *)"
    },
    "NotificationConfiguration" : {
        "SnsConfiguration" : {
            "TopicArn" : "******"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******",
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName" : "derived",
            "TableName" : "dashboard_metrics_1h_agg_2",
            "TimeColumn" : "hour",
            "DimensionMappings" : [
                {
                    "Name": "region",
                    "DimensionValueType" : "VARCHAR"
                }
            ],
            "MeasureNameColumn" : "measure_name",
```

```
            "MultiMeasureMappings" : {
                "MultiMeasureAttributeMappings" : [
                    {
                        "SourceColumn" : "avg_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "min_1h",
                        "MeasureValueType" : "DOUBLE",
                        "TargetMultiMeasureAttributeName": "p0_1h"
                    },
                    {
                        "SourceColumn" : "sum_1h",
                        "MeasureValueType" : "DOUBLE"
                    },
                    {
                        "SourceColumn" : "max_1h",
                        "MeasureValueType" : "DOUBLE",
                        "TargetMultiMeasureAttributeName": "p100_1h"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    }
}
```

La mappatura in questo esempio creerà record multimisura con gli attributi avg_1h, p0_1h, sum_1h, p100_1h e utilizzerà i valori della colonna measure_name nel risultato della query come nome della misura per i record multimisura nella tabella di destinazione. Si noti inoltre che gli esempi precedenti utilizzano facoltativamente il con un sottoinsieme delle mappature per rinominare gli attributi. TargetMultiMeasureAttributeName Ad esempio, min_1h è stato rinominato in p0_1h e max_1h è stato rinominato in p100_1h.

Di seguito è riportato lo schema per la tabella di destinazione una volta materializzata questa query pianificata. Come si può vedere dal Timestream per il tipo di LiveAnalytics attributo nel risultato

seguente, i risultati vengono materializzati in un record multimisura. Se si esamina lo schema di misura, sono stati inseriti nove diversi nomi di misure che corrispondono ai valori visualizzati nei risultati della query.

Colonna	Tipo	Timestream per il tipo di attributo LiveAnalytics
Regione	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
time	timestamp	TIMESTAMP
sum_1h	double	MULTI
p100_1h	double	MULTI
p0_1h	double	MULTI
avg_1h	double	MULTI

Di seguito sono riportate le misure corrispondenti ottenute con una query. SHOW MEASURES

measure_name	data_type	Dimensioni
cpu_idle	multi	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
sistema_cpu	multiplo	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
cpu_user	multiplo	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
disk_free	multi	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
disk_io_write	multi	[{'dimension_name': 'regione', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
disk_used	multi	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
memoria_memorizzata nella cache	multi	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
senza memoria	multiplo	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
senza memoria	multiplo	[{'dimension_name': 'regione', 'data_type': 'varchar'}]

Esempio: mappatura dei risultati su diversi record multimisura con attributi diversi

L'esempio seguente mostra come è possibile mappare diverse colonne del risultato della query in diversi record multimisura con nomi di misure diversi. Se viene visualizzata la seguente definizione di query pianificata, il risultato della query ha le seguenti colonne: region, hour, avg_mem_cached_1h, min_mem_free_1h, max_mem_used_1h, total_disk_io_writes_1h, avg_disk_used_1h, avg_disk_free_1h, max_cpu_utente_1h, sistema_cpu_max_1h, min_cpu_sistema_1h. `region`è mappato alla dimensione ed `hour` è mappato alla colonna del tempo.

La MixedMeasureMappings proprietà in. TargetConfiguration TimestreamConfigurationspecifica come mappare le misure su record multimisura nella tabella derivata.

In questo esempio specifico, avg_mem_cached_1h, min_mem_free_1h, max_mem_used_1h vengono utilizzati in un record multimisura con nome di misura mem_aggregates, total_disk_io_writes_1h, avg_disk_used_1h, avg_disk_free_1h vengono utilizzati in un altro record multimisura con misura name of disk_aggregates e infine max_cpu_user_1h, max_cpu_system_1h, min_cpu_system_1h vengono utilizzati in un altro record multimisura con nome di misura cpu_aggregates.

In queste mappature, è anche possibile utilizzare facoltativamente TargetMultiMeasureAttributeName per rinominare la colonna dei risultati della query in modo che abbia un nome di attributo diverso nella tabella di destinazione. Ad esempio, la colonna dei risultati avg_mem_cached_1h viene rinominata in, total_disk_io_writes_1h viene rinominata in, ecc. avgMemCached totalIOWrites

Quando si definiscono le mappature per i record multimisura, Timestream for ispeziona ogni riga dei risultati della query e ignora automaticamente i valori delle colonne con valori. LiveAnalytics NULL

Di conseguenza, nel caso di mappature con più nomi di misure, se tutti i valori di colonna per quel gruppo nella mappatura si riferiscono a una determinata riga, NULL per quella riga non viene inserito alcun valore per quel nome di misura.

Ad esempio, nella mappatura seguente, avg_mem_cached_1h, min_mem_free_1h e max_mem_used_1h vengono mappati per misurare il nome mem_aggregates. Se per una determinata riga del risultato della query, tutti questi valori della colonna sono uguali, Timestream for non inserirà la misura mem_aggregates per quella riga. NULL LiveAnalytics Se tutte e nove le colonne di una determinata riga lo sonoNULL, nella segnalazione degli errori verrà visualizzato un errore utente.

```
{
    "Name" : "AggsInDifferentMultiMeasureRecords",
    "QueryString" : "SELECT region, bin(time, 1h) as hour, AVG(CASE WHEN measure_name
 = 'memory_cached' THEN measure_value::double ELSE NULL END) as avg_mem_cached_1h,
 MIN(CASE WHEN measure_name = 'memory_free' THEN measure_value::double ELSE
 NULL END) as min_mem_free_1h, MAX(CASE WHEN measure_name = 'memory_used' THEN
 measure_value::double ELSE NULL END) as max_mem_used_1h, SUM(CASE WHEN measure_name =
 'disk_io_writes' THEN measure_value::double ELSE NULL END) as total_disk_io_writes_1h,
 AVG(CASE WHEN measure_name = 'disk_used' THEN measure_value::double ELSE NULL END) as
 avg_disk_used_1h, AVG(CASE WHEN measure_name = 'disk_free' THEN measure_value::double
 ELSE NULL END) as avg_disk_free_1h, MAX(CASE WHEN measure_name = 'cpu_user' THEN
 measure_value::double ELSE NULL END) as max_cpu_user_1h, MAX(CASE WHEN measure_name
 = 'cpu_system' THEN measure_value::double ELSE NULL END) as max_cpu_system_1h,
 MIN(CASE WHEN measure_name = 'cpu_idle' THEN measure_value::double ELSE NULL END)
 as min_cpu_system_1h FROM raw_data.devops WHERE time BETWEEN bin(@scheduled_runtime,
 1h) - 14h AND bin(@scheduled_runtime, 1h) - 2h AND measure_name IN ('memory_cached',
 'memory_free', 'memory_used', 'disk_io_writes', 'disk_used', 'disk_free', 'cpu_user',
 'cpu_system', 'cpu_idle') GROUP BY region, bin(time, 1h)",
    "ScheduleConfiguration" : {
        "ScheduleExpression" : "cron(0 0/1 * * ? *)"
    },
    "NotificationConfiguration" : {
        "SnsConfiguration" : {
            "TopicArn" : "******"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******",
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName" : "derived",
            "TableName" : "dashboard_metrics_1h_agg_3",
```

```
            "TimeColumn" : "hour",
            "DimensionMappings" : [
                {
                    "Name": "region",
                    "DimensionValueType" : "VARCHAR"
                }
            ],
            "MixedMeasureMappings" : [
                {
                    "MeasureValueType" : "MULTI",
                    "TargetMeasureName" : "mem_aggregates",
                    "MultiMeasureAttributeMappings" : [
                        {
                            "SourceColumn" : "avg_mem_cached_1h",
                            "MeasureValueType" : "DOUBLE",
                            "TargetMultiMeasureAttributeName": "avgMemCached"
                        },
                        {
                            "SourceColumn" : "min_mem_free_1h",
                            "MeasureValueType" : "DOUBLE"
                        },
                        {
                            "SourceColumn" : "max_mem_used_1h",
                            "MeasureValueType" : "DOUBLE",
                            "TargetMultiMeasureAttributeName": "maxMemUsed"
                        }
                    ]
                },
                {
                    "MeasureValueType" : "MULTI",
                    "TargetMeasureName" : "disk_aggregates",
                    "MultiMeasureAttributeMappings" : [
                        {
                            "SourceColumn" : "total_disk_io_writes_1h",
                            "MeasureValueType" : "DOUBLE",
                            "TargetMultiMeasureAttributeName": "totalIOWrites"
                        },
                        {
                            "SourceColumn" : "avg_disk_used_1h",
                            "MeasureValueType" : "DOUBLE"
                        },
                        {
                            "SourceColumn" : "avg_disk_free_1h",
                            "MeasureValueType" : "DOUBLE"
```

```
                        }
                    ]
                },
                {
                    "MeasureValueType" : "MULTI",
                    "TargetMeasureName" : "cpu_aggregates",
                    "MultiMeasureAttributeMappings" : [
                        {
                            "SourceColumn" : "max_cpu_user_1h",
                            "MeasureValueType" : "DOUBLE"
                        },
                        {
                            "SourceColumn" : "max_cpu_system_1h",
                            "MeasureValueType" : "DOUBLE"
                        },
                        {
                            "SourceColumn" : "min_cpu_idle_1h",
                            "MeasureValueType" : "DOUBLE",
                            "TargetMultiMeasureAttributeName": "minCpuIdle"
                        }
                    ]
                }
            ]
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    }
}
```

Di seguito è riportato lo schema per la tabella di destinazione una volta materializzata questa query pianificata.

Colonna	Tipo	Timestream per il tipo di attributo LiveAnalytics
Regione	varchar	DIMENSION

Colonna	Tipo	Timestream per il tipo di attributo LiveAnalytics
measure_name	varchar	MEASURE_NAME
time	timestamp	TIMESTAMP
minCpuIdle	double	MULTI
max_cpu_system_1h	double	MULTI
max_cpu_user_1 h	double	MULTI
avgMemCached	double	MULTI
maxMemUsed	double	MULTI
min_mem_free_1 h	double	MULTI
avg_disk_free_1h	double	MULTI
avg_disk_used_1h	double	MULTI
totalIOWrites	double	MULTI

Di seguito sono riportate le misure corrispondenti ottenute con una query. SHOW MEASURES

measure_name	data_type	Dimensioni
cpu_aggregates	multi	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
disk_aggregates	multi	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
mem_aggregates	multi	[{'dimension_name': 'regione', 'data_type': 'varchar'}]

Esempio: mappatura dei risultati su record a misura singola con il nome della misura dai risultati della query

Di seguito è riportato un esempio di interrogazione pianificata i cui risultati vengono materializzati in record a misura singola. In questo esempio, il risultato della query contiene la colonna measure_name i cui valori verranno utilizzati come nomi delle misure nella tabella di destinazione. Si utilizza l' MixedMeasureMappings attributo in. TargetConfiguration TimestreamConfiguration per specificare la mappatura della colonna dei risultati della query alla misura scalare nella tabella di destinazione.

Nella definizione dell'esempio seguente, il risultato della query è previsto in nove valori distinti di measure_name. Si elencano tutti i nomi di queste misure nella mappatura e si specifica quale colonna utilizzare per il valore di misura singola per quel nome di misura. Ad esempio, in questa mappatura, se il nome della misura di memory_cached viene visualizzato per una determinata riga di risultati, il valore nella colonna avg_1h viene utilizzato come valore per la misura quando i dati vengono scritti nella tabella di destinazione. Facoltativamente, è possibile utilizzare per fornire un nuovo nome di misura per TargetMeasureName questo valore.

```
{
    "Name" : "UsingMeasureNameColumnForSingleMeasureMapping",
    "QueryString" : "SELECT region, bin(time, 1h) as hour, measure_name, AVG(CASE WHEN
 measure_name IN ('memory_cached', 'disk_used', 'disk_free') THEN measure_value::double
 ELSE NULL END) as avg_1h, MIN(CASE WHEN measure_name IN ('memory_free', 'cpu_idle')
 THEN measure_value::double ELSE NULL END) as min_1h, SUM(CASE WHEN measure_name
 IN ('disk_io_writes') THEN measure_value::double ELSE NULL END) as sum_1h,
 MAX(CASE WHEN measure_name IN ('memory_used', 'cpu_user', 'cpu_system') THEN
 measure_value::double ELSE NULL END) as max_1h FROM raw_data.devops WHERE time
 BETWEEN bin(@scheduled_runtime, 1h) - 14h AND bin(@scheduled_runtime, 1h) - 2h AND
 measure_name IN ('memory_free', 'memory_used', 'memory_cached', 'disk_io_writes',
 'disk_used', 'disk_free', 'cpu_user', 'cpu_system', 'cpu_idle') GROUP BY region,
 bin(time, 1h), measure_name",
    "ScheduleConfiguration" : {
        "ScheduleExpression" : "cron(0 0/1 * * ? *)"
    },
    "NotificationConfiguration" : {
        "SnsConfiguration" : {
            "TopicArn" : "******"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******",
    "TargetConfiguration": {
```

```
        "TimestreamConfiguration": {
            "DatabaseName" : "derived",
            "TableName" : "dashboard_metrics_1h_agg_4",
            "TimeColumn" : "hour",
            "DimensionMappings" : [
                {
                    "Name": "region",
                    "DimensionValueType" : "VARCHAR"
                }
            ],
            "MeasureNameColumn" : "measure_name",
            "MixedMeasureMappings" : [
                {
                    "MeasureName" : "memory_cached",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "avg_1h",
                    "TargetMeasureName" : "AvgMemCached"
                },
                {
                    "MeasureName" : "disk_used",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "avg_1h"
                },
                {
                    "MeasureName" : "disk_free",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "avg_1h"
                },
                {
                    "MeasureName" : "memory_free",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "min_1h",
                    "TargetMeasureName" : "MinMemFree"
                },
                {
                    "MeasureName" : "cpu_idle",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "min_1h"
                },
                {
                    "MeasureName" : "disk_io_writes",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "sum_1h",
                    "TargetMeasureName" : "total-disk-io-writes"
```

```
                },
                {
                    "MeasureName" : "memory_used",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "max_1h",
                    "TargetMeasureName" : "maxMemUsed"
                },
                {
                    "MeasureName" : "cpu_user",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "max_1h"
                },
                {
                    "MeasureName" : "cpu_system",
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "max_1h"
                }
            ]
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    }
}
```

Di seguito è riportato lo schema per la tabella di destinazione una volta materializzata questa query pianificata. Come si può vedere dallo schema, la tabella utilizza record a misura singola. Se si elenca lo schema di misure per la tabella, verranno visualizzate le nove misure scritte in base alla mappatura fornita nelle specifiche.

Colonna	Tipo	Timestream per il tipo di attributo LiveAnalytics
Regione	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
time	timestamp	TIMESTAMP

Colonna	Tipo	Timestream per il tipo di attributo LiveAnalytics
measure_value::double	double	MEASURE_VALUE

Di seguito sono riportate le misure corrispondenti ottenute con una SHOW MEASURES query.

measure_name	data_type	Dimensioni
AvgMemCached	double	[{'dimension_name': 'region', 'data_type': 'varchar'}]
MinMemFree	double	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
cpu_idle	double	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
sistema_cpu	double	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
cpu_user	double	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
disk_free	double	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
disk_used	double	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
maxMemUsed	double	[{'dimension_name': 'regione', 'data_type': 'varchar'}]
total-disk-io-writes	double	[{'dimension_name': 'regione', 'data_type': 'varchar'}]

Esempio: mappatura dei risultati su record a misura singola con colonne di risultati di query come nomi di misure

In questo esempio, si dispone di una query i cui risultati non hanno una colonna con il nome della misura. Si desidera invece utilizzare il nome della colonna dei risultati della query come nome della misura quando si esegue la mappatura dell'output su record a misura singola. In precedenza c'era un esempio in cui un risultato simile veniva scritto su un record a più misure. In questo esempio, vedrete come mapparlo su record a misura singola, se ciò si adatta allo scenario dell'applicazione.

Ancora una volta, si specifica questa mappatura utilizzando la MixedMeasureMappings proprietà in. TargetConfiguration TimestreamConfiguration. Nell'esempio seguente, si vede che il risultato della query ha nove colonne. Le colonne dei risultati vengono utilizzate come nomi di misure e i valori come valori di singola misura.

Ad esempio, per una determinata riga del risultato della query, il nome di colonna avg_mem_cached_1h viene utilizzato come nome della colonna e valore associato alla colonna e avg_mem_cached_1h viene utilizzato come valore di misura per il record a misura singola. È inoltre possibile utilizzare TargetMeasureName un nome di misura diverso nella tabella di destinazione. Ad esempio, per i valori nella colonna sum_1h, la mappatura specifica di utilizzare total_disk_io_writes_1h come nome della misura nella tabella di destinazione. Se il valore di una NULL colonna è, la misura corrispondente viene ignorata.

```
{
    "Name" : "SingleMeasureMappingWithoutMeasureNameColumnInQueryResult",
    "QueryString" : "SELECT region, bin(time, 1h) as hour, AVG(CASE WHEN measure_name
 = 'memory_cached' THEN measure_value::double ELSE NULL END) as avg_mem_cached_1h,
 AVG(CASE WHEN measure_name = 'disk_used' THEN measure_value::double ELSE NULL END) as
 avg_disk_used_1h, AVG(CASE WHEN measure_name = 'disk_free' THEN measure_value::double
 ELSE NULL END) as avg_disk_free_1h, MIN(CASE WHEN measure_name = 'memory_free' THEN
 measure_value::double ELSE NULL END) as min_mem_free_1h, MIN(CASE WHEN measure_name =
 'cpu_idle' THEN measure_value::double ELSE NULL END) as min_cpu_idle_1h, SUM(CASE WHEN
 measure_name = 'disk_io_writes' THEN measure_value::double ELSE NULL END) as sum_1h,
 MAX(CASE WHEN measure_name = 'memory_used' THEN measure_value::double ELSE NULL END)
 as max_mem_used_1h, MAX(CASE WHEN measure_name = 'cpu_user' THEN measure_value::double
 ELSE NULL END) as max_cpu_user_1h, MAX(CASE WHEN measure_name = 'cpu_system' THEN
 measure_value::double ELSE NULL END) as max_cpu_system_1h FROM raw_data.devops WHERE
 time BETWEEN bin(@scheduled_runtime, 1h) - 14h AND bin(@scheduled_runtime, 1h) - 2h
 AND measure_name IN ('memory_free', 'memory_used', 'memory_cached', 'disk_io_writes',
 'disk_used', 'disk_free', 'cpu_user', 'cpu_system', 'cpu_idle') GROUP BY region,
 bin(time, 1h)",
    "ScheduleConfiguration" : {
```

```
        "ScheduleExpression" : "cron(0 0/1 * * ? *)"
    },
    "NotificationConfiguration" : {
        "SnsConfiguration" : {
            "TopicArn" : "******"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******",
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName" : "derived",
            "TableName" : "dashboard_metrics_1h_agg_5",
            "TimeColumn" : "hour",
            "DimensionMappings" : [
                {
                    "Name": "region",
                    "DimensionValueType" : "VARCHAR"
                }
            ],
            "MixedMeasureMappings" : [
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "avg_mem_cached_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "avg_disk_used_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "avg_disk_free_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "min_mem_free_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "min_cpu_idle_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "sum_1h",
                    "TargetMeasureName" : "total_disk_io_writes_1h"
```

```
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "max_mem_used_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "max_cpu_user_1h"
                },
                {
                    "MeasureValueType" : "DOUBLE",
                    "SourceColumn" : "max_cpu_system_1h"
                }
            ]
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    }
}
```

Di seguito è riportato lo schema per la tabella di destinazione una volta materializzata questa query pianificata. Come puoi vedere, la tabella di destinazione memorizza record con valori a misura singola di tipo double. Analogamente, lo schema delle misure della tabella mostra i nomi delle nove misure. Notate inoltre che il nome della misura total_disk_io_writes_1h è presente sin dalla mappatura rinominata sum_1h in total_disk_io_writes_1h.

Colonna	Tipo	LiveAnalytics Timestream per il tipo di attributo
Regione	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
time	timestamp	TIMESTAMP
measure_value::double	double	MEASURE_VALUE

Di seguito sono riportate le misure corrispondenti ottenute con una SHOW MEASURES query.

measure_name	data_type	Dimensioni
avg_disk_free_1h	double	[{'dimension_name': 'regione', 'tipo_dati': 'varchar'}]
avg_disk_used_1h	double	[{'dimension_name': 'regione', 'tipo_dati': 'varchar'}]
avg_mem_cached_1h	double	[{'dimension_name': 'regione', 'tipo_dati': 'varchar'}]
max_cpu_system_1h	double	[{'dimension_name': 'regione', 'tipo_dati': 'varchar'}]
max_cpu_user_1h	double	[{'dimension_name': 'regione', 'tipo_dati': 'varchar'}]
max_mem_used_1h	double	[{'dimension_name': 'regione', 'tipo_dati': 'varchar'}]
min_cpu_idle_1h	double	[{'dimension_name': 'regione', 'tipo_dati': 'varchar'}]
min_mem_free_1h	double	[{'dimension_name': 'regione', 'tipo_dati': 'varchar'}]
total-disk-io-writes	double	[{'dimension_name': 'regione', 'data_type': 'varchar'}]

Messaggi di notifica delle interrogazioni pianificate

Questa sezione descrive i messaggi inviati da Timestream per la LiveAnalytics creazione, l'eliminazione, l'esecuzione o l'aggiornamento dello stato di una query pianificata.

Nome del messaggio di notifica	Struttura	Descrizione
CreatingNotificationMessage	`CreatingNotificationMessage { String arn; NotificationType type; }`	Questo messaggio di notifica viene inviato prima dell'invio della risposta per`CreateScheduledQuery` . La query pianificata viene abilitata dopo l'invio di questa notifica. arn - La ARN query pianificata che viene creata. tipo - SCHEDULED _ _ QUERY CREATING
UpdateNotificationMessage	`UpdateNotificationMessage { String arn; NotificationType type; QueryState state; }`	Questo messaggio di notifica viene inviato quando viene aggiornata una query pianificata. Timestream for LiveAnalytics può disabilitare automaticamente la query pianificata, nel caso in cui si verifichi un errore non recuperabile, ad esempio: • AssumeRole fallimento • Eventuali errori 4xx riscontrati durante la comunicazione con KMS quando viene specificata una KMS chiave gestita dal cliente. • Eventuali errori 4xx riscontrati durante l'esecuzione della query pianificata.

Nome del messaggio di notifica	Struttura	Descrizione
		• Eventuali errori 4xx riscontrati durante l'inserimento dei risultati della query arn - La ARN query pianificata che viene aggiornata. tipo - SCHEDULED _ _ QUERY UPDATE stato - ENABLED o DISABLED
DeleteNotificationMessage	`DeletionNotificationMessage { String arn; NotificationType type; }`	Questo messaggio di notifica viene inviato quando viene eliminata una query pianificata. arn - L'oggetto ARN della query pianificata che viene creata. tipo - SCHEDULED _ _ QUERY DELETED

Nome del messaggio di notifica	Struttura	Descrizione

SuccessNotificationMessage	`SuccessNotificationMessage {`
	`  NotificationType type;`
	`  String arn;`
	`  Date nextInvocationEpochSecond;`
	`  ScheduledQueryRunSummary runSummary;`
	`}`
	
	`ScheduledQueryRunSummary {`
	`  Date invocationTime;`
	`  Date triggerTime;`
	`  String runStatus;`
	`  ExecutionStats executionstats;`
	`  ErrorReportLocation errorReportLocation;`
	`  String failureReason;`
	`}`
	
	`ExecutionStats {`
	`  Long bytesMetered;`
	`  Long dataWrites;`
	`  Long queryResultRows;`
	`  Long recordsIngested;`
	`  Long executionTimeInMillis;`
	`}`
	
	`ErrorReportLocation {`
		Questo messaggio di notifica viene inviato dopo l'esecuzione della query pianificata e l'inserimento corretto dei risultati.
		
		ARN- La ARN query pianificata che viene eliminata.
		
		NotificationType- AUTO _ TRIGGER _ SUCCESS o MANUAL _ TRIGGER _SUCCESS.
		
		nextInvocationEpochSecondo: la prossima volta che Timestream for LiveAnalytics eseguirà la query pianificata.
		
		runSummary- Informazioni sull'esecuzione pianificata della query.

Nome del messaggio di notifica	Struttura	Descrizione

	S3ReportLocation s3ReportLocation;
	}
	
	S3ReportLocation {
	String bucketName;
	String objectKey;
	}	

Nome del messaggio di notifica	Struttura	Descrizione

FailureNotificationMessage	`FailureNotificationMessage { NotificationType type; String arn; ScheduledQueryRunSummary runSummary; }`
	
	`ScheduledQueryRunSummary { Date invocationTime; Date triggerTime; String runStatus; ExecutionStats executionstats; ErrorReportLocation errorReportLocation; String failureReason; }`
	
	`ExecutionStats { Long bytesMetered; Long dataWrites; Long queryResultRows; Long recordsIngested; Long executionTimeInMillis; }`
	
	`ErrorReportLocation { S3ReportLocation s3ReportLocation;`
		Questo messaggio di notifica viene inviato quando si verifica un errore durante l'esecuzione di una query pianificata o durante l'acquisizione dei risultati della query.
		
		arn - La ARN query pianificata che viene eseguita.
		
		tipo - AUTO _ TRIGGER _ FAILURE o MANUAL _ TRIGGER _FAILURE.
		
		runSummary- Informazioni sull'esecuzione pianificata della query.

Nome del messaggio di notifica	Struttura	Descrizione

	`}`
	
	`S3ReportLocation {`
	`  String bucketName;`
	`  String objectKey;`
	`}`	

Rapporti sugli errori delle query pianificate

Questa sezione descrive la posizione, il formato e i motivi delle segnalazioni di errori generate da Timestream relative agli errori rilevati LiveAnalytics durante l'esecuzione di query pianificate.

Argomenti

- I motivi delle segnalazioni di errore delle query pianificate
- Posizione dei report di errore delle interrogazioni pianificate
- Formato dei report di errore relativi alle interrogazioni pianificate
- Tipi di errori di interrogazione pianificati
- Esempio di report sugli errori di interrogazione pianificata

I motivi delle segnalazioni di errore delle query pianificate

I report sugli errori vengono generati per errori ripristinabili. I report sugli errori non vengono generati per errori non recuperabili. Timestream for LiveAnalytics può disabilitare automaticamente le query pianificate quando si verificano errori non recuperabili. Ciò include:

- `AssumeRole`fallimento
- Eventuali errori 4xx riscontrati durante la comunicazione con KMS quando viene specificata una chiave gestita dal cliente KMS
- Eventuali errori 4xx riscontrati durante l'esecuzione di una query pianificata
- Eventuali errori 4xx riscontrati durante l'inserimento dei risultati della query

Per gli errori non recuperabili, Timestream for LiveAnalytics invia una notifica di errore con un messaggio di errore non recuperabile. Viene inoltre inviata una notifica di aggiornamento che indica che la query pianificata è disabilitata.

Posizione dei report di errore delle interrogazioni pianificate

La posizione di segnalazione degli errori di interrogazione pianificata ha la seguente convenzione di denominazione:

```
s3://customer-bucket/customer-prefix/
```

Di seguito è riportato un esempio di interrogazione ARN pianificata:

```
arn:aws:timestream:us-east-1:000000000000:scheduled-query/test-query-hd734tegrgfd
```

```
s3://customer-bucket/customer-prefix/test-query-hd734tegrgfd/<InvocationTime>/<Auto or
 Manual>/<Actual Trigger Time>
```

Auto indica le interrogazioni pianificate automaticamente pianificate da Timestream per e LiveAnalytics *Manual* indica le query pianificate attivate manualmente da un utente tramite un'`ExecuteScheduledQuery`APIazione in Amazon Timestream for Query. LiveAnalytics Per ulteriori informazioni su, consulta. `ExecuteScheduledQuery` ExecuteScheduledQuery

Formato dei report di errore relativi alle interrogazioni pianificate

Le segnalazioni di errore hanno il seguente JSON formato:

```
{
    "reportId": <String>,            // A unique string ID for all error reports
 belonging to a particular scheduled query run
    "errors": [ <Error>, ... ],      // One or more errors
}
```

Tipi di errori di interrogazione pianificati

L'`Error`oggetto può essere di tre tipi:

- Registra gli errori di inserimento

```
{
```

```
    "reason": <String>,              // The error message String
    "records": [ <Record>, ... ],    // One or more rejected records )
}
```

- Errori di analisi e convalida delle righe

```
{
    "reason": <String>,        // The error message String
    "rawLine": <String>,       // [Optional] The raw line String that is being parsed
 into record(s) to be ingested. This line has encountered the above-mentioned parse
 error.
}
```

- Errori generali

```
{
    "reason": <String>,        // The error message
}
```

Esempio di report sugli errori di interrogazione pianificata

Di seguito è riportato un esempio di segnalazione di errori prodotta a causa di errori di inserimento.

```
{
    "reportId": "C9494AABE012D1FBC162A67EA2C18255",
    "errors": [
        {
            "reason": "The record timestamp is outside the time range
 [2021-11-12T14:18:13.354Z, 2021-11-12T16:58:13.354Z) of the memory store.",
            "records": [
                {
                    "dimensions": [
                        {
                            "name": "dim0",
                            "value": "d0_1",
                            "dimensionValueType": null
                        },
                        {
                            "name": "dim1",
                            "value": "d1_1",
                            "dimensionValueType": null
                        }
```

```
                ],
                "measureName": "random_measure_value",
                "measureValue": "3.141592653589793",
                "measureValues": null,
                "measureValueType": "DOUBLE",
                "time": "1637166175635000000",
                "timeUnit": "NANOSECONDS",
                "version": null
            },
            {
                "dimensions": [
                    {
                        "name": "dim0",
                        "value": "d0_2",
                        "dimensionValueType": null
                    },
                    {
                        "name": "dim1",
                        "value": "d1_2",
                        "dimensionValueType": null
                    }
                ],
                "measureName": "random_measure_value",
                "measureValue": "6.283185307179586",
                "measureValues": null,
                "measureValueType": "DOUBLE",
                "time": "1637166175636000000",
                "timeUnit": "NANOSECONDS",
                "version": null
            },
            {
                "dimensions": [
                    {
                        "name": "dim0",
                        "value": "d0_3",
                        "dimensionValueType": null
                    },
                    {
                        "name": "dim1",
                        "value": "d1_3",
                        "dimensionValueType": null
                    }
                ],
                "measureName": "random_measure_value",
```

```
                    "measureValue": "9.42477796076938",
                    "measureValues": null,
                    "measureValueType": "DOUBLE",
                    "time": "1637166175637000000",
                    "timeUnit": "NANOSECONDS",
                    "version": null
                },
                {
                    "dimensions": [
                        {
                            "name": "dim0",
                            "value": "d0_4",
                            "dimensionValueType": null
                        },
                        {
                            "name": "dim1",
                            "value": "d1_4",
                            "dimensionValueType": null
                        }
                    ],
                    "measureName": "random_measure_value",
                    "measureValue": "12.566370614359172",
                    "measureValues": null,
                    "measureValueType": "DOUBLE",
                    "time": "1637166175638000000",
                    "timeUnit": "NANOSECONDS",
                    "version": null
                }
            ]
        }
    ]
}
```

Modelli ed esempi di interrogazioni pianificate

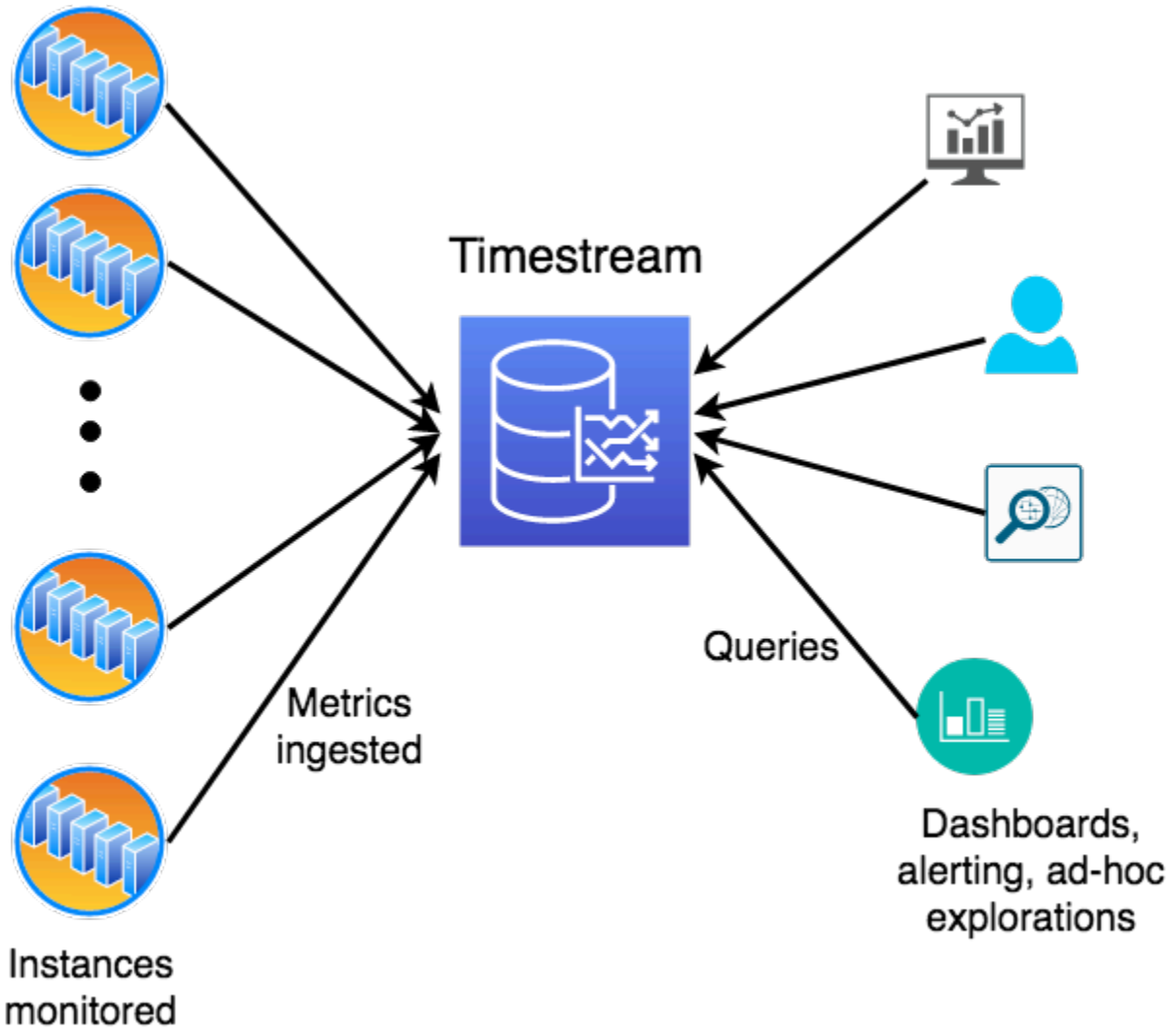
Questa sezione descrive i modelli di utilizzo per le query pianificate e end-to-end alcuni esempi.

Argomenti

- Schema di esempio per le interrogazioni pianificate
- Schemi di interrogazione pianificati
- Esempi di interrogazioni pianificate

Schema di esempio per le interrogazioni pianificate

In questo esempio utilizzeremo un'applicazione di esempio che imita uno DevOps scenario di monitoraggio delle metriche di un'ampia flotta di server. Gli utenti desiderano avvisare sull'utilizzo anomalo delle risorse, creare dashboard sul comportamento e sull'utilizzo aggregati della flotta ed eseguire analisi sofisticate su dati recenti e storici per trovare correlazioni. Il diagramma seguente fornisce un'illustrazione della configurazione in cui un insieme di istanze monitorate emette metriche a Timestream for. LiveAnalytics Un altro gruppo di utenti simultanei invia query per avvisi, dashboard o analisi ad hoc, in cui le query e l'inserimento vengono eseguite in parallelo.



L'applicazione monitorata è modellata come un servizio altamente scalabile distribuito in diverse regioni del mondo. Ogni regione è ulteriormente suddivisa in una serie di unità di scala chiamate celle che presentano un livello di isolamento in termini di infrastruttura all'interno della regione. Ogni cella è ulteriormente suddivisa in silos, che rappresentano un livello di isolamento del software. Ogni silo

dispone di cinque microservizi che comprendono un'istanza isolata del servizio. Ogni microservizio dispone di diversi server con diversi tipi di istanze e versioni del sistema operativo, che vengono distribuiti in tre zone di disponibilità. Questi attributi che identificano i server che emettono le metriche sono modellati come dimensioni in Timestream for. LiveAnalytics In questa architettura, abbiamo una gerarchia di dimensioni (come region, cell, silo e microservice_name) e altre dimensioni che attraversano la gerarchia (come instance_type e availability_zone).

L'applicazione emette una varietà di metriche (come cpu_user e memory_free) ed eventi (come task_completed e gc_reclaimed). Ogni metrica o evento è associato a otto dimensioni (come regione o cella) che identificano in modo univoco il server che lo emette. I dati vengono scritti con le 20 metriche archiviate insieme in un record multimisura con metriche relative ai nomi delle misure e tutti e 5 gli eventi vengono archiviati insieme in un altro record multimisura con eventi relativi ai nomi delle misure. Il modello di dati, lo schema e la generazione dei dati sono disponibili nel generatore di dati open source. Oltre allo schema e alle distribuzioni dei dati, il generatore di dati fornisce un esempio di utilizzo di più scrittori per inserire dati in parallelo, utilizzando la scalabilità di ingestione di Timestream per inserire milioni di misurazioni LiveAnalytics al secondo. Di seguito mostriamo lo schema (tabella e schema di misura) e alcuni dati di esempio dal set di dati.

Argomenti

- Record multimisura
- Record a misura singola

Record multimisura

Schema della tabella

Di seguito è riportato lo schema della tabella una volta che i dati vengono inseriti utilizzando record multimisura. È l'output dell'interrogazione. DESCRIBE Supponendo che i dati vengano inseriti in un database raw_data e nella tabella devops, di seguito è riportata la query.

```
DESCRIBE "raw_data"."devops"
```

Colonna	Tipo	LiveAnalytics Timestream per il tipo di attributo
availability_zone	varchar	DIMENSION

Colonna	Tipo	LiveAnalytics Timestream per il tipo di attributo
microservice_name	varchar	DIMENSION
nome_istanza	varchar	DIMENSION
nome_processo	varchar	DIMENSION
os_version	varchar	DIMENSION
versione jdk	varchar	DIMENSION
cellula	varchar	DIMENSION
Regione	varchar	DIMENSION
silo	varchar	DIMENSION
instance_type	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
time	timestamp	TIMESTAMP
senza memoria	double	MULTI
cpu_steal	double	MULTI
cpu_iowait	double	MULTI
cpu_user	double	MULTI
memoria_memorizzata nella cache	double	MULTI
disk_io_reads	double	MULTI
cpu_hi	double	MULTI
latenza_per_lettura	double	MULTI

Colonna	Tipo	LiveAnalytics Timestream per il tipo di attributo
network_bytes_out	double	MULTI
cpu_idle	double	MULTI
disk_free	double	MULTI
memoria_usata	double	MULTI
sistema_cpu	double	MULTI
descrizioni_di_file in uso	double	MULTI
disk_used	double	MULTI
cpu_nice	double	MULTI
disk_io_write	double	MULTI
cpu_si	double	MULTI
latenza_per_scrittura	double	MULTI
bytes_in di rete	double	MULTI
stato_finale dell'attività	varchar	MULTI
gc_pause	double	MULTI
task_completato	bigint	MULTI
gc_reclaimed	double	MULTI

Schema di misura

Di seguito è riportato lo schema di misura restituito dalla SHOW MEASURES query.

```
SHOW MEASURES FROM "raw_data"."devops"
```

measure_name	data_type	Dimensioni
events	multi	[{"data_type» :"varchar», "dimension_name» :"availab ility_zone "}, {" data_type » :"varchar», "dimensio n_name» :"microservice_nam e "}, {" data_type» :"varchar», "dimension_name» :"nome_is tanza "}, {" tipo_dati» :"varchar » :"varchar», "dimensio n_name» :"process_name "}, {" data_type» :"varchar», "dimension_name» :"jdk_ver sion "}, {" data_type» :"varchar », "dimension_name» :"cell "}, {" data_type» :"varchar», "dimension_name» :"region "}, {" tipo_dati» :"varchar», "dimension_name» :"region "}, {" tipo_dati» :"varchar», "dimension_name» :"region "}, {" tipo_dati» :"varchar »» :"varchar», "nome_dim ensione» :"silo "}]
metriche	multi	[{"data_type» :"varchar», "dimension_name» :"availab ility_zone "}, {" data_type » :"varchar», "dimensio n_name» :"microservice_nam e "}, {" data_type» :"varchar», "dimension_name» :"nome_is tanza "}, {" tipo_dati» :"varchar » :"varchar», "dimensio n_name» :"os_version "},

measure_name	data_type	Dimensioni
		{" data_type» :"varchar», "dimension_name» :"cell "}, {" data_type» :"varchar», "dimension_name» :"region "}, {" data_type» :"varchar », "dimension_name» :"silo "}, {" tipo_dati» :"varchar » :"dimension_name» :"silo "}, {" tipo_dati» :"varchar » :"dimension_name» :"silo "}, {" tipo_dati» :"varchar » :"dimension_name» :"silo "}, {" tipo_dati» :"varchar » :"dimension_name» :"silo "} «varchar», "nome_dim ensione» :"tipo_istanza "}]

Dati di esempio

R	C	S	a	n	n	i	o	n	v	m	C	c	si	c	c	c	c	c	c	c	m	m	d	la	d	la	d	d	b	n	d	s	st	g	ta	g	s	s
			ity	iz	nn	ty	on	e	jo	a			p			t					sa	ea nc	e	ea	ri	eu			re	yt	in u	m	a d vi		le	me	ed
u s e	u s e - c	u s e - c si	S U e s	a i	i - - 2 c - si 0 m c	m e	A			m	1 1 1	6	0	3	0	0	0	0	0	5	8	5	9	3	2	6	2	8	4	3	5							

R	C	S	a	n	n	i	o	n	v	m	C	c	si	c	c	c	c	c	c	c	m	m	d	la	d	la	d	d	b	n	d	s	st	g	ta	g	s	s
			ity	iz	n	ty	n	e	jo	a			p			t						sa	e	e	a	ri	e	u			re	yt	e		a		le	me	ed	
					za																		a										in		vi						
					au																		nc										u		d						
					-																		a																	
					e																																			
u	u	u	S	a	i	m	A			m	1	5	0	3	0	0	0	0	0	0	9	3	5	3	2	9	5	3	7	5	6	2						
e	e	e	U	i	-	e					1																													
	-	-	e		-						1																													
	c	c	s		2																																			
		si			c																																			
					-																																			
					si																																			
					0																																			
					m																																			
					c																																			
					za																																			
					au																																			
					-																																			
					e																																			

R	C	S	a	n	n	i	o	n	v	m	C	c	s	c	c	c	c	c	c	c	m	m	d	l	d	l	d	d	b	n	d	s	s	g	t	g	s	e	l	a
			ity	iz	n	ty	n	e	jc	a			p			t						sa	ea	e	a	ri	u			re	yt	in		vi			le	me	ed		
u	u	u	S	a	i-	m	A			m	1	4	0	4	0	0	0	0	0	9	4	5	3	8	3	7	6	8	5	4	8									
	e-	e-	U		-2	e					1																													
	c	c	e		c						1																													
		si			-																																		
					si																																			
					0																																			
					m																																			
					c																																			
					z																																			
					a																																			
					u																																			
					-																																			
					e																																			
u	u	u	S	a	i-	m	A			m	1	3	0	5	0	0	0	0	0	4	7	2	4	4	3	8	6	2	1	2	1								
	e-	e-	U		-2	e					1																													
	c	c	e		c						1																													
		si			-																																			
					si																																			
					0																																			
					m																																			
					c																																			
					z																																			
					a																																			
					u																																			
					-																																			
					e																																			

R	C	S	a	n	n	i	o	n	v	m	C	c	s	c	c	c	c	c	c	c	m	m	d	l	d	l	d	d	b	n	d	s	s	g	t	g	s	c	l	a
			ity	iz	n	ty	n	e	jc	a			p			t					sa	ea	e	a	ri	u			re	yt	in	vi			le	me	ed				
u	u	u	S	a	i-	m	A			m	1	5	0	3	0	0	0	0	0	0	5	3	8	5	7	1	8	6	7	6	5	3								
	e	u	U		-	e					1																														
	-	e	e		2						1																														
	c	-	s		c						1																														
		c	i		-																																				
		o			s																																				
		n			i																																				
		-			0																																				
		s			m																																				
		i			c																																				
		m			z																																				
		u			a																																				
		l			t																																				
		a			u																																				
		t			-																																				
		e			e																																				
u	u	u	S	a	i-			h	J	e	1																					7	S	8	3	6	3	4	8
	e	e	U		-			g			1																						_							
	-	-	e		2						1																						W							
	c	c			-																												R							
		o	s		c																																				
		n	i		z																																				
		-			a																																				
		s			t																																				
		i			i																																				
		m			o																																				
		u			n																																				
		l			-																																				
		a			e																																				
		t																																							
		e																																							

R	C	S	a	n	i	o	n	v	m	C	c	si	c	c	c	c	c	c	c	m	m	d	la	d	la	d	d	b	n	d	s	st	g	ta	g	s
u	u	u	S	a	i-		h	J	e	1																					7	S	2	4	7	3
	-	-	U		-2		g			1																						I					
	c	c	e		c					1																						T					
		si			-																																
					si																																
					0																																
					m																																
					c																																
					z																																
					a																																
					u																																
					-																																
					e																																
u	u	u	S	a	i-		h	J	e	1																					6	S	6	2	7	2
	-	-	U		-2		g			1																						I					
	c	c	e		c					1																						T					
		si			-																																
					si																																
					0																																
					m																																
					c																																
					z																																
					a																																
					u																																
					-																																
					e																																

R	C	S	a	n	in	o	n	v	m	C	c	si	c	c	c	c	c	c	c	m	m	d	la	d	la	d	d	b	n	d	s	st	g	ta	g	s
			ity	iz	ty	on	e	jo	a			p			t					sa	ea	e	a	ri	u			re	yt	in		vi		le	me	ed
u	u	u	S	a	i-		h	J	e	1																					2	S	5	1	9	9_W
e	e-	e-	U	i	-2		g	L		1																						IT				JL
	c	c	e		c					1																										
		si			-																															
					si																															
					0																															
					m																															
					c																															
					z																															
					a																															
					u																															
					-																															
					e																															
u	u	u	S	a	i-		h	J	e	1																					3	S	7	2	8	2_9
e	e-	e-	U	i	-2		g	L		1																						_I				
	c	c	e		c					1																						W				
		si			-																											R				
					si																															
					0																															
					m																															
					c																															
					z																															
					a																															
					u																															
					-																															
					e																															

Record a misura singola

Timestream for consente LiveAnalytics inoltre di inserire i dati con una misura per record di serie temporali. Di seguito sono riportati i dettagli dello schema quando vengono inseriti utilizzando record a misura singola.

Schema della tabella

Di seguito è riportato lo schema della tabella una volta che i dati vengono inseriti utilizzando record multimisura. È l'output dell'interrogazione. DESCRIBE Supponendo che i dati vengano inseriti in un database raw_data e nella tabella devops, di seguito è riportata la query.

```
DESCRIBE "raw_data"."devops_single"
```

Colonna	Tipo	LiveAnalytics Timestream per il tipo di attributo
availability_zone	varchar	DIMENSION
microservice_name	varchar	DIMENSION
nome_istanza	varchar	DIMENSION
nome_processo	varchar	DIMENSION
os_version	varchar	DIMENSION
versione jdk	varchar	DIMENSION
cellula	varchar	DIMENSION
Regione	varchar	DIMENSION
silo	varchar	DIMENSION
instance_type	varchar	DIMENSION
measure_name	varchar	MEASURE_NAME
time	timestamp	TIMESTAMP

Colonna	Tipo	LiveAnalytics Timestream per il tipo di attributo
measure_value::double	double	MEASURE_VALUE
measure_value::bigint	bigint	MEASURE_VALUE
measure_value::varchar	varchar	MEASURE_VALUE

Schema di misura

Di seguito è riportato lo schema di misura restituito dalla SHOW MEASURES query.

```
SHOW MEASURES FROM "raw_data"."devops_single"
```

measure_name	data_type	Dimensioni
cpu_hi	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_nam e': 'varchar'}: 'dimensio

measure_name	data_type	Dimensioni
		n_name': 'varchar': 'instance _type', 'data_type': 'varchar'}]
cpu_idle	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_nam e': 'varchar'}: 'dimensio n_name': 'varchar': 'instance _type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
cpu_iowait	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_nam e': 'varchar'}: 'dimensio n_name': 'varchar': 'instance _type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
cpu_nice	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_name': 'varchar'}: 'dimension_name': 'varchar': 'instance_type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
cpu_si	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_nam e': 'varchar'}: 'dimensio n_name': 'varchar': 'instance _type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
cpu_steal	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_name': 'varchar'}: 'dimension_name': 'varchar': 'instance_type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
sistema_CPU	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_nam e': 'varchar'}: 'dimensio n_name': 'varchar': 'instance _type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
cpu_user	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_nam e': 'varchar'}: 'dimensio n_name': 'varchar': 'instance _type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
disk_free	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_nam e': 'varchar'}: 'dimensio n_name': 'varchar': 'instance _type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
disk_io_reads	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_nam e': 'varchar'}: 'dimensio n_name': 'varchar': 'instance _type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
disk_io_write	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_nam e': 'varchar'}: 'dimensio n_name': 'varchar': 'instance _type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
disk_used	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_nam e': 'varchar'}: 'dimensio n_name': 'varchar': 'instance _type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
descrittori di file in uso	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_name': 'varchar'}: 'dimension_name': 'varchar': 'instance_type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
gc_pause	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nome_istanza', 'data_type': 'varchar'}, {'dimension_name': 'process_ name': 'process_name' ',' data_type ':' varchar '}, {' dimension_name ':' jdk_versi on ',' data_type ':' varchar '}, {' dimension_name ':' cell ',' data_type ':' varchar '}, {' dimension_name ':' region ',' data_type ':' varchar '}, {' dimension_name ':' varchar '}, {' dimension_name ':' varchar '}, {' dimension_name ':' varchar '}, {' dimension_name ':' varchar '}': 'silo', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
gc_reclaimed	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'nome_istanza', 'data_type': 'varchar'}, {'dimension_name': 'process_name': 'process_name' ',' data_type ':' varchar '}, {' dimension_name ':' jdk_version ',' data_type ':' varchar '}, {' dimension_name ':' cell ',' data_type ':' varchar '}, {' dimension_name ':' region ',' data_type ':' varchar '}, {' dimension_name ':' varchar '}, {' dimension_name ':' varchar '}, {' dimension_name ':' varchar '}, {' dimension_name ':' varchar '}': 'silo', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
latenza_per_lettura	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_name': 'varchar'}: 'dimension_name': 'varchar': 'instance_type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
latenza_per_scrittura	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_name': 'varchar'}: 'dimension_name': 'varchar': 'instance_type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
memoria_memorizzata nella cache	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_name': 'varchar'}: 'dimension_name': 'varchar': 'instance_type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
senza memoria	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'nome_istanza', 'data_type': 'varchar'}, {'dimension_name': 'process_ name': 'process_name' ',' data_type ':' varchar '}, {' dimension_name ':' os_versio n ',' data_type ':' varchar '}, {' dimension_name ':' jdk_versi on ',' data_type ':' varchar '}, {' dimension_name ':' cell ',' data_type ':' varchar '}, {' dimension_name ':' cell ',' data_type ':' varchar '}, {' dimension_name ':' cell ',' data_type ':' varchar '}, {' dimension_name ':' cella ',' data_type ':' varchar '}, {' dimension_name ':' cella ',' data_type ':' varchar '}, {' dimension_name ':' region ',' data_type ':' varchar '}, {' dimension_name ':' silo ',' data_type ':' varchar '}, {' dimension_name ':' instance_ type ', 'data_ tipo': 'varchar'}]

measure_name	data_type	Dimensioni
memoria_usata	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_name': 'varchar'}: 'dimension_name': 'varchar': 'instance_type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
bytes_in di rete	double	[{'dimension_name': 'availabi lity_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_typ e': 'varchar'}, {'dimensi on_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_versi on', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_nam e': 'varchar'}: 'dimensio n_name': 'varchar': 'instance _type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
network_bytes_out	double	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'instance_name', 'data_type': 'varchar'}, {'dimension_name': 'os_version', 'data_type': 'varchar'}, {'dimension_name': 'cell', 'data_type': 'varchar'}, {'dimension_name': 'region', 'data_type': 'varchar'}, {'dimension_name': 'silo', 'data_type': 'varchar'}, {'dimension_name': 'varchar'}, {'dimension_name': 'data_name': 'varchar'}: 'dimension_name': 'varchar': 'instance_type', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
attività_completata	bigint	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'nome_istanza', 'data_type': 'varchar'}, {'dimension_name': 'process_name': 'process_name' ',' data_type ':' varchar '}, {' dimension_name ':' jdk_version ',' data_type ':' varchar '}, {' dimension_name ':' cell ',' data_type ':' varchar '}, {' dimension_name ':' region ',' data_type ':' varchar '}, {' dimension_name ':' varchar '}, {' dimension_name ':' varchar '}, {' dimension_name ':' varchar '}, {' dimension_name ':' varchar '}': 'silo', 'data_type': 'varchar'}]

measure_name	data_type	Dimensioni
stato_finale dell'attività	varchar	[{'dimension_name': 'availability_zone', 'data_type': 'varchar'}, {'dimension_name': 'microservice_name', 'data_type': 'varchar'}, {'dimension_name': 'nome_istanza', 'data_type': 'varchar'}, {'dimension_name': 'process_name': 'process_name' ',' data_type ':' varchar '}, {' dimension_name ':' jdk_version ',' data_type ':' varchar '}, {' dimension_name ':' cell ',' data_type ':' varchar '}, {' dimension_name ':' region ',' data_type ':' varchar '}, {' dimension_name ':' varchar '}, {' dimension_name ':' varchar '}, {' dimension_name ':' varchar '}, {' dimension_name ':' varchar '}': 'silo', 'data_type': 'varchar'}]

Dati di esempio

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	ercole	i- -1-cell-9 silo-2 0000		AL20		eu-west--cell-9	eu-west-	eu-west--celle-silo-2	r5.4xl e	cpu_i	34:57	0,871		

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
		azona om zaZsv hercu eu-west												
eu-west--1	ercole	i- -1-cell-9 silo-20000 azona om zaZsv hercu eu-west		AL20		eu-west--cell-9	eu-west-	eu-west--celle-silo-2	r5.4xl e	cpu_i	34:57	3,462		
eu-west--1	ercole	i- -1-cell-9 silo-20000 azona om zaZsv hercu eu-west		AL20		eu-west--cell-9	eu-west-	eu-west--celle-silo-2	r5.4xl e	cpu_i t	34:57	0,102		

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu-west		AL20		eu-west- - cell-9	eu-west-	eu-west- - celle- silo-2	r5.4xl e	cpu_r	34:57	0,630		
eu-west--1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu-west		AL20		eu-west- - cell-9	eu-west-	eu-west- - celle- silo-2	r5.4xl e	cpu_i	34:57	0,164		

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu-west		AL20		eu-west- - cell-9	eu-west-	eu-west- - celle- silo-2	r5.4xl e	cpu_s	34:57	0,107		
eu-west--1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu-west		AL20		eu-west- - cell-9	eu-west-	eu-west- - celle- silo-2	r5.4xl e	sister pu	34:57	0,457		

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu- west- -1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu- west		AL20		eu- west- - cell-9	eu- west-	eu- west- - celle- silo-2	r5.4xl e	cpu_u	34:57	94,20		
eu- west- -1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu- west		AL20		eu- west- - cell-9	eu- west-	eu- west- - celle- silo-2	r5.4xl e	disk_f	34:57	72,51		

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu- west- -1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu- west		AL20		eu- west- - cell-9	eu- west-	eu- west- - celle- silo-2	r5.4xl e	disk_i eads	34:57	81,73		
eu- west- -1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu- west		AL20		eu- west- - cell-9	eu- west-	eu- west- - celle- silo-2	r5.4xl e	disk_i rite	34:57	77,11		

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu-west		AL20		eu-west- - cell-9	eu-west-	eu-west- - celle- silo-2	r5.4xl e	disk_	34:57	89,42		
eu-west--1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu-west		AL20		eu-west- - cell-9	eu-west-	eu-west- - celle- silo-2	r5.4xl e	descr ni_di_ e in uso	34:57	30,08		

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu-west	serve		JDK_	eu-west- - cell-9	eu-west-	eu-west- - celle- silo-2		gc_pa	34:57	60,28		
eu-west--1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu-west	serve		JDK_	eu-west- - cell-9	eu-west-	eu-west- - celle- silo-2		gc_re med	34:57	75,28		

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu-west		AL20		eu-west- - cell-9	eu-west-	eu-west- - celle-silo-2	r5.4xl e	laten er_let a	34:57	8,076		
eu-west--1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu-west		AL20		eu-west- - cell-9	eu-west-	eu-west- - celle-silo-2	r5.4xl e	laten er_sc ura	34:57	58,11		

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu-west--1	ercole	i- -1-cell-9 silo-2 0000 azona om zaZsv hercu eu-west		AL20		eu-west--cell-9	eu-west-	eu-west--celle-silo-2	r5.4xl e	memo emori a nella cache	34:57	87,56		
eu-west--1	ercole	i- -1-cell-9 silo-2 0000 azona om zaZsv hercu eu-west	serve		JDK_	eu-west--cell-9	eu-west-	eu-west--celle-silo-2		senza memo	34:57	18,95		

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu- west- -1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu- west		AL20		eu- west- - cell-9	eu- west-	eu- west- - celle- silo-2	r5.4xl e	senza memo	34:57	9720		
eu- west- -1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu- west		AL20		eu- west- - cell-9	eu- west-	eu- west- - celle- silo-2	r5.4xl e	memo sata	34:57	12,37		

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu- west- -1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu- west		AL20		eu- west- - cell-9	eu- west-	eu- west- - cell-9 s ilo-2	r5.4xl e	netwo ytes_	34:57	31,02		
eu- west- -1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu- west		AL20		eu- west- - cell-9	eu- west-	eu- west- - cell-9 s ilo-2	r5.4xl e	netwo ytes_	34:57	0,514		

availa ity_zo	micro ice_n	nome nza	nome esso	os_ve n	versio jdk	Cella	Regio	Silo	instar type	meas ame	Orario	meas alue:: ble	meas alue:: int	measure_v alue::var char
eu- west- -1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu- west	serve		JDK_	eu- west- - cell-9	eu- west-	eu- west- - cell-9 s ilo-2		task_ letato	34:57		69	
eu- west- -1	ercole	i- - 1- cell-9 silo-2 0000 azona om zaZsv hercu eu- west	serve		JDK_	eu- west- - cell-9	eu- west-	eu- west- - cell-9 s ilo-2		stato_ ale dell'at vità	34:57			SUCCESS_ ITH_RESUL T

Schemi di interrogazione pianificati

In questa sezione troverai alcuni modelli comuni su come utilizzare Amazon Timestream LiveAnalytics for Scheduled Queries per ottimizzare i dashboard in modo che vengano caricati più velocemente e a costi ridotti. Gli esempi seguenti utilizzano uno scenario DevOps applicativo per illustrare i concetti chiave che si applicano alle query pianificate in generale, indipendentemente dallo scenario dell'applicazione.

Le interrogazioni pianificate in Timestream for LiveAnalytics consentono di esprimere le query utilizzando l'intera superficie di Timestream for. SQL LiveAnalytics La tua query può includere una o più tabelle di origine, eseguire aggregazioni o qualsiasi altra query consentita dal SQL linguaggio di Timestream for e quindi materializzare i risultati LiveAnalytics della query in un'altra tabella di destinazione in Timestream for. LiveAnalytics Per facilitare l'esposizione, questa sezione fa riferimento a questa tabella di destinazione di una query pianificata come tabella derivata.

Di seguito sono riportati i punti chiave trattati in questa sezione.

- Utilizzo di un semplice aggregato a livello di flotta per spiegare come definire un'interrogazione pianificata e comprendere alcuni concetti di base.
- Come combinare i risultati della destinazione di una query pianificata (la tabella derivata) con i risultati della tabella di origine per ottenere i vantaggi in termini di costi e prestazioni delle query pianificate.
- Quali sono i compromessi quando si configura il periodo di aggiornamento delle query pianificate.
- Utilizzo di interrogazioni pianificate per alcuni scenari comuni.
  - Monitoraggio dell'ultimo punto dati di ogni istanza prima di una data specifica.
  - Valori distinti per una dimensione da utilizzare per popolare le variabili in una dashboard.
- Come gestire i dati in arrivo in ritardo nel contesto delle interrogazioni pianificate.
- In che modo è possibile utilizzare esecuzioni manuali una tantum per gestire una varietà di scenari non direttamente coperti dai trigger automatici per le query pianificate.

Argomenti

Scenario

Gli esempi seguenti utilizzano uno scenario di DevOps monitoraggio descritto in. Schema di esempio per le interrogazioni pianificate

Gli esempi forniscono la definizione delle interrogazioni pianificate in cui è possibile inserire le configurazioni appropriate per ricevere le notifiche sullo stato di esecuzione per le query pianificate, dove ricevere i report sugli errori riscontrati durante l'esecuzione di una query pianificata e il IAM ruolo utilizzato dalla query pianificata per eseguire le sue operazioni.

È possibile creare queste interrogazioni pianificate dopo aver compilato le opzioni precedenti, creato la tabella di destinazione (o derivata) ed eseguito tramite. AWS CLI Ad esempio, si supponga che la definizione di una query pianificata sia archiviata in un file,. `scheduled_query_example.json` È possibile creare l'interrogazione utilizzando il CLI comando.

```
aws timestream-query create-scheduled-query --cli-input-json  file://
scheduled_query_example.json --profile aws_profile --region us-east-1
```

Nel comando precedente, il profilo passato utilizzando l'opzione --profile deve disporre delle autorizzazioni appropriate per creare interrogazioni pianificate. Vedi Politiche basate sull'identità per le interrogazioni pianificate per istruzioni dettagliate sulle politiche e le autorizzazioni.

Aggregati semplici a livello di parco

Questo primo esempio illustra alcuni dei concetti di base relativi all'utilizzo di query pianificate utilizzando un semplice esempio di calcolo degli aggregati a livello di flotta. Utilizzando questo esempio, imparerai quanto segue.

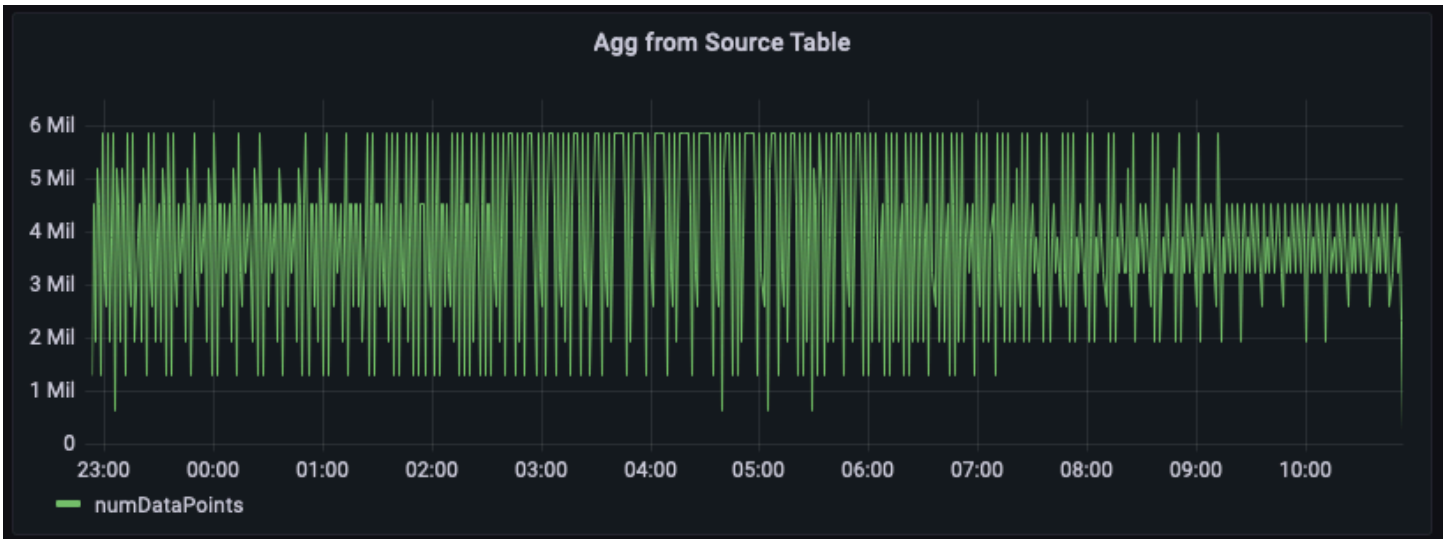
- Come prendere la query del dashboard utilizzata per ottenere statistiche aggregate e mapparla a una query pianificata.
- In che modo Timestream for LiveAnalytics gestisce l'esecuzione delle diverse istanze della query pianificata.
- Come è possibile sovrapporre diverse istanze di query pianificate negli intervalli di tempo e in che modo la correttezza dei dati viene mantenuta nella tabella di destinazione per garantire che la dashboard che utilizza i risultati della query pianificata fornisca risultati corrispondenti allo stesso aggregato calcolato sui dati grezzi.
- Come impostare l'intervallo di tempo e la cadenza di aggiornamento per la query pianificata.
- In che modo è possibile monitorare autonomamente i risultati delle query pianificate per ottimizzarle in modo che la latenza di esecuzione delle istanze di query rientri nei ritardi accettabili associati all'aggiornamento delle dashboard.

Argomenti

- Aggrega dalle tabelle di origine
- Interrogazione pianificata per precalcolare gli aggregati
- Aggregazione da tabella derivata
- Aggregazione che combina tabelle di origine e derivate
- Aggregazione basata su calcoli pianificati aggiornati di frequente

Aggrega dalle tabelle di origine

In questo esempio, stai monitorando il numero di metriche emesse dai server all'interno di una determinata regione in ogni minuto. Il grafico seguente è un esempio di rappresentazione di questa serie temporale per la regione us-east-1.



Di seguito è riportato un esempio di query per calcolare questo aggregato dai dati grezzi. Filtra le righe per la regione us-east-1 e quindi calcola la somma al minuto tenendo conto delle 20 metriche (se measure_name è metrica) o dei 5 eventi (se measure_name è eventi). In questo esempio, l'illustrazione del grafico mostra che il numero di metriche emesse varia tra 1,5 milioni e 6 milioni al minuto. Quando si traccia questa serie temporale per diverse ore (le ultime 12 ore in questa figura), questa interrogazione sui dati grezzi analizza centinaia di milioni di righe.

```
WITH grouped_data AS (
    SELECT region, bin(time, 1m) as minute, SUM(CASE WHEN measure_name = 'metrics' THEN
 20 ELSE 5 END) as numDataPoints
    FROM "raw_data"."devops"
    WHERE time BETWEEN from_milliseconds(1636699996445) AND
 from_milliseconds(1636743196445)
```

```
        AND region = 'us-east-1'
    GROUP BY region, measure_name, bin(time, 1m)
)
SELECT minute, SUM(numDataPoints) AS numDataPoints
FROM grouped_data
GROUP BY minute
ORDER BY 1 desc, 2 desc
```

Interrogazione pianificata per precalcolare gli aggregati

Se desideri ottimizzare i dashboard per velocizzare il caricamento e ridurre i costi scansionando meno dati, puoi utilizzare una query pianificata per precalcolare questi aggregati. Le query pianificate in Timestream for ti LiveAnalytics consentono di materializzare questi precalcoli in un altro Timestream for table, che puoi utilizzare successivamente per i tuoi dashboard. LiveAnalytics

Il primo passaggio per creare una query pianificata consiste nell'identificare la query che si desidera precalcolare. Nota che la dashboard precedente è stata disegnata per la regione us-east-1. Tuttavia, un utente diverso potrebbe desiderare lo stesso aggregato per una regione diversa, ad esempio us-west-2 o eu-west-1. Per evitare di creare un'interrogazione pianificata per ciascuna di queste query, puoi precalcolare l'aggregato per ogni regione e materializzare gli aggregati per regione in un altro Timestream per tabella. LiveAnalytics

La query riportata di seguito fornisce un esempio del precalcolo corrispondente. Come puoi vedere, è simile all'espressione di tabella comune grouped_data utilizzata nella query sui dati grezzi, ad eccezione di due differenze: 1) non utilizza un predicato di regione, quindi possiamo usare una query per il precalcolo per tutte le regioni; e 2) utilizza un predicato temporale parametrizzato con un parametro speciale @scheduled_runtime, che viene spiegato in dettaglio di seguito.

```
SELECT region, bin(time, 1m) as minute,
    SUM(CASE WHEN measure_name = 'metrics' THEN 20 ELSE 5 END) as numDataPoints
FROM raw_data.devops
WHERE time BETWEEN @scheduled_runtime - 10m AND @scheduled_runtime + 1m
GROUP BY bin(time, 1m), region
```

L'interrogazione precedente può essere convertita in un'interrogazione pianificata utilizzando le seguenti specifiche. All'interrogazione pianificata viene assegnato un nome, che è un mnemonico di facile utilizzo. Include quindi, a QueryString ScheduleConfiguration, che è un'espressione cron. Specifica TargetConfiguration quale mappa i risultati della query alla tabella di destinazione in Timestream for. LiveAnalytics Infine, specifica una serie di altre configurazioni, come la

NotificationConfiguration, in cui vengono inviate le notifiche per le singole esecuzioni della query, in cui viene scritto un rapporto nel caso in ErrorReportConfiguration cui la query riscontri errori e la ScheduledQueryExecutionRoleArn, che è il ruolo utilizzato per eseguire le operazioni per la query pianificata.

```
{
    "Name": "MultiPT5mPerMinutePerRegionMeasureCount",
    "QueryString": "SELECT region, bin(time, 1m) as minute, SUM(CASE WHEN measure_name
 = 'metrics' THEN 20 ELSE 5 END) as numDataPoints FROM raw_data.devops WHERE time
 BETWEEN @scheduled_runtime - 10m AND @scheduled_runtime + 1m GROUP BY bin(time, 1m),
 region",
    "ScheduleConfiguration": {
        "ScheduleExpression": "cron(0/5 * * * ? *)"
    },
    "NotificationConfiguration": {
        "SnsConfiguration": {
            "TopicArn": "******"
        }
    },
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName": "derived",
            "TableName": "per_minute_aggs_pt5m",
            "TimeColumn": "minute",
            "DimensionMappings": [
                {
                    "Name": "region",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "numDataPoints",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "numDataPoints",
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
```

```
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

Nell'esempio, il ScheduleExpression cron (0/5 * * *? *) implica che la query venga eseguita una volta ogni 5 minuti al 5, 10, 15... minuti di ogni ora di ogni giorno. Questi timestamp quando viene attivata un'istanza specifica di questa query sono ciò che si traduce nel parametro @scheduled_runtime utilizzato nella query. Ad esempio, si consideri l'istanza di questa query pianificata in esecuzione il 01/12/2021 00:00:00. In questo caso, il parametro @scheduled_runtime viene inizializzato con il timestamp 2021-12-01 00:00:00 quando si richiama la query. Pertanto, questa istanza specifica verrà eseguita al timestamp 2021-12-01 00:00:00 e calcolerà gli aggregati al minuto dall'intervallo temporale 2021-11-30 23:50:00 al 01/12/2021 00:01:00. Allo stesso modo, l'istanza successiva di questa query viene attivata al timestamp 2021-12-01 00:05:00 e in tal caso, la query calcolerà gli aggregati al minuto dall'intervallo di tempo 2021-11-30 23:55:00 al 2021-12-01 00:06:00. Pertanto, il parametro @scheduled_runtime fornisce una query pianificata per precalcolare gli aggregati per gli intervalli di tempo configurati utilizzando il tempo di invocazione per le query.

Nota che due istanze successive della query si sovrappongono nei rispettivi intervalli di tempo. Questo è qualcosa che puoi controllare in base alle tue esigenze. In questo caso, questa sovrapposizione consente a queste query di aggiornare gli aggregati sulla base di tutti i dati il cui arrivo è stato leggermente ritardato, fino a 5 minuti in questo esempio. Per garantire la correttezza delle query materializzate, Timestream for LiveAnalytics assicura che la query del 2021-12-01 00:05:00 venga eseguita solo dopo il completamento della query del 2021-12-01 00:00:00 e che i risultati di queste ultime query possano aggiornare qualsiasi aggregato precedentemente materializzato utilizzando se viene generato un valore più recente. Ad esempio, se alcuni dati al timestamp 2021-11-30 23:59:00 sono arrivati dopo l'esecuzione della query per il 2021-12-01 00:00:00 ma prima della query per il 2021-12-01 00:05:00, l'esecuzione del 2021-12-01 00:05:00 ricalcola gli aggregati per il minuto 2021-11-30 23:59:00 e ciò comporterà l'aggiornamento dell'aggregato precedente con il valore appena calcolato. Puoi fare affidamento su questa semantica delle query pianificate per trovare un compromesso tra la rapidità con cui aggiorni i tuoi precalcoli e la capacità di gestire con garbo alcuni dati con arrivo ritardato. Di seguito vengono illustrate ulteriori considerazioni su come bilanciare questa cadenza di aggiornamento con l'aggiornamento dei dati e su come affrontare l'aggiornamento degli aggregati per i dati che arrivano con un ritardo ancora

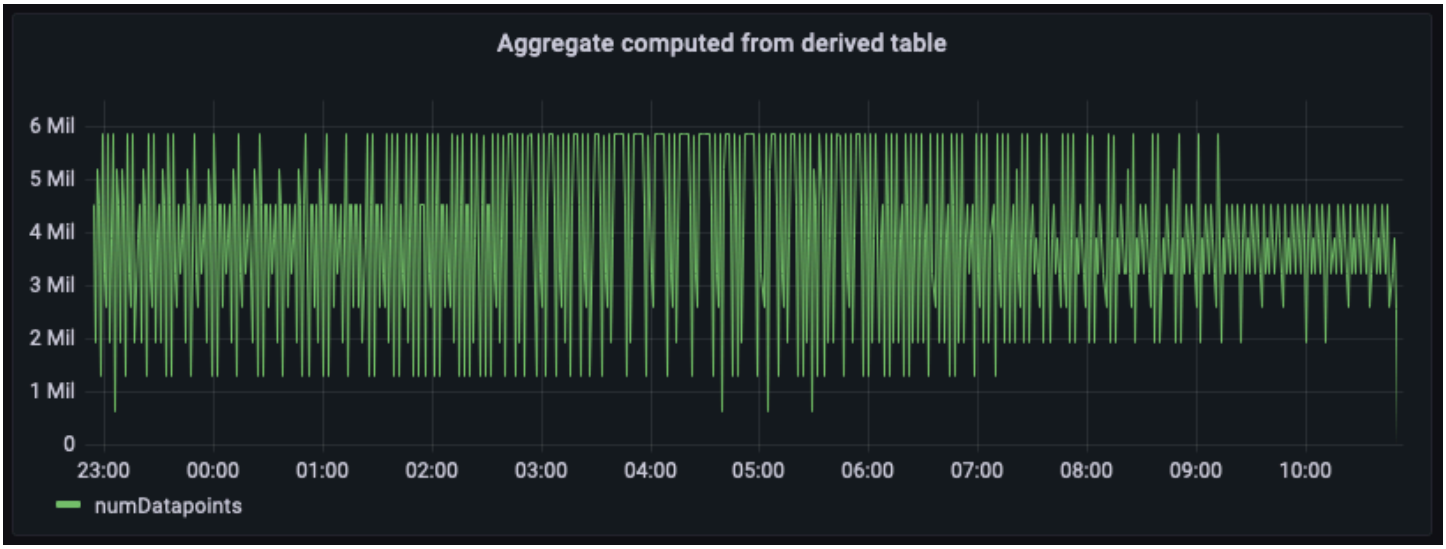
maggiore o se la fonte del calcolo pianificato ha valori aggiornati che richiederebbero il ricalcolo degli aggregati.

Ogni calcolo pianificato ha una configurazione di notifica in cui Timestream for invia una notifica di ogni esecuzione di una configurazione pianificata. LiveAnalytics È possibile configurare un SNS argomento per ricevere notifiche per ogni chiamata. Oltre allo stato di successo o di fallimento di un'istanza specifica, dispone anche di diverse statistiche come il tempo impiegato per l'esecuzione del calcolo, il numero di byte scansionati dal calcolo e il numero di byte scritti dal calcolo nella tabella di destinazione. È possibile utilizzare queste statistiche per ottimizzare ulteriormente la query, pianificare la configurazione o tenere traccia della spesa per le query pianificate. Un aspetto degno di nota è il tempo di esecuzione di un'istanza. In questo esempio, il calcolo pianificato è configurato per l'esecuzione ogni 5 minuti. Il tempo di esecuzione determinerà il ritardo con cui il precalcolo sarà disponibile, il che definirà anche il ritardo nella dashboard quando si utilizzano i dati precalcolati nelle dashboard. Inoltre, se questo ritardo è costantemente superiore all'intervallo di aggiornamento, ad esempio, se il tempo di esecuzione è superiore a 5 minuti per un calcolo configurato per l'aggiornamento ogni 5 minuti, è importante ottimizzare il calcolo in modo che venga eseguito più velocemente per evitare ulteriori ritardi nelle dashboard.

Aggregazione da tabella derivata

Ora che hai impostato le query pianificate e che gli aggregati sono stati precalcolati e materializzati in un altro Timestream per la LiveAnalytics tabella specificata nella configurazione di destinazione del calcolo pianificato, puoi utilizzare i dati in quella tabella per scrivere query per alimentare i dashboard. SQL Di seguito è riportato un equivalente della query che utilizza i preaggregati materializzati per generare l'aggregato di conteggio dei punti dati al minuto per us-east-1.

```
SELECT bin(time, 1m) as minute, SUM(numDataPoints) as numDatapoints
FROM "derived"."per_minute_aggs_pt5m"
WHERE time BETWEEN from_milliseconds(1636699996445) AND
 from_milliseconds(1636743196445)
    AND region = 'us-east-1'
GROUP BY bin(time, 1m)
ORDER BY 1 desc
```



La figura precedente riporta l'aggregato calcolato dalla tabella aggregata. Confrontando questo pannello con il pannello calcolato a partire dai dati di origine grezzi, noterete che corrispondono esattamente, anche se questi aggregati subiscono un ritardo di alcuni minuti, a causa dell'intervallo di aggiornamento configurato per il calcolo programmato più il tempo necessario per eseguirlo.

Questa interrogazione sui dati precalcolati analizza dati di diversi ordini di grandezza inferiori rispetto agli aggregati calcolati sui dati di origine grezza. A seconda della granularità delle aggregazioni, questa riduzione può facilmente ridurre di 100 volte i costi e la latenza delle query. L'esecuzione di questo calcolo pianificato comporta un costo. Tuttavia, a seconda della frequenza con cui questi dashboard vengono aggiornati e del numero di utenti simultanei che li caricano, si finisce per ridurre in modo significativo i costi complessivi utilizzando questi precalcoli. E questo si aggiunge ai tempi di caricamento delle dashboard da 10 a 100 volte più rapidi.

Aggregazione che combina tabelle di origine e derivate

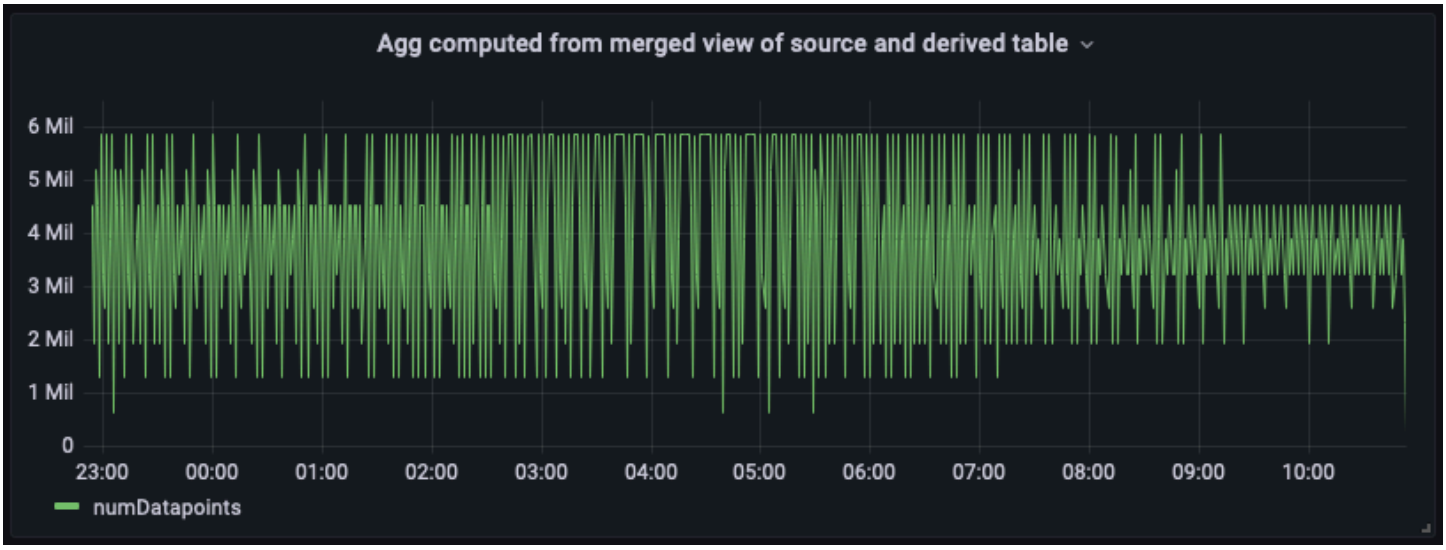
I dashboard creati utilizzando le tabelle derivate possono presentare un ritardo. Se lo scenario applicativo richiede che i dashboard contengano i dati più recenti, puoi utilizzare la potenza e la flessibilità del SQL supporto di Timestream for per LiveAnalytics combinare i dati più recenti della tabella di origine con gli aggregati storici della tabella derivata per formare una vista unita. Questa visualizzazione unita utilizza la semantica di unione SQL e non sovrapposizione degli intervalli di tempo della tabella di origine e della tabella derivata. Nell'esempio seguente, stiamo usando il termine «derivato».» tabella derivata «per_minute_aggs_pt5m». Poiché il calcolo pianificato per quella tabella derivata si aggiorna una volta ogni 5 minuti (in base alla specifica dell'espressione di pianificazione), la seguente query utilizza i 15 minuti di dati più recenti della tabella di origine e tutti i dati più vecchi di 15 minuti dalla tabella derivata, quindi unisce i risultati per creare la vista unita che offre il meglio di entrambi i mondi: l'economia e la bassa latenza leggendo gli aggregati precalcolati

più vecchi dalla tabella derivata e la freschezza dell'aggregazione proviene dalla tabella dei sorgenti per potenziare i casi d'uso dell'analisi in tempo reale.

Tieni presente che questo approccio di unione avrà una latenza di query leggermente superiore rispetto alla sola interrogazione della tabella derivata e avrà anche una scansione dei dati leggermente superiore, poiché aggrega i dati grezzi in tempo reale per riempire l'intervallo di tempo più recente. Tuttavia, questa visualizzazione unificata sarà comunque notevolmente più veloce ed economica rispetto all'aggregazione immediata dalla tabella di origine, in particolare per le dashboard che riproducono giorni o settimane di dati. Per questo esempio, puoi ottimizzare gli intervalli di tempo in base alle esigenze di aggiornamento e alla tolleranza al ritardo dell'applicazione.

```
WITH aggregated_source_data AS (
    SELECT bin(time, 1m) as minute, SUM(CASE WHEN measure_name = 'metrics' THEN 20 ELSE
 5 END) as numDatapoints
    FROM "raw_data"."devops"
    WHERE time BETWEEN bin(from_milliseconds(1636743196439), 1m) - 15m AND
 from_milliseconds(1636743196439)
        AND region = 'us-east-1'
    GROUP BY bin(time, 1m)
), aggregated_derived_data AS (
    SELECT bin(time, 1m) as minute, SUM(numDataPoints) as numDatapoints
    FROM "derived"."per_minute_aggs_pt5m"
    WHERE time BETWEEN from_milliseconds(1636699996439) AND
 bin(from_milliseconds(1636743196439), 1m) - 15m
        AND region = 'us-east-1'
    GROUP BY bin(time, 1m)
)
SELECT minute, numDatapoints
FROM (
    (
    SELECT *
    FROM aggregated_derived_data
    )
    UNION
    (
    SELECT *
    FROM aggregated_source_data
    )
)
ORDER BY 1 desc
```

Di seguito è riportato il pannello della dashboard con questa visualizzazione unificata. Come puoi vedere, la dashboard ha un aspetto quasi identico alla vista calcolata dalla tabella derivata, tranne per il fatto che avrà la parte più up-to-date aggregata nell'estremità più a destra.



Aggregazione basata su calcoli pianificati aggiornati di frequente

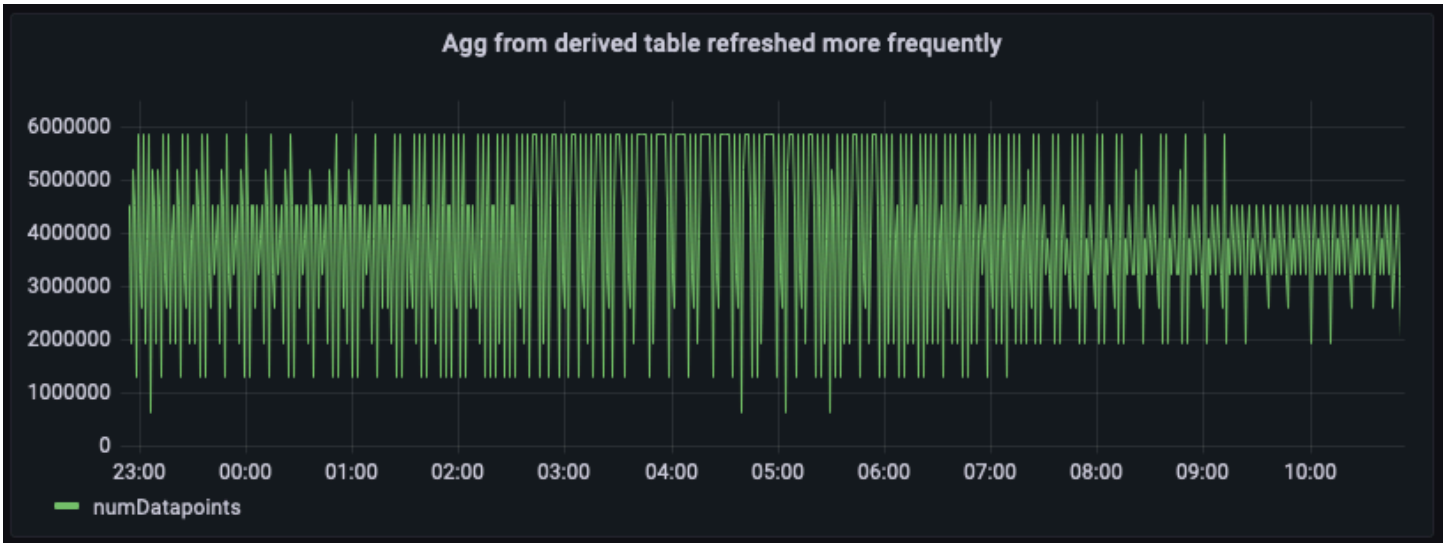
A seconda della frequenza con cui vengono caricate le dashboard e della latenza desiderata per la dashboard, esiste un altro approccio per ottenere risultati più aggiornati nella dashboard: fare in modo che il calcolo pianificato aggiorni gli aggregati più frequentemente. Ad esempio, di seguito è riportata la configurazione dello stesso calcolo pianificato, tranne per il fatto che si aggiorna una volta al minuto (si noti lo schedule express cron (0/1 * * *)? *)). Con questa configurazione, la tabella derivata per_minute_aggs_pt1m avrà aggregati molto più recenti rispetto allo scenario in cui il calcolo specificava una pianificazione di aggiornamento di una volta ogni 5 minuti.

```
{
    "Name": "MultiPT1mPerMinutePerRegionMeasureCount",
    "QueryString": "SELECT region, bin(time, 1m) as minute, SUM(CASE WHEN measure_name
 = 'metrics' THEN 20 ELSE 5 END) as numDataPoints FROM raw_data.devops WHERE time
 BETWEEN @scheduled_runtime - 10m AND @scheduled_runtime + 1m GROUP BY bin(time, 1m),
 region",
    "ScheduleConfiguration": {
        "ScheduleExpression": "cron(0/1 * * * ? *)"
    },
    "NotificationConfiguration": {
        "SnsConfiguration": {
            "TopicArn": "******"
        }
    },
```

```
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName": "derived",
            "TableName": "per_minute_aggs_pt1m",
            "TimeColumn": "minute",
            "DimensionMappings": [
                {
                    "Name": "region",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "numDataPoints",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "numDataPoints",
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

```
SELECT bin(time, 1m) as minute, SUM(numDataPoints) as numDatapoints
FROM "derived"."per_minute_aggs_pt1m"
WHERE time BETWEEN from_milliseconds(1636699996446) AND
 from_milliseconds(1636743196446)
    AND region = 'us-east-1'
GROUP BY bin(time, 1m), region
ORDER BY 1 desc
```

Poiché la tabella derivata contiene aggregati più recenti, ora puoi interrogare direttamente la tabella derivata per_minute_aggs_pt1m per ottenere aggregati più recenti, come si può vedere dalla query precedente e dallo snapshot della dashboard di seguito.



Tieni presente che l'aggiornamento del calcolo pianificato in base a una pianificazione più rapida (ad esempio 1 minuto rispetto a 5 minuti) aumenterà i costi di manutenzione per il calcolo pianificato. Il messaggio di notifica per l'esecuzione di ogni calcolo fornisce statistiche sulla quantità di dati scansionati e sulla quantità di scrittura nella tabella derivata. Allo stesso modo, se si utilizza la vista unita per unire la tabella derivata, si interrogano i costi sulla vista unita e la latenza di caricamento del dashboard sarà maggiore rispetto alla sola interrogazione della tabella derivata. Pertanto, l'approccio scelto dipenderà dalla frequenza di aggiornamento dei dashboard e dai costi di manutenzione delle query pianificate. Se decine di utenti aggiornano le dashboard una volta al minuto circa, un aggiornamento più frequente della tabella derivata comporterà probabilmente una riduzione dei costi complessivi.

Ultimo punto di ogni dispositivo

L'applicazione potrebbe richiedere la lettura dell'ultima misurazione emessa da un dispositivo. Esistono casi d'uso più generali per ottenere l'ultima misurazione di un dispositivo prima di una determinatadate/time or the first measurement for a device after a given date/time. Quando si dispone di milioni di dispositivi e anni di dati, questa ricerca potrebbe richiedere la scansione di grandi quantità di dati.

Di seguito è riportato un esempio di come è possibile utilizzare le interrogazioni pianificate per ottimizzare la ricerca dell'ultimo punto emesso da un dispositivo. È possibile utilizzare lo stesso schema per ottimizzare anche la prima query, se l'applicazione ne ha bisogno.

Argomenti

Calcolato dalla tabella di origine

Di seguito è riportata una query di esempio per trovare l'ultima misurazione emessa dai servizi in una distribuzione specifica (ad esempio, server per un determinato microservizio all'interno di una determinata regione, cella, silo e availability_zone). Nell'applicazione di esempio, questa query restituirà l'ultima misurazione per centinaia di server. Si noti inoltre che questa query ha un predicato temporale illimitato e cerca tutti i dati più vecchi di un determinato timestamp.

> **Note**
>
> Per informazioni sulle funzioni and, vedere. `max` `max_by` Funzioni di aggregazione

```
SELECT instance_name, MAX(time) AS time, MAX_BY(gc_pause, time) AS last_measure
FROM "raw_data"."devops"
WHERE time < from_milliseconds(1636685271872)
    AND measure_name = 'events'
    AND region = 'us-east-1'
    AND cell = 'us-east-1-cell-10'
    AND silo = 'us-east-1-cell-10-silo-3'
    AND availability_zone = 'us-east-1-1'
    AND microservice_name = 'hercules'
GROUP BY region, cell, silo, availability_zone, microservice_name,
    instance_name, process_name, jdk_version
ORDER BY instance_name, time DESC
```

Tabella derivata da precalcolare con granularità giornaliera

È possibile convertire il caso d'uso precedente in un calcolo pianificato. Se i requisiti dell'applicazione sono tali da richiedere l'ottenimento di questi valori per l'intero parco macchine in più aree, celle, silos, zone di disponibilità e microservizi, è possibile utilizzare un unico calcolo di pianificazione per precalcolare i valori per l'intero parco macchine. Questa è la potenza delle query pianificate

senza server di Timestream for LiveAnalytics, che consente a queste query di adattarsi ai requisiti di scalabilità dell'applicazione.

Di seguito è riportata una query per precalcolare l'ultimo punto su tutti i server per un determinato giorno. Nota che la query ha solo un predicato temporale e non un predicato sulle dimensioni. Il predicato temporale limita la query al giorno trascorso dal momento in cui il calcolo viene attivato in base all'espressione di pianificazione specificata.

```
SELECT region, cell, silo, availability_zone, microservice_name,
    instance_name, process_name, jdk_version,
    MAX(time) AS time, MAX_BY(gc_pause, time) AS last_measure
FROM raw_data.devops
WHERE time BETWEEN bin(@scheduled_runtime, 1d) - 1d AND bin(@scheduled_runtime, 1d)
    AND measure_name = 'events'
GROUP BY region, cell, silo, availability_zone, microservice_name,
    instance_name, process_name, jdk_version
```

Di seguito è riportata una configurazione per il calcolo pianificato utilizzando la query precedente, che esegue tale query UTC ogni giorno all'01:00 per calcolare l'aggregato del giorno precedente. L'espressione di pianificazione cron (0) 1 * *? *) controlla questo comportamento e viene eseguito un'ora dopo la fine della giornata, tenendo conto di tutti i dati che arrivano fino a un giorno di ritardo.

```
{
    "Name": "PT1DPerInstanceLastpoint",
    "QueryString": "SELECT region, cell, silo, availability_zone, microservice_name,
 instance_name, process_name, jdk_version, MAX(time) AS time, MAX_BY(gc_pause, time)
 AS last_measure FROM raw_data.devops WHERE time BETWEEN bin(@scheduled_runtime, 1d) -
 1d AND bin(@scheduled_runtime, 1d) AND measure_name = 'events' GROUP BY region, cell,
 silo, availability_zone, microservice_name, instance_name, process_name, jdk_version",
    "ScheduleConfiguration": {
        "ScheduleExpression": "cron(0 1 * * ? *)"
    },
    "NotificationConfiguration": {
        "SnsConfiguration": {
            "TopicArn": "******"
        }
    },
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName": "derived",
            "TableName": "per_timeseries_lastpoint_pt1d",
            "TimeColumn": "time",
```

```
            "DimensionMappings": [
                {
                    "Name": "region",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "cell",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "silo",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "availability_zone",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "microservice_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "instance_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "process_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "jdk_version",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "last_measure",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "last_measure",
                        "MeasureValueType": "DOUBLE"
                    }
                ]
            }
        }
```

```
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

Calcolato da una tabella derivata

Una volta definita la tabella derivata utilizzando la configurazione precedente e dopo che almeno un'istanza della query pianificata ha materializzato i dati nella tabella derivata, è ora possibile interrogare la tabella derivata per ottenere la misurazione più recente. Di seguito è riportato un esempio di query sulla tabella derivata.

```
SELECT instance_name, MAX(time) AS time, MAX_BY(last_measure, time) AS last_measure
FROM "derived"."per_timeseries_lastpoint_pt1d"
WHERE time < from_milliseconds(1636746715649)
    AND measure_name = 'last_measure'
    AND region = 'us-east-1'
    AND cell = 'us-east-1-cell-10'
    AND silo = 'us-east-1-cell-10-silo-3'
    AND availability_zone = 'us-east-1-1'
    AND microservice_name = 'hercules'
GROUP BY region, cell, silo, availability_zone, microservice_name,
    instance_name, process_name, jdk_version
ORDER BY instance_name, time DESC
```

Combinazione da tabella sorgente e derivata

Analogamente all'esempio precedente, tutti i dati della tabella derivata non avranno le scritture più recenti. Pertanto, è possibile utilizzare nuovamente uno schema simile a quello precedente per unire i dati della tabella derivata con i dati precedenti e utilizzare i dati di origine per il suggerimento rimanente. Di seguito è riportato un esempio di interrogazione di questo tipo che utilizza un UNION approccio simile. Poiché il requisito dell'applicazione è quello di trovare la misurazione più recente prima di un periodo di tempo e l'ora di inizio può essere passata, il modo in cui si scrive questa query consiste nell'utilizzare l'ora fornita, utilizzare i dati di origine per un massimo di un giorno a partire dall'ora specificata e quindi utilizzare la tabella derivata sui dati più vecchi. Come si può

vedere dall'esempio di query riportato di seguito, il predicato temporale sui dati di origine è limitato. Ciò garantisce un'elaborazione efficiente sulla tabella di origine, che contiene un volume di dati significativamente maggiore, e quindi il predicato temporale illimitato si trova sulla tabella derivata.

```
WITH last_point_derived AS (
    SELECT instance_name, MAX(time) AS time, MAX_BY(last_measure, time) AS last_measure
    FROM "derived"."per_timeseries_lastpoint_pt1d"
    WHERE time < from_milliseconds(1636746715649)
        AND measure_name = 'last_measure'
        AND region = 'us-east-1'
        AND cell = 'us-east-1-cell-10'
        AND silo = 'us-east-1-cell-10-silo-3'
        AND availability_zone = 'us-east-1-1'
        AND microservice_name = 'hercules'
    GROUP BY region, cell, silo, availability_zone, microservice_name,
        instance_name, process_name, jdk_version
), last_point_source AS (
    SELECT instance_name, MAX(time) AS time, MAX_BY(gc_pause, time) AS last_measure
    FROM "raw_data"."devops"
    WHERE time < from_milliseconds(1636746715649) AND time >
 from_milliseconds(1636746715649) - 26h
        AND measure_name = 'events'
        AND region = 'us-east-1'
        AND cell = 'us-east-1-cell-10'
        AND silo = 'us-east-1-cell-10-silo-3'
        AND availability_zone = 'us-east-1-1'
        AND microservice_name = 'hercules'
    GROUP BY region, cell, silo, availability_zone, microservice_name,
        instance_name, process_name, jdk_version
)
SELECT instance_name, MAX(time) AS time, MAX_BY(last_measure, time) AS last_measure
FROM (
    SELECT * FROM last_point_derived
    UNION
    SELECT * FROM last_point_source
)
GROUP BY instance_name
ORDER BY instance_name, time DESC
```

La precedente è solo un'illustrazione di come è possibile strutturare le tabelle derivate. Se disponi di anni di dati, puoi utilizzare più livelli di aggregazioni. Ad esempio, puoi aggiungere aggregati mensili a quelli giornalieri e aggiungere aggregati orari prima di quelli giornalieri. In questo modo puoi unire

i dati più recenti per inserire l'ultima ora, quelli con frequenza oraria per inserire l'ultimo giorno, quelli giornalieri per compilare l'ultimo mese e quelli mensili per inserire i dati più vecchi. Il numero di livelli impostati rispetto alla pianificazione degli aggiornamenti dipenderà dalle vostre esigenze relative alla frequenza con cui queste query vengono emesse e al numero di utenti che le emettono contemporaneamente.

Valori di dimensione univoci

Potresti avere un caso d'uso in cui disponi di dashboard in cui desideri utilizzare i valori univoci delle dimensioni come variabili per approfondire le metriche corrispondenti a una specifica porzione di dati. L'istantanea seguente è un esempio in cui la dashboard precompila i valori univoci di diverse dimensioni come region, cell, silo, microservice e availability_zone. Qui mostriamo un esempio di come è possibile utilizzare le query pianificate per velocizzare in modo significativo il calcolo di questi valori distinti di queste variabili in base alle metriche che stai monitorando.

Argomenti

Sui dati grezzi

Puoi usarlo SELECT DISTINCT per calcolare i valori distinti visti dai tuoi dati. Ad esempio, se si desidera ottenere i valori distinti della regione, è possibile utilizzare la query di questo modulo.

```
SELECT DISTINCT region
FROM "raw_data"."devops"
WHERE time > ago(1h)
ORDER BY 1
```

Potresti tracciare milioni di dispositivi e miliardi di serie temporali. Tuttavia, nella maggior parte dei casi, queste variabili interessanti riguardano dimensioni di cardinalità inferiori, in cui sono presenti da poche a decine di valori. L'elaborazione DISTINCT a partire da dati grezzi può richiedere la scansione di grandi volumi di dati.

Precalcola valori di dimensione univoci

Desideri che queste variabili si carichino rapidamente in modo che i dashboard siano interattivi. Inoltre, queste variabili vengono spesso calcolate su ogni caricamento del dashboard, quindi

è necessario che siano anche convenienti. È possibile ottimizzare la ricerca di queste variabili utilizzando query pianificate e materializzandole in una tabella derivata.

Innanzitutto, è necessario identificare le dimensioni per le quali è necessario calcolare DISTINCT i valori o le colonne da utilizzare nei predicati durante il calcolo del valore. DISTINCT

In questo esempio, puoi vedere che la dashboard sta inserendo valori distinti per le dimensioni region, cell, silo, availability_zone e microservice. Quindi puoi usare la query seguente per precalcolare questi valori unici.

```
SELECT region, cell, silo, availability_zone, microservice_name,
    min(@scheduled_runtime) AS time, COUNT(*) as numDataPoints
FROM raw_data.devops
WHERE time BETWEEN @scheduled_runtime - 15m AND @scheduled_runtime
GROUP BY region, cell, silo, availability_zone, microservice_name
```

Ci sono alcune cose importanti da notare qui.

- È possibile utilizzare un calcolo pianificato per precalcolare i valori per molte query diverse. Ad esempio, si utilizza la query precedente per precalcolare i valori per cinque diverse variabili. Quindi non è necessario uno per ogni variabile. È possibile utilizzare lo stesso schema per identificare calcoli condivisi su più pannelli per ottimizzare il numero di query pianificate da gestire.
- I valori univoci delle dimensioni non sono intrinsecamente dati di serie temporali. Quindi li converti in serie temporali usando @scheduled_runtime. Associando questi dati al parametro @scheduled_runtime, puoi anche tenere traccia di quali valori univoci sono comparsi in un determinato momento, creando così dati di serie temporali a partire da essi.
- Nell'esempio precedente, vedrete il tracciamento di un valore metrico. Questo esempio utilizza COUNT (*). Puoi calcolare altri aggregati significativi se desideri tenerne traccia per le tue dashboard.

Di seguito è riportata una configurazione per un calcolo pianificato utilizzando la query precedente. In questo esempio, è configurato per l'aggiornamento una volta ogni 15 minuti utilizzando l'espressione di pianificazione cron (0/15 * * *? *).

```
{
    "Name": "PT15mHighCardPerUniqueDimensions",
    "QueryString": "SELECT region, cell, silo, availability_zone, microservice_name,
 min(@scheduled_runtime) AS time, COUNT(*) as numDataPoints FROM raw_data.devops WHERE
```

```
time BETWEEN @scheduled_runtime - 15m AND @scheduled_runtime GROUP BY region, cell,
silo, availability_zone, microservice_name",
   "ScheduleConfiguration": {
       "ScheduleExpression": "cron(0/15 * * * ? *)"
   },
   "NotificationConfiguration": {
       "SnsConfiguration": {
           "TopicArn": "******"
       }
   },
   "TargetConfiguration": {
       "TimestreamConfiguration": {
           "DatabaseName": "derived",
           "TableName": "hc_unique_dimensions_pt15m",
           "TimeColumn": "time",
           "DimensionMappings": [
               {
                   "Name": "region",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "cell",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "silo",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "availability_zone",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "microservice_name",
                   "DimensionValueType": "VARCHAR"
               }
           ],
           "MultiMeasureMappings": {
               "TargetMultiMeasureName": "count_multi",
               "MultiMeasureAttributeMappings": [
                   {
                       "SourceColumn": "numDataPoints",
                       "MeasureValueType": "BIGINT"
                   }
```

```
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

Calcolo delle variabili dalla tabella derivata

Una volta che il calcolo pianificato ha prematerializzato i valori univoci nella tabella derivata hc_unique_dimensions_pt15m, è possibile utilizzare la tabella derivata per calcolare in modo efficiente i valori univoci delle dimensioni. Di seguito sono riportati alcuni esempi di query su come calcolare i valori univoci e su come utilizzare altre variabili come predicati in queste query con valori univoci.

Region

```
SELECT DISTINCT region
FROM "derived"."hc_unique_dimensions_pt15m"
WHERE time > ago(1h)
ORDER BY 1
```

Cella

```
SELECT DISTINCT cell
FROM "derived"."hc_unique_dimensions_pt15m"
WHERE time > ago(1h)
  AND region = '${region}'
ORDER BY 1
```

Silo

```
SELECT DISTINCT silo
FROM "derived"."hc_unique_dimensions_pt15m"
```

```
WHERE time > ago(1h)
   AND region = '${region}' AND cell = '${cell}'
ORDER BY 1
```

Microservizio

```
SELECT DISTINCT microservice_name
FROM "derived"."hc_unique_dimensions_pt15m"
WHERE time > ago(1h)
   AND region = '${region}' AND cell = '${cell}'
ORDER BY 1
```

Zona di disponibilità

```
SELECT DISTINCT availability_zone
FROM "derived"."hc_unique_dimensions_pt15m"
WHERE time > ago(1h)
   AND region = '${region}' AND cell = '${cell}' AND silo = '${silo}'
ORDER BY 1
```

Gestione dei dati in arrivo tardivo

Potrebbero verificarsi scenari in cui è possibile che i dati arrivino con notevole ritardo, ad esempio, il momento in cui i dati sono stati importati in Timestream for LiveAnalytics è notevolmente ritardato rispetto al timestamp associato alle righe che vengono importate. Negli esempi precedenti, avete visto come utilizzare gli intervalli di tempo definiti dal parametro @scheduled_runtime per tenere conto di alcuni dati che arrivano in ritardo. Tuttavia, se avete casi d'uso in cui i dati possono essere ritardati di ore o giorni, potrebbe essere necessario un modello diverso per assicurarvi che i calcoli preliminari nella tabella derivata siano aggiornati in modo appropriato per riflettere tali dati in arrivo in ritardo. Per informazioni generali sui dati in arrivo in ritardo, consulta. Scrittura di dati (inserti e sconvolgimenti)

Di seguito verranno illustrati due modi diversi per gestire questi dati in arrivo in ritardo.

- Se hai ritardi prevedibili nell'arrivo dei dati, puoi utilizzare un altro calcolo pianificato «recuperato» per aggiornare gli aggregati per i dati in arrivo in ritardo.
- Se hai ritardi imprevedibili o dati occasionali relativi agli arrivi tardivi, puoi utilizzare le esecuzioni manuali per aggiornare le tabelle derivate.

Questa discussione tratta gli scenari di arrivo tardivo dei dati. Tuttavia, gli stessi principi si applicano alle correzioni dei dati, quando hai modificato i dati nella tabella di origine e desideri aggiornare gli aggregati nelle tabelle derivate.

Argomenti

Interrogazioni di recupero pianificate

Interroga i dati aggregati che sono arrivati in tempo

Di seguito è riportato uno schema che illustra come è possibile utilizzare un metodo automatizzato per aggiornare gli aggregati in caso di ritardi prevedibili nell'arrivo dei dati. Considera uno degli esempi precedenti di calcolo pianificato su dati in tempo reale riportati di seguito. Questo calcolo pianificato aggiorna la tabella derivata una volta ogni 30 minuti e tiene già conto dei dati con un ritardo fino a un'ora.

```
{
    "Name": "MultiPT30mPerHrPerTimeseriesDPCount",
    "QueryString": "SELECT region, cell, silo, availability_zone, microservice_name,
 instance_type, os_version, instance_name, process_name, jdk_version, bin(time,
 1h) as hour, SUM(CASE WHEN measure_name = 'metrics' THEN 20 ELSE 5 END) as
 numDataPoints FROM raw_data.devops WHERE time BETWEEN bin(@scheduled_runtime, 1h)
 - 1h AND @scheduled_runtime + 1h GROUP BY region, cell, silo, availability_zone,
 microservice_name, instance_type, os_version, instance_name, process_name,
 jdk_version, bin(time, 1h)",
    "ScheduleConfiguration": {
        "ScheduleExpression": "cron(0/30 * * * ? *)"
    },
    "NotificationConfiguration": {
        "SnsConfiguration": {
            "TopicArn": "******"
        }
    },
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName": "derived",
            "TableName": "dp_per_timeseries_per_hr",
            "TimeColumn": "hour",
            "DimensionMappings": [
```

```
                {
                    "Name": "region",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "cell",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "silo",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "availability_zone",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "microservice_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "instance_type",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "os_version",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "instance_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "process_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "jdk_version",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "numDataPoints",
                "MultiMeasureAttributeMappings": [
```

```
                    {
                        "SourceColumn": "numDataPoints",
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

Interrogazione di recupero che aggiorna gli aggregati per i dati in arrivo in ritardo

Ora, se consideri il caso, i tuoi dati possono subire un ritardo di circa 12 ore. Di seguito è riportata una variante della stessa query. Tuttavia, la differenza è che calcola gli aggregati sui dati che subiscono un ritardo fino a 12 ore rispetto a quando viene attivato il calcolo pianificato. Ad esempio, come vedi la query nell'esempio seguente, l'intervallo di tempo a cui si rivolge questa query è compreso tra 2 e 14 ore prima dell'attivazione della query. Inoltre, se notate l'espressione di pianificazione cron (0) 0,12 * *? *), attiverà il calcolo ogni giorno alle 00:00 UTC e alle 12:00. UTC Pertanto, quando la query viene attivata il 01/12/2021 00:00:00, la query aggiorna gli aggregati nell'intervallo di tempo 2021-11-30 dalle 10:00:00 al 2021-11-30 22:00:00. Le query pianificate utilizzano una semantica upsert simile alle scritture di Timestream for, in cui questa query LiveAnalytics di recupero aggiornerà i valori aggregati con valori più recenti se nella finestra sono presenti dati in arrivo in ritardo o se vengono trovati aggregati più recenti (ad esempio, in questo aggregato viene visualizzato un nuovo raggruppamento che non era presente quando è stato attivato il calcolo pianificato originale), quindi il nuovo aggregato verrà inserito nella tabella derivata. Allo stesso modo, quando l'istanza successiva viene attivata il 2021-12-01 12:00:00, quell'istanza aggiornerà gli aggregati nell'intervallo 2021-11-30 22:00:00 e 2021-12-01 10:00:00.

```
       {
    "Name": "MultiPT12HPerHrPerTimeseriesDPCountCatchUp",
    "QueryString": "SELECT region, cell, silo, availability_zone, microservice_name,
 instance_type, os_version, instance_name, process_name, jdk_version, bin(time, 1h)
```

```
as hour, SUM(CASE WHEN measure_name = 'metrics' THEN 20 ELSE 5 END) as numDataPoints
FROM raw_data.devops WHERE time BETWEEN bin(@scheduled_runtime, 1h) - 14h AND
bin(@scheduled_runtime, 1h) - 2h GROUP BY region, cell, silo, availability_zone,
microservice_name, instance_type, os_version, instance_name, process_name,
jdk_version, bin(time, 1h)",
   "ScheduleConfiguration": {
       "ScheduleExpression": "cron(0 0,12 * * ? *)"
   },
   "NotificationConfiguration": {
       "SnsConfiguration": {
           "TopicArn": "******"
       }
   },
   "TargetConfiguration": {
       "TimestreamConfiguration": {
           "DatabaseName": "derived",
           "TableName": "dp_per_timeseries_per_hr",
           "TimeColumn": "hour",
           "DimensionMappings": [
               {
                   "Name": "region",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "cell",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "silo",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "availability_zone",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "microservice_name",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "instance_type",
                   "DimensionValueType": "VARCHAR"
               },
               {
```

```
                    "Name": "os_version",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "instance_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "process_name",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "Name": "jdk_version",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "numDataPoints",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "numDataPoints",
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

L'esempio precedente è un'illustrazione che presuppone che l'arrivo in ritardo sia limitato a 12 ore e che sia possibile aggiornare la tabella derivata una volta ogni 12 ore per i dati che arrivano dopo la finestra in tempo reale. È possibile adattare questo modello per aggiornare la tabella derivata una volta ogni ora in modo che rifletta prima i dati che arrivano in ritardo. Allo stesso modo, puoi

adattare l'intervallo di tempo in modo che sia più vecchio di 12 ore, ad esempio un giorno o anche una settimana o più, per gestire dati prevedibili in arrivo tardivo.

Esecuzioni manuali per dati imprevedibili in arrivo in ritardo

In alcuni casi possono esserci dati imprevedibili con arrivo tardivo o in cui si sono apportate modifiche ai dati di origine e si sono aggiornati alcuni valori a posteriori. In tutti questi casi, è possibile attivare manualmente le interrogazioni pianificate per aggiornare la tabella derivata. Di seguito è riportato un esempio di come è possibile raggiungere questo obiettivo.

Supponiamo di avere il caso d'uso in cui il calcolo è scritto nella tabella derivata dp_per_timeseries_per_hr. I tuoi dati di base nella tabella devops sono stati aggiornati nell'intervallo di tempo 2021-11-30 23:00:00 - 2021-12-01 00:00:00. Esistono due diverse query pianificate che possono essere utilizzate per aggiornare questa tabella derivata: Multi 0 e Multi. PT3 mPerHr PerTimeseries DPCount PT12HPerHrPerTimeseriesDPCountCatchUp Ogni calcolo pianificato per cui crei in Timestream ne LiveAnalytics ha uno univoco ARN che si ottiene quando si crea il calcolo o quando si esegue un'operazione sull'elenco. È possibile utilizzare il ARN per il calcolo e un valore per il parametro @scheduled_runtime utilizzato dalla query per eseguire questa operazione.

Supponiamo che il calcolo per Multi PT3 0 mPerHr PerTimeseries DPCount abbia un ARN arn_1 e che tu voglia usare questo calcolo per aggiornare la tabella derivata. Poiché il calcolo pianificato precedente aggiorna gli aggregati 1 ora prima e 1 ora dopo il valore @scheduled_runtime, puoi coprire l'intervallo di tempo per l'aggiornamento (2021-11-30 23:00:00 - 2021-12-01 00:00:00) utilizzando un valore di 2021-12-01 00:00:00 per il parametro @scheduled_runtime. A tale scopo, è possibile utilizzare il per passare il valore di questo calcolo e il valore del parametro time in secondi epoch (in). ExecuteScheduledQuery API ARN UTC Di seguito è riportato un esempio che utilizza AWS CLI e puoi seguire lo stesso schema utilizzando uno dei formati SDKs supportati da Timestream for. LiveAnalytics

```
aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638316800 --profile profile --region us-east-1
```

Nell'esempio precedente, profile è il AWS profilo che dispone dei privilegi appropriati per effettuare questa API chiamata e 1638316800 corrisponde al secondo periodo del 2021-12-01 00:00:00. Questo trigger manuale si comporta quasi come il trigger automatico, presupponendo che il sistema abbia attivato questa chiamata nel periodo di tempo desiderato.

Se hai ricevuto un aggiornamento in un periodo di tempo più lungo, ad esempio i dati di base sono stati aggiornati per il 2021-11-30 23:00:00 - 01/12/2021 11:00:00, puoi attivare le query precedenti

più volte per coprire l'intero intervallo di tempo. Ad esempio, potresti eseguire sei diverse esecuzioni come segue.

```
aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638316800 --profile profile --region us-east-1

aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638324000 --profile profile --region us-east-1

aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638331200 --profile profile --region us-east-1

aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638338400 --profile profile --region us-east-1

aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638345600 --profile profile --region us-east-1

aws timestream-query execute-scheduled-query --scheduled-query-arn arn_1 --invocation-
time 1638352800 --profile profile --region us-east-1
```

I sei comandi precedenti corrispondono al calcolo pianificato richiamato al 2021-12-01 00:00:00, 2021-12-01 02:00:00, 2021-12-01 04:0:00, 2021-12-01 06:00:00, 2021-12-01 08:00:00e 2021-12-01 10:00:

In alternativa, puoi utilizzare il calcolo Multi PT12HPerHrPerTimeseriesDPCountCatchUp attivato alle 13:00:00 del 01/12/2021 per un'esecuzione per aggiornare gli aggregati per l'intero intervallo di tempo di 12 ore. Ad esempio, se arn_2 è usato per quel calcolo, puoi eseguire il ARN seguente comando da. CLI

```
aws timestream-query execute-scheduled-query --scheduled-query-arn arn_2 --invocation-
time 1638363600 --profile profile --region us-east-1
```

Vale la pena notare che per un trigger manuale, è possibile utilizzare un timestamp per il parametro invocation-time che non deve essere allineato con i timestamp di quel trigger automatico. Ad esempio, nell'esempio precedente, hai attivato il calcolo all'ora 2021-12-01 13:00:00 anche se la pianificazione automatica si attiva solo nei timestamp 2021-12-01 10:00:00, 2021-12-01 12:00:00e 2021-12-02 00:00:00. Timestream for offre la flessibilità necessaria per attivarlo con i valori appropriati necessari per le operazioni manuali. LiveAnalytics

Di seguito sono riportate alcune considerazioni importanti sull'utilizzo di. ExecuteScheduledQuery API

- Se stai attivando più di queste invocazioni, devi assicurarti che queste invocazioni non generino risultati in intervalli di tempo sovrapposti. Ad esempio, negli esempi precedenti, c'erano sei invocazioni. Ogni chiamata copre un intervallo di tempo di 2 ore, quindi i timestamp di invocazione sono stati distribuiti di due ore ciascuno per evitare sovrapposizioni negli aggiornamenti. Ciò garantisce che i dati nella tabella derivata finiscano in uno stato in cui le corrispondenze siano aggregate dalla tabella di origine. Se non riesci a garantire intervalli di tempo non sovrapposti, assicurati che queste esecuzioni vengano attivate in sequenza una dopo l'altra. Se si attivano più esecuzioni contemporaneamente che si sovrappongono nei rispettivi intervalli di tempo, nei report sugli errori relativi a queste esecuzioni è possibile che si verifichino conflitti di versione. Ai risultati generati da una chiamata di interrogazione pianificata viene assegnata una versione in base a quando è stata attivata la chiamata. Pertanto, le righe generate da chiamate più recenti hanno versioni successive. Un record di versione superiore può sovrascrivere un record di versione inferiore. Per le query pianificate attivate automaticamente, Timestream for gestisce LiveAnalytics automaticamente le pianificazioni in modo da non visualizzare questi problemi anche se le chiamate successive hanno intervalli di tempo sovrapposti.
- notato in precedenza, è possibile attivare le invocazioni con qualsiasi valore di timestamp per @scheduled_runtime. Quindi è tua responsabilità impostare i valori in modo appropriato in modo che gli intervalli di tempo appropriati vengano aggiornati nella tabella derivata corrispondente agli intervalli in cui i dati sono stati aggiornati nella tabella di origine.
- È inoltre possibile utilizzare questi trigger manuali per le query pianificate che si trovano nello DISABLED stato. Ciò consente di definire interrogazioni speciali che non vengono eseguite in una pianificazione automatica, poiché si trovano nello DISABLED stato. Piuttosto, puoi utilizzare i trigger manuali su di essi per gestire le correzioni dei dati o i casi di utilizzo in ritardo.

Riempimento dei precalcoli storici

Quando crei un calcolo pianificato, Timestream for LiveAnalytics gestisce le esecuzioni delle query in futuro, laddove l'aggiornamento è regolato dall'espressione di pianificazione fornita. A seconda della quantità di dati storici contenuta nella tabella di origine, potresti voler aggiornare la tabella derivata con aggregati corrispondenti ai dati storici. Puoi utilizzare la logica precedente per i trigger manuali per riempire nuovamente gli aggregati storici.

Ad esempio, se consideriamo la tabella derivata per_timeseries_lastpoint_pt1d, il calcolo pianificato viene aggiornato una volta al giorno per il giorno passato. Se la tabella di origine contiene un anno di dati, puoi utilizzarla ARN per questo calcolo pianificato e attivarla manualmente per ogni giorno fino a un anno in modo che la tabella derivata contenga tutte le query storiche compilate. Nota che qui si applicano tutte le avvertenze relative ai trigger manuali. Inoltre, se la tabella derivata è configurata in

modo tale che l'acquisizione storica venga scritta nell'archivio magnetico della tabella derivata, tenete presente le migliori pratiche e i limiti per le scritture sull'archivio magnetico.

Esempi di interrogazioni pianificate

Questa sezione contiene esempi di come utilizzare Timestream for Scheduled Queries per LiveAnalytics ottimizzare i costi e i tempi di caricamento della dashboard durante la visualizzazione delle statistiche relative all'intera flotta e monitorare efficacemente la flotta di dispositivi. Le query pianificate in Timestream for ti LiveAnalytics consentono di esprimere le tue query utilizzando l'intera superficie di Timestream for. SQL LiveAnalytics La tua query può includere una o più tabelle di origine, eseguire aggregazioni o qualsiasi altra query consentita dal SQL linguaggio di Timestream for e quindi archiviare i risultati LiveAnalytics della query in un'altra tabella di destinazione in Timestream for. LiveAnalytics

Questa sezione si riferisce alla tabella di destinazione di una query pianificata come tabella derivata.

Ad esempio, utilizzeremo un' DevOps applicazione in cui si monitora un'ampia flotta di server distribuiti in più distribuzioni (ad esempio regioni, celle e silos), più microservizi e si monitorano le statistiche a livello di flotta utilizzando Timestream for. LiveAnalytics Lo schema di esempio che utilizzeremo è descritto in Schema di esempio per le query pianificate.

Verranno descritti i seguenti scenari.

- Come convertire una dashboard, tracciare statistiche aggregate dai dati grezzi che inserite in Timestream per LiveAnalytics trasformarle in una query pianificata e quindi come utilizzare gli aggregati precalcolati per creare una nuova dashboard che mostri statistiche aggregate.
- Come combinare le interrogazioni pianificate per ottenere una visualizzazione aggregata e i dati granulari non elaborati per approfondire i dettagli. Ciò consente di archiviare e analizzare i dati grezzi ottimizzando al contempo le operazioni comuni a livello di flotta utilizzando query pianificate.
- Come ottimizzare i costi utilizzando le query pianificate individuando quali aggregati vengono utilizzati in più dashboard e hanno la stessa query pianificata popolare più pannelli nello stesso o in più dashboard.

Argomenti

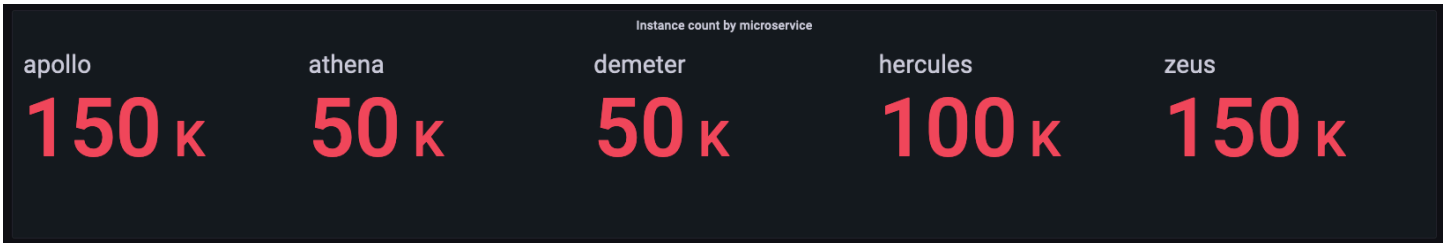
- Conversione di un pannello di controllo aggregato in una query pianificata
- Utilizzo di query pianificate e dati non elaborati per i drill down
- Ottimizzazione dei costi condividendo le query pianificate tra i dashboard

- Confronto di un'interrogazione su una tabella di base con un'interrogazione dei risultati di una query pianificata

Conversione di un pannello di controllo aggregato in una query pianificata

Supponiamo che stiate calcolando le statistiche relative all'intero parco macchine, ad esempio il numero di host presenti nel parco macchine in base ai cinque microservizi e alle sei regioni in cui viene distribuito il servizio. Dall'istantanea qui sotto, puoi vedere che ci sono 500.000 server che emettono metriche e alcune delle regioni più grandi (ad esempio, us-east-1) hanno più di 200.000 server.

Il calcolo di questi aggregati, in cui si calcolano nomi di istanze distinti su centinaia di gigabyte di dati, può comportare una latenza delle query di decine di secondi, oltre al costo della scansione dei dati.



Interrogazione originale sulla dashboard

L'aggregato mostrato nel pannello della dashboard viene calcolato, a partire da dati grezzi, utilizzando la query seguente. La query utilizza più SQL costrutti, come conteggi distinti e più funzioni di aggregazione.

```
SELECT CASE WHEN microservice_name = 'apollo' THEN num_instances ELSE NULL END AS
 apollo,
    CASE WHEN microservice_name = 'athena' THEN num_instances ELSE NULL END AS athena,
    CASE WHEN microservice_name = 'demeter' THEN num_instances ELSE NULL END AS
 demeter,
    CASE WHEN microservice_name = 'hercules' THEN num_instances ELSE NULL END AS
 hercules,
    CASE WHEN microservice_name = 'zeus' THEN num_instances ELSE NULL END AS zeus
FROM (
    SELECT microservice_name, SUM(num_instances) AS num_instances
    FROM (
        SELECT microservice_name, COUNT(DISTINCT instance_name) as num_instances
        FROM "raw_data"."devops"
        WHERE time BETWEEN from_milliseconds(1636526171043) AND
 from_milliseconds(1636612571043)
```

```
            AND measure_name = 'metrics'
        GROUP BY region, cell, silo, availability_zone, microservice_name
    )
    GROUP BY microservice_name
)
```

Conversione in una query pianificata

L'interrogazione precedente può essere convertita in un'interrogazione pianificata come segue. Innanzitutto si calcolano i nomi host distinti all'interno di una determinata distribuzione in una regione, cella, silo, zona di disponibilità e microservizio. Quindi si sommano gli host per calcolare un numero di host all'ora per microservizio. Utilizzando il `@scheduled_runtime` parametro supportato dalle query pianificate, è possibile ricalcolarlo per l'ultima ora in cui viene richiamata la query. La `WHERE` clausola `bin(@scheduled_runtime, 1h)` in della query interna assicura che, anche se l'interrogazione è pianificata a metà ora, si ottengono comunque i dati per l'intera ora.

Anche se la query calcola aggregati orari, come illustrato nella configurazione di calcolo pianificata, è impostata per l'aggiornamento ogni mezz'ora, in modo da ottenere più rapidamente gli aggiornamenti nella tabella derivata. È possibile regolarlo in base ai requisiti di freschezza, ad esempio ricalcolando gli aggregati ogni 15 minuti o ricalcolandoli in base ai limiti delle ore.

```
SELECT microservice_name, hour, SUM(num_instances) AS num_instances
FROM (
        SELECT microservice_name, bin(time, 1h) AS hour,
            COUNT(DISTINCT instance_name) as num_instances
       FROM raw_data.devops
       WHERE time BETWEEN bin(@scheduled_runtime, 1h) - 1h AND @scheduled_runtime

           AND measure_name = 'metrics'
       GROUP BY region, cell, silo, availability_zone, microservice_name, bin(time, 1h)

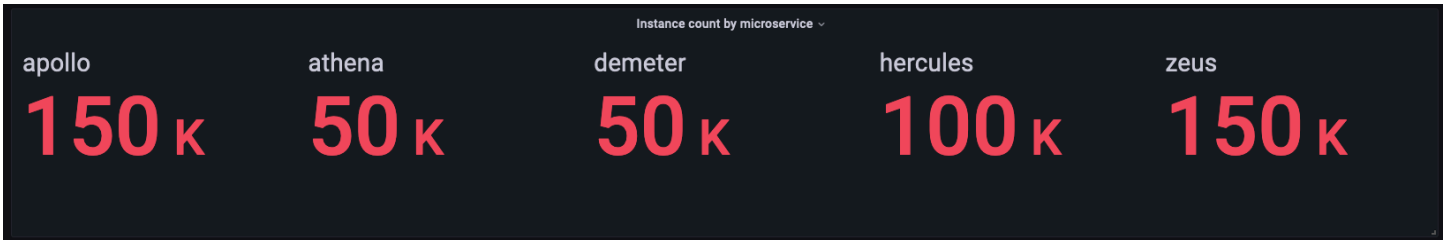
     )
GROUP BY microservice_name, hour
```

```
{
    "Name": "MultiPT30mHostCountMicroservicePerHr",
    "QueryString": "SELECT microservice_name, hour, SUM(num_instances) AS num_instances
    FROM (        SELECT microservice_name, bin(time, 1h) AS hour, COUNT(DISTINCT
 instance_name) as num_instances        FROM raw_data.devops        WHERE time BETWEEN
 bin(@scheduled_runtime, 1h) - 1h AND @scheduled_runtime            AND measure_name
```

```
 = 'metrics'        GROUP BY region, cell, silo, availability_zone, microservice_name,
 bin(time, 1h)    )    GROUP BY microservice_name, hour",
    "ScheduleConfiguration": {
        "ScheduleExpression": "cron(0/30 * * * ? *)"
    },
    "NotificationConfiguration": {
        "SnsConfiguration": {
            "TopicArn": "******"
        }
    },
    "TargetConfiguration": {
        "TimestreamConfiguration": {
            "DatabaseName": "derived",
            "TableName": "host_count_pt1h",
            "TimeColumn": "hour",
            "DimensionMappings": [
                {
                    "Name": "microservice_name",
                    "DimensionValueType": "VARCHAR"
                }
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "num_instances",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "num_instances",
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

Utilizzo dei risultati precalcolati in una nuova dashboard

Ora vedrai come creare la tua dashboard di visualizzazione aggregata utilizzando la tabella derivata dalla query pianificata che hai creato. Dall'istantanea del dashboard, potrai anche verificare che anche gli aggregati calcolati dalla tabella derivata e dalla tabella di base corrispondano. Una volta create le dashboard utilizzando le tabelle derivate, noterete i tempi di caricamento notevolmente più rapidi e i costi inferiori legati all'utilizzo delle tabelle derivate rispetto al calcolo di questi aggregati a partire dai dati grezzi. Di seguito è riportata un'istantanea della dashboard che utilizza dati precalcolati e la query utilizzata per eseguire il rendering di questo pannello utilizzando i dati precalcolati memorizzati nella tabella «derivata».» host_count_pt1h». Nota che la struttura della query è molto simile alla query utilizzata nella dashboard sui dati grezzi, tranne per il fatto che utilizza la tabella derivata che calcola già i conteggi distinti che questa query sta aggregando.




```
SELECT CASE WHEN microservice_name = 'apollo' THEN num_instances ELSE NULL END AS
 apollo,
    CASE WHEN microservice_name = 'athena' THEN num_instances ELSE NULL END AS athena,
    CASE WHEN microservice_name = 'demeter' THEN num_instances ELSE NULL END AS
 demeter,
    CASE WHEN microservice_name = 'hercules' THEN num_instances ELSE NULL END AS
 hercules,
    CASE WHEN microservice_name = 'zeus' THEN num_instances ELSE NULL END AS zeus
FROM (
    SELECT microservice_name, AVG(num_instances) AS num_instances
    FROM (
        SELECT microservice_name, bin(time, 1h), SUM(num_instances) as num_instances
        FROM "derived"."host_count_pt1h"
        WHERE time BETWEEN from_milliseconds(1636567785421) AND
 from_milliseconds(1636654185421)
            AND measure_name = 'num_instances'
        GROUP BY microservice_name, bin(time, 1h)
    )
    GROUP BY microservice_name
)
```

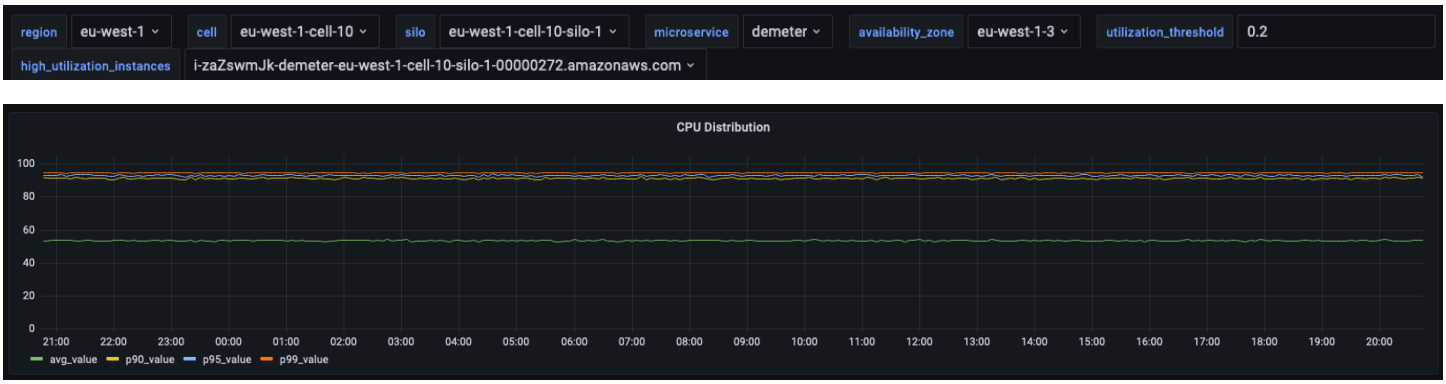
Utilizzo di query pianificate e dati non elaborati per i drill down

Puoi utilizzare le statistiche aggregate del tuo parco macchine per identificare le aree che necessitano di approfondimenti e quindi utilizzare i dati grezzi per approfondire i dati granulari e ottenere informazioni più approfondite.

In questo esempio, vedrete come utilizzare la dashboard aggregata per identificare qualsiasi implementazione (una distribuzione riguarda un determinato microservizio all'interno di una determinata regione, cella, silo e zona di disponibilità) che sembra avere un utilizzo maggiore rispetto ad altre implementazioni. CPU È quindi possibile approfondire per comprendere meglio l'utilizzo dei dati grezzi. Poiché questi approfondimenti potrebbero essere poco frequenti e accedere solo ai dati pertinenti alla distribuzione, è possibile utilizzare i dati grezzi per questa analisi e non è necessario utilizzare query pianificate.

Drill down per distribuzione

La dashboard seguente fornisce informazioni dettagliate su statistiche più granulari e a livello di server all'interno di una determinata implementazione. Per aiutarti ad analizzare in dettaglio le diverse parti della tua flotta, questa dashboard utilizza variabili come region, cell, silo, microservice e availability_zone. Quindi mostra alcune statistiche aggregate per quella distribuzione.



Nella query riportata di seguito, puoi vedere che i valori scelti nell'elenco a discesa delle variabili vengono utilizzati come predicati nella `WHERE` clausola della query, il che consente di concentrarti solo sui dati per la distribuzione. Quindi il pannello traccia le CPU metriche aggregate per le istanze di quella distribuzione. È possibile utilizzare i dati grezzi per eseguire questo drill down con la latenza delle query interattive per ottenere informazioni più approfondite.

```
SELECT bin(time, 5m) as minute,
    ROUND(AVG(cpu_user), 2) AS avg_value,
    ROUND(APPROX_PERCENTILE(cpu_user, 0.9), 2) AS p90_value,
    ROUND(APPROX_PERCENTILE(cpu_user, 0.95), 2) AS p95_value,
```

```
    ROUND(APPROX_PERCENTILE(cpu_user, 0.99), 2) AS p99_value
FROM "raw_data"."devops"
WHERE time BETWEEN from_milliseconds(1636527099476) AND
 from_milliseconds(1636613499476)
    AND region = 'eu-west-1'
    AND cell = 'eu-west-1-cell-10'
    AND silo = 'eu-west-1-cell-10-silo-1'
    AND microservice_name = 'demeter'
    AND availability_zone = 'eu-west-1-3'
    AND measure_name = 'metrics'
GROUP BY bin(time, 5m)
ORDER BY 1
```

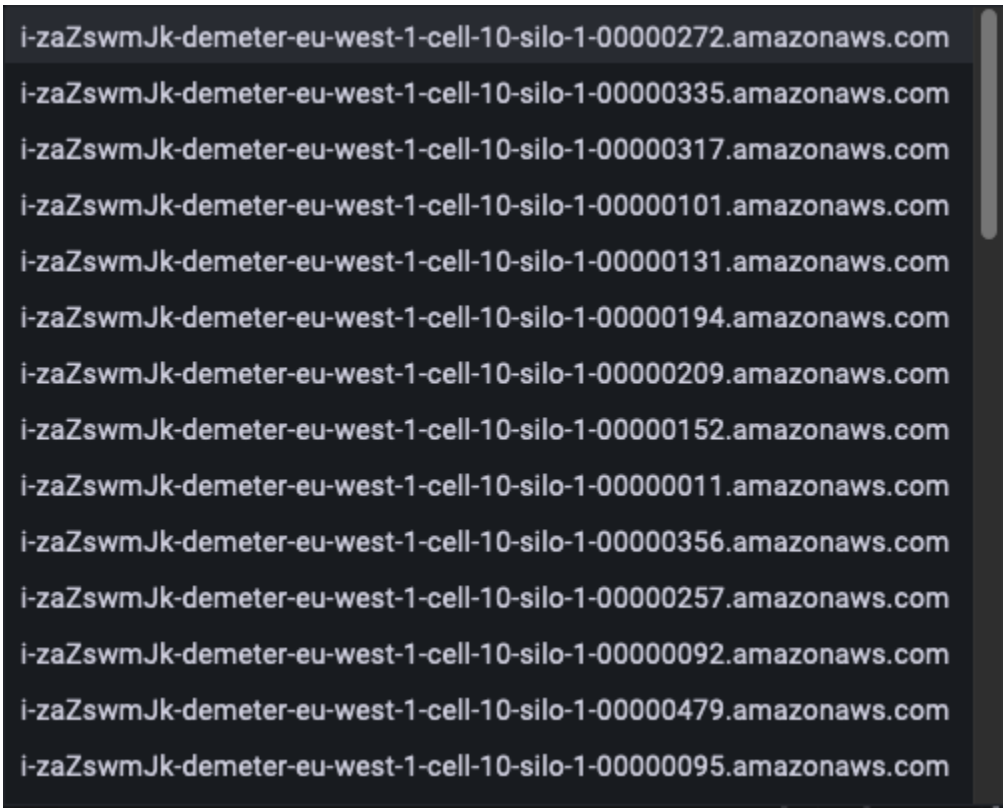
Statistiche a livello di istanza

Questa dashboard calcola ulteriormente un'altra variabile che elenca anche i server/le istanze ad alto CPU utilizzo, ordinati in ordine decrescente di utilizzo. La query utilizzata per calcolare questa variabile viene visualizzata di seguito.

```
WITH microservice_cell_avg AS (
    SELECT AVG(cpu_user) AS microservice_avg_metric
    FROM "raw_data"."devops"
    WHERE $__timeFilter
        AND measure_name = 'metrics'
        AND region = '${region}'
        AND cell = '${cell}'
        AND silo = '${silo}'
        AND availability_zone = '${availability_zone}'
        AND microservice_name = '${microservice}'
), instance_avg AS (
    SELECT instance_name,
        AVG(cpu_user) AS instance_avg_metric
    FROM "raw_data"."devops"
    WHERE $__timeFilter
        AND measure_name = 'metrics'
        AND region = '${region}'
        AND cell = '${cell}'
        AND silo = '${silo}'
        AND microservice_name = '${microservice}'
        AND availability_zone = '${availability_zone}'
    GROUP BY availability_zone, instance_name
)
SELECT i.instance_name
```

```
FROM instance_avg i CROSS JOIN microservice_cell_avg m
WHERE i.instance_avg_metric > (1 + ${utilization_threshold}) *
 m.microservice_avg_metric
ORDER BY i.instance_avg_metric DESC
```

Nella query precedente, la variabile viene ricalcolata dinamicamente in base ai valori scelti per le altre variabili. Una volta compilata la variabile per una distribuzione, puoi selezionare singole istanze dall'elenco per visualizzare ulteriormente le metriche di quell'istanza. Puoi scegliere le diverse istanze dal menu a discesa dei nomi delle istanze, come mostrato nell'istantanea qui sotto.



I pannelli precedenti mostrano le statistiche per l'istanza selezionata e di seguito sono riportate le query utilizzate per recuperare queste statistiche.

```
SELECT BIN(time, 30m) AS time_bin,
    AVG(cpu_user) AS avg_cpu,
    ROUND(APPROX_PERCENTILE(cpu_user, 0.99), 2) as p99_cpu
FROM "raw_data"."devops"
WHERE time BETWEEN from_milliseconds(1636527099477) AND
 from_milliseconds(1636613499477)
    AND measure_name = 'metrics'
    AND region = 'eu-west-1' AND cell = 'eu-west-1-cell-10' AND silo = 'eu-west-1-
cell-10-silo-1'
    AND availability_zone = 'eu-west-1-3' AND microservice_name = 'demeter'
    AND instance_name = 'i-zaZswmJk-demeter-eu-west-1-cell-10-
silo-1-00000272.amazonaws.com'
GROUP BY BIN(time, 30m)
ORDER BY time_bin desc
```

```
SELECT BIN(time, 30m) AS time_bin,
    AVG(memory_used) AS avg_memory,
    ROUND(APPROX_PERCENTILE(memory_used, 0.99), 2) as p99_memory
FROM "raw_data"."devops"
WHERE time BETWEEN from_milliseconds(1636527099477) AND
 from_milliseconds(1636613499477)
    AND measure_name = 'metrics'
```

```
    AND region = 'eu-west-1' AND cell = 'eu-west-1-cell-10' AND silo = 'eu-west-1-
cell-10-silo-1'
    AND availability_zone = 'eu-west-1-3' AND microservice_name = 'demeter'
    AND instance_name = 'i-zaZswmJk-demeter-eu-west-1-cell-10-
silo-1-00000272.amazonaws.com'
GROUP BY BIN(time, 30m)
ORDER BY time_bin desc
```

```
SELECT COUNT(gc_pause)
FROM "raw_data"."devops"
WHERE time BETWEEN from_milliseconds(1636527099477) AND
 from_milliseconds(1636613499478)
    AND measure_name = 'events'
    AND region = 'eu-west-1' AND cell = 'eu-west-1-cell-10' AND silo = 'eu-west-1-
cell-10-silo-1'
    AND availability_zone = 'eu-west-1-3' AND microservice_name = 'demeter'
    AND instance_name = 'i-zaZswmJk-demeter-eu-west-1-cell-10-
silo-1-00000272.amazonaws.com'
```

```
SELECT avg(gc_pause) as avg, round(approx_percentile(gc_pause, 0.99), 2) as p99
FROM "raw_data"."devops"
WHERE time BETWEEN from_milliseconds(1636527099478) AND
 from_milliseconds(1636613499478)
    AND measure_name = 'events'
    AND region = 'eu-west-1' AND cell = 'eu-west-1-cell-10' AND silo = 'eu-west-1-
cell-10-silo-1'
    AND availability_zone = 'eu-west-1-3' AND microservice_name = 'demeter'
    AND instance_name = 'i-zaZswmJk-demeter-eu-west-1-cell-10-
silo-1-00000272.amazonaws.com'
```

```
SELECT BIN(time, 30m) AS time_bin,
    AVG(disk_io_reads) AS avg,
    ROUND(APPROX_PERCENTILE(disk_io_reads, 0.99), 2) as p99
FROM "raw_data"."devops"
WHERE time BETWEEN from_milliseconds(1636527099478) AND
 from_milliseconds(1636613499478)
    AND measure_name = 'metrics'
    AND region = 'eu-west-1' AND cell = 'eu-west-1-cell-10' AND silo = 'eu-west-1-
cell-10-silo-1'
    AND availability_zone = 'eu-west-1-3' AND microservice_name = 'demeter'
    AND instance_name = 'i-zaZswmJk-demeter-eu-west-1-cell-10-
silo-1-00000272.amazonaws.com'
```

```
GROUP BY BIN(time, 30m)
ORDER BY time_bin desc
```

Ottimizzazione dei costi condividendo le query pianificate tra i dashboard

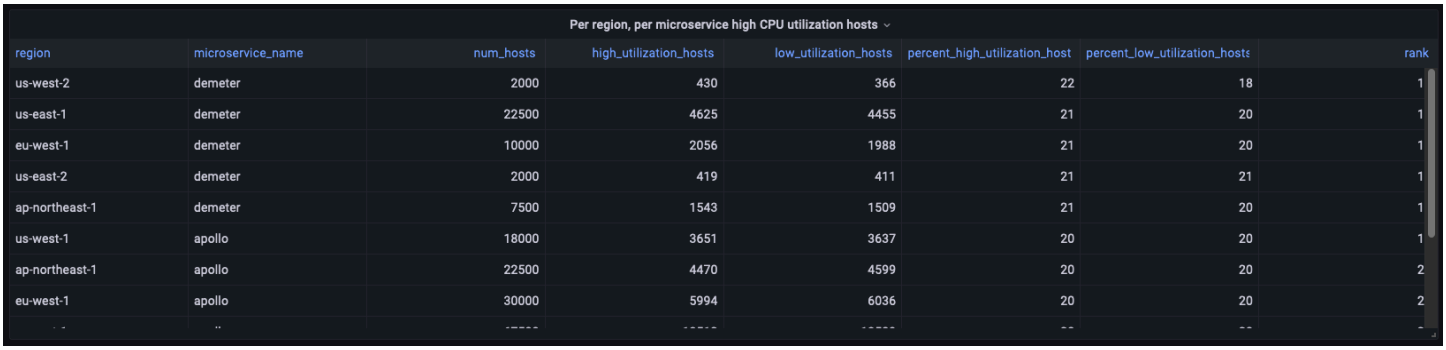
In questo esempio, vedremo uno scenario in cui più pannelli di dashboard visualizzano variazioni di informazioni simili (individuazione di CPU host e frazioni di flotta elevate con un elevato CPU utilizzo) e come è possibile utilizzare la stessa query pianificata per precalcolare i risultati, che vengono poi utilizzati per popolare più pannelli. Questo riutilizzo ottimizza ulteriormente i costi laddove invece di utilizzare diverse query pianificate, una per ogni pannello, si utilizza only owner.

Pannelli di controllo con dati grezzi

CPUutilizzo per regione per microservizio

Il primo pannello calcola le istanze il cui utilizzo medio è una soglia inferiore o superiore all'CPUutilizzo precedente per una determinata distribuzione all'interno di una regione, una cella, un silo, una zona di disponibilità e un microservizio. CPU Quindi ordina la regione e il microservizio con la più alta percentuale di host con un elevato utilizzo. Consente di identificare la temperatura di funzionamento dei server di una specifica implementazione e successivamente approfondisce i problemi per comprendere meglio i problemi.

L'interrogazione per il pannello dimostra la flessibilità del SQL supporto di Timestream for per eseguire attività analitiche complesse con espressioni di tabella comuni, funzioni di finestra, join e LiveAnalytics così via.



Per region, per microservice high CPU utilization hosts

region	microservice_name	num_hosts	high_utilization_hosts	low_utilization_hosts	percent_high_utilization_host	percent_low_utilization_hosts	rank
us-west-2	demeter	2000	430	366	22	18	1
us-east-1	demeter	22500	4625	4455	21	20	1
eu-west-1	demeter	10000	2056	1988	21	20	1
us-east-2	demeter	2000	419	411	21	21	1
ap-northeast-1	demeter	7500	1543	1509	21	20	1
us-west-1	apollo	18000	3651	3637	20	20	1
ap-northeast-1	apollo	22500	4470	4599	20	20	2
eu-west-1	apollo	30000	5994	6036	20	20	2

Interrogazione:

```
WITH microservice_cell_avg AS (
    SELECT region, cell, silo, availability_zone, microservice_name, AVG(cpu_user) AS
 microservice_avg_metric
    FROM "raw_data"."devops"
```

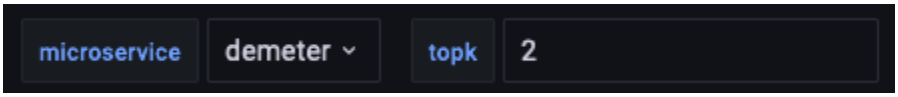
```
    WHERE time BETWEEN from_milliseconds(1636526593876) AND
 from_milliseconds(1636612993876)
        AND measure_name = 'metrics'
    GROUP BY region, cell, silo, availability_zone, microservice_name
), instance_avg AS (
    SELECT region, cell, silo, availability_zone, microservice_name, instance_name,
        AVG(cpu_user) AS instance_avg_metric
    FROM "raw_data"."devops"
    WHERE time BETWEEN from_milliseconds(1636526593876) AND
 from_milliseconds(1636612993876)
        AND measure_name = 'metrics'
    GROUP BY region, cell, silo, availability_zone, microservice_name, instance_name
), instances_above_threshold AS (
  SELECT i.*,
    CASE WHEN i.instance_avg_metric > (1 + 0.2) * m.microservice_avg_metric THEN 1 ELSE
 0 END AS high_utilization,
    CASE WHEN i.instance_avg_metric < (1 - 0.2) * m.microservice_avg_metric THEN 1 ELSE
 0 END AS low_utilization
  FROM instance_avg i INNER JOIN microservice_cell_avg m
    ON i.region = m.region AND i.cell = m.cell AND i.silo = m.silo AND
 i.availability_zone = m.availability_zone
      AND m.microservice_name = i.microservice_name
), per_deployment_high AS (
SELECT region, microservice_name, COUNT(*) AS num_hosts, SUM(high_utilization) AS
 high_utilization_hosts, SUM(low_utilization) AS low_utilization_hosts,
    ROUND(SUM(high_utilization) * 100.0 / COUNT(*), 0) AS
 percent_high_utilization_hosts,
    ROUND(SUM(low_utilization) * 100.0 / COUNT(*), 0) AS percent_low_utilization_hosts
FROM instances_above_threshold
GROUP BY region, microservice_name
), per_region_ranked AS (
    SELECT *,
        DENSE_RANK() OVER (PARTITION BY region ORDER BY percent_high_utilization_hosts
 DESC, high_utilization_hosts DESC) AS rank
    FROM per_deployment_high
)
SELECT *
FROM per_region_ranked
WHERE rank <= 2
ORDER BY percent_high_utilization_hosts desc, rank asc
```

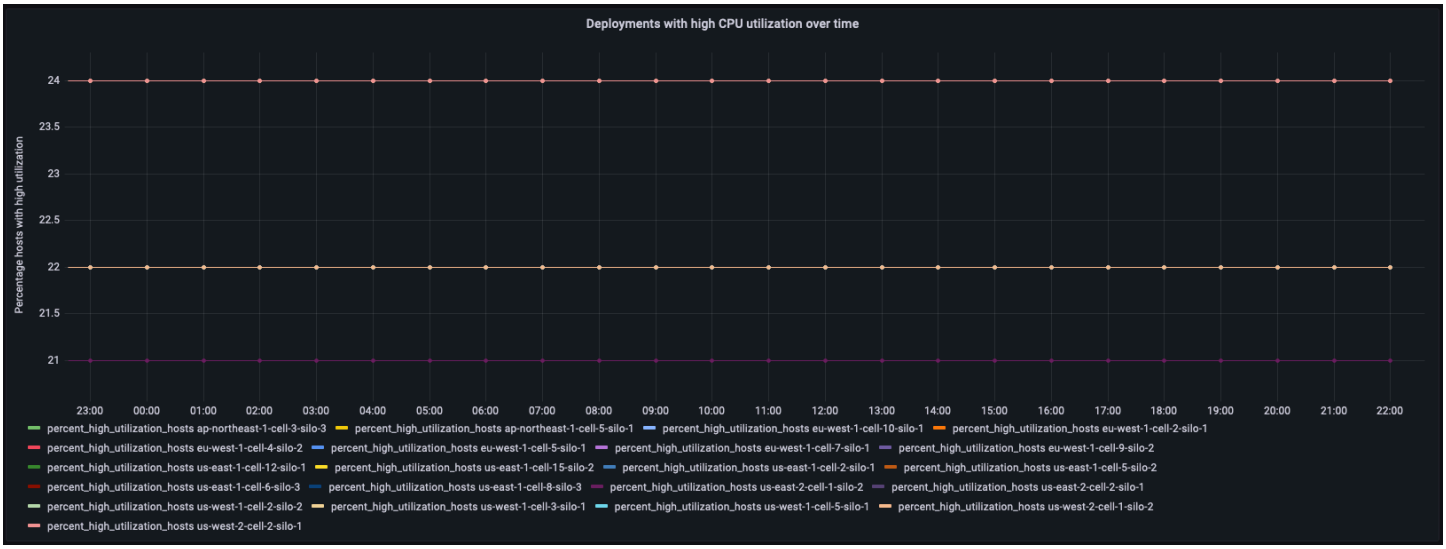
Approfondisci un microservizio per trovare gli hot spot

La dashboard successiva consente di approfondire uno dei microservizi per scoprire la regione, la cella e il silo specifici in cui tale microservizio è in esecuzione, quale frazione della sua flotta è maggiormente utilizzata. CPU Ad esempio, nella dashboard a livello di flotta, hai visto il microservizio demeter apparire nelle prime posizioni in classifica, quindi in questa dashboard vuoi approfondire quel microservizio.

Questa dashboard utilizza una variabile per selezionare il microservizio da approfondire e i valori della variabile vengono compilati utilizzando valori univoci della dimensione. Una volta scelto il microservizio, il resto della dashboard si aggiorna.

Come illustrato di seguito, il primo pannello riporta la percentuale di host in una distribuzione (una regione, una cella e un silo per un microservizio) nel tempo e la query corrispondente utilizzata per tracciare la dashboard. Questo grafico stesso identifica una distribuzione specifica con una percentuale più elevata di host con un livello elevato. CPU





Interrogazione:

```
WITH microservice_cell_avg AS (
    SELECT region, cell, silo, availability_zone, microservice_name, bin(time, 1h) as
 hour, AVG(cpu_user) AS microservice_avg_metric
    FROM "raw_data"."devops"
    WHERE time BETWEEN from_milliseconds(1636526898831) AND
 from_milliseconds(1636613298831)
        AND measure_name = 'metrics'
        AND microservice_name = 'demeter'
```

```
    GROUP BY region, cell, silo, availability_zone, microservice_name, bin(time, 1h)
), instance_avg AS (
    SELECT region, cell, silo, availability_zone, microservice_name, instance_name,
 bin(time, 1h) as hour,
        AVG(cpu_user) AS instance_avg_metric
    FROM "raw_data"."devops"
    WHERE time BETWEEN from_milliseconds(1636526898831) AND
 from_milliseconds(1636613298831)
        AND measure_name = 'metrics'
        AND microservice_name = 'demeter'
    GROUP BY region, cell, silo, availability_zone, microservice_name, instance_name,
 bin(time, 1h)
), instances_above_threshold AS (
  SELECT i.*,
    CASE WHEN i.instance_avg_metric > (1 + 0.2) * m.microservice_avg_metric THEN 1 ELSE
 0 END AS high_utilization
  FROM instance_avg i INNER JOIN microservice_cell_avg m
    ON i.region = m.region AND i.cell = m.cell AND i.silo = m.silo AND
 i.availability_zone = m.availability_zone
      AND m.microservice_name = i.microservice_name AND m.hour = i.hour
), high_utilization_percent AS (
    SELECT region, cell, silo, microservice_name, hour, COUNT(*) AS num_hosts,
 SUM(high_utilization) AS high_utilization_hosts,
        ROUND(SUM(high_utilization) * 100.0 / COUNT(*), 0) AS
 percent_high_utilization_hosts
    FROM instances_above_threshold
    GROUP BY region, cell, silo, microservice_name, hour
), high_utilization_ranked AS (
    SELECT region, cell, silo, microservice_name,
        DENSE_RANK() OVER (PARTITION BY region ORDER BY
 AVG(percent_high_utilization_hosts) desc, AVG(high_utilization_hosts) desc) AS rank
    FROM high_utilization_percent
    GROUP BY region, cell, silo, microservice_name
)
SELECT hup.silo, CREATE_TIME_SERIES(hour, hup.percent_high_utilization_hosts) AS
 percent_high_utilization_hosts
FROM high_utilization_percent hup INNER JOIN high_utilization_ranked hur
    ON hup.region = hur.region AND hup.cell = hur.cell AND hup.silo = hur.silo AND
 hup.microservice_name = hur.microservice_name
WHERE rank <= 2
GROUP BY hup.region, hup.cell, hup.silo
ORDER BY hup.silo
```

Conversione in un'unica query pianificata che ne consente il riutilizzo

È importante notare che un calcolo simile viene eseguito nei diversi pannelli delle due dashboard. È possibile definire un'interrogazione pianificata separata per ogni pannello. Qui vedrete come ottimizzare ulteriormente i costi definendo un'interrogazione pianificata i cui risultati possono essere utilizzati per il rendering di tutti e tre i pannelli.

Di seguito è riportata la query che acquisisce gli aggregati calcolati e utilizzati per tutti i diversi pannelli. Osserverete diversi aspetti importanti nella definizione di questa interrogazione pianificata.

- La flessibilità e la potenza dell'area di SQL superficie supportate dalle query pianificate, in cui è possibile utilizzare espressioni di tabella comuni, join, case statement, ecc.
- È possibile utilizzare una query pianificata per calcolare le statistiche con una granularità più precisa di quella necessaria per un dashboard specifico e per tutti i valori che un dashboard potrebbe utilizzare per diverse variabili. Ad esempio, vedrai che gli aggregati vengono calcolati su una regione, una cella, un silo e un microservizio. Pertanto, puoi combinarli per creare aggregati a livello di regione o regione e a livello di microservizi. Analogamente, la stessa query calcola gli aggregati per tutte le regioni, le celle, i silos e i microservizi. Consente di applicare filtri su queste colonne per ottenere gli aggregati per un sottoinsieme di valori. Ad esempio, puoi calcolare gli aggregati per qualsiasi regione, ad esempio us-east-1, o qualsiasi microservizio, ad esempio demeter, o approfondire un'implementazione specifica all'interno di una regione, cella, silo e microservizio. Questo approccio ottimizza ulteriormente i costi di manutenzione degli aggregati precalcolati.

```
WITH microservice_cell_avg AS (
    SELECT region, cell, silo, availability_zone, microservice_name, bin(time, 1h) as
 hour, AVG(cpu_user) AS microservice_avg_metric
    FROM raw_data.devops
    WHERE time BETWEEN bin(@scheduled_runtime, 1h) - 1h AND bin(@scheduled_runtime, 1h)
 + 1h
        AND measure_name = 'metrics'
    GROUP BY region, cell, silo, availability_zone, microservice_name, bin(time, 1h)

), instance_avg AS (
    SELECT region, cell, silo, availability_zone, microservice_name, instance_name,
 bin(time, 1h) as hour,
        AVG(cpu_user) AS instance_avg_metric
   FROM raw_data.devops
```

```
   WHERE time BETWEEN bin(@scheduled_runtime, 1h) - 1h AND bin(@scheduled_runtime, 1h)
 + 1h
       AND measure_name = 'metrics'
   GROUP BY region, cell, silo, availability_zone, microservice_name, instance_name,
 bin(time, 1h)
), instances_above_threshold AS (
    SELECT i.*,
        CASE WHEN i.instance_avg_metric > (1 + 0.2) * m.microservice_avg_metric THEN 1
 ELSE 0 END AS high_utilization,
        CASE WHEN i.instance_avg_metric < (1 - 0.2) * m.microservice_avg_metric THEN 1
 ELSE 0 END AS low_utilization
   FROM instance_avg i INNER JOIN microservice_cell_avg m
       ON i.region = m.region AND i.cell = m.cell AND i.silo = m.silo AND
 i.availability_zone = m.availability_zone
          AND m.microservice_name = i.microservice_name AND m.hour = i.hour
)
SELECT region, cell, silo, microservice_name, hour,
     COUNT(*) AS num_hosts, SUM(high_utilization) AS high_utilization_hosts,
 SUM(low_utilization) AS low_utilization_hosts
FROM instances_above_threshold GROUP BY region, cell, silo, microservice_name, hour
```

Di seguito è riportata una definizione di interrogazione pianificata per la query precedente. L'espressione di pianificazione è configurata per l'aggiornamento ogni 30 minuti e aggiorna i dati fino a un'ora precedente, utilizzando sempre il costrutto bin (@scheduled_runtime, 1h) per ottenere gli eventi dell'intera ora. A seconda dei requisiti di aggiornamento dell'applicazione, è possibile configurarla in modo che si aggiorni più o meno frequentemente. Utilizzando WHERE time BETWEEN bin (@scheduled_runtime, 1h) - 1h AND bin (@scheduled_runtime, 1h) + 1h, possiamo garantire che, anche se esegui l'aggiornamento una volta ogni 15 minuti, otterrai i dati dell'intera ora per l'ora corrente e l'ora precedente.

Più avanti, vedrai come i tre pannelli utilizzano questi aggregati scritti nella tabella deployment_cpu_stats_per_hr per visualizzare le metriche rilevanti per il pannello.

```
{
    "Name": "MultiPT30mHighCpuDeploymentsPerHr",
    "QueryString": "WITH microservice_cell_avg AS (    SELECT region, cell,
 silo, availability_zone, microservice_name, bin(time, 1h) as hour, AVG(cpu_user)
 AS microservice_avg_metric    FROM raw_data.devops    WHERE time BETWEEN
 bin(@scheduled_runtime, 1h) - 1h AND bin(@scheduled_runtime, 1h) + 1h    AND
 measure_name = 'metrics'    GROUP BY region, cell, silo, availability_zone,
 microservice_name, bin(time, 1h)    ), instance_avg AS (    SELECT region,
 cell, silo, availability_zone, microservice_name, instance_name, bin(time, 1h)
```

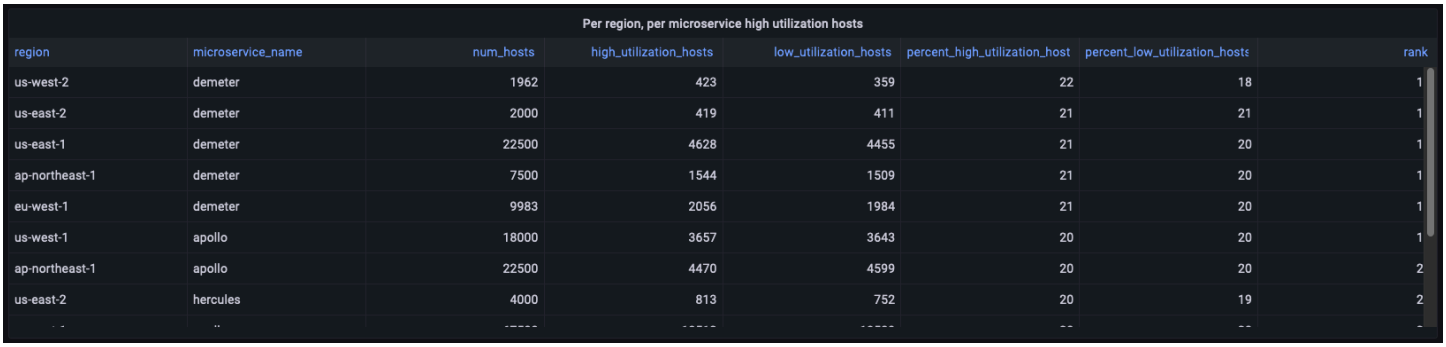
```
as hour,    AVG(cpu_user) AS instance_avg_metric    FROM raw_data.devops
WHERE time BETWEEN bin(@scheduled_runtime, 1h) - 1h AND bin(@scheduled_runtime,
1h) + 1h    AND measure_name = 'metrics'    GROUP BY region, cell, silo,
availability_zone, microservice_name, instance_name, bin(time, 1h)    ),
instances_above_threshold AS (    SELECT i.*,    CASE WHEN i.instance_avg_metric >
(1 + 0.2) * m.microservice_avg_metric THEN 1 ELSE 0 END AS high_utilization,    CASE
WHEN i.instance_avg_metric < (1 - 0.2) * m.microservice_avg_metric THEN 1 ELSE 0 END
AS low_utilization    FROM instance_avg i INNER JOIN microservice_cell_avg m    ON
i.region = m.region AND i.cell = m.cell AND i.silo = m.silo AND i.availability_zone
= m.availability_zone    AND m.microservice_name = i.microservice_name AND m.hour =
i.hour    )     SELECT region, cell, silo, microservice_name, hour,         COUNT(*)
AS num_hosts, SUM(high_utilization) AS high_utilization_hosts, SUM(low_utilization) AS
low_utilization_hosts    FROM instances_above_threshold GROUP BY region, cell, silo,
microservice_name, hour",
   "ScheduleConfiguration": {
       "ScheduleExpression": "cron(0/30 * * * ? *)"
   },
   "NotificationConfiguration": {
       "SnsConfiguration": {
           "TopicArn": "******"
       }
   },
   "TargetConfiguration": {
       "TimestreamConfiguration": {
           "DatabaseName": "derived",
           "TableName": "deployment_cpu_stats_per_hr",
           "TimeColumn": "hour",
           "DimensionMappings": [
               {
                   "Name": "region",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "cell",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "silo",
                   "DimensionValueType": "VARCHAR"
               },
               {
                   "Name": "microservice_name",
                   "DimensionValueType": "VARCHAR"
               }
```

```
            ],
            "MultiMeasureMappings": {
                "TargetMultiMeasureName": "cpu_user",
                "MultiMeasureAttributeMappings": [
                    {
                        "SourceColumn": "num_hosts",
                        "MeasureValueType": "BIGINT"
                    },
                    {
                        "SourceColumn": "high_utilization_hosts",
                        "MeasureValueType": "BIGINT"
                    },
                    {
                        "SourceColumn": "low_utilization_hosts",
                        "MeasureValueType": "BIGINT"
                    }
                ]
            }
        }
    },
    "ErrorReportConfiguration": {
        "S3Configuration" : {
            "BucketName" : "******",
            "ObjectKeyPrefix": "errors",
            "EncryptionOption": "SSE_S3"
        }
    },
    "ScheduledQueryExecutionRoleArn": "******"
}
```

Dashboard con risultati precalcolati

Host ad alto utilizzo CPU

Per gli host ad alto utilizzo, vedrai come i diversi pannelli utilizzano i dati di deployment_cpu_stats_per_hr per calcolare i diversi aggregati necessari per i pannelli. Ad esempio, questo pannello fornisce informazioni a livello di regione, quindi riporta gli aggregati raggruppati per regione e microservizio, senza filtrare alcuna regione o microservizio.



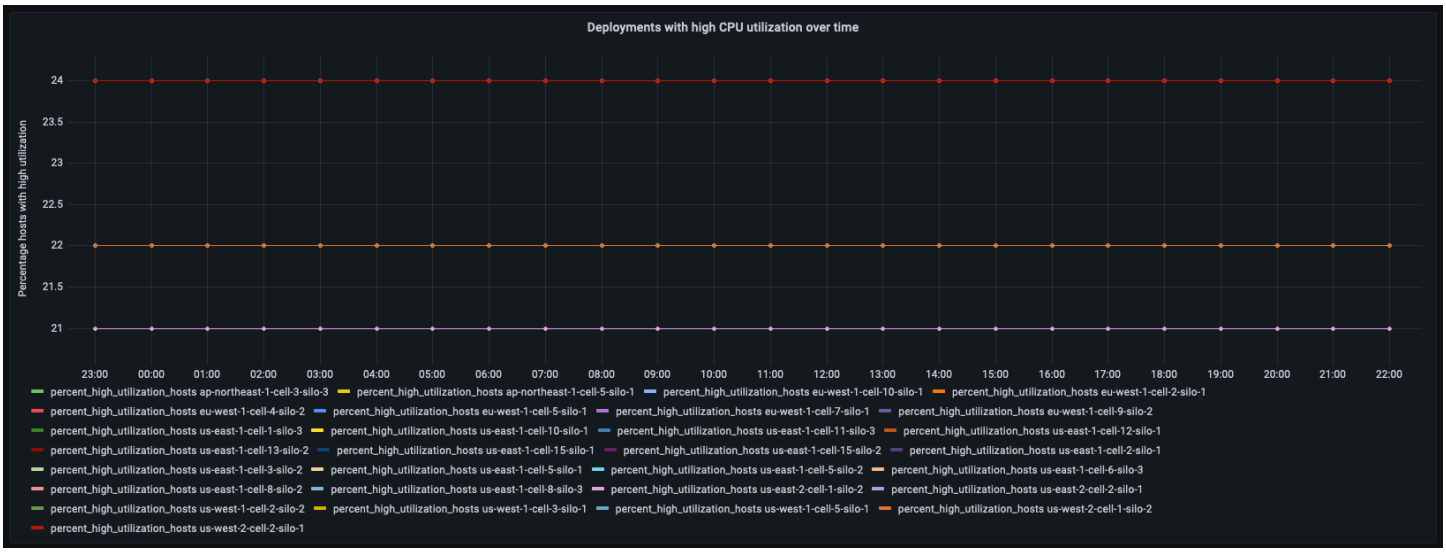
Per region, per microservice high utilization hosts

region	microservice_name	num_hosts	high_utilization_hosts	low_utilization_hosts	percent_high_utilization_host	percent_low_utilization_hosts	rank
us-west-2	demeter	1962	423	359	22	18	1
us-east-2	demeter	2000	419	411	21	21	1
us-east-1	demeter	22500	4628	4455	21	20	1
ap-northeast-1	demeter	7500	1544	1509	21	20	1
eu-west-1	demeter	9983	2056	1984	21	20	1
us-west-1	apollo	18000	3657	3643	20	20	1
ap-northeast-1	apollo	22500	4470	4599	20	20	2
us-east-2	hercules	4000	813	752	20	19	2

```
WITH per_deployment_hosts AS (
    SELECT region, cell, silo, microservice_name,
        AVG(num_hosts) AS num_hosts,
        AVG(high_utilization_hosts) AS high_utilization_hosts,
        AVG(low_utilization_hosts) AS low_utilization_hosts
    FROM "derived"."deployment_cpu_stats_per_hr"
    WHERE time BETWEEN from_milliseconds(1636567785437) AND
 from_milliseconds(1636654185437)
        AND measure_name = 'cpu_user'
    GROUP BY region, cell, silo, microservice_name
), per_deployment_high AS (
    SELECT region, microservice_name,
        SUM(num_hosts) AS num_hosts,
        ROUND(SUM(high_utilization_hosts), 0) AS high_utilization_hosts,
        ROUND(SUM(low_utilization_hosts),0) AS low_utilization_hosts,
        ROUND(SUM(high_utilization_hosts) * 100.0 / SUM(num_hosts)) AS
 percent_high_utilization_hosts,
        ROUND(SUM(low_utilization_hosts) * 100.0 / SUM(num_hosts)) AS
 percent_low_utilization_hosts
    FROM per_deployment_hosts
    GROUP BY region, microservice_name
),
per_region_ranked AS (
    SELECT *,
        DENSE_RANK() OVER (PARTITION BY region ORDER BY percent_high_utilization_hosts
 DESC, high_utilization_hosts DESC) AS rank
    FROM per_deployment_high
)
SELECT *
FROM per_region_ranked
WHERE rank <= 2
ORDER BY percent_high_utilization_hosts desc, rank asc
```

Approfondisci un microservizio per trovare implementazioni ad alto utilizzo CPU

Il prossimo esempio utilizza nuovamente la tabella derivata deployment_cpu_stats_per_hr, ma ora applica un filtro per un microservizio specifico (demeter in questo esempio, poiché riportava host ad alto utilizzo nella dashboard aggregata). Questo pannello tiene traccia della percentuale CPU di host ad alto utilizzo nel tempo.




```
WITH high_utilization_percent AS (
    SELECT region, cell, silo, microservice_name, bin(time, 1h) AS hour, MAX(num_hosts)
 AS num_hosts,
        MAX(high_utilization_hosts) AS high_utilization_hosts,
        ROUND(MAX(high_utilization_hosts) * 100.0 / MAX(num_hosts)) AS
 percent_high_utilization_hosts
    FROM "derived"."deployment_cpu_stats_per_hr"
    WHERE time BETWEEN from_milliseconds(1636525800000) AND
 from_milliseconds(1636612200000)
        AND measure_name = 'cpu_user'
        AND microservice_name = 'demeter'
    GROUP BY region, cell, silo, microservice_name, bin(time, 1h)
), high_utilization_ranked AS (
    SELECT region, cell, silo, microservice_name,
        DENSE_RANK() OVER (PARTITION BY region ORDER BY
 AVG(percent_high_utilization_hosts) desc, AVG(high_utilization_hosts) desc) AS rank
    FROM high_utilization_percent
    GROUP BY region, cell, silo, microservice_name
)
SELECT hup.silo, CREATE_TIME_SERIES(hour, hup.percent_high_utilization_hosts) AS
 percent_high_utilization_hosts
FROM high_utilization_percent hup INNER JOIN high_utilization_ranked hur
```

```
    ON hup.region = hur.region AND hup.cell = hur.cell AND hup.silo = hur.silo AND
 hup.microservice_name = hur.microservice_name
WHERE rank <= 2
GROUP BY hup.region, hup.cell, hup.silo
ORDER BY hup.silo
```

Confronto di un'interrogazione su una tabella di base con un'interrogazione dei risultati di una query pianificata

In questo esempio di query Timestream, utilizziamo lo schema, le query di esempio e gli output seguenti per confrontare una query su una tabella di base con una query su una tabella derivata di risultati di query pianificate. Con una query pianificata correttamente, è possibile ottenere una tabella derivata con un minor numero di righe e altre caratteristiche che possono portare a query più veloci di quanto sarebbe possibile con la tabella base originale.

Per un video che descrive questo scenario, consulta Migliorare le prestazioni delle query e ridurre i costi utilizzando le query pianificate in Amazon Timestream for. LiveAnalytics

Per questo esempio, utilizziamo lo scenario seguente:

- Regione: us-east-1
- Tavolo base — `"clickstream"."shopping"`
- Tabella derivata — `"clickstream"."aggregate"`

Tabella di base

Di seguito viene descritto lo schema per la tabella di base.

Colonna	Tipo	Timestream per LiveAnalytics il tipo di attributo
canale	varchar	MULTI
description	varchar	MULTI
evento	varchar	DIMENSION
ip_address	varchar	DIMENSION

Colonna	Tipo	Timestream per LiveAnalytics il tipo di attributo
measure_name	varchar	MEASURE_NAME
prodotto	varchar	MULTI
product_id	varchar	MULTI
quantity	double	MULTI
query	varchar	MULTI
session_id	varchar	DIMENSION
user_group	varchar	DIMENSION
user_id	varchar	DIMENSION

Di seguito vengono descritte le misure per la tabella base. Una tabella di base si riferisce a una tabella in Timestream su cui viene eseguita la query pianificata.

- nome_misura — `metrics`
- data — multiplo
- dimensioni:

```
[ ( user_group, varchar ),( user_id, varchar ),( session_id, varchar ),( ip_address,
 varchar ),( event, varchar ) ]
```

Interrogazione su una tabella di base

Di seguito è riportata una query ad hoc che raccoglie i conteggi in base a un aggregato di 5 minuti in un determinato intervallo di tempo.

```
SELECT BIN(time, 5m) as time,
channel,
product_id,
SUM(quantity) as product_quantity
FROM "clickstream"."shopping"
```

```
WHERE BIN(time, 5m) BETWEEN '2023-05-11 10:10:00.000000000' AND '2023-05-11
 10:30:00.000000000'
AND channel = 'Social media'
and product_id = '431412'
GROUP BY BIN(time, 5m),channel,product_id
```

Output:

```
duration:1.745 sec
Bytes scanned: 29.89 MB
Query Id: AEBQEANMHG7MHHBHCKJ3BSOE3QUGIDBGWCCP5I6J6YUW5CVJZ2M3JCJ27QRMM7A
Row count:5
```

Query pianificata

Di seguito è riportata una query pianificata che viene eseguita ogni 5 minuti.

```
SELECT BIN(time, 5m) as time, channel as measure_name, product_id, product,
SUM(quantity) as product_quantity
FROM "clickstream"."shopping"
WHERE time BETWEEN BIN(@scheduled_runtime, 5m) - 10m AND BIN(@scheduled_runtime, 5m) -
 5m
AND channel = 'Social media'
GROUP BY BIN(time, 5m), channel, product_id, product
```

Interrogazione su una tabella derivata

Di seguito è riportata una query ad hoc su una tabella derivata. Una tabella derivata si riferisce a una tabella Timestream che contiene i risultati di una query pianificata.

```
SELECT time, measure_name, product_id,product_quantity
FROM "clickstream"."aggregate"
WHERE time BETWEEN '2023-05-11 10:10:00.000000000' AND '2023-05-11 10:30:00.000000000'
AND measure_name = 'Social media'
and product_id = '431412'
```

Output:

```
duration: 0.2960 sec
Bytes scanned: 235.00 B
QueryID: AEBQEANMHHAAQU4FFTT6CFM6UYXTL4SMLZV22MFP4KV2Z7IRVOPLOMLDD6BR33Q
```

```
Row count: 5
```

Confronto

Di seguito è riportato un confronto tra i risultati di una query su una tabella di base e di un'interrogazione su una tabella derivata. La stessa query su una tabella derivata con risultati aggregati eseguiti tramite una query pianificata viene completata più rapidamente con un minor numero di byte scansionati.

Questi risultati mostrano l'utilità dell'utilizzo di query pianificate per aggregare i dati per ottenere query più rapide.

	Interrogazione sulla tabella di base	Interrogazione sulla tabella derivata
Durata	1.745 sec	0,2960 sec
Byte scansionati	29,89 MB	235 byte
Numero di righe	5	5

Utilizzo UNLOAD per esportare i risultati delle query in S3 da Timestream per LiveAnalytics

Amazon Timestream LiveAnalytics per ora ti consente di esportare i risultati delle tue query su Amazon S3 in modo economico e sicuro utilizzando l'istruzione. `UNLOAD` Utilizzando l'`UNLOAD`istruzione, ora puoi esportare i dati delle serie temporali in bucket S3 selezionati in formato Apache Parquet o Comma Separated Values (CSV), il che offre la flessibilità necessaria per archiviare, combinare e analizzare i dati delle serie temporali con altri servizi. L'`UNLOAD`istruzione consente di esportare i dati in modo compresso, il che riduce i dati trasferiti e lo spazio di archiviazione richiesto. `UNLOAD`supporta anche il partizionamento basato su attributi selezionati durante l'esportazione dei dati, migliorando le prestazioni e riducendo i tempi di elaborazione dei servizi a valle che accedono ai dati. Inoltre, puoi utilizzare le chiavi gestite di Amazon S3 (SSE-S3) o AWS le chiavi gestite di Amazon S3 (SSE-AWS KMSKMS) per crittografare i dati esportati.

I vantaggi di Timestream per UNLOAD LiveAnalytics

I principali vantaggi dell'utilizzo della `UNLOAD` dichiarazione sono i seguenti.

- Facilità operativa: con l'UNLOADistruzione, è possibile esportare gigabyte di dati in una singola richiesta di query in Apache Parquet o in CSV formato, offrendo la flessibilità necessaria per selezionare il formato più adatto alle esigenze di elaborazione a valle e semplificando la creazione di data lake.
- Sicuro ed economico: UNLOAD statement offre la possibilità di esportare i dati in un bucket S3 in modo compresso e di crittografare (SSE- KMS o SSE _S3) i dati utilizzando chiavi gestite dal cliente, riducendo i costi di archiviazione dei dati e proteggendo dall'accesso non autorizzato.
- Prestazioni: utilizzando l'UNLOADistruzione, puoi partizionare i dati durante l'esportazione in un bucket S3. Il partizionamento dei dati consente ai servizi a valle di elaborare i dati in parallelo, riducendone i tempi di elaborazione. Inoltre, i servizi a valle possono elaborare solo i dati di cui hanno bisogno, riducendo le risorse di elaborazione necessarie e quindi i costi associati.

Casi d'uso di UNLOAD from Timestream per LiveAnalytics

Puoi utilizzare l'UNLOADistruzione per scrivere dati nel tuo bucket S3 nei seguenti modi.

- Crea un data warehouse: puoi esportare gigabyte di risultati delle query nel bucket S3 e aggiungere più facilmente dati di serie temporali al tuo data lake. Puoi utilizzare servizi come Amazon Athena e Amazon Redshift per combinare i dati delle tue serie temporali con altri dati pertinenti per ricavare informazioni aziendali complesse.
- Crea pipeline di dati AI e ML: la UNLOAD dichiarazione ti consente di creare facilmente pipeline di dati per i tuoi modelli di machine learning che accedono ai dati delle serie temporali, semplificando l'utilizzo dei dati delle serie temporali con servizi come Amazon e SageMaker Amazon. EMR
- Semplifica ETL l'elaborazione: l'esportazione dei dati in bucket S3 può semplificare il processo di esecuzione delle operazioni Extract, Transform, Load (ETL) sui dati, consentendoti di utilizzare senza problemi strumenti o servizi di terze parti AWS come AWS Glue per elaborare e trasformare i dati.

UNLOADConcetti

Sintassi

```
UNLOAD (SELECT statement)
 TO 's3://bucket-name/folder'
 WITH ( option = expression [, ...] )
```

optiondov'è

```
{ partitioned_by = ARRAY[ col_name[,…] ]
 | format = [ '{ CSV | PARQUET }' ]
 | compression = [ '{ GZIP | NONE }' ]
 | encryption = [ '{ SSE_KMS | SSE_S3 }' ]
 | kms_key = '<string>'
 | field_delimiter ='<character>'
 | escaped_by = '<character>'
 | include_header = ['{true, false}']
 | max_file_size = '<value>'
 | }
```

Parametri

SELECTdichiarazione

L'istruzione di query utilizzata per selezionare e recuperare i dati da uno o più Timestream per le tabelle. LiveAnalytics

```
(SELECT column 1, column 2, column 3 from database.table
      where measure_name = "ABC" and timestamp between ago (1d) and now() )
```

Clausola TO

```
TO 's3://bucket-name/folder'
```

oppure

```
TO 's3://access-point-alias/folder'
```

La T0 clausola dell'UNL0ADistruzione specifica la destinazione per l'output dei risultati della query. È necessario fornire il percorso completo, incluso il nome del bucket Amazon S3 o Amazon S3 con access-point-alias posizione della cartella su Amazon S3 dove Timestream for scrive gli oggetti del file di output. LiveAnalytics Il bucket S3 deve appartenere allo stesso account e nella stessa regione. Oltre al set di risultati della query, Timestream for LiveAnalytics scrive i file manifest e di metadati nella cartella di destinazione specificata.

PARTITIONEDClausola _BY

```
partitioned_by = ARRAY [col_name[,…] , (default: none)
```

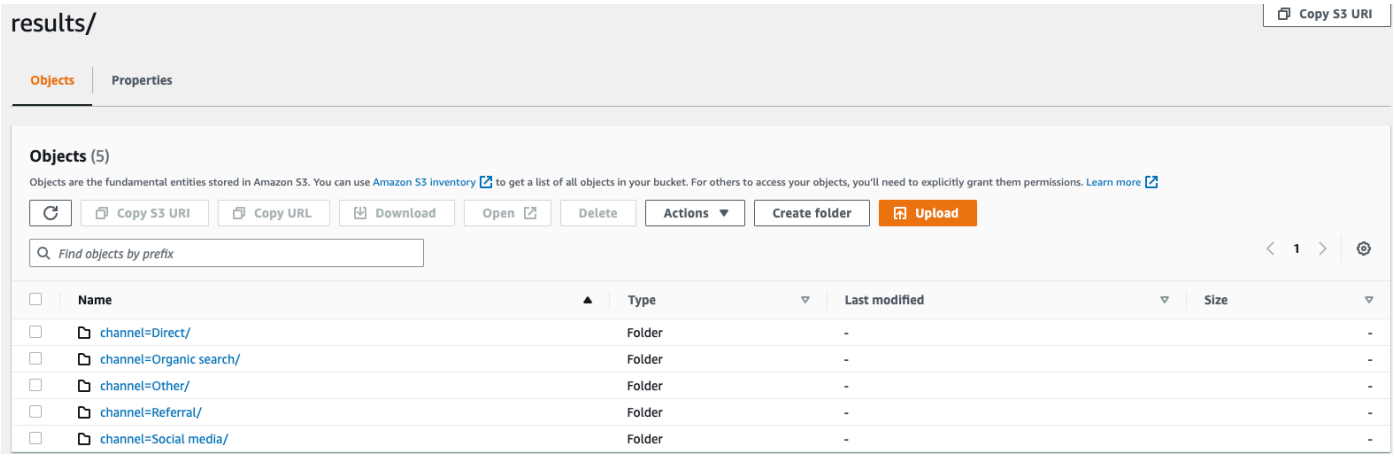
La `partitioned_by` clausola viene utilizzata nelle query per raggruppare e analizzare i dati a livello granulare. Quando esporti i risultati della query nel bucket S3, puoi scegliere di partizionare i dati in base a una o più colonne nella query di selezione. Durante il partizionamento dei dati, i dati esportati vengono suddivisi in sottoinsiemi in base alla colonna della partizione e ogni sottoinsieme viene archiviato in una cartella separata. All'interno della cartella dei risultati che contiene i dati esportati, viene creata automaticamente una sottocartella. `folder/results/partition column = partition value/` Tuttavia, tieni presente che le colonne partizionate non sono incluse nel file di output.

`partitioned_by`non è una clausola obbligatoria nella sintassi. Se si sceglie di esportare i dati senza alcun partizionamento, è possibile escludere la clausola nella sintassi.

Example

Supponendo che tu stia monitorando i dati clickstream del tuo sito Web e che tu abbia 5 canali di traffico, vale a dire,, e. `direct` `Social Media` `Organic Search` `Other` `Referral` Quando si esportano i dati, è possibile scegliere di partizionarli utilizzando la colonna. `Channel` All'interno della tua cartella dati`s3://bucketname/results`, avrai cinque cartelle ciascuna con il rispettivo nome del canale, ad esempio, `s3://bucketname/results/channel=Social Media/.` all'interno di questa cartella troverai i dati di tutti i clienti che sono arrivati sul tuo sito web attraverso il `Social Media` canale. Allo stesso modo, avrai altre cartelle per i canali rimanenti.

Dati esportati partizionati per colonna Channel



FORMAT

```
format = [ '{ CSV | PARQUET }' , default: CSV
```

Le parole chiave per specificare il formato dei risultati della query scritti nel bucket S3. È possibile esportare i dati come valore separato da virgole (CSV) utilizzando una virgola (,) come delimitatore predefinito o nel formato Apache Parquet, un efficiente formato di archiviazione a colonne aperto per l'analisi.

COMPRESSION

```
compression = [ '{ GZIP | NONE }' ], default: GZIP
```

È possibile comprimere i dati esportati utilizzando l'algoritmo di compressione GZIP o decomprimerli specificando l'opzione. `NONE`

ENCRYPTION

```
encryption = [ '{ SSE_KMS | SSE_S3 }' ], default: SSE_S3
```

I file di output su Amazon S3 vengono crittografati utilizzando l'opzione di crittografia selezionata. Oltre ai dati, anche i file manifest e i file di metadati vengono crittografati in base all'opzione di crittografia selezionata. Al momento supportiamo la SSE crittografia _S3 e SSE _. KMS SSE_S3 è una crittografia lato server con Amazon S3 che crittografa i dati utilizzando la crittografia Advanced Encryption Standard () a 256 bit. AES SSE_ KMS è una crittografia lato server per crittografare i dati utilizzando chiavi gestite dal cliente.

KMS_KEY

```
kms_key = '<string>'
```

KMSKey è una chiave definita dal cliente per crittografare i risultati delle query esportate. KMSLa chiave è gestita in modo sicuro da AWS Key Management Service (AWS KMS) e utilizzata per crittografare i file di dati su Amazon S3.

FIELD_DELIMITER

```
field_delimiter ='<character>' , default: (,)
```

Quando si esportano i dati in CSV formato, questo campo specifica un singolo ASCII carattere utilizzato per separare i campi nel file di output, ad esempio un carattere pipe (|), una virgola (,) o un tab (/t). Il delimitatore predefinito per CSV i file è un carattere virgola. Se un valore nei dati contiene il delimitatore scelto, il delimitatore verrà citato tra virgolette. Ad esempio, se il valore dei dati contiene`Time,stream`, questo valore verrà citato come nei dati esportati. `"Time,stream"` Il carattere di virgoletta usato da Timestream per sono le LiveAnalytics virgolette doppie («).

Evitate di specificare il carattere di ritorno al carrello (ASCII130D, hex, text '\ r') o il carattere di interruzione di riga (ASCII10, hex 0A, text'\n') come `FIELD_DELIMITER` se desiderate includere le intestazioni inCSV, poiché ciò impedirà a molti parser di analizzare correttamente le intestazioni nell'output risultante. CSV

ESCAPED_DI

```
escaped_by = '<character>', default: (\)
```

Quando si esportano i dati in CSV formato, questo campo specifica il carattere che deve essere trattato come carattere di escape nel file di dati scritto nel bucket S3. L'escape avviene nei seguenti scenari:

1. Se il valore stesso contiene il carattere di virgoletta («), verrà eliminato utilizzando un carattere di escape. Ad esempio, se il valore è`Time"stream`, dove (\) è il carattere di escape configurato, allora verrà escluso come. `Time\"stream`
2. Se il valore contiene il carattere di escape configurato, verrà eliminato. Ad esempio, se il valore è`Time\stream`, allora verrà scappato come. `Time\\stream`

> **Note**
>
> Se l'output esportato contiene tipi di dati complessi come Arrays, Rows o Timeseries, verrà serializzato come stringa. JSON Di seguito è riportato un esempio.

Tipo di dati	Valore effettivo	Come viene eseguito l'escape del valore nel CSV formato [stringa serializz ataJSON]
Array	`[ 23,24,25 ]`	`"[23,24,25]"`
Riga	`( x=23.0, y=hello )`	`"{\"x\":23.0,\"y\": \"hello\"}"`
Serie temporali	`[ ( time=1970-01-01 00:00:00.000000010 , value=100.0 ), ( time=1970-01-01`	`"[{\"time\":\"1970 -01-01 00:00:00. 000000010Z\",\"val ue\":100.0},{\"tim`

Tipo di dati	Valore effettivo	Come viene eseguito l'escape del valore nel CSV formato [stringa serializz ataJSON]
	`00:00:00.000000012, value=120.0 ) ]`	`e\":\"1970-01-01 00:00:00.000000012 Z\",\"value\":120. 0}]"`

INCLUDE_HEADER

```
include_header = 'true' , default: 'false'
```

Quando si esportano i dati in CSV formato, questo campo consente di includere i nomi delle colonne come prima riga dei file di dati CSV esportati.

I valori accettati sono «true» e «false» e il valore predefinito è «false». Le opzioni di trasformazione del testo come `escaped_by` e `field_delimiter` si applicano anche alle intestazioni.

> **Note**
>
> Quando si includono le intestazioni, è importante non selezionare un carattere di ritorno (ASCII13, hex 0D, text '\ r') o un carattere di interruzione di riga (10, hex 0A, text'\n') come carattere di interruzione di riga (ASCII10, hex 0A, text'\n')`FIELD_DELIMITER`, poiché ciò impedirà a molti parser di analizzare correttamente le intestazioni nell'output risultante. CSV

MAX_FILE_SIZE

```
max_file_size = 'X[MB|GB]' , default: '78GB'
```

Questo campo specifica la dimensione massima dei file che l'`UNLOAD`istruzione crea in Amazon S3. L'`UNLOAD`istruzione può creare più file, ma la dimensione massima di ogni file scritto in Amazon S3 sarà approssimativamente quella specificata in questo campo.

Il valore del campo deve essere compreso tra 16 MB e 78 GB, inclusi. È possibile specificarlo in numeri interi come `12GB` o in decimali come o. `0.5GB` `24.7MB` Il valore predefinito è 78 GB.

La dimensione effettiva del file è approssimativa al momento della scrittura del file, pertanto la dimensione massima effettiva potrebbe non essere esattamente uguale al numero specificato.

Cosa viene scritto nel mio bucket S3?

Per ogni UNLOAD query eseguita correttamente, Timestream for LiveAnalytics scrive i risultati della query, il file di metadati e il file manifest nel bucket S3. Se hai partizionato i dati, hai tutte le cartelle delle partizioni nella cartella dei risultati. Il file manifesto contiene un elenco dei file che sono stati scritti dal comando. UNLOAD Il file di metadati contiene informazioni che descrivono le caratteristiche, le proprietà e gli attributi dei dati scritti.

Qual è il nome del file esportato?

Il nome del file esportato contiene due componenti, il primo componente è il QueryID e il secondo è un identificatore univoco.

CSVfile

```
S3://bucket_name/results/<queryid>_<UUID>.csv
S3://bucket_name/results/<partitioncolumn>=<partitionvalue>/<queryid>_<UUID>.csv
```

file compresso CSV

```
S3://bucket_name/results/<partitioncolumn>=<partitionvalue>/<queryid>_<UUID>.gz
```

File Parquet

```
S3://bucket_name/results/<partitioncolumn>=<partitionvalue>/<queryid>_<UUID>.parquet
```

Metadati e file manifesto

```
S3://bucket_name/<queryid>_<UUID>_manifest.json
S3://bucket_name/<queryid>_<UUID>_metadata.json
```

Poiché i dati in CSV formato vengono archiviati a livello di file, quando si comprimono i dati durante l'esportazione in S3, il file avrà un'estensione «.gz». Tuttavia, i dati in Parquet vengono compressi

a livello di colonna, quindi anche quando si comprimono i dati durante l'esportazione, il file avrà comunque l'estensione.parquet.

Quali informazioni contiene ogni file?

File manifest

Il file manifest fornisce informazioni sull'elenco dei file che vengono esportati con l'UNLOADesecuzione. Il file manifest è disponibile nel bucket S3 fornito con un nome di file:. `s3://<bucket_name>/<queryid>_<UUID>_manifest.json` Il file manifest conterrà l'URL dei file nella cartella dei risultati, il numero di record e le dimensioni dei rispettivi file e i metadati della query (ovvero i byte totali e le righe totali esportati in S3 per la query).

```
{
  "result_files": [
    {
        "url":"s3://my_timestream_unloads/ec2_metrics/
AEDAGANLHLBH4OLISD3CVOZZRWPX5GV2XCXRBKCVD554N6GWPWWXBP7LSG74V2Q_1448466917_szCL4YgVYzGXj2lS.gz"
        "file_metadata":
            {
                "content_length_in_bytes": 32295,
                "row_count": 10
            }
    },
    {
        "url":"s3://my_timestream_unloads/ec2_metrics/
AEDAGANLHLBH4OLISD3CVOZZRWPX5GV2XCXRBKCVD554N6GWPWWXBP7LSG74V2Q_1448466917_szCL4YgVYzGXj2lS.gz"
        "file_metadata":
            {
                "content_length_in_bytes": 62295,
                "row_count": 20
            }
    },
  ],
  "query_metadata":
    {
      "content_length_in_bytes": 94590,
      "total_row_count": 30,
      "result_format": "CSV",
      "result_version": "Amazon Timestream version 1.0.0"
    },
  "author": {
        "name": "Amazon Timestream",
```

```
        "manifest_file_version": "1.0"
  }
}
```

Metadati

Il file di metadati fornisce informazioni aggiuntive sul set di dati come il nome della colonna, il tipo di colonna e lo schema. <queryid>Il file di metadati è disponibile nel bucket S3 fornito con un nome di file: S3: //bucket_name/ _< >_metadata.json UUID

Di seguito è riportato un esempio di file di metadati.

```
{
    "ColumnInfo": [
        {
            "Name": "hostname",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        },
        {
            "Name": "region",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        },
        {
            "Name": "measure_name",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        },
        {
            "Name": "cpu_utilization",
            "Type": {
                "TimeSeriesMeasureValueColumnInfo": {
                    "Type": {
                        "ScalarType": "DOUBLE"
                    }
                }
            }
        }
  ],
```

```
  "Author": {
        "Name": "Amazon Timestream",
        "MetadataFileVersion": "1.0"
  }
}
```

Le informazioni sulle colonne condivise nel file di metadati hanno la stessa struttura di quelle `ColumnInfo` inviate in Query API Response for `SELECT` Query.

Risultati

La cartella dei risultati contiene i dati esportati in formato Apache Parquet o in formato. CSV

Esempio

Quando invii una richiesta come quella riportata di seguito tramite `UNLOAD` Query, API

```
UNLOAD(SELECT user_id, ip_address, event, session_id, measure_name, time, query,
 quantity, product_id, channel
                    FROM sample_clickstream.sample_shopping WHERE time BETWEEN ago(2d)
 AND now())
                TO 's3://my_timestream_unloads/withoutpartition/' WITH ( format='CSV',
 compression='GZIP')
```

`UNLOAD`la risposta alla query avrà 1 riga * 3 colonne. Queste 3 colonne sono:

- righe di tipoBIGINT, che indicano il numero di righe esportate
- metadataFile di tipoVARCHAR, che è l'S3 del file URI di metadati esportato
- manifestFile di tipoVARCHAR: che è l'S3 del file manifesto URI esportato

Riceverai la seguente risposta da Query: API

```
{
    "Rows": [
        {
            "Data": [
                {
                    "ScalarValue": "20" # No of rows in output across all files
                },
                {
```

```
                    "ScalarValue": "s3://my_timestream_unloads/withoutpartition/
AEDAAANGH3D7FYHOBQGQQMEAISCJ45B42OWWJMOT4N6RRJICZUA7R25VYVOHJIY_<UUID>_metadata.json"
 #Metadata file
                },
                {
                    "ScalarValue": "s3://my_timestream_unloads/withoutpartition/
AEDAAANGH3D7FYHOBQGQQMEAISCJ45B42OWWJMOT4N6RRJICZUA7R25VYVOHJIY_<UUID>_manifest.json"
 #Manifest file
                }
            ]
        }
    ],
    "ColumnInfo": [
        {
            "Name": "rows",
            "Type": {
                "ScalarType": "BIGINT"
            }
        },
        {
            "Name": "metadataFile",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        },
        {
            "Name": "manifestFile",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        }
    ],
    "QueryId": "AEDAAANGH3D7FYHOBQGQQMEAISCJ45B42OWWJMOT4N6RRJICZUA7R25VYVOHJIY",
    "QueryStatus": {
        "ProgressPercentage": 100.0,
        "CumulativeBytesScanned": 1000,
        "CumulativeBytesMetered": 10000000
    }
}
```

Tipi di dati

L'UNLOADistruzione supporta tutti i tipi di dati del linguaggio di query LiveAnalytics di Timestream for descritto in Tipi di dati supportati tranne `time` e. `unknown`

Prerequisiti per from Timestream for UNLOAD LiveAnalytics

Di seguito sono riportati i prerequisiti per la scrittura di dati su S3 utilizzando from Timestream for. `UNLOAD` LiveAnalytics

- È necessario disporre dell'autorizzazione a leggere i dati dal Timestream per le LiveAnalytics tabelle da utilizzare in un comando. `UNLOAD`
- È necessario disporre di un bucket Amazon S3 nella stessa AWS regione del tuo Timestream per le risorse. LiveAnalytics
- Per il bucket S3 selezionato, assicurati che la policy del bucket S3 disponga anche delle autorizzazioni per consentire a Timestream di esportare i dati. LiveAnalytics
- Le credenziali utilizzate per eseguire la `UNLOAD` query devono disporre delle autorizzazioni AWS Identity and Access Management (IAM) necessarie che consentano a Timestream di LiveAnalytics scrivere i dati su S3. Un esempio di politica potrebbe essere il seguente:

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": [
                "timestream:Select",
                "timestream:ListMeasures",
                "timestream:WriteRecords",
                "timestream:Unload"
            ],
            "Resource": "arn:aws:timestream:<region>:<account_id>:database/
<database_name>/table/<table_name>"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetBucketAcl",
                "s3:PutObject",
                "s3:GetObjectMetadata",
                "s3:AbortMultipartUpload"
```

```
            ],
            "Resource": [
                "arn:aws:s3:::<S3_Bucket_Created>",
                "arn:aws:s3:::<S3_Bucket_Created>/*"
            ]
        }
    ]
}
```

Per ulteriori informazioni su queste autorizzazioni di scrittura S3, consulta la guida di Amazon Simple Storage Service. Se utilizzi una KMS chiave per crittografare i dati esportati, consulta quanto segue per le politiche aggiuntive richieste. IAM

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
            "kms:DescribeKey",
            "kms:Decrypt",
            "kms:GenerateDataKey*"
        ],
        "Resource": "<account_id>-arn:aws:kms:<region>:<account_id>:key/*",
        "Condition": {
            "ForAnyValue:StringLike": {
                "kms:ResourceAliases": "alias/<Alias_For_Generated_Key>"
            }
        }
    }, {
        "Effect": "Allow",
        "Action": [
            "kms:CreateGrant"
        ],
        "Resource": "<account_id>-arn:aws:kms:<region>:<account_id>:key/*",
        "Condition": {
            "ForAnyValue:StringEquals": {
                "kms:EncryptionContextKeys": "aws:timestream:<database_name>"
            },
            "Bool": {
                "kms:GrantIsForAWSResource": true
            },
            "StringLike": {
```

```
                "kms:ViaService": "timestream.<region>.amazonaws.com"
            },
            "ForAnyValue:StringLike": {
                "kms:ResourceAliases": "alias/<Alias_For_Generated_Key>"
            }
        }
    }
]
}
```

Procedure consigliate per UNLOAD From Timestream per LiveAnalytics

Di seguito sono riportate le best practice relative al UNLOAD comando.

- La quantità di dati che può essere esportata nel bucket S3 utilizzando il `UNLOAD` comando non è limitata. Tuttavia, la query scade dopo 60 minuti e consigliamo di esportare non più di 60 GB di dati in una singola query. Se devi esportare più di 60 GB di dati, suddividi il lavoro su più query.
- Sebbene sia possibile inviare migliaia di richieste a S3 per caricare i dati, si consiglia di parallelizzare le operazioni di scrittura su più prefissi S3. Fai riferimento alla documentazione qui. La velocità di API chiamata di S3 potrebbe essere ridotta quando più lettori/scrittori accedono alla stessa cartella.
- Dato il limite alla lunghezza della chiave S3 per la definizione di un prefisso, consigliamo di avere nomi di bucket e cartelle entro 10-15 caratteri, specialmente quando si utilizza la clausola. `partitioned_by`
- Quando ricevi un 4XX o 5XX per le query contenenti l'`UNLOAD`istruzione, è possibile che i risultati parziali vengano scritti nel bucket S3. Timestream for LiveAnalytics non elimina alcun dato dal bucket. Prima di eseguire un'altra `UNLOAD` query con la stessa destinazione S3, consigliamo di eliminare manualmente i file creati dalla query fallita. È possibile identificare i file scritti da una query fallita con i file corrispondenti`QueryExecutionId`. Per le query non riuscite, Timestream for LiveAnalytics non esporta un file manifest nel bucket S3.
- Timestream for LiveAnalytics utilizza il caricamento in più parti per esportare i risultati delle query su S3. Quando ricevi un 4XX o 5XX da Timestream per domande contenenti un'UNLOADistruzione, Timestream LiveAnalytics for evita al massimo il caricamento di più parti, LiveAnalytics ma è possibile che alcune parti incomplete rimangano incomplete. Pertanto, ti consigliamo di impostare una pulizia automatica dei caricamenti incompleti in più parti nel tuo bucket S3 seguendo le linee guida riportate qui.

Consigli per accedere ai dati in formato tramite parser CSV CSV

- CSVi parser non ti consentono di avere lo stesso carattere nei caratteri delimiter, escape e quote.
- Alcuni CSV parser non sono in grado di interpretare tipi di dati complessi come gli array, consigliamo di interpretarli tramite il deserializzatore. JSON

Raccomandazioni per l'accesso ai dati in formato Parquet

1. Se il tuo caso d'uso richiede il supporto di UTF -8 caratteri nello schema, ovvero nel nome della colonna, ti consigliamo di utilizzare la libreria Parquet-MR.
2. Il timestamp nei risultati è rappresentato come un numero intero a 12 byte () INT96
3. Le serie temporali verranno rappresentate come`array<row<time, value>>`, le altre strutture annidate utilizzeranno i tipi di dati corrispondenti supportati nel formato Parquet

Utilizzo della clausola partition_by

- La colonna utilizzata nel `partitioned_by` campo deve essere l'ultima colonna nella query di selezione. Se nel `partitioned_by` campo viene utilizzata più di una colonna, le colonne devono essere le ultime colonne della query di selezione e nello stesso ordine in cui sono utilizzate nel `partition_by` campo.
- I valori delle colonne utilizzati per partizionare i dati (`partitioned_by`campo) possono contenere solo ASCII caratteri. Mentre Timestream for LiveAnalytics consente UTF -8 caratteri nei valori, S3 supporta solo ASCII caratteri come chiavi oggetto.

Esempio di utilizzo di from Timestream per UNLOAD LiveAnalytics

Supponiamo che tu stia monitorando le metriche della sessione utente, le sorgenti di traffico e gli acquisti di prodotti del tuo sito di e-commerce. Stai utilizzando Timestream per LiveAnalytics ottenere informazioni in tempo reale sul comportamento degli utenti, sulle vendite di prodotti ed eseguire analisi di marketing sui canali di traffico (ricerca organica, social media, traffico diretto, campagne a pagamento e altri) che indirizzano i clienti al sito web.

Argomenti

- Esportazione dei dati senza partizioni
- Partizionamento dei dati per canale

- Partizionamento dei dati per evento
- Partizionamento dei dati sia per canale che per evento
- File di manifesto e di metadati
- Utilizzo dei crawler Glue per creare Glue Data Catalog

Esportazione dei dati senza partizioni

Vuoi esportare gli ultimi due giorni dei tuoi dati in CSV formato.

```
UNLOAD(SELECT user_id, ip_address, event, session_id, measure_name, time,
query, quantity, product_id, channel
FROM sample_clickstream.sample_shopping
WHERE time BETWEEN ago(2d) AND now())
TO 's3://<bucket_name>/withoutpartition'
WITH (  format='CSV',
compression='GZIP')
```

Partizionamento dei dati per canale

Desideri esportare i dati degli ultimi due giorni in CSV formato ma desideri avere i dati di ciascun canale di traffico in una cartella separata. A tale scopo, è necessario partizionare i dati utilizzando la `channel` colonna, come illustrato di seguito.

```
UNLOAD(SELECT user_id, ip_address, event, session_id, measure_name, time,
query, quantity, product_id, channel
FROM sample_clickstream.sample_shopping
WHERE time BETWEEN ago(2d) AND now())
TO 's3://<bucket_name>/partitionbychannel/'
WITH (
partitioned_by = ARRAY ['channel'],
format='CSV',
compression='GZIP')
```

Partizionamento dei dati per evento

Desideri esportare i dati degli ultimi due giorni in CSV formato ma desideri avere i dati per ogni evento in una cartella separata. A tale scopo, è necessario partizionare i dati utilizzando la `event` colonna, come illustrato di seguito.

```
UNLOAD(SELECT user_id, ip_address, channel, session_id, measure_name, time,
```

```
query, quantity, product_id, event
FROM sample_clickstream.sample_shopping
WHERE time BETWEEN ago(2d) AND now())
TO 's3://<bucket_name>/partitionbyevent/'
WITH (
partitioned_by = ARRAY ['event'],
format='CSV',
compression='GZIP')
```

Partizionamento dei dati sia per canale che per evento

Desiderate esportare i dati degli ultimi due giorni in CSV formato, ma desiderate che i dati per ogni canale e all'interno del canale memorizzino ogni evento in una cartella separata. A tale scopo, è necessario partizionare i dati utilizzando entrambe `channel` le `event` colonne, come illustrato di seguito.

```
UNLOAD(SELECT user_id, ip_address, session_id, measure_name, time,
query, quantity, product_id, channel,event
FROM sample_clickstream.sample_shopping
WHERE time BETWEEN ago(2d) AND now())
TO 's3://<bucket_name>/partitionbychannelevent/'
WITH (
partitioned_by = ARRAY ['channel','event'],
format='CSV',
compression='GZIP')
```

File di manifesto e di metadati

File manifest

Il file manifest fornisce informazioni sull'elenco dei file che vengono esportati con l'UNLOADesecuzione. Il file manifest è disponibile nel bucket S3 fornito con un nome di file:. `S3://bucket_name/<queryid>_<UUID>_manifest.json` Il file manifest conterrà l'URL dei file nella cartella dei risultati, il numero di record e le dimensioni dei rispettivi file e i metadati della query (ovvero i byte totali e le righe totali esportati in S3 per la query).

```
{
  "result_files": [
    {
        "url":"s3://my_timestream_unloads/ec2_metrics/
AEDAGANLHLBH4OLISD3CVOZZRWPX5GV2XCXRBKCVD554N6GWPWWXBP7LSG74V2Q_1448466917_szCL4YgVYzGXj2lS.gz"
        "file_metadata":
```

```
            {
                "content_length_in_bytes": 32295,
                "row_count": 10
            }
    },
    {
        "url":"s3://my_timestream_unloads/ec2_metrics/
AEDAGANLHLBH4OLISD3CVOZZRWPX5GV2XCXRBKCVD554N6GWPWWXBP7LSG74V2Q_1448466917_szCL4YgVYzGXj2lS.gz"
        "file_metadata":
            {
                "content_length_in_bytes": 62295,
                "row_count": 20
            }
    },
  ],
  "query_metadata":
    {
      "content_length_in_bytes": 94590,
      "total_row_count": 30,
      "result_format": "CSV",
      "result_version": "Amazon Timestream version 1.0.0"
    },
  "author": {
        "name": "Amazon Timestream",
        "manifest_file_version": "1.0"
  }
}
```

Metadati

Il file di metadati fornisce informazioni aggiuntive sul set di dati come il nome della colonna, il tipo di colonna e lo schema. <queryid>Il file di metadati è disponibile nel bucket S3 fornito con un nome di file: S3: //bucket_name/ _< >_metadata.json UUID

Di seguito è riportato un esempio di file di metadati.

```
{
    "ColumnInfo": [
        {
            "Name": "hostname",
            "Type": {
                "ScalarType": "VARCHAR"
            }
```

```
        },
        {
            "Name": "region",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        },
        {
            "Name": "measure_name",
            "Type": {
                "ScalarType": "VARCHAR"
            }
        },
        {
            "Name": "cpu_utilization",
            "Type": {
                "TimeSeriesMeasureValueColumnInfo": {
                    "Type": {
                        "ScalarType": "DOUBLE"
                    }
                }
            }
        }
  ],
  "Author": {
        "Name": "Amazon Timestream",
        "MetadataFileVersion": "1.0"
  }
}
```

Le informazioni sulle colonne condivise nel file di metadati hanno la stessa struttura di quelle `ColumnInfo` inviate in Query API Response for `SELECT` Query.

Utilizzo dei crawler Glue per creare Glue Data Catalog

1. Accedi al tuo account con le credenziali di amministratore per la seguente convalida.
2. Crea un database Crawler for Glue utilizzando le linee guida fornite qui. Tieni presente che la cartella S3 da fornire nell'origine dati dovrebbe essere la cartella dei risultati, ad esempio. `UNLOAD s3://my_timestream_unloads/results`
3. Esegui il crawler seguendo le linee guida riportate qui.
4. Visualizza la tabella Glue.

- Vai a AWS Glue → Tabelle.
- Vedrai una nuova tabella creata con il prefisso della tabella fornito durante la creazione del crawler.
- È possibile visualizzare lo schema e le informazioni sulla partizione facendo clic sulla visualizzazione dei dettagli della tabella.

Di seguito sono riportati altri AWS servizi e progetti open source che utilizzano il AWS Glue Data Catalog.

- Amazon Athena: per ulteriori informazioni, consulta Comprendere tabelle, database e cataloghi di dati nella Guida per l'utente di Amazon Athena.
- Amazon Redshift Spectrum — Per ulteriori informazioni, consulta la sezione Interrogazione di dati esterni utilizzando Amazon Redshift Spectrum nella Amazon Redshift Database Developer Guide.
- Amazon EMR — Per ulteriori informazioni, consulta Utilizzare politiche basate sulle risorse per l'accesso di Amazon EMR a AWS Glue Data Catalog nella Amazon EMR Management Guide.
- AWS Client Glue Data Catalog per Apache Hive metastore — Per ulteriori informazioni su questo progetto GitHub, consulta AWS Glue Data Catalog Client for Apache Hive Metastore.

UNLOADLimiti per From Timestream per LiveAnalytics

Di seguito sono riportati i limiti relativi al `UNLOAD` comando.

- La concorrenza per le query che utilizzano l'`UNLOAD`istruzione è pari a 1 query al secondo ()QPS. Il superamento della frequenza di query potrebbe comportare una limitazione.
- Le query contenenti `UNLOAD` istruzioni possono esportare al massimo 100 partizioni per query. Si consiglia di controllare il numero distinto della colonna selezionata prima di utilizzarla per partizionare i dati esportati.
- Le interrogazioni contenenti `UNLOAD` dichiarazioni scadono dopo 60 minuti.
- La dimensione massima dei file creati dall'`UNLOAD`istruzione in Amazon S3 è di 78 GB.

Per altri limiti per Timestream per, vedi LiveAnalytics Quote

Utilizzo di informazioni dettagliate sulle query per ottimizzare le query in Amazon Timestream

Query Insights è una funzionalità di ottimizzazione delle prestazioni che consente di ottimizzare le query, migliorarne le prestazioni e ridurre i costi. Con Query Insights, puoi valutare l'efficienza di eliminazione delle tue query basata su chiavi di partizione temporali, temporali e spaziali. Utilizzando le informazioni sulle query, è inoltre possibile identificare le aree di miglioramento per migliorare le prestazioni delle query. Inoltre, con Query Insights, è possibile valutare l'efficacia con cui le query utilizzano l'indicizzazione basata sul tempo e sulla chiave di partizione per ottimizzare il recupero dei dati. Per ottimizzare le prestazioni delle query, è essenziale ottimizzare i parametri temporali e spaziali che regolano l'esecuzione delle query.

Argomenti

Vantaggi degli approfondimenti sulle query

Di seguito sono riportati i principali vantaggi dell'utilizzo di Query Insights:

- Identificazione delle query inefficienti: Query Insights fornisce informazioni sull'eliminazione basata sul tempo e sugli attributi delle tabelle a cui accede la query. Queste informazioni consentono di identificare le tabelle a cui si accede in modo non ottimale.
- Ottimizzazione del modello di dati e del partizionamento: è possibile utilizzare le informazioni di Query Insights per accedere e perfezionare il modello di dati e la strategia di partizionamento.
- Ottimizzazione delle query: Query Insights evidenzia le opportunità di utilizzare gli indici in modo più efficace.

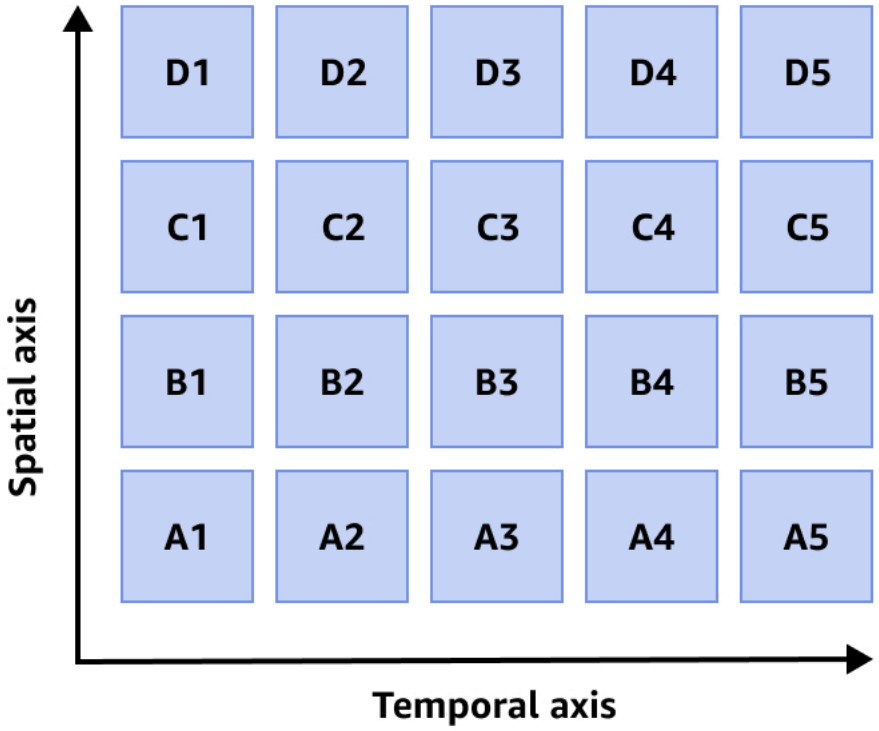
Ottimizzazione dell'accesso ai dati in Amazon Timestream

Puoi ottimizzare i modelli di accesso ai dati in Amazon Timestream utilizzando lo schema di partizionamento Timestream o le tecniche di organizzazione dei dati.

Argomenti

Schema di partizionamento Timestream

Amazon Timestream utilizza uno schema di partizionamento altamente scalabile in cui ogni tabella Timestream può avere centinaia, migliaia o persino milioni di partizioni indipendenti. Un servizio di tracciamento e indicizzazione delle partizioni ad alta disponibilità gestisce il partizionamento, minimizzando l'impatto degli errori e rendendo il sistema più resiliente.



Organizzazione dei dati

Timestream archivia ogni punto dati che acquisisce in un'unica partizione. Quando si inseriscono i dati in una tabella Timestream, Timestream crea automaticamente partizioni in base ai timestamp, alla chiave di partizione e ad altri attributi di contesto presenti nei dati. Oltre a partizionare i dati in base al tempo (partizionamento temporale), Timestream partiziona anche i dati in base alla chiave

di partizionamento selezionata e ad altre dimensioni (partizionamento spaziale). Questo approccio è progettato per distribuire il traffico di scrittura e consentire un'efficace eliminazione dei dati per le query.

La funzionalità di analisi delle query fornisce informazioni preziose sull'efficienza di eliminazione delle query, che include la copertura spaziale delle query e la copertura temporale delle query.

Argomenti

- QuerySpatialCoverage
- QueryTemporalCoverage

QuerySpatialCoverage

La QuerySpatialCoveragemetrica fornisce informazioni sulla copertura spaziale della query eseguita e sulla tabella con la riduzione spaziale più inefficiente. Queste informazioni possono aiutarti a identificare le aree di miglioramento nella strategia di partizionamento per migliorare la potatura spaziale. Il valore della `QuerySpatialCoverage` metrica è compreso tra 0 e 1. Più basso è il valore della metrica, più ottimale è l'eliminazione delle query sull'asse spaziale. Ad esempio, un valore di 0,1 indica che la query analizza il 10% dell'asse spaziale. Il valore 1 indica che l'interrogazione analizza il 100% dell'asse spaziale.

Example Utilizzo di Query Insights per analizzare la copertura spaziale di un'interrogazione

Supponiamo che tu abbia un database Timestream che memorizza i dati meteorologici. Supponiamo che la temperatura venga registrata ogni ora dalle stazioni meteorologiche situate in diversi stati degli Stati Uniti. Immagina di scegliere `State` come chiave di partizionamento definita dal cliente (CDPK) per partizionare i dati per stato.

Supponiamo di eseguire una query per recuperare la temperatura media di tutte le stazioni meteorologiche della California tra le 14:00 e le 16:00 di un giorno specifico. L'esempio seguente mostra la query per questo scenario.

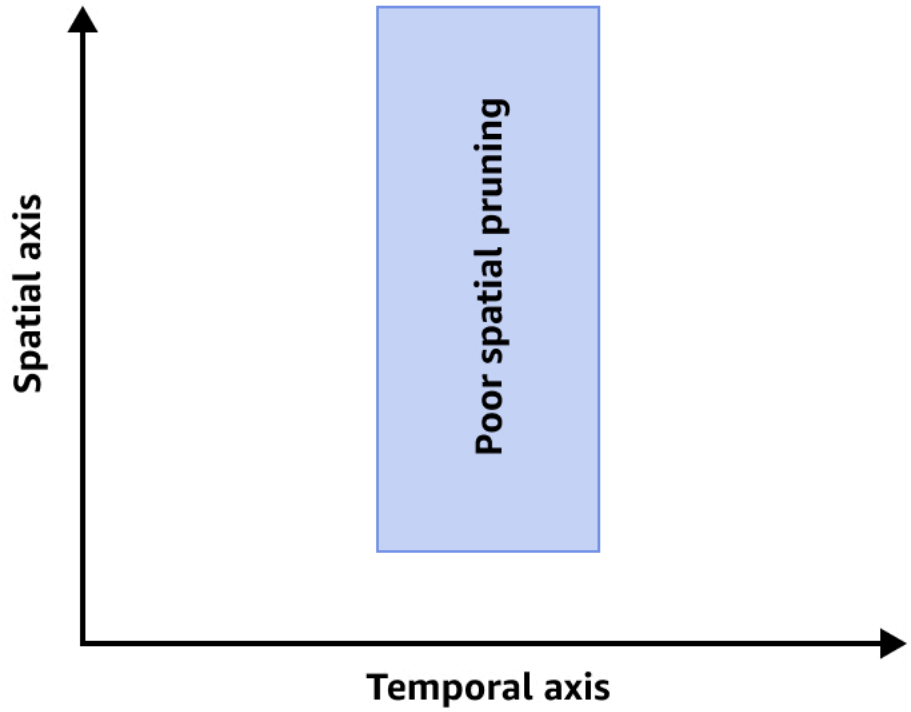
```
SELECT AVG(temperature)
FROM "weather_data"."hourly_weather"
WHERE time >= '2024-10-01 14:00:00' AND time < '2024-10-01 16:00:00'
  AND state = 'CA';
```

Utilizzando la funzionalità Query Insights, è possibile analizzare la copertura spaziale dell'interrogazione. Immagina che la `QuerySpatialCoverage` metrica restituisca un valore di 0,02.

Ciò significa che la query ha scansionato solo il 2% dell'asse spaziale, il che è efficiente. In questo caso, l'interrogazione è stata in grado di ridurre efficacemente l'intervallo spaziale, recuperando solo i dati dalla California e ignorando i dati provenienti da altri stati.

Al contrario, se la `QuerySpatialCoverage` metrica restituisse un valore di 0,8, indicherebbe che la query ha scansionato l'80% dell'asse spaziale, il che è meno efficiente. Ciò potrebbe suggerire che la strategia di partizionamento debba essere perfezionata per migliorare la potatura spaziale. Ad esempio, è possibile selezionare la chiave di partizione come città o regione anziché come stato. Analizzando la `QuerySpatialCoverage` metrica, è possibile identificare le opportunità per ottimizzare la strategia di partizionamento e migliorare le prestazioni delle query.

L'immagine seguente mostra una potatura spaziale scadente.



Per migliorare l'efficienza della potatura spaziale, puoi eseguire una o entrambe le seguenti operazioni:

- Aggiungi`measure_name`, la chiave di partizionamento predefinita, o usa i CDPK predicati nella tua query.

- Se hai già aggiunto gli attributi menzionati nel punto precedente, rimuovi le funzioni relative a questi attributi o clausole, ad esempio. `LIKE`

QueryTemporalCoverage

La `QueryTemporalCoverage` metrica fornisce informazioni sull'intervallo temporale analizzato dalla query eseguita, inclusa la tabella con l'intervallo di tempo più ampio scansionato. Il valore della `QueryTemporalCoverage` metrica è l'intervallo di tempo rappresentato in nanosecondi. Più basso è il valore di questa metrica, più ottimale è l'eliminazione delle query sull'intervallo temporale. Ad esempio, una query che analizza i dati degli ultimi minuti è più efficiente di una query che analizza l'intero intervallo di tempo della tabella.

Example

Supponiamo di avere un database Timestream che memorizza i dati dei sensori IoT, con misurazioni effettuate ogni minuto dai dispositivi situati in uno stabilimento di produzione. Supponiamo di aver partizionato i dati per. `device_ID`

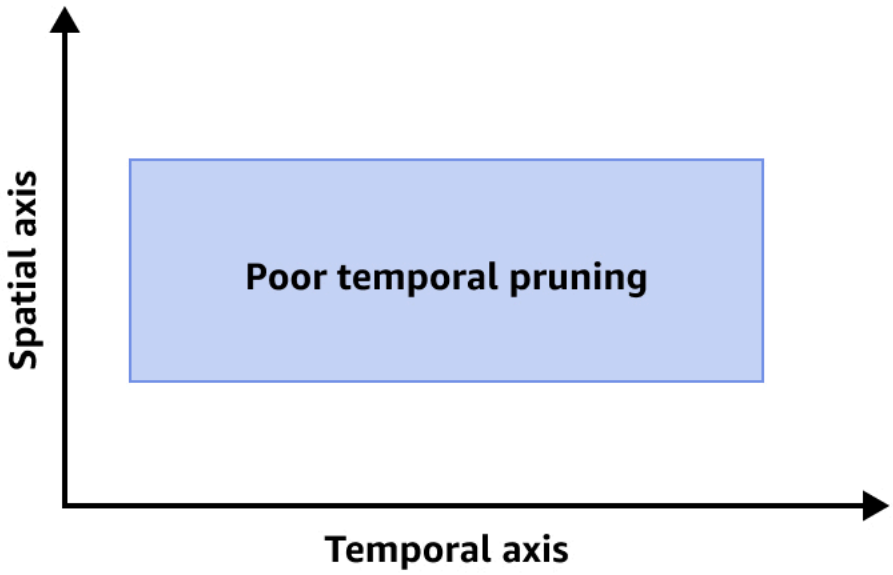
Supponiamo di eseguire una query per recuperare la lettura media del sensore per un dispositivo specifico negli ultimi 30 minuti. L'esempio seguente mostra l'interrogazione per questo scenario.

```
SELECT AVG(sensor_reading)
FROM "sensor_data"."factory_1"
WHERE device_id = 'DEV_123'
  AND time >= NOW() - INTERVAL 30 MINUTE and time < NOW();
```

Utilizzando la funzionalità Query Insights, è possibile analizzare l'intervallo temporale analizzato dalla query. Immagina che la `QueryTemporalCoverage` metrica restituisca un valore di 1800000000000 nanosecondi (30 minuti). Ciò significa che la query ha analizzato solo i dati degli ultimi 30 minuti, ovvero un intervallo temporale relativamente ristretto. Questo è un buon segno perché indica che la query è stata in grado di ridurre efficacemente il partizionamento temporale e ha recuperato solo i dati richiesti.

Al contrario, se la `QueryTemporalCoverage` metrica ha restituito un valore di 1 anno in nanosecondi, indica che la query ha analizzato un intervallo di tempo di un anno nella tabella, il che è meno efficiente. Ciò potrebbe suggerire che la query non è ottimizzata per l'eliminazione temporale e che è possibile migliorarla aggiungendo filtri temporali.

L'immagine seguente mostra una scarsa potatura temporale.



Per migliorare la potatura temporale, ti consigliamo di eseguire una o tutte le seguenti operazioni:

- Aggiungi i predicati temporali mancanti nella query e assicurati che i predicati temporali riducano la finestra temporale desiderata.
- Rimuovete le funzioni, ad esempio`MAX()`, i predicati temporali.
- Aggiungi predicati temporali a tutte le sottointerrogazioni. Questo è importante se le tue sottoquery uniscono tabelle di grandi dimensioni o eseguono operazioni complesse.

Attivazione di informazioni dettagliate sulle query in Amazon Timestream

Puoi abilitare gli approfondimenti sulle query per le tue query con approfondimenti forniti direttamente tramite la risposta alla query. L'attivazione di Query Insights non richiede un'infrastruttura aggiuntiva né comporta costi aggiuntivi. Quando si abilita Query Insights, vengono restituiti i campi di metadati relativi alle prestazioni delle query oltre ai risultati delle query come parte della risposta alla query. È possibile utilizzare queste informazioni per ottimizzare le query per migliorare le prestazioni delle query e ridurre i costi delle query.

Per informazioni sull'attivazione di Query Insights, consultaEsecuzione di una query .

Per visualizzare esempi delle risposte restituite abilitando Query Insights, consulta Esempi di query pianificate.

> **Note**
>
> - Quando abiliti Query Insights, it rate limita la query a 1 query al secondo (QPS). Per evitare impatti sulle prestazioni, ti consigliamo vivamente di abilitare Query Insights solo durante la fase di valutazione delle query, prima di distribuirle in produzione.
> - Le informazioni fornite in Query Insights alla fine sono coerenti, il che significa che potrebbero cambiare man mano che nuovi dati vengono continuamente inseriti nelle tabelle.

Ottimizzazione delle query utilizzando la risposta a Query Insights

Supponiamo che tu stia utilizzando Amazon Timestream LiveAnalytics per monitorare il consumo di energia in varie località. Immagina di avere due tabelle nel tuo database denominate `raw-metrics` e. `aggregate-metrics`

La `raw-metrics` tabella memorizza dati energetici dettagliati a livello di dispositivo e contiene le seguenti colonne:

- Timestamp
- Stato, ad esempio, Washington
- ID dispositivo
- Consumo energetico

I dati di questa tabella vengono raccolti e archiviati in modo minute-by-minute granulare. La tabella utilizza `State` come. CDPK

La `aggregate-metrics` tabella memorizza il risultato di un'interrogazione pianificata per aggregare i dati sul consumo energetico di tutti i dispositivi su base oraria. Questa tabella contiene le seguenti colonne:

- Timestamp
- Stato, ad esempio, Washington
- Consumo energetico totale

La `aggregate-metrics` tabella memorizza questi dati con una granularità oraria. La tabella utilizza `State` come. CDPK

Argomenti

Interrogazione del consumo energetico delle ultime 24 ore

Supponiamo di voler estrarre l'energia totale consumata a Washington nelle ultime 24 ore.
Per trovare questi dati, puoi sfruttare i punti di forza di entrambe le tabelle: `raw-metrics` e. `aggregate-metrics` La `aggregate-metrics` tabella fornisce dati orari sul consumo energetico nelle ultime 23 ore, mentre la `raw-metrics` tabella offre dati granulari al minuto per l'ultima ora. Interrogando entrambe le tabelle, è possibile ottenere un quadro completo e preciso del consumo di energia a Washington nelle ultime 24 ore.

```
SELECT am.time, am.state, am.total_energy_consumption,
rm.time, rm.state, rm.device_id, rm.energy_consumption
FROM
 "metrics"."aggregate-metrics" am
 LEFT JOIN "metrics"."raw-metrics" rm ON am.state = rm.state
WHERE rm.time >= ago(1h) and rm.time < now()
```

Questa query di esempio viene fornita solo a scopo illustrativo e potrebbe non funzionare così com'è. Ha lo scopo di dimostrare il concetto, ma potrebbe essere necessario modificarlo per adattarlo al caso d'uso o all'ambiente specifico.

Dopo aver eseguito questa query, potresti notare che il tempo di risposta alla query è più lento del previsto. Per identificare la causa principale di questo problema di prestazioni, è possibile utilizzare la funzionalità Query Insights per analizzare le prestazioni della query e ottimizzarne l'esecuzione.

L'esempio seguente mostra la risposta a Query Insights.

```
queryInsightsResponse={
                QuerySpatialCoverage: {
```

```
                    Max: {
                        Value: 1.0,
                        TableArn: arn:aws:timestream:us-east-1:123456789012:database/
metrics/table/raw-metrics,
                        PartitionKey: [State]
                    }
                },
                QueryTemporalRange: {
                    Max: {
                        Value:31540000000000000 //365 days,
                        TableArn: arn:aws:timestream:us-east-1:123456789012:database/
metrics/table/aggregate-metrics
                    }
                },
                QueryTableCount: 2,
                OutputRows: 83,
                OutputBytes: 590
```

La risposta a Query Insights fornisce le seguenti informazioni:

- Intervallo temporale: la query ha analizzato un intervallo temporale eccessivo di 365 giorni per la tabella. `aggregate-metrics` Ciò indica un uso inefficiente del filtro temporale.
- Copertura spaziale: l'interrogazione ha analizzato l'intero intervallo spaziale (100%) della tabella. `raw-metrics` Ciò suggerisce che il filtro spaziale non viene utilizzato in modo efficace.

Se la query accede a più di una tabella, Query Insights fornisce le metriche per la tabella con il modello di accesso meno ottimale.

Ottimizzazione della query per l'intervallo temporale

In base alla risposta a Query Insights, è possibile ottimizzare la query per l'intervallo temporale, come illustrato nell'esempio seguente.

```
SELECT am.time, am.state, am.total_energy_consumption,
rm.time, rm.state, rm.device_id, rm.energy_consumption
FROM
  "metrics"."aggregate-metrics" am
  LEFT JOIN "metrics"."raw-metrics" rm ON am.state = rm.state
WHERE
  am.time >=  ago(23h) and am.time < now()
  AND rm.time >=  ago(1h) and rm.time < now()
```

```
  AND rm.state = 'Washington'
```

Se si esegue nuovamente il `QueryInsights` comando, viene restituita la seguente risposta.

```
queryInsightsResponse={
                QuerySpatialCoverage: {
                    Max: {
                        Value: 1.0,
                        TableArn: arn:aws:timestream:us-east-1:123456789012:database/
metrics/table/aggregate-metrics,
                        PartitionKey: [State]
                    }
                },
                QueryTemporalRange: {
                    Max: {
                        Value: 82800000000000 //23 hours,
                        TableArn: arn:aws:timestream:us-east-1:123456789012:database/
metrics/table/aggregate-metrics
                    }
                },
                QueryTableCount: 2,
                OutputRows: 83,
                OutputBytes: 590
```

Questa risposta mostra che la copertura spaziale della `aggregate-metrics` tabella è ancora del 100%, il che è inefficiente. La sezione seguente mostra come ottimizzare l'interrogazione per la copertura spaziale.

Ottimizzazione dell'interrogazione per la copertura spaziale

In base alla risposta a Query Insights, è possibile ottimizzare la query per la copertura spaziale, come illustrato nell'esempio seguente.

```
SELECT am.time, am.state, am.total_energy_consumption,
rm.time, rm.state, rm.device_id, rm.energy_consumption
FROM
  "metrics"."aggregate-metrics" am
  LEFT JOIN "metrics"."raw-metrics" rm ON am.state = rm.state
WHERE
  am.time >=  ago(23h) and am.time < now()
  AND am.state ='Washington'
  AND rm.time >=  ago(1h) and rm.time < now()
```

```
  AND rm.state = 'Washington'
```

Se si esegue nuovamente il `QueryInsights` comando, viene restituita la seguente risposta.

```
queryInsightsResponse={
                QuerySpatialCoverage: {
                    Max: {
                        Value: 0.02,
                        TableArn: arn:aws:timestream:us-east-1:123456789012:database/
metrics/table/aggregate-metrics,
                        PartitionKey: [State]
                    }
                },
                QueryTemporalRange: {
                    Max: {
                        Value: 82800000000000 //23 hours,
                        TableArn: arn:aws:timestream:us-east-1:123456789012:database/
metrics/table/aggregate-metrics
                    }
                },
                QueryTableCount: 2,
                OutputRows: 83,
                OutputBytes: 590
```

Prestazioni di interrogazione migliorate

Dopo aver ottimizzato la query, Query Insights fornisce le seguenti informazioni:

- La potatura temporale della `aggregate-metrics` tavola è di 23 ore. Ciò indica che vengono scansionate solo 23 ore dell'intervallo temporale.
- La potatura spaziale per `aggregate-metrics` la tavola è 0,02. Ciò indica che viene scansionato solo il 2% dei dati relativi all'intervallo spaziale della tabella. La query analizza una parte molto piccola delle tabelle, con conseguente aumento delle prestazioni e riduzione dell'utilizzo delle risorse. La migliore efficienza di potatura indica che la query è ora ottimizzata per le prestazioni.

Lavorare con AWS Backup

La funzionalità di protezione dei dati di Amazon Timestream LiveAnalytics for è una soluzione completamente gestita per aiutarti a soddisfare i requisiti di conformità normativa e continuità aziendale. La funzionalità è abilitata attraverso l'integrazione nativa con AWS Backup, un servizio di

backup unificato progettato per semplificare la creazione, la migrazione, il ripristino e l'eliminazione dei backup, fornendo al contempo report e audit migliorati. Attraverso l'integrazione con AWS Backup, è possibile utilizzare una soluzione di protezione dei dati centralizzata completamente gestita e basata su policy per creare backup immutabili e gestire centralmente la protezione dei dati delle applicazioni, compresi Timestream e altri servizi supportati da. AWS AWS Backup

Per utilizzare la funzionalità, devi dare il consenso alla protezione delle tue risorse Timestream. AWS Backup Le opzioni di attivazione si applicano all'account e alla AWS regione specifici, quindi potresti dover attivare più regioni utilizzando lo stesso account. Per ulteriori informazioni sul AWS Backup, consulta la Guida per AWS Backup gli sviluppatori.

La funzionalità di protezione dei dati disponibile AWS Backup include quanto segue.

Backup pianificati: puoi impostare backup pianificati regolarmente del tuo Timestream per LiveAnalytics le tabelle utilizzando piani di backup.

Copia tra account e più regioni: è possibile copiare automaticamente i backup in un altro archivio di backup in una AWS regione o account diverso, in modo da soddisfare i requisiti di protezione dei dati.

Archiviazione a freddo su più livelli: è possibile configurare i backup per implementare le regole del ciclo di vita per eliminare o trasferire i backup a sistemi di archiviazione più freddi. Questo può aiutarti a ottimizzare i costi di backup.

Tag: è possibile etichettare automaticamente i backup per scopi di fatturazione e allocazione dei costi.

Crittografia: i dati di backup vengono archiviati nel vault. AWS Backup Ciò consente di crittografare e proteggere i backup utilizzando una AWS KMS chiave indipendente dalla chiave di crittografia Timestream for Table. LiveAnalytics

Backup sicuri utilizzando il WORM modello: puoi utilizzare AWS Backup Vault Lock per abilitare un' write-once-read-manyimpostazione () per i tuoi backup. WORM Con AWS Backup Vault Lock, puoi aggiungere un ulteriore livello di difesa che protegge i backup da operazioni di eliminazione involontarie o dolose, modifiche ai periodi di conservazione dei backup e aggiornamenti delle impostazioni del ciclo di vita. Per ulteriori informazioni, consulta AWS Backup Vault Lock.

La funzionalità di protezione dei dati è disponibile in tutte le regioni. Per ulteriori informazioni sulla funzionalità, consulta la Guida per gli sviluppatori.AWS Backup

Backup e ripristino delle tabelle Timestream: come funziona

Puoi creare backup delle tue tabelle Amazon Timestream. In questa sezione viene fornita una panoramica di ciò che accade durante il processo di backup e ripristino.

Argomenti

Backup

Puoi utilizzare la funzionalità di backup su richiesta per creare backup completi del tuo Amazon Timestream per tabelle. LiveAnalytics In questa sezione viene fornita una panoramica di ciò che accade durante il processo di backup e ripristino.

Puoi creare un backup dei tuoi dati Timestream con una tabella di granularità. È possibile avviare un backup della tabella selezionata utilizzando la console Timestream o la console oppure. AWS Backup SDK CLI Il backup viene creato in modo asincrono e nel backup vengono inclusi tutti i dati della tabella fino all'ora di avvio del backup. Tuttavia, esiste la possibilità che alcuni dei dati inseriti nella tabella mentre il backup è in corso possano essere inclusi nel backup. Per proteggere i dati, è possibile creare un backup su richiesta una tantum o pianificare un backup ricorrente della tabella.

Mentre è in corso un backup, non è possibile effettuare le seguenti operazioni.

- Sospendere o annullare l'operazione di backup;
- Eliminare la tabella di origine del backup;
- Disabilitare i backup su una tabella se uno di questi è in corso.

Una volta configurato, AWS Backup fornisce pianificazioni di backup automatizzate, gestione della conservazione e gestione del ciclo di vita, eliminando la necessità di script personalizzati e processi manuali. Per ulteriori informazioni, consulta la Guida per gli sviluppatori AWS Backup

Tutti i timestream per i LiveAnalytics backup sono di natura incrementale, il che implica che il primo backup di una tabella è un backup completo e ogni backup successivo della stessa tabella è un backup incrementale, in cui vengono copiate solo le modifiche ai dati dall'ultimo backup. Poiché i dati in Timestream for LiveAnalytics sono archiviati in una raccolta di partizioni, tutte le partizioni modificate a causa dell'acquisizione di nuovi dati o degli aggiornamenti dei dati esistenti dall'ultimo backup vengono copiate durante i backup successivi.

Se utilizzi Timestream per LiveAnalytics console, i backup creati per tutte le risorse dell'account sono elencati nella scheda Backup. Inoltre, i backup sono elencati anche nei dettagli della Tabella.

Ripristini

È possibile ripristinare una tabella dal Timestream per LiveAnalytics console, o AWS Backup console, SDK o. AWS CLI È possibile ripristinare tutti i dati dal backup o configurare le impostazioni di conservazione delle tabelle per ripristinare determinati dati. Quando si avvia un ripristino, è possibile configurare le seguenti impostazioni della tabella.

- Database Name (Nome database)
- Nome tabella
- Conservazione dell'archivio di memoria
- Conservazione magnetica dell'archivio
- Abilita le scritture con archiviazione magnetica
- Posizione dei log di errore S3 (opzionale)
- IAMruolo che AWS Backup assumerà durante il ripristino del backup

Le configurazioni precedenti sono indipendenti dalla tabella di origine. Per ripristinare tutti i dati del backup, si consiglia di configurare le nuove impostazioni della tabella in modo che la somma del periodo di conservazione dell'archivio di memoria e del periodo di conservazione dell'archivio magnetico sia maggiore della differenza tra il timestamp più vecchio e quello attuale. Quando si seleziona un backup incrementale da ripristinare, vengono ripristinati tutti i dati (incrementali + dati completi sottostanti). Una volta completato il ripristino, la tabella è in stato attivo ed è possibile eseguire operazioni di inserimento e/o interrogazione sulla tabella ripristinata. Tuttavia, non è possibile eseguire queste operazioni mentre il ripristino è in corso. Una volta ripristinata, la tabella è simile a qualsiasi altra tabella del tuo account.

Example Ripristina tutti i dati da un backup

Questo esempio ha i seguenti presupposti.

Timestamp più vecchio: `August 1, 2021 0:00:00`

- Adesso — `November 9, 2022 0:00:00`

Per ripristinare tutti i dati da un backup, inserisci e confronta i valori come segue.

1. Inserisci Memory store retention e Magnetic store retention. Ad esempio, assumete questi valori.
   - Conservazione dell'archivio di memoria: -12 ore
   - Conservazione magnetica: 500 giorni
2. Calcola la somma della conservazione dell'archivio di memoria e della conservazione dell'archivio magnetico.

```
12 hours + (500 * 24 hours) =
12 hours + 12,000 hours =
12,012 hours
```

3. Trova la differenza tra il timestamp più vecchio e quello attuale.

```
November 9, 2022 0:00:00 - August 1, 2021 0:00:00 =
465 days =
465 * 24 hours =
11,160 hours
```

4. Assicurati che la somma dei valori di conservazione nel secondo passaggio sia maggiore della differenza di tempo nel terzo passaggio. Se necessario, regolate i tempi di conservazione.

```
12,012 > 11,160
true
```

Example Ripristina dati selezionati da un backup

Questo esempio si basa sul seguente presupposto.

- Ora — `November 9, 2022 0:00:00`

Per ripristinare solo alcuni dati da un backup, inserisci e confronta i valori come segue.

1. Determina il primo timestamp richiesto. Ad esempio, supponiamo. `December 4, 2021 0:00:00`
2. Trova la differenza tra il primo timestamp richiesto e quello attuale.

```
November 9, 2022 0:00:00 - December 4, 2021 0:00:00 =
340 days =
340 * 24 hours =
8,160 hours
```

3. Inserisci il valore desiderato per la conservazione dell'archivio di memoria. Ad esempio, inserire 12 ore.
4. Sottrai il valore dalla differenza nel secondo passaggio.

```
8,160 hours - 12 hours =
8148 hours
```

5. Inserisci quel valore per Magnetic store retention.

Puoi copiare un backup del tuo Timestream per i dati della LiveAnalytics tabella in un'altra AWS regione e quindi ripristinarlo in quella nuova regione. Puoi copiare e quindi ripristinare i backup tra regioni AWS commerciali e regioni AWS GovCloud (Stati Uniti). I prezzi sono calcolati solo in base ai dati che copi dalla regione di origine e dai dati che ripristini in una nuova tabella nella regione di destinazione.

Una volta ripristinata la tabella, è necessario impostare manualmente quanto segue sulla tabella ripristinata.

- AWS Politiche di Identity and Access Management (IAM)
- Tag
- Interrogazioni pianificate

I tempi di ripristino sono direttamente correlati alla configurazione delle tabelle. Questi includono la dimensione delle tabelle, il numero di partizioni sottostanti, la quantità di dati ripristinati nell'archivio di memoria e altre variabili. Una procedura ottimale per la pianificazione del disaster recovery consiste nel documentare regolarmente i tempi medi di completamento del ripristino e stabilire in che modo tali tempi influiscano sul Recovery Time Objective complessivo (RTO).

Tutte le operazioni e la console di backup e API ripristino vengono acquisite e registrate AWS CloudTrail per la registrazione, il monitoraggio continuo e il controllo.

Creazione di backup delle tabelle Amazon Timestream

Questa sezione descrive come abilitare AWS Backup e creare backup su richiesta e pianificati per Amazon Timestream.

Argomenti

- Attivazione della protezione di Timestream AWS Backup per i dati LiveAnalytics

- Creazione di backup on demand
- Backup pianificati

Attivazione della protezione di Timestream AWS Backup per i dati LiveAnalytics

È necessario abilitarlo AWS Backup per utilizzarlo con Timestream for. LiveAnalytics

Per abilitarlo AWS Backup in Timestream per LiveAnalytics console, procedi nel seguente modo.

1. Accedi alla console di AWS gestione.
2. Nella parte superiore della pagina Timestream for LiveAnalytics dashboard viene visualizzato un banner pop-up per consentire il supporto di Timestream AWS Backup per i dati. LiveAnalytics Altrimenti, dal pannello di navigazione, scegli Backup.
3. Nella finestra Backup, vedrai il banner da abilitare AWS Backup. Scegli Abilita .

   La protezione dei dati tramite data AWS Backup è ora disponibile per il tuo Timestream for LiveAnalytics tables.

Per attivarla AWS Backup, consulta la AWS Backup documentazione relativa all'attivazione tramite console e a livello di codice.

Se scegli di disabilitare la protezione AWS Backup del tuo Timestream per LiveAnalytics i dati dopo averli abilitati, accedi tramite la AWS Backup console e sposta l'interruttore verso sinistra.

Se non riesci ad abilitare o disabilitare le AWS Backup funzionalità, potrebbe essere necessario che l' AWS amministratore esegua tali azioni.

Creazione di backup on demand

Per creare un backup su richiesta di un Timestream for LiveAnalytics table, segui questi passaggi.

1. Accedi alla console di gestione.AWS
2. Nel riquadro di navigazione sul lato sinistro della console scegliere Backups (Backup).
3. Scegliere Create on-demand backup (Crea backup on demand).
4. Continua a selezionare le impostazioni nella finestra di backup.
5. È possibile creare subito un backup, avviarlo immediatamente o selezionare una finestra di backup per avviare il backup.

6. Seleziona la politica di gestione del ciclo di vita del tuo backup. È possibile trasferire i dati di backup in celle frigorifere, dove è necessario conservare il backup per un minimo di 90 giorni. È possibile impostare il periodo di conservazione richiesto per il backup. È possibile selezionare un archivio esistente oppure selezionare crea un nuovo archivio di backup per accedere alla AWS Backup console e creare un nuovo archivio di backup. <documentation link on creating a new backup vault here>
7. Seleziona il ruolo appropriato. IAM
8. Se si desidera assegnare uno o più tag al proprio backup on demand, immettere una chiave e, facoltativamente, un valore, quindi scegliere Aggiungi tag.
9. Scegli di creare un backup su richiesta. Verrai indirizzato alla pagina Backup, dove vedrai un elenco di lavori.
10. Scegliere ID del lavoro di Backup per la risorsa scelta per il backup per visualizzare i dettagli del lavoro.

Backup pianificati

Per pianificare un backup, consulta Creare un backup pianificato.

Ripristino di un backup di una tabella Amazon Timestream

Questa sezione descrive come ripristinare un backup di una tabella Amazon Timestream.

Argomenti

- Ripristino di un Timestream per una tabella da LiveAnalytics AWS Backup
- Ripristino di un Timestream per la LiveAnalytics tabella in un'altra regione o account

Ripristino di un Timestream per una tabella da LiveAnalytics AWS Backup

Per ripristinare Timestream for LiveAnalytics table dall' AWS Backup utilizzo di Timestream per console, segui questi passaggi. LiveAnalytics

1. Accedi alla console di gestione.AWS
2. Nel riquadro di navigazione sul lato sinistro della console scegliere Backups (Backup).
3. Per ripristinare una risorsa, scegli il pulsante di opzione accanto all'ID del punto di ripristino della risorsa. Nell'angolo superiore destro del riquadro, scegliere Ripristina.

4. Immettete le impostazioni di configurazione della tabella, vale a dire il nome del database e il nome della tabella. Si noti che il nome della tabella ripristinata deve essere diverso dal nome della tabella di origine originale.
5. Configura le impostazioni di conservazione della memoria e dell'archivio magnetico.
6. Per il ruolo di ripristino, scegli il IAM ruolo che AWS Backup assumerai per questo ripristino.
7. Scegli Restore backup (Ripristina backup). Un messaggio nella parte superiore della pagina fornisce informazioni sul lavoro di ripristino.

> **Note**
>
> Ti verrà addebitato il costo del ripristino dell'intero backup indipendentemente dalla memoria configurata e dai periodi di conservazione dell'archivio magnetico. Tuttavia, una volta completato il ripristino, la tabella ripristinata conterrà solo i dati entro i periodi di conservazione configurati.

Ripristino di un Timestream per la LiveAnalytics tabella in un'altra regione o account

Per ripristinare un Timestream per la LiveAnalytics tabella in un'altra regione o account, dovrai prima copiare il backup in quella nuova regione o account. Per copiare in un altro account, tale account deve prima concederti l'autorizzazione. Dopo aver copiato il Timestream per il LiveAnalytics backup nella nuova regione o account, è possibile ripristinarlo con la procedura descritta nella sezione precedente.

Copiare un backup di una tabella Amazon Timestream

Puoi creare una copia di un backup corrente. Puoi copiare i backup su più AWS account o AWS regioni su richiesta o automaticamente come parte di un piano di backup pianificato. La replica tra regioni è particolarmente utile se hai requisiti di continuità aziendale o di conformità per archiviare i backup a una distanza minima dai dati di produzione.

I backup tra account sono utili per copiare in modo sicuro i backup su uno o più account AWS della tua organizzazione per motivi operativi o di sicurezza. Se il backup originale viene eliminato inavvertitamente, puoi copiare il backup dall'account di destinazione all'account di origine e quindi avviare il ripristino. Prima di eseguire questa operazione, è necessario disporre di due account che appartengono alla stessa organizzazione nel servizio Organizations e le autorizzazioni necessarie

per gli account. Quando si copia un backup incrementale in un altro account o regione, viene copiato anche il backup completo associato.

Le copie ereditano la configurazione del backup di origine, a meno che non venga specificato diversamente. Esiste tuttavia un'eccezione. Se si specifica che la nuova copia è impostata su «Mai», scadono. Con questa impostazione, la nuova copia eredita ancora la data di scadenza dell'origine. Se vuoi che la nuova copia di backup sia permanente, imposta i backup di origine in modo che non scadano mai o specifica che la nuova copia scadrà 100 anni dopo la sua creazione.

Per copiare un backup dalla console Timestream, segui questi passaggi.

1. Accedi alla console di AWS gestione.
2. Nel riquadro di navigazione sul lato sinistro della console scegliere Backups (Backup).
3. Scegli il pulsante di opzione accanto all'ID del punto di ripristino della risorsa. Nell'angolo in alto a destra del riquadro, seleziona Azioni e scegli Copia.
4. Seleziona Continua con il AWS backup e segui i passaggi per il backup su più account.

La copia dei backup su richiesta e pianificati tra account e regioni non è attualmente supportata a livello nativo in Timestream per LiveAnalytics console e devi accedere per eseguire l'operazione. AWS Backup

Eliminazione di backup

Questa sezione descrive come eliminare un backup di un Timestream for table. LiveAnalytics

Per eliminare un backup dalla console Timestream, segui questi passaggi.

1. Accedi alla console di AWS gestione.
2. Nel riquadro di navigazione sul lato sinistro della console scegliere Backups (Backup).
3. Scegli il pulsante di opzione accanto all'ID del punto di ripristino della risorsa. Nell'angolo in alto a destra del riquadro, seleziona Azioni e scegli Elimina.
4. Seleziona Continua con il AWS backup e segui i passaggi per l'eliminazione dei backup in Eliminazione dei backup.

> **Note**
>
> Quando elimini un backup incrementale, viene eliminato solo il backup incrementale e il backup completo sottostante non viene eliminato.

Quote e limiti

AWS Backup limita i backup a un backup simultaneo per risorsa. Pertanto, le richieste di backup aggiuntive pianificate o su richiesta per la risorsa vengono messe in coda e verranno avviate solo dopo il completamento del processo di backup esistente. Se il processo di backup non viene avviato o completato all'interno della finestra di backup, la richiesta ha esito negativo. Per ulteriori informazioni sui AWS Backup limiti, consulta AWS Backup Limits nella AWS Backup Developer Guide.

Quando si crea un backup, è possibile eseguire fino a quattro backup simultanei per account. Allo stesso modo, puoi eseguire un ripristino simultaneo per account. Quando si avviano più di quattro processi di backup contemporaneamente, vengono avviati solo quattro processi di backup e i processi rimanenti verranno ripetuti periodicamente. Una volta avviato, se il processo di backup non viene completato entro la durata della finestra di backup configurata, il processo di backup ha esito negativo. Se il processo di backup non riuscito è un backup su richiesta, è possibile riprovare il backup e, per i backup pianificati, il processo viene tentato secondo la pianificazione seguente.

Chiavi di partizione definite dal cliente

Amazon Timestream LiveAnalytics per chiavi di partizione definite dal cliente è una funzionalità di Timestream che consente ai clienti di definire le proprie chiavi di partizione LiveAnalytics per le proprie tabelle. Il partizionamento è una tecnica utilizzata per distribuire i dati su più unità di archiviazione fisiche, che consente un recupero dei dati più rapido ed efficiente. Con le chiavi di partizione definite dal cliente, i clienti possono creare uno schema di partizionamento che si allinea meglio ai modelli di query e ai casi d'uso.

Con Timestream per le chiavi di partizione LiveAnalytics definite dal cliente, i clienti possono scegliere un nome di dimensione come chiave di partizione per le proprie tabelle. Ciò consente una maggiore flessibilità nella definizione dello schema di partizionamento per i dati. Selezionando la chiave di partizione corretta, i clienti possono ottimizzare il modello di dati, migliorare le prestazioni delle query e ridurre la latenza delle query.

Argomenti

Utilizzo di chiavi di partizione definite dal cliente

Se disponi di uno schema di query ben definito con dimensioni di cardinalità elevate e richiedi una bassa latenza di query, un Timestream per la chiave di partizione LiveAnalytics definita dal cliente può essere uno strumento utile per migliorare il modello di dati. Ad esempio, se sei un'azienda di vendita al dettaglio che monitora le interazioni con i clienti sul tuo sito web, i modelli di accesso principali sarebbero probabilmente l'ID del cliente e il timestamp. Definendo l'ID cliente come chiave di partizione, i dati possono essere distribuiti in modo uniforme, riducendo la latenza e, in ultima analisi, migliorando l'esperienza utente.

Un altro esempio è il settore sanitario, dove i dispositivi indossabili raccolgono i dati dei sensori per tracciare i segni vitali dei pazienti. Il modello di accesso principale sarebbe l'ID del dispositivo e il timestamp, con una cardinalità elevata su entrambe le dimensioni. Definendo Device ID come chiave di partizione, è possibile ottimizzare l'esecuzione delle query e garantire prestazioni di query sostenute a lungo termine.

In sintesi, Timestream per le chiavi di partizione LiveAnalytics definite dal cliente è particolarmente utile quando si dispone di uno schema di query chiaro, dimensioni di cardinalità elevate e si richiede una bassa latenza per le query. Definendo una chiave di partizione in linea con il modello di query, è possibile ottimizzare l'esecuzione delle query e garantire prestazioni sostenute a lungo termine.

Guida introduttiva alle chiavi di partizione definite dal cliente

Dalla console, scegli Tabelle e crea una nuova tabella. Puoi anche utilizzare an SDK per accedere all'`CreateTable`azione per creare nuove tabelle che possono includere una chiave di partizione definita dal cliente.

Crea una tabella con una chiave di partizione di tipo di dimensione

È possibile utilizzare i seguenti frammenti di codice per creare una tabella con una chiave di partizione di tipo dimensione.

Java

```
public void createTableWithDimensionTypePartitionKeyExample() {
        System.out.println("Creating table");
        CreateTableRequest createTableRequest = new CreateTableRequest();
        createTableRequest.setDatabaseName(DATABASE_NAME);
        createTableRequest.setTableName(TABLE_NAME);
        final RetentionProperties retentionProperties = new RetentionProperties()
                .withMemoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                .withMagneticStoreRetentionPeriodInDays(CT_TTL_DAYS);
        createTableRequest.setRetentionProperties(retentionProperties);

        // Can specify enforcement level with OPTIONAL or REQUIRED
        final List<PartitionKey> partitionKeyWithDimensionAndOptionalEnforcement =
 Collections.singletonList(new PartitionKey()
            .withName(COMPOSITE_PARTITION_KEY_DIM_NAME)
            .withType(PartitionKeyType.DIMENSION)
            .withEnforcementInRecord(PartitionKeyEnforcementLevel.OPTIONAL));
        Schema schema = new Schema();

 schema.setCompositePartitionKey(partitionKeyWithDimensionAndOptionalEnforcement);
        createTableRequest.setSchema(schema);

        try {
            writeClient.createTable(createTableRequest);
            System.out.println("Table [" + TABLE_NAME + "] successfully created.");
        } catch (ConflictException e) {
            System.out.println("Table [" + TABLE_NAME + "] exists on database [" +
 DATABASE_NAME + "] . Skipping database creation");
        }
    }
```

Java v2

```
public void createTableWithDimensionTypePartitionKeyExample() {
        System.out.println("Creating table");
        final RetentionProperties retentionProperties =
 RetentionProperties.builder()
                .memoryStoreRetentionPeriodInHours(HT_TTL_HOURS)
                .magneticStoreRetentionPeriodInDays(CT_TTL_DAYS)
                .build();
        // Can specify enforcement level with OPTIONAL or REQUIRED
        final List<PartitionKey> partitionKeyWithDimensionAndOptionalEnforcement =
 Collections.singletonList(PartitionKey
                .builder()
                .name(COMPOSITE_PARTITION_KEY_DIM_NAME)
                .type(PartitionKeyType.DIMENSION)
                .enforcementInRecord(PartitionKeyEnforcementLevel.OPTIONAL)
                .build());
        final Schema schema = Schema.builder()

 .compositePartitionKey(partitionKeyWithDimensionAndOptionalEnforcement).build();
        final CreateTableRequest createTableRequest = CreateTableRequest.builder()
                .databaseName(DATABASE_NAME)
                .tableName(TABLE_NAME)
                .retentionProperties(retentionProperties)
                .schema(schema)
                .build();

        try {
            writeClient.createTable(createTableRequest);
            System.out.println("Table [" + TABLE_NAME + "] successfully created.");
        } catch (ConflictException e) {
            System.out.println("Table [" + TABLE_NAME + "] exists on database [" +
 DATABASE_NAME + "] . Skipping database creation");
        }
    }
```

Go v1

```
func createTableWithDimensionTypePartitionKeyExample(){
        // Can specify enforcement level with OPTIONAL or REQUIRED
        partitionKeyWithDimensionAndOptionalEnforcement :=
 []*timestreamwrite.PartitionKey{
                {
```

```
                    Name:                aws.String(CompositePartitionKeyDimName),
                    EnforcementInRecord: aws.String("OPTIONAL"),
                    Type:                aws.String("DIMENSION"),
                },
        }
        createTableInput := &timestreamwrite.CreateTableInput{
             DatabaseName: aws.String(*databaseName),
             TableName:    aws.String(*tableName),
             // Enable MagneticStoreWrite for Table
             MagneticStoreWriteProperties:
&timestreamwrite.MagneticStoreWriteProperties{
                 EnableMagneticStoreWrites: aws.Bool(true),
                 // Persist MagneticStoreWrite rejected records in S3
                 MagneticStoreRejectedDataLocation:
&timestreamwrite.MagneticStoreRejectedDataLocation{
                     S3Configuration: &timestreamwrite.S3Configuration{
                         BucketName:       aws.String("timestream-sample-bucket"),
                         ObjectKeyPrefix:  aws.String("TimeStreamCustomerSampleGo"),
                         EncryptionOption: aws.String("SSE_S3"),
                     },
                 },
             },
             Schema: &timestreamwrite.Schema{
                 CompositePartitionKey:
partitionKeyWithDimensionAndOptionalEnforcement,
             }
         }
         _, err := writeSvc.CreateTable(createTableInput)
    }
```

Go v2

```
 func (timestreamBuilder TimestreamBuilder)
CreateTableWithDimensionTypePartitionKeyExample() error {
        partitionKeyWithDimensionAndOptionalEnforcement := []types.PartitionKey{
            {
                Name:                aws.String(CompositePartitionKeyDimName),
                EnforcementInRecord: types.PartitionKeyEnforcementLevelOptional,
                Type:                types.PartitionKeyTypeDimension,
            },
        }
        _, err := timestreamBuilder.WriteSvc.CreateTable(context.TODO(),
&timestreamwrite.CreateTableInput{
```

```
            DatabaseName: aws.String(databaseName),
            TableName:    aws.String(tableName),
            MagneticStoreWriteProperties: &types.MagneticStoreWriteProperties{
                EnableMagneticStoreWrites: aws.Bool(true),
                // Persist MagneticStoreWrite rejected records in S3
                MagneticStoreRejectedDataLocation:
 &types.MagneticStoreRejectedDataLocation{
                    S3Configuration: &types.S3Configuration{
                        BucketName:       aws.String(s3BucketName),
                        EncryptionOption: "SSE_S3",
                    },
                },
            },
            Schema: &types.Schema{
                CompositePartitionKey:
 partitionKeyWithDimensionAndOptionalEnforcement,
            },
        })

        if err != nil {
            fmt.Println("Error:")
            fmt.Println(err)
        } else {
            fmt.Println("Create table is successful")
        }
        return err
    }
```

Python

```
def create_table_with_measure_name_type_partition_key(self):
        print("Creating table")
        retention_properties = {
            'MemoryStoreRetentionPeriodInHours': HT_TTL_HOURS,
            'MagneticStoreRetentionPeriodInDays': CT_TTL_DAYS
        }
        partitionKey_with_measure_name = [
            {'Type': 'MEASURE'}
        ]
        schema = {
            'CompositePartitionKey': partitionKey_with_measure_name
        }
        try:
```

```
            self.client.create_table(DatabaseName=DATABASE_NAME,
TableName=TABLE_NAME,
                                     RetentionProperties=retention_properties,
Schema=schema)
            print("Table [%s] successfully created." % TABLE_NAME)
        except self.client.exceptions.ConflictException:
            print("Table [%s] exists on database [%s]. Skipping table creation" % (
                TABLE_NAME, DATABASE_NAME))
        except Exception as err:
            print("Create table failed:", err)
```

Verifica della configurazione dello schema di partizionamento

È possibile verificare la configurazione di una tabella per lo schema di partizionamento in un paio di modi. Dalla console, scegli Database e scegli la tabella da controllare. Puoi anche usare un SDK per accedere all'`DescribeTable`azione.

Descrivi una tabella con una chiave di partizione

È possibile utilizzare i seguenti frammenti di codice per descrivere una tabella con una chiave di partizione.

Java

```
   public void describeTable() {
        System.out.println("Describing table");
        final DescribeTableRequest describeTableRequest = new
DescribeTableRequest();
        describeTableRequest.setDatabaseName(DATABASE_NAME);
        describeTableRequest.setTableName(TABLE_NAME);
        try {
            DescribeTableResult result =
amazonTimestreamWrite.describeTable(describeTableRequest);
            String tableId = result.getTable().getArn();
            System.out.println("Table " + TABLE_NAME + " has id " + tableId);
            // If table is created with composite partition key, it can be described
with
            //
System.out.println(result.getTable().getSchema().getCompositePartitionKey());
        } catch (final Exception e) {
            System.out.println("Table " + TABLE_NAME + " doesn't exist = " + e);
            throw e;
```

```
        }
    }
```

Di seguito è riportato un esempio di output.

1. La tabella ha una chiave di partizione del tipo di dimensione

```
[{Type: DIMENSION,Name: hostId,EnforcementInRecord: OPTIONAL}]
```

2. La tabella ha il nome della misura, il tipo di chiave di partizione

```
[{Type: MEASURE,}]
```

3. Ottenere la chiave di partizione composita da una tabella creata senza specificare la chiave di partizione composita

```
[{Type: MEASURE,}]
```

Java v2

```
 public void describeTable() {
       System.out.println("Describing table");
       final DescribeTableRequest describeTableRequest =
DescribeTableRequest.builder()
               .databaseName(DATABASE_NAME).tableName(TABLE_NAME).build();
       try {
           DescribeTableResponse response =
writeClient.describeTable(describeTableRequest);
           String tableId = response.table().arn();
           System.out.println("Table " + TABLE_NAME + " has id " + tableId);
           // If table is created with composite partition key, it can be described
with
           //
System.out.println(response.table().schema().compositePartitionKey());
       } catch (final Exception e) {
           System.out.println("Table " + TABLE_NAME + " doesn't exist = " + e);
           throw e;
       }
   }
```

Di seguito è riportato un esempio di output.

1. La tabella ha una chiave di partizione del tipo di dimensione

```
[PartitionKey(Type=DIMENSION, Name=hostId, EnforcementInRecord=OPTIONAL)]
```

2. La tabella ha il nome della misura, il tipo di chiave di partizione

```
[PartitionKey(Type=MEASURE)]
```

3. Verrà restituita la chiave di partizione composita da una tabella creata senza specificare la chiave di partizione composita

```
[PartitionKey(Type=MEASURE)]
```

Go v1

```
    <tablistentry>
     <tabname> Go </tabname>
     <tabcontent>
      <programlisting language="go"></programlisting>
     </tabcontent>
    </tablistentry>
```

Di seguito è riportato un esempio di output.

```
{
  Table: {
    Arn: "arn:aws:timestream:us-west-2:533139590831:database/devops/table/
host_metrics_dim_pk_1",
    CreationTime: 2023-05-31 01:52:00.511 +0000 UTC,
    DatabaseName: "devops",
    LastUpdatedTime: 2023-05-31 01:52:00.511 +0000 UTC,
    MagneticStoreWriteProperties: {
      EnableMagneticStoreWrites: true,
      MagneticStoreRejectedDataLocation: {
        S3Configuration: {
          BucketName: "timestream-sample-bucket-west",
          EncryptionOption: "SSE_S3",
          ObjectKeyPrefix: "TimeStreamCustomerSampleGo"
        }
      }
    },
```

```
    RetentionProperties: {
      MagneticStoreRetentionPeriodInDays: 73000,
      MemoryStoreRetentionPeriodInHours: 6
    },
    Schema: {
      CompositePartitionKey: [{
          EnforcementInRecord: "OPTIONAL",
          Name: "hostId",
          Type: "DIMENSION"
        }]
    },
    TableName: "host_metrics_dim_pk_1",
    TableStatus: "ACTIVE"
  }
}
```

Go v2

```
func (timestreamBuilder TimestreamBuilder) DescribeTable()
(*timestreamwrite.DescribeTableOutput, error) {
        describeTableInput := &timestreamwrite.DescribeTableInput{
            DatabaseName: aws.String(databaseName),
            TableName:    aws.String(tableName),
        }
       describeTableOutput, err :=
timestreamBuilder.WriteSvc.DescribeTable(context.TODO(), describeTableInput)

       if err != nil {
           fmt.Printf("Failed to describe table with Error: %s", err.Error())
       } else {
           fmt.Printf("Describe table is successful : %s\n",
JsonMarshalIgnoreError(*describeTableOutput))
           // If table is created with composite partition key, it will be included
in the output
       }

       return describeTableOutput, err
   }
```

Di seguito è riportato un esempio di output.

```
{
  "Table": {
```

```
    "Arn":"arn:aws:timestream:us-east-1:351861611069:database/cdpk-wr-db/table/
host_metrics_dim_pk",
    "CreationTime":"2023-05-31T22:36:10.66Z",
    "DatabaseName":"cdpk-wr-db",
    "LastUpdatedTime":"2023-05-31T22:36:10.66Z",
    "MagneticStoreWriteProperties":{
      "EnableMagneticStoreWrites":true,
      "MagneticStoreRejectedDataLocation":{
        "S3Configuration":{
          "BucketName":"error-configuration-sample-s3-bucket-cq8my",
          "EncryptionOption":"SSE_S3",
          "KmsKeyId":null,"ObjectKeyPrefix":null
        }
      }
    },
    "RetentionProperties":{
      "MagneticStoreRetentionPeriodInDays":73000,
      "MemoryStoreRetentionPeriodInHours":6
    },
    "Schema":{
      "CompositePartitionKey":[{
        "Type":"DIMENSION",
        "EnforcementInRecord":"OPTIONAL",
        "Name":"hostId"
      }]
    },
    "TableName":"host_metrics_dim_pk",
    "TableStatus":"ACTIVE"
  },
  "ResultMetadata":{}
}
```

Python

```
  def describe_table(self):
        print('Describing table')
        try:
            result = self.client.describe_table(DatabaseName=DATABASE_NAME,
 TableName=TABLE_NAME)
            print("Table [%s] has id [%s]" % (TABLE_NAME, result['Table']['Arn']))
            # If table is created with composite partition key, it can be described
 with
            # print(result['Table']['Schema'])
```

```
        except self.client.exceptions.ResourceNotFoundException:
            print("Table doesn't exist")
        except Exception as err:
            print("Describe table failed:", err)
```

Di seguito è riportato un esempio di output.

1. La tabella ha una chiave di partizione del tipo di dimensione

```
[{'CompositePartitionKey': [{'Type': 'DIMENSION', 'Name': 'hostId',
 'EnforcementInRecord': 'OPTIONAL'}]}]
```

2. La tabella ha il nome della misura, il tipo di chiave di partizione

```
[{'CompositePartitionKey': [{'Type': 'MEASURE'}]}]
```

3. Ottenere la chiave di partizione composita da una tabella creata senza specificare la chiave di partizione composita

```
[{'CompositePartitionKey': [{'Type': 'MEASURE'}]}]
```

Aggiornamento della configurazione dello schema di partizionamento

È possibile aggiornare la configurazione della tabella per lo schema di partizionamento con un'azione SDK with access the. `UpdateTable`

Aggiorna una tabella con una chiave di partizione

È possibile utilizzare i seguenti frammenti di codice per aggiornare una tabella con una chiave di partizione.

Java

```
    public void updateTableCompositePartitionKeyEnforcement() {
        System.out.println("Updating table");

        UpdateTableRequest updateTableRequest = new UpdateTableRequest();
        updateTableRequest.setDatabaseName(DATABASE_NAME);
        updateTableRequest.setTableName(TABLE_NAME);
```

```
       // Can update enforcement level for dimension type partition key with
OPTIONAL or REQUIRED enforcement
       final List<PartitionKey> partitionKeyWithDimensionAndRequiredEnforcement =
Collections.singletonList(new PartitionKey()
           .withName(COMPOSITE_PARTITION_KEY_DIM_NAME)
           .withType(PartitionKeyType.DIMENSION)
           .withEnforcementInRecord(PartitionKeyEnforcementLevel.REQUIRED));
       Schema schema = new Schema();

schema.setCompositePartitionKey(partitionKeyWithDimensionAndRequiredEnforcement);
       updateTableRequest.withSchema(schema);

       writeClient.updateTable(updateTableRequest);
       System.out.println("Table updated");
```

Java v2

```
   public void updateTableCompositePartitionKeyEnforcement() {
       System.out.println("Updating table");
       // Can update enforcement level for dimension type partition key with
OPTIONAL or REQUIRED enforcement
       final List<PartitionKey> partitionKeyWithDimensionAndRequiredEnforcement =
Collections.singletonList(PartitionKey
           .builder()
           .name(COMPOSITE_PARTITION_KEY_DIM_NAME)
           .type(PartitionKeyType.DIMENSION)
           .enforcementInRecord(PartitionKeyEnforcementLevel.REQUIRED)
           .build());
       final Schema schema = Schema.builder()

.compositePartitionKey(partitionKeyWithDimensionAndRequiredEnforcement).build();
       final UpdateTableRequest updateTableRequest = UpdateTableRequest.builder()

.databaseName(DATABASE_NAME).tableName(TABLE_NAME).schema(schema).build();

       writeClient.updateTable(updateTableRequest);
       System.out.println("Table updated");
```

Go v1

```
// Update table partition key enforcement attribute
   updateTableInput := &timestreamwrite.UpdateTableInput{
        DatabaseName: aws.String(*databaseName),
```

```
        TableName:    aws.String(*tableName),
        // Can update enforcement level for dimension type partition key with
OPTIONAL or REQUIRED enforcement
        Schema: &timestreamwrite.Schema{
            CompositePartitionKey: []*timestreamwrite.PartitionKey{
                {
                        Name:
aws.String(CompositePartitionKeyDimName),
                        EnforcementInRecord: aws.String("REQUIRED"),
                        Type:               aws.String("DIMENSION"),
                },
            }},
    }
    updateTableOutput, err := writeSvc.UpdateTable(updateTableInput)
        if err != nil {
            fmt.Println("Error:")
            fmt.Println(err)
        } else {
            fmt.Println("Update table is successful, below is the output:")
            fmt.Println(updateTableOutput)
        }
```

Go v2

```
// Update table partition key enforcement attribute
        updateTableInput := &timestreamwrite.UpdateTableInput{
            DatabaseName: aws.String(*databaseName),
            TableName:    aws.String(*tableName),
            // Can update enforcement level for dimension type partition key with
OPTIONAL or REQUIRED enforcement
            Schema: &types.Schema{
                CompositePartitionKey: []types.PartitionKey{
                    {
                        Name:
aws.String(CompositePartitionKeyDimName),
                        EnforcementInRecord:
types.PartitionKeyEnforcementLevelRequired,
                        Type:               types.PartitionKeyTypeDimension,
                    },
                }},
        }
        updateTableOutput, err :=
timestreamBuilder.WriteSvc.UpdateTable(context.TODO(), updateTableInput)
```

```
        if err != nil {
            fmt.Println("Error:")
            fmt.Println(err)
        } else {
            fmt.Println("Update table is successful, below is the output:")
            fmt.Println(updateTableOutput)
        }
```

Python

```
   def update_table(self):
       print('Updating table')
       try:
           # Can update enforcement level for dimension type partition key with
OPTIONAL or REQUIRED enforcement
           partition_key_with_dimension_and_required_enforcement = [
               {
                   'Type': 'DIMENSION',
                   'Name': COMPOSITE_PARTITION_KEY_DIM_NAME,
                   'EnforcementInRecord': 'REQUIRED'
               }
           ]
           schema = {
               'CompositePartitionKey':
partition_key_with_dimension_and_required_enforcement
           }
           self.client.update_table(DatabaseName=DATABASE_NAME,
TableName=TABLE_NAME,
                                    Schema=schema)
           print('Table updated.')
       except Exception as err:
           print('Update table failed:', err)
```

Vantaggi delle chiavi di partizione definite dal cliente

Prestazioni di query migliorate: le chiavi di partizione definite dal cliente consentono di ottimizzare l'esecuzione delle query e migliorare le prestazioni complessive delle query. Definendo chiavi di partizione in linea con i modelli di query, è possibile ridurre al minimo la scansione dei dati e ottimizzare l'eliminazione dei dati, con conseguente riduzione della latenza delle query.

Migliore prevedibilità delle prestazioni a lungo termine: le chiavi di partizione definite dal cliente consentono ai clienti di distribuire i dati in modo uniforme tra le partizioni, migliorando l'efficienza della gestione dei dati. Ciò garantirà che le prestazioni delle query rimangano stabili man mano che i dati archiviati aumenteranno nel tempo.

Limitazioni delle chiavi di partizione definite dal cliente

Come Timestream per LiveAnalytics l'utente, è importante tenere a mente le limitazioni relative alla chiave di partizione del cliente. Innanzitutto, richiede una buona comprensione del carico di lavoro e dei modelli di query. Ciò significa che è necessario avere un'idea chiara di quali dimensioni vengono utilizzate più frequentemente come condizioni di filtro principali nelle query e disporre di un'elevata cardinalità per utilizzare al meglio le chiavi di partizione.

In secondo luogo, le chiavi di partizione devono essere definite al momento della creazione della tabella e non possono essere aggiunte alle tabelle esistenti. Ciò significa che è necessario considerare attentamente la strategia di partizionamento prima di creare una tabella per assicurarsi che sia in linea con le esigenze aziendali.

Infine, è importante notare che una volta creata la tabella, non è possibile modificare la chiave di partizione in seguito. Ciò significa che è necessario testare e valutare a fondo la strategia di partizionamento prima di impegnarsi. Tenendo presenti queste limitazioni, la chiave di partizione definita dal cliente di Timestream può migliorare notevolmente le prestazioni delle query e la soddisfazione a lungo termine.

Chiavi di partizione definite dal cliente e dimensioni a bassa cardinalità

Se si decide di utilizzare una chiave di partizione con una cardinalità molto bassa, ad esempio una regione o uno stato specifici, è importante notare che i dati di altre entità come`customerID`, e altre`ProductCategory`, potrebbero finire per essere distribuiti su troppe partizioni a volte con pochi o nessun dato presente. Ciò può comportare un'esecuzione inefficiente delle query e una riduzione delle prestazioni.

Per evitare ciò, ti consigliamo di scegliere dimensioni che non solo rientrino nella tua condizione di filtro chiave, ma abbiano una cardinalità più elevata. Ciò contribuirà a garantire che i dati siano distribuiti uniformemente tra le partizioni e a migliorare le prestazioni delle query.

Creazione di chiavi di partizione per tabelle esistenti

Se disponi già di tabelle in Timestream LiveAnalytics e desideri utilizzare chiavi di partizione definite dal cliente, dovrai migrare i dati in una nuova tabella con la definizione dello schema di

partizionamento desiderata. Ciò può essere fatto utilizzando contemporaneamente l'esportazione in S3 e il caricamento in batch, il che comporta l'esportazione dei dati dalla tabella esistente in S3, la modifica dei dati per includere la chiave di partizione (se necessario) e l'aggiunta di intestazioni ai CSV file, quindi l'importazione dei dati in una nuova tabella con lo schema di partizionamento desiderato definito. Tieni presente che questo metodo può richiedere molto tempo e denaro, soprattutto per tabelle di grandi dimensioni.

In alternativa, è possibile utilizzare le query pianificate per migrare i dati in una nuova tabella con lo schema di partizionamento desiderato. Questo metodo prevede la creazione di una query pianificata che legge dalla tabella esistente e scrive nella nuova tabella. L'interrogazione pianificata può essere impostata per essere eseguita regolarmente fino alla migrazione di tutti i dati. Tieni presente che ti verranno addebitati i costi per la lettura e la scrittura dei dati durante il processo di migrazione.

Timestream per la convalida LiveAnalytics dello schema con chiavi di partizione composite personalizzate

La convalida dello schema in Timestream LiveAnalytics aiuta a garantire che i dati inseriti nel database siano conformi allo schema specificato, riducendo al minimo gli errori di inserimento e migliorando la qualità dei dati. In particolare, la convalida dello schema è particolarmente utile quando si adotta una chiave di partizione definita dal cliente con l'obiettivo di ottimizzare le prestazioni delle query.

Cos'è Timestream per la convalida dello schema con chiavi di partizione definite dal cliente? LiveAnalytics

Timestream per la convalida LiveAnalytics dello schema è una funzionalità che convalida i dati inseriti in un Timestream for table in base a uno schema predefinito. LiveAnalytics Questo schema definisce il modello di dati, inclusa la chiave di partizione, i tipi di dati e i vincoli per i record da inserire.

Quando si utilizza una chiave di partizione definita dal cliente, la convalida dello schema diventa ancora più cruciale. Le chiavi di partizione consentono di specificare una chiave di partizione, che determina il modo in cui i dati vengono archiviati in Timestream for. LiveAnalytics Convalidando i dati in entrata rispetto allo schema con una chiave di partizione personalizzata, è possibile imporre la coerenza dei dati, rilevare tempestivamente gli errori e migliorare la qualità complessiva dei dati archiviati in Timestream for. LiveAnalytics

Come utilizzare Timestream per la convalida dello schema con chiavi di partizione composite personalizzate LiveAnalytics

Per utilizzare Timestream per la convalida LiveAnalytics dello schema con chiavi di partizione composite personalizzate, procedi nel seguente modo:

Pensate a come saranno i vostri modelli di query: per scegliere e definire correttamente lo schema per la LiveAnalytics tabella Timestream for, dovreste iniziare con i requisiti di query.

Specificare chiavi di partizione composite personalizzate: quando si crea la tabella, specificare una chiave di partizione personalizzata. Questa chiave determina l'attributo che verrà utilizzato per partizionare i dati della tabella. È possibile scegliere tra chiavi di dimensione e chiavi di misura per il partizionamento. Una chiave di dimensione partiziona i dati in base al nome di una dimensione, mentre una chiave di misura partiziona i dati in base al nome della misura.

Imposta i livelli di applicazione: per garantire il corretto partizionamento dei dati e i vantaggi che ne derivano, Amazon Timestream LiveAnalytics for ti consente di impostare i livelli di applicazione per ogni chiave di partizione del tuo schema. Il livello di applicazione determina se la dimensione della chiave di partizione è obbligatoria o facoltativa durante l'importazione dei record. È possibile scegliere tra due opzioni:`REQUIRED`, il che significa che la chiave di partizione deve essere presente nel record importato e`OPTIONAL`, il che significa che la chiave di partizione non deve essere presente. Si consiglia di utilizzare il livello di `REQUIRED` applicazione quando si utilizza una partizione definita dal cliente per garantire che i dati siano partizionati correttamente e ottenere tutti i vantaggi di questa funzionalità. Inoltre, è possibile modificare la configurazione del livello di applicazione in qualsiasi momento dopo la creazione dello schema per adattarla ai requisiti di inserimento dei dati.

Inserisci dati: quando si inseriscono dati nella LiveAnalytics tabella Timestream for, il processo di convalida dello schema verificherà i record rispetto allo schema definito con chiavi di partizione composite personalizzate. Se i record non aderiscono allo schema, Timestream for restituirà un errore di convalida. LiveAnalytics

Gestisci gli errori di convalida: in caso di errori di convalida, Timestream for LiveAnalytics restituirà un `ValidationException` o un`RejectedRecordsException`, a seconda del tipo di errore. Assicurati di gestire queste eccezioni nell'applicazione e di adottare le misure appropriate, ad esempio correggere i record errati e riprovare l'inserimento.

Aggiorna i livelli di applicazione: se necessario, puoi aggiornare il livello di applicazione delle chiavi di partizione dopo la creazione della tabella utilizzando l'azione. `UpdateTable` Tuttavia, è importante

notare che alcuni aspetti della configurazione delle chiavi di partizione, come il nome e il tipo, non possono essere modificati dopo la creazione della tabella. Se si modifica il livello di applicazione da `REQUIRED` a`OPTIONAL`, tutti i record verranno accettati indipendentemente dalla presenza dell'attributo selezionato come chiave di partizione definita dal cliente. Al contrario, se si modifica il livello di applicazione da `OPTIONAL` a`REQUIRED`, è possibile che si verifichino errori di scrittura 4xx per i record che non soddisfano questa condizione. Pertanto, è essenziale scegliere il livello di applicazione appropriato per il caso d'uso al momento della creazione della tabella, in base ai requisiti di partizionamento dei dati.

Quando utilizzare Timestream per la convalida LiveAnalytics dello schema con chiavi di partizione composite personalizzate

Il timestream per la convalida LiveAnalytics dello schema con chiavi di partizione composite personalizzate deve essere utilizzato in scenari in cui la coerenza dei dati, la qualità e il partizionamento ottimizzato sono fondamentali. Applicando uno schema durante l'inserimento dei dati, è possibile prevenire errori e incongruenze che potrebbero portare a analisi errate o alla perdita di informazioni preziose.

Interazione con i processi di caricamento in batch

Quando si configura un processo di caricamento in batch per importare dati in una tabella con una chiave di partizione definita dal cliente, esistono alcuni scenari che potrebbero influire sul processo:

1. Se il livello di applicazione è impostato su`OPTIONAL`, verrà visualizzato un avviso sulla console durante il flusso di creazione se la chiave di partizione non viene mappata durante la configurazione del processo. Questo avviso non verrà visualizzato quando si utilizza o. API CLI
2. Se il livello di applicazione è impostato su`REQUIRED`, la creazione del lavoro verrà rifiutata a meno che la chiave di partizione non sia mappata su una colonna di dati di origine.
3. Se il livello di applicazione viene impostato su `REQUIRED` dopo la creazione del processo, il processo continuerà a essere eseguito, ma tutti i record che non dispongono della mappatura corretta per la chiave di partizione verranno rifiutati con un errore 4xx.

Interazione con una query pianificata

Quando si configura un processo di interrogazione pianificato per il calcolo e l'archiviazione di aggregati, rollup e altre forme di dati preelaborati in una tabella con una chiave di partizione definita dal cliente, esistono alcuni scenari che potrebbero influire sul processo:

1. Se il livello di applicazione è impostato su`OPTIONAL`, verrà visualizzato un avviso se la chiave di partizione non viene mappata durante la configurazione del processo. Questo avviso non verrà visualizzato quando si utilizza o. API CLI
2. Se il livello di applicazione è impostato su`REQUIRED`, la creazione del lavoro verrà rifiutata a meno che la chiave di partizione non sia mappata su una colonna di dati di origine.
3. Se il livello di applicazione viene impostato su `REQUIRED` dopo la creazione del lavoro e i risultati della query pianificata non contengono la dimensione della chiave di partizione, tutte le iterazioni successive del lavoro avranno esito negativo.

Aggiunta di tag ed etichette alle risorse

Puoi etichettare Amazon Timestream LiveAnalytics per le risorse utilizzando i tag. I tag consentono di classificare le risorse in diversi modi, ad esempio per scopo, proprietario, ambiente o altri criteri. I tag consentono di eseguire le seguenti operazioni:

- identificare rapidamente una risorsa in base ai tag a questa assegnati.
- Visualizza le AWS fatture suddivise per tag.

Il tagging è supportato da AWS servizi come Amazon Elastic Compute Cloud (AmazonEC2), Amazon Simple Storage Service (Amazon S3), Timestream for e molti altri. LiveAnalytics Un tagging efficiente può fornire informazioni dettagliate sui costi abilitando la creazione di report su servizi che recano tag specifici.

Per iniziare a utilizzare il tagging, procedi come segue:

1. Comprendi le restrizioni relative ai tag.
2. Crea tag utilizzando le operazioni di tagging.

Infine, è buona norma seguire strategie di tagging ottimali. Per informazioni, consulta Strategie di tagging di AWS.

Restrizioni di tagging

Ogni tag consiste di una chiave e di un valore, entrambi personalizzabili. Le restrizioni si applicano come segue:

- Ogni Timestream per LiveAnalytics tabella può avere solo un tag con la stessa chiave. Se provi ad aggiungere un tag esistente, il valore del tag esistente viene aggiornato al nuovo valore.
- Un valore funge da descrittore all'interno di una categoria di tag. In Timestream LiveAnalytics il valore non può essere vuoto o nullo.
- I valori e le chiavi dei tag rispettano la distinzione tra maiuscole e minuscole.
- La lunghezza massima della chiave è di 128 caratteri Unicode.
- La lunghezza massima del valore è di 256 caratteri Unicode.
- i caratteri consentiti sono lettere, spazi vuoti, numeri e i seguenti caratteri speciali: `+ - = . _ : /`
- Il numero massimo di tag per risorsa è 50.
- AWS-ai nomi e ai valori dei tag assegnati viene assegnato automaticamente il `aws:` prefisso, che non è possibile assegnare. AWS-i nomi dei tag assegnati non vengono conteggiati ai fini del limite di 50 tag. I nomi dei tag assegnati dall'utente hanno il prefisso `user:` nel report di allocazione dei costi;
- Non puoi retrodatare l'applicazione di un tag.

Operazioni di etichettatura

Puoi aggiungere, elencare, modificare o eliminare tag per database e tabelle utilizzando Amazon Timestream LiveAnalytics per console, linguaggio di interrogazione o AWS Command Line Interface ().AWS CLI

Argomenti

- Aggiungere tag a database e tabelle nuovi o esistenti utilizzando la console

Aggiungere tag a database e tabelle nuovi o esistenti utilizzando la console

Puoi utilizzare Timestream per LiveAnalytics console per aggiungere tag a nuovi database, tabelle e query pianificate al momento della creazione. Puoi anche aggiungere, modificare o eliminare tag per tabelle esistenti.

Per etichettare i database durante la loro creazione (console)

1. Apri la console Timestream in /timestream. https://console.aws.amazon.com
2. Nel riquadro di navigazione, scegli Database, quindi scegli Crea database.

3. Nella pagina Crea database, fornisci un nome per il database. Inserisci una chiave e un valore per il tag, quindi scegli Aggiungi nuovo tag.
4. Scegliere Crea database.

Per etichettare le tabelle durante la creazione (console)

1. Apri la console Timestream in /timestream. https://console.aws.amazon.com
2. Nel pannello di navigazione, scegli Tabelle, quindi seleziona Crea tabella.
3. Nella pagina Crea timestream per la tabella, fornisci un LiveAnalytics nome per la tabella. Immettete una chiave e un valore per il tag e scegliete Aggiungi nuovo tag.
4. Scegliere Create table (Crea tabella).

Per etichettare le query pianificate durante la creazione (console)

1. Apri la console Timestream in /timestream. https://console.aws.amazon.com
2. Nel riquadro di navigazione, scegli Interrogazioni pianificate, quindi scegli Crea interrogazione pianificata.
3. Nel passaggio 3. Configura la pagina delle impostazioni della query, scegli Aggiungi nuovo tag. Digitare una Key (Chiave) e un Value (Valore) per il tag. Scegli Aggiungi nuovo tag per aggiungere altri tag.
4. Scegli Next (Successivo).

Per assegnare tag alle risorse esistenti (console)

1. Apri la console Timestream in /timestream. https://console.aws.amazon.com
2. Nel riquadro di navigazione, scegli Database, Tabelle o Interrogazioni pianificate.
3. Scegli un database o una tabella nell'elenco. Quindi scegli Gestisci tag per aggiungere, modificare o eliminare i tag.

Per informazioni sulla struttura dei tag, consulta Restrizioni di tagging.

Sicurezza in Timestream per LiveAnalytics

La sicurezza del cloud AWS è la massima priorità. In qualità di AWS cliente, puoi beneficiare di un data center e di un'architettura di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra AWS te e te. Il modello di responsabilità condivisa descrive questo come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud: AWS è responsabile della protezione dell'infrastruttura che gestisce AWS i servizi nel AWS cloud. AWS ti fornisce anche servizi che puoi utilizzare in modo sicuro. L'efficacia della nostra sicurezza è regolarmente testata e verificata da revisori di terze parti come parte dei programmi di conformitàAWS. Per ulteriori informazioni sui programmi di conformità applicabili a Timestream for LiveAnalytics, consulta AWS Services in Scope by Compliance Program.
- Sicurezza nel cloud: la tua responsabilità è determinata dal AWS servizio che utilizzi. L'utente è anche responsabile per altri fattori, tra cui la riservatezza dei dati, i requisiti dell'azienda, nonché le leggi e le normative applicabili.

Questa documentazione ti aiuterà a capire come applicare il modello di responsabilità condivisa quando usi Timestream for. LiveAnalytics I seguenti argomenti mostrano come configurare Timestream per soddisfare gli obiettivi LiveAnalytics di sicurezza e conformità. Imparerai anche come utilizzare altri AWS servizi che possono aiutarti a monitorare e proteggere le risorse di Timestream. LiveAnalytics

Argomenti

Protezione dei dati in Timestream per LiveAnalytics

Il modello di responsabilità AWS condivisa modello si applica alla protezione dei dati in Amazon Timestream Live Analytics. Come descritto in questo modello, AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutti i. Cloud AWS L'utente è responsabile del controllo dei contenuti ospitati su questa infrastruttura. L'utente è inoltre responsabile della configurazione della protezione e delle attività di gestione per i Servizi AWS utilizzati. Per ulteriori informazioni sulla privacy dei dati, consulta la sezione Privacy dei dati FAQ. Per informazioni sulla protezione dei dati in Europa, consulta il Modello di responsabilitàAWS condivisa e GDPR il post sul blog sulla AWS sicurezza.

Ai fini della protezione dei dati, ti consigliamo di proteggere Account AWS le credenziali e di configurare i singoli utenti con AWS IAM Identity Center o AWS Identity and Access Management (IAM). In tal modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere i suoi compiti. Ti suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Utilizza l'autenticazione a più fattori (MFA) con ogni account.
- UsaSSL/TLSper comunicare con AWS le risorse. Richiediamo TLS 1.2 e consigliamo TLS 1.3.
- Configurazione API e registrazione delle attività degli utenti con AWS CloudTrail. Per informazioni sull'uso dei CloudTrail percorsi per registrare AWS le attività, consulta Lavorare con i CloudTrail percorsi nella Guida per l'AWS CloudTrail utente.
- Utilizza soluzioni di AWS crittografia, insieme a tutti i controlli di sicurezza predefiniti all'interno Servizi AWS.
- Utilizza i servizi di sicurezza gestiti avanzati, come Amazon Macie, che aiutano a individuare e proteggere i dati sensibili archiviati in Amazon S3.
- Se hai bisogno di FIPS 140-3 moduli crittografici convalidati per accedere AWS tramite un'interfaccia a riga di comando o unAPI, usa un endpoint. FIPS Per ulteriori informazioni sugli FIPS endpoint disponibili, vedere Federal Information Processing Standard () 140-3. FIPS

Ti consigliamo vivamente di non inserire mai informazioni riservate o sensibili, ad esempio gli indirizzi e-mail dei clienti, nei tag o nei campi di testo in formato libero, ad esempio nel campo Nome. Ciò include quando lavori con Timestream Live Analytics o altro Servizi AWS utilizzando la console,, API o. AWS CLI AWS SDKs I dati inseriti nei tag o nei campi di testo in formato libero utilizzati per i nomi possono essere utilizzati per i la fatturazione o i log di diagnostica. Se fornisci un URL a un server esterno, ti consigliamo vivamente di non includere le informazioni sulle credenziali URL per convalidare la tua richiesta a quel server.

Per informazioni più dettagliate su Timestream per argomenti relativi alla protezione LiveAnalytics dei dati come Encryption at Rest e Key Management, seleziona uno degli argomenti disponibili di seguito.

Argomenti

- Crittografia dei dati inattivi
- Crittografia in transito
- Gestione delle chiavi

Crittografia dei dati inattivi

Timestream for LiveAnalytics encryption at rest offre una maggiore sicurezza crittografando tutti i dati inattivi utilizzando le chiavi di crittografia archiviate in ().AWS Key Management ServiceAWS KMS Questa funzionalità consente di ridurre gli oneri operativi e la complessità associati alla protezione dei dati sensibili. La crittografia dei dati inattivi consente di creare applicazioni ad alto livello di sicurezza che rispettano rigorosi requisiti normativi e di conformità per la crittografia.

- La crittografia è attivata per impostazione predefinita su Timestream for LiveAnalytics database e non può essere disattivata. L'algoritmo di crittografia standard del settore AES -256 è l'algoritmo di crittografia predefinito utilizzato.
- AWS KMS è necessario per la crittografia a riposo in Timestream per. LiveAnalytics
- Non è possibile criptare solo un sottoinsieme di elementi in una tabella.
- Non è quindi necessario modificare le applicazioni client di database per utilizzare la crittografia.

Se non fornisci una chiave, Timestream for LiveAnalytics crea e utilizza una AWS KMS chiave denominata `alias/aws/timestream` nel tuo account.

Puoi utilizzare la tua chiave di accesso gestita dal cliente KMS per crittografare i dati del tuo Timestream. LiveAnalytics Per ulteriori informazioni sulle chiavi in Timestream for, vedi. LiveAnalytics Gestione delle chiavi

Timestream for LiveAnalytics archivia i dati in due livelli di archiviazione, memory store e magnetic store. I dati dell'archivio di memoria vengono crittografati utilizzando una chiave di servizio Timestream. LiveAnalytics I dati dell'archivio magnetico vengono crittografati utilizzando la tua AWS KMS chiave.

Il servizio Timestream Query richiede credenziali per accedere ai dati. Queste credenziali vengono crittografate utilizzando la tua chiave. KMS

> **Note**
>
> Timestream for LiveAnalytics non richiede ogni operazione di AWS KMS Decrypt. Invece, mantiene una cache locale di chiavi per 5 minuti con traffico attivo. Qualsiasi modifica delle autorizzazioni viene propagata tramite il Timestream per il LiveAnalytics sistema con un'eventuale coerenza entro un massimo di 5 minuti.

Crittografia in transito

Tutti i dati di Timestream Live Analytics vengono crittografati in transito. Per impostazione predefinita, tutte le comunicazioni da e verso Timestream for LiveAnalytics sono protette utilizzando la crittografia Transport Layer Security (). TLS

Gestione delle chiavi

Puoi gestire le chiavi per Amazon Timestream Live Analytics utilizzando AWS il Key Management Service ().AWS KMS Timestream Live Analytics richiede l'uso di KMS per crittografare i dati. Sono disponibili le seguenti opzioni per la gestione delle chiavi, a seconda del livello di controllo richiesto sulle chiavi:

Risorse di database e tabelle

- Chiave gestita da Timestream Live Analytics: se non fornisci una chiave, Timestream Live Analytics creerà una chiave utilizzando. `alias/aws/timestream` KMS
- Chiave gestita dal cliente: le chiavi gestite dal cliente sono supportate. KMS Scegliete questa opzione se avete bisogno di un maggiore controllo sulle autorizzazioni e sul ciclo di vita delle vostre chiavi, inclusa la possibilità di farle ruotare automaticamente su base annuale.

Risorsa di query pianificata

- Chiave di proprietà di Timestream Live Analytics: se non si fornisce una chiave, Timestream Live Analytics utilizzerà una chiave propria per crittografare la risorsa Query, KMS questa chiave è presente nell'account timestream. Per maggiori dettagli, consulta le chiavi possedute nella guida per gli sviluppatori AWS . KMS

- Chiave gestita dal KMS cliente: le chiavi gestite dal cliente sono supportate. Scegliete questa opzione se avete bisogno di un maggiore controllo sulle autorizzazioni e sul ciclo di vita delle vostre chiavi, inclusa la possibilità di farle ruotare automaticamente su base annuale.

KMSle chiavi in un archivio di chiavi esterno (XKS) non sono supportate.

Gestione delle identità e degli accessi per Amazon Timestream for LiveAnalytics

AWS Identity and Access Management (IAM) è un programma Servizio AWS che aiuta un amministratore a controllare in modo sicuro l'accesso alle risorse. AWS IAMgli amministratori controllano chi può essere autenticato (effettuato l'accesso) e autorizzato (disporre delle autorizzazioni) a utilizzare Timestream per le risorse. LiveAnalytics IAMè un file Servizio AWS che puoi utilizzare senza costi aggiuntivi.

Argomenti

Destinatari

Il modo in cui usi AWS Identity and Access Management (IAM) varia a seconda del lavoro che svolgi in Timestream. LiveAnalytics

Utente del servizio: se utilizzi il LiveAnalytics servizio Timestream for per svolgere il tuo lavoro, l'amministratore ti fornisce le credenziali e le autorizzazioni necessarie. Man mano che utilizzi più Timestream per LiveAnalytics le funzionalità per svolgere il tuo lavoro, potresti aver bisogno di autorizzazioni aggiuntive. La comprensione della gestione dell'accesso ti consente di richiedere le autorizzazioni corrette all'amministratore. Se non riesci ad accedere a una funzionalità di Timestream per, consulta. LiveAnalytics Risoluzione dei problemi relativi all'identità e all'accesso di Amazon Timestream LiveAnalytics

Amministratore del servizio: se sei responsabile delle LiveAnalytics risorse di Timestream della tua azienda, probabilmente hai pieno accesso a Timestream per. LiveAnalytics È tuo compito determinare a quale Timestream per le LiveAnalytics funzionalità e le risorse a cui gli utenti del servizio devono accedere. È quindi necessario inviare richieste all'IAMamministratore per modificare le autorizzazioni degli utenti del servizio. Consulta le informazioni contenute in questa pagina per comprendere i concetti di base diIAM. Per ulteriori informazioni su come la tua azienda può utilizzare IAM Timestream per LiveAnalytics, consulta. Come funziona Amazon Timestream for LiveAnalytics con IAM

IAMamministratore: se sei un IAM amministratore, potresti voler conoscere i dettagli su come scrivere politiche per gestire l'accesso a Timestream per. LiveAnalytics Per visualizzare un esempio di Timestream per le politiche LiveAnalytics basate sull'identità che puoi utilizzare in, consulta. IAM Amazon Timestream LiveAnalytics per esempi di policy basate sull'identità

Autenticazione con identità

L'autenticazione è il modo in cui accedi utilizzando le tue credenziali di identità. AWS È necessario autenticarsi (accedere a AWS) come Utente root dell'account AWS, come IAM utente o assumendo un ruolo. IAM

È possibile accedere AWS come identità federata utilizzando le credenziali fornite tramite una fonte di identità. AWS IAM Identity Center Gli utenti (IAMIdentity Center), l'autenticazione Single Sign-On della tua azienda e le tue credenziali di Google o Facebook sono esempi di identità federate. Quando accedi come identità federata, l'amministratore aveva precedentemente configurato la federazione delle identità utilizzando i ruoli. IAM Quando si accede AWS utilizzando la federazione, si assume indirettamente un ruolo.

A seconda del tipo di utente, puoi accedere al AWS Management Console o al portale di AWS accesso. Per ulteriori informazioni sull'accesso a AWS, vedi Come accedere al tuo Account AWS nella Guida per l'Accedi ad AWS utente.

Se accedi a AWS livello di codice, AWS fornisce un kit di sviluppo software (SDK) e un'interfaccia a riga di comando () per firmare crittograficamente le tue richieste utilizzando le tue credenziali. CLI Se non utilizzi AWS strumenti, devi firmare tu stesso le richieste. Per ulteriori informazioni sull'utilizzo del metodo consigliato per firmare autonomamente le richieste, consulta AWS Signature Version 4 per API le richieste nella Guida per l'IAMutente.

A prescindere dal metodo di autenticazione utilizzato, potrebbe essere necessario specificare ulteriori informazioni sulla sicurezza. Ad esempio, ti AWS consiglia di utilizzare l'autenticazione a più fattori

(MFA) per aumentare la sicurezza del tuo account. Per ulteriori informazioni, consulta Autenticazione a più fattori nella Guida per l'AWS IAM Identity Center utente e Autenticazione a AWS più fattori IAM nella Guida per l'IAMutente.

IAM users and groups

Un IAMutente è un'identità interna all'utente Account AWS che dispone di autorizzazioni specifiche per una singola persona o applicazione. Laddove possibile, consigliamo di fare affidamento su credenziali temporanee anziché creare IAM utenti con credenziali a lungo termine come password e chiavi di accesso. Tuttavia, se hai casi d'uso specifici che richiedono credenziali a lungo termine con IAM gli utenti, ti consigliamo di ruotare le chiavi di accesso. Per ulteriori informazioni, consulta Ruotare regolarmente le chiavi di accesso per i casi d'uso che richiedono credenziali a lungo termine nella Guida per l'utente. IAM

Un IAMgruppo è un'identità che specifica un insieme di utenti. IAM Non è possibile eseguire l'accesso come gruppo. È possibile utilizzare gruppi per specificare le autorizzazioni per più utenti alla volta. I gruppi semplificano la gestione delle autorizzazioni per set di utenti di grandi dimensioni. Ad esempio, è possibile assegnare un nome a un gruppo IAMAdminse concedere a tale gruppo le autorizzazioni per IAM amministrare le risorse.

Gli utenti sono diversi dai ruoli. Un utente è associato in modo univoco a una persona o un'applicazione, mentre un ruolo è destinato a essere assunto da chiunque ne abbia bisogno. Gli utenti dispongono di credenziali a lungo termine permanenti, mentre i ruoli forniscono credenziali temporanee. Per ulteriori informazioni, consulta Casi d'uso per IAM gli utenti nella Guida per l'IAMutente.

IAMruoli

Un IAMruolo è un'identità interna all'utente Account AWS che dispone di autorizzazioni specifiche. È simile a un IAM utente, ma non è associato a una persona specifica. Per assumere temporaneamente un IAM ruolo in AWS Management Console, puoi passare da un utente a un IAM ruolo (console). È possibile assumere un ruolo chiamando un' AWS APIoperazione AWS CLI or o utilizzando un'operazione personalizzataURL. Per ulteriori informazioni sui metodi di utilizzo dei ruoli, vedere Metodi per assumere un ruolo nella Guida per l'IAMutente.

IAMi ruoli con credenziali temporanee sono utili nelle seguenti situazioni:

- Accesso utente federato: per assegnare le autorizzazioni a una identità federata, è possibile creare un ruolo e definire le autorizzazioni per il ruolo. Quando un'identità federata viene autenticata, l'identità viene associata al ruolo e ottiene le autorizzazioni da esso definite. Per informazioni

sui ruoli per la federazione, vedere Creazione di un ruolo per un provider di identità di terze parti nella Guida per l'IAMutente. Se utilizzi IAM Identity Center, configuri un set di autorizzazioni. Per controllare a cosa possono accedere le identità dopo l'autenticazione, IAM Identity Center correla il set di autorizzazioni a un ruolo in. IAM Per informazioni sui set di autorizzazioni, consulta Set di autorizzazioni nella Guida per l'utente di AWS IAM Identity Center .

- Autorizzazioni IAM utente temporanee: un IAM utente o un ruolo può assumere il IAM ruolo di assumere temporaneamente autorizzazioni diverse per un'attività specifica.
- Accesso su più account: puoi utilizzare un IAM ruolo per consentire a qualcuno (un responsabile fidato) di un altro account di accedere alle risorse del tuo account. I ruoli sono lo strumento principale per concedere l'accesso multi-account. Tuttavia, con alcuni Servizi AWS, è possibile allegare una policy direttamente a una risorsa (anziché utilizzare un ruolo come proxy). Per conoscere la differenza tra ruoli e politiche basate sulle risorse per l'accesso tra account diversi, consulta la sezione Accesso alle risorse su più account IAM nella Guida per l'utente. IAM
- Accesso tra servizi: alcuni Servizi AWS utilizzano funzionalità in altri. Servizi AWS Ad esempio, quando effettui una chiamata in un servizio, è normale che quel servizio esegua applicazioni in Amazon EC2 o archivi oggetti in Amazon S3. Un servizio può eseguire questa operazione utilizzando le autorizzazioni dell'entità chiamante, utilizzando un ruolo di servizio o utilizzando un ruolo collegato al servizio.
  - Sessioni di accesso diretto (FAS): quando utilizzi un IAM utente o un ruolo per eseguire azioni AWS, sei considerato un principale. Quando si utilizzano alcuni servizi, è possibile eseguire un'operazione che attiva un'altra operazione in un servizio diverso. FASutilizza le autorizzazioni del principale che chiama an Servizio AWS, in combinazione con la richiesta di effettuare richieste Servizio AWS ai servizi downstream. FASIe richieste vengono effettuate solo quando un servizio riceve una richiesta che richiede interazioni con altri Servizi AWS o risorse per essere completata. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le azioni. Per i dettagli FAS delle politiche relative alle richieste, consulta Forward access sessions.
  - Ruolo di servizio: un ruolo di servizio è un IAMruolo che un servizio assume per eseguire azioni per conto dell'utente. Un IAM amministratore può creare, modificare ed eliminare un ruolo di servizio dall'internoIAM. Per ulteriori informazioni, vedere Creazione di un ruolo per delegare le autorizzazioni a un utente Servizio AWS nella Guida per l'IAMutente.
  - Ruolo collegato al servizio: un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un. Servizio AWS Il servizio può assumere il ruolo per eseguire un'azione per tuo conto. I ruoli collegati al servizio vengono visualizzati nel tuo account Account AWS e sono di proprietà del servizio. Un IAM amministratore può visualizzare, ma non modificare le autorizzazioni per i ruoli collegati al servizio.

- Applicazioni in esecuzione su Amazon EC2: puoi utilizzare un IAM ruolo per gestire le credenziali temporanee per le applicazioni in esecuzione su un'EC2istanza e che effettuano AWS CLI o richiedono AWS API. Ciò è preferibile alla memorizzazione delle chiavi di accesso all'interno dell'EC2istanza. Per assegnare un AWS ruolo a un'EC2istanza e renderlo disponibile per tutte le sue applicazioni, create un profilo di istanza collegato all'istanza. Un profilo di istanza contiene il ruolo e consente ai programmi in esecuzione sull'EC2istanza di ottenere credenziali temporanee. Per ulteriori informazioni, consulta Usare un IAM ruolo per concedere le autorizzazioni alle applicazioni in esecuzione su EC2 istanze Amazon nella Guida per l'IAMutente.

Gestione dell'accesso con policy

Puoi controllare l'accesso AWS creando policy e associandole a AWS identità o risorse. Una policy è un oggetto AWS che, se associato a un'identità o a una risorsa, ne definisce le autorizzazioni. AWS valuta queste politiche quando un principale (utente, utente root o sessione di ruolo) effettua una richiesta. Le autorizzazioni nelle policy determinano l'approvazione o il rifiuto della richiesta. La maggior parte delle politiche viene archiviata AWS come JSON documenti. Per ulteriori informazioni sulla struttura e il contenuto dei documenti relativi alle JSON politiche, vedere Panoramica delle JSON politiche nella Guida per l'IAMutente.

Gli amministratori possono utilizzare AWS JSON le politiche per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire azioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Per concedere agli utenti l'autorizzazione a eseguire azioni sulle risorse di cui hanno bisogno, un IAM amministratore può creare IAM politiche. L'amministratore può quindi aggiungere le IAM politiche ai ruoli e gli utenti possono assumerli.

IAMle politiche definiscono le autorizzazioni per un'azione indipendentemente dal metodo utilizzato per eseguire l'operazione. Ad esempio, supponiamo di disporre di una policy che consente l'operazione `iam:GetRole`. Un utente con tale criterio può ottenere informazioni sul ruolo da AWS Management Console, da o da. AWS CLI AWS API

Policy basate su identità

I criteri basati sull'identità sono documenti relativi alle politiche di JSON autorizzazione che è possibile allegare a un'identità, ad esempio un IAM utente, un gruppo di utenti o un ruolo. Tali policy definiscono le azioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una politica basata sull'identità, consulta Definire le IAM autorizzazioni personalizzate con le politiche gestite dal cliente nella Guida per l'utente. IAM

Le policy basate su identità possono essere ulteriormente classificate come policy inline o policy gestite. Le policy inline sono integrate direttamente in un singolo utente, gruppo o ruolo. Le politiche gestite sono politiche autonome che puoi allegare a più utenti, gruppi e ruoli all'interno del tuo. Account AWS Le politiche gestite includono politiche AWS gestite e politiche gestite dai clienti. Per informazioni su come scegliere tra una politica gestita o una politica in linea, consulta Scegliere tra politiche gestite e politiche in linea nella Guida per l'IAMutente.

Policy basate su risorse

Le politiche basate sulle risorse sono documenti di JSON policy allegati a una risorsa. Esempi di politiche basate sulle risorse sono le policy di trust dei IAM ruoli e le policy dei bucket di Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarli per controllare l'accesso a una risorsa specifica. Quando è collegata a una risorsa, una policy definisce le azioni che un principale può eseguire su tale risorsa e a quali condizioni. È necessario specificare un principale in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o. Servizi AWS

Le policy basate sulle risorse sono policy inline che si trovano in tale servizio. Non è possibile utilizzare le politiche AWS gestite IAM in una politica basata sulle risorse.

Elenchi di controllo degli accessi () ACLs

Le liste di controllo degli accessi (ACLs) controllano quali principali (membri dell'account, utenti o ruoli) dispongono delle autorizzazioni per accedere a una risorsa. ACLssono simili alle politiche basate sulle risorse, sebbene non utilizzino il formato del documento di policy. JSON

Amazon S3 e Amazon VPC sono esempi di servizi che supportano. AWS WAF ACLs Per ulteriori informazioniACLs, consulta la panoramica di Access control list (ACL) nella Amazon Simple Storage Service Developer Guide.

Altri tipi di policy

AWS supporta tipi di policy aggiuntivi e meno comuni. Questi tipi di policy possono impostare il numero massimo di autorizzazioni concesse dai tipi di policy più comuni.

- Limiti delle autorizzazioni: un limite di autorizzazioni è una funzionalità avanzata in cui si impostano le autorizzazioni massime che una politica basata sull'identità può concedere a un'entità (utente o ruolo). IAM IAM È possibile impostare un limite delle autorizzazioni per un'entità. Le autorizzazioni risultanti sono l'intersezione delle policy basate su identità dell'entità e i relativi

limiti delle autorizzazioni. Le policy basate su risorse che specificano l'utente o il ruolo nel campo `Principal`sono condizionate dal limite delle autorizzazioni. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni sui limiti delle autorizzazioni, consulta Limiti delle autorizzazioni per le entità nella Guida per l'utente. IAM IAM

- Politiche di controllo del servizio (SCPs): SCPs sono JSON politiche che specificano le autorizzazioni massime per un'organizzazione o un'unità organizzativa (OU) in. AWS Organizations AWS Organizations è un servizio per il raggruppamento e la gestione centralizzata di più Account AWS di proprietà dell'azienda. Se abiliti tutte le funzionalità di un'organizzazione, puoi applicare le politiche di controllo del servizio (SCPs) a uno o tutti i tuoi account. SCPLimita le autorizzazioni per le entità negli account dei membri, inclusa ciascuna Utente root dell'account AWS. Per ulteriori informazioni su Organizations andSCPs, consulta le politiche di controllo dei servizi nella Guida AWS Organizations per l'utente.
- Policy di sessione: le policy di sessione sono policy avanzate che vengono trasmesse come parametro quando si crea in modo programmatico una sessione temporanea per un ruolo o un utente federato. Le autorizzazioni della sessione risultante sono l'intersezione delle policy basate su identità del ruolo o dell'utente e le policy di sessione. Le autorizzazioni possono anche provenire da una policy basata su risorse. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni, consulta le politiche di sessione nella Guida IAM per l'utente.

Più tipi di policy

Quando più tipi di policy si applicano a una richiesta, le autorizzazioni risultanti sono più complicate da comprendere. Per informazioni su come AWS determinare se consentire una richiesta quando sono coinvolti più tipi di policy, consulta Logica di valutazione delle politiche nella Guida per l'IAMutente.

Come funziona Amazon Timestream for LiveAnalytics con IAM

Prima di utilizzare IAM per gestire l'accesso a Timestream for LiveAnalytics, è necessario comprendere quali IAM funzionalità sono disponibili per l'uso con Timestream for. LiveAnalytics Per avere una panoramica generale del funzionamento di Timestream for LiveAnalytics e di altri AWS serviziIAM, consulta AWS Services That Work with nella Guida per l'utente. IAM IAM

Argomenti

- Timestream per politiche basate sull'identità LiveAnalytics
- Timestream per politiche basate sulle risorse LiveAnalytics

- Autorizzazione basata su Timestream per i tag LiveAnalytics
- Timestream per i ruoli LiveAnalytics IAM

Timestream per politiche basate sull'identità LiveAnalytics

Con le politiche IAM basate sull'identità, è possibile specificare azioni e risorse consentite o negate, nonché le condizioni in base alle quali le azioni sono consentite o negate. Timestream for LiveAnalytics supporta azioni e risorse specifiche e chiavi di condizione. Per informazioni su tutti gli elementi utilizzati in una JSON policy, consulta IAMJSONPolicy Elements Reference nella Guida per l'IAMutente.

Azioni

Gli amministratori possono utilizzare AWS JSON le policy per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'`Action`elemento di una JSON policy descrive le azioni che è possibile utilizzare per consentire o negare l'accesso a una policy. Le azioni politiche in genere hanno lo stesso nome dell' AWS APIoperazione associata. Esistono alcune eccezioni, come le azioni basate solo sulle autorizzazioni che non hanno un'operazione corrispondente. API Esistono anche alcune operazioni che richiedono più operazioni in una policy. Queste operazioni aggiuntive sono denominate operazioni dipendenti.

Includi le operazioni in una policy per concedere le autorizzazioni a eseguire l'operazione associata.

È possibile specificare le seguenti azioni nell'elemento Azione di una dichiarazione di policy. IAM Utilizzate le politiche per concedere le autorizzazioni per eseguire un'operazione inAWS. Quando si utilizza un'azione in una politica, in genere si consente o si nega l'accesso all'APIoperazione, al CLI comando o al SQL comando con lo stesso nome.

In alcuni casi, una singola azione controlla l'accesso a un'APIoperazione e a un SQL comando. In alternativa, alcune operazioni richiedono operazioni differenti.

Per un elenco dei Timestream supportati per LiveAnalytics `Action`'s, consulta la tabella seguente:

> **Note**
>
> Per tutti i database specifici`Actions`, puoi specificare un database ARN per limitare l'azione a un determinato database.

Azioni	Descrizione	Livello di accesso	Tipi di risorsa (*obbligatorio)
DescribeEndpoints	Restituisce l'endpoint Timestream a cui devono essere fatte le richieste successive.	Tutti	*
Select	Esegui query su Timestream che selezionano i dati da una o più tabelle. Consulta questa nota per una spiegazione dettagliata	Lettura	tabella*
CancelQuery	Annullare una richiesta.	Lettura	*
ListTables	Ottieni l'elenco delle tabelle.	Elenco	database*
ListDatabases	Ottieni l'elenco dei database.	Elenco	*
ListMeasures	Ottieni l'elenco delle misure.	Lettura	tabella*
DescribeTable	Ottieni la descrizione della tabella.	Lettura	tavolo*
DescribeDatabase	Ottieni la descrizione del database.	Lettura	database*
SelectValues	Esegui query che non richiedono la specificazione di una risorsa particolare. Per una spiegazione dettaglia	Lettura	*

Azioni	Descrizione	Livello di accesso	Tipi di risorsa (*obbligatorio)
	ta, consulta questa nota.		
WriteRecords	Inserisci i dati in Timestream.	Scrittura	tabella*
CreateTable	Creare una tabella .	Scrittura	database*
CreateDatabase	Crea un database.	Scrittura	*
DeleteDatabase	Eliminare un database.	Scrittura	*
UpdateDatabase	Aggiornare un database.	Scrittura	*
DeleteTable	Eliminare una tabella.	Scrittura	database*
UpdateTable	Aggiorna una tabella.	Scrittura	database*

SelectValues vs. seleziona:

`SelectValues`è un file `Action` che viene utilizzato per le interrogazioni che non richiedono una risorsa. Un esempio di query che non richiede una risorsa è il seguente:

```
SELECT 1
```

Notate che questa query non si riferisce a un particolare flusso temporale per la LiveAnalytics risorsa. Consideriamo un altro esempio:

```
SELECT now()
```

Questa query restituisce il timestamp corrente utilizzando la `now()` funzione, ma non richiede la specificazione di una risorsa. `SelectValues`viene spesso utilizzato per i test, in modo che Timestream for LiveAnalytics possa eseguire query senza risorse. Ora, considera una domanda: `Select`

```
SELECT * FROM database.table
```

Questo tipo di query richiede una risorsa, in particolare un Timestream per LiveAnalytics `table`, in modo che i dati specificati possano essere recuperati dalla tabella.

Risorse

Gli amministratori possono utilizzare AWS JSON le policy per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento `Resource` JSON policy specifica l'oggetto o gli oggetti a cui si applica l'azione. Le istruzioni devono includere un elemento `Resource`o un elemento `NotResource`. Come best practice, specifica una risorsa utilizzando il relativo Amazon Resource Name (ARN). Puoi eseguire questa operazione per azioni che supportano un tipo di risorsa specifico, note come autorizzazioni a livello di risorsa.

Per le azioni che non supportano le autorizzazioni a livello di risorsa, ad esempio le operazioni di elenco, utilizza un carattere jolly (*) per indicare che l'istruzione si applica a tutte le risorse.

```
"Resource": "*"
```

In Timestream per LiveAnalytics database e tabelle può essere utilizzato nell'`Resource`elemento delle IAM autorizzazioni.

La risorsa Timestream per il LiveAnalytics database ha quanto segue: ARN

```
arn:${Partition}:timestream:${Region}:${Account}:database/${DatabaseName}
```

La risorsa Timestream for LiveAnalytics table è la seguente: ARN

```
arn:${Partition}:timestream:${Region}:${Account}:database/${DatabaseName}/table/
${TableName}
```

Per ulteriori informazioni sul formato diARNs, consulta Amazon Resource Names (ARNs) e AWS Service Namespaces.

Ad esempio, per specificare `database` lo spazio delle chiavi nella tua dichiarazione, usa quanto segue: ARN

```
"Resource": "arn:aws:timestream:us-east-1:123456789012:database/mydatabase"
```

Per specificare tutti i database che appartengono a un account specifico, usa il carattere jolly (*):

```
"Resource": "arn:aws:timestream:us-east-1:123456789012:database/*"
```

Alcuni Timestream per LiveAnalytics le azioni, come quelle per la creazione di risorse, non possono essere eseguiti su una risorsa specifica. In questi casi, è necessario utilizzare il carattere jolly (*).

```
"Resource": "*"
```

Chiavi di condizione

Timestream for LiveAnalytics non fornisce chiavi di condizione specifiche del servizio, ma supporta l'utilizzo di alcune chiavi di condizione globali. Per visualizzare tutte le chiavi di condizione AWS globali, consulta AWS Global Condition Context Keys nella Guida per l'utente. IAM

Esempi

Per visualizzare esempi di Timestream per le politiche basate sull' LiveAnalytics identità, vedere. Amazon Timestream LiveAnalytics per esempi di policy basate sull'identità

Timestream per politiche basate sulle risorse LiveAnalytics

Timestream for non supporta le politiche basate sulle risorse. LiveAnalytics Per visualizzare un esempio di una pagina di policy basata su risorse dettagliata, consulta https://docs.aws.amazon.com/lambda/latest/dg/access-control-resource-based.html.

Autorizzazione basata su Timestream per i tag LiveAnalytics

Puoi gestire l'accesso alle LiveAnalytics risorse del tuo Timestream utilizzando i tag. Per gestire l'accesso alle risorse in base ai tag, fornisci le informazioni sui tag nell'elemento condition di una policy utilizzando i `timestream:ResourceTag/`*`key-name`* tastiaws:RequestTag/*key-name*, o `aws:TagKeys` condition. Per ulteriori informazioni sull'assegnazione di tag a Timestream per LiveAnalytics le risorse, consulta. the section called “Applicazione di tag alle risorse ”

Per visualizzare policy basate sulle identità di esempio per limitare l'accesso a una risorsa basata su tag su tale risorsa, consulta Timestream per l'accesso alle LiveAnalytics risorse in base ai tag.

Timestream per i ruoli LiveAnalytics IAM

Un IAMruolo è un'entità all'interno del tuo AWS account che dispone di autorizzazioni specifiche.

Utilizzo di credenziali temporanee con Timestream per LiveAnalytics

Puoi utilizzare credenziali temporanee per accedere con la federazione, assumere un IAM ruolo o assumere un ruolo tra account. È possibile ottenere credenziali di sicurezza temporanee chiamando AWS STS API operazioni come o. AssumeRoleGetFederationToken

Ruoli collegati ai servizi

Timestream for LiveAnalytics non supporta i ruoli collegati al servizio.

Ruoli di servizio

Timestream for LiveAnalytics non supporta i ruoli di servizio.

AWS politiche gestite per Amazon Timestream Live Analytics

Una politica AWS gestita è una politica autonoma creata e amministrata da. AWS AWS le politiche gestite sono progettate per fornire autorizzazioni per molti casi d'uso comuni, in modo da poter iniziare ad assegnare autorizzazioni a utenti, gruppi e ruoli.

Tieni presente che le policy AWS gestite potrebbero non concedere le autorizzazioni con il privilegio minimo per i tuoi casi d'uso specifici, poiché sono disponibili per tutti i clienti. AWS Ti consigliamo pertanto di ridurre ulteriormente le autorizzazioni definendo policy gestite dal cliente specifiche per i tuoi casi d'uso.

Non è possibile modificare le autorizzazioni definite nelle politiche gestite. AWS Se AWS aggiorna le autorizzazioni definite in una politica AWS gestita, l'aggiornamento ha effetto su tutte le identità principali (utenti, gruppi e ruoli) a cui è associata la politica. AWS è più probabile che aggiorni una policy AWS gestita quando ne Servizio AWS viene lanciata una nuova o quando diventano disponibili nuove API operazioni per i servizi esistenti.

Per ulteriori informazioni, consulta le politiche AWS gestite nella Guida IAM per l'utente.

Argomenti

- AWS politica gestita: AmazonTimestreamReadOnlyAccess
- AWS politica gestita: AmazonTimestreamConsoleFullAccess
- AWS politica gestita: AmazonTimestreamFullAccess
- Aggiornamenti di Timestream Live Analytics alle AWS politiche gestite

AWS politica gestita: AmazonTimestreamReadOnlyAccess

Puoi collegarti `AmazonTimestreamReadOnlyAccess` ai tuoi utenti, gruppi e ruoli. La policy fornisce l'accesso in sola lettura ad Amazon Timestream.

Dettagli dell'autorizzazione

Questa policy include la seguente autorizzazione:

- `Amazon Timestream`— Fornisce accesso in sola lettura ad Amazon Timestream. Questa politica concede inoltre l'autorizzazione ad annullare qualsiasi query in esecuzione.

Per rivedere il JSON formato di questa politica, vedere AmazonTimestreamReadOnlyAccess.

AWS politica gestita: AmazonTimestreamConsoleFullAccess

Puoi collegarti `AmazonTimestreamConsoleFullAccess` ai tuoi utenti, gruppi e ruoli.

La policy fornisce l'accesso completo alla gestione di Amazon Timestream utilizzando. AWS Management Console Questa politica concede anche le autorizzazioni per determinate AWS KMS operazioni e operazioni di gestione delle query salvate.

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni:

- `Amazon Timestream`— Garantisce ai mandanti l'accesso completo ad Amazon Timestream.

- `AWS KMS`— Consente ai mandanti di elencare gli alias e descrivere le chiavi.
- `Amazon S3`— Consente ai responsabili di elencare tutti i bucket Amazon S3.
- `Amazon SNS`— Consente ai responsabili di elencare gli SNS argomenti di Amazon.
- `IAM`— Consente ai presidi di IAM elencare i ruoli.
- `DBQMS`: consente ai principali di accedere, creare, eliminare, descrivere e aggiornare le query. Il Database Query Metadata Service (dbqms) è un servizio solo interno. Fornisce le tue query recenti e salvate per l'editor di query su multipli Servizi AWS, incluso Amazon Timestream. AWS Management Console

Per rivedere il JSON formato di questa politica, consulta. AmazonTimestreamConsoleFullAccess

AWS politica gestita: AmazonTimestreamFullAccess

Puoi collegarti `AmazonTimestreamFullAccess` ai tuoi utenti, gruppi e ruoli.

La policy fornisce l'accesso completo ad Amazon Timestream. Questa politica concede anche le autorizzazioni per determinate operazioni. AWS KMS

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni:

- `Amazon Timestream`— Garantisce ai mandanti l'accesso completo ad Amazon Timestream.
- `AWS KMS`— Consente ai mandanti di elencare gli alias e descrivere le chiavi.
- `Amazon S3`— Consente ai responsabili di elencare tutti i bucket Amazon S3.

Per rivedere il JSON formato di questa politica, consulta. AmazonTimestreamFullAccess

Aggiornamenti di Timestream Live Analytics alle AWS politiche gestite

Visualizza i dettagli sugli aggiornamenti alle politiche AWS gestite per Timestream Live Analytics da quando questo servizio ha iniziato a tenere traccia di queste modifiche. Per ricevere avvisi automatici sulle modifiche a questa pagina, iscriviti al RSS feed nella pagina della cronologia dei documenti di Timestream Live Analytics.

Modifica	Descrizione	Data
AmazonTimestreamReadOnlyAccess: aggiornamento a una policy esistente	È stata aggiunta l'`timestream:DescribeAccountSettings` azione alla politica gestita esistente`AmazonTimestreamReadOnlyAccess` . Questa azione viene utilizzata per descrivere Account AWS le impostazioni. Timestream Live Analytics ha inoltre aggiornato questa politica gestita aggiungendo un campo. `Sid` L'aggiornamento della politica non influisce sull'utilizzo della politica `AmazonTimestreamReadOnlyAccess` gestita.	03 giugno 2024
AmazonTimestreamReadOnlyAccess: aggiornamento a una policy esistente	Sono state aggiunte le `timestream:ListBatchLoadTasks` azioni `timestream:DescribeBatchLoadTask` e alla politica `AmazonTimestreamReadOnlyAccess` gestita esistente. Queste azioni vengono utilizzate per elencare e descrivere le attività di caricamento in batch. L'aggiornamento della politica non influisce sull'utilizzo	24 febbraio 2023

Modifica	Descrizione	Data
	della politica `AmazonTimestreamReadOnlyAccess` gestita.	
AmazonTimestreamReadOnlyAccess: aggiornamento a una policy esistente	Sono state aggiunte le `timestream:ListScheduledQueries` azioni `timestream:DescribeScheduledQuery` e alla politica `AmazonTimestreamReadOnlyAccess` gestita esistente. Queste azioni vengono utilizzate per elencare e descrivere le interrogazioni pianificate esistenti. L'aggiornamento della politica non influisce sull'utilizzo della politica `AmazonTimestreamReadOnlyAccess` gestita.	29 novembre 2021

Modifica	Descrizione	Data
AmazonTimestreamConsoleFullAccess: aggiornamento a una policy esistente	L'`s3:ListAllMyBuckets` azione è stata aggiunta alla politica `AmazonTimestreamConsoleFullAccess` gestita esistente. Questa azione viene utilizzata quando si specifica un bucket Amazon S3 per Timestream per registrare gli errori di scrittura nell'archivio magnetico. L'aggiornamento della policy non influisce sull'utilizzo della policy gestita. `AmazonTimestreamConsoleFullAccess`	29 novembre 2021
AmazonTimestreamFullAccess: aggiornamento a una policy esistente	L'`s3:ListAllMyBuckets` azione è stata aggiunta alla politica `AmazonTimestreamFullAccess` gestita esistente. Questa azione viene utilizzata quando si specifica un bucket Amazon S3 per Timestream per registrare gli errori di scrittura nell'archivio magnetico. L'aggiornamento della policy non influisce sull'utilizzo della policy gestita. `AmazonTimestreamFullAccess`	29 novembre 2021

Modifica	Descrizione	Data
AmazonTimestreamConsoleFullAccess: aggiornamento a una policy esistente	Azioni ridondanti rimosse dalla politica `AmazonTimestreamConsoleFullAccess` gestita esistente. In precedenza, questa politica includeva un'azione ridondante. `dbqms:DescribeQueryHistory` La politica aggiornata rimuove l'azione ridondante. L'aggiornamento della politica non influisce sull'utilizzo della politica `AmazonTimestreamConsoleFullAccess` gestita.	23 Aprile 2021
Timestream Live Analytics ha iniziato a tracciare le modifiche	Timestream Live Analytics ha iniziato a tenere traccia delle modifiche alle sue politiche gestite. AWS	21 aprile 2021

Amazon Timestream LiveAnalytics per esempi di policy basate sull'identità

Per impostazione predefinita, IAM gli utenti e i ruoli non dispongono dell'autorizzazione per creare o modificare Timestream per le risorse. LiveAnalytics Inoltre, non possono eseguire attività utilizzando AWS Management Console, CQLSH AWS CLI, o. AWS API Un IAM amministratore deve creare IAM politiche che concedano a utenti e ruoli l'autorizzazione a eseguire API operazioni specifiche sulle risorse specificate di cui ha bisogno. L'amministratore deve quindi allegare tali politiche agli IAM utenti o ai gruppi che richiedono tali autorizzazioni.

Per informazioni su come creare una politica IAM basata sull'identità utilizzando questi documenti di esempioJSON, consulta Creazione di politiche nella JSON scheda nella Guida per l'utente. IAM

Argomenti

- Best practice per le policy
- Utilizzo del Timestream per console LiveAnalytics
- Consentire agli utenti di visualizzare le loro autorizzazioni
- Operazioni comuni in Timestream per LiveAnalytics
- Timestream per l'accesso alle LiveAnalytics risorse in base ai tag
- Interrogazioni pianificate

Best practice per le policy

Le politiche basate sull'identità determinano se qualcuno può creare, accedere o eliminare Timestream per le risorse del tuo account. LiveAnalytics Queste azioni possono comportare costi aggiuntivi per l' Account AWS. Quando crei o modifichi policy basate su identità, segui queste linee guida e raccomandazioni:

- Inizia con le policy AWS gestite e passa alle autorizzazioni con privilegi minimi: per iniziare a concedere autorizzazioni a utenti e carichi di lavoro, utilizza le politiche gestite che concedono le autorizzazioni per molti casi d'uso comuni.AWS Sono disponibili nel tuo. Account AWS Ti consigliamo di ridurre ulteriormente le autorizzazioni definendo politiche gestite dai AWS clienti specifiche per i tuoi casi d'uso. Per ulteriori informazioni, consulta le politiche AWS gestite o le politiche AWS gestite per le funzioni lavorative nella Guida per l'IAMutente.
- Applica le autorizzazioni con privilegi minimi: quando imposti le autorizzazioni con le IAM politiche, concedi solo le autorizzazioni necessarie per eseguire un'attività. Puoi farlo definendo le azioni che possono essere intraprese su risorse specifiche in condizioni specifiche, note anche come autorizzazioni con privilegi minimi. Per ulteriori informazioni sull'utilizzo per applicare le autorizzazioni, consulta Politiche e autorizzazioni nella Guida IAM per l'utente. IAM IAM
- Utilizza le condizioni nelle IAM politiche per limitare ulteriormente l'accesso: puoi aggiungere una condizione alle tue politiche per limitare l'accesso ad azioni e risorse. Ad esempio, puoi scrivere una condizione di policy per specificare che tutte le richieste devono essere inviate utilizzandoSSL. È inoltre possibile utilizzare condizioni per concedere l'accesso alle azioni di servizio se vengono utilizzate tramite uno specifico Servizio AWS, ad esempio AWS CloudFormation. Per ulteriori informazioni, consulta Elementi IAM JSON della politica: Condizione nella Guida IAM per l'utente.
- Usa IAM Access Analyzer per convalidare IAM le tue policy e garantire autorizzazioni sicure e funzionali: IAM Access Analyzer convalida le policy nuove ed esistenti in modo che aderiscano al linguaggio delle IAM policy () e alle best practice. JSON IAM IAMAccess Analyzer fornisce più di 100 controlli delle politiche e consigli pratici per aiutarti a creare policy sicure e funzionali. Per

ulteriori informazioni, consulta Convalida delle politiche con IAM Access Analyzer nella Guida per l'utente. IAM

- Richiedi l'autenticazione a più fattori (MFA): se hai uno scenario che richiede l'utilizzo di IAM utenti o di un utente root Account AWS, attiva questa opzione MFA per una maggiore sicurezza. Per richiedere MFA quando vengono richiamate API le operazioni, aggiungi MFA delle condizioni alle tue politiche. Per ulteriori informazioni, consulta Secure API access with MFA nella Guida IAM per l'utente.

Per ulteriori informazioni sulle best practice inIAM, consulta la sezione Procedure consigliate in materia di sicurezza IAM nella Guida IAM per l'utente.

Utilizzo del Timestream per console LiveAnalytics

Timestream for non LiveAnalytics richiede autorizzazioni specifiche per accedere ad Amazon Timestream per console. LiveAnalytics Hai bisogno almeno delle autorizzazioni di sola lettura per elencare e visualizzare i dettagli sul Timestream per le risorse del tuo account. LiveAnalytics AWS Se crei una politica basata sull'identità più restrittiva delle autorizzazioni minime richieste, la console non funzionerà come previsto per le entità (utenti o ruoli) che applicano tale politica. IAM

Consentire agli utenti di visualizzare le loro autorizzazioni

Questo esempio mostra come è possibile creare una politica che consenta IAM agli utenti di visualizzare le politiche in linea e gestite allegate alla propria identità utente. Questa politica include le autorizzazioni per completare questa azione sulla console o utilizzando o a livello di codice. AWS CLI AWS API

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
```

```
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Operazioni comuni in Timestream per LiveAnalytics

Di seguito sono riportati alcuni esempi IAM di policy che consentono operazioni comuni in Timestream for service. LiveAnalytics

Argomenti

- Consentire tutte le operazioni
- Consentire le operazioni SELECT
- Consentire SELECT operazioni su più risorse
- Consentire le operazioni sui metadati
- Consentire le operazioni INSERT
- Consentire CRUD le operazioni
- Annulla le interrogazioni e seleziona i dati senza specificare le risorse
- Creare, descrivere, eliminare e descrivere un database
- Limita i database elencati per tag {"Owner": "${username}"}
- Elenca tutte le tabelle in un database
- Crea, descrivi, elimina, aggiorna e seleziona in una tabella
- Limita una query per tabella

Consentire tutte le operazioni

Di seguito è riportato un esempio di politica che consente tutte le operazioni in Timestream per. LiveAnalytics

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Consentire le operazioni SELECT

La seguente politica di esempio consente interrogazioni `SELECT` in stile C su una risorsa specifica.

> **Note**
>
> `<account_ID>`Sostituiscilo con l'ID del tuo account Amazon.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:Select",
                "timestream:DescribeTable",
                "timestream:ListMeasures"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/
table/DevOps"
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "timestream:DescribeEndpoints",
                "timestream:SelectValues",
                "timestream:CancelQuery"
            ],
            "Resource": "*"
        }
    ]
}
```

Consentire SELECT operazioni su più risorse

La seguente politica di esempio consente interrogazioni in stile «SELECT-style» su più risorse.

> **Note**
>
> <account_ID>Sostituiscilo con l'ID del tuo account Amazon.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:Select",
                "timestream:DescribeTable",
                "timestream:ListMeasures"
            ],
            "Resource": [
                "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/table/
DevOps",
                "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/table/
DevOps1",
                "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/table/
DevOps2"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "timestream:DescribeEndpoints",
                "timestream:SelectValues",
                "timestream:CancelQuery"
            ],
            "Resource": "*"
        }
    ]
}
```

Consentire le operazioni sui metadati

La seguente politica di esempio consente all'utente di eseguire query sui metadati, ma non consente all'utente di eseguire operazioni di lettura o scrittura di dati effettivi in Timestream for. LiveAnalytics

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:DescribeEndpoints",
                "timestream:DescribeTable",
                "timestream:ListMeasures",
                "timestream:SelectValues",
                "timestream:ListTables",
                "timestream:ListDatabases",
                "timestream:CancelQuery"
            ],
            "Resource": "*"
        }
    ]
}
```

Consentire le operazioni INSERT

La seguente politica di esempio consente a un utente di eseguire un'`INSERT`operazione sull'account `database/sampleDB/table/DevOps` in`<account_id>`.

> **Note**
>
> `<account_ID>`Sostituiscilo con l'ID del tuo account Amazon.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "timestream:WriteRecords"
            ],
            "Resource": [
                "arn:aws:timestream:us-east-1:<account_id>:database/sampleDB/table/
DevOps"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "timestream:DescribeEndpoints"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Consentire CRUD le operazioni

La seguente politica di esempio consente a un utente di eseguire CRUD operazioni in Timestream per. LiveAnalytics

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:DescribeEndpoints",
                "timestream:CreateTable",
                "timestream:DescribeTable",
                "timestream:CreateDatabase",
                "timestream:DescribeDatabase",
                "timestream:ListTables",
                "timestream:ListDatabases",
                "timestream:DeleteTable",
                "timestream:DeleteDatabase",
```

```
                "timestream:UpdateTable",
                "timestream:UpdateDatabase"
            ],
            "Resource": "*"
        }
    ]
}
```

Annulla le interrogazioni e seleziona i dati senza specificare le risorse

La seguente politica di esempio consente a un utente di annullare le query ed eseguire `Select` query sui dati che non richiedono la specificazione delle risorse:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:SelectValues",
                "timestream:CancelQuery"
            ],
            "Resource": "*"
        }
    ]
}
```

Creare, descrivere, eliminare e descrivere un database

La seguente politica di esempio consente a un utente di creare, descrivere, eliminare e descrivere un database`sampleDB`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:CreateDatabase",
                "timestream:DescribeDatabase",
                "timestream:DeleteDatabase",
                "timestream:UpdateDatabase"
            ],
```

```
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB"
        }
    ]
}
```

Limita i database elencati per tag **`{"Owner": "${username}"}`**

La seguente politica di esempio consente a un utente di elencare tutti i database etichettati con una coppia chiave-valore`{"Owner": "${username}"}`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:ListDatabases"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "${aws:username}"
                }
            }
        }
    ]
}
```

Elenca tutte le tabelle in un database

La seguente politica di esempio per elencare tutte le tabelle nel database`sampleDB`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:ListTables"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/"
        }
    ]
```

```
}
```

Crea, descrivi, elimina, aggiorna e seleziona in una tabella

La seguente politica di esempio consente a un utente di creare tabelle, descrivere tabelle, eliminare tabelle, aggiornare tabelle ed eseguire `Select` query sulla tabella `DevOps` nel database`sampleDB`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:CreateTable",
                "timestream:DescribeTable",
                "timestream:DeleteTable",
                "timestream:UpdateTable",
                "timestream:Select"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/
table/DevOps"
        }
    ]
}
```

Limita una query per tabella

La seguente politica di esempio consente a un utente di interrogare tutte le tabelle tranne quelle `DevOps` del database`sampleDB`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:Select"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/
table/*"
        },
        {
            "Effect": "Deny",
```

```
            "Action": [
                "timestream:Select"
            ],
            "Resource": "arn:aws:timestream:us-east-1:<account_ID>:database/sampleDB/
table/DevOps"
        }
    ]
}
```

Timestream per l'accesso alle LiveAnalytics risorse in base ai tag

Puoi utilizzare le condizioni della tua politica basata sull'identità per controllare l'accesso a Timestream per le risorse basate sui tag. LiveAnalytics Questa sezione fornisce alcuni esempi.

L'esempio seguente mostra come è possibile creare una politica che conceda a un utente le autorizzazioni per visualizzare una tabella se la tabella `Owner` contiene il valore del nome utente di quell'utente.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadOnlyAccessTaggedTables",
            "Effect": "Allow",
            "Action": "timestream:Select",
            "Resource": "arn:aws:timestream:us-east-2:111122223333:database/mydatabase/
table/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Owner": "${aws:username}"
                }
            }
        }
    ]
}
```

Puoi allegare questa politica agli IAM utenti del tuo account. Se un utente denominato `richard-roe` tenta di visualizzare un flusso temporale per la LiveAnalytics tabella, la tabella deve essere contrassegnata con `Owner=richard-roe` o. `owner=richard-roe` In caso contrario, gli viene negato l'accesso. La chiave di tag di condizione `Owner` corrisponde a `Owner` e `owner` perché i nomi delle chiavi di condizione non effettuano la distinzione tra maiuscole e minuscole. Per ulteriori informazioni, consulta Elementi IAM JSON della politica: Condizione nella Guida per l'IAMutente.

La seguente politica concede le autorizzazioni a un utente per creare tabelle con tag se il tag passato nella richiesta ha una chiave `Owner` e un valore: `username`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateTagTableUser",
            "Effect": "Allow",
            "Action": [
                "timestream:Create",
                "timestream:TagResource"
            ],
            "Resource": "arn:aws:timestream:us-east-2:111122223333:database/mydatabase/
table/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:RequestTag/Owner": "${aws:username}"
                }
            }
        }
    ]
}
```

La politica seguente consente l'uso di `DescribeDatabase` API su qualsiasi database con il `env` tag impostato su uno dei seguenti o`dev`: `test`

```
{ "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowDescribeEndpoints",
      "Effect": "Allow",
      "Action": [
        "timestream:DescribeEndpoints"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowDevTestAccess",
      "Effect": "Allow",
      "Action": [
        "timestream:DescribeDatabase"
      ],
```

```
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "timestream:tag/env": [
              "dev",
              "test"
          ]
        }
      }
    }
  ]
}
{ "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowTagAccessForDevResources",
      "Effect": "Allow",
      "Action": [
        "timestream:TagResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/env": [
            "test",
            "dev"
          ]
        }
      }
    }
  ]
}
```

Questa politica utilizza una `Condition` chiave per consentire l'aggiunta o l'aggiunta di un tag con la chiave `env` e il valore di `test` o `dev` a una risorsa. `qa`

Interrogazioni pianificate

Elenca, elimina, aggiorna, esegui ScheduledQuery

La seguente politica di esempio consente a un utente di elencare, eliminare, aggiornare ed eseguire query pianificate.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:DeleteScheduledQuery",
                "timestream:ExecuteScheduledQuery",
                "timestream:UpdateScheduledQuery",
                "timestream:ListScheduledQueries",
                "timestream:DescribeEndpoints"
            ],
            "Resource": "*"
        }
    ]
}
```

CreateScheduledQuery utilizzando una chiave gestita KMS dal cliente

La seguente politica di esempio consente a un utente di creare una query pianificata crittografata utilizzando una KMS chiave gestita dal cliente; *<keyid for ScheduledQuery>*.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::123456789012:role/ScheduledQueryExecutionRole"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "timestream:CreateScheduledQuery",
                "timestream:DescribeEndpoints"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
```

```
            "Action": [
                "kms:DescribeKey",
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:us-west-2:123456789012:key/<keyid for
 ScheduledQuery>",
            "Effect": "Allow"
        }
    ]
}
```

DescribeScheduledQuery utilizzando una KMS chiave gestita dal cliente

La seguente politica di esempio consente a un utente di descrivere una query pianificata creata utilizzando una KMS chiave gestita dal cliente; *<keyid for ScheduledQuery>*.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "timestream:DescribeScheduledQuery",
                "timestream:DescribeEndpoints"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-west-2:123456789012:key/<keyid for
 ScheduledQuery>",
            "Effect": "Allow"
        }
    ]
}
```

Autorizzazioni per il ruolo di esecuzione (utilizzo di una KMS chiave gestita dal cliente per le query pianificate e SSE - KMS per le segnalazioni di errori)

Allega la seguente politica di esempio al IAM ruolo specificato nel `ScheduledQueryExecutionRoleArn` parametro, `CreateScheduledQuery` API che utilizza la

KMS chiave gestita dal cliente per la crittografia delle query pianificate e la `SSE-KMS` crittografia per le segnalazioni di errori.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "kms:GenerateDataKey",
            ],
            "Resource": "arn:aws:kms:us-west-2:123456789012:key/<keyid for
 ScheduledQuery>",
            "Effect": "Allow"
        },
        {
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-west-2:123456789012:key/<keyid for database-1>",
                "arn:aws:kms:us-west-2:123456789012:key/<keyid for database-n>",
                "arn:aws:kms:us-west-2:123456789012:key/<keyid for ScheduledQuery>"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "sns:Publish"
            ],
            "Resource": [
                "arn:aws:sns:us-west-2:123456789012:scheduled-query-notification-topic-
*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "timestream:Select",
                "timestream:SelectValues",
                "timestream:WriteRecords"
            ],
            "Resource": "*",
            "Effect": "Allow"
```

```
        },
        {
            "Action": [
                "s3:PutObject",
                "s3:GetBucketAcl"
            ],
            "Resource": [
                "arn:aws:s3:::scheduled-query-error-bucket",
                "arn:aws:s3:::scheduled-query-error-bucket/*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

Ruolo di esecuzione, relazione di fiducia.

Di seguito è riportata la relazione di trust per il IAM ruolo specificato nel `ScheduledQueryExecutionRoleArn` parametro di `CreateScheduledQuery`API.

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

Consenti l'accesso a tutte le interrogazioni pianificate create all'interno di un account

Allega la seguente politica di esempio al IAM ruolo specificato nel `ScheduledQueryExecutionRoleArn` parametro, of `CreateScheduledQuery`API, per consentire l'accesso a tutte le query pianificate create all'interno di un account *`Account_ID`*.

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "timestream.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "Account_ID"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:timestream:us-
west-2:Account_ID:scheduled-query/*"
                }
            }
        }
    ]
}
```

Consenti l'accesso a tutte le query pianificate con un nome specifico

Allega la seguente policy di esempio al IAM ruolo specificato nel `ScheduledQueryExecutionRoleArn` parametro, of `CreateScheduledQuery`API, per consentire l'accesso a tutte le query pianificate con un nome che inizia con *`Scheduled_Query_Name`*, all'interno dell'account *`Account_ID`*.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "timestream.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "Account_ID"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:timestream:us-
west-2:Account_ID:scheduled-query/Scheduled_Query_Name*"
```

```
                }
            }
        }
    ]
}
```

Risoluzione dei problemi relativi all'identità e all'accesso di Amazon Timestream LiveAnalytics

Utilizza le seguenti informazioni per aiutarti a diagnosticare e risolvere i problemi più comuni che potresti riscontrare quando lavori con Timestream per e. LiveAnalytics IAM

Argomenti

- Non sono autorizzato a eseguire un'azione in Timestream per LiveAnalytics
- Non sono autorizzato a eseguire iam: PassRole
- Desidero consentire a persone esterne al mio AWS account di accedere al mio Timestream per ottenere risorse LiveAnalytics

Non sono autorizzato a eseguire un'azione in Timestream per LiveAnalytics

Se ti AWS Management Console dice che non sei autorizzato a eseguire un'azione, devi contattare l'amministratore per ricevere assistenza. L'amministratore è colui che ti ha fornito le credenziali di accesso.

L'errore di esempio seguente si verifica quando l'`mateojacksonIAM`utente tenta di utilizzare la console per visualizzare i dettagli relativi a *`table`* ma non dispone `timestream:`*`Select`* delle autorizzazioni per la tabella.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 timestream:Select on resource: mytable
```

In questo caso, Mateo richiede al suo amministratore di aggiornare le policy per poter accedere alla risorsa *`mytable`* utilizzando l'azione `timestream:`*`Select`*.

Non sono autorizzato a eseguire iam: PassRole

Se ricevi un messaggio di errore indicante che non sei autorizzato a eseguire l'`iam:PassRole`azione, le tue politiche devono essere aggiornate per consentirti di assegnare un ruolo a Timestream for. LiveAnalytics

Alcuni Servizi AWS consentono di trasferire un ruolo esistente a quel servizio invece di creare un nuovo ruolo di servizio o un ruolo collegato al servizio. Per eseguire questa operazione, è necessario disporre delle autorizzazioni per trasmettere il ruolo al servizio.

L'errore di esempio seguente si verifica quando un IAM utente denominato `marymajor` tenta di utilizzare la console per eseguire un'azione in Timestream for. LiveAnalytics Tuttavia, l'azione richiede che il servizio disponga delle autorizzazioni concesse da un ruolo di servizio. Mary non dispone delle autorizzazioni per passare il ruolo al servizio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In questo caso, le policy di Mary devono essere aggiornate per poter eseguire l'operazione `iam:PassRole`.

Se hai bisogno di aiuto, contatta l'amministratore AWS . L'amministratore è la persona che ti ha fornito le credenziali di accesso.

Desidero consentire a persone esterne al mio AWS account di accedere al mio Timestream per ottenere risorse LiveAnalytics

È possibile creare un ruolo con il quale utenti in altri account o persone esterne all'organizzazione possono accedere alle tue risorse. È possibile specificare chi è attendibile per l'assunzione del ruolo. Per i servizi che supportano politiche basate sulle risorse o liste di controllo degli accessi (ACLs), puoi utilizzare tali politiche per consentire alle persone di accedere alle tue risorse.

Per ulteriori informazioni, consulta gli argomenti seguenti:

- Per sapere se Timestream for LiveAnalytics supporta queste funzionalità, consulta. Come funziona Amazon Timestream for LiveAnalytics con IAM
- Per sapere come fornire l'accesso alle risorse di tua proprietà, consulta Fornire l'accesso a un IAM utente di un altro Account AWS utente di tua proprietà nella Guida per l'IAMutente. Account AWS
- Per scoprire come fornire l'accesso alle tue risorse a terze parti Account AWS, consulta Fornire l'accesso a persone Account AWS di proprietà di terzi nella Guida per l'IAMutente.
- Per informazioni su come fornire l'accesso tramite la federazione delle identità, consulta Fornire l'accesso agli utenti autenticati esternamente (federazione delle identità) nella Guida per l'IAMutente.

- Per conoscere la differenza tra l'utilizzo di ruoli e politiche basate sulle risorse per l'accesso tra account diversi, consulta la sezione Accesso alle risorse tra account nella Guida per l'utente. IAM IAM

Registrazione e monitoraggio in Timestream per LiveAnalytics

Il monitoraggio è una parte importante del mantenimento dell'affidabilità, della disponibilità e delle prestazioni di Timestream for LiveAnalytics e delle vostre soluzioni. AWS È necessario raccogliere i dati di monitoraggio da tutte le parti della AWS soluzione in modo da poter eseguire più facilmente il debug di un errore multipunto, se si verifica. Tuttavia, prima di iniziare a monitorare Timestream for LiveAnalytics, è necessario creare un piano di monitoraggio che includa le risposte alle seguenti domande:

- Quali sono gli obiettivi del monitoraggio?
- Di quali risorse si intende eseguire il monitoraggio?
- Con quale frequenza sarà eseguito il monitoraggio di queste risorse?
- Quali strumenti di monitoraggio verranno utilizzati?
- Chi eseguirà i processi di monitoraggio?
- Chi deve ricevere una notifica quando si verifica un problema?

Il passaggio successivo consiste nello stabilire una linea di base per il normale Timestream per LiveAnalytics le prestazioni nell'ambiente in uso, misurando le prestazioni in diversi momenti e in diverse condizioni di carico. Durante il monitoraggio di Timestream LiveAnalytics, memorizzate i dati di monitoraggio storici in modo da poterli confrontare con i dati sulle prestazioni correnti, identificare i modelli di prestazioni normali e le anomalie delle prestazioni e ideare metodi per risolvere i problemi.

Per stabilire una baseline, devi monitorare almeno gli elementi seguenti:

- Errori di sistema, in modo da poter determinare se le richieste hanno generato un errore.

Argomenti

- Strumenti di monitoraggio
- Registrazione del timestream per le chiamate con LiveAnalytics API AWS CloudTrail

Strumenti di monitoraggio

AWS fornisce vari strumenti che è possibile utilizzare per monitorare Timestream per. LiveAnalytics Alcuni di questi strumenti possono essere configurati in modo che eseguano automaticamente il monitoraggio, mentre altri richiedono l'intervento manuale. Si consiglia di automatizzare il più possibile i processi di monitoraggio.

Argomenti

Strumenti di monitoraggio automatici

Puoi utilizzare i seguenti strumenti di monitoraggio automatizzato per guardare Timestream LiveAnalytics e segnalare quando qualcosa non va:

- Amazon CloudWatch Alarms: monitora una singola metrica in un periodo di tempo specificato ed esegui una o più azioni in base al valore della metrica rispetto a una determinata soglia in diversi periodi di tempo. L'azione è una notifica inviata a un argomento di Amazon Simple Notification Service (AmazonSNS) o a una politica di Amazon EC2 Auto Scaling. CloudWatch gli allarmi non richiamano azioni semplicemente perché si trovano in uno stato particolare; lo stato deve essere cambiato e mantenuto per un determinato numero di periodi. Per ulteriori informazioni, consulta Monitoraggio con Amazon CloudWatch.

Strumenti di monitoraggio manuali

Un'altra parte importante del monitoraggio di Timestream for LiveAnalytics riguarda il monitoraggio manuale degli elementi che gli CloudWatch allarmi non coprono. Il Timestream for LiveAnalytics, CloudWatch Trusted Advisor, e altri AWS Management Console dashboard forniscono una at-a-glance visione dello stato dell'ambiente. AWS

- La CloudWatch home page mostra quanto segue:
  - Stato e allarmi attuali
  - Grafici degli allarmi e delle risorse
  - Stato di integrità dei servizi

  Inoltre, è possibile utilizzare CloudWatch per effettuare le seguenti operazioni:

- Crea pannelli di controllo personalizzati per monitorare i servizi di interesse.
- Crea grafici dei dati dei parametri per la risoluzione di problemi e il rilevamento di tendenze.
- Cerca e sfoglia tutte le metriche AWS delle tue risorse
- Crea e modifica gli allarmi per ricevere le notifiche dei problemi.

Registrazione del timestream per le chiamate con LiveAnalytics API AWS CloudTrail

Timestream for LiveAnalytics è integrato con AWS CloudTrail, un servizio che fornisce una registrazione delle azioni intraprese da un utente, ruolo o un AWS servizio in Timestream for. LiveAnalytics CloudTrail acquisisce le API chiamate Data Definition Language (DDL) per Timestream come eventi. LiveAnalytics Le chiamate acquisite includono le chiamate da Timestream per LiveAnalytics console e le chiamate in codice verso Timestream per le operazioni. LiveAnalytics API Se crei un trail, puoi abilitare la distribuzione continua di CloudTrail eventi a un bucket Amazon Simple Storage Service (Amazon S3), inclusi gli eventi per Timestream for. LiveAnalytics Se non configuri un percorso, puoi comunque visualizzare gli eventi più recenti sulla CloudTrail console nella cronologia degli eventi. Utilizzando le informazioni raccolte da CloudTrail, puoi determinare per quale richiesta è stata effettuata a Timestream LiveAnalytics, l'indirizzo IP da cui è stata effettuata la richiesta, chi ha effettuato la richiesta, quando è stata effettuata e altri dettagli.

Per ulteriori informazioni CloudTrail, consulta la Guida per l'AWS CloudTrail utente.

Timestream per informazioni in LiveAnalytics CloudTrail

CloudTrail è abilitato sul tuo AWS account al momento della creazione dell'account. Quando si verifica un'attività in Timestream for LiveAnalytics, tale attività viene registrata in un CloudTrail evento insieme ad altri eventi di AWS servizio nella cronologia degli eventi. È possibile visualizzare, cercare e scaricare gli eventi recenti nell'account AWS . Per ulteriori informazioni, consulta Visualizzazione degli eventi con CloudTrail la cronologia degli eventi.

> **Warning**
>
> Attualmente, Timestream for LiveAnalytics genera CloudTrail eventi per tutte le `Query` API operazioni e la gestione, ma non genera eventi per `WriteRecords` e. `DescribeEndpoints` APIs

Per una registrazione continua degli eventi nel tuo AWS account, inclusi gli eventi per Timestream for LiveAnalytics, crea un percorso. Un trail consente di CloudTrail inviare file di log a un bucket Amazon S3. Per impostazione predefinita, quando crei un percorso nella console, il percorso si applica a tutte le AWS regioni. Il trail registra gli eventi di tutte le regioni della AWS partizione e consegna i file di log al bucket Amazon S3 specificato. Inoltre, puoi configurare altri AWS servizi per analizzare ulteriormente e agire in base ai dati sugli eventi raccolti nei log. CloudTrail

Per ulteriori informazioni, consulta gli argomenti seguenti nella Guida per l'utente di AWS CloudTrail :

- Panoramica della creazione di un trail
- CloudTrail Servizi e integrazioni supportati
- Configurazione di Amazon SNS Notifications per CloudTrail
- Ricezione di file di CloudTrail registro da più regioni
- Ricezione di file di CloudTrail registro da più account
- Registrazione degli eventi relativi ai dati

Ogni evento o voce di log contiene informazioni sull'utente che ha generato la richiesta. Le informazioni di identità consentono di determinare quanto segue:

- Se la richiesta è stata effettuata con credenziali utente root o AWS Identity and Access Management (IAM)
- Se la richiesta è stata effettuata con le credenziali di sicurezza temporanee per un ruolo o un utente federato.
- Se la richiesta è stata effettuata da un altro AWS servizio

Per ulteriori informazioni, consulta l'CloudTrail userIdentityelemento.

Per `Query` API gli eventi:

- Crea un percorso che riceva tutti gli eventi o seleziona gli eventi con Timestream per il tipo di LiveAnalytics `AWS::Timestream::Database` risorsa o. `AWS::Timestream::Table`
- `Query`APIIle richieste che non accedono a nessun database o tabella o che determinano un'eccezione di convalida dovuta a una stringa di query non valida vengono registrate CloudTrail con un tipo di risorsa `AWS::Timestream::Database` e un valore di: ARN

```
arn:aws:timestream:(region):(accountId):database/NO_RESOURCE_ACCESSED
```

Questi eventi vengono consegnati solo ai trail che ricevono eventi con tipo di risorsa. `AWS::Timestream::Database`

Resilienza in Amazon Timestream Live Analytics

L'infrastruttura AWS globale è costruita attorno AWS a regioni e zone di disponibilità. AWS Le regioni forniscono più zone di disponibilità fisicamente separate e isolate, collegate con reti a bassa latenza, ad alto throughput e altamente ridondanti. Con le zone di disponibilità, puoi progettare e gestire applicazioni e database che eseguono automaticamente il failover tra zone di disponibilità senza interruzioni. Le zone di disponibilità sono più disponibili, tolleranti ai guasti e scalabili rispetto alle infrastrutture a data center singolo o multiplo tradizionali.

Per ulteriori informazioni su AWS regioni e zone di disponibilità, consulta Global Infrastructure.AWS

Per informazioni sulla funzionalità di protezione dei dati per Timestream disponibile tramite AWS Backup, vedere. Lavorare con AWS Backup

Sicurezza dell'infrastruttura in Amazon Timestream Live Analytics

In quanto servizio gestito, Amazon Timestream Live Analytics è protetto dalle AWS procedure di sicurezza di rete globali descritte nel white paper Amazon Web Services: Overview of Security Processes.

Utilizzi le API chiamate AWS pubblicate per accedere a Timestream Live Analytics attraverso la rete. I client devono supportare Transport Layer Security (TLS) 1.0 o versione successiva. Consigliamo TLS 1.2 o versioni successive. I client devono inoltre supportare suite di crittografia con Perfect Forward Secrecy (PFS) come Ephemeral Diffie-Hellman () o Elliptic Curve Ephemeral Diffie-Hellman (). DHE ECDHE La maggior parte dei sistemi moderni, come Java 7 e versioni successive, supporta tali modalità.

Inoltre, le richieste devono essere firmate utilizzando un ID chiave di accesso e una chiave di accesso segreta associata a un principale. IAM O puoi utilizzare AWS Security Token Service (AWS STS) per generare credenziali di sicurezza temporanee per sottoscrivere le richieste.

Timestream Live Analytics è progettato in modo tale che il traffico sia isolato AWS nella regione specifica in cui risiede l'istanza di Timestream Live Analytics.

Analisi della configurazione e delle vulnerabilità in Timestream

La configurazione e i controlli IT sono una responsabilità condivisa tra voi AWS e voi, i nostri clienti. Per ulteriori informazioni, consulta il modello di responsabilità AWS condivisa. Oltre al modello di responsabilità condivisa, Timestream for LiveAnalytics user deve tenere conto di quanto segue:

- È responsabilità del cliente applicare patch alle applicazioni client con le relative dipendenze lato client.
- I clienti dovrebbero prendere in considerazione i test di penetrazione, se appropriato (vedere https://aws.amazon.com/security/ penetration-testing/.)

Risposta agli incidenti in Timestream per LiveAnalytics

Gli incidenti di servizio di Amazon Timestream LiveAnalytics for Service sono segnalati nella Personal Health Dashboard. Puoi saperne di più sulla dashboard e qui. AWS Health

Timestream for LiveAnalytics supporta la reportistica utilizzando. AWS CloudTrail Per ulteriori informazioni, consulta Registrazione del timestream per le chiamate con LiveAnalytics API AWS CloudTrail.

VPCendpoint ()AWS PrivateLink

Puoi stabilire una connessione privata tra te VPC e Amazon Timestream creando un endpoint LiveAnalytics di interfaccia. VPC Gli endpoint di interfaccia sono alimentati da AWS PrivateLink, una tecnologia che consente di accedere privatamente a Timestream LiveAnalytics APIs senza un gateway Internet, un NAT dispositivo, una VPN connessione o una connessione Direct Connect AWS . Le istanze in uso VPC non necessitano di indirizzi IP pubblici per comunicare con Timestream. LiveAnalytics APIs Il traffico tra il tuo VPC e Timestream for LiveAnalytics non esce dalla rete Amazon.

Ogni endpoint dell'interfaccia è rappresentato da una o più interfacce di rete elastiche nelle tue sottoreti. Per ulteriori informazioni sugli VPC endpoint dell'interfaccia, consulta Interface VPC endpoints (AWS PrivateLink) nella Amazon VPC User Guide.

Per iniziare a usare Timestream for LiveAnalytics and VPC endpoints, abbiamo fornito informazioni su considerazioni specifiche per Timestream for LiveAnalytics with VPC endpoints, sulla creazione di un endpoint di interfaccia per Timestream for, sulla creazione di una policy sugli VPC endpoint per Timestream for LiveAnalytics e sull'utilizzo del client Timestream (per Write o Query) con gli VPC LiveAnalytics endpoint. SDK VPC

Argomenti

Come VPC funzionano gli endpoint con Timestream

Quando crei un VPC endpoint per accedere a Timestream Write o Timestream QuerySDK, tutte le richieste vengono indirizzate agli endpoint all'interno della rete Amazon e non accedono alla rete Internet pubblica. Più specificamente, le tue richieste vengono indirizzate agli endpoint di scrittura e interrogazione della cella su cui è stato mappato il tuo account per una determinata regione. Per saperne di più sull'architettura cellulare di Timestream e sugli endpoint specifici delle celle, puoi fare riferimento a. Architettura cellulare Ad esempio, supponiamo che il tuo account sia stato mappato su `cell1` in `us-west-2` e che tu abbia impostato gli endpoint VPC dell'interfaccia per write () e queries (). `ingest-cell1.timestream.us-west-2.amazonaws.com` `query-cell1.timestream.us-west-2.amazonaws.com` In questo caso, tutte le richieste di scrittura inviate utilizzando questi endpoint rimarranno interamente all'interno della rete Amazon e non accederanno alla rete Internet pubblica.

Considerazioni sugli endpoint Timestream VPC

Considera quanto segue quando crei un VPC endpoint per Timestream:

- Prima di configurare un VPC endpoint di interfaccia per Timestream for LiveAnalytics, assicurati di esaminare le proprietà e le limitazioni degli endpoint dell'interfaccia nella Amazon User Guide. VPC
- Timestream for LiveAnalytics supporta l'esecuzione di chiamate a tutte le sue azioni dal tuo. API VPC
- VPCIe politiche degli endpoint sono supportate per Timestream for. LiveAnalytics Per impostazione predefinita, l'accesso completo a Timestream for LiveAnalytics è consentito tramite l'endpoint. Per ulteriori informazioni, consulta Controllare l'accesso ai servizi con VPC endpoint nella Amazon VPC User Guide.
- A causa dell'architettura di Timestream, l'accesso alle azioni Write e Query richiede la creazione di due endpoint di VPC interfaccia, uno per ciascuno. SDK Inoltre, devi specificare un endpoint di cella (potrai creare un endpoint solo per la cella Timestream a cui sei mappato). Informazioni dettagliate sono disponibili nella sezione Crea un VPC endpoint di interfaccia per Timestream di questa guida. LiveAnalytics

Ora che hai capito come LiveAnalytics funziona Timestream for con gli endpoint, crea un VPC endpoint di interfaccia per Timestream for. VPC LiveAnalytics

Creazione di un endpoint di interfaccia per Timestream per VPC LiveAnalytics

Puoi creare un VPCendpoint di interfaccia per il LiveAnalytics servizio Timestream for utilizzando la VPC console Amazon o il AWS Command Line Interface ().AWS CLI Per creare un VPC endpoint per Timestream, completa i passaggi specifici di Timestream descritti di seguito.

> **Note**
>
> Prima di completare i passaggi seguenti, assicurati di aver compreso le considerazioni specifiche per gli endpoint Timestream. VPC

Creazione di un nome di servizio VPC endpoint utilizzando la cella Timestream

A causa dell'architettura unica di Timestream, è necessario creare endpoint di VPC interfaccia separati per ciascuno (Write e Query). SDK Inoltre, devi specificare un endpoint della cella Timestream (potrai creare un endpoint solo per la cella Timestream a cui sei mappato). Per utilizzare Interface VPC Endpoints per connetterti direttamente a Timestream dall'interno del tuo, completa i passaggi seguenti: VPC

1. Innanzitutto, trova un endpoint di cella Timestream disponibile. Per trovare un endpoint di cella disponibile, usa l'`DescribeEndpoints`azione (disponibile sia tramite Write che QueryAPIs) per elencare gli endpoint di cella disponibili nel tuo account Timestream. Vedi l'esempio per ulteriori dettagli.
2. Dopo aver selezionato un endpoint di cella da utilizzare, crea una stringa di endpoint di VPC interfaccia per Timestream Write o Query: API
   - Per il Write: API

     ```
     com.amazonaws.<region>.timestream.ingest-<cell>
     ```

   - Per l'interrogazioneAPI:

     ```
     com.amazonaws.<region>.timestream.query-<cell>
     ```

dove *<region>* è un codice AWS regionale valido e *<cell>* è uno degli indirizzi degli endpoint delle celle (ad esempio `cell1` `ocell2`) restituiti nell'oggetto Endpoints dall'azioneDescribeEndpoints . Vedi l'esempio per ulteriori dettagli.

3. Ora che hai creato un nome di servizio per l'VPCendpoint, crea un endpoint di interfaccia. Quando ti viene chiesto di fornire un nome di servizio per l'VPCendpoint, usa il nome del servizio VPC endpoint che hai creato nel passaggio 2.

Esempio: creazione del nome del servizio endpoint VPC

Nell'esempio seguente, l'`DescribeEndpoints`azione viene eseguita AWS CLI utilizzando la regione Write API in the`us-west-2`:

```
aws timestream-write describe-endpoints --region us-west-2
```

Questo comando restituirà il seguente risultato:

```
{
    "Endpoints": [
        {
            "Address": "ingest-cell1.timestream.us-west-2.amazonaws.com",
            "CachePeriodInMinutes": 1440
        }
    ]
}
```

In questo caso *cell1* è il *<cell>* e *us-west-2* è il *<region>*. Quindi, il nome del servizio VPC endpoint risultante sarà simile a:

```
com.amazonaws.us-west-2.timestream.ingest-cell1
```

Ora che hai creato un VPC endpoint di interfaccia per Timestream for LiveAnalytics, crea una policy di VPC endpoint per Timestream for. LiveAnalytics

Creazione di una policy sugli endpoint per Timestream per VPC LiveAnalytics

Puoi allegare una policy per gli endpoint al tuo VPC endpoint che controlli l'accesso a Timestream for. LiveAnalytics La policy specifica le informazioni riportate di seguito:

- Il principale che può eseguire operazioni.
- Le azioni che possono essere eseguite.
- Le risorse sui cui si possono eseguire azioni.

Per ulteriori informazioni, consulta Controllare l'accesso ai servizi con VPC endpoint nella Amazon VPC User Guide.

Esempio: policy VPC sugli endpoint per Timestream for actions LiveAnalytics

Di seguito è riportato un esempio di policy sugli endpoint per Timestream for. LiveAnalytics Se associata a un endpoint, questa politica consente l'accesso al Timestream elencato per LiveAnalytics le azioni (in questo caso `ListDatabases`) per tutti i principali su tutte le risorse.

```
{
   "Statement":[
      {
         "Principal":"*",
         "Effect":"Allow",
         "Action":[
            "timestream:ListDatabases"
         ],
         "Resource":"*"
      }
   ]
}
```

Best practice di sicurezza per Amazon Timestream for LiveAnalytics

Amazon Timestream LiveAnalytics for offre una serie di funzionalità di sicurezza da prendere in considerazione durante lo sviluppo e l'implementazione delle proprie politiche di sicurezza. Le seguenti best practice sono linee guida generali e non rappresentano una soluzione di sicurezza completa. Poiché queste best practice potrebbero non essere appropriate o sufficienti per l'ambiente, gestiscile come considerazioni utili anziché prescrizioni.

Argomenti

- Timestream per le migliori pratiche di sicurezza preventiva LiveAnalytics

Timestream per le migliori pratiche di sicurezza preventiva LiveAnalytics

Le seguenti best practice possono aiutarti ad anticipare e prevenire gli incidenti di sicurezza in Timestream for. LiveAnalytics

Crittografia a riposo

Timestream for LiveAnalytics crittografa a riposo tutti i dati utente memorizzati nelle tabelle utilizzando le chiavi di crittografia archiviate in ().AWS Key Management ServiceAWS KMS Questo fornisce un livello aggiuntivo di protezione dei dati dagli accessi non autorizzati allo storage sottostante.

Timestream for LiveAnalytics utilizza una singola chiave di servizio predefinita (di AWS proprietàCMK) per crittografare tutte le tabelle. Se questa chiave non esiste, viene creata per te. Le chiavi predefinite del servizio non possono essere disabilitate. Per ulteriori informazioni, consulta Timestream for LiveAnalytics Encryption at Rest.

Usa IAM i ruoli per autenticare l'accesso a Timestream per LiveAnalytics

Gli utenti, le applicazioni e gli altri AWS servizi per cui accedere a Timestream devono includere LiveAnalytics credenziali valide AWS nelle loro richieste. AWS API Non è necessario archiviare AWS le credenziali direttamente nell'applicazione o nell'istanza. EC2 Si tratta di credenziali a lungo termine che non vengono automaticamente ruotate e, pertanto, potrebbero avere un impatto aziendale significativo se compromesse. Un IAM ruolo consente di ottenere chiavi di accesso temporanee che possono essere utilizzate per accedere a AWS servizi e risorse.

Per ulteriori informazioni, consulta la sezione relativa ai ruoli IAM.

Utilizza IAM le politiche per Timestream per LiveAnalytics l'autorizzazione di base

Quando concedi le autorizzazioni, sei tu a decidere chi le ottiene, per quale Timestream le LiveAnalytics APIs ottengono e le azioni specifiche che desideri consentire su tali risorse. L'implementazione dei privilegi minimi è fondamentale per ridurre i rischi di sicurezza e l'impatto che può risultare da errori o intenzioni dannose.

Associa le politiche di autorizzazione alle IAM identità (ovvero utenti, gruppi e ruoli) e quindi concedi le autorizzazioni per eseguire operazioni su Timestream per le risorse. LiveAnalytics

Puoi farlo usando quanto segue:

- AWS politiche gestite (predefinite)

- Policy gestite dal cliente
- autorizzazione basata su tag

Valutazione della crittografia lato client

Se memorizzi dati sensibili o riservati in Timestream for LiveAnalytics, potresti voler crittografare tali dati il più vicino possibile alla loro origine in modo che siano protetti per tutto il loro ciclo di vita. La crittografia dei dati sensibili in transito e in archivio aiuta a garantire che i dati in formato testo non crittografato non siano disponibili per terze parti.

Utilizzo di altri servizi

Amazon Timestream LiveAnalytics for si integra con una varietà di servizi e strumenti AWS di terze parti diffusi. Attualmente, Timestream for LiveAnalytics supporta integrazioni con quanto segue:

Argomenti

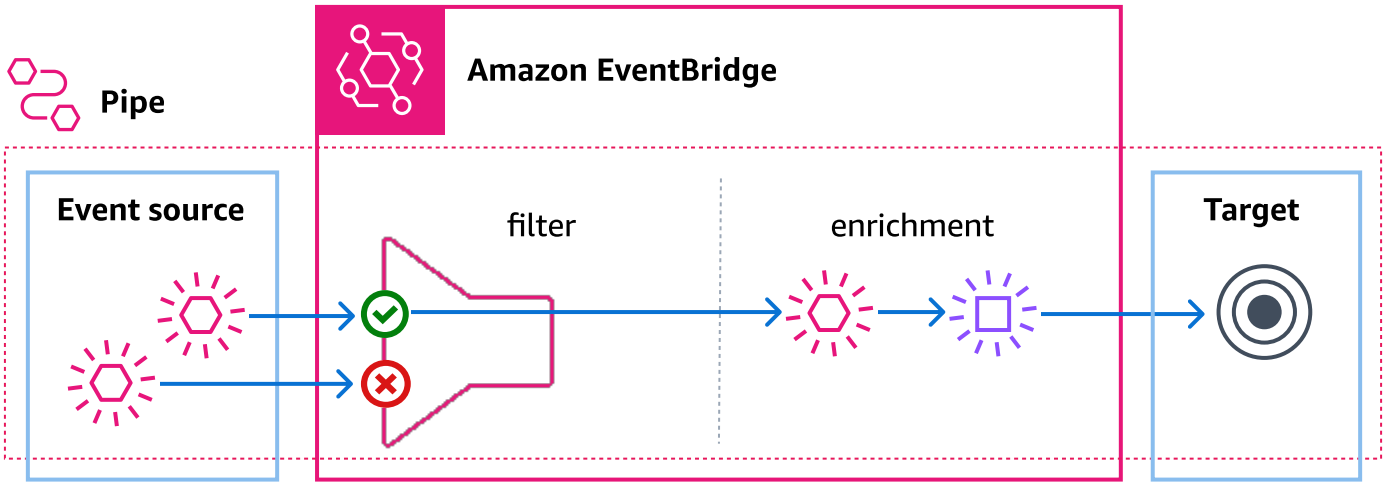
- Amazon DynamoDB
- AWS Lambda
- AWS IoT Core
- Servizio gestito da Amazon per Apache Flink
- Amazon Kinesis
- Amazon MQ
- Amazon MSK
- Amazon QuickSight
- Amazon SageMaker
- Amazon SQS
- Utilizzo DBeaver per lavorare con Amazon Timestream
- Grafana
- Utilizzo SquaredUp per lavorare con Amazon Timestream
- Telegraf open source
- JDBC
- ODBC
- VPCpunti finali ()AWS PrivateLink

Amazon DynamoDB

Utilizzo di EventBridge Pipes per inviare dati DynamoDB a Timestream

Puoi utilizzare EventBridge Pipes per inviare dati da un flusso DynamoDB a un Amazon Timestream for table. LiveAnalytics

Le pipe sono destinate point-to-point alle integrazioni tra sorgenti e destinazioni supportate, con supporto per trasformazioni e arricchimenti avanzati. Le pipe riducono la necessità di conoscenze specializzate e codice di integrazione durante lo sviluppo di architetture basate sugli eventi. Per configurare una pipe, si sceglie l'origine, si aggiungono filtri facoltativi, si definisce l'arricchimento facoltativo e si sceglie la destinazione per i dati dell'evento.



Per ulteriori informazioni su EventBridge Pipes, consultate EventBridge Pipes nella Guida EventBridge per l'utente. Per informazioni sulla configurazione di una pipe per inviare eventi a un Amazon LiveAnalytics Timestream for table EventBridge , consulta le specifiche del target Pipes.

AWS Lambda

Puoi creare funzioni Lambda che interagiscono con Timestream for. LiveAnalytics Ad esempio, puoi creare una funzione Lambda che viene eseguita a intervalli regolari per eseguire una query su Timestream e inviare una SNS notifica in base ai risultati della query che soddisfano uno o più criteri. Per ulteriori informazioni su Lambda, consulta la documentazione di Lambda AWS .

Argomenti

- Crea funzioni AWS Lambda usando Amazon Timestream for with Python LiveAnalytics

- Crea funzioni AWS Lambda utilizzando Amazon Timestream for with LiveAnalytics JavaScript
- Crea funzioni AWS Lambda utilizzando Amazon Timestream for with Go LiveAnalytics
- Crea funzioni AWS Lambda usando Amazon Timestream per C# LiveAnalytics

Crea funzioni AWS Lambda usando Amazon Timestream for with Python LiveAnalytics

Per creare funzioni AWS Lambda utilizzando Amazon Timestream LiveAnalytics for with Python, segui i passaggi seguenti.

1. Crea un IAM ruolo da far assumere a Lambda che conceda le autorizzazioni necessarie per accedere al servizio Timestream, come indicato in. Fornisci Timestream per l'accesso LiveAnalytics
2. Modifica la relazione di trust del IAM ruolo per aggiungere il servizio Lambda. Puoi usare i comandi seguenti per aggiornare un ruolo esistente in modo che AWS Lambda possa assumerlo:

   a. Crea il documento relativo alla politica di fiducia:

```
cat > Lambda-Role-Trust-Policy.json << EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "lambda.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

   b. Aggiorna il ruolo del passaggio precedente con il documento di fiducia

```
aws iam update-assume-role-policy --role-name <name_of_the_role_from_step_1> --
policy-document file://Lambda-Role-Trust-Policy.json
```

I riferimenti correlati sono disponibili in TimestreamWritee TimestreamQuery.

Crea funzioni AWS Lambda utilizzando Amazon Timestream for with LiveAnalytics JavaScript

Per creare funzioni AWS Lambda utilizzando Amazon Timestream LiveAnalytics for JavaScript with, segui le istruzioni riportate qui.

I riferimenti correlati sono disponibili su Timestream Write Client - AWS SDK per JavaScript v3 e Timestream Query Client - per v3. AWS SDK JavaScript

Crea funzioni AWS Lambda utilizzando Amazon Timestream for with Go LiveAnalytics

Per creare funzioni AWS Lambda utilizzando Amazon Timestream LiveAnalytics for with Go, segui le istruzioni riportate qui.

I riferimenti correlati sono disponibili su timestreamwrite e timestreamquery.

Crea funzioni AWS Lambda usando Amazon Timestream per C# LiveAnalytics

Per creare funzioni AWS Lambda utilizzando Amazon Timestream LiveAnalytics for with C#, segui le istruzioni descritte qui.

I riferimenti correlati sono disponibili su Amazon. TimestreamWritee Amazon. TimestreamQuery.

AWS IoT Core

Puoi raccogliere dati dai dispositivi IoT utilizzando AWS IoT Core e indirizzarli ad Amazon Timestream tramite azioni delle regole IoT Core. AWS Le azioni delle regole IoT specificano cosa fare quando viene attivata una regola. Puoi definire azioni per inviare dati a una tabella Amazon Timestream, a un database Amazon DynamoDB e richiamare una funzione Lambda. AWS

L'azione Timestream in IoT Rules viene utilizzata per inserire i dati dai messaggi in arrivo direttamente in Timestream. L'azione analizza i risultati dell'SQListruzione IoT Core e archivia i dati in Timestream. I nomi dei campi del set di SQL risultati restituito vengono utilizzati come measure: :name e il valore del campo è measure: :value.

Ad esempio, considera l'SQListruzione e il payload del messaggio di esempio:

```
SELECT temperature, humidity from 'iot/topic'
```

```
{
```

```
  "dataFormat": 5,
  "rssi": -88,
  "temperature": 24.04,
  "humidity": 43.605,
  "pressure": 101082,
  "accelerationX": 40,
  "accelerationY": -20,
  "accelerationZ": 1016,
  "battery": 3007,
  "txPower": 4,
  "movementCounter": 219,
  "device_id": 46216,
  "device_firmware_sku": 46216
}
```

Se viene creata un'azione della regola IoT Core per Timestream con l'SQListruzione precedente, verranno aggiunti due record a Timestream con i nomi delle misure temperatura e umidità e valori di misura rispettivamente di 24,04 e 43,605.

È possibile modificare il nome della misura di un record aggiunto a Timestream utilizzando l'operatore AS nell'istruzione. SELECT L'SQListruzione seguente creerà un record con il nome del messaggio temp anziché temperature.

Il tipo di dati della misura viene dedotto dal tipo di dati del valore del payload del messaggio. JSONi tipi di dati come integer, double, boolean e string vengono mappati ai tipi di dati Timestream di,, e rispettivamente. BIGINT DOUBLE BOOLEAN VARCHAR I dati possono anche essere impostati su tipi di dati specifici utilizzando la funzione cast (). È possibile specificare il timestamp della misura. Se il timestamp viene lasciato vuoto, viene utilizzata l'ora in cui la voce è stata elaborata.

Puoi fare riferimento alla documentazione sulle regole e le azioni di Timestream per ulteriori dettagli

Per creare un'azione della regola IoT Core per archiviare i dati in Timestream, procedi nel seguente modo:

Argomenti

- Prerequisiti
- Utilizzo della console
- Usando il CLI
- Applicazione di esempio
- Tutorial video

Prerequisiti

1. Crea un database in Amazon Timestream utilizzando le istruzioni descritte in. Creazione di un database
2. Crea una tabella in Amazon Timestream utilizzando le istruzioni descritte in. Creare una tabella

Utilizzo della console

1. Utilizza la console di AWS gestione per AWS IoT Core per creare una regola facendo clic su Gestisci > Routing dei messaggi > Regole seguito da Crea regola.
2. Imposta il nome della regola su un nome a tua scelta e poi sul testo SQL mostrato di seguito

```
SELECT temperature as temp, humidity from 'iot/topic'
```

3. Seleziona Timestream dall'elenco delle azioni
4. Specificate il database, la tabella e i nomi delle dimensioni Timestream insieme al ruolo per scrivere i dati in Timestream. Se il ruolo non esiste, puoi crearne uno facendo clic su Crea ruoli
5. Per testare la regola, segui le istruzioni mostrate qui.

Usando il CLI

Se non hai installato l'interfaccia a riga di AWS comando (AWS CLI), fallo da qui.

1. Salva il seguente payload della regola in un JSON file chiamato timestream_rule.json. Replace (Sostituisci) *arn:aws:iam::123456789012:role/TimestreamRole* con il tuo ruolo arn che garantisce l'accesso AWS IoT per archiviare i dati in Amazon Timestream

```
{
    "actions": [
            {
                "timestream": {
                    "roleArn": "arn:aws:iam::123456789012:role/TimestreamRole",
                    "tableName": "devices_metrics",
                    "dimensions": [
                        {
                            "name": "device_id",
                            "value": "${clientId()}"
                        },
```

```
                        {
                            "name": "device_firmware_sku",
                            "value": "My Static Metadata"
                        }
                    ],
                    "databaseName": "record_devices"
                }
            }
    ],
    "sql": "select * from 'iot/topic'",
    "awsIotSqlVersion": "2016-03-23",
    "ruleDisabled": false
}
```

2. Crea una regola tematica utilizzando il seguente comando

```
aws iot create-topic-rule --rule-name timestream_test --topic-rule-payload file://
<path/to/timestream_rule.json> --region us-east-1
```

3. Recupera i dettagli della regola dell'argomento utilizzando il seguente comando

```
aws iot get-topic-rule --rule-name timestream_test
```

4. Salva il seguente payload di messaggi in un file chiamato timestream_msg.json

```
{
  "dataFormat": 5,
  "rssi": -88,
  "temperature": 24.04,
  "humidity": 43.605,
  "pressure": 101082,
  "accelerationX": 40,
  "accelerationY": -20,
  "accelerationZ": 1016,
  "battery": 3007,
  "txPower": 4,
  "movementCounter": 219,
  "device_id": 46216,
  "device_firmware_sku": 46216
}
```

5. Verificate la regola utilizzando il seguente comando

```
aws iot-data publish --topic 'iot/topic' --payload file://<path/to/
timestream_msg.json>
```

Applicazione di esempio

Per aiutarti a iniziare a utilizzare Timestream con AWS IoT Core, abbiamo creato un'applicazione di esempio completamente funzionale che crea gli artefatti necessari in AWS IoT Core e Timestream per creare una regola tematica e un'applicazione di esempio per la pubblicazione di dati sull'argomento.

1. Clona il GitHub repository per l'applicazione di esempio per l'integrazione AWS IoT Core seguendo le istruzioni di GitHub
2. Segui le istruzioni riportate nella sezione READMEper utilizzare un AWS CloudFormation modello per creare gli artefatti necessari in Amazon Timestream e AWS IoT Core e pubblicare messaggi di esempio sull'argomento.

Tutorial video

Questo video spiega come IoT Core funziona con Timestream.

Servizio gestito da Amazon per Apache Flink

Puoi utilizzare Apache Flink per trasferire i dati delle serie temporali da Amazon Managed Service per Apache Flink, MSK Amazon, Apache Kafka e altre tecnologie di streaming direttamente in Amazon Timestream for. LiveAnalytics Abbiamo creato un connettore dati di esempio Apache Flink per Timestream. Abbiamo anche creato un'applicazione di esempio per l'invio di dati ad Amazon Kinesis in modo che i dati possano fluire da Kinesis a Managed Service for Apache Flink e infine ad Amazon Timestream. Tutti questi artefatti sono disponibili in. GitHub Questo video tutorial descrive la configurazione.

> **Note**
>
> Java 11 è la versione consigliata per l'utilizzo dell'applicazione Managed Service for Apache Flink. Se disponete di più versioni di Java, assicuratevi di esportare Java 11 nella variabile di HOME ambiente JAVA _.

Argomenti

Applicazione di esempio

Per iniziare, seguite la procedura seguente:

1. Crea un database in Timestream con il nome `kdaflink` seguendo le istruzioni descritte in Creazione di un database
2. Crea una tabella in Timestream con il nome `kinesisdata1` seguendo le istruzioni descritte in Creare una tabella
3. Crea un Amazon Kinesis Data Stream con il nome `TimestreamTestStream` seguendo le istruzioni descritte in Creazione di un flusso
4. Clona il GitHub repository per il connettore dati Apache Flink per Timestream seguendo le istruzioni di GitHub
5. Per compilare, eseguire e utilizzare l'applicazione di esempio, segui le istruzioni nel connettore di dati di esempio Apache Flink README
6. Compilate l'applicazione Managed Service for Apache Flink seguendo le istruzioni per la compilazione del codice dell'applicazione
7. Carica il file binario dell'applicazione Managed Service for Apache Flink seguendo le istruzioni per caricare il codice di streaming Apache Flink

    a. Dopo aver fatto clic su Crea applicazione, fai clic sul collegamento del ruolo dell'IAMapplicazione

    b. Allega le IAM politiche per AmazonKinesisReadOnlyAccesse AmazonTimestreamFullAccess.

    > **Note**
    >
    > Le IAM politiche di cui sopra non sono limitate a risorse specifiche e non sono adatte all'uso in produzione. Per un sistema di produzione, prendi in considerazione l'utilizzo di politiche che limitano l'accesso a risorse specifiche.

8. Clona il GitHub repository per l'applicazione di esempio che scrive i dati su Kinesis seguendo le istruzioni di GitHub

9. Segui le istruzioni riportate nella sezione README per eseguire l'applicazione di esempio per la scrittura di dati su Kinesis
10. Esegui una o più query in Timestream per assicurarti che i dati vengano inviati da Kinesis a Managed Service for Apache Flink to Timestream seguendo le istruzioni per Creare una tabella

Tutorial video

Questo video spiega come utilizzare Timestream con Managed Service per Apache Flink.

Amazon Kinesis

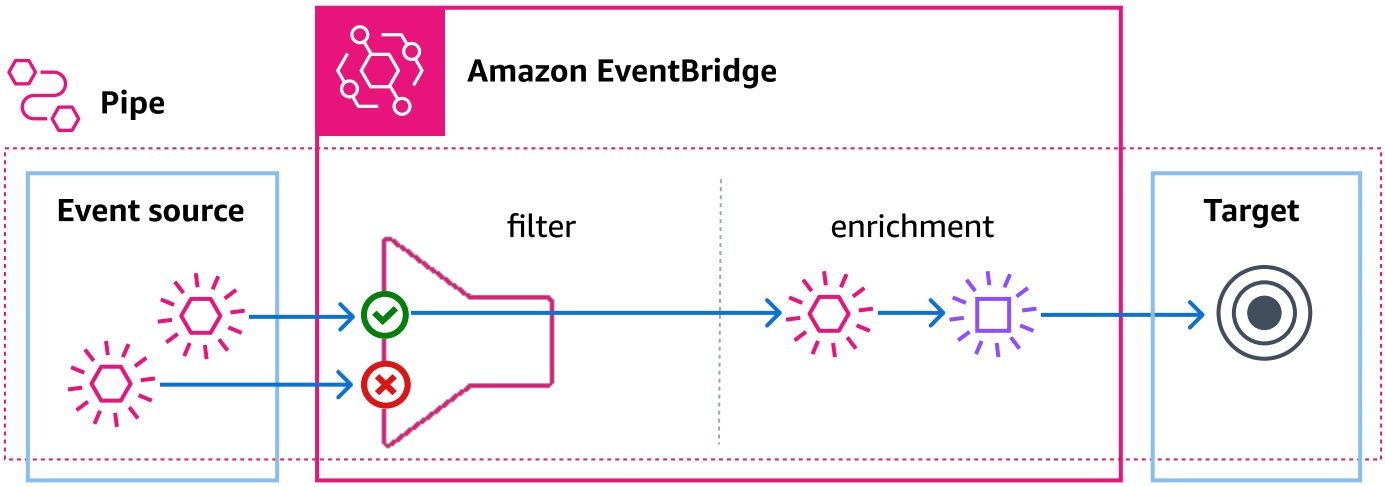
Usando Servizio gestito da Amazon per Apache Flink

È possibile inviare dati da Kinesis Data Streams a LiveAnalytics Timestream per utilizzare il connettore dati di esempio per Managed Service Timestream for Apache Flink. Per ulteriori informazioni, consulta Apache Servizio gestito da Amazon per Apache Flink Flink.

Utilizzo di EventBridge Pipes per inviare dati Kinesis a Timestream

Puoi utilizzare EventBridge Pipes per inviare dati da un flusso Kinesis a un Amazon Timestream for table. LiveAnalytics

Le pipe sono destinate point-to-point alle integrazioni tra sorgenti e destinazioni supportate, con supporto per trasformazioni e arricchimenti avanzati. Le pipe riducono la necessità di conoscenze specializzate e codice di integrazione durante lo sviluppo di architetture basate sugli eventi. Per configurare una pipe, si sceglie l'origine, si aggiungono filtri facoltativi, si definisce l'arricchimento facoltativo e si sceglie la destinazione per i dati dell'evento.



Questa integrazione consente di sfruttare la potenza delle funzionalità di analisi dei dati delle serie temporali Timestream di cui dispone, semplificando al contempo la pipeline di inserimento dei dati.

L'utilizzo di EventBridge Pipes with offre i seguenti vantaggi: Timestream

- Inserimento di dati in tempo reale: trasmetti i dati da Kinesis direttamente a Timestream per LiveAnalytics consentire analisi e monitoraggio in tempo reale.
- Integrazione perfetta: utilizza EventBridge Pipes per gestire il flusso di dati senza la necessità di complesse integrazioni personalizzate.
- Filtraggio e trasformazione migliorati: filtra o trasforma i record Kinesis prima che vengano archiviati Timestream per soddisfare i requisiti specifici di elaborazione dei dati.
- Scalabilità: gestisci flussi di dati ad alta velocità e garantisci un'elaborazione efficiente dei dati con funzionalità integrate di parallelismo e batch.

Configurazione

Per configurare una EventBridge Pipe per lo streaming di dati da Kinesis a Timestream, segui questi passaggi:

1. Creare un flusso Kinesis

   Assicurati di avere un flusso di dati Kinesis attivo da cui desideri importare i dati.
2. Crea un Timestream database e una tabella

   Configura il Timestream database e la tabella in cui verranno archiviati i dati.
3. Configura la EventBridge Pipe:
   - Sorgente: seleziona il tuo stream Kinesis come sorgente.
   - Obiettivo: scegli Timestream come destinazione.
   - Impostazioni di batch: definisci la finestra di batch e le dimensioni del batch per ottimizzare l'elaborazione dei dati e ridurre la latenza.

> **Important**
>
> Quando si configura una tubazione, si consiglia di verificare la correttezza di tutte le configurazioni inserendo alcuni record. Tieni presente che la corretta creazione di una tubazione non garantisce che la pipeline sia corretta e che i dati scorrano senza errori. Potrebbero verificarsi errori di runtime, come una tabella errata, un parametro del percorso

dinamico errato o un Timestream record non valido dopo l'applicazione della mappatura, che verranno rilevati quando i dati effettivi fluiranno attraverso la pipe.

Le seguenti configurazioni determinano la velocità con cui i dati vengono inseriti:

- BatchSize: La dimensione massima del batch per cui verrà inviato a Timestream. LiveAnalytics Intervallo: 0 - 100. Si consiglia di mantenere questo valore su 100 per ottenere la massima produttività.
- MaximumBatchingWindowInSeconds: Il tempo massimo di attesa per il riempimento batchSize prima che il batch venga inviato a Timestream per la destinazione. LiveAnalytics A seconda della frequenza degli eventi in arrivo, questa configurazione determinerà il ritardo di inserimento. Si consiglia di mantenere questo valore < 10 secondi per continuare a inviare i dati quasi in tempo reale. Timestream
- ParallelizationFactor: il numero di batch da elaborare contemporaneamente da ogni shard. Si consiglia di utilizzare il valore massimo di 10 per ottenere la massima produttività e un'ingestione quasi in tempo reale.

  Se lo stream viene letto da più destinatari, utilizza il fan-out avanzato per fornire un pubblico dedicato alla tua pipe e ottenere un throughput elevato. Per ulteriori informazioni, consulta la sezione Sviluppo di utenti con fan-out avanzati nella Guida per l' Kinesis Data Streams APIutente.Kinesis Data Streams

> **Note**
>
> Il throughput massimo che può essere raggiunto è limitato dalle esecuzioni simultanee di pipe per account.

La seguente configurazione garantisce la prevenzione della perdita di dati:

- DeadLetterConfig: Si consiglia di eseguire sempre la configurazione DeadLetterConfig in modo da evitare qualsiasi perdita di dati nei casi in cui gli eventi non possano essere importati in Timestream a LiveAnalytics causa di errori dell'utente.

Ottimizza le prestazioni della tua pipe con le seguenti impostazioni di configurazione, che aiutano a evitare che i record causino rallentamenti o blocchi.

- MaximumRecordAgeInSeconds: I record più vecchi di questo non verranno elaborati e verranno spostati direttamente in. DLQ Si consiglia di impostare questo valore in modo che non sia superiore al periodo di conservazione dell'archivio di memoria configurato per la Timestream tabella di destinazione.
- MaximumRetryAttempts: Il numero di tentativi di riprovare un record prima che il record venga inviato a DeadLetterQueue. Si consiglia di configurarlo a 10. Questo dovrebbe aiutare a risolvere eventuali problemi temporanei e, in caso di problemi persistenti, il record verrà spostato DeadLetterQueue e sbloccato nel resto dello stream.
- OnPartialBatchItemFailure: Per le fonti che supportano l'elaborazione parziale in batch, consigliamo di abilitarla e configurarla come AUTOMATIC _ BISECT per riprovare ulteriormente i record non riusciti prima di eliminare/inviarli a. DLQ

Esempio di configurazione

Ecco un esempio di come configurare un EventBridge Pipe per lo streaming di dati da un flusso Kinesis a una Timestream tabella:

Example IAM aggiornamenti delle politiche per Timestream

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream:WriteRecords"
            ],
            "Resource": [
                "arn:aws:timestream:us-east-1:123456789012:database/my-database/table/
my-table"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "timestream:DescribeEndpoints"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

Example Configurazione del flusso Kinesis

```
{
  "Source": "arn:aws:kinesis:us-east-1:123456789012:stream/my-kinesis-stream",
  "SourceParameters": {
    "KinesisStreamParameters": {
        "BatchSize": 100,
        "DeadLetterConfig": {
            "Arn": "arn:aws:sqs:us-east-1:123456789012:my-sqs-queue"
        },
       "MaximumBatchingWindowInSeconds": 5,
        "MaximumRecordAgeInSeconds": 1800,
        "MaximumRetryAttempts": 10,
        "StartingPosition": "LATEST",
       "OnPartialBatchItemFailure": "AUTOMATIC_BISECT"
    }
  }
}
```

Example Timestream configurazione del target

```
{
    "Target": "arn:aws:timestream:us-east-1:123456789012:database/my-database/table/my-
table",
    "TargetParameters": {
        "TimestreamParameters": {
            "DimensionMappings": [
                {
                    "DimensionName": "sensor_id",
                    "DimensionValue": "$.data.device_id",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "DimensionName": "sensor_type",
                    "DimensionValue": "$.data.sensor_type",
                    "DimensionValueType": "VARCHAR"
                },
                {
                    "DimensionName": "sensor_location",
                    "DimensionValue": "$.data.sensor_loc",
                    "DimensionValueType": "VARCHAR"
```

```
                }
            ],
            "MultiMeasureMappings": [
                {
                    "MultiMeasureName": "readings",
                    "MultiMeasureAttributeMappings": [
                        {
                            "MultiMeasureAttributeName": "temperature",
                            "MeasureValue": "$.data.temperature",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "MultiMeasureAttributeName": "humidity",
                            "MeasureValue": "$.data.humidity",
                            "MeasureValueType": "DOUBLE"
                        },
                        {
                            "MultiMeasureAttributeName": "pressure",
                            "MeasureValue": "$.data.pressure",
                            "MeasureValueType": "DOUBLE"
                        }
                    ]
                }
            ],
            "SingleMeasureMappings": [],
            "TimeFieldType": "TIMESTAMP_FORMAT",
            "TimestampFormat": "yyyy-MM-dd HH:mm:ss.SSS",
            "TimeValue": "$.data.time",
            "VersionValue": "$.approximateArrivalTimestamp"
        }
    }
}
```

trasformazione degli eventi

EventBridge Le pipe ti consentono di trasformare i dati prima che arrivino. TimestreamÈ possibile definire regole di trasformazione per modificare i Kinesis record in entrata, ad esempio cambiare i nomi dei campi.

Supponiamo che lo Kinesis stream contenga dati di temperatura e umidità. Puoi utilizzare una EventBridge trasformazione per rinominare questi campi prima di inserirli. Timestream

Best practice

Batching e buffering

- Configura la finestra e le dimensioni del batch per bilanciare la latenza di scrittura e l'efficienza di elaborazione.
- Utilizzate una finestra di batch per accumulare una quantità sufficiente di dati prima dell'elaborazione, riducendo il sovraccarico dovuto alla frequente presenza di piccoli batch.

Elaborazione parallela

Utilizzate l'ParallelizationFactorimpostazione per aumentare la concorrenza, in particolare per i flussi ad alta velocità. Ciò garantisce che più batch di ogni frammento possano essere elaborati contemporaneamente.

Trasformazione dei dati

Sfrutta le capacità di trasformazione di EventBridge Pipes per filtrare e migliorare i record prima di archiviarli. Timestream Questo può aiutare ad allineare i dati ai requisiti analitici.

Sicurezza

- Assicuratevi che i IAM ruoli utilizzati per EventBridge Pipes dispongano delle autorizzazioni necessarie per leggere Kinesis e scrivere. Timestream
- Utilizza misure di crittografia e controllo degli accessi per proteggere i dati in transito e a riposo.

Errori di debug

- Disattivazione automatica delle tubazioni

  Le pipe verranno disattivate automaticamente in circa 2 ore se la destinazione non esiste o presenta problemi di autorizzazione
- Throttles

  I tubi hanno la capacità di spegnersi automaticamente e riprovare fino a quando l'acceleratore non si riduce.
- Abilitazione dei registri

Ti consigliamo di abilitare i log a ERROR livello e includere i dati di esecuzione per ottenere maggiori informazioni sugli errori. In caso di errore, questi registri request/response sent/received conterranno from. Timestream Questo aiuta a comprendere l'errore associato e, se necessario, a rielaborare i record dopo averlo corretto.

Monitoraggio

Ti consigliamo di impostare allarmi su quanto segue per rilevare eventuali problemi con il flusso di dati:

- Età massima del record nella fonte
  - `GetRecords.IteratorAgeMilliseconds`
- Metriche dei guasti in Pipes
  - `ExecutionFailed`
  - `TargetStageFailed`
- Timestream Errori di scrittura API
  - `UserErrors`

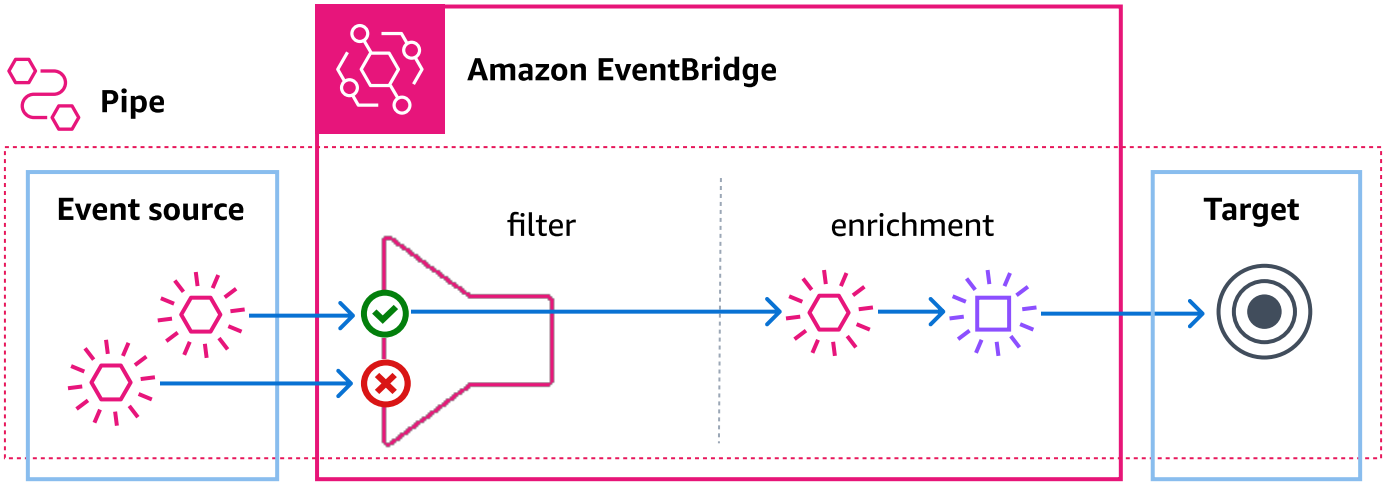
Per ulteriori metriche di monitoraggio, consulta Monitoraggio EventBridge nella Guida per l'EventBridge utente.

Amazon MQ

Utilizzo di EventBridge Pipes per inviare dati Amazon MQ a Timestream

Puoi utilizzare EventBridge Pipes per inviare dati da un broker Amazon MQ a un Amazon Timestream LiveAnalytics per tabelle.

Le pipe sono destinate point-to-point alle integrazioni tra sorgenti e destinazioni supportate, con supporto per trasformazioni e arricchimenti avanzati. Le pipe riducono la necessità di conoscenze specializzate e codice di integrazione durante lo sviluppo di architetture basate sugli eventi. Per configurare una pipe, si sceglie l'origine, si aggiungono filtri facoltativi, si definisce l'arricchimento facoltativo e si sceglie la destinazione per i dati dell'evento.



Per ulteriori informazioni su EventBridge Pipes, consultate EventBridge Pipes nella Guida EventBridge per l'utente. Per informazioni sulla configurazione di una pipe per inviare eventi a un Amazon LiveAnalytics Timestream for table EventBridge , consulta le specifiche del target Pipes.

Amazon MSK

Utilizzo di Managed Service for Apache Flink per inviare Amazon MSK dati a Timestream per LiveAnalytics

Puoi inviare dati Timestream da Amazon MSK a creando un connettore dati simile al connettore dati di esempio Timestream per Managed Service for Apache Flink. Per ulteriori informazioni, consulta Servizio gestito da Amazon per Apache Flink.

Utilizzo di Kafka Connect per inviare MSK dati Amazon a Timestream per LiveAnalytics

Puoi usare Kafka Connect per importare i dati delle tue serie temporali Amazon MSK direttamente da Timestream for. LiveAnalytics

Abbiamo creato un esempio di Kafka Sink Connector per. Timestream Abbiamo anche creato un esempio di piano di jMeter test Apache per la pubblicazione di dati su un argomento di Kafka, in modo che i dati possano fluire dall'argomento attraverso il Timestream Kafka Sink Connector, a un Timestream per tabella. LiveAnalytics Tutti questi artefatti sono disponibili su. GitHub

> **Note**
>
> Java 11 è la versione consigliata per l'utilizzo del Timestream Kafka Sink Connector. Se disponi di più versioni di Java, assicurati di esportare Java 11 nella variabile di ambiente _. JAVA HOME

Creazione di un'applicazione di esempio

Per iniziare, segui la procedura riportata di seguito.

1. In Timestream for LiveAnalytics, crea un database con il nome. `kafkastream`

   Consulta la procedura ??? per istruzioni dettagliate.
2. In Timestream for LiveAnalytics, crea una tabella con il nome. `purchase_history`

   Per istruzioni dettagliate, consulta ??? la procedura.
3. Segui le istruzioni condivise in per creare quanto segue:, e.

   - Un Amazon MSK cluster
   - Un' Amazon EC2 istanza configurata come macchina client di Kafka Producer
   - Un argomento di Kafka

   Consulta i prerequisiti del progetto kafka_ingestor per istruzioni dettagliate.
4. Clona il repository Kafka Sink Connector.Timestream

   Vedi Clonazione di un repository su per istruzioni dettagliate. GitHub
5. Compila il codice del plugin.

   Vedi Connector - Build from source on GitHub per istruzioni dettagliate.
6. Carica i seguenti file in un bucket S3: seguendo le istruzioni descritte in.

   - Il file jar (kafka-connector-timestream-> VERSION <- jar-with-dependencies .jar) dalla directory `/target`
   - Il file di schema json di esempio,. `purchase_history.json`

   Per istruzioni dettagliate, consulta Caricamento di oggetti nella Guida per Amazon S3 l'utente.

7. Crea due VPC endpoint. Questi endpoint verrebbero utilizzati dal MSK Connettore per accedere alle risorse utilizzando. AWS PrivateLink

   - Uno per accedere al bucket Amazon S3
   - Uno per accedere al Timestream per la tabella. LiveAnalytics

   Vedi VPCEndpoints per istruzioni dettagliate.
8. Crea un plugin personalizzato con il file jar caricato.

   Per istruzioni dettagliate, consulta Plugins nella Amazon MSK Developer Guide.
9. Crea una configurazione di worker personalizzata con il JSON contenuto descritto nei parametri di configurazione del lavoratore, seguendo le istruzioni descritte in

   Per istruzioni dettagliate, consulta Creazione di una configurazione di worker personalizzata nella Amazon MSK Developer Guide.
10. Crea un IAM ruolo di esecuzione del servizio.

    Vedi IAM Service Role per istruzioni dettagliate.
11. Crea un Amazon MSK connettore con il plug-in personalizzato, la configurazione personalizzata del worker e il IAM ruolo di esecuzione del servizio creati nei passaggi precedenti e con la configurazione del connettore di esempio.

    Per istruzioni dettagliate, consulta Creazione di un connettore nella Amazon MSK Developer Guide.

    Assicurati di aggiornare i valori dei seguenti parametri di configurazione con i rispettivi valori. Vedi Parametri di configurazione del connettore per i dettagli.

    - `aws.region`
    - `timestream.schema.s3.bucket.name`
    - `timestream.ingestion.endpoint`

    Il completamento della creazione del connettore richiede 5-10 minuti. La pipeline è pronta quando il suo stato cambia in. `Running`
12. Pubblica un flusso continuo di messaggi per scrivere dati sull'argomento Kafka creato.

    Vedi Come usarlo per istruzioni dettagliate.

13. Esegui una o più query per assicurarti che i dati vengano inviati da MSK Connect Amazon MSK to the Timestream for table. LiveAnalytics

    Consulta la procedura ??? per istruzioni dettagliate.

Risorse aggiuntive

Il blog, Real-time serverless data ingestion from your Kafka clusters in Timestream for using Kafka Connect, spiega come configurare una end-to-end pipeline utilizzando Timestream for LiveAnalytics Kafka Sink Connector, a partire da una macchina client LiveAnalytics Kafka Producer che utilizza il piano di jMeter test Apache per pubblicare migliaia di messaggi di esempio su un argomento Kafka per verificare i record importati in un Timestream for table. LiveAnalytics

Amazon QuickSight

Puoi utilizzare Amazon QuickSight per analizzare e pubblicare dashboard di dati che contengono i dati di Amazon Timestream. Questa sezione descrive come creare una nuova connessione a una fonte di QuickSight dati, modificare le autorizzazioni, creare nuovi set di dati ed eseguire un'analisi. Questo video tutorial descrive come lavorare con Timestream e Amazon. QuickSight

> **Note**
>
> Tutti i set di dati in Amazon QuickSight sono di sola lettura. Non puoi apportare modifiche ai dati effettivi in Timestream utilizzando Amazon QuickSight per rimuovere la fonte di dati, il set di dati o i campi.

Argomenti

- Accesso ad Amazon Timestream da QuickSight
- Crea una nuova connessione a una fonte di dati per Timestream QuickSight
- Modifica le autorizzazioni per la connessione all'origine QuickSight dati per Timestream
- Crea un nuovo QuickSight set di dati per Timestream
- Crea una nuova analisi per Timestream
- Tutorial video

Accesso ad Amazon Timestream da QuickSight

Prima di procedere, Amazon QuickSight deve essere autorizzato a connettersi ad Amazon Timestream. Se le connessioni non sono abilitate, riceverai un errore quando tenti di connetterti. Un QuickSight amministratore può autorizzare le connessioni alle AWS risorse. Per autorizzare una connessione da QuickSight Timestream, segui la procedura in Using Other AWS Services: Scoping Down Access, scegliendo Amazon Timestream nel passaggio 5.

Crea una nuova connessione a una fonte di dati per Timestream QuickSight

> **Note**
>
> La connessione tra Amazon QuickSight e Amazon Timestream è crittografata in transito SSL utilizzando TLS (1.2). Non è possibile creare una connessione non crittografata.

1. Assicurati di aver configurato le autorizzazioni appropriate per consentire QuickSight ad Amazon di accedere ad Amazon Timestream, come descritto in. Accesso ad Amazon Timestream da QuickSight
2. Inizia creando un nuovo set di dati. Scegli Datasets dal pannello di navigazione, quindi scegli Nuovo set di dati.
3. Seleziona la scheda sorgente dati Timestream.
4. Per Nome dell'origine dati, inserisci un nome per la tua connessione all'origine dati Timestream, ad esempio. `US Timestream Data`

   > **Note**
   >
   > Poiché puoi creare molti set di dati da una connessione a Timestream, è preferibile assegnare un nome semplice.

5. Scegli Convalida connessione per verificare di poterti connettere correttamente a Timestream.

   > **Note**
   >
   > La convalida della connessione convalida solo la possibilità di connettersi. Tuttavia, non convalida una tabella o una query specifica.

6. Seleziona Crea origine dati per procedere.

7. Per Database, scegli Seleziona... per visualizzare l'elenco delle opzioni disponibili. Scegli quella che vuoi usare.
8. Scegli Seleziona per continuare.
9. Seleziona una delle seguenti opzioni:
   - Per importare i dati nel QuickSight motore in memoria (chiamatoSPICE), scegli Importa in SPICE per un'analisi più rapida.
   - QuickSight Per consentire l'esecuzione di una query sui dati ogni volta che aggiorni il set di dati o utilizzi l'analisi o la dashboard, scegli Interroga direttamente i tuoi dati.
10. Scegli Modifica/Anteprima e poi Salva per salvare il set di dati e chiuderlo.

Modifica le autorizzazioni per la connessione all'origine QuickSight dati per Timestream

La procedura seguente descrive come visualizzare, aggiungere e revocare le autorizzazioni per altri QuickSight utenti in modo che possano accedere alla stessa fonte di dati Timestream. Le persone devono essere utenti attivi QuickSight prima di poterle aggiungere.

> **Note**
>
> In QuickSight, le fonti di dati hanno due livelli di autorizzazione: utente e proprietario.
>
> - Scegli utente per consentire l'accesso in lettura.
> - Scegli il proprietario per consentire a quell'utente di modificare, condividere o eliminare questa fonte di QuickSight dati.

1. Assicurati di aver configurato le autorizzazioni appropriate per consentire QuickSight ad Amazon di accedere ad Amazon Timestream, come descritto in. Accesso ad Amazon Timestream da QuickSight
2. Scegli Set di dati sulla sinistra, quindi scorri verso il basso per trovare il la scheda dell'origine dati per la tua connessione Timestream. Ad esempio, `US Timestream Data`.
3. Scegli la scheda sorgente dei `Timestream` dati.
4. Scegli `Share data source`. Viene visualizzato un elenco delle autorizzazioni correnti.
5. (Facoltativo) Per modificare le autorizzazioni, puoi scegliere `user` o. `owner`

6. (Facoltativo) Per revocare le autorizzazioni, scegli. `Revoke access` Le persone che revochi non possono creare nuovi set di dati da questa fonte di dati. Tuttavia, i loro set di dati esistenti avranno ancora accesso a questa fonte di dati.
7. Per aggiungere autorizzazioni, scegli`Invite users`, quindi segui questi passaggi per aggiungere un utente:
    a. Aggiungi persone per consentire loro di utilizzare la stessa fonte di dati.
    b. Per ognuna, scegli `Permission` quella che desideri applicare.
8. Quando hai finito, scegli`Close`.

Crea un nuovo QuickSight set di dati per Timestream

1. Assicurati di aver configurato le autorizzazioni appropriate per consentire QuickSight ad Amazon di accedere ad Amazon Timestream, come descritto in. Accesso ad Amazon Timestream da QuickSight
2. Scegli Set di dati sulla sinistra, quindi scorri verso il basso per trovare il la scheda dell'origine dati per la tua connessione Timestream. Se disponi di molte fonti di dati, puoi utilizzare la barra di ricerca nella parte superiore della pagina per trovarle con una corrispondenza parziale sul nome.
3. Scegli la scheda sorgente dati Timestream. Quindi scegli Crea set di dati.
4. Per Database, scegli Seleziona per visualizzare l'elenco delle opzioni disponibili. Scegli il database che desideri utilizzare.
5. Per Tabelle, seleziona la tabella che desideri utilizzare.
6. Scegli Modifica/Anteprima.
7. (Facoltativo) Per aggiungere altri dati, scegli Aggiungi dati in alto a destra.
    a. Scegli Cambia origine dati e scegli un'origine dati diversa.
    b. Segui le istruzioni dell'interfaccia utente per completare l'aggiunta dei dati.
    c. Dopo aver aggiunto nuovi dati allo stesso set di dati, scegli Configura questo join (i due punti rossi). Imposta un join per ogni tabella aggiuntiva.
    d. Se desideri aggiungere campi calcolati, scegli Aggiungi campo calcolato.
    e. Per usare Sagemaker, scegli Aumenta con. SageMaker Questa opzione è disponibile solo nell' QuickSight edizione Enterprise.
    f. Deseleziona tutti i campi che desideri omettere.
    g. Aggiorna tutti i tipi di dati che desideri modificare.

8. Al termine, scegli Salva per salvare e chiudere il set di dati.

Crea una nuova analisi per Timestream

1. Assicurati di aver configurato le autorizzazioni appropriate per consentire QuickSight ad Amazon di accedere ad Amazon Timestream, come descritto in. Accesso ad Amazon Timestream da QuickSight
2. Scegli Analisi sulla sinistra.
3. Seleziona una delle seguenti opzioni:
   - Per creare una nuova analisi, scegli Nuova analisi sulla destra.
   - Per aggiungere il set di dati Timestream a un'analisi esistente, apri l'analisi che desideri modificare. Scegli l'icona a forma di matita in alto a sinistra, quindi Aggiungi set di dati.
4. Inizia la prima visualizzazione dei dati scegliendo i campi a sinistra.
5. Per ulteriori informazioni, consulta Working with Analyses - Amazon QuickSight

Tutorial video

Questo video spiega come Amazon QuickSight funziona con Timestream.

Amazon SageMaker

Puoi usare Amazon SageMaker Notebooks per integrare i tuoi modelli di machine learning con Amazon Timestream. Per aiutarti a iniziare, abbiamo creato un esempio di SageMaker Notebook che elabora i dati di Timestream. I dati vengono inseriti in Timestream da un'applicazione Python multithread che invia continuamente dati. Il codice sorgente per l'esempio SageMaker Notebook e l'applicazione Python di esempio sono disponibili in. GitHub

1. Create un database e una tabella seguendo le istruzioni descritte in Creazione di un database e Creare una tabella
2. Clona il GitHub repository per l'applicazione di esempio Python multithread seguendo le istruzioni di GitHub
3. Clona il GitHub repository per il Timestream Notebook di esempio seguendo le istruzioni di. SageMaker GitHub
4. Esegui l'applicazione per l'inserimento continuo di dati in Timestream seguendo le istruzioni contenute nel README

5. Segui le istruzioni per creare un bucket Amazon S3 per Amazon SageMaker come descritto qui.
6. Crea un' SageMaker istanza Amazon con l'ultima versione di boto3 installata: oltre alle istruzioni descritte qui, segui i passaggi seguenti:
   a. Nella pagina Crea un'istanza notebook, fai clic su Configurazione aggiuntiva
   b. Fai clic su Configurazione del ciclo di vita (opzionale) e seleziona Crea una nuova configurazione del ciclo di vita
   c. Nella casella Create lifecycle configuration wizard, effettuate le seguenti operazioni:
      i. Inserisci il nome desiderato per la configurazione, ad es. `on-start`
      ii. Nello script Start Notebook, copia e incolla il contenuto dello script da Github
      iii. Sostituisci `PACKAGE=scipy` con `PACKAGE=boto3` nello script incollato.
7. Fai clic su Crea configurazione
8. Vai al IAM servizio nella console di AWS gestione e trova il ruolo di SageMaker esecuzione appena creato per l'istanza del notebook.
9. Allega la IAM policy for `AmazonTimestreamFullAccess` al ruolo di esecuzione.

> **Note**
>
> La `AmazonTimestreamFullAccess` IAM politica non è limitata a risorse specifiche e non è adatta all'uso in produzione. Per un sistema di produzione, prendi in considerazione l'utilizzo di politiche che limitano l'accesso a risorse specifiche.

10. Quando lo stato dell'istanza del notebook è InService, scegli Open Jupyter per avviare un SageMaker Notebook per l'istanza
11. Carica i file **`timestreamquery.py`** e **`Timestream_SageMaker_Demo.ipynb`** inseriscili nel Notebook selezionando il pulsante Carica
12. Scegliere `Timestream_SageMaker_Demo.ipynb`

> **Note**
>
> Se vedi un pop-up con Kernel not found, scegli conda_python3 e fai clic su Set Kernel.

13. Modifica`DB_NAME`, `TABLE_NAME`e `bucket`, e in modo che corrispondano al nome del database, `ENDPOINT` al nome della tabella, al nome del bucket S3 e alla regione per i modelli di addestramento.

14. Scegli l'icona di riproduzione per eseguire le singole celle
15. Quando arrivi alla cella`Leverage Timestream to find hosts with average CPU utilization across the fleet`, assicurati che l'output restituisca almeno 2 nomi host.

> **Note**
>
> Se nell'output sono presenti meno di 2 nomi host, potrebbe essere necessario rieseguire l'applicazione Python di esempio che inserisce i dati in Timestream con un numero maggiore di thread e su scala host.

16. Quando arrivate alla cella, modificatela in base alle risorse richieste per il vostro `Train a Random Cut Forest (RCF) model using the CPU utilization history` lavoro di formazione `train_instance_type`
17. Quando arrivi alla cella`Deploy the model for inference`, modificala in `instance_type` base ai requisiti di risorse per il tuo lavoro di inferenza

> **Note**
>
> Potrebbero essere necessari alcuni minuti per addestrare il modello. Al termine dell'addestramento, nell'output della cella verrà visualizzato il messaggio Completato - Training job completato.

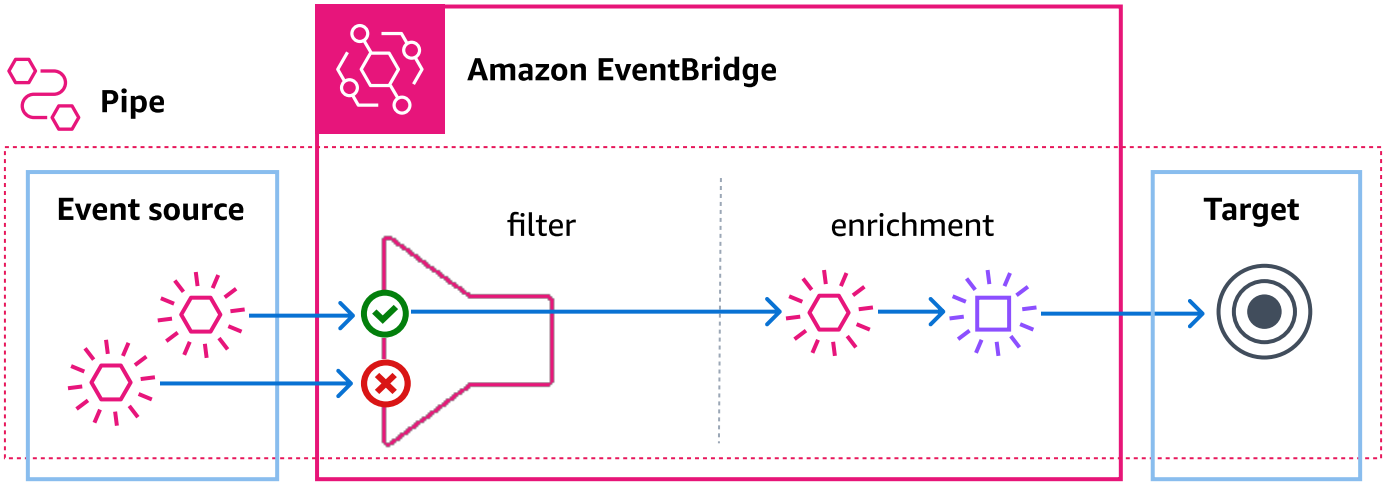
18. Esegui la cella `Stop and delete the endpoint` per ripulire le risorse. Puoi anche interrompere ed eliminare l'istanza dalla SageMaker console

Amazon SQS

Utilizzo di EventBridge Pipes per inviare SQS dati Amazon a Timestream

Puoi utilizzare EventBridge Pipes per inviare dati da una SQS coda Amazon a un Amazon LiveAnalytics Timestream per tabella.

Le pipe sono destinate point-to-point alle integrazioni tra sorgenti e destinazioni supportate, con supporto per trasformazioni e arricchimenti avanzati. Le pipe riducono la necessità di conoscenze specializzate e codice di integrazione durante lo sviluppo di architetture basate sugli eventi. Per configurare una pipe, si sceglie l'origine, si aggiungono filtri facoltativi, si definisce l'arricchimento facoltativo e si sceglie la destinazione per i dati dell'evento.



Per ulteriori informazioni su EventBridge Pipes, consultate EventBridge Pipes nella Guida EventBridge per l'utente. Per informazioni sulla configurazione di una pipe per inviare eventi a un Amazon LiveAnalytics Timestream for table EventBridge , consulta le specifiche del target Pipes.

Utilizzo DBeaver per lavorare con Amazon Timestream

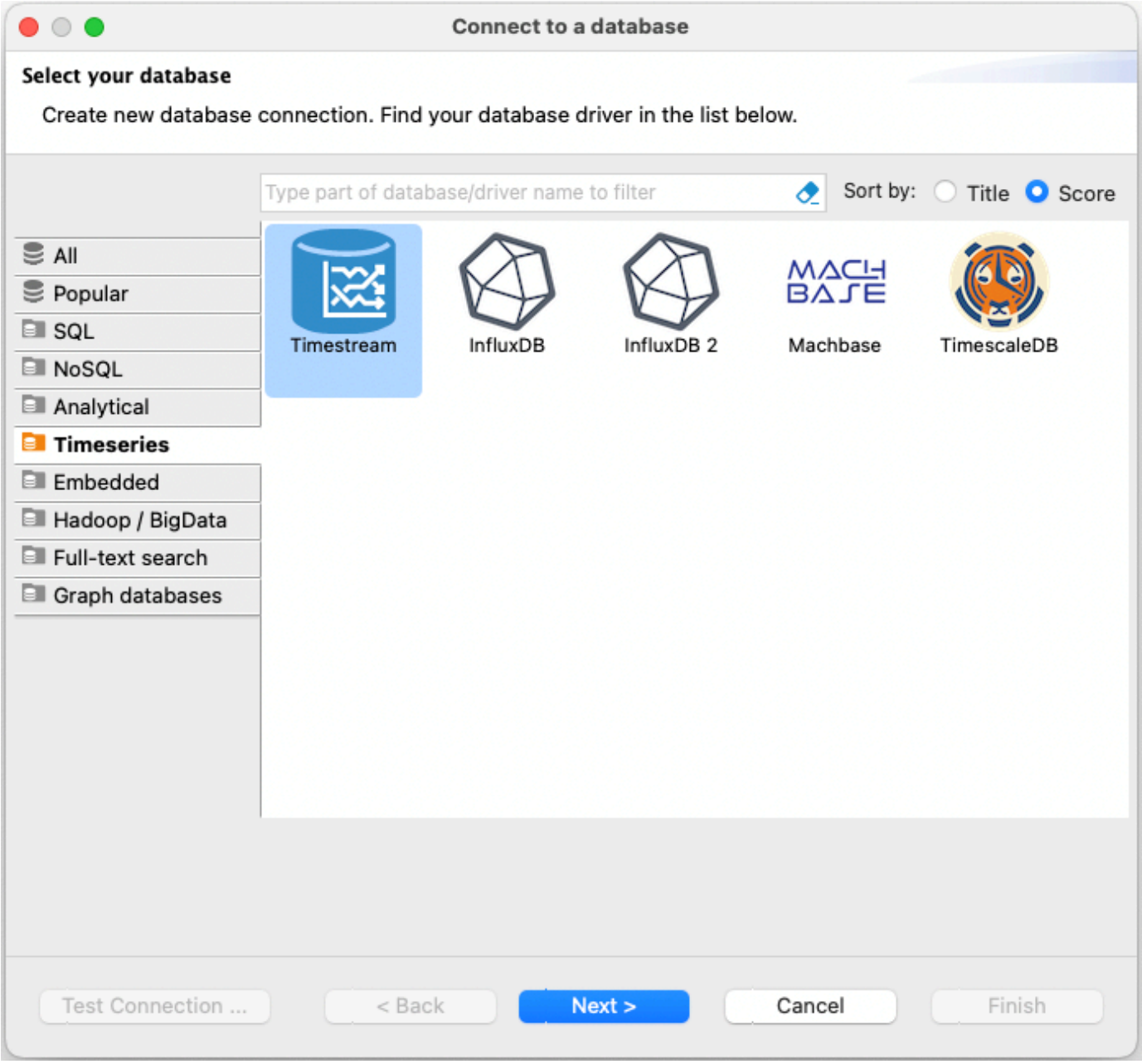
DBeaverè un SQL client universale gratuito che può essere utilizzato per gestire qualsiasi database dotato di JDBC driver. È ampiamente utilizzato dagli sviluppatori e dagli amministratori di database grazie alle sue solide capacità di visualizzazione, modifica e gestione dei dati.

Utilizzando le opzioni DBeaver di connettività cloud di Amazon, puoi connetterti DBeaver ad Amazon Timestream in modo nativo. DBeaverfornisce un'interfaccia completa e intuitiva per lavorare con i dati delle serie temporali direttamente dall'interno di un'applicazione. DBeaver Utilizzando le credenziali, consente inoltre l'accesso completo a tutte le interrogazioni che è possibile eseguire da un'altra interfaccia di interrogazione. Consente anche di creare grafici per una migliore comprensione e visualizzazione dei risultati delle query.

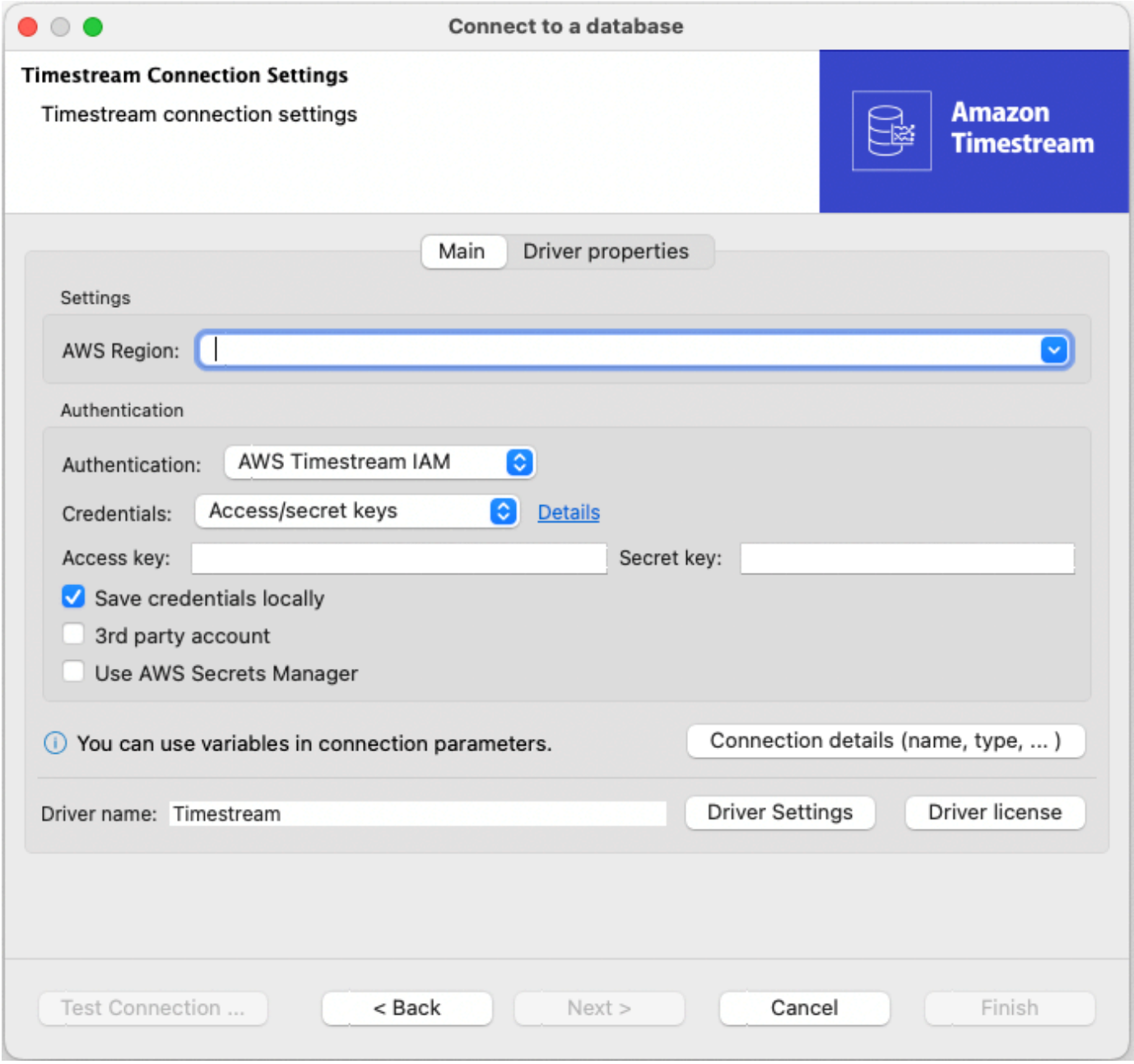
Configurazione DBeaver per lavorare con Timestream

Segui i seguenti passaggi per configurare l'utilizzo DBeaver di Timestream:

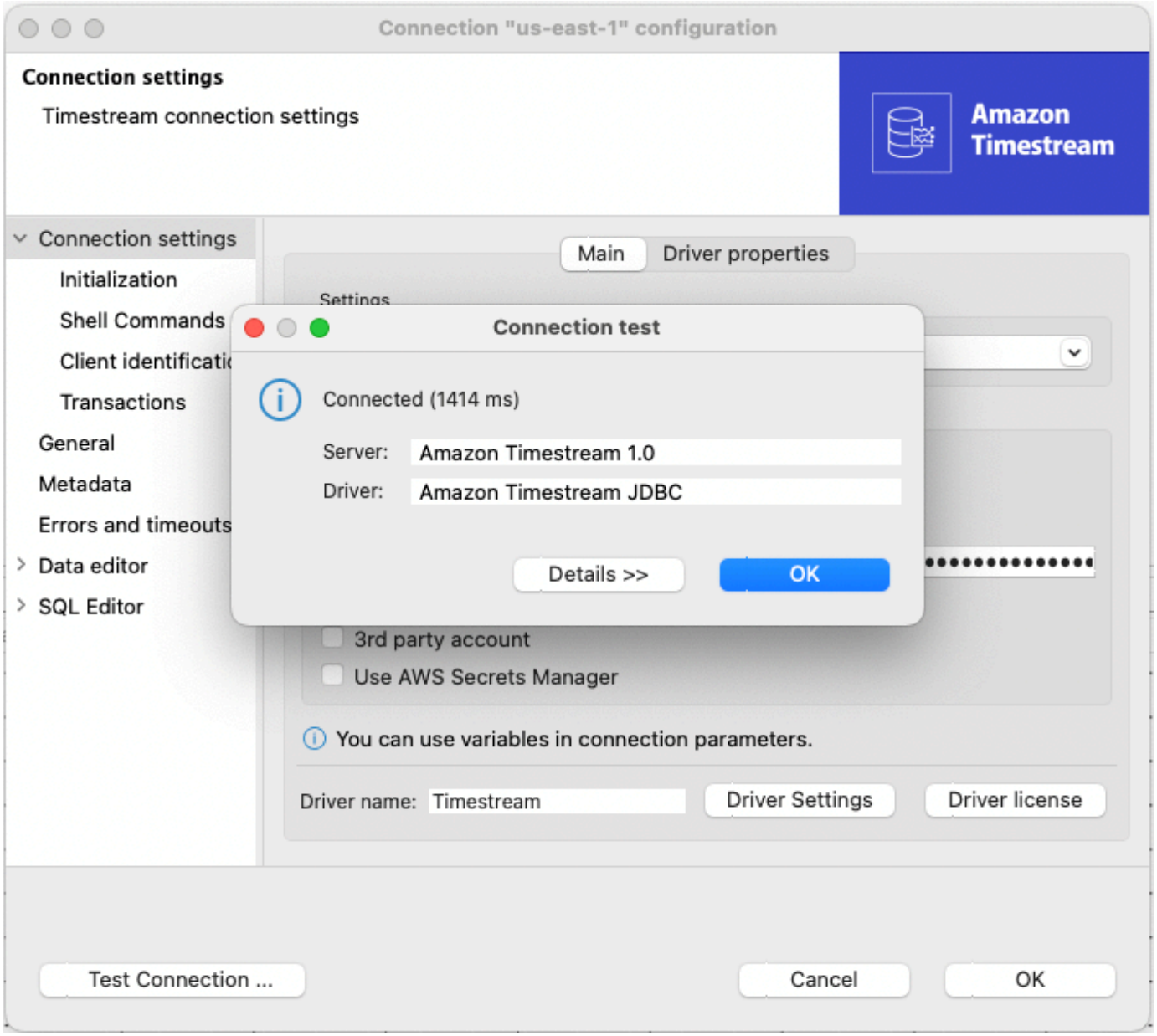
1. Scarica e installa DBeaver sul tuo computer locale.
2. AvviaDBeaver, vai all'area di selezione del database, scegli Serie temporali nel riquadro a sinistra, quindi seleziona l'icona Timestream nel riquadro a destra:



3. Nella finestra Impostazioni di connessione Timestream, inserisci tutte le informazioni necessarie per connetterti al tuo database Amazon Timestream. Assicurati che le chiavi utente che inserisci abbiano le autorizzazioni necessarie per accedere al tuo database Timestream. Inoltre, assicurati di mantenere le informazioni e le chiavi che inserisci in modo DBeaver sicuro e privato, come per qualsiasi informazione sensibile.



4. Verifica la connessione per assicurarti che tutto sia configurato correttamente:



5. Se il test di connessione ha esito positivo, ora puoi interagire con il tuo database Amazon Timestream proprio come faresti con qualsiasi altro database in esso. DBeaver Ad esempio, puoi accedere all'SQLeditor o alla vista ER Diagram per eseguire le query:

6. DBeaverfornisce anche potenti strumenti di visualizzazione dei dati. Per utilizzarli, esegui la query, quindi seleziona l'icona del grafico per visualizzare il set di risultati. Lo strumento grafico può aiutarti a comprendere meglio le tendenze dei dati nel tempo.

L'associazione di Amazon DBeaver Timestream a crea un ambiente efficace per la gestione dei dati delle serie temporali. Puoi integrarlo perfettamente nel tuo flusso di lavoro esistente per migliorare la produttività e l'efficienza.

Grafana

Puoi visualizzare i dati delle tue serie temporali e creare avvisi utilizzando Grafana. Per aiutarti a iniziare con la visualizzazione dei dati, abbiamo creato una dashboard di esempio in Grafana che visualizza i dati inviati a Timestream da un'applicazione Python e un video tutorial che descrive la configurazione.

Argomenti

- Applicazione di esempio
- Tutorial video

Applicazione di esempio

1. Crea un database e una tabella in Timestream seguendo le istruzioni descritte in per ulteriori informazioni. Creazione di un database

   > **Note**
   >
   > Il nome del database e il nome della tabella predefiniti per la dashboard di Grafana sono impostati rispettivamente su GrafanaDB. grafanaTable Usa questi nomi per ridurre al minimo la configurazione.

2. Installa Python 3.7 o versioni successive
3. Installa e configura Timestream Python SDK
4. Clona il GitHub repository per l'applicazione Python multithread inserendo continuamente i dati in Timestream seguendo le istruzioni di GitHub
5. Esegui l'applicazione per l'inserimento continuo di dati in Timestream seguendo le istruzioni contenute nel README
6. Completa Guida introduttiva ad Amazon Managed Grafana o completa Installa Grafana.
7. Se installi Grafana anziché utilizzare Amazon Managed Grafana, completa Installa il plug-in Timestream per Grafana.

8. Apri la dashboard Grafana utilizzando un browser a tua scelta. Se hai installato Grafana localmente, puoi seguire le istruzioni descritte nella documentazione Grafana per accedere
9. Dopo aver avviato Grafana, vai su Datasources, fai clic su Aggiungi origine dati, cerca Timestream e seleziona l'origine dati Timestream
10. Configura il provider di autenticazione e la regione e fai clic su Salva e prova
11. Imposta le macro predefinite
    a. Imposta $__database sul nome del tuo database Timestream (ad esempio GrafanADB)
    b. Imposta $__table sul nome della tua tabella Timestream (ad esempio) grafanaTable
    c. Imposta $__measure sulla misura più comunemente usata nella tabella
12. Fai clic su Salva e prova
13. Fai clic sulla scheda Dashboard
14. Fai clic su Importa per importare la dashboard
15. Fate doppio clic sulla dashboard dell'applicazione di esempio
16. Fai clic sulle impostazioni della dashboard
17. Seleziona Variabili
18. Modifica dbName e fai tableName corrispondere i nomi del database e della tabella Timestream
19. Fai clic su Salva
20. Aggiorna la dashboard
21. Per creare avvisi, segui le istruzioni descritte nella documentazione di Grafana per creare una regola di avviso gestito Grafana
22. Per risolvere i problemi relativi agli avvisi, segui le istruzioni descritte nella documentazione Grafana per la risoluzione dei problemi
23. Per ulteriori informazioni, consulta la documentazione Grafana

Tutorial video

Questo video spiega come Grafana funziona con Timestream.

Utilizzo SquaredUp per lavorare con Amazon Timestream

SquaredUpè una piattaforma di osservabilità che si integra con Amazon Timestream. Puoi utilizzare l'intuitivo designer SquaredUp di dashboard per visualizzare, analizzare e monitorare i dati delle serie

temporali. Le dashboard possono essere condivise pubblicamente o privatamente e possono essere creati canali di notifica per avvisare l'utente quando cambia lo stato di salute di un monitor.

Utilizzo SquaredUp con Amazon Timestream

1. Iscriviti SquaredUpe inizia gratuitamente.
2. Aggiungi una fonte di AWS dati.
3. Crea un riquadro del dashboard che utilizza il flusso di dati Timestream Query.
4. Facoltativamente, abilita il monitoraggio del riquadro, crea un canale di notifica o condividi la dashboard pubblicamente o privatamente.
5. Facoltativamente, crea altri riquadri per visualizzare i dati di Timestream insieme ai dati degli altri strumenti di monitoraggio e osservabilità.

Telegraf open source

Puoi utilizzare il plug-in Timestream for LiveAnalytics output per Telegraf per scrivere metriche in Timestream direttamente da Telegraf open source. LiveAnalytics

Questa sezione fornisce una spiegazione su come installare Telegraf con il plug-in Timestream for output, come eseguire Telegraf con il plug-in Timestream for LiveAnalytics output e come Telegraf open source funziona con Timestream for. LiveAnalytics LiveAnalytics

Argomenti

- LiveAnalyticsInstallazione di Telegraf con il plug-in Timestream for output
- Esecuzione di Telegraf con il plug-in Timestream for output LiveAnalytics
- Mappatura delle metriche di Telegraf/InfluxDB sul Timestream per il modello LiveAnalytics

LiveAnalyticsInstallazione di Telegraf con il plug-in Timestream for output

A partire dalla versione 1.16, il plug-in Timestream for LiveAnalytics output è disponibile nella versione ufficiale di Telegraf. Per installare il plug-in di output sulla maggior parte dei principali sistemi operativi, segui i passaggi descritti nella documentazione di Telegraf. InfluxData Per l'installazione sul sistema operativo Amazon Linux 2, segui le istruzioni riportate di seguito.

Installazione di Telegraf con il LiveAnalytics plug-in Timestream for output su Amazon Linux 2

Per installare Telegraf con il plug-in Timestream Output su Amazon Linux 2, procedi nel seguente modo.

1. Installa Telegraf usando il gestore di pacchetti. `yum`

```
cat <<EOF | sudo tee /etc/yum.repos.d/influxdb.repo
[influxdb]
name = InfluxDB Repository - RHEL \$releasever
baseurl = https://repos.influxdata.com/rhel/\$releasever/\$basearch/stable
enabled = 1
gpgcheck = 1
gpgkey = https://repos.influxdata.com/influxdb.key
EOF
```

2. Esegui il comando seguente.

```
sudo sed -i "s/\$releasever/$(rpm -E %{rhel})/g" /etc/yum.repos.d/influxdb.repo
```

3. Installa e avvia Telegraf.

```
sudo yum install telegraf
sudo service telegraf start
```

Esecuzione di Telegraf con il plug-in Timestream for output LiveAnalytics

Puoi seguire le istruzioni riportate di seguito per eseguire Telegraf con il plugin Timestream for. LiveAnalytics

1. Genera una configurazione di esempio usando Telegraf.

```
telegraf --section-filter agent:inputs:outputs --input-filter cpu:mem --output-
filter timestream config > example.config
```

2. Crea un database in Timestream utilizzando la console di gestione, oppure. CLISDKs
3. Nel `example.config` file, aggiungi il nome del tuo database modificando la seguente chiave nella `[[outputs.timestream]]` sezione.

```
database_name = "yourDatabaseNameHere"
```

4. Per impostazione predefinita, Telegraf creerà una tabella. Se desideri creare una tabella manualmente, imposta `false` e segui le istruzioni `create_table_if_not_exists` per creare una tabella utilizzando la console di gestione, CLIoppure. SDKs
5. Nel file example.config, configura le credenziali nella sezione. `[[outputs.timestream]]` Le credenziali devono consentire le seguenti operazioni.

```
timestream:DescribeEndpoints
timestream:WriteRecords
```

> **Note**
>
> Se lasci `create_table_if_not_exists` impostato su`true`, includi:
>
> ```
> timestream:CreateTable
> ```

> **Note**
>
> Se lo `describe_database_on_start` impostitrue, includi quanto segue.
>
> ```
> timestream:DescribeDatabase
> ```

6. Puoi modificare il resto della configurazione in base alle tue preferenze.
7. Quando hai finito di modificare il file di configurazione, esegui Telegraf con quanto segue.

```
./telegraf --config example.config
```

8. Le metriche dovrebbero apparire entro pochi secondi, a seconda della configurazione dell'agente. Dovresti vedere anche le nuove tabelle, cpu e mem, nella console Timestream.

Mappatura delle metriche di Telegraf/InfluxDB sul Timestream per il modello LiveAnalytics

Quando si scrivono dati da Telegraf a Timestream for, i dati vengono mappati come segue. LiveAnalytics

- Il timestamp viene scritto come campo temporale.
- I tag vengono scritti come dimensioni.
- I campi sono scritti come misure.
- Le misurazioni sono per lo più scritte come nomi di tabelle (ne parleremo più avanti).

Il plug-in Timestream for LiveAnalytics output per Telegraf offre diverse opzioni per l'organizzazione e l'archiviazione dei dati in Timestream for. LiveAnalytics Questo può essere descritto con un esempio che inizia con i dati in formato protocollo di linea.

```
weather,location=us-midwest,season=summer temperature=82,humidity=71
1465839830100400200 airquality,location=us-west no2=5,pm25=16
1465839830100400200
```

Di seguito vengono descritti i dati.

- I nomi delle misurazioni sono `weather` e `airquality`.
- I tag sono `location` e `season`.
- I campi sono `temperature` e `humidity`, `no2`, e `pm25`.

Argomenti

- Memorizzazione dei dati in più tabelle
- Memorizzazione dei dati in un'unica tabella

Memorizzazione dei dati in più tabelle

Puoi scegliere di creare una tabella separata per misurazione e memorizzare ogni campo in una riga separata per tabella.

La configurazione è `mapping_mode = "multi-table"`.

- Il Timestream for LiveAnalytics adapter creerà due tabelle, `weather` e `airquality` a dire e.
- Ogni riga della tabella conterrà un solo campo.

Il Timestream risultante per le LiveAnalytics tabelle, `weather` e `airquality`, avrà il seguente aspetto.

`weather`

time	posizione	stagione	measure_name	measure_value::bigint
2016-06-13 17:43:50	Stati Uniti-Midwest	estate	temperature	82
2016-06-13 17:43:50	Stati Uniti-Midwest	estate	umidità	71

`airquality`

time	posizione	measure_name	measure_value::bigint
2016-06-13 17:43:50	Stati Uniti-Midwest	n° 2	5
2016-06-13 17:43:50	Stati Uniti-Midwest	pm 25	16

Memorizzazione dei dati in un'unica tabella

Puoi scegliere di memorizzare tutte le misurazioni in un'unica tabella e memorizzare ogni campo in una riga separata della tabella.

La configurazione è`mapping_mode = "single-table"`. Sono disponibili due configurazioni aggiuntive quando si utilizza`single-table`, `single_table_name` e`single_table_dimension_name_for_telegraf_measurement_name`.

- Il plug-in Timestream for LiveAnalytics output creerà una singola tabella con nome *`<single_table_name>`* che include un *`<single_table_dimension_name_for_telegraf_measurement_name>`* colonna.
- La tabella può contenere più campi in un'unica riga della tabella.

Il Timestream risultante per la LiveAnalytics tabella sarà simile a questo.

weather

time	posizione	stagione	*<single_table_dimension_name_for_telegraf_measurement_name>*	measure_name	measure_value::bigint
2016-06-13 17:43:50	Stati Uniti-Midwest	estate	meteo	temperature	82
2016-06-13 17:43:50	Stati Uniti-Midwest	estate	meteo	umidità	71
2016-06-13 17:43:50	Stati Uniti-Midwest	estate	qualità dell'aria	n. 2	5
2016-06-13 17:43:50	Stati Uniti-Midwest	estate	meteo	pm 25	16

JDBC

Puoi utilizzare una connessione Java Database Connectivity (JDBC) per connettere Timestream LiveAnalytics ai tuoi strumenti di business intelligence e ad altre applicazioni, come SQL Workbench. Il Timestream for LiveAnalytics JDBC driver attualmente supporta SSO Okta e Microsoft Azure AD.

Argomenti

- Configurazione del driver per Timestream per JDBC LiveAnalytics
- Proprietà di connessione
- JDBCURLesempi
- Configurazione di Timestream per l'autenticazione LiveAnalytics JDBC Single Sign-On con Okta
- Configurazione di Timestream per l'autenticazione LiveAnalytics JDBC Single Sign-On con Microsoft Azure AD

Configurazione del driver per Timestream per JDBC LiveAnalytics

Segui i passaggi seguenti per configurare il JDBC driver.

Argomenti

Timestream per il conducente LiveAnalytics JDBC JARs

Puoi ottenere il Timestream per il LiveAnalytics JDBC driver tramite download diretto o aggiungendo il driver come dipendenza Maven.

- Come download diretto:. Per scaricare direttamente Timestream for LiveAnalytics JDBC driver, completa i seguenti passaggi:

  1. Passa a /releases https://github.com/awslabs/ amazon-timestream-driver-jdbc
  2. Puoi `amazon-timestream-jdbc-1.0.1-shaded.jar` utilizzarlo direttamente con i tuoi strumenti e applicazioni di business intelligence
  3. `amazon-timestream-jdbc-1.0.1-javadoc.jar`Scaricatelo in una directory a vostra scelta.
  4. Nella directory in cui hai scaricato`amazon-timestream-jdbc-1.0.1-javadoc.jar`, esegui il seguente comando per estrarre i file JavadocHTML:

     ```
     jar -xvf amazon-timestream-jdbc-1.0.1-javadoc.jar
     ```

- Come dipendenza Maven: per aggiungere Timestream for LiveAnalytics JDBC driver come dipendenza Maven, completa i seguenti passaggi:

  1. Accedi e apri il file dell'applicazione in un editor a tua sceltapom.xml.
  2. Aggiungi il JDBC driver come dipendenza nel `pom.xml` file dell'applicazione:

     ```
     <!-- https://mvnrepository.com/artifact/software.amazon.timestream/amazon-
     timestream-jdbc -->
     <dependency>
         <groupId>software.amazon.timestream</groupId>
         <artifactId>amazon-timestream-jdbc</artifactId>
     ```

```
    <version>1.0.1</version>
</dependency>
```

Timestream per la classe e il formato del LiveAnalytics JDBC driver URL

La classe di driver per Timestream for driver è: LiveAnalytics JDBC

```
software.amazon.timestream.jdbc.TimestreamDriver
```

Il JDBC driver Timestream richiede il seguente formato: JDBC URL

```
jdbc:timestream:
```

Per specificare le proprietà del database tramite JDBCURL, utilizzate il formato seguente: URL

```
jdbc:timestream://
```

Applicazione di esempio

Per aiutarti a iniziare a usare Timestream for LiveAnalytics withJDBC, abbiamo creato un'applicazione di esempio completamente funzionale in. GitHub

1. Crea un database con dati di esempio seguendo le istruzioni descritte qui.
2. Clona il GitHub repository per l'applicazione di esempio per JDBC seguire le istruzioni di. GitHub
3. Segui le istruzioni riportate nella sezione READMEper iniziare a usare l'applicazione di esempio.

Proprietà di connessione

Il Timestream for LiveAnalytics JDBC driver supporta le seguenti opzioni:

Argomenti

- Opzioni di autenticazione di base
- Opzione standard di informazioni sul cliente
- Opzione di configurazione del driver
- SDKopzione
- Opzione di configurazione dell'endpoint

- Opzioni del fornitore di credenziali
- SAMLopzioni di autenticazione basate per Okta
- SAMLopzioni di autenticazione basate per Azure AD

> **Note**
>
> Se non viene fornita nessuna delle proprietà, Timestream for LiveAnalytics JDBC driver utilizzerà la catena di credenziali predefinita per caricare le credenziali.

> **Note**
>
> Tutte le chiavi di proprietà fanno distinzione tra maiuscole e minuscole.

Opzioni di autenticazione di base

La tabella seguente descrive le opzioni di autenticazione di base disponibili.

Opzione	Descrizione	Default
AccessKeyId	L'id della chiave di accesso AWS utente.	NONE
SecretAccessKey	La chiave di accesso segreta AWS dell'utente.	NONE
SessionToken	Il token di sessione temporane o necessario per accedere a un database con l'autenti cazione a più fattori (MFA) abilitata.	NONE

Opzione standard di informazioni sul cliente

La tabella seguente descrive l'opzione Standard Client Info.

Opzione	Descrizione	Default
ApplicationName	Il nome dell'applicazione che attualmente utilizza la connessione. `ApplicationName` viene utilizzato per scopi di debug e non verrà comunicato al servizio Timestream. LiveAnalytics	Il nome dell'applicazione rilevato dal driver.

Opzione di configurazione del driver

La tabella seguente descrive l'opzione di configurazione del driver.

Opzione	Descrizione	Default
EnableMetaDataPreparedStatement	Abilita Timestream per il LiveAnalytics JDBC driver per cui restituire i metadati`PreparedStatements` , ma ciò comporterà un costo aggiuntivo con Timestream per il recupero dei metadati. LiveAnalytics	FALSE
Regione	L'area del database.	us-east-1

SDKopzione

La tabella seguente descrive l'opzione. SDK

Opzione	Descrizione	Default
RequestTimeout	Il tempo in millisecondi in cui AWS SDK attenderà una	0

Opzione	Descrizione	Default
	richiesta di query prima del timeout. Il valore non positivo disabilita il timeout della richiesta.	
SocketTimeout	Il tempo in millisecondi che AWS SDK attenderà il trasferimento dei dati su una connessione aperta prima del timeout. Il valore non deve essere negativo. Un valore di 0 disabilita il timeout del socket.	50000
MaxRetryCountClient	Il numero massimo di tentativi per errori ripetibili con codici di errore 5XX in. SDK Il valore non deve essere negativo.	NONE
MaxConnections	Il numero massimo di HTTP connessioni aperte contemporaneamente consentite al servizio Timestream for. LiveAnalytics Il valore deve essere positivo.	50

Opzione di configurazione dell'endpoint

La tabella seguente descrive l'opzione di configurazione degli endpoint.

Opzione	Descrizione	Default
Endpoint	L'endpoint per il servizio Timestream for. LiveAnalytics	NONE

Opzioni del fornitore di credenziali

La tabella seguente descrive le opzioni disponibili del Credential Provider.

Opzione	Descrizione	Default
AwsCredentialsProviderClass	Una delle `PropertiesFileCredentialsProvider` o `InstanceProfileCredentialsProvider` da utilizzare per l'autenticazione.	NONE
CustomCredentialsFilePath	Il percorso di un file delle proprietà contenente le credenziali AWS di sicurezza `accessKey` e `secretKey`. È richiesto solo se `AwsCredentialsProviderClass` è specificato come `PropertiesFileCredentialsProvider`.	NONE

SAMLopzioni di autenticazione basate per Okta

La tabella seguente descrive le opzioni di autenticazione SAML basate disponibili per Okta.

Opzione	Descrizione	Default
IdpName	Il nome dell'Identity Provider (Idp) da utilizzare per l'autenticazione SAML basata. Uno dei nostri `Okta`, `AzureAD`	NONE
IdpHost	Il nome host dell'Idp specificato.	NONE

Opzione	Descrizione	Default
IdpUserName	Il nome utente per l'account Idp specificato.	NONE
IdpPassword	La password per l'account Idp specificato.	NONE
OktaApplicationID	L'ID univoco fornito da OKTA associato all'applicazione Timestream for. LiveAnalytics `AppId`può essere trovato nel `entityID` campo fornito nei metadati dell'applicazione. Considerate il seguente esempio: `entityID = http://www.okta.com//IdpAppID`	NONE
Ruolo ARN	L'Amazon Resource Name (ARN) del ruolo assunto dal chiamante.	NONE
Idp ARN	L'Amazon Resource Name (ARN) del SAML provider IAM che descrive l'Idp.	NONE

SAMLopzioni di autenticazione basate per Azure AD

La tabella seguente descrive le opzioni di autenticazione SAML basate disponibili per Azure AD.

Opzione	Descrizione	Default
IdpName	Il nome dell'Identity Provider (Idp) da usare per l'autenti cazione SAML basata. Uno dei nostri`Okta`. `AzureAD`	NONE

Opzione	Descrizione	Default
IdpHost	Il nome host dell'Idp specifica to.	NONE
IdpUserName	Il nome utente per l'account Idp specificato.	NONE
IdpPassword	La password per l'account Idp specificato.	NONE
AADApplicationID	L'id univoco dell'applicazione registrata in Azure AD.	NONE
AADClientSecret	Il client secret associato all'applicazione registrata su Azure AD usato per autorizza re il recupero dei token.	NONE
AADTenant	L'ID del tenant di Azure AD.	NONE
Idp ARN	L'Amazon Resource Name (ARN) del SAML provider IAM che descrive l'Idp.	NONE

JDBCURLesempi

Questa sezione descrive come creare una JDBC connessione URL e fornisce esempi. Per specificare le proprietà di connessione opzionali, utilizzate il seguente URL formato:

```
jdbc:timestream://PropertyName1=value1;PropertyName2=value2...
```

> **Note**
>
> Tutte le proprietà di connessione sono opzionali. Tutte le chiavi di proprietà fanno distinzione tra maiuscole e minuscole.

Di seguito sono riportati alcuni esempi di connessione. JDBC URLs

Esempio con opzioni di autenticazione di base e regione:

```
jdbc:timestream://
AccessKeyId=<myAccessKeyId>;SecretAccessKey=<mySecretAccessKey>;SessionToken=<mySessionToken
east-1
```

Esempio con informazioni sul cliente, regione e SDK opzioni:

```
jdbc:timestream://ApplicationName=MyApp;Region=us-
east-1;MaxRetryCountClient=10;MaxConnections=5000;RequestTimeout=20000
```

Connect utilizzando la catena di provider di credenziali predefinita con AWS credenziali impostate nelle variabili di ambiente:

```
jdbc:timestream
```

Connettiti utilizzando la catena di provider di credenziali predefinita con AWS credenziali impostate nella connessione: URL

```
jdbc:timestream://
AccessKeyId=<myAccessKeyId>;SecretAccessKey=<mySecretAccessKey>;SessionToken=<mySessionToken
```

Connect utilizzando PropertiesFileCredentialsProvider come metodo di autenticazione:

```
jdbc:timestream://
AwsCredentialsProviderClass=PropertiesFileCredentialsProvider;CustomCredentialsFilePath=<pat
 to properties file>
```

Connect utilizzando InstanceProfileCredentialsProvider come metodo di autenticazione:

```
jdbc:timestream://AwsCredentialsProviderClass=InstanceProfileCredentialsProvider
```

Connect utilizzando le credenziali Okta come metodo di autenticazione:

```
jdbc:timestream://
IdpName=Okta;IdpHost=<host>;IdpUserName=<name>;IdpPassword=<password>;OktaApplicationID=<id>
```

Connect usando le credenziali di Azure AD come metodo di autenticazione:

```
jdbc:timestream:// 
IdpName=AzureAD;IdpUserName=<name>;IdpPassword=<password>;AADApplicationID=<id>;AADClientSec
```

Connect con un endpoint specifico:

```
jdbc:timestream://Endpoint=abc.us-east-1.amazonaws.com;Region=us-east-1
```

Configurazione di Timestream per l'autenticazione LiveAnalytics JDBC Single Sign-On con Okta

Timestream for LiveAnalytics supporta Timestream per l'autenticazione Single Sign-On con Okta LiveAnalytics JDBC. Per utilizzare Timestream per l'autenticazione LiveAnalytics JDBC Single Sign-On con Okta, completa ciascuna delle sezioni elencate di seguito.

Argomenti

- Prerequisiti
- AWS federazione degli account in Okta
- Configurazione di Okta per SAML

Prerequisiti

Assicurati di aver soddisfatto i seguenti prerequisiti prima di utilizzare Timestream per l'autenticazione Single Sign-on con Okta: LiveAnalytics JDBC

- Autorizzazioni di amministratore AWS per creare il provider di identità e i ruoli.
- Un account Okta (vai a https://www.okta.com/login/creare un account).
- Accesso ad Amazon LiveAnalytics Timestream per.

Ora che hai completato i prerequisiti, puoi procedere a. AWS federazione degli account in Okta

AWS federazione degli account in Okta

Il Timestream for LiveAnalytics JDBC driver supporta AWS Account Federation in Okta. Per configurare AWS Account Federation in Okta, completa i seguenti passaggi:

1. Accedi alla dashboard di amministrazione di Okta utilizzando quanto segue: URL

```
https://<company-domain-name>-admin.okta.com/admin/apps/active
```

> **Note**
>
> Sostituisci < company-domain-name > con il tuo nome di dominio.

2. Dopo aver effettuato correttamente l'accesso, scegli Aggiungi applicazione e cerca AWS Account Federation.
3. Scegli Aggiungi
4. Cambia il URL login con quello appropriatoURL.
5. Seleziona Next (Successivo).
6. Scegli SAML2.0 come metodo di accesso
7. Scegli i metadati di Identity Provider per aprire il file di metadati. XML Salva il file localmente.
8. Lascia vuote tutte le altre opzioni di configurazione.
9. Seleziona Done (Fatto).

Ora che hai completato AWS Account Federation in Okta, puoi procedere aConfigurazione di Okta per SAML.

Configurazione di Okta per SAML

1. Scegli la scheda Sign On (Accedi). Scegli la visualizzazione.
2. Scegli il pulsante Istruzioni di configurazione nella sezione Impostazioni.

Trovare il documento di metadati Okta

1. Per trovare il documento, vai a:

```
https://<domain>-admin.okta.com/admin/apps/active
```

> **Note**
>
> <domain>è il tuo nome di dominio univoco per il tuo account Okta.

2. Scegli l'applicazione AWS Account Federation
3. Scegli la scheda Accedi

Configurazione di Timestream per l'autenticazione LiveAnalytics JDBC Single Sign-On con Microsoft Azure AD

Timestream for LiveAnalytics supporta Timestream per l'autenticazione LiveAnalytics JDBC Single Sign-On con Microsoft Azure AD. Per usare Timestream per l'autenticazione LiveAnalytics JDBC Single Sign-On con Microsoft Azure AD, completa ciascuna delle sezioni elencate di seguito.

Argomenti

- Prerequisiti
- Configurazione di Azure AD
- Configurazione IAM dell'Identity Provider e dei ruoli in AWS

Prerequisiti

Assicurati di aver soddisfatto i seguenti prerequisiti prima di utilizzare Timestream per l'autenticazione LiveAnalytics JDBC Single Sign-On con Microsoft Azure AD:

- Autorizzazioni amministrative AWS per creare il provider di identità e i ruoli.
- Un account Azure Active Directory (vai a https://azure.microsoft.com/en-ca/services/active-directory/ per creare un account)
- Accesso ad Amazon LiveAnalytics Timestream per.

Configurazione di Azure AD

1. Accedi al portale di Azure
2. Scegli Azure Active Directory nell'elenco dei servizi di Azure. Questo reindirizzerà alla pagina Default Directory.
3. Scegli Applicazioni aziendali nella sezione Gestisci nella barra laterale
4. Scegli + Nuova applicazione.
5. Trova e seleziona Amazon Web Services.
6. Scegli Single Sign-On nella sezione Gestisci nella barra laterale

7. Scegli SAMLcome metodo Single Sign-On
8. Nella sezione SAML Configurazione di base, inserisci quanto segue sia URL per l'identificatore che per la risposta: URL

```
https://signin.aws.amazon.com/saml
```

9. Seleziona Salva
10. Scarica i metadati della federazione XML nella sezione Certificato di SAML firma. Verrà utilizzato durante la creazione dell'IAMIdentity Provider in un secondo momento
11. Torna alla pagina Directory predefinita e scegli Registrazioni app in Gestisci.
12. Scegli Timestream for LiveAnalytics dalla sezione Tutte le applicazioni. La pagina verrà reindirizzata alla pagina Panoramica dell'applicazione

> **Note**
>
> Annota l'ID dell'applicazione (client) e l'ID della directory (tenant). Questi valori sono necessari per la creazione di una connessione.

13. Scegli Certificati e segreti
14. In Client secrets, crea un nuovo client secret con + New client secret.

> **Note**
>
> Nota il segreto del client generato, poiché è necessario quando si crea una connessione a Timestream for. LiveAnalytics

15. Nella barra laterale, sotto Gestisci, seleziona Autorizzazioni API
16. Nella sezione Autorizzazioni configurate, usa Aggiungi un'autorizzazione per concedere ad Azure AD l'autorizzazione ad accedere a Timestream per. LiveAnalytics Scegli Microsoft Graph nella pagina Richiedi API autorizzazioni.
17. Scegli Autorizzazioni delegate e seleziona l'autorizzazione User.Read
18. Scegli Aggiungi autorizzazioni
19. Scegli Concedi il consenso dell'amministratore per la directory predefinita

Configurazione IAM dell'Identity Provider e dei ruoli in AWS

Completa ogni sezione seguente per configurare Timestream IAM per l'autenticazione LiveAnalytics JDBC Single Sign-On con Microsoft Azure AD:

Argomenti

SAMLCrea un provider di identità

Per creare un SAML Identity Provider per il Timestream per l'autenticazione LiveAnalytics JDBC Single Sign-On con Microsoft Azure AD, completa i seguenti passaggi:

1. Accedi alla console di gestione AWS
2. Scegli Servizi e seleziona IAMin Sicurezza, identità e conformità
3. Scegli i provider di identità in Gestione degli accessi
4. Scegli Crea provider e scegli SAMLcome tipo di provider. Inserisci il nome del provider. Questo esempio utilizzerà zureADProvider A.
5. Carica il file Federation Metadata XML scaricato in precedenza
6. Scegli Avanti, quindi scegli Crea.
7. Al termine, la pagina verrà reindirizzata alla pagina dei provider di identità

Creare un ruolo IAM.

Per creare un IAM ruolo per il Timestream per l'autenticazione LiveAnalytics JDBC Single Sign-On con Microsoft Azure AD, completa i seguenti passaggi:

1. Nella barra laterale, seleziona Ruoli in Gestione degli accessi
2. Seleziona Create Role (Crea ruolo).
3. Scegli la federazione SAML 2.0 come entità affidabile
4. Scegli il provider Azure AD

5. Scegli Consenti l'accesso programmatico e alla console AWS di gestione
6. Seleziona Next: Permissions (Successivo: Autorizzazioni)
7. Allega le politiche di autorizzazione o passa a Next:Tags
8. Aggiungi tag opzionali o continua con Next:Review
9. Immettere un nome di ruolo in Role name (Nome ruolo). Questo esempio utilizzerà AzureSAMLRole
10. Fornisci una descrizione del ruolo
11. Scegli Crea ruolo per completare

Crea una IAM politica

Per creare una IAM policy per il Timestream per l'autenticazione LiveAnalytics JDBC Single Sign-On con Microsoft Azure AD, completa i seguenti passaggi:

1. Nella barra laterale, scegli Politiche in Gestione degli accessi
2. Scegli Crea politica e seleziona la scheda JSON
3. Aggiungi la seguente politica

```
{
"Version": "2012-10-17",
"Statement": [
    {
          "Effect": "Allow",
          "Action": [
                "iam:ListRoles",
                "iam:ListAccountAliases"
           ],
           "Resource": "*"
      }
]
}
```

4. Selezionare Creare policy
5. Inserire un nome per la policy. Questo esempio utilizzerà TimestreamAccessPolicy.
6. Scegli Crea politica
7. Nella barra laterale, scegli Ruoli in Gestione degli accessi.

8. Scegli il ruolo Azure AD creato in precedenza e scegli Allega policy in Autorizzazioni.
9. Seleziona la policy di accesso creata in precedenza.

Provisioning

Per fornire al provider di identità per Timestream l'autenticazione LiveAnalytics JDBC Single Sign-On con Microsoft Azure AD, completa i seguenti passaggi:

1. Torna al portale di Azure
2. Scegli Azure Active Directory nell'elenco dei servizi di Azure. Questo reindirizzerà alla pagina Default Directory
3. Scegli Applicazioni aziendali nella sezione Gestisci nella barra laterale
4. Scegli Provisioning
5. Scegli la modalità automatica per il metodo di provisioning
6. In Credenziali di amministratore, inserisci il tuo AwsAccessKeyID per clientsecret e per Secret Token SecretAccessKey
7. Imposta lo stato di approvvigionamento su Attivato
8. Scegli salva. Ciò consente ad Azure AD di caricare i ruoli necessari IAM
9. Una volta completato lo stato del ciclo corrente, scegli Utenti e gruppi nella barra laterale
10. Scegli + Aggiungi utente
11. Scegli l'utente Azure AD per cui fornire l'accesso a Timestream LiveAnalytics
12. Scegli il ruolo IAM Azure AD e il corrispondente Azure Identity Provider creato in AWS
13. Scegli Assegna

ODBC

Il ODBCdriver open source per Amazon LiveAnalytics Timestream for SQL fornisce un'interfaccia relazionale a LiveAnalytics Timestream per gli sviluppatori e consente la connettività da strumenti di business intelligence (BI) come Power BI Desktop e Microsoft Excel. Il Timestream per il LiveAnalytics ODBC driver è attualmente disponibile su Windows, macOS e Linux e supporta anche SSO Okta e Microsoft Azure Active Directory (AD).

Per ulteriori informazioni, consulta Amazon Timestream LiveAnalytics ODBC per la documentazione dei driver su. GitHub

Argomenti

Configurazione del Timestream per il conducente LiveAnalytics ODBC

Configura l'accesso a Timestream nel tuo account LiveAnalytics AWS

Se non hai ancora configurato il tuo AWS account per l'utilizzo di Timestream LiveAnalytics, segui le istruzioni riportate in. Accesso a Timestream per LiveAnalytics

Installa il driver sul tuo sistema ODBC

Scarica il programma di installazione del ODBC driver Timestream appropriato per il tuo sistema dal ODBC GitHubrepository, e segui le istruzioni di installazione applicabili al tuo sistema:.

- Guida all'installazione di Windows
- Guida all'installazione di macOS
- Guida all'installazione di Linux

Imposta il nome di una fonte di dati (DSN) per il ODBC driver

Segui le istruzioni contenute nella guida alla DSN configurazione del tuo sistema:

- DSNConfigurazione di Windows
- Configurazione macOS DSN
- Configurazione Linux DSN

Configura la tua applicazione di business intelligence (BI) in modo che funzioni con il ODBC driver

Di seguito sono riportate le istruzioni per impostare diverse applicazioni di BI comuni in modo che funzionino con il ODBC driver:

- Configurazione di Microsoft Power BI.

- Configurazione di Microsoft Excel
- Configurazione di Tableau

Per altre applicazioni

Sintassi e opzioni della stringa di connessione per il driver ODBC

La sintassi per specificare le opzioni della stringa di connessione per il driver è la ODBC seguente:

```
DRIVER={Amazon Timestream ODBC Driver};(option)=(value);
```

Le opzioni disponibili sono riportate di seguito:

Opzioni di connessione del driver

- **`Driver`**(obbligatorio): il driver utilizzato conODBC.

  L'impostazione predefinita è Amazon Timestream.
- **`DSN`**— Il nome dell'origine dati (DSN) da utilizzare per configurare la connessione.

  Il valore predefinito è `NONE`.
- **`Auth`**— La modalità di autenticazione. Deve essere una delle seguenti:
  - `AWS_PROFILE`— Utilizza la catena di credenziali predefinita.
  - `IAM`— Usa le AWS IAM credenziali.
  - `AAD`— Usa il provider di identità Azure Active Directory (AD).
  - `OKTA`— Usa il provider di identità Okta.

  Il valore predefinito è `AWS_PROFILE`.

Opzioni di configurazione degli endpoint

- **`EndpointOverride`**— L'override dell'endpoint per il servizio Timestream for. LiveAnalytics Questa è un'opzione avanzata che sostituisce la regione. Per esempio:

  ```
  query-cell2.timestream.us-east-1.amazonaws.com
  ```
- **`Region`**— La regione di firma per l'endpoint Timestream for service. LiveAnalytics

Il valore predefinito è `us-east-1`.

Opzione del fornitore di credenziali

- **`ProfileName`**— Il nome del profilo nel file di AWS configurazione.

  Il valore predefinito è `NONE`.

AWS IAMopzioni di autenticazione

- **`UID`**o **`AccessKeyId`**— L'id della chiave di accesso AWS dell'utente. Se entrambi `UID` `AccessKeyId` sono forniti nella stringa di connessione, il `UID` valore verrà utilizzato a meno che non sia vuoto.

  Il valore predefinito è `NONE`.
- **`PWD`**o **`SecretKey`**— La chiave di accesso segreta AWS dell'utente. Se entrambi `PWD` `SecretKey` sono forniti nella stringa di connessione, verrà utilizzato il `PWD` valore with a meno che non sia vuoto.

  Il valore predefinito è `NONE`.
- **`SessionToken`**— Il token di sessione temporaneo necessario per accedere a un database con l'autenticazione a più fattori (MFA) abilitata. Non includete un trailing = nell'input.

  Il valore predefinito è `NONE`.

SAMLopzioni di autenticazione basate su Okta

- **`IdPHost`**— Il nome host dell'IdP specificato.

  Il valore predefinito è `NONE`.
- **`UID`**o **`IdPUserName`**— Il nome utente per l'account IdP specificato. Se entrambi `UID` `IdPUserName` sono forniti nella stringa di connessione, il `UID` valore verrà utilizzato a meno che non sia vuoto.

  Il valore predefinito è `NONE`.
- **`PWD`**o **`IdPPassword`**— La password per l'account IdP specificato. Se entrambi `PWD` `IdPPassword` sono forniti nella stringa di connessione, il `PWD` valore verrà utilizzato a meno che non sia vuoto.

Il valore predefinito è NONE.

- **OktaApplicationID**— L'ID univoco fornito da OKTA associato al Timestream for application. LiveAnalytics Un posto dove trovare l'ID dell'applicazione (AppId) si trova nel `entityID` campo fornito nei metadati dell'applicazione. Un esempio è:

  ```
  entityID="http://www.okta.com//(IdPAppID)
  ```

  Il valore predefinito è NONE.
- **RoleARN**— L'Amazon Resource Name (ARN) del ruolo assunto dal chiamante.

  Il valore predefinito è NONE.
- **IdPARN**— L'Amazon Resource Name (ARN) del SAML provider IAM che descrive l'IdP.

  Il valore predefinito è NONE.

SAMLopzioni di autenticazione basate per Azure Active Directory

- **UID**o **IdPUserName**— Il nome utente per l'account IdP specificato.

  Il valore predefinito è NONE.
- **PWD**o **IdPPassword**— La password per l'account IdP specificato.

  Il valore predefinito è NONE.
- **AADApplicationID**— L'id univoco dell'applicazione registrata su Azure AD.

  Il valore predefinito è NONE.
- **AADClientSecret**— Il segreto del client associato all'applicazione registrata su Azure AD usato per autorizzare il recupero dei token.

  Il valore predefinito è NONE.
- **AADTenant**— L'ID del tenant di Azure AD.

  Il valore predefinito è NONE.
- **RoleARN**— L'Amazon Resource Name (ARN) del ruolo assunto dal chiamante.

  Il valore predefinito è NONE.
- **IdPARN**— L'Amazon Resource Name (ARN) del SAML provider IAM che descrive l'IdP.

Il valore predefinito è `NONE`.

AWS SDKOpzioni (avanzate)

- **`RequestTimeout`**— Il tempo in millisecondi in cui AWS SDK attende una richiesta di query prima del timeout. Qualsiasi valore non positivo disabilita il timeout della richiesta.

  Il valore predefinito è `3000`.
- **`ConnectionTimeout`**— Il tempo in millisecondi in cui AWS SDK attende il trasferimento dei dati su una connessione aperta prima del timeout. Il valore 0 disattiva il timeout della connessione. Questo valore non deve essere negativo.

  Il valore predefinito è `1000`.
- **`MaxRetryCountClient`**— Il numero massimo di tentativi per errori ripetibili con codici di errore 5xx nel. SDK Il valore non deve essere negativo.

  Il valore predefinito è `0`.
- **`MaxConnections`**— Il numero massimo di HTTP connessioni aperte contemporaneamente al servizio Timestream. Il valore deve essere positivo.

  Il valore predefinito è `25`.

ODBCopzioni di registrazione dei driver

- **`LogLevel`**— Il livello di registro per la registrazione dei driver. Deve essere un valore tra:
  - (0OFF).
  - (1ERROR).
  - (2WARNING).
  - (3INFO).
  - (4DEBUG).

  L'impostazione predefinita è 1 (ERROR).

  Avviso: le informazioni personali potrebbero essere registrate dal conducente quando si utilizza la modalità DEBUG di registrazione.
- **`LogOutput`**— Cartella in cui archiviare il file di registro.

L'impostazione predefinita è:

- Windows:`%USERPROFILE%`, o se non disponibile,`%HOMEDRIVE%%HOMEPATH%`.
- macOS e Linux: o se non disponibile`$HOME`, il campo della funzione `pw_dir getpwuid(getuid())` restituisce il valore.

SDKopzioni di registrazione

Il livello di AWS SDK registro è separato dal Timestream per il livello di registro del LiveAnalytics ODBC driver. L'impostazione di uno non influisce sull'altro.

Il livello di SDK registro viene impostato utilizzando la variabile di ambiente`TS_AWS_LOG_LEVEL`. I valori validi sono:

- `OFF`
- `ERROR`
- `WARN`
- `INFO`
- `DEBUG`
- `TRACE`
- `FATAL`

Se non `TS_AWS_LOG_LEVEL` è impostato, il livello di SDK registro viene impostato sul valore predefinito, ovvero`WARN`.

Connessione tramite proxy

Il ODBC driver supporta la connessione ad Amazon Timestream LiveAnalytics tramite un proxy. Per utilizzare questa funzionalità, configura le seguenti variabili di ambiente in base alle impostazioni del proxy:

- **`TS_PROXY_HOST`**— l'host proxy.
- **`TS_PROXY_PORT`**— Il numero di porta del proxy.
- **`TS_PROXY_SCHEME`**— Lo schema proxy, `http` o`https`.
- **`TS_PROXY_USER`**— Il nome utente per l'autenticazione proxy.
- **`TS_PROXY_PASSWORD`**— La password utente per l'autenticazione proxy.

- **TS_PROXY_SSL_CERT_PATH**— Il file del SSL certificato da utilizzare per la connessione a un HTTPS proxy.
- **TS_PROXY_SSL_CERT_TYPE**— Il tipo di SSL certificato del client proxy.
- **TS_PROXY_SSL_KEY_PATH**— Il file di chiave privata da utilizzare per la connessione a un HTTPS proxy.
- **TS_PROXY_SSL_KEY_TYPE**— Il tipo di file di chiave privata utilizzato per connettersi a un HTTPS proxy.
- **TS_PROXY_SSL_KEY_PASSWORD**— La passphrase del file di chiave privata utilizzato per connettersi a un proxy. HTTPS

Esempi di stringhe di connessione per il Timestream for driver LiveAnalytics ODBC

Esempio di connessione al ODBC driver con credenziali IAM

```
Driver={Amazon Timestream ODBC Driver};Auth=IAM;AccessKeyId=(your access key
 ID);secretKey=(your secret key);SessionToken=(your session token);Region=us-east-2;
```

Esempio di connessione al ODBC driver con un profilo

```
Driver={Amazon Timestream ODBC Driver};ProfileName=(the profile name);region=us-west-2;
```

Il driver tenterà di connettersi utilizzando le credenziali fornite in o`~/.aws/credentials`, se un file è specificato nella variabile di ambiente`AWS_SHARED_CREDENTIALS_FILE`, utilizzando le credenziali in quel file.

Esempio di connessione al ODBC driver con Okta

```
driver={Amazon Timestream ODBC Driver};auth=okta;region=us-west-2;idPHost=(your host at
 Okta);idPUsername=(your user name);idPPassword=(your password);OktaApplicationID=(your
 Okta AppId);roleARN=(your role ARN);idPARN=(your Idp ARN);
```

Esempio di connessione al ODBC driver con Azure Active Directory () AAD

```
driver={Amazon Timestream ODBC Driver};auth=aad;region=us-west-2;idPUsername=(your
 user name);idPPassword=(your password);aadApplicationID=(your AAD
 AppId);aadClientSecret=(your AAD client secret);aadTenant=(your AAD
 tenant);roleARN=(your role ARN);idPARN=(your idP ARN);
```

Esempio di connessione al ODBC driver con un endpoint specificato e un livello di registro pari a 2 () WARNING

```
Driver={Amazon Timestream ODBC Driver};Auth=IAM;AccessKeyId=(your access
 key ID);secretKey=(your secret key);EndpointOverride=ingest.timestream.us-
west-2.amazonaws.com;Region=us-east-2;LogLevel=2;
```

Risoluzione dei problemi di connessione con il driver ODBC

> **Note**
>
> Quando il nome utente e la password sono già specificati inDSN, non è necessario specificarli nuovamente quando il gestore del ODBC driver li richiede.

Un codice di errore `01S02` con un messaggio `Re-writing` *`(connection string option)`* `(have you specified it several times?` si verifica quando un'opzione della stringa di connessione viene passata più di una volta nella stringa di connessione. Se si specifica un'opzione più di una volta, viene generato un errore. Quando si effettua una connessione con una stringa di connessione DSN e una stringa di connessione, se un'opzione di connessione è già specificata inDSN, non specificarla nuovamente nella stringa di connessione.

VPCpunti finali ()AWS PrivateLink

Puoi stabilire una connessione privata tra te VPC e Amazon Timestream creando un endpoint LiveAnalytics di interfaccia. VPC Per ulteriori informazioni, consulta VPCendpoint ()AWS PrivateLink.

Best practice

Per sfruttare appieno i vantaggi di Amazon Timestream LiveAnalytics for, segui le best practice descritte di seguito.

> **Note**
>
> Quando esegui proof-of-concept le applicazioni, considera la quantità di dati che l'applicazione accumulerà nell'arco di alcuni mesi o anni, valutando al contempo le prestazioni e la scalabilità di Timestream for. LiveAnalytics Man mano che i dati crescono nel tempo, noterai che le prestazioni di Timestream for LiveAnalytics rimangono per lo più invariate, perché la sua architettura serverless può sfruttare enormi quantità di parallelismo per

l'elaborazione di grandi volumi di dati e scalare automaticamente in base alle esigenze della tua applicazione.

Argomenti

Modellazione dei dati

Amazon Timestream LiveAnalytics for è progettato per raccogliere, archiviare e analizzare dati di serie temporali da applicazioni e dispositivi che emettono una sequenza di dati con un timestamp. Per prestazioni ottimali, i dati inviati a Timestream LiveAnalytics devono avere caratteristiche temporali e il tempo deve essere un componente essenziale dei dati.

Timestream for LiveAnalytics offre la flessibilità necessaria per modellare i dati in diversi modi per soddisfare i requisiti dell'applicazione. In questa sezione, trattiamo diversi di questi modelli e forniamo linee guida per ottimizzare costi e prestazioni. Acquisite familiarità con i principali argomenti relativi a Timestream per LiveAnalytics concetti quali dimensioni e misure. In questa sezione, scoprirai di più su:

Nel decidere se creare una singola tabella o più tabelle per archiviare i dati, considera quanto segue:

- Quali dati inserire nella stessa tabella rispetto a quando desideri separare i dati su più tabelle e database.
- Come scegliere tra Timestream per record a LiveAnalytics più misure e record a misura singola e i vantaggi della modellazione utilizzando record multimisura, soprattutto quando l'applicazione tiene traccia di più misurazioni contemporaneamente e istantaneamente.
- Quali attributi modellare come dimensioni o come misure.

- Come utilizzare in modo efficace gli attributi del nome della misura per ottimizzare la latenza delle query.

Argomenti

Tabella singola vs. tabelle multiple

Mentre si modellano i dati nell'applicazione, un altro aspetto importante è come modellare i dati in tabelle e database. I database e le tabelle in Timestream for LiveAnalytics sono astrazioni per il controllo degli accessi, la specificazione delle KMS chiavi, i periodi di conservazione, ecc. Timestream consente il partizionamento LiveAnalytics automatico dei dati ed è progettato per scalare le risorse in base al carico e ai requisiti di acquisizione, archiviazione e query delle applicazioni.

Una tabella in Timestream for è LiveAnalytics scalabile fino a petabyte di dati archiviati, decine di gigabyte/sec di scrittura dei dati e le query possono elaborarne centinaia all'ora. TBs Le query in Timestream for LiveAnalytics possono estendersi su più tabelle e database, fornendo join e unioni per fornire un accesso senza interruzioni ai dati su più tabelle e database. Pertanto, la scala dei dati o il volume delle richieste di solito non sono la preoccupazione principale quando si decide come organizzare i dati in Timestream for. LiveAnalytics Di seguito sono riportate alcune considerazioni importanti per decidere quali dati collocare nella stessa tabella anziché in tabelle diverse o tabelle in database diversi.

- Le politiche di conservazione dei dati (conservazione dell'archivio di memoria, conservazione dell'archivio magnetico, ecc.) sono supportate in base alla granularità di una tabella. Pertanto, i dati che richiedono politiche di conservazione diverse devono trovarsi in tabelle diverse.
- AWS KMS le chiavi utilizzate per crittografare i dati sono configurate a livello di database. Pertanto, i diversi requisiti delle chiavi di crittografia implicano che i dati dovranno trovarsi in database diversi.
- Timestream for LiveAnalytics supporta il controllo degli accessi basato sulle risorse con la granularità di tabelle e database. Considera i tuoi requisiti di controllo degli accessi quando decidi quali dati scrivere nella stessa tabella rispetto a tabelle diverse.

- Tieni presente i limiti relativi al numero di dimensioni, ai nomi delle misure e ai nomi degli attributi multimisura quando decidi quali dati archiviare in quale tabella.
- Quando decidi come organizzare i dati, prendi in considerazione il carico di lavoro e i modelli di accesso delle query, poiché la latenza delle query e la facilità di scrittura delle query dipenderanno da ciò.
  - Se si archiviano i dati su cui si eseguono frequentemente query nella stessa tabella, in genere si semplificherà il modo in cui si scrivono le query, in modo da evitare spesso la necessità di scrivere join, unioni o espressioni di tabella comuni. Ciò comporta in genere anche una minore latenza delle query. È possibile utilizzare i predicati sulle dimensioni e sui nomi delle misure per filtrare i dati pertinenti alle query.

    Ad esempio, si consideri un caso in cui si archiviano dati da dispositivi situati in sei continenti. Se le tue query accedono spesso ai dati provenienti da tutti i continenti per ottenere una vista aggregata globale, l'archiviazione dei dati di questi continenti nella stessa tabella semplificherà la scrittura delle query. D'altra parte, se memorizzi dati su tabelle diverse, puoi comunque combinare i dati nella stessa query, tuttavia dovrai scrivere una query per unire i dati di più tabelle.
  - Timestream for LiveAnalytics utilizza il partizionamento e l'indicizzazione adattivi dei dati. Pertanto alle query vengono addebitati solo i dati pertinenti alle tue domande. Ad esempio, se disponi di una tabella che memorizza i dati di un milione di dispositivi in sei continenti, se la tua query contiene predicati del tipo `WHERE device_id = 'abcdef'` or`WHERE continent = 'North America'`, alle query vengono addebitati solo i dati relativi al dispositivo o al continente.
  - Ove possibile, se si utilizza il nome della misura per separare i dati nella stessa tabella che non vengono emessi contemporaneamente o non vengono interrogati frequentemente, utilizzando i predicati come `WHERE measure_name = 'cpu'` nella query, non solo si ottengono i vantaggi di misurazione, ma Timestream for LiveAnalytics può anche eliminare efficacemente le partizioni che non hanno il nome della misura utilizzato nel predicato di query. Ciò consente di archiviare i dati correlati con nomi di misure diversi nella stessa tabella senza influire sulla latenza o sui costi delle query ed evita la diffusione dei dati in più tabelle. Il nome della misura viene utilizzato essenzialmente per partizionare i dati ed eliminare le partizioni non pertinenti alla query.

Record multimisura vs. record a misura singola

Timestream for LiveAnalytics consente di scrivere dati con più misure per record (multimisura) o una singola misura per record (misura singola).

Record multimisura

In molti casi d'uso, un dispositivo o un'applicazione monitorata può emettere più metriche o eventi contemporaneamente. In questi casi, puoi memorizzare tutte le metriche emesse nello stesso timestamp nello stesso record multimisura. Cioè, tutte le misure memorizzate nello stesso record multimisura vengono visualizzate come colonne diverse nella stessa riga di dati.

Considerate, ad esempio, che la vostra applicazione emetta metriche come cpu, memory, disk_iops da un dispositivo misurato nello stesso istante. Di seguito è riportato un esempio di tabella di questo tipo in cui più metriche emesse nello stesso istante vengono memorizzate nella stessa riga. Vedrai che due host emettono le metriche una volta al secondo.

Hostname (Nome host)	measure_name	Orario	cpu	Memoria	disk_iops
Host-24 GJU	metriche	01/12/2021 19:00:00	35	54,9	38,2
Host-24 GJU	metriche	01/12/2021 19:00:01	36	58	39
Host-28 GJU	metriche	01/12/2021 19:00:00	15	55	92
Host-28 GJU	metriche	01/12/2021 19:00:01	16	50	40

Record a misura singola

I record a singola misura sono adatti quando i dispositivi emettono metriche diverse in periodi di tempo diversi o si utilizza una logica di elaborazione personalizzata che emette metrics/events at different time periods (for instance, when a device's reading/state modifiche). Poiché ogni misura ha un timestamp unico, le misure possono essere archiviate nei propri record in Timestream for. LiveAnalytics Ad esempio, prendiamo in considerazione un sensore IoT, che monitora la temperatura e l'umidità del suolo, che emette un record solo quando rileva un cambiamento rispetto alla voce precedente riportata. L'esempio seguente fornisce un esempio di tali dati emessi utilizzando record di misure singole.

device_id	measure_name	Orario	measure_value::double	measure_value::bigint
sensor-sea478	temperature	2021-12-01 19:22:32	35	NULL
sensore-sea478	temperature	2021-12-01 18:07:51	36	NULL
sensore-sea478	umidità	2021-12-01 19:05:30	NULL	21
sensore-sea478	umidità	2021-12-01 19:00:01	NULL	23

Confronto tra record a misura singola e multimisura

Timestream for LiveAnalytics offre la flessibilità necessaria per modellare i dati come record a misura singola o multipla a seconda dei requisiti e delle caratteristiche dell'applicazione. Una singola tabella può memorizzare sia record a singola misura che a più misure, se i requisiti dell'applicazione lo richiedono. In generale, quando l'applicazione emette più misurazioni/eventi contemporaneamente, la modellazione dei dati come record multimisura è generalmente consigliata per un accesso efficiente ai dati e un'archiviazione dei dati conveniente.

Ad esempio, se si considera un caso DevOps d'uso che monitora metriche ed eventi da centinaia di migliaia di server, ogni server emette periodicamente 20 metriche e 5 eventi, in cui gli eventi e le metriche vengono emessi nello stesso istante. Tali dati possono essere modellati utilizzando record a singola misura o utilizzando record a più misure (consulta il generatore di dati open source per lo schema risultante). In questo caso d'uso, la modellazione dei dati utilizzando record a più misure rispetto a record a misura singola produce:

- Misurazione dell'ingestione: i record con più misure consentono di ridurre di circa il 40% i byte di ingestione scritti.
- Ingestione in batch: i record su più misure comportano l'invio di batch di dati più grandi, il che implica che i client abbiano bisogno di meno thread e meno thread per elaborare l'ingestione. CPU
- Misurazione dello storage: i record su più misure riducono lo storage di circa 8 volte, con un notevole risparmio di storage sia per la memoria che per l'archiviazione magnetica.

- Latenza delle query: i record con più misure determinano una latenza di query inferiore per la maggior parte dei tipi di query rispetto ai record a misura singola.
- Byte misurati dalle query: per le query che analizzano dati inferiori a 10 MB, i record a misura singola e quelli a più misure sono comparabili. Per le interrogazioni che accedono a una singola misura e scansionano dati di oltre 10 MB, i record a misurazione singola generano in genere una riduzione dei byte misurati. Per le query che fanno riferimento a 3 o più misure, i record con più misure comportano una riduzione dei byte misurati.
- Facilità di esprimere interrogazioni con più misure: quando le query fanno riferimento a più misure, la modellazione dei dati con record multimisura consente di scrivere query più semplici da scrivere.

I fattori precedenti varieranno a seconda del numero di metriche monitorate, delle dimensioni dei dati, ecc. Sebbene l'esempio precedente fornisca alcuni dati concreti, ad esempio, vediamo in molti scenari applicativi e casi d'uso in cui se l'applicazione emette più misure contemporaneamente, l'archiviazione dei dati come record multimisura è più efficace. Inoltre, i record multimisura offrono la flessibilità dei tipi di dati e la memorizzazione di più altri valori come contesto (ad esempio, l'archiviazione di richieste e timestamp aggiuntiviIDs, di cui parleremo più avanti).

Nota che un record multimisura può anche modellare misure sparse come nell'esempio precedente per i record di misura singola: puoi usare measure_name per memorizzare il nome della misura e utilizzare un nome di attributo multimisura generico, come value_double per memorizzare `DOUBLE` le misure, value_bigint per memorizzare `BIGINT` le misure, value_timestamp per memorizzare `TIMESTAMP` valori aggiuntivi, ecc.

Dimensioni e misure

Una tabella in Timestream per ti LiveAnalytics consente di memorizzare dimensioni (attributi identificativi del dispositivo/dati che stai archiviando) e misure (le metriche/valori che stai monitorando), vedi Timestream per i LiveAnalytics concetti per maggiori dettagli. Mentre state modellando la vostra applicazione su Timestream, il modo in cui mappate i dati in dimensioni e misure influisce LiveAnalytics sulla latenza di inserimento e di interrogazione. Di seguito sono riportate le linee guida su come modellare i dati sotto forma di dimensioni e misure da applicare al caso d'uso.

Scelta delle dimensioni

I dati che identificano la fonte che invia i dati delle serie temporali sono una scelta naturale per le dimensioni, che sono attributi che non cambiano nel tempo. Ad esempio, se disponi di un server che emette metriche, gli attributi che identificano il server, come hostname, Region, rack, Availability

Zone, sono candidati per le dimensioni. Allo stesso modo, per un dispositivo IoT con più sensori che riportano dati di serie temporali, l'id del dispositivo, l'id del sensore, ecc. sono candidati per le dimensioni.

Se si scrivono dati come record multimisura, le dimensioni e gli attributi multimisura vengono visualizzati come colonne nella tabella quando si esegue DESCRIBE o si esegue un'SELECTistruzione sulla tabella. Pertanto, quando si scrivono le query, è possibile utilizzare liberamente le dimensioni e le misure della stessa query. Tuttavia, quando create il record di scrittura per importare dati, tenete presente quanto segue quando scegliete quali attributi sono specificati come dimensioni e quali sono valori di misura:

- I nomi delle dimensioni, i valori, il nome della misura e il timestamp identificano in modo univoco i dati delle serie temporali. Timestream for LiveAnalytics utilizza questo identificatore univoco per deduplicare automaticamente i dati. Cioè, se Timestream for LiveAnalytics riceve due punti dati con gli stessi valori di nomi di dimensione, valori di dimensione, nome di misura e timestamp, se i valori hanno lo stesso numero di versione, Timestream for deduplica. LiveAnalytics Se la nuova richiesta di scrittura ha una versione inferiore a quella dei dati già esistenti in Timestream for, la richiesta di scrittura viene rifiutata. LiveAnalytics Se la nuova richiesta di scrittura ha una versione superiore, il nuovo valore sovrascrive il vecchio valore. Pertanto, la scelta dei valori delle dimensioni influirà su questo comportamento di deduplicazione.
- I nomi e i valori delle dimensioni non possono essere aggiornati, mentre il valore di misura può esserlo. Pertanto, qualsiasi dato che potrebbe richiedere aggiornamenti viene meglio modellato come valori di misura. Ad esempio, se in fabbrica è presente una macchina il cui colore può cambiare, è possibile modellare il colore come valore di misura, a meno che non si desideri utilizzare il colore anche come attributo identificativo necessario per la deduplicazione. In altre parole, i valori di misura possono essere utilizzati per memorizzare attributi che cambiano solo lentamente nel tempo.

Tieni presente che una tabella in Timestream for LiveAnalytics non limita il numero di combinazioni univoche di nomi e valori delle dimensioni. Ad esempio, puoi avere miliardi di queste combinazioni di valori uniche memorizzate in una tabella. Tuttavia, come vedrete negli esempi seguenti, un'attenta scelta delle dimensioni e delle misure può ottimizzare in modo significativo la latenza delle richieste, in particolare per le query.

Unico IDs nelle dimensioni

Se lo scenario applicativo richiede la memorizzazione di un identificatore univoco per ogni punto dati (ad esempio, un ID di richiesta, un ID di transazione o un ID di correlazione), la modellazione

dell'attributo ID come valore di misura comporterà una latenza di query significativamente migliore. Quando si modellano i dati con record multimisura, l'ID viene visualizzato nella stessa riga nel contesto delle altre dimensioni e dei dati delle serie temporali, in modo che le query possano continuare a utilizzarli in modo efficace. Ad esempio, considerando un caso DevOps d'uso in cui ogni punto dati emesso da un server ha un attributo ID di richiesta univoco, la modellazione dell'ID della richiesta come valore di misura comporta una latenza di query fino a 4 volte inferiore su diversi tipi di query, anziché modellare l'ID di richiesta univoco come dimensione.

È possibile utilizzare l'analogia simile per attributi che non sono completamente unici per ogni punto dati, ma hanno centinaia di migliaia o milioni di valori univoci. È possibile modellare questi attributi sia come dimensioni che come valori di misura. È consigliabile modellarlo come dimensione se i valori sono necessari per la deduplicazione sul percorso di scrittura, come illustrato in precedenza, oppure lo si utilizza spesso come predicato (ad esempio, nella `WHERE` clausola con un predicato di uguaglianza su un valore di quell'attributo, ad esempio `device_id = ‘abcde’` quando l'applicazione monitora milioni di dispositivi) nelle query.

Ampia gamma di tipi di dati con record multimisura

I record multimisura offrono la flessibilità necessaria per modellare efficacemente i dati. I dati archiviati in un record di più misure vengono visualizzati come colonne nella tabella in modo simile alle dimensioni, garantendo così la stessa facilità di ricerca delle dimensioni e dei valori di misura. Hai visto alcuni di questi modelli negli esempi discussi in precedenza. Di seguito troverai modelli aggiuntivi per utilizzare in modo efficace i record multimisura per soddisfare i casi d'uso dell'applicazione.

I record multimisura supportano gli attributi dei tipi di dati`DOUBLE`,`BIGINT`, `VARCHAR``BOOLEAN`, e. `TIMESTAMP` Pertanto, si adattano naturalmente a diversi tipi di attributi:

- Informazioni sulla posizione: ad esempio, se si desidera tracciare la posizione (espressa come latitudine e longitudine), modellarla come attributo multimisura comporterà una minore latenza di interrogazione rispetto alla memorizzazione delle stesse come `VARCHAR` dimensioni, specialmente quando si hanno predicati sulla latitudine e sulle longitudini.
- Più timestamp in un record: se lo scenario dell'applicazione richiede di tenere traccia di più timestamp per un record di serie temporali, è possibile modellarli come attributi aggiuntivi nel record multimisura. Questo modello può essere utilizzato per archiviare dati con timestamp futuri o passati. Tieni presente che ogni record utilizzerà comunque il timestamp nella colonna dell'ora per partizionare, indicizzare e identificare in modo univoco un record.

In particolare, se nella query sono presenti dati numerici o timestamp su cui sono presenti predicati, la modellazione di tali attributi come attributi multimisura anziché come dimensioni comporterà una minore latenza delle query. Questo perché quando si modellano tali dati utilizzando i ricchi tipi di dati supportati nei record multimisura, è possibile esprimere i predicati utilizzando tipi di dati nativi anziché trasferire valori da VARCHAR un altro tipo di dati se tali dati sono stati modellati come dimensioni.

Utilizzo del nome della misura con record multimisura

Le tabelle in Timestream LiveAnalytics supportano un attributo speciale (o colonna) chiamato nome della misura. Specificate un valore per questo attributo per ogni record per cui scrivete su Timestream. LiveAnalytics Per i record a singola misura, è naturale utilizzare il nome della metrica (ad esempio cpu, memoria per le metriche del server o temperatura, pressione per le metriche dei sensori). Quando si utilizzano record multimisura, poiché gli attributi di un record multimisura sono denominati (e questi nomi diventano nomi di colonne nella tabella), cpu, memoria o temperatura, la pressione può diventare nomi di attributi con più misure. Quindi una domanda naturale è come utilizzare efficacemente il nome della misura.

Timestream for LiveAnalytics utilizza i valori nell'attributo del nome della misura per partizionare e indicizzare i dati. Pertanto, se una tabella ha più nomi di misure diversi e se le query utilizzano tali valori come predicati di query, Timestream for LiveAnalytics può utilizzare il partizionamento e l'indicizzazione personalizzati per eliminare i dati non pertinenti alle query. Ad esempio, se la tabella contiene nomi di misure di CPU e memoria e la query ha un predicato`WHERE measure_name = 'cpu'`, Timestream for LiveAnalytics può eliminare efficacemente i dati per i nomi di misure non pertinenti alla query, ad esempio le righe con memoria dei nomi di misura in questo esempio. Questa eliminazione si applica anche quando si utilizzano nomi di misure con record di più misure. È possibile utilizzare efficacemente l'attributo measure name come attributo di partizionamento per una tabella. Il nome della misura insieme ai nomi e ai valori delle dimensioni e l'ora vengono utilizzati per partizionare i dati in un flusso temporale per tabella. LiveAnalytics Tieni presente i limiti al numero di nomi di misura univoci consentiti in un Timestream for table. LiveAnalytics Si noti inoltre che il nome di una misura è associato anche a un tipo di dati del valore di misura, ad esempio, un singolo nome di misura può essere associato a un solo tipo di valore di misura. Questo tipo può essere uno tra `DOUBLE` `BIGINT``BOOLEAN`,`VARCHAR`, e`MULTI`. I record a più misure memorizzati con un nome di misura avranno il seguente `MULTI` tipo di dati. Poiché un singolo record multimisura può memorizzare più metriche con tipi di dati diversi (`DOUBLE`,, `BIGINT``VARCHAR`, e`TIMESTAMP`)`BOOLEAN`, è possibile associare dati di tipi diversi in un record con più misure.

Le sezioni seguenti descrivono alcuni esempi diversi di come l'attributo measure name può essere efficacemente utilizzato per raggruppare diversi tipi di dati nella stessa tabella.

Sensori IoT che segnalano qualità e valore

Immagina di avere un'applicazione che monitora i dati provenienti dai sensori IoT. Ogni sensore rileva diverse misure, come temperatura, pressione. Oltre ai valori effettivi, i sensori segnalano anche la qualità delle misurazioni, che è una misura dell'accuratezza della lettura e un'unità di misura. Poiché qualità, unità e valore vengono emessi insieme, possono essere modellati come record multimisura, come mostrato nei dati di esempio riportati di seguito, in cui device_id è una dimensione, qualità, valore e unità sono attributi multimisura:

device_id	measure_name	Orario	Qualità	Valore	Unità
sensor-sea478	temperature	2021-12-01 19:22:32	92	35	c
sensor-sea478	temperature	2021-12-01 18:07:51	93	34	c
sensor-sea478	pressure	2021-12-01 19:05:30	98	31	psi
sensor-sea478	pressure	2021-12-01 19:00:01	24	132	psi

Questo approccio consente di combinare i vantaggi dei record multimisura con il partizionamento e l'eliminazione dei dati utilizzando i valori di measure name. Se le query fanno riferimento a una singola misura, ad esempio la temperatura, è possibile includere un predicato del nome della misura nella query. Di seguito è riportato un esempio di tale interrogazione, che proietta anche l'unità per le misurazioni la cui qualità è superiore a 90.

```
SELECT device_id, time, value AS temperature, unit
FROM db.table
WHERE time > ago(1h)
    AND measure_name = 'temperature'
    AND quality > 90
```

L'utilizzo del predicato measure_name sulla query consente a Timestream di eliminare efficacemente partizioni e dati non pertinenti LiveAnalytics alla query, migliorando così la latenza della query.

È anche possibile archiviare tutte le metriche nello stesso record multimisura se tutte le metriche vengono emesse con lo stesso timestamp e/o più metriche vengono interrogate insieme nella stessa query. Ad esempio, puoi costruire un record multimisura con gli attributi temperature_quality, temperature_value, temperature_unit, pressure_quality, pressure_value, pressure_unit, ecc. Molti dei punti discussi in precedenza sulla modellazione dei dati utilizzando record a singola misura anziché a più misure si applicano alla decisione su come modellare i dati. Considerate i modelli di accesso alle query e il modo in cui i dati vengono generati per scegliere un modello che ottimizzi i costi, l'inserimento e la latenza delle query e la facilità di scrittura delle query.

Diversi tipi di metriche nella stessa tabella

Un altro caso d'uso in cui è possibile combinare record di più misure con i valori dei nomi delle misure consiste nel modellare diversi tipi di dati emessi indipendentemente dallo stesso dispositivo. Considerate il caso d'uso del DevOps monitoraggio: i server emettono due tipi di dati: metriche emesse regolarmente ed eventi irregolari. Un esempio di questo approccio è lo schema discusso nel generatore di dati che modella un caso d'uso. DevOps In questo caso, è possibile memorizzare i diversi tipi di dati emessi dallo stesso server nella stessa tabella utilizzando nomi di misure diversi. Ad esempio, tutte le metriche emesse nello stesso istante vengono archiviate con le metriche dei nomi delle misure. Tutti gli eventi emessi in un istante temporale diverso dalle metriche vengono archiviati con gli eventi relativi ai nomi delle misure. Lo schema di misura per la tabella (ad esempio, l'output della `SHOW MEASURES` query) è:

measure_name	data_type	Dimensioni
events	multi	[{"data_type» :"varchar», "dimension_name» :"availability_zone "}, {" data_type» :"varchar», "dimension_name» :"microservice_name "}, {" data_type» :"varchar», "dimension_name» :"nome_istanza "}, {" tipo_dati» :"varchar» :"varchar», "dimension_name» :"process_name "}, {" data_type» :"varchar», "dimension_name» :"jdk_version "}, {" data_type» :"varchar», "dimension_name» :"cell

measure_name	data_type	Dimensioni
		"}, {" data_type» :"varchar», "dimension_name» :"region "}, {" tipo_dati» :"varchar», "dimension_name» :"region "}, {" tipo_dati» :"varchar», "dimension_name» :"region "}, {" tipo_dati» :"varchar »» :"varchar», "nome_dim ensione» :"silo "}]

measure_name	data_type	Dimensioni
metriche	multi	[{"data_type» :"varchar», "dimension_name» :"availab ility_zone "}, {" data_type » :"varchar», "dimensio n_name» :"microservice_nam e "}, {" data_type» :"varchar», "dimension_name» :"nome_is tanza "}, {" tipo_dati» :"varchar » :"varchar», "dimensio n_name» :"os_version "}, {" data_type» :"varchar», "dimension_name» :"cell "}, {" data_type» :"varchar», "dimension_name» :"region "}, {" data_type» :"varchar », "dimension_name» :"silo "}, {" tipo_dati» :"varchar » :"dimension_name» :"silo "}, {" tipo_dati» :"varchar » :"dimension_name» :"silo "}, {" tipo_dati» :"varchar » :"dimension_name» :"silo "}, {" tipo_dati» :"varchar » :"dimension_name» :"silo "} «varchar», "nome_dim ensione» :"tipo_istanza "}]

In questo caso, puoi vedere che gli eventi e le metriche hanno anche diversi set di dimensioni, dove gli eventi hanno dimensioni diverse jdk_version e process_name mentre le metriche hanno dimensioni instance_type e os_version.

L'utilizzo di nomi di misure diversi consente di scrivere query con predicati in modo da ottenere solo le metriche. `WHERE measure_name = 'metrics'` Inoltre, avere tutti i dati emessi dalla stessa istanza nella stessa tabella implica che puoi anche scrivere una query più semplice con il predicato

instance_name per ottenere tutti i dati per quell'istanza. Ad esempio, un predicato del modulo `WHERE instance_name = 'instance-1234'` senza un predicato measure_name restituirà tutti i dati per un'istanza specifica del server.

Consigli per il partizionamento di record multimisura

> **Important**
>
> Questa sezione è obsoleta!
> Queste raccomandazioni non sono aggiornate. Il partizionamento è ora meglio controllato utilizzando chiavi di partizione definite dal cliente.

Abbiamo visto che nell'ecosistema delle serie temporali c'è un numero crescente di carichi di lavoro che richiedono l'acquisizione e l'archiviazione di enormi quantità di dati e allo stesso tempo richiedono risposte alle query a bassa latenza quando si accede ai dati tramite un set di valori dimensionali ad alta cardinalità.

A causa di tali caratteristiche, i consigli contenuti in questa sezione saranno utili per i carichi di lavoro dei clienti che presentano le seguenti caratteristiche.

- Ha adottato o desidera adottare record multimisura.
- Aspettatevi di ricevere nel sistema un volume elevato di dati che verranno archiviati per lunghi periodi.
- Richiedono tempi di risposta a bassa latenza per i loro modelli di accesso (interrogazione) principali.
- Sappi che i modelli di interrogazione più importanti implicano una condizione di filtraggio di qualche tipo nel predicato. Questa condizione di filtraggio si basa su una dimensione di cardinalità elevata. Ad esempio, considera eventi o aggregazioni per, ServerID UserId DeviceId, host-name e così via.

In questi casi un unico nome per tutte le misure multimisura non sarà di aiuto, poiché il nostro motore utilizza nomi con più misure per partizionare i dati e avere un unico valore limita il vantaggio di partizione che si ottiene. Il partizionamento di questi record si basa principalmente su due dimensioni. Supponiamo che il tempo sia sull'asse x e su un hash di nomi di dimensione e quindi sull'`measure_name`asse y. `measure_name`In questi casi funziona quasi come una chiave di partizionamento.

La nostra raccomandazione è la seguente.

- Quando modellate i dati per casi d'uso come quello che abbiamo menzionato, utilizzate un modello `measure_name` che sia una derivazione diretta del modello di accesso alle query principale. Per esempio:
  - Il tuo caso d'uso richiede il monitoraggio delle prestazioni delle applicazioni e della QoE dal punto di vista dell'utente finale. Potrebbe trattarsi anche del tracciamento delle misurazioni per un singolo server o dispositivo IoT.
  - Se stai eseguendo interrogazioni e filtrando UserId in base a, al momento dell'inserimento devi trovare il modo migliore per associarti a. `measure_name` UserId
  - Poiché una tabella multimisura può contenere solo 8192 nomi di misure diversi, qualunque formula venga adottata non dovrebbe generare più di 8192 valori diversi.
- Un approccio che abbiamo applicato con successo per i valori di stringa consiste nell'applicare un algoritmo di hashing al valore della stringa. Quindi esegui l'operazione del modulo con il valore assoluto del risultato dell'hash e 8192.

```
measure_name = getMeasureName(UserId)
int getMeasureName(value) {
    hash_value =  abs(hash(value))
    return hash_value % 8192
}
```

- Abbiamo anche aggiunto `abs()` di rimuovere il segno, eliminando la possibilità che i valori vadano da -8192 a 8192. Questa operazione deve essere eseguita prima dell'operazione del modulo.
- Utilizzando questo metodo, le query possono essere eseguite in una frazione del tempo necessario per l'esecuzione su un modello di dati non partizionato.
- Quando interrogate i dati, assicuratevi di includere una condizione di filtro nel predicato che utilizzi il nuovo valore derivato di measure_name. Per esempio:
  - ```
    SELECT * FROM your_database.your_table
    WHERE host_name = 'Host-1235' time BETWEEN '2022-09-01'
        AND '2022-09-18'
        AND measure_name = (SELECT
     cast(abs(from_big_endian_64(xxhash64(CAST('HOST-1235' AS varbinary))))%8192 AS
     varchar))
    ```
  - Ciò ridurrà al minimo il numero totale di partizioni scansionate per ottenere dati che si tradurranno in query più rapide nel tempo.
    ```

Tieni presente che se desideri ottenere i vantaggi di questo schema di partizione, l'hash deve essere calcolato sul lato client e passato a Timestream LiveAnalytics come valore statico al motore di query. L'esempio precedente fornisce un modo per verificare che l'hash generato possa essere risolto dal motore quando necessario.

time	host_name	posizione	tipo_server	cpu_usage	memoria_disponibile	cpu_temp
2022-09-07 21:48:44 .000 0	ospitare-1235	us-east1	5,8 xl	55	16,2	78
R2022-09-07 21:48:44 .000 0	host-3587	Stati Uniti-West1	5,8 xl	62	18,1	81
2022-09-07 21:48:450 00 000000	host-258743	eu-central	5,8 xl	88	9.4	91
2022-09-07 21:48:45 .000 0	host-35654	Stati Uniti - East2	5,8 xl	29	24	54
R2022-09-07 21:48:45 .000 0	ospite-254	Stati Uniti-West1	5,8 xl	44	32	48

Per generare il codice associato `measure_name` seguendo la nostra raccomandazione, esistono due percorsi che dipendono dal modello di ingestione.

1. Per l'ingestione in batch di dati storici: puoi aggiungere la trasformazione al tuo codice di scrittura se intendi utilizzare il tuo codice per il processo in batch.

   Basandosi sull'esempio precedente.

```
        List<String> hosts = new ArrayList<>();

        hosts.add("host-1235");
        hosts.add("host-3587");
        hosts.add("host-258743");
        hosts.add("host-35654");
        hosts.add("host-254");

        for (String h: hosts){
            ByteBuffer buf2 = ByteBuffer.wrap(h.getBytes());
            partition = abs(hasher.hash(buf2, 0L)) % 8192;
            System.out.println(h + " - " + partition);

        }
```

Output

```
host-1235 - 6445
host-3587 - 6399
host-258743 - 640
host-35654 - 2093
host-254 - 7051
```

Set di dati risultante

time	host_name	posizione	measure_name	tipo_server	cpu_usage	memoria_disponibile	cpu_temp
2022-09-07 21:48:44 .00	ospitare-1235	us-east1	6445	5,8 xl	55	16,2	78
R2022-09-07 21:48:44 .00	host-3587	Stati Uniti-West1	6399	5,8 xl	62	18,1	81

time	host_name	posizione	measure_name	tipo_server	cpu_usage	memoria_disponibile	cpu_temp
2022-09-07 21:48:45000 000000	host-258743	eu-central	640	5,8 xl	88	9.4	91
2022-09-07 21:48:45 .00	host-356554	Stati Uniti - East2	2093	5,8 xl	29	24	54
R2022-09-07 21:48:45 .00	ospite-254	Stati Uniti-West1	7051	5,8 xl	44	32	48

2. Per l'ingestione in tempo reale: è necessario generare i dati in `measure_name` volo man mano che arrivano i dati.

In entrambi i casi, ti consigliamo di testare l'algoritmo di generazione dell'hash su entrambe le estremità (ingestione e interrogazione) per assicurarti di ottenere gli stessi risultati.

Di seguito sono riportati alcuni esempi di codice su cui generare il valore hash. `host_name`

Example Python

```
>>> import xxhash
>>> from bitstring import BitArray
>>> b=xxhash.xxh64('HOST-ID-1235').digest()
>>> BitArray(b).int % 8192
### 3195
```

Example Go

```
package main
```

```
import (
    "bytes"
    "fmt"
    "github.com/cespare/xxhash"
)

func main() {
    buf := bytes.NewBufferString("HOST-ID-1235")
    x := xxhash.New()
    x.Write(buf.Bytes())
    // convert unsigned integer to signed integer before taking mod
    fmt.Printf("%f\n", abs(int64(x.Sum64())) % 8192)
}

func abs(x int64) int64 {
    if (x < 0) {
        return -x
    }
    return x
}
```

Example Java

```
import java.nio.ByteBuffer;

import net.jpountz.xxhash.XXHash64;

public class test {
    public static void main(String[] args) {
        XXHash64 hasher = net.jpountz.xxhash.XXHashFactory.fastestInstance().hash64();

        String host = "HOST-ID-1235";
        ByteBuffer buf = ByteBuffer.wrap(host.getBytes());

        Long result = Math.abs(hasher.hash(buf, 0L));
        Long partition = result % 8192;

        System.out.println(result);
        System.out.println(partition);
    }
}
```

Example dipendenza in Maven

```
        <dependency>
            <groupId>net.jpountz.lz4</groupId>
            <artifactId>lz4</artifactId>
            <version>1.3.0</version>
        </dependency>
```

Sicurezza

- Per un accesso continuo a Timestream for LiveAnalytics, assicurati che le chiavi di crittografia siano protette e non vengano revocate o rese inaccessibili.
- Monitora i registri di accesso da. API AWS CloudTrail Controlla e revoca qualsiasi modello di accesso anomalo da parte di utenti non autorizzati.
- Segui le linee guida aggiuntive descritte in. Best practice di sicurezza per Amazon Timestream for LiveAnalytics

Configurazione di Amazon Timestream per LiveAnalytics

Configura il periodo di conservazione dei dati per l'archivio di memoria e l'archivio magnetico in modo che soddisfi i requisiti di elaborazione, archiviazione, prestazioni delle query e costo dei dati.

- Imposta la conservazione dei dati dell'archivio di memoria in modo che soddisfi i requisiti dell'applicazione per l'elaborazione dei dati in arrivo tardivo. I dati in arrivo tardivo sono dati in arrivo con un timestamp precedente all'ora corrente. Viene emesso da risorse che raggruppano gli eventi per un periodo di tempo prima di inviare i dati a Timestream e anche da risorse con connettività intermittente LiveAnalytics, ad esempio un sensore IoT che è online a intermittenza.
- Se prevedi che i dati in arrivo tardivo arrivino occasionalmente con timestamp prima della conservazione nell'archivio di memoria, dovresti abilitare le scritture magnetiche sull'archivio per la tua tabella. Una volta impostata l' EnableMagneticStoreWrites impostazione di una tabella, la tabella accetterà dati con data e ora precedenti alla conservazione nell'archivio di memoria, ma entro il periodo di conservazione dell'archivio magnetico. MagneticStoreWritesProperties
- Considerate le caratteristiche delle query che intendete eseguire su Timestream, ad LiveAnalytics esempio i tipi di query, la frequenza, l'intervallo di tempo e i requisiti prestazionali. Questo perché l'archivio di memoria e l'archivio magnetico sono ottimizzati per diversi scenari. L'archivio di memoria è ottimizzato per point-in-time query veloci che elaborano piccole quantità di dati recenti

inviati a Timestream per. LiveAnalytics L'archivio magnetico è ottimizzato per query analitiche veloci che elaborano volumi di dati medio-grandi inviati a Timestream per. LiveAnalytics

- Il periodo di conservazione dei dati dovrebbe essere influenzato anche dai requisiti di costo del sistema.

  Ad esempio, prendiamo in considerazione uno scenario in cui la soglia di arrivo tardivo dei dati per l'applicazione è di 2 ore e le applicazioni inviano molte query che elaborano dati relativi a un giorno, a una settimana o a un mese. In tal caso, potresti voler configurare un periodo di conservazione più breve per l'archivio di memoria (2-3 ore) e consentire a più dati di fluire verso l'archivio magnetico, dato che l'archivio magnetico è ottimizzato per query analitiche veloci.

Comprendete l'impatto dell'aumento o della diminuzione del periodo di conservazione dei dati dell'archivio di memoria e dell'archivio magnetico di una tabella esistente.

- Quando si riduce il periodo di conservazione dell'archivio di memoria, i dati vengono spostati dall'archivio di memoria all'archivio magnetico e il trasferimento dei dati è permanente. Timestream for LiveAnalytics non recupera i dati dall'archivio magnetico per popolare l'archivio di memoria. Quando si riduce il periodo di conservazione dell'archivio magnetico, i dati vengono eliminati dal sistema e l'eliminazione dei dati è permanente.
- Quando si aumenta il periodo di conservazione dell'archivio di memoria o dell'archivio magnetico, la modifica ha effetto sui dati inviati a Timestream a LiveAnalytics partire da quel momento in poi. Timestream for LiveAnalytics non recupera i dati dall'archivio magnetico per popolare l'archivio di memoria. Ad esempio, se il periodo di conservazione dell'archivio di memoria è stato inizialmente impostato su 2 ore e poi aumentato a 24 ore, occorreranno 22 ore prima che l'archivio di memoria contenga 24 ore di dati.

Scrive

- Assicurarsi che il timestamp dei dati in entrata non sia precedente alla conservazione dei dati configurata per l'archivio di memoria e non oltre il periodo di inserimento futuro definito in. Quote L'invio di dati con un timestamp al di fuori di questi limiti comporterà il rifiuto dei dati da parte di Timestream, a meno che non abiliti le scritture con memorizzazione magnetica per la LiveAnalytics tabella. Se abiliti le scritture nell'archivio magnetico, assicurati che il timestamp per i dati in entrata non sia precedente alla conservazione dei dati configurata per l'archivio magnetico.
- Se prevedi che i dati arrivino in ritardo, attiva le scritture sull'archivio magnetico per la tua tabella. Ciò consentirà l'inserimento di dati con timestamp che non rientrano nel periodo di

conservazione dell'archivio di memoria ma che rientrano comunque nel periodo di conservazione dell'archivio magnetico. Puoi impostarlo aggiornando la `EnableMagneticStoreWrites` bandiera nella tua tabella. `MagneticStoreWritesProperties` Questa proprietà è falsa per impostazione predefinita. Si noti che le operazioni di scrittura nell'archivio magnetico non saranno immediatamente disponibili per l'interrogazione. Saranno disponibili entro 6 ore.

- Indirizza i carichi di lavoro ad alto throughput all'archivio di memoria assicurandoti che i timestamp dei dati acquisiti rientrino nei limiti di conservazione dell'archivio di memoria. Le scritture sull'archivio magnetico sono limitate a un numero massimo di partizioni di archiviazione magnetica attive che possono ricevere l'ingestione simultanea per un database. È possibile visualizzare questa metrica in. `ActiveMagneticStorePartitions` CloudWatch Per ridurre le partizioni di immagazzinamento magnetico attivo, cercate di ridurre il numero di serie e la durata del tempo di ingestione dei magazzini magnetici in contemporanea.
- Durante l'invio dei dati a Timestream for LiveAnalytics, raggruppa più record in un'unica richiesta per ottimizzare le prestazioni di acquisizione dei dati.
  - È utile raggruppare i record della stessa serie temporale e i record con lo stesso nome di misura.
  - Batch il maggior numero possibile di record in una singola richiesta, purché le richieste rientrino nei limiti di servizio definiti inQuote.
  - Utilizza attributi comuni, ove possibile, per ridurre i costi di trasferimento e ingestione dei dati. Per ulteriori informazioni, vedere. WriteRecords API
- Se riscontri errori parziali sul lato client durante la scrittura dei dati su Timestream for LiveAnalytics, puoi inviare nuovamente il batch di record che non è riuscito a inserire dopo aver risolto la causa del rifiuto.
- I dati ordinati per timestamp offrono prestazioni di scrittura migliori.
- Amazon Timestream LiveAnalytics for è progettato per adattarsi automaticamente alle esigenze della tua applicazione. Quando Timestream for LiveAnalytics Notices aumenta il numero di richieste di scrittura provenienti dall'applicazione, l'applicazione potrebbe subire un certo livello di limitazione iniziale dell'archiviazione di memoria. Se la tua applicazione riscontra una limitazione dell'archiviazione di memoria, continua a inviare dati a Timestream alla stessa velocità (o maggiore) per consentire LiveAnalytics a Timestream di scalare automaticamente per soddisfare le esigenze dell'applicazione. LiveAnalytics Se notate una limitazione delle riserve magnetiche, dovreste ridurre la velocità di ingestione delle riserve magnetiche fino a ridurne il numero. `ActiveMagneticStorePartitions`

Caricamento in batch

Le migliori pratiche per il caricamento in batch sono descritte inProcedure ottimali per il caricamento in batch.

Query

Di seguito sono riportate le best practice suggerite per le query con Amazon Timestream for. LiveAnalytics

- Includi solo i nomi delle misure e delle dimensioni essenziali per l'interrogazione. L'aggiunta di colonne estranee aumenterà le scansioni dei dati, con un impatto sulle prestazioni delle query.
- Prima di distribuire la query in produzione, si consiglia di esaminare le informazioni sulle query per assicurarsi che la riduzione spaziale e temporale sia ottimale. Per ulteriori informazioni, consulta Utilizzo di informazioni dettagliate sulle query per ottimizzare le query in Amazon Timestream.
- Ove possibile, trasferite il calcolo dei dati su Timestream per LiveAnalytics utilizzare gli aggregati e le funzioni scalari integrate nella clausola e nella SELECT clausola, a seconda dei casi, per migliorare le prestazioni delle query e WHERE ridurre i costi. Consulta SELECT e Funzioni di aggregazione.
- Ove possibile, utilizza funzioni approssimative. Ad esempio, utilizzate APPROX _ DISTINCT invece di COUNT (DISTINCTcolumn_name) per ottimizzare le prestazioni delle query e ridurre il costo delle query. Per informazioni, consulta Funzioni di aggregazione.
- Utilizzate un'CASEespressione per eseguire aggregazioni complesse invece di selezionare più volte dalla stessa tabella. Per informazioni, consulta La dichiarazione CASE.
- Se possibile, includi un intervallo di tempo nella WHERE clausola della query. Ciò ottimizza le prestazioni e i costi delle query. Ad esempio, se hai bisogno solo dell'ultima ora di dati nel tuo set di dati, includi un predicato temporale come time > ago (1h). Consulta SELECT e Intervallo e durata.
- Quando una query accede a un sottoinsieme di misure in una tabella, includi sempre i nomi delle misure nella clausola della WHERE query.
- Ove possibile, utilizzate l'operatore di uguaglianza per confrontare dimensioni e misure nella WHERE clausola di una query. Un predicato di uguaglianza sui nomi delle dimensioni e delle misure consente di migliorare le prestazioni delle query e ridurre i costi delle query.
- Ove possibile, evitate di utilizzare le funzioni della WHERE clausola per ottimizzare i costi.
- Evita di utilizzare la LIKE clausola più volte. Utilizzate piuttosto le espressioni regolari quando filtrate più valori su una colonna di stringhe. Per informazioni, consulta Funzioni di espressioni regolari.

- Utilizzate solo le colonne necessarie nella clausola GROUP BY di una query.
- Se il risultato della query deve essere in un ordine specifico, specificate esplicitamente tale ordine nella clausola ORDER BY della query più esterna. Se il risultato della query non richiede l'ordinamento, evita di utilizzare una clausola ORDER BY per migliorare le prestazioni delle query.
- Utilizzate una LIMIT clausola se avete bisogno solo delle prime N righe della query.
- Se utilizzi una clausola ORDER BY per esaminare i valori N superiori o inferiori, utilizza una LIMIT clausola per ridurre i costi delle query.
- Utilizzate il token di impaginazione della risposta restituita per recuperare i risultati della query. Per ulteriori informazioni, consulta la sezione Query.
- Se hai iniziato a eseguire una query e ti rendi conto che la query non restituirà i risultati che stai cercando, annulla la query per risparmiare sui costi. Per ulteriori informazioni, consulta CancelQuery.
- Se la tua applicazione presenta problemi di throttling, continua a inviare dati ad Amazon Timestream alla stessa velocità LiveAnalytics per consentire ad Amazon Timestream di scalare automaticamente LiveAnalytics per soddisfare le esigenze di throughput delle query della tua applicazione.
- Se i requisiti di concorrenza delle query delle tue applicazioni superano i limiti predefiniti di Timestream for, contatta per eventuali aumenti dei limiti. LiveAnalytics AWS Support

Interrogazioni pianificate

Le interrogazioni pianificate consentono di ottimizzare i dashboard precalcolando alcune statistiche aggregate relative all'intero parco macchine. Quindi una domanda naturale da porsi è come prendere in considerazione il caso d'uso e identificare quali risultati precalcolare e come utilizzare questi risultati archiviati in una tabella derivata per creare la dashboard. La prima fase di questo processo consiste nell'identificare i pannelli da precalcolare. Di seguito sono riportate alcune linee guida di alto livello:

- Considera i byte scansionati dalle query utilizzate per popolare i pannelli, la frequenza di ricarica del dashboard e il numero di utenti simultanei che caricherebbero questi dashboard. È necessario iniziare con i dashboard caricati più frequentemente e scansionare quantità significative di dati. Le prime due dashboard nell'esempio di dashboard aggregata e la dashboard aggregata nell'esempio drill down sono buoni esempi di tali dashboard.
- Considerate quali calcoli vengono utilizzati ripetutamente. Sebbene sia possibile creare un'interrogazione pianificata per ogni pannello e ogni valore variabile utilizzato nel pannello, è

possibile ottimizzare in modo significativo i costi e il numero di query pianificate cercando soluzioni per utilizzare un unico calcolo per precalcolare i dati necessari per più pannelli.

- Considerate la frequenza delle interrogazioni pianificate per aggiornare i risultati materializzati nella tabella derivata. È consigliabile analizzare la frequenza di aggiornamento di una dashboard rispetto alla finestra temporale richiesta in una dashboard rispetto al time binning utilizzato nel precalcolo e ai pannelli nei dashboard. Ad esempio, se una dashboard che traccia gli aggregati orari degli ultimi giorni viene aggiornata solo una volta ogni poche ore, potresti voler configurare le query pianificate in modo che si aggiornino solo una volta ogni 30 minuti o un'ora. D'altra parte, se disponi di una dashboard che traccia gli aggregati al minuto e viene aggiornata ogni minuto circa, vorrai che le tue query pianificate aggiornino i risultati ogni minuto o pochi minuti.
- Considerate quali modelli di query possono essere ulteriormente ottimizzati (sia dal punto di vista del costo che della latenza delle query) utilizzando le query pianificate. Ad esempio, quando si calcolano i valori di dimensione univoci utilizzati spesso come variabili nei dashboard, o si restituisce l'ultimo punto dati emesso da un sensore o il primo punto dati emesso da un sensore dopo una certa data, ecc. Alcuni di questi modelli di esempio sono discussi in questa guida.

Le considerazioni precedenti avranno un impatto significativo sui vostri risparmi quando sposterete la dashboard per interrogare le tabelle derivate, sulla freschezza dei dati nelle dashboard e sui costi sostenuti dalle query pianificate.

Applicazioni client e integrazioni supportate

Esegui l'applicazione client dalla stessa regione di Timestream per ridurre le latenze di rete e LiveAnalytics i costi di trasferimento dei dati. Per ulteriori informazioni sull'utilizzo di altri servizi, consulta. Utilizzo di altri servizi Di seguito sono riportati alcuni altri link utili.

- Migliori pratiche per AWS lo sviluppo con AWS SDK for Java
- Procedure consigliate per l'utilizzo AWS Lambda delle funzioni
- Best practice per Amazon Managed Service per Apache Flink
- Le migliori pratiche per la creazione di dashboard in Grafana

Generali

- Assicurati di seguire The AWS Well-Architected Framework quando usi Timestream per. LiveAnalytics Questo white paper fornisce indicazioni sulle migliori pratiche in materia di eccellenza operativa, sicurezza, affidabilità, efficienza delle prestazioni e ottimizzazione dei costi.

Misurazione e ottimizzazione dei costi

Con Amazon Timestream LiveAnalytics for, paghi solo per ciò che usi. Timestream per LiveAnalytics contatori separati per scritture, dati archiviati e dati scansionati tramite query. Il prezzo di ciascuna dimensione di misurazione è specificato nella pagina dei prezzi. Puoi stimare la tua fattura mensile utilizzando il calcolatore dei prezzi di Amazon Timestream LiveAnalytics for Pricing.

Questa sezione descrive come funziona la misurazione per le scritture, l'archiviazione e le query in Timestream for. LiveAnalytics Vengono inoltre forniti scenari e calcoli di esempio. Inoltre, è incluso un elenco di best practice per l'ottimizzazione dei costi. Puoi selezionare un argomento qui sotto:

Argomenti

Scrive

La dimensione di scrittura di ogni evento della serie temporale viene calcolata come somma della dimensione del timestamp e di uno o più nomi di dimensione, valori di dimensione, nomi di misure e valori di misura. La dimensione del timestamp è di 8 byte. Le dimensioni dei nomi delle dimensioni, dei valori delle dimensioni e dei nomi delle misure sono la lunghezza degli UTF -8 byte codificati della stringa che rappresenta ogni nome di dimensione, valore di dimensione e nome di misura. La dimensione del valore di misura dipende dal tipo di dati. È 1 byte per il tipo di dati booleano, 8 byte per bigint e double e la lunghezza degli UTF -8 byte codificati per le stringhe. Ogni scrittura viene conteggiata in unità di 1 KiB.

Di seguito vengono forniti due esempi di calcolo:

Argomenti

Calcolo della dimensione di scrittura di un evento di serie temporali

Consideriamo un evento di serie temporale che rappresenta l'CPUutilizzo di un'EC2istanza come illustrato di seguito:

Orario	Regione	az	vpc	Hostname (Nome host)	measure_name	measure_value::double
1602983435238563000	us-east-1	1d	vpc-1a2b3c4d	Host-24GJU	utilizzazione_cpu_	35,0

La dimensione di scrittura dell'evento della serie temporale può essere calcolata come:

- tempo = 8 byte
- prima dimensione = 15 byte (+) `region us-east-1`
- seconda dimensione = 4 byte (+) `az 1d`
- terza dimensione = 15 byte (+) `vpc vpc-1a2b3c4d`
- quarta dimensione = 18 byte (+) `hostname host-24Gju`
- nome della misura = 15 byte () `cpu_utilization`
- valore della misura = 8 byte

Dimensione di scrittura dell'evento della serie temporale = 83 byte

Calcolo del numero di scritture

Consideriamo ora 100 EC2 istanze, simili all'istanza descritta inCalcolo della dimensione di scrittura di un evento di serie temporali, che emettono metriche ogni 5 secondi. Il totale delle scritture mensili per le EC2 istanze varierà in base al numero di eventi di serie temporali esistenti per scrittura e all'eventuale utilizzo di attributi comuni durante il raggruppamento degli eventi delle serie temporali. Viene fornito un esempio di calcolo del totale delle scritture mensili per ciascuno dei seguenti scenari:

Argomenti

- Suddivisione in batch di eventi di serie temporali e utilizzo di attributi comuni in una scrittura

Un evento di serie temporale per scrittura

Se ogni scrittura contiene un solo evento di serie temporali, le scritture mensili totali vengono calcolate come segue:

- 100 eventi di serie temporali = 100 scritture ogni 5 secondi
- x 12 scritture al minuto = 1.200 scritture
- x 60 minuti/ora = 72.000 scritture
- x 24 ore/giorno = 1.728.000 scritture
- x 30 giorni/mese = 51.840.000 scritture

Totale delle scritture mensili = 51.840.000

Raggruppamento di eventi di serie temporali in una scrittura

Dato che ogni scrittura è misurata in unità di 1 KB, una scrittura può contenere un batch di 12 eventi di serie temporali (998 byte) e le scritture mensili totali vengono calcolate come segue:

- 100 eventi di serie temporali = 9 scritture (12 eventi di serie temporali per scrittura) ogni 5 secondi
- x 12 scritture al minuto = 108 scritture
- x 60 minuti/ora = 6.480 scritture
- x 24 ore/giorno = 155.520 scritture
- x 30 giorni/mese = 4.665.600 scritture

Totale delle scritture mensili = 4.665.600

Suddivisione in batch di eventi di serie temporali e utilizzo di attributi comuni in una scrittura

Se la regione, az, vpc e il nome della misura sono comuni tra 100 EC2 istanze, i valori comuni possono essere specificati una sola volta per scrittura e vengono definiti attributi comuni. In questo caso, la dimensione degli attributi comuni è di 52 byte e la dimensione degli eventi delle serie temporali è di 27 byte. Dato che ogni scrittura è misurata in unità di 1 KB, una scrittura può contenere 36 eventi di serie temporali e attributi comuni e le scritture mensili totali vengono calcolate come segue:

- 100 eventi di serie temporali = 3 scritture (36 eventi di serie temporali per scrittura) ogni 5 secondi
- x 12 scritture al minuto = 36 scritture
- x 60 minuti/ora = 2.160 scritture
- x 24 ore/giorno = 51.840 scritture
- x 30 giorni/mese = 1.555.200 scritture

Totale delle scritture mensili = 1.555.200

> **Note**
>
> A causa dell'utilizzo del batch, degli attributi comuni e dell'arrotondamento delle scritture a unità di 1 KB, la dimensione di archiviazione degli eventi della serie temporale potrebbe essere diversa dalla dimensione di scrittura.

Storage

La dimensione di archiviazione di ogni evento della serie temporale nell'archivio di memoria e nell'archivio magnetico viene calcolata come somma delle dimensioni del timestamp, dei nomi delle dimensioni, dei valori delle dimensioni, dei nomi delle misure e dei valori delle misure. La dimensione del timestamp è di 8 byte. Le dimensioni dei nomi delle dimensioni, dei valori delle dimensioni e dei nomi delle misure sono la lunghezza degli UTF -8 byte codificati di ogni stringa che rappresenta il nome della dimensione, il valore della dimensione e il nome della misura. La dimensione del valore di misura dipende dal tipo di dati. È pari a 1 byte per i tipi di dati booleani, 8 byte per bigint e double e la lunghezza degli UTF -8 byte codificati per le stringhe. Ogni misura viene archiviata come record separato in Amazon Timestream LiveAnalytics per, ad esempio se l'evento della serie temporale ha quattro misure, ci saranno quattro record per quell'evento di serie temporale in archiviazione.

Considerando l'esempio dell'evento della serie temporale che rappresenta l'CPUutilizzo di un'EC2istanza (vediCalcolo della dimensione di scrittura di un evento di serie temporali), la dimensione di archiviazione dell'evento della serie temporale viene calcolata come segue:

- tempo = 8 byte
- prima dimensione = 15 byte (+) `region us-east-1`
- seconda dimensione = 4 byte (+) `az 1d`
- terza dimensione = 15 byte (+) `vpc vpc-1a2b3c4d`

- quarta dimensione = 18 byte (+) `hostname` `host-24Gju`
- nome della misura = 15 byte () `cpu_utilization`
- valore della misura = 8 byte

Dimensione di memorizzazione dell'evento della serie temporale = 83 byte

> **Note**
>
> L'archivio di memoria viene misurato in GB/ora e l'archivio magnetico viene misurato in GB/mese.

Query

Le query vengono addebitate in base alla durata delle unità di calcolo Timestream (TCUs) utilizzate dall'applicazione in TCU ore, come specificato nella pagina dei prezzi di Amazon Timestream. Amazon Timestream LiveAnalytics per il motore di query elimina i dati irrilevanti durante l'elaborazione di una query. Le interrogazioni con proiezioni e predicati che includono intervalli di tempo, nomi di misure e/o nomi di dimensioni consentono al motore di elaborazione delle query di ridurre una quantità significativa di dati e aiutano a ridurre i costi delle query.

Ottimizzazione dei costi

Per ottimizzare i costi di scrittura, archiviazione e query, utilizza le seguenti best practice con Amazon Timestream per: LiveAnalytics

- Batch di più eventi di serie temporali per scrittura per ridurre il numero di richieste di scrittura.
- Prendi in considerazione l'utilizzo di record a più misure, che consentono di scrivere più misure di serie temporali in un'unica richiesta di scrittura e di archiviare i dati in modo più compatto. Ciò riduce il numero di richieste di scrittura, nonché i costi di archiviazione dei dati e i costi delle query.
- Utilizza gli attributi comuni con il batching per raggruppare più eventi di serie temporali per scrittura e ridurre ulteriormente il numero di richieste di scrittura.
- Imposta la conservazione dei dati dell'archivio di memoria in modo che soddisfi i requisiti dell'applicazione per l'elaborazione dei dati in arrivo in ritardo. I dati in arrivo tardivo sono dati in entrata con un timestamp precedente all'ora corrente e al di fuori del periodo di conservazione dell'archivio di memoria.

- Imposta la conservazione dei dati dell'archivio magnetico in modo che soddisfi i tuoi requisiti di archiviazione dei dati a lungo termine.
- Durante la scrittura delle query, includi solo i nomi delle misure e delle dimensioni essenziali per l'interrogazione. L'aggiunta di colonne estranee aumenterà le scansioni dei dati e quindi aumenterà anche il costo delle query. Ti consigliamo di esaminare gli approfondimenti delle query per valutare l'efficienza di potatura delle dimensioni e delle misure incluse.
- Ove possibile, includi un intervallo di tempo nella WHERE clausola della richiesta. Ad esempio, se hai bisogno solo dell'ultima ora di dati nel tuo set di dati, includi un predicato temporale come. `time > ago(1h)`
- Quando una query accede a un sottoinsieme di misure in una tabella, includi sempre i nomi delle misure nella WHERE clausola della query.
- Se hai iniziato a eseguire una query e ti rendi conto che la query non restituirà i risultati che stai cercando, annulla la query per risparmiare sui costi.

Monitoraggio con Amazon CloudWatch

Puoi monitorare Timestream per utilizzare LiveAnalytics Amazon CloudWatch, che raccoglie ed elabora i dati grezzi da Timestream per LiveAnalytics trasformarli in metriche leggibili. near-real-time Registra queste statistiche per due settimane in modo da poter accedere alle informazioni storiche e avere una prospettiva migliore sulle prestazioni della tua applicazione o del tuo servizio web. Per impostazione predefinita, Timestream per i dati LiveAnalytics metrici viene inviato automaticamente CloudWatch in periodi di 1 minuto o 15 minuti. Per ulteriori informazioni, consulta What Is Amazon CloudWatch? nella Amazon CloudWatch User Guide.

Argomenti

- Come posso usare Timestream per LiveAnalytics le metriche?
- Timestream per LiveAnalytics metriche e dimensioni
- Creazione di CloudWatch allarmi per monitorare Timestream LiveAnalytics

Come posso usare Timestream per LiveAnalytics le metriche?

Le metriche riportate da Timestream LiveAnalytics forniscono informazioni che puoi analizzare in diversi modi. L'elenco seguente mostra alcuni usi comuni dei parametri. Questi suggerimenti sono solo introduttivi e non costituiscono un elenco completo.

In che modo?	Parametri rilevanti
How can I determine if any system errors occurred?	È possibile monitorare SystemErrors per determinare se eventuali richieste hanno generato un codice di errore del server. In genere, questo parametro deve essere uguale a zero. In caso contrario, potresti voler indagare.
How can I monitor the amount of data in the memory store?	È possibile monitorare MemoryCumulativeBytesMetered durante il periodo di tempo specificato, per monitorare la quantità di dati archiviati nell'archivio di memoria in byte. Questa metrica viene emessa ogni ora ed è possibile tenere traccia dei byte archiviati in un account e della granularità del database. L'archiviazione di memoria viene misurata in GB all'ora (il costo di archiviazione di 1 GB di dati per un'ora). Quindi, moltiplicando il valore orario di MemoryCumulativeBytesMetered con i prezzi in GB all'ora nella tua regione, otterrai il costo orario sostenuto. Dimensioni: operazione (archiviazione), nome della metrica DatabaseName
How can I monitor the amount of data in the magnetic store?	È possibile monitorare MagneticCumulativeBytesMetered durante il periodo di tempo specificato, per monitorare la quantità di dati archiviati nell'archivio magnetico in byte. Questa metrica viene emessa ogni ora ed è possibile tenere traccia dei byte archiviati in un account e della granularità del database. L'archiviazione di memoria viene misurata in GB al mese (il costo di archiviazione di 1 GB di dati per un mese). Quindi, moltiplicando il valore orario MagneticCumulativeBytesMetered con i prezzi in GB al mese nella tua regione, otterrai il costo orario sostenuto. Ad esempio, se il valore di MagneticCumulativeBytesMetered è 107374182400 byte (100 GB), la tariffa oraria di 1 GB di dati nell'archivio magnetico = (0,03) (prezzo us-east-1)/(30,4*24). Moltiplicando questo valore per il valore in GB si otterrà ~0,004 $ per quell'ora. MagneticCumulativeBytesMetered

In che modo?	Parametri rilevanti
	Dimensioni: operazione (archiviazione), nome della metrica DatabaseName
How can I monitor the data scanned by queries?	È possibile monitorare `CumulativeBytesMetered` durante il periodo di tempo specificato, per monitorare i dati scansionati dalle query (in byte) inviate a Timestream for. LiveAnalytics Questa metrica viene emessa dopo l'esecuzione della query ed è possibile tenere traccia dei dati scansionati in base alla granularità dell'account e del database. Puoi calcolare il costo delle query per un determinato periodo moltiplicando il valore della metrica per il prezzo per GB scansionato nella tua regione. I byte scansionati dalle query pianificate vengono contabilizzati in questa metrica. Dimensioni: operazione (interrogazione), nome della metrica DatabaseName
How can I monitor the data scanned by scheduled queries?	È possibile monitorare `CumulativeBytesMetered` durante il periodo di tempo specificato, per monitorare i dati scansionati dalle query pianificate (in byte) eseguite da Timestream for. LiveAnalytics Questa metrica viene emessa dopo l'esecuzione della query ed è possibile tenere traccia dei dati scansionati in base alla granularità dell'account e del database. Puoi calcolare il costo delle query per un determinato periodo moltiplicando il valore della metrica per il prezzo per GB scansionato nella tua regione. **Note** Nella query vengono contabilizzati anche i byte misurati. `CumulativeBytesMetered` Dimensioni: operazione (TriggeredScheduledQuery), DatabaseName nome della metrica

In che modo?	Parametri rilevanti
How can I monitor the number of records ingested?	È possibile monitorare `NumberOfRecords` nel periodo di tempo specificato per monitorare il numero di record acquisiti. È possibile tenere traccia dei byte archiviati in un account e della granularità del database. È inoltre possibile utilizzare questa metrica per monitorare le scritture effettuate da Scheduled Queries quando i risultati delle query vengono scritti in una tabella separata. Quando si utilizza `WriteRecords` API, la metrica viene emessa per ogni `WriteRecords` richiesta, con la dimension e Operation impostata CloudWatch su. `WriteRecords` Quando si utilizza `BatchLoad` o `ScheduledQuery` APIs, la metrica viene emessa a intervalli determinati dal servizio fino al completamento dell'attività. La dimensione CloudWatch Operazione per questa metrica è `BatchLoad` o`Scheduled Query` , a seconda della dimensione utilizzata. API Dimensioni: operazione (WriteRecords, o ScheduledQuery) BatchLoad DatabaseName, nome della metrica

In che modo?	Parametri rilevanti
How can I monitor the cost of records ingested?	È possibile monitorare CumulativeBytesMetered per monitorare il numero di byte ingeriti che generano costi. È possibile tenere traccia dei byte archiviati in un account e della granularità del database. I record importati vengono misurati in byte cumulativi. Moltiplicando il valore dei prezzi di CumulativeBytesMetered By Writes nella tua regione, otterrai i costi di importazione sostenuti. Quando si utilizza WriteRecords API, questa metrica viene emessa per ogni WriteRecords richiesta, con la dimension e Operation impostata su. CloudWatch WriteRecords Quando si utilizza BatchLoad o ScheduledQuery API, la metrica viene emessa a intervalli determinati dal servizio fino al completamento dell'attività. La dimensione CloudWatch Operazione per questa metrica è BatchLoad o Scheduled Query dipende da quale viene utilizzata.. API Dimensioni: Operazione (WriteRecords, BatchLoad, or ScheduledQuery) DatabaseName, Nome della metrica
How can I monitor the Timestream Compute Units (TCUs) used in my account?	È possibile eseguire il monitoraggio nell'QueryTCUarco del periodo di tempo specificato, per monitorare le unità di calcolo utilizzate per il carico di lavoro delle query nell'account. Questa metrica viene emessa con unità di calcolo massime e minime per ogni minuto durante il carico di lavoro di interrogazione attivo dall'account. Unità: Count Statistiche valide: minimo, massimo Metrica: ResourceCount Dimensioni:Service: Timestream ,,Resource: QueryTCU, Type: Resource Class: OnDemand

Timestream per LiveAnalytics metriche e dimensioni

Quando interagisci con Timestream for LiveAnalytics, invia le seguenti metriche e dimensioni ad Amazon. CloudWatch Tutte le metriche vengono aggregate e riportate ogni minuto. Puoi utilizzare le seguenti procedure per visualizzare le metriche di Timestream for. LiveAnalytics

Per visualizzare le metriche utilizzando la console CloudWatch

I parametri vengono raggruppati prima in base allo spazio dei nomi del servizio e successivamente in base alle diverse combinazioni di dimensioni all'interno di ogni spazio dei nomi.

1. Apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Se necessario, modificare la regione . Nella barra di navigazione, scegli la regione in cui risiedono AWS le tue risorse. Per ulteriori informazioni, consulta AWS Endpoint del servizio.
3. Nel riquadro di navigazione, seleziona Parametri.
4. Nella scheda Tutte le metriche, scegli `AWS/Timestream for LiveAnalytics`.

Per visualizzare le metriche utilizzando il AWS CLI

- Al prompt dei comandi utilizza il comando seguente.

```
aws cloudwatch list-metrics --namespace "AWS/Timestream"
```

Dimensioni per Timestream per le metriche LiveAnalytics

Le metriche per Timestream for LiveAnalytics sono qualificate dai valori dell'account, del nome della tabella o dell'operazione. Puoi utilizzare la CloudWatch console per recuperare Timestream per LiveAnalytics i dati relativi a una qualsiasi delle dimensioni nella tabella seguente:

Dimensione	Descrizione
`DatabaseName`	Questa dimensione limita i dati a uno specifico Timestream per il database. LiveAnalytics Questo valore può essere qualsiasi database nella regione corrente e nell'account corrente AWS
`Operation`	Questa dimensione limita i dati a uno dei Timestream per LiveAnalytics operazioni come`Storage`, `WriteReco`

Dimensione	Descrizione
	`rds BatchLoad` , o. `ScheduledQuery` Vedi Timestream for LiveAnalytics Query API Reference per un elenco di valori disponibili.
`TableName`	Questa dimensione limita i dati a una tabella specifica in un database Timestream for. LiveAnalyticss

> **Important**
>
> `CumulativeBytesMeteredUserErrorse` le `SystemErrors` metriche hanno solo la dimensione. `Operation` `SuccessfulRequestLatencyle` metriche hanno sempre `Operation` una dimensione, ma possono anche avere le `TableName` dimensioni `DatabaseName` e, a seconda del valore di. `Operation` Questo perché Timestream per le operazioni a LiveAnalytics livello di tabella ha `DatabaseName` e `TableName` come dimensioni, ma le operazioni a livello di account no.

Timestream per le metriche LiveAnalytics

> **Note**
>
> Amazon CloudWatch aggrega tutti i seguenti valori Timestream per le LiveAnalytics metriche a intervalli di un minuto.

Metriche generali

Parametro	Descrizione
`SuccessfulRequestLatency`	Le richieste riuscite a Timestream per il LiveAnalytics periodo di tempo specificato. SuccessfulRequestLatency può fornire due diversi tipi di informazioni: • Il tempo trascorso per le richieste riuscite (minimo, massimo, somma o media).

Parametro	Descrizione
	• Il numero di richieste eseguite correttamente (SampleCount). SuccessfulRequestLatency riflette l'attività solo all'interno di Timestream LiveAnalytics e non tiene conto della latenza di rete o dell'attività sul lato client. Unità: `Milliseconds` Dimensioni • `DatabaseName` • `TableName` • `Operation` Statistiche valide: • `Minimum` • `Maximum` • `Average` • `SampleCount` • `P10` • `p50` • `p90` • `p95` • `p99`

Metriche di scrittura e archiviazione

Parametro	Descrizione
`MagneticStoreRejectedRecordCount`	Il numero di record scritti di archiviazione magnetica che sono stati rifiutati in modo asincrono. Ciò può accadere se il nuovo record ha una versione inferiore alla versione corrente o se il nuovo record ha una versione uguale alla versione corrente ma contiene dati diversi. Unità: `Count` Dimensioni • `DatabaseName` • `TableName` • `Operation` Statistiche valide: • `Sum` • `SampleCount`
`MagneticStoreRejectedUpload UserFailures`	Il numero di report relativi ai record respinti dall'archivio magnetico che non sono stati caricati a causa di errori dell'utente. Ciò può essere dovuto a IAM autorizzazioni non configurate correttamente o all'eliminazione di un bucket S3. Unità: `Count` Dimensioni • `DatabaseName` • `TableName` • `Operation`

Parametro	Descrizione
	Statistiche valide: • `Sum` • `SampleCount`
`MagneticStoreRejectedUploadSystemFailures`	Il numero di Magnetic Store ha rifiutato i report relativi ai record che non sono stati caricati a causa di errori di sistema. Unità: `Count` Dimensioni • `DatabaseName` • `TableName` • `Operation` Statistiche valide: • `Sum` • `SampleCount`

Parametro	Descrizione
`ActiveMagneticStorePartitions`	Il numero di partizioni di archiviazione magnetica che assorbono attivamente i dati in un determinato momento. Unità: `Count` Dimensioni • `DatabaseName` • `Operation` Statistiche valide: • `Sum` • `SampleCount`

Parametro	Descrizione
`MagneticStorePendingRecordsLatency`	La scrittura più vecchia su un archivio magnetico che non è disponibile per l'interro gazione. I record scritti nell'archivio magnetico saranno disponibili per l'interrogazione entro 6 ore. Unità: `Milliseconds` Dimensioni • `DatabaseName` • `TableName` • `Operation` Statistiche valide: • `Minimum` • `Maximum` • `Average` • `SampleCount` • `P10` • `p50` • `p90` • `p95` • `p99`

Parametro	Descrizione
`MemoryCumulativeBytesMetered`	La quantità di dati archiviati nell'archivio di memoria, in byte Unità: `Bytes` Dimensioni: `Operation` Statistiche valide: • `Average`
`MagneticCumulativeBytesMetered`	La quantità di dati archiviati nell'archivio magnetico, in byte Unità: `Bytes` Dimensioni: `Operation` Statistiche valide: • `Average`
`CumulativeBytesMetered`	La quantità di dati misurata dall'ingestione in Timestream for, in byte. LiveAnalytics Unità: `Bytes` Dimensioni: `Operation` Statistiche valide: `Sum`
`NumberOfRecords`	Il numero di record inseriti in Timestream per. LiveAnalytics Unità: `Count` Dimensioni: `Operation` Statistiche valide: `Sum`

Metriche di interrogazione

Parametro	Descrizione
`CumulativeBytesMetered`	La quantità di dati analizzati dalle query inviate a Timestream per, in byte. LiveAnalytics Unità: `Bytes` Dimensioni: `Operation` Statistiche valide: • `Sum`
`ResourceCount`	Le unità di calcolo Timestream (TCUs) utilizzate per il carico di lavoro delle query nell'account. Questa metrica viene emessa con unità di calcolo massime e minime per ogni minuto durante il carico di lavoro di interrogazione attivo dall'account. Unità: `Count` Statistiche valide: minimo, massimo Dimensioni:`Service: Timestream` ,`Resource: QueryTCU`,`Type: Resource`, `Class: OnDemand`

Metriche degli errori

Parametro	Descrizione
`SystemErrors`	Le richieste a Timestream LiveAnalytics generano un messaggio SystemError durante il periodo di tempo specificato. A di SystemError solito indica un errore interno del servizio. Unità: `Count`

Parametro	Descrizione
	Dimensioni: `Operation` Statistiche valide: • `Sum` • `SampleCount`
`UserErrors`	Le richieste a Timestream LiveAnalytics generano un InvalidRequest errore durante il periodo di tempo specificato. An indica in InvalidRequest genere un errore sul lato client, ad esempio una combinazione di parametri non valida, un tentativo di aggiornare una tabella inesistente o una firma di richiesta errata. UserErrors rappresenta l'aggregato di richieste non valide per la regione corrente e l'account corrente. AWS AWS Unità: `Count` Dimensioni: `Operation` Statistiche valide: • `Sum` • `SampleCount`

Important

Non tutte le statistiche, come `Average` o `Sum`, si applicano a tutti i parametri. Tuttavia, tutti questi valori sono disponibili tramite Timestream per LiveAnalytics console o utilizzando la CloudWatch console o per tutte le AWS CLI metriche. AWS SDKs

Creazione di CloudWatch allarmi per monitorare Timestream LiveAnalytics

Puoi creare un CloudWatch allarme Amazon per Timestream LiveAnalytics che invia un messaggio Amazon Simple Notification Service SNS (Amazon) quando l'allarme cambia stato. Un allarme monitora un singolo parametro per un periodo di tempo specificato. L'allarme esegue una o più operazioni basate sul valore del parametro relativo a una soglia prestabilita per un certo numero di periodi. L'azione è una notifica inviata a un SNS argomento di Amazon o a una politica di Auto Scaling.

Gli allarmi richiamano azioni solo per modifiche di stato sostenute. CloudWatch gli allarmi non richiamano azioni semplicemente perché si trovano in uno stato particolare. Lo stato deve essere cambiato e restare costante per un numero specificato di periodi.

Per ulteriori informazioni sulla creazione di CloudWatch allarmi, consulta Using Amazon CloudWatch Alarms nella Amazon CloudWatch User Guide.

Risoluzione dei problemi

Questa sezione contiene informazioni sulla risoluzione dei problemi relativi a Timestream per. LiveAnalytics

Argomenti

- Gestione delle valvole a farfalla WriteRecords
- Gestione dei record rifiutati
- Risoluzione dei problemi UNLOAD da Timestream per LiveAnalytics
- Timestream per codici di errore LiveAnalytics specifici

Gestione delle valvole a farfalla WriteRecords

Le richieste di scrittura dell'archivio di memoria su Timestream possono essere limitate man mano che Timestream si adatta alle esigenze di acquisizione dei dati dell'applicazione. Se le applicazioni presentano eccezioni di limitazione, è necessario continuare a inviare dati con la stessa velocità di trasmissione (o superiore) per consentire a Timestream di scalare automaticamente in base alle esigenze dell'applicazione.

Le richieste di scrittura dell'archivio magnetico su Timestream potrebbero essere limitate se il limite massimo di partizioni di archiviazione magnetica che ricevono l'ingestione. Vedrai un messaggio di accelerazione che ti invita a controllare la metrica di Cloudwatch per questo database.

`ActiveMagneticStorePartitions` Questa accelerazione può richiedere fino a 6 ore per risolversi. Per evitare questo rallentamento, è consigliabile utilizzare l'archivio di memoria per qualsiasi carico di lavoro di ingestione ad alta velocità. Per quanto riguarda l'ingestione di memorie magnetiche, è possibile effettuare l'acquisizione in un minor numero di partizioni limitando il numero di serie e la durata dell'acquisizione

Per ulteriori informazioni sulle migliori pratiche di inserimento dei dati, vedere. Scrive

Gestione dei record rifiutati

Se Timestream rifiuta i record, riceverai un messaggio `RejectedRecordsException` con i dettagli sul rifiuto. Per ulteriori informazioni su come estrarre queste informazioni dalla risposta, consulta Gestione degli errori di scrittura. WriteRecords

Tutti i rifiuti verranno inclusi in questa risposta ad eccezione degli aggiornamenti dell'archivio magnetico in cui la versione del nuovo record è inferiore o uguale alla versione del record esistente. In questo caso, Timestream non aggiornerà il record esistente che contiene la versione superiore. Timestream rifiuterà il nuovo record con una versione inferiore o uguale e scriverà questi errori in modo asincrono nel bucket S3. Per ricevere queste segnalazioni di errori asincrone, è necessario impostare la proprietà nella tabella. `MagneticStoreRejectedDataLocation` `MagneticStoreWriteProperties`

Risoluzione dei problemi UNLOAD da Timestream per LiveAnalytics

Di seguito sono riportate le indicazioni per la risoluzione dei problemi relativi al UNLOAD comando.

Categoria	Messaggio di errore	Come risolvere i problemi
Lunghezza della chiave S3	UNLOADla chiave del file dei risultati quando si utilizza il prefisso S3 [%s] fornito nella destinazione supererà la lunghezza della chiave S3 consentita. Consulta la documentazione per maggiori dettagli.	Quando si esportano i risultati della query utilizzan do l'UNLOADistruzione, la lunghezza della chiave S3, che comprende la somma della lunghezza del nome e del prefisso del bucket S3, supera la lunghezza massima della chiave S3 supportata. Ti consigliamo di ridurre la

Categoria	Messaggio di errore	Come risolvere i problemi
		lunghezza del prefisso o del nome del bucket.
	UNLOADla chiave del file dei risultati quando si usa partitioned_by [%s] supererà la lunghezza della chiave consentita da S3. Consulta la documentazione per maggiori dettagli.	Quando si esportano i risultati della query utilizzando l'UNLOADistruzione, la lunghezza della chiave S3 utilizzando la colonna partitioned_by supera la lunghezza massima della chiave S3 supportata. Si consiglia di eseguire il partizionamento con una colonna alternativa o di ridurre la lunghezza della colonna partitioned_column (se possibile).
	UNLOADla chiave del file dei risultati quando si utilizza il prefisso S3 [%s] insieme a partitioned_by [%s] supererà la lunghezza della chiave S3 consentita. Consulta la documentazione per maggiori dettagli.	Quando si esportano i risultati della query utilizzando l'UNLOADistruzione, la lunghezza della chiave S3, che comprende la somma della lunghezza del nome del bucket S3, del prefisso e del nome della colonna partitioned_by, supera la lunghezza massima della chiave S3 supportata. Ti consigliamo di ridurre la lunghezza del prefisso e del nome del bucket o di utilizzare una colonna alternativa per partizionare i dati.

Categoria	Messaggio di errore	Come risolvere i problemi
	La chiave dell'oggetto S3 generata: %s è troppo lunga. Consulta la documentazione per maggiori dettagli.	Durante l'elaborazione della query utilizzando l'UNLOADistruzione, uno dei valori nella colonna partizion ata supera la lunghezza massima supportata della chiave S3. La colonna e il valore della partizione si trovano nella chiave dell'ogge tto generata.
Acceleratori S3	Abbiamo rilevato che Amazon S3 limita le scritture dal comando. UNLOAD Per ulteriori informazioni, consulta la documentazione di Amazon Timestream	Consulta la documentazione di S3 qui. La velocità di API chiamata di S3 potrebbe essere ridotta quando più lettori/scrittori accedono alla stessa cartella. Controlla il volume delle chiamate in base al bucket fornito. Se utilizzi lo stesso bucket per più UNLOAD query simultanee, prova a utilizzare bucket diversi per lo stesso. Se utilizzi lo stesso bucket per più operazioni diverse da Timestream for LiveAnalytics UNLOAD, valuta la possibilità di spostare i risultati in un bucket separato. UNLOAD

Timestream per codici di errore LiveAnalytics specifici

Questa sezione contiene i codici di errore specifici per Timestream for. LiveAnalytics

Timestream per errori di scrittura LiveAnalytics API

InternalServerException

HTTPCodice di stato: 500

ThrottlingException

HTTPCodice di stato: 429

ValidationException

HTTPCodice di stato: 400

ConflictException

HTTPCodice di stato: 409

AccessDeniedException

Non disponi dell'autorizzazione di accesso sufficiente per eseguire questa operazione.

HTTPCodice di stato: 403

ServiceQuotaExceededException

HTTPCodice di stato: 402

ResourceNotFoundException

HTTPCodice di stato: 404

RejectedRecordsException

HTTPCodice di stato: 419

InvalidEndpointException

HTTPCodice di stato: 421

Timestream per gli errori di interrogazione LiveAnalytics API

ValidationException

HTTPCodice di stato: 400

QueryExecutionException

HTTPCodice di stato: 400

ConflictException

HTTPCodice di stato: 409

ThrottlingException

HTTPCodice di stato: 429

InternalServerException

HTTPCodice di stato: 500

InvalidEndpointException

HTTPCodice di stato: 421

Quote

Questo argomento descrive le quote correnti, note anche come limiti, all'interno di Amazon Timestream for. LiveAnalytics Salvo dove diversamente specificato, ogni quota si applica a una Regione specifica.

Argomenti

Quote di default

La tabella seguente contiene il Timestream per le LiveAnalytics quote e i valori predefiniti.

displayName	Descrizione	defaultValue
Database per account	Il numero massimo di database che è possibile creare per. Account AWS	500
Tabelle per account	Il numero massimo di tabelle che è possibile creare per Account AWS.	50000
Frequenza di accelerazione per CRUD APIs	Il numero massimo di API richieste di Create/Update/List /Describe/Delete database/ table/scheduled interroga zione consentite al secondo per account, nella regione corrente.	1
Query pianificate per account	Il numero massimo di query pianificate che è possibile creare per. Account AWS	10000
Numero massimo di partizion i di archiviazione magnetiche attive	Il numero massimo di partizion i di archiviazione magnetica attive per database. Una partizione può rimanere attiva fino a sei ore dopo la ricezione dell'ingestione.	250
maxQueryTCU	Il numero massimo di interrogazioni TCUs che puoi impostare per il tuo account.	1000

Limiti del servizio

La tabella seguente contiene il Timestream per i limiti del LiveAnalytics servizio e i valori predefiniti. Per modificare la conservazione dei dati per una tabella dalla console, vedi Modificare una tabella.

displayName	Descrizione	defaultValue
Periodo di importazione futuro in minuti	Il tempo di esecuzione massimo (in minuti) per i dati di serie temporali rispetto all'ora di sistema corrente. Ad esempio, se il periodo di inserimento futuro è di 15 minuti, Timestream for LiveAnalytics accetterà dati anticipati fino a 15 minuti rispetto all'ora corrente del sistema.	15
Periodo minimo di conservazione per l'archiviazione in memoria in ore	La durata minima (in giorni) per la quale i dati possono essere conservati nell'archivio della memoria per tabella.	1
Periodo massimo di conservazione per l'archivio della memoria in ore	La durata massima (in giorni) per la quale i dati possono essere mantenuti nell'archivio della memoria per tabella.	8766
Periodo minimo di conservazione per l'archiviazione magnetica in giorni	La durata minima (in giorni) per la quale i dati devono essere conservati nell'archivio magnetico per tabella.	1
Periodo massimo di conservazione per l'archiviazione magnetica in giorni	La durata massima (in giorni) per la quale i dati possono essere mantenuti nell'archivio magnetico. Questo valore equivale a 200 anni.	73000
Periodo di conservazione predefinito per l'archivio magnetico (in giorni)	Il valore predefinito (in giorni) per il quale i dati vengono conservati nell'archivio	73000

displayName	Descrizione	defaultValue
	magnetico per tabella. Questo valore è equivalente a 200 anni.	
Periodo di conservazione predefinito per l'archiviazione in memoria, in ore	La durata predefinita (in ore) per la quale i dati vengono conservati nell'archivio di memoria.	6
Dimensioni per tabella	Il numero massimo di dimensioni per tabella.	128
Nomi delle misure per tabella	Il numero massimo di nomi di misure univoci per tabella.	8192
Dimensione coppia nome-valore dimensione per serie	La dimensione massima della coppia nome-valore dimensione per serie.	2 KB
Dimensione massima dei record	La dimensione massima di un record.	2 KB
Record per WriteRecords API richiesta	Il numero massimo di record in una WriteRecords API richiesta.	100
Lunghezza del nome della dimensione	Il numero massimo di byte per un nome di dimensione.	60 byte
Lunghezza del nome della misura	Il numero massimo di byte per un nome di misura.	256 byte
Lunghezza del nome del database	Il numero massimo di byte per un nome di database.	256 byte
Lunghezza del nome della tabella	Il numero massimo di byte per un nome di tabella.	256 byte

displayName	Descrizione	defaultValue
QueryString lunghezza in KB	La lunghezza massima (in KB) di una stringa di query in UTF -8 caratteri codificati per una query.	256
Durata dell'esecuzione delle query in ore	La durata massima di esecuzione (in ore) per una query. Le query che richiedon o più tempo scadranno.	1
Informazioni sulle interroga zioni	Il numero massimo di API richieste Query consentite con Query Insights abilitate al secondo per account, nella regione corrente.	1
Dimensione dei metadati per il risultato della query	La dimensione massima dei metadati per il risultato di una query.	100 KB
Dimensione dei dati per il risultato della query	La dimensione massima dei dati per il risultato di una query.	5 Gigabyte
Misure per record di più misure	Il numero massimo di misure per record di più misure.	256
Dimensione del valore della misura per record di più misure	La dimensione massima dei valori delle misure per record di più misure.	2048
Misure univoche su record di più misure per tabella	Le misure univoche in tutti i record di più misure definiti in una singola tabella.	1.024
Unità di calcolo Timestream () TCUs per account	Il numero massimo predefinito per accountTCUs.	200

Tipi di dati supportati

La tabella seguente descrive i tipi di dati supportati per i valori di misura e dimensione.

Descrizione	Timestream per il valore LiveAnalytics
Tipi di dati supportati per i valori di misura.	Big int, double, string, boolean, MULTI Timestamp
Tipi di dati supportati per i valori delle dimensioni.	Stringa

Caricamento in batch

Le quote attuali, note anche come limiti, all'interno del carico in batch sono le seguenti.

Descrizione	Timestream per quanto riguarda il valore LiveAnalytics
Dimensione massima dell'attività di caricamento in batch	La dimensione massima dell'attività di caricamento in batch non può superare i 100 GB.
Quantità di file	Un'operazione di caricamento in batch non può contenere più di 100 file.
Dimensione massima dei file	La dimensione massima dei file in un'operazione di caricamento in batch non può superare i 5 GB.
CSVdimensione delle righe del file	Una riga in un CSV file non può superare i 16 MB. Si tratta di un limite rigido che non può essere aumentato.
Attività di caricamento in batch attive	Una tabella non può avere più di 5 attività di caricamento batch attive e un account non può avere più di 10 attività di caricamento batch attive. Timestream for LiveAnalytics limiterà le nuove attività di caricamento in batch fino a quando non saranno disponibili più risorse.

Vincoli per la denominazione

La tabella seguente descrive i vincoli di denominazione.

Descrizione	Timestream per valore LiveAnalytics
La lunghezza massima del nome di una dimensione.	60 byte
La lunghezza massima del nome di una misura.	256 byte
La lunghezza massima del nome di una tabella o del nome di database.	256 byte
Nome della tabella e del database	• Si consiglia di non utilizzareIdentificatori di sistema. • Può contenere a-z A-Z 0-9 _ (trattino basso) - (trattino). (punto). • Tutti i nomi devono essere codificati come UTF -8 e fanno distinzione tra maiuscole e minuscole. **Note** I nomi delle tabelle e dei database vengono confrontati utilizzando la rappresentazione binaria UTF -8. Ciò significa che il confronto tra ASCII caratteri fa distinzione tra maiuscole e minuscole.
Nome della misura	• Non deve contenere Identificatori di sistema i due punti ':'. • Non deve iniziare con un prefisso riservato (`ts_`,`measure_value`).

Descrizione	Timestream per valore LiveAnalytics
	Note I nomi delle tabelle e dei database vengono confronta ti utilizzando una rappresentazione binaria UTF -8. Ciò significa che il confronto tra ASCII caratteri fa distinzione tra maiuscole e minuscole.
Nome della dimensione	• Non deve contenere Identificatori di sistema i due punti ':' o le virgolette doppie («). • Non deve iniziare con un prefisso riservato (`ts_`,`measure_value`). • Non deve contenere caratteri Unicode [0,31] elencati qui o «\ u2028" o «\ u2029". **Note** I nomi delle dimensioni e delle misure vengono confronta ti utilizzando una rappresentazione binaria -8. UTF Ciò significa che il confronto tra ASCII caratteri fa distinzione tra maiuscole e minuscole.
Tutti i nomi delle colonne	I nomi delle colonne non possono essere duplicati. Poiché i record composti da più misure rappresentano dimensioni e misure come colonne, il nome di una dimensione non può essere uguale al nome di una misura. I nomi rispettano la distinzione tra lettere maiuscole e minuscole.

Parole chiave riservate

Tutte le seguenti sono parole chiave riservate:

- ALTER
- AND

- AS
- BETWEEN
- BY
- CASE
- CAST
- CONSTRAINT
- CREATE
- CROSS
- CUBE
- CURRENT_DATE
- CURRENT_TIME
- CURRENT_TIMESTAMP
- CURRENT_USER
- DEALLOCATE
- DELETE
- DESCRIBE
- DISTINCT
- DROP
- ELSE
- END
- ESCAPE
- EXCEPT
- EXECUTE
- EXISTS
- EXTRACT
- FALSE
- FOR
- FROM
- FULL

- GROUP
- GROUPING
- HAVING
- IN
- INNER
- INSERT
- INTERSECT
- INTO
- IS
- JOIN
- LEFT
- LIKE
- LOCALTIME
- LOCALTIMESTAMP
- NATURAL
- NORMALIZE
- NOT
- NULL
- ATTIVATO
- O
- ORDER
- OUTER
- PREPARE
- RECURSIVE
- RIGHT
- ROLLUP
- SELECT
- TABLE
- THEN

- TRUE
- UESCAPE
- UNION
- UNNEST
- USING
- VALUES
- WHEN
- WHERE
- WITH

Identificatori di sistema

Riserviamo che i nomi delle colonne «measure_value», «ts_non_existent_col» e «time» siano Timestream per gli identificatori di sistema. LiveAnalytics Inoltre, i nomi delle colonne non possono iniziare con «ts_» o «measure_name». Gli identificatori di sistema distinguono tra maiuscole e minuscole. Identificatori confrontati utilizzando una rappresentazione binaria UTF -8. Ciò significa che il confronto per gli identificatori fa distinzione tra maiuscole e minuscole.

> **Note**
>
> Gli identificatori di sistema non possono essere utilizzati per i nomi delle dimensioni o delle misure. Si consiglia di non utilizzare identificatori di sistema per i nomi di database o tabelle.

UNLOAD

Per i limiti relativi al `UNLOAD` comando, vedi Utilizzo UNLOAD per esportare i risultati delle query in S3 da Timestream.

Riferimento al linguaggio di interrogazione

> **Note**
>
> Questo riferimento al linguaggio di interrogazione include la seguente documentazione di terze parti della Trino Software Foundation (precedentemente Presto Software Foundation),

che è concessa in licenza con la licenza Apache, versione 2.0. Non è possibile utilizzare questo file se non in conformità con questa licenza. Per ottenere una copia della licenza Apache, versione 2.0, visitate il sito web di Apache.

Timestream for LiveAnalytics supporta un linguaggio di interrogazione avanzato per lavorare con i dati. Di seguito puoi vedere i tipi di dati, gli operatori, le funzioni e i costrutti disponibili.

Puoi anche iniziare subito con il linguaggio di interrogazione di Timestream nella sezione. Query di esempio

Argomenti

- Funzioni finestra
- Query di esempio

Tipi di dati supportati

Il linguaggio di interrogazione LiveAnalytics di Timestream for supporta i seguenti tipi di dati.

> **Note**
>
> I tipi di dati supportati per le scritture sono descritti in Tipi di dati.

Tipo di dati	Descrizione
`int`	Rappresenta un numero intero a 32 bit.
`bigint`	Rappresenta un numero intero con segno a 64 bit.
`boolean`	Uno dei due valori di verità della logica, `True` e`False`.
`double`	Rappresenta un tipo di dati a precisione variabile a 64 bit. Implementa IEEElo standard 754 per l'aritmetica binaria a virgola mobile. **Note** Il linguaggio di interrogazione serve per leggere i dati. Esistono funzioni `Infinity` e valori `NaN` doppi che possono essere utilizzati nelle query. Ma non puoi scrivere quei valori su Timestream.
`varchar`	Dati di caratteri a lunghezza variabile con una dimensione massima di 2 KB.
`array[`*T*`,...]`	Contiene uno o più elementi di un tipo di dati specificato *T*dove: *T* può essere uno qualsiasi dei tipi di dati supportati in Timestream.

Tipo di dati	Descrizione
`row(T,...)`	Contiene uno o più campi denominati del tipo di dati *T*. I campi possono essere di qualsiasi tipo di dati supportato da Timestream e sono accessibili con l'operatore di riferimento del campo a punti: `.`
`date`	Rappresenta una data nel modulo*YYYY*-*MM*-*DD*. dove *YYYY* è l'anno, *MM* è il mese, e *DD* è il giorno, rispettivamente. L'intervallo supportato va da `1970-01-01` a`2262-04-11` . Esempio: `1971-02-03`
`time`	Rappresenta l'ora del giorno in UTC. Il `time` tipo di dati è rappresentato nel modulo *HH*.*MM*.*SS*.*sssssssss* . Supporta la precisione in nanosecondi. Esempio: `17:02:07.496000000`
`timestamp`	Rappresenta un'istanza nel tempo utilizzando il tempo di precisione in nanosecondi. UTC *YYYY*-*MM*-*DD hh*:*mm*:*ss*.*sssssssss* La query supporta timestamp compresi tra. `1677-09-21 00:12:44.000000000` `2262-04-11 23:47:16.854775807`

Tipo di dati	Descrizione

`interval`	Rappresenta un intervallo di tempo come stringa letterale*X*`t`, composta da due parti, *X* e *t*.
	
	X è un valore numerico maggiore o uguale a, e `0` *t* è un'unità di tempo come il secondo o l'ora. L'unità non è pluralizzata. L'unità di tempo *t* is deve essere una delle seguenti stringhe letterali:
	
	• `nanosecond`
	• `microsecond`
	• `millisecond`
	• `second`
	• `minute`
	• `hour`
	• `day`
	• `ns`(uguale a) `nanosecond`
	• `us`(uguale a`microsecond`)
	• `ms`(uguale a`millisecond`)
	• `s`(uguale a`second`)
	• `m`(uguale a`minute`)
	• h(uguale a`hour`)
	• d(uguale a`day`)
	
	Esempi:
	
	`17s`
	
	`12second`
	
	`21hour`

Tipo di dati	Descrizione
	`2d`
`timeseries[row(timestamp, T,...)]`	Rappresenta i valori di una misura registrata in un intervallo di tempo come `array` composta da `row` oggetti. Ciascuno `row` contiene uno `timestamp` o più valori di misura del tipo di dati *T*dove: *T* può essere uno qualsiasi dei `bigint`,`boolean`,`double`, `ovarchar`. Le righe sono ordinate in ordine crescente per. `timestamp` Il tipo di dati della serie temporale rappresenta i valori di una misura nel tempo.
`unknown`	Rappresenta dati nulli.

Funzionalità integrata per le serie temporali

Timestream for LiveAnalytics offre funzionalità integrate di serie temporali che trattano i dati delle serie temporali come un concetto di prima classe.

La funzionalità integrata delle serie temporali può essere suddivisa in due categorie: visualizzazioni e funzioni.

Di seguito puoi leggere informazioni su ciascun costrutto.

Argomenti

- Visualizzazioni delle serie temporali
- Funzioni delle serie temporali

Visualizzazioni delle serie temporali

Timestream for LiveAnalytics supporta le seguenti funzioni per trasformare i dati nel tipo di dati: `timeseries`

Argomenti

- CREATE_TIME_SERIES
- UNNEST

CREATE_TIME_SERIES

CREATE_ TIME _ SERIES è una funzione di aggregazione che prende tutte le misurazioni grezze di una serie temporale (valori temporali e di misura) e restituisce un tipo di dati della serie temporale. La sintassi di questa funzione è la seguente:

```
CREATE_TIME_SERIES(time, measure_value::<data_type>)
```

where <data_type> è il tipo di dati del valore della misura e può essere uno tra bigint, boolean, double o varchar. Il secondo parametro non può essere nullo.

Considerate l'CPUutilizzo delle EC2 istanze archiviate in una tabella denominata metrics, come illustrato di seguito:

Orario	Regione	az	vpc	instance_id	measure_name	measure_value::double
2019-12-04 19:00:00.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef0	utilizzo della cpu_	35,0
2019-12-04 19:00:01.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef0	utilizzo della cpu_	38.2
2019-12-04 19:00:02,000 000000	us-east-1	Stati Uniti - est-1d	vpc-1a2b3c4d	i-1234567890abcdef0	utilizzo della cpu_	45,3
2019-12-04 19:00:00.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef1	cpu_utilization	54.1

Orario	Regione	az	vpc	instance_id	measure_name	measure_value::double
2019-12-04 19:00:01.000 000000	us-east-1	Stati Uniti est-1d	vpc-1a2b3c4d	i-1234567890abcdef1	cpu_utilization	42,5
2019-12-04 19:00:02,000 000000	us-east-1	Stati Uniti - est-1d	vpc-1a2b3c4d	i-1234567890abcdef1	cpu_utilization	33.7

Esecuzione della query:

```
SELECT region, az, vpc, instance_id, CREATE_TIME_SERIES(time, measure_value::double) as
 cpu_utilization FROM metrics
    WHERE measure_name='cpu_utilization'
    GROUP BY region, az, vpc, instance_id
```

restituirà tutte le serie che hanno `cpu_utilization` come valore di misura. In questo caso, abbiamo due serie:

Regione	az	vpc	instance_id	cpu_utilization
us-east-1	us-east-1d	vpc-1a2b3c4d	i-1234567890abcdef0	[{ora: 2019-12-04 19:00:00.000 000000, valore_misura: :doppio: 35.0}, {ora: 2019-12-04 19:00:01.000 000000, valore_misura: :doppio:

Regione	az	vpc	instance_id	cpu_utilization
				38.2}, {ora: 2019-12-04 19:00:02.000 000000, valore_misura: :doppio: 45,3}]
us-east-1	us-east-1d	vpc-1a2b3c4d	i-1234567890abcdef1	[{ora: 2019-12-04 19:00:00.000 000000, valore_misura: :doppio: 35.1}, {ora: 2019-12-04 19:00:01.000 000000, valore_misura: :doppio: 38,5}, {ora: 2019-12-04 19:00:02.000 000000, valore_misura: :doppio: 45,7}]

UNNEST

`UNNEST`è una funzione di tabella `timeseries` che consente di trasformare i dati in un modello piatto. La sintassi è esposta di seguito:

`UNNEST``timeseries`trasforma a in due colonne, vale a dire `time` `evalue`. Puoi anche usare alias con UNNEST come mostrato di seguito:

```
UNNEST(timeseries) AS <alias_name> (time_alias, value_alias)
```

dove <alias_name> è l'alias per la tabella piatta, `time_alias` è l'alias per la `time` colonna ed `value_alias` è l'alias per la colonna. `value`

Ad esempio, considera lo scenario in cui alcune istanze del tuo parco EC2 istanze sono configurate per emettere metriche a intervalli di 5 secondi, altre emettono metriche a intervalli di 15 secondi e hai bisogno delle metriche medie per tutte le istanze con una granularità di 10 secondi nelle ultime 6 ore. Per ottenere questi dati, trasformi le metriche nel modello delle serie temporali utilizzando _ _. CREATE TIME SERIES È quindi possibile utilizzare INTERPOLATE_ LINEAR per ottenere i valori mancanti con una granularità di 10 secondi. Successivamente, trasformi i dati nel modello flat utilizzando UNNESTe quindi utilizzando AVGper ottenere le metriche medie su tutte le istanze.

```
WITH interpolated_timeseries AS (
    SELECT region, az, vpc, instance_id,
        INTERPOLATE_LINEAR(
            CREATE_TIME_SERIES(time, measure_value::double),
                SEQUENCE(ago(6h), now(), 10s)) AS interpolated_cpu_utilization
    FROM timestreamdb.metrics
    WHERE measure_name= ‘cpu_utilization’ AND time >= ago(6h)
    GROUP BY region, az, vpc, instance_id
)
SELECT region, az, vpc, instance_id, avg(t.cpu_util)
FROM interpolated_timeseries
CROSS JOIN UNNEST(interpolated_cpu_utilization) AS t (time, cpu_util)
GROUP BY region, az, vpc, instance_id
```

La query precedente dimostra l'uso di UNNESTcon un alias. Di seguito è riportato un esempio della stessa query senza utilizzare un alias per: UNNEST

```
WITH interpolated_timeseries AS (
    SELECT region, az, vpc, instance_id,
        INTERPOLATE_LINEAR(
            CREATE_TIME_SERIES(time, measure_value::double),
                SEQUENCE(ago(6h), now(), 10s)) AS interpolated_cpu_utilization
    FROM timestreamdb.metrics
    WHERE measure_name= ‘cpu_utilization’ AND time >= ago(6h)
    GROUP BY region, az, vpc, instance_id
)
SELECT region, az, vpc, instance_id, avg(value)
FROM interpolated_timeseries
```

```
CROSS JOIN UNNEST(interpolated_cpu_utilization)
GROUP BY region, az, vpc, instance_id
```

Funzioni delle serie temporali

Amazon Timestream LiveAnalytics for supporta funzioni di serie temporali, come derivati, integrali e correlazioni, oltre ad altre, per ricavare informazioni più approfondite dai dati delle serie temporali. Questa sezione fornisce informazioni sull'utilizzo di ciascuna di queste funzioni, oltre a query di esempio. Seleziona uno dei seguenti argomenti per saperne di più.

Argomenti

Funzioni di interpolazione

Se nei dati delle serie temporali mancano valori per gli eventi in determinati momenti, puoi stimare i valori di tali eventi mancanti utilizzando l'interpolazione. Amazon Timestream supporta quattro varianti di interpolazione: interpolazione lineare, interpolazione spline cubica, interpolazione dell'ultima osservazione portata avanti (locf) e interpolazione costante. Questa sezione fornisce informazioni sull'utilizzo di Timestream per le funzioni di interpolazione, oltre a query di esempio. LiveAnalytics

Informazioni di utilizzo

Funzione	Tipo di dati di output	Descrizione
`interpolate_linear (timeseries, array[timestamp])`	serie temporali	Compila i dati mancanti utilizzando l'interpolazione lineare.
`interpolate_linear (timeseries, timestamp)`	double	Compila i dati mancanti utilizzando l'interpolazione lineare.

Funzione	Tipo di dati di output	Descrizione
`interpolate_spline_cubic(timeseries, array[timestamp])`	serie temporali	Compila i dati mancanti utilizzando l'interpolazione spline cubica.
`interpolate_spline_cubic(timeseries, timestamp)`	double	Compila i dati mancanti utilizzando l'interpolazione spline cubica.
`interpolate_locf(timeseries, array[timestamp])`	serie temporali	Compila i dati mancanti utilizzando l'ultimo valore campionato.
`interpolate_locf(timeseries, timestamp)`	double	Compila i dati mancanti utilizzando l'ultimo valore campionato.
`interpolate_fill(timeseries, array[timestamp], double)`	serie temporali	Compila i dati mancanti utilizzando un valore costante.
`interpolate_fill(timeseries, timestamp, double)`	double	Compila i dati mancanti utilizzando un valore costante.

Esempi di query

Example

Trova l'CPUutilizzo medio suddiviso a intervalli di 30 secondi per un EC2 host specifico nelle ultime 2 ore, inserendo i valori mancanti utilizzando l'interpolazione lineare:

```
WITH binned_timeseries AS (
SELECT hostname, BIN(time, 30s) AS binned_timestamp, ROUND(AVG(measure_value::double),
 2) AS avg_cpu_utilization
FROM "sampleDB".DevOps
WHERE measure_name = 'cpu_utilization'
    AND hostname = 'host-Hovjv'
```

```
    AND time > ago(2h)
GROUP BY hostname, BIN(time, 30s)
), interpolated_timeseries AS (
SELECT hostname,
    INTERPOLATE_LINEAR(
        CREATE_TIME_SERIES(binned_timestamp, avg_cpu_utilization),
            SEQUENCE(min(binned_timestamp), max(binned_timestamp), 15s)) AS
 interpolated_avg_cpu_utilization
FROM binned_timeseries
GROUP BY hostname
)
SELECT time, ROUND(value, 2) AS interpolated_cpu
FROM interpolated_timeseries
CROSS JOIN UNNEST(interpolated_avg_cpu_utilization)
```

Example

Trova l'CPUutilizzo medio raggruppato a intervalli di 30 secondi per un EC2 host specifico nelle ultime 2 ore, inserendo i valori mancanti utilizzando l'interpolazione basata sull'ultima osservazione effettuata:

```
WITH binned_timeseries AS (
SELECT hostname, BIN(time, 30s) AS binned_timestamp, ROUND(AVG(measure_value::double),
 2) AS avg_cpu_utilization
FROM "sampleDB".DevOps
WHERE measure_name = 'cpu_utilization'
    AND hostname = 'host-Hovjv'
    AND time > ago(2h)
GROUP BY hostname, BIN(time, 30s)
), interpolated_timeseries AS (
SELECT hostname,
    INTERPOLATE_LOCF(
        CREATE_TIME_SERIES(binned_timestamp, avg_cpu_utilization),
            SEQUENCE(min(binned_timestamp), max(binned_timestamp), 15s)) AS
 interpolated_avg_cpu_utilization
FROM binned_timeseries
GROUP BY hostname
)
SELECT time, ROUND(value, 2) AS interpolated_cpu
FROM interpolated_timeseries
CROSS JOIN UNNEST(interpolated_avg_cpu_utilization)
```

Funzioni derivate

I derivati vengono utilizzati per calcolare il tasso di variazione di una determinata metrica e possono essere utilizzati per rispondere in modo proattivo a un evento. Ad esempio, supponiamo di calcolare la derivata dell'CPUutilizzo delle EC2 istanze negli ultimi 5 minuti e di notare un derivato positivo significativo. Questo può essere indicativo di una maggiore domanda per il carico di lavoro, quindi potresti decidere di avviare più EC2 istanze per gestire meglio il tuo carico di lavoro.

Amazon Timestream supporta due varianti di funzioni derivate. Questa sezione fornisce informazioni sull'utilizzo di Timestream per le funzioni LiveAnalytics derivate, oltre a query di esempio.

Informazioni di utilizzo

Funzione	Tipo di dati di output	Descrizione
`derivative_linear( timeseries, interval)`	serie temporali	Calcola la derivata di ogni punto per il `timeseries` valore specificato. `interval`
`non_negative_deriv ative_linear(times eries, interval)`	serie temporali	Uguale a`derivativ e_linear(timeserie s, interval)` , ma restituisce solo valori positivi.

Esempi di query

Example

Calcola il tasso di variazione dell'CPUutilizzo ogni 5 minuti nell'ultima ora:

```
SELECT DERIVATIVE_LINEAR(CREATE_TIME_SERIES(time, measure_value::double), 5m) AS
 result
FROM “sampleDB”.DevOps
WHERE measure_name = 'cpu_utilization'
AND hostname = 'host-Hovjv' and time > ago(1h)
GROUP BY hostname, measure_name
```

Example

Calcola il tasso di aumento degli errori generati da uno o più microservizi:

```
WITH binned_view as (
    SELECT bin(time, 5m) as binned_timestamp, ROUND(AVG(measure_value::double), 2) as
 value
    FROM “sampleDB”.DevOps
    WHERE micro_service = 'jwt'
    AND time > ago(1h)
    AND measure_name = 'service_error'
    GROUP BY bin(time, 5m)
)
SELECT non_negative_derivative_linear(CREATE_TIME_SERIES(binned_timestamp, value), 1m)
 as rateOfErrorIncrease
FROM binned_view
```

Funzioni integrali

Puoi usare gli integrali per trovare l'area sotto la curva per unità di tempo per gli eventi delle serie temporali. Ad esempio, supponiamo di tenere traccia del volume di richieste ricevute dall'applicazione per unità di tempo. In questo scenario, puoi utilizzare la funzione integrale per determinare il volume totale di richieste servite per intervallo specificato in un periodo di tempo specifico.

Amazon Timestream supporta una variante di funzioni integrali. Questa sezione fornisce informazioni sull'utilizzo della funzione Timestream for LiveAnalytics Integral, oltre a query di esempio.

Informazioni di utilizzo

Funzione	Tipo di dati di output	Descrizione

`integral_trapezoidal(timeseries(double))`
 	
`integral_trapezoidal(timeseries(double), interval day to second)`
 	
`integral_trapezoidal(timeseries(bigint))`
	double	Approssima l'integrale secondo quanto specificato `interval day to second` per il modulo `timeseries` fornito, utilizzando la regola trapezoidale. Il parametro dell'intervallo dal giorno al secondo è facoltativo e l'impostazione predefinita è. `1s` Per ulteriori informazioni

Funzione	Tipo di dati di output	Descrizione

`integral_trapezoidal(timeseries(bigint), interval day to second)`
 	
`integral_trapezoidal(timeseries(integer), interval day to second)`
 	
`integral_trapezoidal(timeseries(integer))`
		sugli intervalli, vedere. Intervallo e durata

Esempi di query

Example

Calcola il volume totale di richieste servite ogni cinque minuti nell'ultima ora da un host specifico:

```
SELECT INTEGRAL_TRAPEZOIDAL(CREATE_TIME_SERIES(time, measure_value::double), 5m) AS
 result FROM sample.DevOps
WHERE measure_name = 'request'
AND hostname = 'host-Hovjv'
AND time > ago (1h)
GROUP BY hostname, measure_name
```

Funzioni di correlazione

Considerate due serie temporali di lunghezza simile, le funzioni di correlazione forniscono un coefficiente di correlazione, che spiega l'andamento delle due serie temporali nel tempo. Il coefficiente di correlazione varia da a. `-1.0` `1.0` `-1.0`indica che le due serie temporali hanno una tendenza in direzioni opposte alla stessa velocità, mentre `1.0` indica che le due serie temporali tendono nella stessa direzione alla stessa velocità. Un valore di `0` indica l'assenza di correlazione tra le due serie temporali. Ad esempio, se il prezzo del petrolio aumenta e il prezzo delle azioni di una compagnia petrolifera aumenta, la tendenza all'aumento del prezzo del petrolio e all'aumento del prezzo della compagnia petrolifera avrà un coefficiente di correlazione positivo.

Un coefficiente di correlazione positivo elevato indicherebbe che i due prezzi hanno un andamento simile. Analogamente, il coefficiente di correlazione tra i prezzi delle obbligazioni e i rendimenti obbligazionari è negativo, il che indica che questi due valori tendono nella direzione opposta nel tempo.

Amazon Timestream supporta due varianti di funzioni di correlazione. Questa sezione fornisce informazioni sull'utilizzo di Timestream per le funzioni di LiveAnalytics correlazione, oltre a query di esempio.

Informazioni di utilizzo

Funzione	Tipo di dati di output	Descrizione
`correlate_pearson( timeseries, timeseries)`	double	Calcola il coefficiente di correlazione di Pearson per i due. `timeseries` Le serie temporali devono avere gli stessi timestamp.
`correlate_spearman (timeseries, timeseries)`	double	Calcola il coefficiente di correlazione di Spearman per i due. `timeseries` Le serie temporali devono avere gli stessi timestamp.

Esempi di query

Example

```
WITH cte_1 AS (
    SELECT INTERPOLATE_LINEAR(
        CREATE_TIME_SERIES(time, measure_value::double),
        SEQUENCE(min(time), max(time), 10m)) AS result
    FROM sample.DevOps
    WHERE measure_name = 'cpu_utilization'
    AND hostname = 'host-Hovjv' AND time > ago(1h)
    GROUP BY hostname, measure_name
),
```

```
cte_2 AS (
    SELECT INTERPOLATE_LINEAR(
        CREATE_TIME_SERIES(time, measure_value::double),
        SEQUENCE(min(time), max(time), 10m)) AS result
    FROM sample.DevOps
    WHERE measure_name = 'cpu_utilization'
    AND hostname = 'host-Hovjv' AND time > ago(1h)
    GROUP BY hostname, measure_name
)
SELECT correlate_pearson(cte_1.result, cte_2.result) AS result
FROM cte_1, cte_2
```

Filtra e riduci le funzioni

Amazon Timestream supporta funzioni per l'esecuzione di filtri e la riduzione delle operazioni sui dati delle serie temporali. Questa sezione fornisce informazioni sull'utilizzo di Timestream per le funzioni di LiveAnalytics filtro e riduzione, oltre a query di esempio.

Informazioni di utilizzo

Funzione	Tipo di dati di output	Descrizione
`filter(timeseries(T), function(T, Boolean))`	serie temporali (T)	Costruisce una serie temporale da una serie temporale di input, utilizzando i valori per i quali il passato restituisce. `function` `true`
`reduce(timeseries(T), initialState S, inputFunction(S, T, S), outputFunction(S, R))`	R	Restituisce un singolo valore, ridotto dalla serie temporale . `inputFunction` Verrà invocato su ogni elemento delle serie temporali in ordine. Oltre a prendere l'elemento corrente, inputFunction prende lo stato corrente (inizialm `enteinitialState`) e restituisce il nuovo stato. `outputFunction` Verrà

Funzione	Tipo di dati di output	Descrizione
		invocato per trasformare lo stato finale nel valore del risultato. `outputFunction` Può essere una funzione di identità.

Esempi di query

Example

Costruisci una serie temporale di CPU utilizzo di un host e filtra i punti con misurazioni superiori a 70:

```
WITH time_series_view AS (
    SELECT INTERPOLATE_LINEAR(
        CREATE_TIME_SERIES(time, ROUND(measure_value::double,2)),
            SEQUENCE(ago(15m), ago(1m), 10s)) AS cpu_user
    FROM sample.DevOps
    WHERE hostname = 'host-Hovjv' and measure_name = 'cpu_utilization'
        AND time > ago(30m)
    GROUP BY hostname
)
SELECT FILTER(cpu_user, x -> x.value > 70.0) AS cpu_above_threshold
from time_series_view
```

Example

Costruisci una serie temporale di CPU utilizzo di un host e determina la somma al quadrato delle misurazioni:

```
WITH time_series_view AS (
    SELECT INTERPOLATE_LINEAR(
        CREATE_TIME_SERIES(time, ROUND(measure_value::double,2)),
            SEQUENCE(ago(15m), ago(1m), 10s)) AS cpu_user
    FROM sample.DevOps
    WHERE hostname = 'host-Hovjv' and measure_name = 'cpu_utilization'
        AND time > ago(30m)
    GROUP BY hostname
)
SELECT REDUCE(cpu_user,
```

```
    DOUBLE '0.0',
    (s, x) -> x.value * x.value + s,
    s -> s)
from time_series_view
```

Example

Costruisci una serie temporale di CPU utilizzo di un host e determina la frazione di campioni che supera la soglia: CPU

```
WITH time_series_view AS (
    SELECT INTERPOLATE_LINEAR(
        CREATE_TIME_SERIES(time, ROUND(measure_value::double,2)),
            SEQUENCE(ago(15m), ago(1m), 10s)) AS cpu_user
    FROM sample.DevOps
    WHERE hostname = 'host-Hovjv' and measure_name = 'cpu_utilization'
        AND time > ago(30m)
    GROUP BY hostname
)
SELECT ROUND(
    REDUCE(cpu_user,
      -- initial state
      CAST(ROW(0, 0) AS ROW(count_high BIGINT, count_total BIGINT)),
      -- function to count the total points and points above a certain threshold
      (s, x) -> CAST(ROW(s.count_high + IF(x.value > 70.0, 1, 0), s.count_total + 1) AS
 ROW(count_high BIGINT, count_total BIGINT)),
      -- output function converting the counts to fraction above threshold
      s -> IF(s.count_total = 0, NULL, CAST(s.count_high AS DOUBLE) / s.count_total)),
    4) AS fraction_cpu_above_threshold
from time_series_view
```

SQLsupporto

Timestream for LiveAnalytics supporta alcuni costrutti comuniSQL. Puoi leggere di più qui sotto.

Argomenti

- SELECT
- supporto per le subquery
- SHOWdichiarazioni
- DESCRIBEdichiarazioni

- UNLOAD

SELECT

SELECTle istruzioni possono essere utilizzate per recuperare dati da una o più tabelle. Il linguaggio di interrogazione di Timestream supporta la seguente sintassi per le istruzioni: SELECT

```
[ WITH with_query [, ...] ]
            SELECT [ ALL | DISTINCT ] select_expr [, ...]
            [ function (expression) OVER (
            [ PARTITION BY partition_expr_list ]
            [ ORDER BY order_list ]
            [ frame_clause ] )
            [ FROM from_item [, ...] ]
            [ WHERE condition ]
            [ GROUP BY [ ALL | DISTINCT ] grouping_element [, ...] ]
            [ HAVING condition]
            [ { UNION | INTERSECT | EXCEPT } [ ALL | DISTINCT ] select ]
            [ ORDER BY order_list ]
            [ LIMIT [ count | ALL ] ]
```

dove

- `function (expression)`è una delle funzioni di finestra supportate.
- `partition_expr_list`è:

  ```
  expression | column_name [, expr_list ]
  ```

- `order_list`è:

  ```
  expression | column_name [ ASC | DESC ]
  [ NULLS FIRST | NULLS LAST ]
  [, order_list ]
  ```

- `frame_clause`è:

  ```
  ROWS | RANGE
  { UNBOUNDED PRECEDING | expression PRECEDING | CURRENT ROW } |
  {BETWEEN
  { UNBOUNDED PRECEDING | expression { PRECEDING | FOLLOWING } |
  CURRENT ROW}
  ```

```
AND
{ UNBOUNDED FOLLOWING | expression { PRECEDING | FOLLOWING } |
CURRENT ROW }}
```

- `from_item`è uno dei:

```
table_name [ [ AS ] alias [ ( column_alias [, ...] ) ] ]
from_item join_type from_item [ ON join_condition | USING ( join_column [, ...] ) ]
```

- `join_type`è uno dei seguenti:

```
[ INNER ] JOIN
LEFT [ OUTER ] JOIN
RIGHT [ OUTER ] JOIN
FULL [ OUTER ] JOIN
```

- `grouping_element`è uno dei seguenti:

```
()
expression
```

supporto per le subquery

Timestream supporta sottoquery e predicati. `EXISTS` `IN` Il `EXISTS` predicato determina se una sottoquery restituisce delle righe. Il `IN` predicato determina se i valori prodotti dalla sottoquery corrispondono ai valori o all'espressione della clausola IN. Il linguaggio di interrogazione Timestream supporta le sottoquery correlate e di altro tipo.

```
SELECT t.c1
FROM (VALUES 1, 2, 3, 4, 5) AS t(c1)
WHERE EXISTS
(SELECT t.c2
 FROM (VALUES 1, 2, 3) AS t(c2)
 WHERE t.c1= t.c2
)
ORDER BY t.c1
```

c1
1

c1
2
3

```
SELECT t.c1
FROM (VALUES 1, 2, 3, 4, 5) AS t(c1)
WHERE t.c1 IN
(SELECT t.c2
 FROM (VALUES 2, 3, 4) AS t(c2)
)
ORDER BY t.c1
```

c1
2
3
4

SHOWdichiarazioni

È possibile visualizzare tutti i database di un account utilizzando l'`SHOW DATABASES`estratto conto. La sintassi è esposta di seguito:

```
SHOW DATABASES [LIKE pattern]
```

dove la `LIKE` clausola può essere utilizzata per filtrare i nomi dei database.

È possibile visualizzare tutte le tabelle di un account utilizzando l'`SHOW TABLES`estratto conto. La sintassi è esposta di seguito:

```
SHOW TABLES [FROM database] [LIKE pattern]
```

dove la `FROM` clausola può essere utilizzata per filtrare i nomi dei database e la `LIKE` clausola può essere utilizzata per filtrare i nomi delle tabelle.

È possibile visualizzare tutte le misure di una tabella utilizzando l'`SHOW MEASURES`istruzione. La sintassi è esposta di seguito:

```
SHOW MEASURES FROM database.table [LIKE pattern]
```

dove la `FROM` clausola verrà utilizzata per specificare il nome del database e della tabella e la `LIKE` clausola può essere utilizzata per filtrare i nomi delle misure.

DESCRIBEdichiarazioni

È possibile visualizzare i metadati di una tabella utilizzando l'`DESCRIBE`istruzione. La sintassi è esposta di seguito:

```
DESCRIBE database.table
```

dove `table` contiene il nome della tabella. L'istruzione describe restituisce i nomi delle colonne e i tipi di dati per la tabella.

UNLOAD

Timestream for LiveAnalytics supporta un `UNLOAD` comando come estensione del relativo SQL supporto. I tipi di dati supportati da `UNLOAD` sono descritti in. Tipi di dati supportati I `unknown` tipi `time` e non si applicano a`UNLOAD`.

```
UNLOAD (SELECT statement)
 TO 's3://bucket-name/folder'
 WITH ( option = expression [, ...] )
```

dove si trova l'opzione

```
{ partitioned_by = ARRAY[ col_name[,…] ]
 | format = [ '{ CSV | PARQUET }' ]
 | compression = [ '{ GZIP | NONE }' ]
 | encryption = [ '{ SSE_KMS | SSE_S3 }' ]
 | kms_key = '<string>'
 | field_delimiter ='<character>'
 | escaped_by = '<character>'
 | include_header = ['{true, false}']
 | max_file_size = '<value>'
}
```

SELECTdichiarazione

L'istruzione di query utilizzata per selezionare e recuperare i dati da uno o più Timestream per le tabelle. LiveAnalytics

```
(SELECT column 1, column 2, column 3 from database.table
      where measure_name = "ABC" and timestamp between ago (1d) and now() )
```

Clausola TO

```
TO 's3://bucket-name/folder'
```

oppure

```
TO 's3://access-point-alias/folder'
```

La TO clausola dell'UNLOADistruzione specifica la destinazione per l'output dei risultati della query. È necessario fornire il percorso completo, incluso il nome del bucket Amazon S3 o Amazon S3 con access-point-alias posizione della cartella su Amazon S3 dove Timestream for scrive gli oggetti del file di output. LiveAnalytics Il bucket S3 deve appartenere allo stesso account e nella stessa regione. Oltre al set di risultati della query, Timestream for LiveAnalytics scrive i file manifest e di metadati nella cartella di destinazione specificata.

PARTITIONEDClausola _BY

```
partitioned_by = ARRAY [col_name[,…] , (default: none)
```

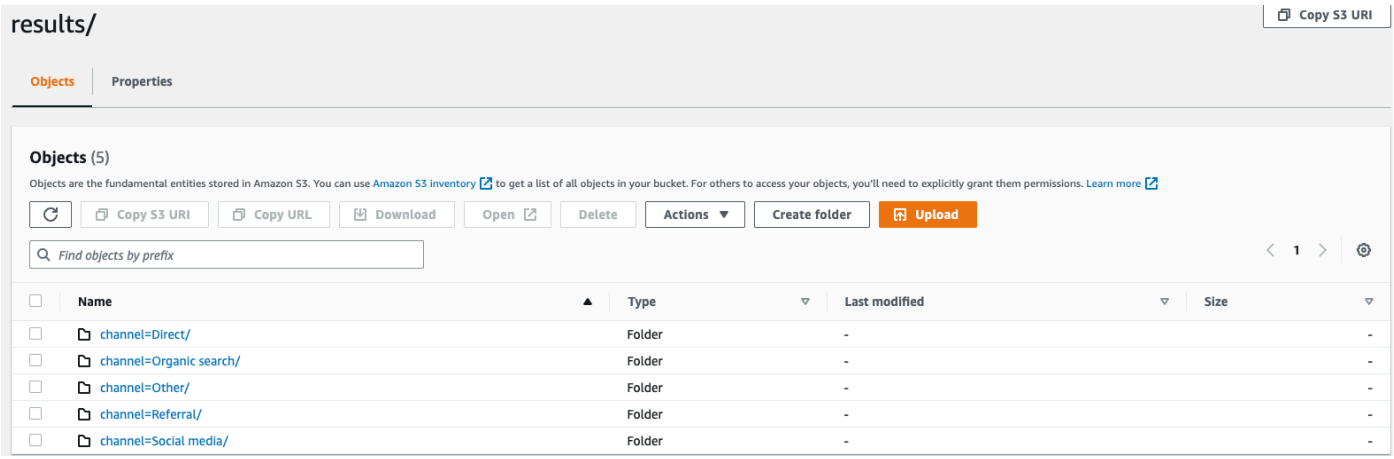
La `partitioned_by` clausola viene utilizzata nelle query per raggruppare e analizzare i dati a livello granulare. Quando esporti i risultati della query nel bucket S3, puoi scegliere di partizionare i dati in base a una o più colonne nella query di selezione. Durante il partizionamento dei dati, i dati esportati vengono suddivisi in sottoinsiemi in base alla colonna della partizione e ogni sottoinsieme viene archiviato in una cartella separata. All'interno della cartella dei risultati che contiene i dati esportati, viene creata automaticamente una sottocartella. `folder/results/partition column = partition value/` Tuttavia, tieni presente che le colonne partizionate non sono incluse nel file di output.

`partitioned_by`non è una clausola obbligatoria nella sintassi. Se si sceglie di esportare i dati senza alcun partizionamento, è possibile escludere la clausola nella sintassi.

Example

Supponendo che tu stia monitorando i dati clickstream del tuo sito Web e che tu abbia 5 canali di traffico, vale a dire,, e. `direct` `Social Media` `Organic Search` `Other` `Referral` Quando si esportano i dati, è possibile scegliere di partizionarli utilizzando la colonna. `Channel` All'interno della tua cartella dati`s3://bucketname/results`, avrai cinque cartelle ciascuna con il rispettivo nome del canale, ad esempio, `s3://bucketname/results/channel=Social Media/.` all'interno di questa cartella troverai i dati di tutti i clienti che sono arrivati sul tuo sito web attraverso il `Social Media` canale. Allo stesso modo, avrai altre cartelle per i canali rimanenti.

Dati esportati partizionati per colonna Channel



FORMAT

```
format = [ '{ CSV | PARQUET }' , default: CSV
```

Le parole chiave per specificare il formato dei risultati della query scritti nel bucket S3. È possibile esportare i dati come valore separato da virgole (CSV) utilizzando una virgola (,) come delimitatore predefinito o nel formato Apache Parquet, un efficiente formato di archiviazione a colonne aperto per l'analisi.

COMPRESSION

```
compression = [ '{ GZIP | NONE }' ], default: GZIP
```

È possibile comprimere i dati esportati utilizzando l'algoritmo di compressione GZIP o decomprimerli specificando l'opzione. `NONE`

ENCRYPTION

```
encryption = [ '{ SSE_KMS | SSE_S3 }' ], default: SSE_S3
```

I file di output su Amazon S3 vengono crittografati utilizzando l'opzione di crittografia selezionata. Oltre ai dati, anche i file manifest e i file di metadati vengono crittografati in base all'opzione di crittografia selezionata. Al momento supportiamo la SSE crittografia _S3 e SSE _. KMS SSE_S3 è una crittografia lato server con Amazon S3 che crittografa i dati utilizzando la crittografia Advanced Encryption Standard () a 256 bit. AES SSE_ KMS è una crittografia lato server per crittografare i dati utilizzando chiavi gestite dal cliente.

KMS_KEY

```
kms_key = '<string>'
```

KMSKey è una chiave definita dal cliente per crittografare i risultati delle query esportate. KMSLa chiave è gestita in modo sicuro da AWS Key Management Service (AWS KMS) e utilizzata per crittografare i file di dati su Amazon S3.

FIELD_DELIMITER

```
field_delimiter ='<character>' , default: (,)
```

Quando si esportano i dati in CSV formato, questo campo specifica un singolo ASCII carattere utilizzato per separare i campi nel file di output, ad esempio un carattere pipe (|), una virgola (,) o un tab (/t). Il delimitatore predefinito per CSV i file è un carattere virgola. Se un valore nei dati contiene il delimitatore scelto, il delimitatore verrà citato tra virgolette. Ad esempio, se il valore dei dati contiene`Time,stream`, questo valore verrà citato come nei dati esportati. `"Time,stream"` Il carattere di virgoletta usato da Timestream per sono le LiveAnalytics virgolette doppie («).

Evitate di specificare il carattere di ritorno al carrello (ASCII`13`0D, hex, text '\ r') o il carattere di interruzione di riga (ASCII10, hex 0A, text'\n') come `FIELD_DELIMITER` se desiderate includere le intestazioni inCSV, poiché ciò impedirà a molti parser di analizzare correttamente le intestazioni nell'output risultante. CSV

ESCAPED_DI

```
escaped_by = '<character>', default: (\)
```

Quando si esportano i dati in CSV formato, questo campo specifica il carattere che deve essere trattato come carattere di escape nel file di dati scritto nel bucket S3. L'escape avviene nei seguenti scenari:

1. Se il valore stesso contiene il carattere di virgoletta («), verrà eliminato utilizzando un carattere di escape. Ad esempio, se il valore è`Time"stream`, dove (\) è il carattere di escape configurato, allora verrà escluso come. `Time\"stream`
2. Se il valore contiene il carattere di escape configurato, verrà eliminato. Ad esempio, se il valore è`Time\stream`, verrà scappato come. `Time\\stream`

> **Note**
>
> Se l'output esportato contiene tipi di dati complessi come Arrays, Rows o Timeseries, verrà serializzato come stringa. JSON Di seguito è riportato un esempio.

Tipo di dati	Valore effettivo	Modalità di escape del valore nel CSV formato [stringa serializzataJSON]
Array	`[ 23,24,25 ]`	`"[23,24,25]"`
Riga	`( x=23.0, y=hello )`	`"{\"x\":23.0,\"y\":\"hello\"}"`
Serie temporali	`[ ( time=1970-01-01 00:00:00.000000010 , value=100.0 ), ( time=1970-01-01 00:00:00.000000012, value=120.0 ) ]`	`"[{\"time\":\"1970-01-01 00:00:00.000000010Z\",\"value\":100.0},{\"time\":\"1970-01-01 00:00:00.000000012Z\",\"value\":120.0}]"`

INCLUDE_HEADER

```
include_header = 'true' , default: 'false'
```

Quando si esportano i dati in CSV formato, questo campo consente di includere i nomi delle colonne come prima riga dei file di dati CSV esportati.

I valori accettati sono «true» e «false» e il valore predefinito è «false». Le opzioni di trasformazione del testo come `escaped_by` e `field_delimiter` si applicano anche alle intestazioni.

> **Note**
>
> Quando si includono le intestazioni, è importante non selezionare un carattere di ritorno (ASCII13, hex 0D, text '\ r') o un carattere di interruzione di riga (10, hex 0A, text'\n') come carattere di interruzione di riga (ASCII10, hex 0A, text'\n')`FIELD_DELIMITER`, poiché ciò impedirà a molti parser di analizzare correttamente le intestazioni nell'output risultante. CSV

MAX_FILE_SIZE

```
max_file_size = 'X[MB|GB]' , default: '78GB'
```

Questo campo specifica la dimensione massima dei file che l'`UNLOAD`istruzione crea in Amazon S3. L'`UNLOAD`istruzione può creare più file, ma la dimensione massima di ogni file scritto in Amazon S3 sarà approssimativamente quella specificata in questo campo.

Il valore del campo deve essere compreso tra 16 MB e 78 GB, inclusi. È possibile specificarlo in numeri interi come `12GB` o in decimali come o. `0.5GB` `24.7MB` Il valore predefinito è 78 GB.

La dimensione effettiva del file è approssimativa al momento della scrittura del file, pertanto la dimensione massima effettiva potrebbe non essere esattamente uguale al numero specificato.

Operatori logici

Timestream for LiveAnalytics supporta i seguenti operatori logici.

Operatore	Descrizione	Esempio
AND	Vero se entrambi i valori sono veri	a AND b

Operatore	Descrizione	Esempio
O	Vero se uno dei due valori è vero	a OPPURE b
NOT	Vero se il valore è falso	NOTa

- Il risultato di un `AND` confronto può essere `NULL` se uno o entrambi i lati dell'espressione lo sono`NULL`.
- Se almeno un lato di un `AND` operatore è, `FALSE` l'espressione restituisce. `FALSE`
- Il risultato di un `OR` confronto può essere `NULL` se uno o entrambi i lati dell'espressione lo sono`NULL`.
- Se almeno un lato di un `OR` operatore è, `TRUE` l'espressione restituisce. `TRUE`
- Il complemento logico di `NULL` is`NULL`.

La seguente tabella di verità mostra la gestione di `NULL` in `AND` e`OR`:

A	B	A e b	A o b
null	null	null	null
false	null	false	null
null	false	false	null
true	null	null	true
null	true	null	true
false	false	false	false
true	false	false	true
false	true	false	true
true	true	true	true

La seguente tabella di verità mostra la gestione di NULL inNOT:

A	Non un
null	null
true	false
false	true

Operatori di confronto

Timestream for LiveAnalytics supporta i seguenti operatori di confronto.

Operatore	Descrizione
<	Minore di
>	Maggiore di
<=	Minore o uguale a
>=	Maggiore o uguale a
=	Uguale
<>	Non uguale
!=	Non uguale

Note

- L'`BETWEEN`operatore verifica se un valore rientra in un intervallo specificato. La sintassi è esposta di seguito:

```
BETWEEN min AND max
```

La presenza di NULL in un'NOT BETWEENistruzione BETWEEN or comporterà la valutazione diNULL.

- IS NULL e IS NOT NULL gli operatori verificano se un valore è nullo (non definito). L'utilizzo di NULL with IS NULL restituisce vero.
- InSQL, un NULL valore indica un valore sconosciuto.

Funzioni di confronto

Timestream for LiveAnalytics supporta le seguenti funzioni di confronto.

Argomenti

più grande ()

La funzione greatest () restituisce il più grande dei valori forniti. Restituisce NULL se uno qualsiasi dei valori forniti lo èNULL. La sintassi è esposta di seguito.

```
greatest(value1, value2, ..., valueN)
```

almeno ()

La funzione least () restituisce il più piccolo dei valori forniti. Restituisce NULL se uno qualsiasi dei valori forniti lo èNULL. La sintassi è esposta di seguito.

```
least(value1, value2, ..., valueN)
```

ALL(), ANY () e SOME ()

I ALL SOME quantificatori ANY e possono essere utilizzati insieme agli operatori di confronto nel modo seguente.

Expression	Significato
A = ALL (...)	Restituisce vero quando A è uguale a tutti i valori.
A <> ALL (...)	Restituisce vero quando A non corrisponde a nessun valore.
A < ALL (...)	Restituisce vero quando A è inferiore al valore più piccolo.
A = ANY (...)	Restituisce vero quando A è uguale a uno qualsiasi dei valori.
A <> ANY (...)	Restituisce vero quando A non corrisponde a uno o più valori.
A < ANY (...)	Restituisce vero quando A è inferiore al valore più grande.

Esempi e note di utilizzo

> **Note**
>
> Quando si utilizza `ANY` o `ALLSOME`, la parola chiave `VALUES` deve essere utilizzata se i valori di confronto sono un elenco di valori letterali.

Esempio: **`ANY()`**

Di seguito è riportato un esempio di istruzione `ANY()` in una query.

```
SELECT 11.7 = ANY (VALUES 12.0, 13.5, 11.7)
```

Di seguito è riportata una sintassi alternativa per la stessa operazione.

```
SELECT 11.7 = ANY (SELECT 12.0 UNION ALL SELECT 13.5 UNION ALL SELECT 11.7)
```

In questo caso, `ANY()` restituisce. `True`

Esempio: **`ALL()`**

Di seguito è riportato un esempio di istruzione `ALL()` in una query.

```
SELECT 17 < ALL (VALUES 19, 20, 15);
```

Di seguito è riportata una sintassi alternativa per la stessa operazione.

```
SELECT 17 < ALL (SELECT 19 UNION ALL SELECT 20 UNION ALL SELECT 15);
```

In questo caso, `ALL()` restituisce. `False`

Esempio: **`SOME()`**

Di seguito è riportato un esempio di istruzione `SOME()` in una query.

```
SELECT 50 >= SOME (VALUES 53, 77, 27);
```

Di seguito è riportata una sintassi alternativa per la stessa operazione.

```
SELECT 50 >= SOME (SELECT 53 UNION ALL SELECT 77 UNION ALL SELECT 27);
```

In questo caso, `SOME()` restituisce. `True`

Espressioni condizionali

Timestream for LiveAnalytics supporta le seguenti espressioni condizionali.

Argomenti

La dichiarazione CASE

L'CASEistruzione cerca ogni espressione di valore da sinistra a destra finché non ne trova una ugualeexpression. Se trova una corrispondenza, viene restituito il risultato per il valore

corrispondente. Se non viene trovata alcuna corrispondenza, viene restituito il risultato della `ELSE` clausola se esiste; in caso contrario `null` viene restituito. La sintassi è esposta di seguito:

```
CASE expression
    WHEN value THEN result
    [ WHEN ... ]
    [ ELSE result ]
END
```

Timestream supporta anche la seguente sintassi per le istruzioni. CASE In questa sintassi, il modulo «cercato» valuta ogni condizione booleana da sinistra a destra finché non ne è presente una e restituisce il risultato corrispondente. `true` Se non ci sono condizioni`true`, viene restituito il risultato della `ELSE` clausola se esiste; in caso contrario viene restituito. `null` Vedi sotto per la sintassi alternativa:

```
CASE
    WHEN condition THEN result
    [ WHEN ... ]
    [ ELSE result ]
END
```

L'istruzione IF

L'istruzione IF valuta una condizione come vera o falsa e restituisce il valore appropriato. Timestream supporta le seguenti due rappresentazioni sintattiche per IF:

```
if(condition, true_value)
```

Questa sintassi valuta e restituisce `true_value` se la condizione è`true`; altrimenti viene restituita e non viene `null` valutata. `true_value`

```
if(condition, true_value, false_value)
```

Questa sintassi valuta e restituisce `true_value` se la condizione è`true`, altrimenti valuta e restituisce. `false_value`

Esempi

```
SELECT
```

```
  if(true, 'example 1'),
  if(false, 'example 2'),
  if(true, 'example 3 true', 'example 3 false'),
  if(false, 'example 4 true', 'example 4 false')
```

_col0	_col1	_col2	_col3
example 1	- null	example 3 true	example 4 false

La dichiarazione COALESCE

COALESCErestituisce il primo valore non nullo in un elenco di argomenti. La sintassi è esposta di seguito:

```
coalesce(value1, value2[,...])
```

La dichiarazione NULLIF

L'istruzione IF valuta una condizione come vera o falsa e restituisce il valore appropriato. Timestream supporta le seguenti due rappresentazioni sintattiche per IF:

NULLIFrestituisce null se `value1` è uguale; altrimenti restituisce. `value2` `value1` La sintassi è esposta di seguito:

```
nullif(value1, value2)
```

La TRY dichiarazione

La TRYfunzione valuta un'espressione e gestisce determinati tipi di errori `null` restituendoli. La sintassi è esposta di seguito:

```
try(expression)
```

Funzioni di conversione

Timestream for LiveAnalytics supporta le seguenti funzioni di conversione.

Argomenti

cast()

La sintassi della funzione cast per trasmettere esplicitamente un valore come tipo è la seguente.

```
cast(value AS type)
```

try_cast ()

Timestream for supporta LiveAnalytics anche la funzione try_cast che è simile a cast ma restituisce null se il cast fallisce. La sintassi è esposta di seguito.

```
try_cast(value AS type)
```

Operatori matematici

Timestream for supporta i seguenti operatori matematici. LiveAnalytics

Operatore	Descrizione
+	Addizione
-	Sottrazione
*	Moltiplicazione
/	Divisione (la divisione di numeri interi esegue il troncamento)
%	Modulo (resto)

Funzioni matematiche

Timestream for LiveAnalytics supporta le seguenti funzioni matematiche.

Funzione	Tipo di dati di output	Descrizione
abs (x)	[uguale all'input]	Restituisce il valore assoluto di x.
cbrt (x)	double	Restituisce la radice cubica di x.
soffitto (x) o soffitto (x)	[uguale all'input]	Restituisce x arrotondato al numero intero più vicino.
gradi (x)	double	Converte l'angolo x in radianti in gradi.
e ()	double	Restituisce il numero costante di Eulero.
exp (x)	double	Restituisce il numero di Eulero elevato alla potenza di x.
pavimento (x)	[uguale all'input]	Restituisce x arrotondato per difetto al numero intero più vicino.
from_base (stringa, radice)	bigint	Restituisce il valore di una stringa interpretata come un numero di radice di base.
ln (x)	double	Restituisce il logaritmo naturale di x.
log2 (x)	double	Restituisce il logaritmo in base 2 di x.
log10 (x)	double	Restituisce il logaritmo in base 10 di x.
modalità (n, m)	[uguale all'input]	Restituisce il modulo (resto) di n diviso per m.

Funzione	Tipo di dati di output	Descrizione
pi ()	double	Restituisce la costante Pi.
pow (x, p) o power (x, p)	double	Restituisce x elevato alla potenza di p.
radianti (x)	double	Converte l'angolo x in gradi in radianti.
rand () o random ()	double	Restituisce un valore pseudo-casuale nell'intervallo 0,0 1,0.
casuale (n)	[uguale all'input]	Restituisce un numero pseudo-casuale compreso tra 0 e n (esclusivo).
rotondo (x)	[uguale all'input]	Restituisce x arrotondato al numero intero più vicino.
rotondo (x, d)	[uguale all'input]	Restituisce x arrotondato a d cifre decimali.
segno (x)	[uguale all'input]	Restituisce la funzione signum di x, ovvero: • 0 se l'argomento è 0 • 1 se l'argomento è maggiore di 0 • -1 se l'argomento è minore di 0. Per argomenti doppi, la funzione restituisce inoltre: • NaN se l'argomento è NaN • 1 se l'argomento è +Infinity • -1 se l'argomento è -Infinity.

Funzione	Tipo di dati di output	Descrizione
sqrt (x)	double	Restituisce la radice quadrata di x.
to_base (x, radix)	varchar	Restituisce la rappresentazione della radice di base di x.
troncare (x)	double	Restituisce x arrotondato a un numero intero eliminando le cifre dopo il punto decimale.
tacos (x)	double	Restituisce l'arcoseno di x.
asino (x)	double	Restituisce l'arcoseno di x.
atan (x)	double	Restituisce l'arcotangente di x.
atan2 (y, x)	double	Restituisce l'arcotangente di y/x.
cos (x)	double	Restituisce il coseno di x.
cosh (x)	double	Restituisce il coseno iperbolico di x.
peccato (x)	double	Restituisce il seno di x.
tan (x)	double	Restituisce la tangente di x.
tanh (x)	double	Restituisce la tangente iperbolica di x.
infinito ()	double	Restituisce la costante che rappresenta l'infinito positivo.
is_finite (x)	booleano	Determina se x è finito.
is_infinite (x)	booleano	Determina se x è infinito.

Funzione	Tipo di dati di output	Descrizione
is_nan (x)	booleano	Determina se x è. not-a-number
nano ()	double	Restituisce la costante che rappresenta not-a-number.

Operatori di stringa

Timestream for LiveAnalytics supporta l'| |operatore per concatenare una o più stringhe.

Funzioni stringa

> **Note**
>
> Si presume che il tipo di dati di input di queste funzioni sia varchar, se non diversamente specificato.

Funzione	Tipo di dati di output	Descrizione
chr (n)	varchar	Restituisce il punto di codice Unicode n come varchar.
codepoint (x)	integer	Restituisce il punto di codice Unicode dell'unico carattere di str.
concat (x1,..., xN)	varchar	Restituisce la concatenazione di x1, x2,..., xN.
hamming_distance (x1, x2)	bigint	Restituisce la distanza di Hamming di x1 e x2, ovvero il numero di posizioni in cui i caratteri corrispondenti sono diversi. Nota che i due input

Funzione	Tipo di dati di output	Descrizione
		varchar devono avere la stessa lunghezza.
lunghezza (x)	bigint	Restituisce la lunghezza di x in caratteri.
levenshtein_distance (x1, x2)	bigint	Restituisce la distanza di modifica di Levenshtein di x1 e x2, ovvero il numero minimo di modifiche a carattere singolo (inserimenti, eliminazioni o sostituzioni) necessarie per cambiare x1 in x2.
inferiore (x)	varchar	Converte x in lettere minuscole.
carico (x1, grande dimensione, x2)	varchar	Pad sinistro x1 per ridimensi onare i caratteri con x2. Se la dimensione è inferiore alla lunghezza di x1, il risultato viene troncato in caratteri di dimensione. la dimensione non deve essere negativa e x2 non deve essere vuota.
ltrim (x)	varchar	Rimuove gli spazi bianchi iniziali da x.
sostituisci (x1, x2)	varchar	Rimuove tutte le istanze di x2 da x1.
sostituisci (x1, x2, x3)	varchar	Sostituisce tutte le istanze di x2 con x3 in x1.
Inverso (x)	varchar	Restituisce x con i caratteri in ordine inverso.

Funzione	Tipo di dati di output	Descrizione
rpad (x1, grande dimensione, x2)	varchar	Il tasto destro compatta x1 per ridimensionare i caratteri con x2. Se la dimensione è inferiore alla lunghezza di x1, il risultato viene troncato in caratteri di dimensione. la dimensione non deve essere negativa e x2 non deve essere vuota.
rtrim (x)	varchar	Rimuove gli spazi bianchi finali da x.
dividere (x1, x2)	array(varchar)	Divide x1 sul delimitatore x2 e restituisce un array.
split (x1, x2, limite bigint)	array(varchar)	Divide x1 sul delimitatore x2 e restituisce un array. L'ultimo elemento dell'array contiene sempre tutto ciò che rimane nel limite x1. Il limite deve essere un numero positivo.
split_part (x1, x2, bigint pos)	varchar	Divide x1 sul delimitatore x2 e restituisce il campo varchar in pos. Gli indici dei campi iniziano con 1. Se pos è maggiore del numero di campi, viene restituito null.
strpos (x1, x2)	bigint	Restituisce la posizione iniziale della prima istanza di x2 in x1. Le posizioni iniziano con 1. Se non viene trovato, viene restituito 0.

Funzione	Tipo di dati di output	Descrizione
strpos (x1, x2, istanza bigint)	bigint	Restituisce la posizione dell'ennesima istanza di x2 in x1. L'istanza deve essere un numero positivo. Le posizioni iniziano con 1. Se non viene trovato, viene restituito 0.
strrpos (x1, x2)	bigint	Restituisce la posizione iniziale dell'ultima istanza di x2 in x1. Le posizioni iniziano con 1. Se non viene trovato, viene restituito 0.
strrpos (x1, x2, bigint instance)	bigint	Restituisce la posizione dell'ennesima istanza di x2 in x1 a partire dalla fine di x1. l'istanza deve essere un numero positivo. Le posizioni iniziano con 1. Se non viene trovato, viene restituito 0.
posizione (x2 IN x1)	bigint	Restituisce la posizione iniziale della prima istanza di x2 in x1. Le posizioni iniziano con 1. Se non viene trovato, viene restituito 0.
substr (x, bigint start)	varchar	Restituisce il resto di x dalla posizione iniziale di inizio. Le posizioni iniziano con 1. Una posizione iniziale negativa viene interpretata come relativa alla fine di x.

Funzione	Tipo di dati di output	Descrizione
substr (x, bigint start, bigint len)	varchar	Restituisce una sottostringa da x di lunghezza len dalla posizione iniziale start. Le posizioni iniziano con 1. Una posizione iniziale negativa viene interpretata come relativa alla fine di x.
tagliare (x)	varchar	Rimuove gli spazi bianchi iniziali e finali da x.
superiore (x)	varchar	Converte x in maiuscolo.

Operatori di matrice

Timestream for LiveAnalytics supporta i seguenti operatori di array.

Operatore	Descrizione
[]	Accedi a un elemento di un array in cui il primo indice inizia da 1.
||	Concatena un array con un altro array o elemento dello stesso tipo.

Funzioni di array

Timestream for LiveAnalytics supporta le seguenti funzioni di array.

Funzione	Tipo di dati di output	Descrizione
array_distinct (x)	array	Rimuove i valori duplicati dall'array x.

Funzione	Tipo di dati di output	Descrizione
		`SELECT array_distinct(ARRAY[1,2,2,3])` Risultato di esempio: `[ 1,2,3 ]`
array_intersect (x, y)	array	Restituisce una matrice degli elementi nell'intersezione di x e y, senza duplicati. `SELECT array_intersect(ARRAY[1,2,3], ARRAY[3,4,5])` Risultato di esempio: `[ 3 ]`
array_union (x, y)	array	Restituisce una matrice degli elementi nell'unione di x e y, senza duplicati. `SELECT array_union(ARRAY[1,2,3], ARRAY[3,4,5])` Risultato di esempio: `[ 1,2,3,4,5 ]`

Funzione	Tipo di dati di output	Descrizione
array_except (x, y)	array	Restituisce una matrice di elementi in x ma non in y, senza duplicati. `SELECT array_except(ARRAY[1,2,3], ARRAY[3,4,5])` Risultato di esempio: `[ 1,2 ]`
array_join (x, delimiter, null_replacement)	varchar	Concatena gli elementi dell'array specificato utilizzando il delimitatore e una stringa opzionale per sostituire i valori null. `SELECT array_join(ARRAY[1,2,3], ';', '')` Risultato di esempio: `1;2;3`
array_max (x)	come gli elementi dell'array	Restituisce il valore massimo dell'array di input. `SELECT array_max(ARRAY[1,2,3])` Risultato di esempio: `3`

Funzione	Tipo di dati di output	Descrizione
array_min (x)	come gli elementi dell'array	Restituisce il valore minimo dell'array di input. `SELECT array_min (ARRAY[1,2,3])` Risultato di esempio: 1
array_position (x, element)	bigint	Restituisce la posizione della prima occorrenza dell'elem ento nell'array x (o 0 se non viene trovata). `SELECT array_pos ition(ARRAY[3,4,5,9], 5)` Risultato di esempio: 3
array_remove (x, element)	array	Rimuove tutti gli elementi che sono uguali all'elemento dall'array x. `SELECT array_rem ove(ARRAY[3,4,5,9], 4)` Risultato di esempio: `[ 3,5,9 ]`

Funzione	Tipo di dati di output	Descrizione
array_sort (x)	array	Ordina e restituisce l'array x. Gli elementi di x devono essere ordinabili. Gli elementi nulli verranno posizionati alla fine dell'array restituito. `SELECT array_sort(ARRAY[6,8,2,9,3])` Risultato di esempio: `[ 2,3,6,8,9 ]`
arrays_overlap (x, y)	booleano	Verifica se gli array x e y hanno elementi non nulli in comune. Restituisce null se non ci sono elementi non nulli in comune ma entrambi gli array contengono null. `SELECT arrays_overlap(ARRAY[6,8,2,9,3], ARRAY[6,8])` Risultato di esempio: `true`
cardinalità (x)	bigint	Restituisce la dimensione dell'array x. `SELECT cardinality(ARRAY[6,8,2,9,3])` Risultato di esempio: 5

Funzione	Tipo di dati di output	Descrizione
concat (matrice1, matrice2,..., matriceN)	array	Concatena gli array array1, array2,..., arrayN. `SELECT concat(ARRAY[6,8,2,9,3], ARRAY[11,32], ARRAY[6,8,2,0,14])` Risultato di esempio: `[ 6,8,2,9,3,11,32,6,8,2,0,14 ]`
element_at (array (E), indice)	E	Restituisce un elemento dell'array in un determinato indice. Se index < 0, element_at accede agli elementi dall'ultimo al primo. `SELECT element_at(ARRAY[6,8,2,9,3], 1)` Risultato di esempio: 6
repeat (elemento, conteggio)	array	Ripeti l'elemento per contare i tempi. `SELECT repeat(1, 3)` Risultato di esempio: `[ 1,1,1 ]`

Funzione	Tipo di dati di output	Descrizione

inversa (x)	array	Restituisce un array che ha l'ordine inverso dell'array x.
		`SELECT reverse(ARRAY[6,8,2,9,3])`
		Risultato di esempio:
		`[ 3,9,2,8,6 ]`

sequenza (inizio, arresto)	matrice (bigint)	Genera una sequenza di numeri interi dall'inizio alla fine, incrementandola di 1 se start è minore o uguale a stop, altrimenti -1.
		`SELECT sequence(3, 8)`
		Risultato di esempio:
		`[ 3,4,5,6,7,8 ]`

sequenza (inizio, arresto, fase)	matrice (bigint)	Genera una sequenza di numeri interi dall'inizio alla fine, incrementandola passo dopo passo.
		`SELECT sequence(3, 15, 2)`
		Risultato di esempio:
		`[ 3,5,7,9,11,13,15 ]`

Funzione	Tipo di dati di output	Descrizione

sequenza (inizio, arresto)	array (timestamp)	Genera una sequenza di timestamp dalla data di inizio alla data di fine, con un incremento di 1 giorno.
		
		`SELECT sequence( '2023-04-02 19:26:12.941000000', '2023-04-06 19:26:12.941000000', 1d)`
		
		Risultato di esempio:
		`[ 2023-04-02 19:26:12.941000000 ,2023-04-03 19:26:12.941000000 ,2023-04-04 19:26:12.941000000 ,2023-04-05 19:26:12.941000000 ,2023-04-06 19:26:12.941000000 ]`

Funzione	Tipo di dati di output	Descrizione

sequenza (inizio, arresto, fase)	matrice (timestamp)	Genera una sequenza di timestamp dall'inizio alla fine, incrementandola passo dopo passo. Il tipo di dati del passaggio è intervallo.
		`SELECT sequence( '2023-04-02 19:26:12.941000000', '2023-04-10 19:26:12.941000000', 2d)`
		Risultato di esempio:
		`[ 2023-04-02 19:26:12.941000000 ,2023-04-04 19:26:12.941000000 ,2023-04-06 19:26:12.941000000 ,2023-04-08 19:26:12.941000000 ,2023-04-10 19:26:12.941000000 ]`

shuffle (x)	array	Genera una permutazione casuale dell'array dato x.
		`SELECT shuffle(ARRAY[6,8,2,9,3])`
		Risultato di esempio:
		`[ 6,3,2,9,8 ]`

Funzione	Tipo di dati di output	Descrizione
slice (x, start, length)	array	Array di sottoinsiemi x a partire dall'indice start (o dalla fine se start è negativo) con una lunghezza di lunghezza. `SELECT slice(ARRAY[6,8,2,9,3], 1, 3)` Risultato di esempio: `[ 6,8,2 ]`
zip (array1, array2 [,...])	matrice (riga)	Unisce gli array dati, per elemento, in un unico array di righe. Se gli argomenti hanno una lunghezza non uniforme, i valori mancanti vengono riempiti con. NULL `SELECT zip(ARRAY[6,8,2,9,3], ARRAY[15, 24])` Risultato di esempio: `[ ( 6, 15 ),( 8, 24 ),( 2, - ),( 9, - ),( 3, - ) ]`

Funzioni bit per bit

Timestream for LiveAnalytics supporta le seguenti funzioni bit per bit.

Funzione	Tipo di dati di output	Descrizione
bit_count (bigint, bigint)	bigint (complemento a due)	Restituisce il conteggio dei bit nel primo parametro bigint

Funzione	Tipo di dati di output	Descrizione

		dove il secondo parametro è un numero intero con segno di bit come 8 o 64.
		`SELECT bit_count(19, 8)`
		Risultato di esempio: `3`
		`SELECT bit_count(19, 2)`
		Risultato di esempio: `Number must be represent able with the bits specified. 19 can not be represented with 2 bits`

bitwise_and (bigint, bigint)	bigint (complemento a due)	Restituisce i parametri bigint bit AND per bit. `SELECT bitwise_and(12, 7)` Risultato di esempio: `4`
bitwise_not (bigint)	bigint (complemento a due)	Restituisce il valore bit per bit NOT del parametro bigint. `SELECT bitwise_not(12)` Risultato di esempio: `-13`

Funzione	Tipo di dati di output	Descrizione
bitwise_or (bigint, bigint)	bigint (complemento a due)	Restituisce l'OR bit per bit dei parametri bigint. `SELECT bitwise_or(12, 7)` Risultato di esempio: 15
bitwise_xor (bigint, bigint)	bigint (complemento a due)	Restituisce i parametri bigint bit XOR per bit. `SELECT bitwise_xor(12, 7)` Risultato di esempio: 11

Funzioni di espressioni regolari

Le funzioni di espressione regolare in Timestream LiveAnalytics supportano la sintassi del pattern Java. Timestream for LiveAnalytics supporta le seguenti funzioni di espressione regolare.

Funzione	Tipo di dati di output	Descrizione
regexp_extract_all (stringa, modello)	array(varchar)	Restituisce la o le sottostrinche corrispondenti al modello di espressione regolare in string. `SELECT regexp_extract_all('example expect complex', 'ex\w')` Risultato di esempio: `[ exa,exp ]`

Funzione	Tipo di dati di output	Descrizione
regexp_extract_all (stringa, pattern, gruppo)	array(varchar)	Trova tutte le occorrenze del modello di espressione regolare nella stringa e restituisce il gruppo numerico del gruppo di acquisizione. `SELECT regexp_extract_all('example expect complex', '(ex)(\w)', 2)` Risultato di esempio: `[ a,p ]`
regexp_extract (stringa, pattern)	varchar	Restituisce la prima sottostringa corrispondente al modello di espressione regolare in string. `SELECT regexp_extract('example expect', 'ex\w')` Risultato di esempio: `exa`

Funzione	Tipo di dati di output	Descrizione
regexp_extract (stringa, pattern, group)	varchar	Trova la prima occorrenza del modello di espressione regolare nella stringa e restituisce il gruppo numerico del gruppo di acquisizione. `SELECT regexp_extract('example expect', '(ex)(\w)', 2)` Risultato di esempio: a
regexp_like (stringa, pattern)	booleano	Valuta il modello di espressione regolare e determina se è contenuto all'interno di una stringa. Questa funzione è simile all'LIKEoperatore, tranne per il fatto che il pattern deve essere contenuto solo all'interno di una stringa, anziché dover corrispondere a tutta la stringa. In altre parole, esegue un'operazione di contenimento anziché un'operazione di abbinamento. È possibile abbinare l'intera stringa ancorando il pattern utilizzando ^ e $. `SELECT regexp_like('example', 'ex')` Risultato di esempio: `true`

Funzione	Tipo di dati di output	Descrizione
regexp_replace (stringa, pattern)	varchar	Rimuove ogni istanza della sottostringa corrispondente al modello di espressione regolare dalla stringa. `SELECT regexp_replace('example expect', 'expect')` Risultato di esempio: `example`
regexp_replace (stringa, pattern, sostituzione)	varchar	Sostituisce ogni istanza della sottostringa corrispondente al pattern regex nella stringa con una sostituzione. È possibile fare riferimento ai gruppi di acquisizione in sostituzione utilizzando $g per un gruppo numerato o $ {name} per un gruppo denominato. Un simbolo del dollaro ($) può essere incluso nella sostituzione facendone evaporare con una barra rovesciata (\ $). `SELECT regexp_replace('example expect', 'expect', 'surprise')` Risultato di esempio: `example surprise`

Funzione	Tipo di dati di output	Descrizione
regexp_replace (stringa, pattern, funzione)	varchar	Sostituisce ogni istanza della sottostringa corrispondente al modello di espressione regolare nella stringa utilizzando la funzione. La funzione di espressione lambda viene richiamata per ogni corrispondenza con i gruppi di acquisizione passati come array. L'acquisizione dei numeri di gruppo inizia da uno; non esiste un gruppo per l'intera corrispondenza (se necessario, racchiudi l'intera espressione tra parentesi). `SELECT regexp_replace('example', '(\w)', x -> upper(x[1]))` Risultato di esempio: `EXAMPLE`
regexp_split (stringa, pattern)	array(varchar)	Divide la stringa usando il modello di espressione regolare e restituisce un array. Le stringhe vuote finali vengono conservate. `SELECT regexp_split('example', 'x')` Risultato di esempio: `[ e,ample ]`

Operatori data/ora

> **Note**
>
> Timestream for non LiveAnalytics supporta valori temporali negativi. Qualsiasi operazione che genera un tempo negativo genera un errore.

Timestream for LiveAnalytics supporta le seguenti operazioni su `timestampsdates`, e. `intervals`

Operatore	Descrizione
+	Addizione
-	Sottrazione

Argomenti

Operazioni

Il tipo di risultato di un'operazione si basa sugli operandi. È `3s` possibile utilizzare valori letterali a intervalli come `1day` e.

```
SELECT date '2022-05-21' + interval '2' day
```

```
SELECT date '2022-05-21' + 2d
```

```
SELECT date '2022-05-21' + 2day
```

Esempio di risultato per ciascuno: `2022-05-23`

Le unità di intervallo includonosecond,`minute`,`hour`,`day`, `weekmonth`, `eyear`. Ma in alcuni casi non tutte sono applicabili. Ad esempio, secondi, minuti e ore non possono essere aggiunti o sottratti da una data.

```
SELECT interval '4' year + interval '2' month
```

Risultato di esempio: `4-2`

```
SELECT typeof(interval '4' year + interval '2' month)
```

Risultato di esempio: `interval year to month`

Il tipo di risultato delle operazioni a intervalli può essere `'interval year to month'` o `'interval day to second'` dipende dagli operandi. Gli intervalli possono essere aggiunti o sottratti da e. `dates` `timestamps` Ma un `date` o `timestamp` non può essere aggiunto o sottratto da un o. `date` `timestamp` Per trovare intervalli o durate correlati a date o timestamp, vedere e funzioni correlate in. `date_diff` Intervallo e durata

Addizione

Example

```
SELECT date '2022-05-21' + interval '2' day
```

Risultato di esempio: `2022-05-23`

Example

```
SELECT typeof(date '2022-05-21' + interval '2' day)
```

Risultato di esempio: `date`

Example

```
SELECT interval '2' year + interval '4' month
```

Risultato di esempio: `2-4`

Example

```
SELECT typeof(interval '2' year + interval '4' month)
```

Risultato di esempio: `interval year to month`

Sottrazione

Example

```
SELECT timestamp '2022-06-17 01:00' - interval '7' hour
```

Risultato di esempio: `2022-06-16 18:00:00.000000000`

Example

```
SELECT typeof(timestamp '2022-06-17 01:00' - interval '7' hour)
```

Risultato di esempio: `timestamp`

Example

```
SELECT interval '6' day - interval '4' hour
```

Risultato di esempio: `5 20:00:00.000000000`

Example

```
SELECT typeof(interval '6' day - interval '4' hour)
```

Risultato di esempio: `interval day to second`

Funzioni data/ora

> **Note**
>
> Timestream for non LiveAnalytics supporta valori temporali negativi. Qualsiasi operazione che genera un tempo negativo genera un errore.

Timestream for LiveAnalytics utilizza il UTC fuso orario per data e ora. Timestream supporta le seguenti funzioni per data e ora.

Argomenti

Generale e conversione

Timestream for LiveAnalytics supporta le seguenti funzioni generali e di conversione per data e ora.

Funzione	Tipo di dati di output	Descrizione
data_corrente	data	Restituisce la data corrente in. UTC Nessuna parentesi utilizzata. `SELECT current_date` Risultato di esempio: `2022-07-07` **Note** Anche questa è una parola chiave riservata . Per un elenco di parole chiave riservate , vedereParole chiave riservate.
tempo_corrente	time	Restituisce l'ora corrente in. UTC Nessuna parentesi utilizzata.

Funzione	Tipo di dati di output	Descrizione
		`SELECT current_time` Risultato di esempio: `17:41:52.827000000` **Note** Anche questa è una parola chiave riservata . Per un elenco di parole chiave riservate , vedereParole chiave riservate.
current_timestamp o now ()	timestamp	Restituisce il timestamp corrente in. UTC `SELECT current_t imestamp` Risultato di esempio: `2022-07-07 17:42:32. 939000000` **Note** Anche questa è una parola chiave riservata . Per un elenco di parole chiave riservate , vedereParole chiave riservate.

Funzione	Tipo di dati di output	Descrizione
current_timezone ()	varchar Il valore sarà '.' UTC	Timestream utilizza il UTC fuso orario per data e ora. `SELECT current_timezone()` Risultato di esempio: UTC
data (varchar (x)), data (timestamp)	data	`SELECT date(TIMESTAMP '2022-07-07 17:44:43.771000000')` Risultato di esempio: `2022-07-07`
last_day_of_month (timestamp), last_day_of_month (data)	data	`SELECT last_day_of_month(TIMESTAMP '2022-07-07 17:44:43.771000000')` Risultato di esempio: `2022-07-31`
from_iso8601_timestamp (stringa)	timestamp	Analizza il timestamp 8601 nel formato timestamp interno. ISO `SELECT from_iso8601_timestamp('2022-06-17T08:04:05.000000000+05:00')` Risultato di esempio: `2022-06-17 03:04:05.000000000`

Funzione	Tipo di dati di output	Descrizione
from_iso8601_date (stringa)	data	Analizza la stringa della data ISO 8601 nel formato timestamp interno per le 00:00:00 della data specifica ta. UTC `SELECT from_iso8601_date('2022-07-17')` Risultato di esempio: `2022-07-17`
to_iso8601 (timestamp), to_iso8601 (data)	varchar	Restituisce una ISO stringa formattata 8601 per l'input. `SELECT to_iso8601(from_iso8601_date('2022-06-17'))` Risultato di esempio: `2022-06-17`
from_milliseconds (bigint)	timestamp	`SELECT from_milliseconds(1)` Risultato di esempio: `1970-01-01 00:00:00.001000000`

Funzione	Tipo di dati di output	Descrizione
from_nanoseconds (bigint)	timestamp	`select from_nanoseconds(300000001)` Risultato di esempio: `1970-01-01 00:00:00.300000001`
from_unixtime (double)	timestamp	Restituisce un timestamp che corrisponde all'unixtime fornito. `SELECT from_unixtime(1)` Risultato di esempio: `1970-01-01 00:00:01.000000000`
ora locale	time	Restituisce l'ora corrente inUTC. Nessuna parentesi utilizzata. `SELECT localtime` Risultato di esempio: `17:58:22.654000000` **Note** Anche questa è una parola chiave riservata . Per un elenco di parole chiave riservate , vedereParole chiave riservate.

Funzione	Tipo di dati di output	Descrizione
timestamp locale	timestamp	Restituisce il timestamp corrente in. UTC Nessuna parentesi utilizzata. `SELECT localtimestamp` Risultato di esempio: `2022-07-07 17:59:04.368000000` **Note** Anche questa è una parola chiave riservata . Per un elenco di parole chiave riservate , vedereParole chiave riservate.

to_milliseconds (intervallo da giorno a secondo), to_millis econds (timestamp)	bigint	`SELECT to_millis econds(INTERVAL '2' DAY + INTERVAL '3' HOUR)`
		Risultato di esempio: `183600000`
		`SELECT to_millis econds(TIMESTAMP '2022-06-17 17:44:43.771000000')`
		Risultato di esempio: `1655487883771`

Funzione	Tipo di dati di output	Descrizione

to_nanoseconds (intervallo da giorno a secondo), to_nanoseconds (timestamp)	bigint	`SELECT to_nanoseconds(INTERVAL '2' DAY + INTERVAL '3' HOUR)`
		
		Risultato di esempio: `183600000000000`
		
		`SELECT to_nanoseconds(TIMESTAMP '2022-06-17 17:44:43.771000678')`
		
		Risultato di esempio: `1655487883771000678`

to_unixtime (timestamp)	double	Restituisce unixtime per il timestamp fornito. `SELECT to_unixtime('2022-06-17 17:44:43.771000000')` Risultato di esempio: `1.6554878837710001E9`

Funzione	Tipo di dati di output	Descrizione
date_trunc (unità, timestamp)	timestamp	Restituisce il timestamp troncato in unità, dove l'unità è uno dei [secondo, minuto, ora, giorno, settimana, mese, trimestre o anno]. `SELECT date_trun c('minute', TIMESTAMP '2022-06-17 17:44:43. 771000000')` Risultato di esempio: `2022-06-17 17:44:00. 000000000`

Intervallo e durata

Timestream for LiveAnalytics supporta le seguenti funzioni di intervallo e durata per data e ora.

Funzione	Tipo di dati di output	Descrizione
date_add (unit, bigint, date), date_add (unit, bigint, time), date_add (varchar (x), bigint, timestamp)	timestamp	Aggiunge un insieme di unità, dove l'unità è una tra [secondo, minuto, ora, giorno, settimana, mese, trimestre o anno]. `SELECT date_add('hour', 9, TIMESTAMP '2022-06-17 00:00:00')` Risultato di esempio: `2022-06-17 09:00:00. 000000000`

Funzione	Tipo di dati di output	Descrizione
date_diff (unità, data, data), date_diff (unità, ora, ora), date_diff (unità, timestamp, timestamp)	bigint	Restituisce una differenza, dove l'unità è uno dei [secondo, minuto, ora, giorno, settimana, mese, trimestre o anno]. `SELECT date_diff('day', DATE '2020-03-01', DATE '2020-03-02')` Risultato di esempio: 1

parse_duration (stringa)	intervallo	Analizza la stringa di input per restituire un equivalente. `interval`
		`SELECT parse_duration('42.8ms')`
		Risultato di esempio: `0 00:00:00.042800000`
		`SELECT typeof(parse_duration('42.8ms'))`
		Risultato di esempio: `interval day to second`

Funzione	Tipo di dati di output	Descrizione
bin (timestamp, intervallo)	timestamp	Arrotonda il valore intero del `timestamp` parametro al multiplo più vicino del valore intero del `interval` parametro. Il significato di questo valore restituito potrebbe non essere ovvio. Viene calcolato utilizzando l'aritmetica dei numeri interi prima dividendo il numero intero del timestamp per il numero intero dell'intervallo e quindi moltiplicando il risultato per il numero intero dell'intervallo. Tenendo presente che un timestamp specifica un UTC punto nel tempo come un numero di frazioni di secondo trascorse dall'epoca (1° gennaio 1970), il valore restituito raramente si allinea alle unità del calendario. POSIX Ad esempio, se si specifica un intervallo di 30 giorni, tutti i giorni trascorsi dall'epoca vengono divisi in incrementi di 30 giorni e viene restituito l'inizio dell'incremento di 30 giorni più recente, che non ha alcuna relazione con i mesi del calendario.

Funzione	Tipo di dati di output	Descrizione

		Ecco alcuni esempi:
		`bin(TIMESTAMP '2022-06-17 10:15:20', 5m) ==> 2022-06-17 10:15:00.000000000`
		`bin(TIMESTAMP '2022-06-17 10:15:20', 1d) ==> 2022-06-17 00:00:00.000000000`
		`bin(TIMESTAMP '2022-06-17 10:15:20', 10day) ==> 2022-06-17 00:00:00.000000000`
		`bin(TIMESTAMP '2022-06-17 10:15:20', 30day) ==> 2022-05-28 00:00:00.000000000`

fa (intervallo)	timestamp	Restituisce il valore corrispondente a `interval` current_timestamp. `SELECT ago(1d)` Risultato di esempio: `2022-07-06 21:08:53.245000000`

Funzione	Tipo di dati di output	Descrizione
valori letterali a intervalli come 1h, 1d e 30m	intervallo	I valori letterali a intervalli sono utili per parse_duration (string). Ad esempio, `1d` è equivalente a `parse_duration('1d')` . Ciò consente l'uso dei valori letterali ovunque venga utilizzato un intervallo. Ad esempio `ago(1d)` e `bin(<timestamp> , 1m)`.

Alcuni valori letterali a intervalli fungono da abbreviazione di parse_duration. Ad esempio,,`parse_duration('1day')`, e ogni ritorno in cui `1day` si `parse_duration('1d')` trova il tipo`1d`. `1 00:00:00.000000000` `interval day to second` Lo spazio è consentito nel formato fornito a`parse_duration`. Ad esempio, restituisce `parse_duration('1day')` anche`00:00:00.000000000`. Ma non `1 day` è un intervallo letterale.

Le unità relative a `interval day to second` sono ns, nanosecond, us, microsecond, ms, millisecond, s, second, m, minute, h, hour, d e day.

C'è anche. `interval year to month` Le unità relative all'intervallo da un anno all'altro sono y, anno e mese. Ad esempio, `SELECT 1year` restituisce`1-0`. `SELECT 12month`restituisce anche`1-0`. `SELECT 8month`ritorna`0-8`.

Sebbene l'unità di `quarter` sia disponibile anche per alcune funzioni come `date_trunc` e`date_add`, non `quarter` è disponibile come parte di un intervallo letterale.

Formattazione e analisi

Timestream for LiveAnalytics supporta le seguenti funzioni di formattazione e analisi per data e ora.

Funzione	Tipo di dati di output	Descrizione
date_format (timestamp, varchar (x))	varchar	Per ulteriori informazioni sugli specificatori di formato usati da questa funzione, vedere

Funzione	Tipo di dati di output	Descrizione
		# https://trino.io/docs/current/functions/datetime.html mysql-date-functions `SELECT date_format(TIMESTAMP '2019-10-20 10:20:20', '%Y-%m-%d %H:%i:%s')` Risultato di esempio: `2019-10-20 10:20:20`
date_parse (varchar (x), varchar (y))	timestamp	Per ulteriori informazioni sugli specificatori di formato usati da questa funzione, vedere # https://trino.io/docs/current/functions/datetime.html mysql-date-functions `SELECT date_parse('2019-10-20 10:20:20', '%Y-%m-%d %H:%i:%s')` Risultato di esempio: `2019-10-20 10:20:20.000000000`

Funzione	Tipo di dati di output	Descrizione
format_datetime (timestamp, varchar (x))	varchar	Per ulteriori informazioni sulla stringa di formato utilizzata da questa funzione, vedere http://joda-time.sourceforge. net/apidocs/org/joda/time/format/DateTimeFormat.html `SELECT format_datetime(parse_datetime('1968-01-13 12', 'yyyy-MM-dd HH'), 'yyyy-MM-dd HH')` Risultato di esempio: `1968-01-13 12`
parse_datetime (varchar (x), varchar (y))	timestamp	Per ulteriori informazioni sulla stringa di formato utilizzata da questa funzione, vedere http://joda-time.sourceforge. net/apidocs/org/joda/time/format/DateTimeFormat.html `SELECT parse_datetime('2019-12-29 10:10 PST', 'uuuu-LL-dd HH:mm z')` Risultato di esempio: `2019-12-29 18:10:00.000000000`

Estrazione

Timestream for LiveAnalytics supporta le seguenti funzioni di estrazione per data e ora. La funzione di estrazione è la base per le restanti funzioni di praticità.

Funzione	Tipo di dati di output	Descrizione
estratto	bigint	Estrae un campo da un timestamp, dove field è uno dei [YEAR,,,, _OF_ QUARTER MONTHWEEK, _OF_DAY,, DAY _OF_MONTH ,, DAY _OF_WEEK,DOW,,, DAY or]. YEAR DOY YEAR WEEK YOW HOUR MINUTE SECOND `SELECT extract(YEAR FROM '2019-10-12 23:10:34.000000000')` Risultato di esempio: 2019
giorno (timestamp), giorno (data), giorno (intervallo da giorno a secondo)	bigint	`SELECT day('2019-10-12 23:10:34.000000000')` Risultato di esempio: 12
day_of_month (timestam p), day_of_month (data), day_of_month (intervallo da giorno a secondo)	bigint	`SELECT day_of_mo nth('2019-10-12 23:10:34.000000000')` Risultato di esempio: 12
day_of_week (timestamp), day_of_week (data)	bigint	`SELECT day_of_we ek('2019-10-12 23:10:34.000000000')` Risultato di esempio: 6

Funzione	Tipo di dati di output	Descrizione
day_of_year (timestamp), day_of_year (data)	bigint	`SELECT day_of_year('2019-10-12 23:10:34.000000000')` Risultato di esempio: 285
dow (timestamp), dow (data)	bigint	Alias per day_of_week
doy (timestamp), doy (data)	bigint	Alias per day_of_year
ora (timestamp), ora (ora), ora (intervallo da giorno a secondo)	bigint	`SELECT hour('2019-10-12 23:10:34.000000000')` Risultato di esempio: 23
millisecondo (timestamp), millisecondo (ora), millisecondo (intervallo da giorno a secondo)	bigint	`SELECT millisecond('2019-10-12 23:10:34.000000000')` Risultato di esempio: 0
minuto (timestamp), minuto (ora), minuto (intervallo da giorno a secondo)	bigint	`SELECT minute('2019-10-12 23:10:34.000000000')` Risultato di esempio: 10
mese (timestamp), mese (data), mese (intervallo da anno a mese)	bigint	`SELECT month('2019-10-12 23:10:34.000000000')` Risultato di esempio: 10

Funzione	Tipo di dati di output	Descrizione
nanosecondo (timestamp), nanosecondo (ora), nanosecondo (intervallo da giorno a secondo)	bigint	`SELECT nanosecond(current_timestamp)` Risultato di esempio: `162000000`
trimestre (timestamp), trimestre (data)	bigint	`SELECT quarter('2019-10-12 23:10:34.000000000')` Risultato di esempio: `4`
secondo (timestamp), secondo (ora), secondo (intervallo da giorno a secondo)	bigint	`SELECT second('2019-10-12 23:10:34.000000000')` Risultato di esempio: `34`
settimana (timestamp), settimana (data)	bigint	`SELECT week('2019-10-12 23:10:34.000000000')` Risultato di esempio: `41`
settimana_del_anno (timestamp), settimana_del_anno (data)	bigint	Alias per settimana
anno (timestamp), anno (data), anno (intervallo da anno a mese)	bigint	`SELECT year('2019-10-12 23:10:34.000000000')` Risultato di esempio: `2019`

Funzione	Tipo di dati di output	Descrizione
anno_di_settimana (timestamp), anno_del_settimana (data)	bigint	`SELECT year_of_week('2019-10-12 23:10:34.000000000')` Risultato di esempio: 2019
yow (timestamp), yow (data)	bigint	Alias per year_of_week

Funzioni di aggregazione

Timestream for supporta le seguenti funzioni aggregate. LiveAnalytics

Funzione	Tipo di dati di output	Descrizione
arbitrario (x)	[uguale all'input]	Restituisce un valore arbitrario non nullo di x, se ne esiste uno. `SELECT arbitrary(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: 1
array_agg (x)	array< [uguale all'input]	Restituisce un array creato dagli elementi x di input. `SELECT array_agg(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: `[ 1,2,3,4 ]`

Funzione	Tipo di dati di output	Descrizione
avg (x)	double	Restituisce la media (media aritmetica) di tutti i valori di input. `SELECT avg(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: `2.5`
bool_and (boolean) ogni (boolean)	booleano	Restituisce TRUE se ogni valore di input è, altrimenti. TRUE FALSE `SELECT bool_and(t.c) FROM (VALUES true, true, false, true) AS t(c)` Risultato di esempio: `false`
bool_or (booleano)	booleano	Restituisce TRUE se un valore di input è, altrimenti. TRUE FALSE `SELECT bool_or(t.c) FROM (VALUES true, true, false, true) AS t(c)` Risultato di esempio: `true`

Funzione	Tipo di dati di output	Descrizione
conteggio (*) conteggio (x)	bigint	count (*) restituisce il numero di righe di input. count (x) restituisce il numero di valori di input non nulli. `SELECT count(t.c) FROM (VALUES true, true, false, true) AS t(c)` Risultato di esempio: 4
count_if (x)	bigint	Restituisce il numero di valori di TRUE input. `SELECT count_if(t.c) FROM (VALUES true, true, false, true) AS t(c)` Risultato di esempio: 3
geometric_mean (x)	double	Restituisce la media geometrica di tutti i valori di input. `SELECT geometric_mean(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: 2.213363839400643

Funzione	Tipo di dati di output	Descrizione
max_by (x, y)	[uguale a x]	Restituisce il valore di x associato al valore massimo di y su tutti i valori di input. `SELECT max_by(t.c1, t.c2) FROM (VALUES (('a', 1)), (('b', 2)), (('c', 3)), (('d', 4))) AS t(c1, c2)` Risultato di esempio: d
max_by (x, y, n)	matrice< [same as x] >	Restituisce n valori di x associati al valore n più grande di tutti i valori di input di y in ordine decrescente di y. `SELECT max_by(t.c1, t.c2, 2) FROM (VALUES (('a', 1)), (('b', 2)), (('c', 3)), (('d', 4))) AS t(c1, c2)` Risultato di esempio: `[ d,c ]`
min_by (x, y)	[uguale a x]	Restituisce il valore di x associato al valore minimo di y su tutti i valori di input. `SELECT min_by(t.c1, t.c2) FROM (VALUES (('a', 1)), (('b', 2)), (('c', 3)), (('d', 4))) AS t(c1, c2)` Risultato di esempio: a

Funzione	Tipo di dati di output	Descrizione
min_by (x, y, n)	matrice< [same as x] >	Restituisce n valori di x associati al valore n più piccolo di tutti i valori di input di y in ordine crescente di y. `SELECT min_by(t.c1, t.c2, 2) FROM (VALUES (('a', 1)), (('b', 2)), (('c', 3)), (('d', 4))) AS t(c1, c2)` Risultato di esempio: `[ a,b ]`
max (x)	[uguale all'input]	Restituisce il valore massimo di tutti i valori di input. `SELECT max(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: 4
max (x, n)	matrice< [same as x] >	Restituisce n valori più grandi di tutti i valori di input di x. `SELECT max(t.c, 2) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: `[ 4,3 ]`

Funzione	Tipo di dati di output	Descrizione
min (x)	[uguale all'input]	Restituisce il valore minimo di tutti i valori di input. `SELECT min(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: 1
min (x, n)	matrice< [same as x] >	Restituisce n valori più piccoli di tutti i valori di input di x. `SELECT min(t.c, 2) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: `[ 1,2 ]`
somma (x)	[uguale all'input]	Restituisce la somma di tutti i valori di input. `SELECT sum(t.c) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: `10`

Funzione	Tipo di dati di output	Descrizione
bitwise_and_agg (x)	bigint	Restituisce il bit per bit di tutti i valori di input nella rappresentazione AND del complemento a 2 secondi. `SELECT bitwise_and_agg(t.c) FROM (VALUES 1, -3) AS t(c)` Risultato di esempio: 1
bitwise_or_agg (x)	bigint	Restituisce l'OR bit per bit di tutti i valori di input nella rappresentazione del complemento a 2 secondi. `SELECT bitwise_or_agg(t.c) FROM (VALUES 1, -3) AS t(c)` Risultato di esempio: -3

Funzione	Tipo di dati di output	Descrizione
approx_distinct (x)	bigint	Restituisce il numero approssimativo di valori di input distinti. Questa funzione fornisce un'approssimazione di count (DISTINCTx). Viene restituito zero se tutti i valori di input sono nulli. Questa funzione dovrebbe produrre un errore standard del 2,3%, che è la deviazione standard della distribuzione degli errori (approssimativamente normale) su tutti i set possibili . Non garantisce un limite massimo dell'errore per alcun set di input specifico. `SELECT approx_distinct(t.c) FROM (VALUES 1, 2, 3, 4, 8) AS t(c)` Risultato di esempio: 5

Funzione	Tipo di dati di output	Descrizione
approx_distinct (x, e)	bigint	Restituisce il numero approssimativo di valori di input distinti. Questa funzione fornisce un'approssimazione di count (DISTINCTx). Viene restituito zero se tutti i valori di input sono nulli. Questa funzione dovrebbe produrre un errore standard non superiore a e, che è la deviazione standard della distribuz ione degli errori (approssi mativamente normale) su tutti i set possibili. Non garantisce un limite massimo dell'errore per alcun set di input specifico . L'attuale implementazione di questa funzione richiede che e sia compreso nell'intervallo [0,0040625, 0,26000]. `SELECT approx_di stinct(t.c, 0.2) FROM (VALUES 1, 2, 3, 4, 8) AS t(c)` Risultato di esempio: 5

Funzione	Tipo di dati di output	Descrizione
approx_percentile (x, percentile)	[uguale a x]	Restituisce il percentile approssimativo per tutti i valori di input di x alla percentuale specificata. Il valore della percentuale deve essere compreso tra zero e uno e deve essere costante per tutte le righe di input. `SELECT approx_percentile(t.c, 0.4) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: 2
approx_percentile (x, percentages)	matrice< [same as x] >	Restituisce il percentile approssimativo per tutti i valori di input di x per ciascuna delle percentuali specifiche. Ogni elemento dell'array delle percentuali deve essere compreso tra zero e uno e l'array deve essere costante per tutte le righe di input. `SELECT approx_percentile(t.c, ARRAY[0.1, 0.8, 0.8]) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: `[ 1,4,4 ]`

Funzione	Tipo di dati di output	Descrizione
approx_percentile (x, w, percentile)	[uguale a x]	Restituisce il percentile pesato approssimativo per tutti i valori di input di x utilizzando il peso per articolo w alla percentua le p. Il peso deve essere un valore intero di almeno uno. È effettivamente un conteggio delle repliche per il valore x nel set di percentili. Il valore di p deve essere compreso tra zero e uno e deve essere costante per tutte le righe di input. `SELECT approx_pe rcentile(t.c, 1, 0.1) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: 1

Funzione	Tipo di dati di output	Descrizione
approx_percentile (x, w, percentuali)	matrice< [same as x] >	Restituisce il percentile pesato approssimativo per tutti i valori di input di x utilizzan do il peso per articolo w in ciascuna delle percentuali specificate nell'array. Il peso deve essere un valore intero di almeno uno. È effettivamente un conteggio delle repliche per il valore x nel set di percentili. Ogni elemento dell'array deve essere compreso tra zero e uno e l'array deve essere costante per tutte le righe di input. `SELECT approx_pe rcentile(t.c, 1, ARRAY[0.1, 0.8, 0.8]) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: `[ 1,4,4 ]`

Funzione	Tipo di dati di output	Descrizione
approx_percentile (x, w, percentuale, precisione)	[uguale a x]	Restituisce il percentile pesato approssimativo per tutti i valori di input di x utilizzando il peso per articolo w alla percentuale p, con un errore di precisione massimo. Il peso deve essere un valore intero di almeno uno. È effettivamente un conteggio delle repliche per il valore x nel set di percentili. Il valore di p deve essere compreso tra zero e uno e deve essere costante per tutte le righe di input. La precisione deve essere un valore maggiore di zero e minore di uno e deve essere costante per tutte le righe di input. `SELECT approx_percentile(t.c, 1, 0.1, 0.5) FROM (VALUES 1, 2, 3, 4) AS t(c)` Risultato di esempio: 1

Funzione	Tipo di dati di output	Descrizione
corr (y, x)	double	Restituisce il coefficiente di correlazione dei valori di input. `SELECT corr(t.c1, t.c2) FROM (VALUES ((1, 1)), ((2, 2)), ((3, 3)), ((4, 4))) AS t(c1, c2)` Risultato di esempio: `1.0`
covar_pop (y, x)	double	Restituisce la covarianza della popolazione dei valori di input. `SELECT covar_pop(t.c1, t.c2) FROM (VALUES ((1, 1)), ((2, 2)), ((3, 3)), ((4, 4))) AS t(c1, c2)` Risultato di esempio: `1.25`
covar_samp (y, x)	double	Restituisce la covarianza campionaria dei valori di input. `SELECT covar_samp(t.c1, t.c2) FROM (VALUES ((1, 1)), ((2, 2)), ((3, 3)), ((4, 4))) AS t(c1, c2)` Risultato di esempio: `1.6666666666666667`

Funzione	Tipo di dati di output	Descrizione
regr_intercept (y, x)	double	Restituisce l'intercetta di regressione lineare dei valori di ingresso. y è il valore dipendente. x è il valore indipendente. `SELECT regr_intercept(t.c1, t.c2) FROM (VALUES ((1, 1)), ((2, 2)), ((3, 3)), ((4, 4))) AS t(c1, c2)` Risultato di esempio: `0.0`
regr_slope (y, x)	double	Restituisce la pendenza di regressione lineare dei valori di ingresso. y è il valore dipendente. x è il valore indipendente. `SELECT regr_slope(t.c1, t.c2) FROM (VALUES ((1, 1)), ((2, 2)), ((3, 3)), ((4, 4))) AS t(c1, c2)` Risultato di esempio: `1.0`

Funzione	Tipo di dati di output	Descrizione
asimmetria (x)	double	Restituisce l'asimmetria di tutti i valori di input. `SELECT skewness(t.c1) FROM (VALUES 1, 2, 3, 4, 8) AS t(c1)` Risultato di esempio: `0.8978957037987335`
stddev_pop (x)	double	Restituisce la deviazione standard della popolazione di tutti i valori di input. `SELECT stddev_pop(t.c1) FROM (VALUES 1, 2, 3, 4, 8) AS t(c1)` Risultato di esempio: `2.4166091947189146`
stddev_samp (x) stddev (x)	double	Restituisce la deviazione standard del campione di tutti i valori di input. `SELECT stddev_samp(t.c1) FROM (VALUES 1, 2, 3, 4, 8) AS t(c1)` Risultato di esempio: `2.701851217221259`

Funzione	Tipo di dati di output	Descrizione
var_pop (x)	double	Restituisce la varianza della popolazione di tutti i valori di input. `SELECT var_pop(t.c1) FROM (VALUES 1, 2, 3, 4, 8) AS t(c1)` Risultato di esempio: `5.840000000000001`
var_samp (x) varianza (x)	double	Restituisce la varianza campionaria di tutti i valori di input. `SELECT var_samp(t.c1) FROM (VALUES 1, 2, 3, 4, 8) AS t(c1)` Risultato di esempio: `7.300000000000001`

Funzioni finestra

Le funzioni della finestra eseguono calcoli tra le righe del risultato della query. Vengono eseguite dopo la HAVING clausola ma prima della clausola ORDER BY. L'invocazione di una funzione window richiede una sintassi speciale che utilizza la OVER clausola per specificare la finestra. Una finestra ha tre componenti:

- La specifica della partizione, che separa le righe di input in partizioni diverse. Questo è analogo al modo in cui la clausola GROUP BY separa le righe in diversi gruppi per le funzioni aggregate.
- La specifica di ordinamento, che determina l'ordine in cui le righe di input verranno elaborate dalla funzione finestra.
- La cornice della finestra, che specifica una finestra scorrevole di righe che devono essere elaborate dalla funzione per una determinata riga. Se il frame non è specificato, il valore predefinito è

RANGE UNBOUNDEDPRECEDING, che è lo stesso di. RANGE BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW Questo frame contiene tutte le righe dall'inizio della partizione fino all'ultimo peer della riga corrente.

Tutte le funzioni aggregate possono essere utilizzate come funzioni di finestra aggiungendo la clausola. OVER La funzione di aggregazione viene calcolata per ogni riga sulle righe all'interno della cornice della finestra della riga corrente. Oltre alle funzioni di aggregazione, Timestream for LiveAnalytics supporta le seguenti funzioni di classificazione e valore.

Funzione	Tipo di dati di output	Descrizione
cume_dist ()	bigint	Restituisce la distribuzione cumulativa di un valore in un gruppo di valori. Il risultato è il numero di righe che precedono o coincidono con la riga nella partizione della finestra diviso per il numero totale di righe nella partizion e della finestra. Pertanto, tutti i valori di parità presenti nell'ordinamento verranno restituiti allo stesso valore di distribuzione.
dense_rank ()	bigint	Restituisce il rango di un valore in un gruppo di valori. È simile a rank (), tranne per il fatto che i valori di parità non producono spazi vuoti nella sequenza.
intile (n)	bigint	Divide le righe per ogni partizione di finestra in n bucket che vanno da 1 a un massimo n. I valori del bucket differiranno al massimo

Funzione	Tipo di dati di output	Descrizione
		di 1. Se il numero di righe nella partizione non si divide equamente nel numero di bucket, i valori rimanenti vengono distribuiti uno per bucket, a partire dal primo bucket.
percent_rank ()	double	Restituisce la classificazione percentuale di un valore in un gruppo di valori. Il risultato è (r - 1)/(n - 1) dove r è il rango () della riga e n è il numero totale di righe nella partizione della finestra.
rango ()	bigint	Restituisce il rango di un valore in un gruppo di valori. Il rango è uno più il numero di righe che precedono la riga che non sono uguali alla riga. Pertanto, i valori di parità nell'ordine produrranno lacune nella sequenza. La classific azione viene eseguita per ogni partizione di finestra.
numero_riga ()	bigint	Restituisce un numero sequenziale univoco per ogni riga, a partire da uno, in base all'ordine delle righe all'interno della partizione della finestra.

Funzione	Tipo di dati di output	Descrizione
primo_valore (x)	[uguale all'input]	Restituisce il primo valore della finestra. Questa funzione è limitata alla cornice della finestra. La funzione accetta un'espressione o un obiettivo come parametro.
last_value (x)	[uguale all'input]	Restituisce l'ultimo valore della finestra. Questa funzione è limitata alla cornice della finestra. La funzione accetta un'espressione o un obiettivo come parametro.
nth_value (x, offset)	[uguale all'input]	Restituisce il valore all'offse t specificato dall'inizio della finestra. Gli offset partono da 1. L'offset può essere qualsiasi espressione scalare. Se l'offset è nullo o maggiore del numero di valori nella finestra, viene restituito null. È un errore che l'offset sia zero o negativo. La funzione accetta un'espressione o un obiettivo come primo parametro.

Funzione	Tipo di dati di output	Descrizione
lead (x [, offset [, default_value]])	[uguale all'input]	Restituisce il valore nelle righe di offset dopo la riga corrente nella finestra. Gli offset iniziano da 0, che è la riga corrente. L'offset può essere qualsiasi espressione scalare. L'offset predefinito è 1. Se l'offset è nullo o maggiore della finestra, viene restituito il valore default_value o, se non è specificato, viene restituito null. La funzione accetta un'espressione o un obiettivo come primo parametro.
lag (x [, offset [, valore_predefinito]])	[uguale all'input]	Restituisce il valore nelle righe di offset prima della riga corrente nella finestra Offsets start at 0, che è la riga corrente. L'offset può essere qualsiasi espressione scalare. L'offset predefinito è 1. Se l'offset è nullo o maggiore della finestra, viene restituito il valore default_value o, se non è specificato, viene restituito null. La funzione accetta un'espressione o un obiettivo come primo parametro.

Query di esempio

Questa sezione include esempi di casi d'uso del linguaggio di interrogazione LiveAnalytics di Timestream for.

Argomenti

Interrogazioni semplici

Di seguito vengono ottenuti i 10 punti dati aggiunti più di recente per una tabella.

```
SELECT * FROM <database_name>.<table_name>
ORDER BY time DESC
LIMIT 10
```

Di seguito vengono ottenuti i 5 punti dati più vecchi per una misura specifica.

```
SELECT * FROM <database_name>.<table_name>
WHERE measure_name = '<measure_name>'
ORDER BY time ASC
LIMIT 5
```

Quanto segue funziona con timestamp di granularità in nanosecondi.

```
SELECT now() AS time_now
, now() - (INTERVAL '12' HOUR) AS twelve_hour_earlier -- Compatibility with ANSI SQL
, now() - 12h AS also_twelve_hour_earlier -- Convenient time interval literals
, ago(12h) AS twelve_hours_ago -- More convenience with time functionality
, bin(now(), 10m) AS time_binned -- Convenient time binning support
, ago(50ns) AS fifty_ns_ago -- Nanosecond support
, now() + (1h + 50ns) AS hour_fifty_ns_future
```

I valori di misura per i record con più misure sono identificati dal nome della colonna. I valori di misura per i record a misura singola sono identificati da`measure_value::<data_type>`, dove *`<data_type>`* è uno di`double`, `bigint`, `boolean`, o `varchar` come descritto in. Tipi di dati

supportati Per ulteriori informazioni su come vengono modellati i valori di misura, consulta Tabella singola e tabelle multiple.

Di seguito vengono recuperati i valori per una misura richiamata `speed` da record con più misure con un di. `measure_name` `IoTMulti-stats`

```
SELECT speed FROM <database_name>.<table_name> where measure_name = 'IoTMulti-stats'
```

Quanto segue recupera `double` i valori dai record a misura singola con un di. `measure_name` `load`

```
SELECT measure_value::double FROM <database_name>.<table_name> WHERE measure_name =
 'load'
```

Interrogazioni con funzioni di serie temporali

Argomenti

- Set di dati e interrogazioni di esempio

Set di dati e interrogazioni di esempio

È possibile utilizzare Timestream per LiveAnalytics comprendere e migliorare le prestazioni e la disponibilità dei servizi e delle applicazioni. Di seguito è riportato un esempio di tabella e delle query di esempio eseguite su tale tabella.

La tabella `ec2_metrics` memorizza i dati di telemetria, come l'CPUutilizzo e altre metriche delle istanze. EC2 È possibile visualizzare la tabella riportata di seguito.

Orario	Regione	az	Hostname (Nome host)	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:00.000 000000	us-east-1	us-east-1a	front-end 01	cpu_utilization	35.1	null
2019-12-04 19:00:00.	us-east-1	us-east-1a	front-end 01	memory_utilization	5.3	null

Orario	Regione	az	Hostname (Nome host)	measure_name	measure_value::double	measure_value::bigint
000 000000						
2019-12-04 19:00:00.000 000000	us-east-1	us-east-1a	front-end 01	network_bytes_in	null	1.500
2019-12-04 19:00:00.000 000000	us-east-1	us-east-1a	front-end 01	network_bytes_out	null	6.700
2019-12-04 19:00:00.000 000000	us-east-1	us-east-1b	front-end 02	cpu_utilization	38,5	null
2019-12-04 19:00:00.000 000000	us-east-1	us-east-1b	front-end 02	memory_utilization	58,4	null
2019-12-04 19:00:00.000 000000	us-east-1	us-east-1b	front-end 02	network_bytes_in	null	23.000
2019-12-04 19:00:00.000 000000	us-east-1	us-east-1b	front-end 02	network_bytes_out	null	12.000

Orario	Regione	az	Hostname (Nome host)	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:00.000000000	us-east-1	us-east-1c	front-end03	cpu_utilization	45.0	null
2019-12-04 19:00:00.000000000	us-east-1	us-east-1c	front-end03	memory_utilization	65,8	null
2019-12-04 19:00:00.000000000	us-east-1	us-east-1c	front-end03	network_bytes_in	null	15.000
2019-12-04 19:00:00.000000000	us-east-1	us-east-1c	front-end03	network_bytes_out	null	836.000
2019-12-04 19:00:05.000000000	us-east-1	us-east-1a	front-end01	cpu_utilization	55.2	null
2019-12-04 19:00:05.000000000	us-east-1	us-east-1a	front-end01	memory_utilization	75.0	null
2019-12-04 19:00:05.000000000	us-east-1	us-east-1a	front-end01	network_bytes_in	null	1.245

Orario	Regione	az	Hostname (Nome host)	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:05.000 000000	us-east-1	us-east-1a	front-end 01	network_bytes_out	null	68.432
2019-12-04 19:00:08000 000000	us-east-1	us-east-1b	front-end 02	cpu_utilization	65,6	null
2019-12-04 19:00:08000 000000	us-east-1	us-east-1b	front-end 02	memory_utilization	85,3	null
2019-12-04 19:00:08000 000000	us-east-1	us-east-1b	front-end 02	network_bytes_in	null	1.245
2019-12-04 19:00:08000 000000	us-east-1	us-east-1b	front-end 02	network_bytes_out	null	68.432
2019-12-04 19:00:20000 000000	us-east-1	us-east-1c	front-end 03	cpu_utilization	12.1	null
2019-12-04 19:00:20000 000000	us-east-1	us-east-1c	front-end 03	memory_utilization	32,0	null
2019-12-04 19:00:20000 000000	us-east-1	us-east-1c	front-end 03	network_bytes_in	null	1.400

Orario	Regione	az	Hostname (Nome host)	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:20000 000000	us-east-1	us-east-1c	front-end 03	network_bytes_out	null	345
2019-12-04 19:00:10.000 000000	us-east-1	us-east-1a	front-end 01	cpu_utilization	15.3	null
2019-12-04 19:00:10.000 000000	us-east-1	us-east-1a	front-end 01	memory_utilization	35,4	null
2019-12-04 19:00:10.000 000000	us-east-1	us-east-1a	front-end 01	network_bytes_in	null	23
2019-12-04 19:00:10.000 000000	us-east-1	us-east-1a	front-end 01	network_bytes_out	null	0
2019-12-04 19:00:16.000 000000	us-east-1	us-east-1b	front-end 02	cpu_utilization	44.0	null
2019-12-04 19:00:16.000 000000	us-east-1	us-east-1b	front-end 02	memory_utilization	64.2	null

Orario	Regione	az	Hostname (Nome host)	measure_name	measure_value::double	measure_value::bigint
2019-12-04 19:00:16000 000000	us-east-1	us-east-1b	front-end02	network_bytes_in	null	1.450
2019-12-04 19:00:16000 000000	us-east-1	us-east-1b	front-end02	network_bytes_out	null	200
2019-12-04 19:00:40.000 000000	us-east-1	us-east-1c	front-end03	cpu_utilization	6.4	null
2019-12-04 19:00:40,000 000000	us-east-1	us-east-1c	front-end03	memory_utilization	86,3	null
2019-12-04 19:00:40,000 000000	us-east-1	us-east-1c	front-end03	network_bytes_in	null	300
2019-12-04 19:00:40.000 000000	us-east-1	us-east-1c	front-end03	network_bytes_out	null	423

Trova l'CPUutilizzo medio, p90, p95 e p99 per un EC2 host specifico nelle ultime 2 ore:

```
SELECT region, az, hostname, BIN(time, 15s) AS binned_timestamp,
    ROUND(AVG(measure_value::double), 2) AS avg_cpu_utilization,
    ROUND(APPROX_PERCENTILE(measure_value::double, 0.9), 2) AS p90_cpu_utilization,
    ROUND(APPROX_PERCENTILE(measure_value::double, 0.95), 2) AS p95_cpu_utilization,
```

```
    ROUND(APPROX_PERCENTILE(measure_value::double, 0.99), 2) AS p99_cpu_utilization
FROM "sampleDB".DevOps
WHERE measure_name = 'cpu_utilization'
    AND hostname = 'host-Hovjv'
    AND time > ago(2h)
GROUP BY region, hostname, az, BIN(time, 15s)
ORDER BY binned_timestamp ASC
```

Identifica EC2 gli host con un CPU utilizzo superiore del 10% o più rispetto all'CPUutilizzo medio dell'intera flotta nelle ultime 2 ore:

```
WITH avg_fleet_utilization AS (
    SELECT COUNT(DISTINCT hostname) AS total_host_count, AVG(measure_value::double) AS
 fleet_avg_cpu_utilization
    FROM "sampleDB".DevOps
    WHERE measure_name = 'cpu_utilization'
        AND time > ago(2h)
), avg_per_host_cpu AS (
    SELECT region, az, hostname, AVG(measure_value::double) AS avg_cpu_utilization
    FROM "sampleDB".DevOps
    WHERE measure_name = 'cpu_utilization'
        AND time > ago(2h)
    GROUP BY region, az, hostname
)
SELECT region, az, hostname, avg_cpu_utilization, fleet_avg_cpu_utilization
FROM avg_fleet_utilization, avg_per_host_cpu
WHERE avg_cpu_utilization > 1.1 * fleet_avg_cpu_utilization
ORDER BY avg_cpu_utilization DESC
```

Trova l'CPUutilizzo medio suddiviso a intervalli di 30 secondi per un EC2 host specifico nelle ultime 2 ore:

```
SELECT BIN(time, 30s) AS binned_timestamp, ROUND(AVG(measure_value::double), 2) AS
 avg_cpu_utilization
FROM "sampleDB".DevOps
WHERE measure_name = 'cpu_utilization'
    AND hostname = 'host-Hovjv'
    AND time > ago(2h)
GROUP BY hostname, BIN(time, 30s)
ORDER BY binned_timestamp ASC
```

Trova l'CPUutilizzo medio associato a intervalli di 30 secondi per un EC2 host specifico nelle ultime 2 ore, inserendo i valori mancanti utilizzando l'interpolazione lineare:

```
WITH binned_timeseries AS (
    SELECT hostname, BIN(time, 30s) AS binned_timestamp,
 ROUND(AVG(measure_value::double), 2) AS avg_cpu_utilization
    FROM "sampleDB".DevOps
    WHERE measure_name = 'cpu_utilization'
        AND hostname = 'host-Hovjv'
        AND time > ago(2h)
    GROUP BY hostname, BIN(time, 30s)
), interpolated_timeseries AS (
    SELECT hostname,
        INTERPOLATE_LINEAR(
            CREATE_TIME_SERIES(binned_timestamp, avg_cpu_utilization),
                SEQUENCE(min(binned_timestamp), max(binned_timestamp), 15s)) AS
 interpolated_avg_cpu_utilization
    FROM binned_timeseries
    GROUP BY hostname
)
SELECT time, ROUND(value, 2) AS interpolated_cpu
FROM interpolated_timeseries
CROSS JOIN UNNEST(interpolated_avg_cpu_utilization)
```

Trova l'CPUutilizzo medio raggruppato a intervalli di 30 secondi per un EC2 host specifico nelle ultime 2 ore, inserendo i valori mancanti utilizzando l'interpolazione basata sull'ultima osservazione effettuata:

```
WITH binned_timeseries AS (
    SELECT hostname, BIN(time, 30s) AS binned_timestamp,
 ROUND(AVG(measure_value::double), 2) AS avg_cpu_utilization
    FROM "sampleDB".DevOps
    WHERE measure_name = 'cpu_utilization'
        AND hostname = 'host-Hovjv'
        AND time > ago(2h)
    GROUP BY hostname, BIN(time, 30s)
), interpolated_timeseries AS (
    SELECT hostname,
        INTERPOLATE_LOCF(
            CREATE_TIME_SERIES(binned_timestamp, avg_cpu_utilization),
                SEQUENCE(min(binned_timestamp), max(binned_timestamp), 15s)) AS
 interpolated_avg_cpu_utilization
    FROM binned_timeseries
```

```
    GROUP BY hostname
)
SELECT time, ROUND(value, 2) AS interpolated_cpu
FROM interpolated_timeseries
CROSS JOIN UNNEST(interpolated_avg_cpu_utilization)
```

Interrogazioni con funzioni aggregate

Di seguito è riportato un esempio di set di dati di esempio di scenario IoT per illustrare le query con funzioni aggregate.

Argomenti

- Dati di esempio
- Query di esempio

Dati di esempio

Timestream consente di archiviare e analizzare i dati dei sensori IoT come la posizione, il consumo di carburante, la velocità e la capacità di carico di una o più flotte di camion per consentire una gestione efficace della flotta. Di seguito sono riportati lo schema e alcuni dati di una tabella iot_trucks che memorizza dati di telemetria come posizione, consumo di carburante, velocità e capacità di carico dei camion.

Orario	truck_id	Make	Modello	Parco istanze	capacità_carburante	capacità_di carico	measure_name	measure_value::double	measure_value::varchar
2019-12-4 19:00:00.000 000000	1234567	GMC	Astro	Alpha (Afa)	100	500	fuel_reading	65,2	null
2019-12-4 19:00:00.000 000000	1234567	GMC	Astro	Alpha (Afa)	100	500	caricare	400,0	null

Orario	truck_id	Make	Modello	Parco istanze	capacità_carburante	capacità_di carico	measure_name	measure_value::double	measure_value::varchar
2019-12-4 19:00:00.000 000000	1234567	GMC	Astro	Alpha (Afa)	100	500	speed	90,2	null
2019-12-4 19:00:00.000 000000	1234567	GMC	Astro	Alpha (Afa)	100	500	posizione	null	47,6062 NM, 122.321 W
2019-12-4 19:00:00.000 000000	1234567	Kenworth	W900	Alpha (Afa)	150	1000	lettura del carburante	10.1	null
2019-12-4 19:00:00.000 000000	1234567	Kenworth	W900	Alpha (Afa)	150	1000	caricare	950,3	null
2019-12-4 19:00:00.000 000000	1234567	Kenworth	W900	Alpha (Afa)	150	1000	speed	50,8	null

Orario	truck_id	Make	Modello	Parco istanze	capacità_carburante	capacità_di carico	measure_name	measure_value::double	measure_value::varchar
2019-12-4 19:00:00.000 000000	1234567	Kenworth	W900	Alpha (Afa)	150	1000	posizione	null	40.7128 gradi N, 74.0060 gradi W

Query di esempio

Ottieni un elenco di tutti gli attributi e i valori dei sensori monitorati per ogni camion della flotta.

```
SELECT
    truck_id,
    fleet,
    fuel_capacity,
    model,
    load_capacity,
    make,
    measure_name
FROM "sampleDB".IoT
GROUP BY truck_id, fleet, fuel_capacity, model, load_capacity, make, measure_name
```

Ottieni i dati più recenti relativi al carburante di ogni camion della flotta nelle ultime 24 ore.

```
WITH latest_recorded_time AS (
    SELECT
        truck_id,
        max(time) as latest_time
    FROM "sampleDB".IoT
    WHERE measure_name = 'fuel-reading'
    AND time >= ago(24h)
    GROUP BY truck_id
)
SELECT
    b.truck_id,
    b.fleet,
```

```
    b.make,
    b.model,
    b.time,
    b.measure_value::double as last_reported_fuel_reading
FROM
latest_recorded_time a INNER JOIN "sampleDB".IoT b
ON a.truck_id = b.truck_id AND b.time = a.latest_time
WHERE b.measure_name = 'fuel-reading'
AND b.time > ago(24h)
ORDER BY b.truck_id
```

Identifica i camion che hanno utilizzato poco carburante (meno del 10%) nelle ultime 48 ore:

```
WITH low_fuel_trucks AS (
    SELECT time, truck_id, fleet, make, model, (measure_value::double/
cast(fuel_capacity as double)*100) AS fuel_pct
    FROM "sampleDB".IoT
    WHERE time >= ago(48h)
    AND (measure_value::double/cast(fuel_capacity as double)*100) < 10
    AND measure_name = 'fuel-reading'
),
other_trucks AS (
SELECT time, truck_id, (measure_value::double/cast(fuel_capacity as double)*100) as
 remaining_fuel
    FROM "sampleDB".IoT
    WHERE time >= ago(48h)
    AND truck_id IN (SELECT truck_id FROM low_fuel_trucks)
    AND (measure_value::double/cast(fuel_capacity as double)*100) >= 10
    AND measure_name = 'fuel-reading'
),
trucks_that_refuelled AS (
    SELECT a.truck_id
    FROM low_fuel_trucks a JOIN other_trucks b
    ON a.truck_id = b.truck_id AND b.time >= a.time
)
SELECT DISTINCT truck_id, fleet, make, model, fuel_pct
FROM low_fuel_trucks
WHERE truck_id NOT IN (
    SELECT truck_id FROM trucks_that_refuelled
)
```

Calcola il carico medio e la velocità massima di ogni camion nell'ultima settimana:

```
SELECT
    bin(time, 1d) as binned_time,
    fleet,
    truck_id,
    make,
    model,
    AVG(
        CASE WHEN measure_name = 'load' THEN measure_value::double ELSE NULL END
    ) AS avg_load_tons,
    MAX(
        CASE WHEN measure_name = 'speed' THEN measure_value::double ELSE NULL END
    ) AS max_speed_mph
FROM "sampleDB".IoT
WHERE time >= ago(7d)
AND measure_name IN ('load', 'speed')
GROUP BY fleet, truck_id, make, model, bin(time, 1d)
ORDER BY truck_id
```

Calcola l'efficienza di carico di ogni camion nell'ultima settimana:

```
WITH average_load_per_truck AS (
    SELECT
        truck_id,
        avg(measure_value::double)  AS avg_load
    FROM "sampleDB".IoT
    WHERE measure_name = 'load'
    AND time >= ago(7d)
    GROUP BY truck_id, fleet, load_capacity, make, model
),
truck_load_efficiency AS (
    SELECT
        a.truck_id,
        fleet,
        load_capacity,
        make,
        model,
        avg_load,
        measure_value::double,
        time,
        (measure_value::double*100)/avg_load as load_efficiency -- ,
 approx_percentile(avg_load_pct, DOUBLE '0.9')
    FROM "sampleDB".IoT a JOIN average_load_per_truck b
    ON a.truck_id = b.truck_id
```

```
    WHERE a.measure_name = 'load'
)
SELECT
    truck_id,
    time,
    load_efficiency
FROM truck_load_efficiency
ORDER BY truck_id, time
```

APIriferimento

Questa sezione contiene la documentazione API di riferimento per Amazon Timestream.

Timestream ne ha dueAPIs: Query e Write.

- Write API consente di eseguire operazioni come la creazione di tabelle, l'etichettatura delle risorse e la scrittura di record su Timestream.
- La Query API consente di eseguire operazioni di interrogazione.

> **Note**
>
> Entrambi APIs includono l' DescribeEndpoints azione. Sia per Query che per Write, l' DescribeEndpoints azione è identica.

APIDi seguito puoi leggere ulteriori informazioni su ciascuna di esse, insieme ai tipi di dati, agli errori e ai parametri comuni.

> **Note**
>
> Per i codici di errore comuni a tutti i AWS servizi, consulta la sezione AWS Support.

Argomenti

- Azioni
- Tipi di dati
- Errori comuni

- Parametri comuni

Azioni

Le seguenti azioni sono supportate da Amazon Timestream Write:

- CreateBatchLoadTask
- CreateDatabase
- CreateTable
- DeleteDatabase
- DeleteTable
- DescribeBatchLoadTask
- DescribeDatabase
- DescribeEndpoints
- DescribeTable
- ListBatchLoadTasks
- ListDatabases
- ListTables
- ListTagsForResource
- ResumeBatchLoadTask
- TagResource
- UntagResource
- UpdateDatabase
- UpdateTable
- WriteRecords

Le seguenti azioni sono supportate da Amazon Timestream Query:

- CancelQuery
- CreateScheduledQuery
- DeleteScheduledQuery
- DescribeAccountSettings

- DescribeEndpoints
- DescribeScheduledQuery
- ExecuteScheduledQuery
- ListScheduledQueries
- ListTagsForResource
- PrepareQuery
- Query
- TagResource
- UntagResource
- UpdateAccountSettings
- UpdateScheduledQuery

Amazon Timestream Write

Le seguenti azioni sono supportate da Amazon Timestream Write:

- CreateBatchLoadTask
- CreateDatabase
- CreateTable
- DeleteDatabase
- DeleteTable
- DescribeBatchLoadTask
- DescribeDatabase
- DescribeEndpoints
- DescribeTable
- ListBatchLoadTasks
- ListDatabases
- ListTables
- ListTagsForResource
- ResumeBatchLoadTask
- TagResource
- UntagResource

- UpdateDatabase
- UpdateTable
- WriteRecords

CreateBatchLoadTask
Servizio: Amazon Timestream Write

Crea una nuova attività di caricamento in batch di Timestream. Un'attività di caricamento in batch elabora i dati da un'CSVorigine in una posizione S3 e li scrive in una tabella Timestream. Una mappatura dall'origine alla destinazione è definita in un'attività di caricamento in batch. Gli errori e gli eventi vengono scritti in un report in una posizione S3. Per il report, se la AWS KMS chiave non è specificata, il report verrà crittografato con una chiave gestita S3, se `SSE_S3` possibile. Altrimenti viene generato un errore. Per ulteriori informazioni, consulta chiavi gestite da AWS . Si applicano le quote di servizio. Per i dettagli, vedi esempio di codice.

Sintassi della richiesta

```
{
   "ClientToken": "string",
   "DataModelConfiguration": {
      "DataModel": {
         "DimensionMappings": [
            {
               "DestinationColumn": "string",
               "SourceColumn": "string"
            }
         ],
         "MeasureNameColumn": "string",
         "MixedMeasureMappings": [
            {
               "MeasureName": "string",
               "MeasureValueType": "string",
               "MultiMeasureAttributeMappings": [
                  {
                     "MeasureValueType": "string",
                     "SourceColumn": "string",
                     "TargetMultiMeasureAttributeName": "string"
                  }
               ],
               "SourceColumn": "string",
               "TargetMeasureName": "string"
            }
         ],
         "MultiMeasureMappings": {
            "MultiMeasureAttributeMappings": [
               {
```

```
                    "MeasureValueType": "string",
                    "SourceColumn": "string",
                    "TargetMultiMeasureAttributeName": "string"
                 }
              ],
              "TargetMultiMeasureName": "string"
           },
           "TimeColumn": "string",
           "TimeUnit": "string"
        },
        "DataModelS3Configuration": {
           "BucketName": "string",
           "ObjectKey": "string"
        }
     },
     "DataSourceConfiguration": {
        "CsvConfiguration": {
           "ColumnSeparator": "string",
           "EscapeChar": "string",
           "NullValue": "string",
           "QuoteChar": "string",
           "TrimWhiteSpace": boolean
        },
        "DataFormat": "string",
        "DataSourceS3Configuration": {
           "BucketName": "string",
           "ObjectKeyPrefix": "string"
        }
     },
     "RecordVersion": number,
     "ReportConfiguration": {
        "ReportS3Configuration": {
           "BucketName": "string",
           "EncryptionOption": "string",
           "KmsKeyId": "string",
           "ObjectKeyPrefix": "string"
        }
     },
     "TargetDatabaseName": "string",
     "TargetTableName": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

ClientToken

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 64 caratteri.

Campo obbligatorio: no

DataModelConfiguration

Tipo: oggetto DataModelConfiguration

Campo obbligatorio: no

DataSourceConfiguration

Definisce i dettagli di configurazione sull'origine dati per un'attività di caricamento in batch.

Tipo: oggetto DataSourceConfiguration

Campo obbligatorio: sì

RecordVersion

Tipo: long

Campo obbligatorio: no

ReportConfiguration

Configurazione del report per un'attività di caricamento in batch. Contiene dettagli sulla posizione in cui vengono archiviate le segnalazioni di errori.

Tipo: oggetto ReportConfiguration

Campo obbligatorio: sì

TargetDatabaseName

Database Target Timestream per un'attività di caricamento in batch.

Tipo: stringa

Modello: `[a-zA-Z0-9_.-]+`

Campo obbligatorio: sì

TargetTableName

Tabella Target Timestream per un'attività di caricamento in batch.

Tipo: stringa

Modello: `[a-zA-Z0-9_.-]+`

Campo obbligatorio: sì

Sintassi della risposta

```
{
   "TaskId": "string"
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

TaskId

L'ID dell'operazione di caricamento in batch.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 32 caratteri.

Modello: `[A-Z0-9]+`

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

ConflictException

Timestream non è stato in grado di elaborare questa richiesta perché contiene una risorsa già esistente.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha tentato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ServiceQuotaExceededException

La quota di risorse dell'istanza è stata superata per questo account.

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

CreateDatabase

Servizio: Amazon Timestream Write

Crea un nuovo database Timestream. Se la AWS KMS chiave non è specificata, il database verrà crittografato con una AWS KMS chiave gestita da Timestream situata nel tuo account. Per ulteriori informazioni, consulta chiavi gestite da AWS . Si applicano le quote di servizio. Per i dettagli, consulta l'esempio di codice.

Sintassi della richiesta

```
{
   "DatabaseName": "string",
   "KmsKeyId": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

DatabaseName

Nome del database Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Modello: `[a-zA-Z0-9_.-]+`

Campo obbligatorio: sì

KmsKeyId

La AWS KMS chiave per il database. Se la AWS KMS chiave non è specificata, il database verrà crittografato con una AWS KMS chiave gestita da Timestream situata nel tuo account. Per ulteriori informazioni, consulta chiavi gestite da AWS .

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: no

Tags

Un elenco di coppie chiave-valore per etichettare la tabella.

Tipo: matrice di oggetti Tag

Membri dell'array: numero minimo di 0 elementi. Numero massimo di 200 elementi.

Campo obbligatorio: no

Sintassi della risposta

```
{
   "Database": {
      "Arn": "string",
      "CreationTime": number,
      "DatabaseName": "string",
      "KmsKeyId": "string",
      "LastUpdatedTime": number,
      "TableCount": number
   }
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

Database

Il database Timestream appena creato.

Tipo: oggetto Database

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

ConflictException

Timestream non è stato in grado di elaborare questa richiesta perché contiene una risorsa già esistente.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ServiceQuotaExceededException

La quota di risorse dell'istanza è stata superata per questo account.

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

CreateTable

Servizio: Amazon Timestream Write

Aggiunge una nuova tabella a un database esistente nel tuo account. In un Account AWS, i nomi delle tabelle devono essere almeno univoci all'interno di ogni regione se si trovano nello stesso database. Potresti avere nomi di tabella identici nella stessa regione se le tabelle si trovano in database separati. Durante la creazione della tabella, è necessario specificare il nome della tabella, il nome del database e le proprietà di conservazione. Si applicano le quote di servizio. Consulta l'esempio di codice per i dettagli.

Sintassi della richiesta

```
{
   "DatabaseName": "string",
   "MagneticStoreWriteProperties": {
      "EnableMagneticStoreWrites": boolean,
      "MagneticStoreRejectedDataLocation": {
         "S3Configuration": {
            "BucketName": "string",
            "EncryptionOption": "string",
            "KmsKeyId": "string",
            "ObjectKeyPrefix": "string"
         }
      }
   },
   "RetentionProperties": {
      "MagneticStoreRetentionPeriodInDays": number,
      "MemoryStoreRetentionPeriodInHours": number
   },
   "Schema": {
      "CompositePartitionKey": [
         {
            "EnforcementInRecord": "string",
            "Name": "string",
            "Type": "string"
         }
      ]
   },
   "TableName": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
```

```
        }
    ]
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

DatabaseName

Nome del database Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Modello: `[a-zA-Z0-9_.-]+`

Campo obbligatorio: sì

MagneticStoreWriteProperties

Contiene le proprietà da impostare nella tabella quando si abilitano le scritture dello store magnetico.

Tipo: oggetto MagneticStoreWriteProperties

Campo obbligatorio: no

RetentionProperties

La durata per la quale i dati delle serie temporali devono essere archiviati nell'archivio di memoria e nell'archivio magnetico.

Tipo: oggetto RetentionProperties

Campo obbligatorio: no

Schema

Lo schema della tabella.

Tipo: oggetto Schema

Campo obbligatorio: no

TableName

Nome della tabella Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Modello: `[a-zA-Z0-9_.-]+`

Campo obbligatorio: sì

Tags

Un elenco di coppie chiave-valore per etichettare la tabella.

Tipo: matrice di oggetti Tag

Membri dell'array: numero minimo di 0 elementi. Numero massimo di 200 elementi.

Campo obbligatorio: no

Sintassi della risposta

```
{
   "Table": {
      "Arn": "string",
      "CreationTime": number,
      "DatabaseName": "string",
      "LastUpdatedTime": number,
      "MagneticStoreWriteProperties": {
         "EnableMagneticStoreWrites": boolean,
         "MagneticStoreRejectedDataLocation": {
            "S3Configuration": {
               "BucketName": "string",
               "EncryptionOption": "string",
               "KmsKeyId": "string",
               "ObjectKeyPrefix": "string"
            }
         }
      },
      "RetentionProperties": {
```

```
         "MagneticStoreRetentionPeriodInDays": number,
         "MemoryStoreRetentionPeriodInHours": number
      },
      "Schema": {
         "CompositePartitionKey": [
            {
               "EnforcementInRecord": "string",
               "Name": "string",
               "Type": "string"
            }
         ]
      },
      "TableName": "string",
      "TableStatus": "string"
   }
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

Table

La tabella Timestream appena creata.

Tipo: oggetto Table

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

ConflictException

Timestream non è stato in grado di elaborare questa richiesta perché contiene una risorsa già esistente.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha tentato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ServiceQuotaExceededException

La quota di risorse dell'istanza è stata superata per questo account.

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

DeleteDatabase
Servizio: Amazon Timestream Write

Elimina un determinato database Timestream. Si tratta di un'operazione irreversibile. Dopo l'eliminazione di un database, i dati delle serie temporali dalle relative tabelle non possono essere recuperati.

> **Note**
>
> Tutte le tabelle del database devono essere prima eliminate, altrimenti verrà ValidationException generato un errore.
> A causa della natura dei tentativi distribuiti, l'operazione può restituire un risultato positivo o un. ResourceNotFoundException I clienti dovrebbero considerarli equivalenti.

Consulta l'esempio di codice per i dettagli.

Sintassi della richiesta

```
{
   "DatabaseName": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

DatabaseName

Il nome del database Timestream da eliminare.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200 con un corpo vuotoHTTP.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha tentato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API codice in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

DeleteTable
Servizio: Amazon Timestream Write

Elimina una determinata tabella Timestream. Si tratta di un'operazione irreversibile. Dopo l'eliminazione di una tabella del database Timestream, i dati delle serie temporali memorizzati nella tabella non possono essere recuperati.

> **Note**
>
> A causa della natura dei tentativi distribuiti, l'operazione può restituire un risultato positivo o un. ResourceNotFoundException I clienti dovrebbero considerarli equivalenti.

Consulta l'esempio di codice per i dettagli.

Sintassi della richiesta

```
{
   "DatabaseName": "string",
   "TableName": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

DatabaseName

Il nome del database in cui deve essere eliminato il database Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

TableName

Il nome della tabella Timestream da eliminare.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200 con un corpo vuotoHTTP.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha cercato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

DescribeBatchLoadTask
Servizio: Amazon Timestream Write

Restituisce informazioni sull'attività di caricamento in batch, tra cui configurazioni, mappature, avanzamento e altri dettagli. Si applicano le quote di servizio. Consulta l'esempio di codice per i dettagli.

Sintassi della richiesta

```
{
   "TaskId": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel formato. JSON

TaskId

L'ID dell'operazione di caricamento in batch.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 32 caratteri.

Modello: `[A-Z0-9]+`

Campo obbligatorio: sì

Sintassi della risposta

```
{
   "BatchLoadTaskDescription": {
      "CreationTime": number,
      "DataModelConfiguration": {
         "DataModel": {
            "DimensionMappings": [
               {
                  "DestinationColumn": "string",
                  "SourceColumn": "string"
               }
```

```
            ],
            "MeasureNameColumn": "string",
            "MixedMeasureMappings": [
               {
                  "MeasureName": "string",
                  "MeasureValueType": "string",
                  "MultiMeasureAttributeMappings": [
                     {
                        "MeasureValueType": "string",
                        "SourceColumn": "string",
                        "TargetMultiMeasureAttributeName": "string"
                     }
                  ],
                  "SourceColumn": "string",
                  "TargetMeasureName": "string"
               }
            ],
            "MultiMeasureMappings": {
               "MultiMeasureAttributeMappings": [
                  {
                     "MeasureValueType": "string",
                     "SourceColumn": "string",
                     "TargetMultiMeasureAttributeName": "string"
                  }
               ],
               "TargetMultiMeasureName": "string"
            },
            "TimeColumn": "string",
            "TimeUnit": "string"
         },
         "DataModelS3Configuration": {
            "BucketName": "string",
            "ObjectKey": "string"
         }
      },
      "DataSourceConfiguration": {
         "CsvConfiguration": {
            "ColumnSeparator": "string",
            "EscapeChar": "string",
            "NullValue": "string",
            "QuoteChar": "string",
            "TrimWhiteSpace": boolean
         },
         "DataFormat": "string",
```

```
         "DataSourceS3Configuration": {
            "BucketName": "string",
            "ObjectKeyPrefix": "string"
         }
      },
      "ErrorMessage": "string",
      "LastUpdatedTime": number,
      "ProgressReport": {
         "BytesMetered": number,
         "FileFailures": number,
         "ParseFailures": number,
         "RecordIngestionFailures": number,
         "RecordsIngested": number,
         "RecordsProcessed": number
      },
      "RecordVersion": number,
      "ReportConfiguration": {
         "ReportS3Configuration": {
            "BucketName": "string",
            "EncryptionOption": "string",
            "KmsKeyId": "string",
            "ObjectKeyPrefix": "string"
         }
      },
      "ResumableUntil": number,
      "TargetDatabaseName": "string",
      "TargetTableName": "string",
      "TaskId": "string",
      "TaskStatus": "string"
   }
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

BatchLoadTaskDescription

Descrizione dell'operazione di caricamento in batch.

Tipo: oggetto BatchLoadTaskDescription

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha cercato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET

- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

DescribeDatabase

Servizio: Amazon Timestream Write

Restituisce informazioni sul database, incluso il nome del database, l'ora di creazione del database e il numero totale di tabelle trovate all'interno del database. Si applicano le quote di servizio. Consulta l'esempio di codice per i dettagli.

Sintassi della richiesta

```
{
   "DatabaseName": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

DatabaseName

Nome del database Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

Sintassi della risposta

```
{
   "Database": {
      "Arn": "string",
      "CreationTime": number,
      "DatabaseName": "string",
      "KmsKeyId": "string",
      "LastUpdatedTime": number,
      "TableCount": number
   }
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

Database

Nome della tabella Timestream.

Tipo: oggetto Database

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha cercato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

DescribeEndpoints
Servizio: Amazon Timestream Write

Restituisce un elenco di endpoint disponibili su cui effettuare chiamate TimestreamAPI. Questa API operazione è disponibile sia tramite Write che Query. APIs

Poiché i Timestream SDKs sono progettati per funzionare in modo trasparente con l'architettura del servizio, inclusa la gestione e la mappatura degli endpoint del servizio, non è consigliabile utilizzare questa operazione a meno che: API

- Stai usando VPC endpoints () con Timestream AWS PrivateLink
- L'applicazione utilizza un linguaggio di programmazione che non supporta ancora SDK
- È necessario un controllo migliore sull'implementazione lato client

Per informazioni dettagliate su come e quando utilizzare e implementare DescribeEndpoints, consulta The Endpoint Discovery Pattern.

Sintassi della risposta

```
{
   "Endpoints": [
      {
         "Address": "string",
         "CachePeriodInMinutes": number
      }
   ]
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

Endpoints

Un `Endpoints` oggetto viene restituito quando viene effettuata una `DescribeEndpoints` richiesta.

Tipo: matrice di oggetti Endpoint

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

DescribeTable

Servizio: Amazon Timestream Write

Restituisce informazioni sulla tabella, tra cui il nome della tabella, il nome del database, la durata di conservazione dell'archivio di memoria e dell'archivio magnetico. Si applicano le quote di servizio. Consulta l'esempio di codice per i dettagli.

Sintassi della richiesta

```
{
   "DatabaseName": "string",
   "TableName": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

DatabaseName

Nome del database Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

TableName

Nome della tabella Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

Sintassi della risposta

```
{
   "Table": {
```

```
      "Arn": "string",
      "CreationTime": number,
      "DatabaseName": "string",
      "LastUpdatedTime": number,
      "MagneticStoreWriteProperties": {
         "EnableMagneticStoreWrites": boolean,
         "MagneticStoreRejectedDataLocation": {
            "S3Configuration": {
               "BucketName": "string",
               "EncryptionOption": "string",
               "KmsKeyId": "string",
               "ObjectKeyPrefix": "string"
            }
         }
      },
      "RetentionProperties": {
         "MagneticStoreRetentionPeriodInDays": number,
         "MemoryStoreRetentionPeriodInHours": number
      },
      "Schema": {
         "CompositePartitionKey": [
            {
               "EnforcementInRecord": "string",
               "Name": "string",
               "Type": "string"
            }
         ]
      },
      "TableName": "string",
      "TableStatus": "string"
   }
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

Table

La tabella Timestream.

Tipo: oggetto Table

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha tentato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

ListBatchLoadTasks
Servizio: Amazon Timestream Write

Fornisce un elenco delle attività di caricamento in batch, insieme al nome, allo stato, alla data di ripresa dell'attività e ad altri dettagli. Consulta l'esempio di codice per i dettagli.

Sintassi della richiesta

```
{
   "MaxResults": number,
   "NextToken": "string",
   "TaskStatus": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

MaxResults

Il numero totale di elementi da restituire nell'output. Se il numero totale di articoli disponibili è superiore al valore specificato, nell'output NextToken viene fornito un. Per riprendere l'impaginazione, fornite il NextToken valore come argomento di una chiamata successivaAPI.

Tipo: integer

Intervallo valido: valore minimo di 1. valore massimo pari a 100.

Campo obbligatorio: no

NextToken

Token per specificare dove iniziare l'impaginazione. Questo è il risultato di una risposta precedentemente NextToken troncata.

Tipo: string

Campo obbligatorio: no

TaskStatus

Stato dell'operazione di caricamento in batch.

Tipo: stringa

Valori validi: `CREATED | IN_PROGRESS | FAILED | SUCCEEDED | PROGRESS_STOPPED | PENDING_RESUME`

Campo obbligatorio: no

Sintassi della risposta

```
{
   "BatchLoadTasks": [
      {
         "CreationTime": number,
         "DatabaseName": "string",
         "LastUpdatedTime": number,
         "ResumableUntil": number,
         "TableName": "string",
         "TaskId": "string",
         "TaskStatus": "string"
      }
   ],
   "NextToken": "string"
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

BatchLoadTasks

Un elenco di dettagli dell'attività di caricamento in batch.

Tipo: matrice di oggetti BatchLoadTask

NextToken

Token per specificare dove iniziare l'impaginazione. Fornisci il successivo ListBatchLoadTasksRequest.

Tipo: stringa

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++

- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

ListDatabases
Servizio: Amazon Timestream Write

Restituisce un elenco dei tuoi database Timestream. Si applicano le quote di servizio. Consulta l'esempio di codice per i dettagli.

Sintassi della richiesta

```
{
   "MaxResults": number,
   "NextToken": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati in JSON formato.

MaxResults

Il numero totale di elementi da restituire nell'output. Se il numero totale di articoli disponibili è superiore al valore specificato, nell'output NextToken viene fornito un. Per riprendere l'impaginazione, fornite il NextToken valore come argomento di una chiamata successivaAPI.

Tipo: integer

Intervallo valido: valore minimo di 1. Valore massimo di 20.

Campo obbligatorio: no

NextToken

Il token di impaginazione. Per riprendere l'impaginazione, fornite il NextToken valore come argomento di una chiamata successiva. API

Tipo: string

Campo obbligatorio: no

Sintassi della risposta

```
{
```

```
   "Databases": [
      {
         "Arn": "string",
         "CreationTime": number,
         "DatabaseName": "string",
         "KmsKeyId": "string",
         "LastUpdatedTime": number,
         "TableCount": number
      }
   ],
   "NextToken": "string"
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di 200. HTTP

I seguenti dati vengono restituiti in JSON formato dal servizio.

Databases

Un elenco di nomi di database.

Tipo: matrice di oggetti Database

NextToken

Il token di impaginazione. Questo parametro viene restituito quando la risposta viene troncata.

Tipo: stringa

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

ListTables
Servizio: Amazon Timestream Write

Fornisce un elenco di tabelle, insieme al nome, allo stato e alle proprietà di conservazione di ciascuna tabella. Consulta l'esempio di codice per i dettagli.

Sintassi della richiesta

```
{
   "DatabaseName": "string",
   "MaxResults": number,
   "NextToken": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

DatabaseName

Nome del database Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: no

MaxResults

Il numero totale di elementi da restituire nell'output. Se il numero totale di articoli disponibili è superiore al valore specificato, nell'output NextToken viene fornito un. Per riprendere l'impaginazione, fornite il NextToken valore come argomento di una chiamata successivaAPI.

Tipo: integer

Intervallo valido: valore minimo di 1. Valore massimo di 20.

Campo obbligatorio: no

NextToken

Il token di impaginazione. Per riprendere l'impaginazione, fornite il NextToken valore come argomento di una chiamata successiva. API

Tipo: string

Campo obbligatorio: no

Sintassi della risposta

```
{
   "NextToken": "string",
   "Tables": [
      {
         "Arn": "string",
         "CreationTime": number,
         "DatabaseName": "string",
         "LastUpdatedTime": number,
         "MagneticStoreWriteProperties": {
            "EnableMagneticStoreWrites": boolean,
            "MagneticStoreRejectedDataLocation": {
               "S3Configuration": {
                  "BucketName": "string",
                  "EncryptionOption": "string",
                  "KmsKeyId": "string",
                  "ObjectKeyPrefix": "string"
               }
            }
         },
         "RetentionProperties": {
            "MagneticStoreRetentionPeriodInDays": number,
            "MemoryStoreRetentionPeriodInHours": number
         },
         "Schema": {
            "CompositePartitionKey": [
               {
                  "EnforcementInRecord": "string",
                  "Name": "string",
                  "Type": "string"
               }
            ]
         },
```

```
            "TableName": "string",
            "TableStatus": "string"
        }
    ]
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di 200. HTTP

I seguenti dati vengono restituiti in JSON formato dal servizio.

NextToken

Token per specificare dove iniziare l'impaginazione. Si tratta NextToken di una risposta precedentemente troncata.

Tipo: stringa

Tables

Un elenco di tabelle.

Tipo: matrice di oggetti Table

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha cercato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

ListTagsForResource
Servizio: Amazon Timestream Write

Elenca tutti i tag su una risorsa Timestream.

Sintassi della richiesta

```
{
   "ResourceARN": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

ResourceARN

La risorsa Timestream con i tag da elencare. Questo valore è un Amazon Resource Name (ARN).

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. Lunghezza massima di 1011.

Campo obbligatorio: sì

Sintassi della risposta

```
{
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

Tags

I tag attualmente associati alla risorsa Timestream.

Tipo: matrice di oggetti Tag

Membri dell'array: numero minimo di 0 elementi. Numero massimo di 200 elementi.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha cercato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

ResumeBatchLoadTask
Servizio: Amazon Timestream Write

Sintassi della richiesta

```
{
   "TaskId": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

TaskId

L'ID dell'operazione di caricamento in batch da riprendere.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 32 caratteri.

Modello: `[A-Z0-9]+`

Campo obbligatorio: sì

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200 con un HTTP corpo vuoto.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha cercato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2

- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

TagResource
Servizio: Amazon Timestream Write

Associa un set di tag a una risorsa Timestream. È quindi possibile attivare questi tag definiti dall'utente in modo che vengano visualizzati nella console di Billing and Cost Management per il monitoraggio dell'allocazione dei costi.

Sintassi della richiesta

```
{
   "ResourceARN": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel formato. JSON

ResourceARN

Identifica la risorsa Timestream a cui aggiungere i tag. Questo valore è un Amazon Resource Name (ARN).

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. Lunghezza massima di 1011.

Campo obbligatorio: sì

Tags

I tag da assegnare alla risorsa Timestream.

Tipo: matrice di oggetti Tag

Membri dell'array: numero minimo di 0 elementi. Numero massimo di 200 elementi.

Campo obbligatorio: sì

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200 con un corpo vuotoHTTP.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha cercato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ServiceQuotaExceededException

La quota di risorse dell'istanza è stata superata per questo account.

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

UntagResource
Servizio: Amazon Timestream Write

Rimuove l'associazione di tag da una risorsa Timestream.

Sintassi della richiesta

```
{
   "ResourceARN": "string",
   "TagKeys": [ "string" ]
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

ResourceARN

La risorsa Timestream da cui verranno rimossi i tag. Questo valore è un Amazon Resource Name (ARN).

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. Lunghezza massima di 1011.

Campo obbligatorio: sì

TagKeys

Un elenco di tag e chiavi. I tag esistenti della risorsa le cui chiavi sono membri di questo elenco verranno rimossi dalla risorsa Timestream.

Tipo: matrice di stringhe

Membri dell'array: numero minimo di 0 elementi. Numero massimo di 200 elementi.

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 128 caratteri.

Campo obbligatorio: sì

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta HTTP 200 con un corpo vuotoHTTP.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha tentato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ServiceQuotaExceededException

La quota di risorse dell'istanza è stata superata per questo account.

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

UpdateDatabase
Servizio: Amazon Timestream Write

Modifica la AWS KMS chiave per un database esistente. Durante l'aggiornamento del database, è necessario specificare il nome del database e l'identificatore della nuova AWS KMS chiave da utilizzare ()`KmsKeyId`. Se ci sono `UpdateDatabase` richieste simultanee, vince il primo scrittore.

Consulta l'esempio di codice per i dettagli.

Sintassi della richiesta

```
{
   "DatabaseName": "string",
   "KmsKeyId": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

DatabaseName

Nome del database.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

KmsKeyId

L'identificatore della nuova AWS KMS chiave (`KmsKeyId`) da utilizzare per crittografare i dati memorizzati nel database. Se il `KmsKeyId` valore attualmente registrato nel database è lo stesso indicato `KmsKeyId` nella richiesta, non ci sarà alcun aggiornamento.

È possibile specificare l'`KmsKeyId`utilizzo di uno dei seguenti:

- ID chiave: `1234abcd-12ab-34cd-56ef-1234567890ab`
- ChiaveARN: `arn:aws:kms:us-east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab`
- Nome alias: `alias/ExampleAlias`

- PseudonimoARN: `arn:aws:kms:us-east-1:111122223333:alias/ExampleAlias`

  Tipo: stringa

  Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

  Campo obbligatorio: sì

Sintassi della risposta

```
{
   "Database": {
      "Arn": "string",
      "CreationTime": number,
      "DatabaseName": "string",
      "KmsKeyId": "string",
      "LastUpdatedTime": number,
      "TableCount": number
   }
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

Database

Un contenitore di primo livello per una tabella. I database e le tabelle sono i concetti di gestione fondamentali in Amazon Timestream. Tutte le tabelle di un database sono crittografate con la stessa AWS KMS chiave.

Tipo: oggetto Database

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha cercato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ServiceQuotaExceededException

La quota di risorse dell'istanza è stata superata per questo account.

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

UpdateTable

Servizio: Amazon Timestream Write

Modifica la durata di conservazione dell'archivio di memoria e dell'archivio magnetico per la tabella Timestream. Si noti che la modifica della durata di conservazione ha effetto immediato. Ad esempio, se il periodo di conservazione dell'archivio di memoria è stato inizialmente impostato su 2 ore e poi modificato su 24 ore, l'archivio di memoria sarà in grado di contenere 24 ore di dati, ma verrà popolato con 24 ore di dati 22 ore dopo la modifica. Timestream non recupera i dati dall'archivio magnetico per popolare l'archivio di memoria.

Consulta l'esempio di codice per i dettagli.

Sintassi della richiesta

```
{
   "DatabaseName": "string",
   "MagneticStoreWriteProperties": {
      "EnableMagneticStoreWrites": boolean,
      "MagneticStoreRejectedDataLocation": {
         "S3Configuration": {
            "BucketName": "string",
            "EncryptionOption": "string",
            "KmsKeyId": "string",
            "ObjectKeyPrefix": "string"
         }
      }
   },
   "RetentionProperties": {
      "MagneticStoreRetentionPeriodInDays": number,
      "MemoryStoreRetentionPeriodInHours": number
   },
   "Schema": {
      "CompositePartitionKey": [
         {
            "EnforcementInRecord": "string",
            "Name": "string",
            "Type": "string"
         }
      ]
   },
   "TableName": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel formato. JSON

DatabaseName

Nome del database Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

MagneticStoreWriteProperties

Contiene le proprietà da impostare nella tabella quando si abilitano le scritture dello store magnetico.

Tipo: oggetto MagneticStoreWriteProperties

Campo obbligatorio: no

RetentionProperties

La durata di conservazione dell'archivio di memoria e dell'archivio magnetico.

Tipo: oggetto RetentionProperties

Campo obbligatorio: no

Schema

Lo schema della tabella.

Tipo: oggetto Schema

Campo obbligatorio: no

TableName

Nome della tabella Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

Sintassi della risposta

```
{
   "Table": {
      "Arn": "string",
      "CreationTime": number,
      "DatabaseName": "string",
      "LastUpdatedTime": number,
      "MagneticStoreWriteProperties": {
         "EnableMagneticStoreWrites": boolean,
         "MagneticStoreRejectedDataLocation": {
            "S3Configuration": {
               "BucketName": "string",
               "EncryptionOption": "string",
               "KmsKeyId": "string",
               "ObjectKeyPrefix": "string"
            }
         }
      },
      "RetentionProperties": {
         "MagneticStoreRetentionPeriodInDays": number,
         "MemoryStoreRetentionPeriodInHours": number
      },
      "Schema": {
         "CompositePartitionKey": [
            {
               "EnforcementInRecord": "string",
               "Name": "string",
               "Type": "string"
            }
         ]
      },
      "TableName": "string",
      "TableStatus": "string"
   }
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

Table

La tabella Timestream aggiornata.

Tipo: oggetto Table

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha tentato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

WriteRecords
Servizio: Amazon Timestream Write

Consente di scrivere i dati delle serie temporali in Timestream. È possibile specificare un singolo punto dati o un batch di punti dati da inserire nel sistema. Timestream ti offre uno schema flessibile che rileva automaticamente i nomi delle colonne e i tipi di dati per le tue tabelle Timestream in base ai nomi delle dimensioni e ai tipi di dati dei punti dati specificati quando richiami le scritture nel database.

Timestream supporta la semantica di lettura con eventuale coerenza. Ciò significa che quando si interrogano i dati immediatamente dopo aver scritto un batch di dati in Timestream, i risultati della query potrebbero non riflettere i risultati di un'operazione di scrittura completata di recente. I risultati possono includere anche alcuni dati obsoleti. Se si ripete la richiesta di interrogazione dopo un breve periodo, i risultati dovrebbero restituire i dati più recenti. Si applicano le quote di servizio.

Consulta l'esempio di codice per i dettagli.

Sconvolge

È possibile utilizzare il `Version` parametro in una `WriteRecords` richiesta di aggiornamento dei punti dati. Timestream tiene traccia di un numero di versione per ogni record. `Version`il valore predefinito è 1 quando non è specificato per il record nella richiesta. Timestream aggiorna il valore di misura di un record esistente insieme al suo `Version` quando riceve una richiesta di scrittura con un `Version` numero più alto per quel record. Quando riceve una richiesta di aggiornamento in cui il valore di misura è uguale a quello del record esistente, Timestream si aggiorna comunque`Version`, se è maggiore del valore esistente di. `Version` È possibile aggiornare un punto dati tutte le volte che si desidera, purché il valore di aumenti `Version` continuamente.

Ad esempio, supponete di scrivere un nuovo record senza indicarlo `Version` nella richiesta. Timestream memorizza questo record e lo imposta su. `Version` `1` Supponiamo ora di provare ad aggiornare questo record con una `WriteRecords` richiesta dello stesso record con un valore di misura diverso ma, come in precedenza, di non fornire. `Version` In questo caso, Timestream rifiuterà questo aggiornamento con un `RejectedRecordsException` poiché la versione del record aggiornato non è maggiore del valore esistente di Version.

Tuttavia, se dovessi inviare nuovamente la richiesta di aggiornamento con `Version` set to2, Timestream riuscirà ad aggiornare il valore del record e verrebbe impostato su. `Version` `2` Supponiamo quindi di aver inviato una `WriteRecords` richiesta con lo stesso record e un valore di misura identico, ma impostata su. `Version` `3` In questo caso, Timestream si aggiornerebbe solo a.

`Version` 3 Eventuali ulteriori aggiornamenti dovrebbero inviare un numero di versione maggiore di3, oppure le richieste di aggiornamento riceverebbero un. `RejectedRecordsException`

Sintassi della richiesta

```
{
   "CommonAttributes": {
      "Dimensions": [
         {
            "DimensionValueType": "string",
            "Name": "string",
            "Value": "string"
         }
      ],
      "MeasureName": "string",
      "MeasureValue": "string",
      "MeasureValues": [
         {
            "Name": "string",
            "Type": "string",
            "Value": "string"
         }
      ],
      "MeasureValueType": "string",
      "Time": "string",
      "TimeUnit": "string",
      "Version": number
   },
   "DatabaseName": "string",
   "Records": [
      {
         "Dimensions": [
            {
               "DimensionValueType": "string",
               "Name": "string",
               "Value": "string"
            }
         ],
         "MeasureName": "string",
         "MeasureValue": "string",
         "MeasureValues": [
            {
               "Name": "string",
               "Type": "string",
```

```
                "Value": "string"
            }
        ],
        "MeasureValueType": "string",
        "Time": "string",
        "TimeUnit": "string",
        "Version": number
    }
  ],
  "TableName": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

CommonAttributes

Un record che contiene gli attributi comuni di misura, dimensione, ora e versione condivisi tra tutti i record della richiesta. Gli attributi di misura e dimensione specificati verranno uniti agli attributi di misura e dimensione nell'oggetto records quando i dati vengono scritti in Timestream. Le dimensioni non possono sovrapporsi o verrà generato un`ValidationException`. In altre parole, un record deve contenere dimensioni con nomi univoci.

Tipo: oggetto Record

Campo obbligatorio: no

DatabaseName

Nome del database Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

Records

Una matrice di record che contiene gli attributi univoci di misura, dimensione, ora e versione per ogni punto dati della serie temporale.

Tipo: matrice di oggetti Record

Membri dell'array: numero minimo di 1 elemento. Numero massimo di 100 elementi.

Campo obbligatorio: sì

TableName

Nome della tabella Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

Sintassi della risposta

```
{
   "RecordsIngested": {
      "MagneticStore": number,
      "MemoryStore": number,
      "Total": number
   }
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

RecordsIngested

Informazioni sui record acquisiti da questa richiesta.

Tipo: oggetto RecordsIngested

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Timestream non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 500

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

RejectedRecordsException

WriteRecords genererebbe questa eccezione nei seguenti casi:

- Record con dati duplicati in cui sono presenti più record con le stesse dimensioni, timestamp e nomi di misure ma:
  - I valori di misura sono diversi
  - La versione non è presente nella richiesta oppure il valore della versione nel nuovo record è uguale o inferiore al valore esistente

  In questo caso, se Timestream rifiuta i dati, il `ExistingVersion` campo nella `RejectedRecords` risposta indicherà la versione del record corrente. Per forzare un aggiornamento, puoi inviare nuovamente la richiesta con una versione del record impostata su un valore maggiore di. `ExistingVersion`
- Record con timestamp che non rientrano nella durata di conservazione dell'archivio di memoria.
- Record con dimensioni o misure che superano i limiti definiti dal Timestream.

Per ulteriori informazioni, consulta Quotas nella Amazon Timestream Developer Guide.

HTTPCodice di stato: 400

ResourceNotFoundException

L'operazione ha tentato di accedere a una risorsa inesistente. La risorsa potrebbe non essere specificata correttamente o il relativo stato potrebbe non esserlo. ACTIVE

HTTPCodice di stato: 400

ThrottlingException

Troppe richieste sono state fatte da un utente e hanno superato le quote di servizio. La richiesta è stata sottoposta a throttling.

HTTPCodice di stato: 400

ValidationException

Una richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

Interrogazione su Amazon Timestream

Le seguenti azioni sono supportate da Amazon Timestream Query:

- CancelQuery
- CreateScheduledQuery
- DeleteScheduledQuery
- DescribeAccountSettings

- DescribeEndpoints
- DescribeScheduledQuery
- ExecuteScheduledQuery
- ListScheduledQueries
- ListTagsForResource
- PrepareQuery
- Query
- TagResource
- UntagResource
- UpdateAccountSettings
- UpdateScheduledQuery

CancelQuery
Servizio: Amazon Timestream Query

Annulla una richiesta che è stata emessa. L'annullamento viene fornito solo se l'esecuzione della richiesta non è stata completata prima dell'emissione della richiesta di cancellazione. Poiché l'annullamento è un'operazione idempotente, le richieste di annullamento successive restituiranno a`CancellationMessage`, a indicare che la query è già stata annullata. Consulta l'esempio di codice per i dettagli.

Sintassi della richiesta

```
{
   "QueryId": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel formato. JSON

QueryId

L'ID della query che deve essere annullata. `QueryID`viene restituito come parte del risultato della query.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 64 caratteri.

Modello: `[a-zA-Z0-9]+`

Campo obbligatorio: sì

Sintassi della risposta

```
{
   "CancellationMessage": "string"
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

CancellationMessage

A `CancellationMessage` viene restituito quando è già `QueryId` stata emessa una `CancelQuery` richiesta per la query specificata da.

Tipo: stringa

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Il servizio non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

CreateScheduledQuery

Servizio: Amazon Timestream Query

Creare una query pianificata che verrà eseguita per conto dell'utente in base alla pianificazione configurata. Per l'esecuzione della query, a Timestream viene assegnato il ruolo di esecutore nell'ambito del parametro `ScheduledQueryExecutionRoleArn`. È possibile utilizzare il parametro `NotificationConfiguration` per configurare l'invio di una notifica per le operazioni di query pianificate.

Sintassi della richiesta

```
{
   "ClientToken": "string",
   "ErrorReportConfiguration": {
      "S3Configuration": {
         "BucketName": "string",
         "EncryptionOption": "string",
         "ObjectKeyPrefix": "string"
      }
   },
   "KmsKeyId": "string",
   "Name": "string",
   "NotificationConfiguration": {
      "SnsConfiguration": {
         "TopicArn": "string"
      }
   },
   "QueryString": "string",
   "ScheduleConfiguration": {
      "ScheduleExpression": "string"
   },
   "ScheduledQueryExecutionRoleArn": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ],
   "TargetConfiguration": {
      "TimestreamConfiguration": {
         "DatabaseName": "string",
         "DimensionMappings": [
            {
```

```
                "DimensionValueType": "string",
                "Name": "string"
             }
          ],
          "MeasureNameColumn": "string",
          "MixedMeasureMappings": [
             {
                "MeasureName": "string",
                "MeasureValueType": "string",
                "MultiMeasureAttributeMappings": [
                   {
                      "MeasureValueType": "string",
                      "SourceColumn": "string",
                      "TargetMultiMeasureAttributeName": "string"
                   }
                ],
                "SourceColumn": "string",
                "TargetMeasureName": "string"
             }
          ],
          "MultiMeasureMappings": {
             "MultiMeasureAttributeMappings": [
                {
                   "MeasureValueType": "string",
                   "SourceColumn": "string",
                   "TargetMultiMeasureAttributeName": "string"
                }
             ],
             "TargetMultiMeasureName": "string"
          },
          "TableName": "string",
          "TimeColumn": "string"
       }
    }
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

ClientToken

L'uso di a ClientToken rende la chiamata CreateScheduledQuery idempotente, in altre parole, ripetere la stessa richiesta produrrà lo stesso risultato. Effettuare più CreateScheduledQuery richieste identiche ha lo stesso effetto di fare una singola richiesta.

- Se CreateScheduledQuery viene chiamata senza un`ClientToken`, la Query SDK genera un messaggio `ClientToken` per conto dell'utente.
- Dopo 8 ore, qualsiasi richiesta con lo stesso `ClientToken` viene considerata e gestita come una nuova richiesta.

Tipo: stringa

Vincoli di lunghezza: lunghezza minima di 32. La lunghezza massima è 128 caratteri.

Campo obbligatorio: no

ErrorReportConfiguration

Configurazione della segnalazione di errori. I report di errore vengono generati quando si verifica un problema durante la scrittura dei risultati della query.

Tipo: oggetto ErrorReportConfiguration

Campo obbligatorio: sì

KmsKeyId

La KMS chiave Amazon utilizzata per crittografare la risorsa di query pianificata, a riposo. Se la KMS chiave Amazon non è specificata, la risorsa di query pianificata verrà crittografata con una chiave Amazon KMS di proprietà di Timestream. Per specificare una KMS chiave, usa l'ID della chiaveARN, il nome alias o l'alias. ARN Quando si utilizza un nome alias, aggiungere al nome il prefisso alias/

Se viene ErrorReportConfiguration utilizzato `SSE_KMS` come tipo di crittografia, lo stesso KmsKeyId viene utilizzato per crittografare il rapporto di errore inattivo.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: no

Name

Nome dell'interrogazione pianificata.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 64 caratteri.

Modello: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Campo obbligatorio: sì

NotificationConfiguration

Configurazione della notifica per la query pianificata. Quando l'esecuzione di una query pianificata termina oppure lo stato della query viene aggiornato oppure la query viene eliminata, Timestream invia una notifica.

Tipo: oggetto NotificationConfiguration

Campo obbligatorio: sì

QueryString

La stringa query da eseguire. I nomi dei parametri possono essere specificati nella stringa query dal carattere `@` seguito da un identificatore. Il parametro denominato `@scheduled_runtime` è riservato e può essere utilizzato nella query per ottenere l'ora in cui è pianificata l'esecuzione della query.

Il timestamp calcolato in base al ScheduleConfiguration parametro sarà il valore del `@scheduled_runtime` parametro per ogni esecuzione di query. Ad esempio, si può considerare un'istanza di una query pianificata in esecuzione alla data/ora 2021-12-01 00:00:00. In questa istanza, il parametro `@scheduled_runtime` viene inizializzato in corrispondenza del timestamp 2021-12-01 00:00:00 quando si richiama la query.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. Lunghezza massima di 262144.

Campo obbligatorio: sì

ScheduleConfiguration

La configurazione della pianificazione per la query.

Tipo: oggetto ScheduleConfiguration

Campo obbligatorio: sì

ScheduledQueryExecutionRoleArn

Il ARN per il IAM ruolo che Timestream assumerà durante l'esecuzione della query pianificata.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: sì

Tags

Un elenco di coppie chiave-valore per etichettare la query pianificata.

Tipo: matrice di oggetti Tag

Membri dell'array: numero minimo di 0 elementi. Numero massimo di 200 elementi.

Campo obbligatorio: no

TargetConfiguration

Configurazione utilizzata per scrivere il risultato di una query.

Tipo: oggetto TargetConfiguration

Campo obbligatorio: no

Sintassi della risposta

```
{
   "Arn": "string"
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

Arn

ARNper l'interrogazione pianificata creata.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

ConflictException

Impossibile visualizzare i risultati del sondaggio per una query annullata.

HTTPCodice di stato: 400

InternalServerException

Il servizio non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ServiceQuotaExceededException

Hai superato la quota di servizio.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

DeleteScheduledQuery
Servizio: Amazon Timestream Query

Elimina una determinata query pianificata. Si tratta di un'operazione irreversibile.

Sintassi della richiesta

```
{
   "ScheduledQueryArn": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

ScheduledQueryArn

L'ARN della query pianificata.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: sì

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200 con un HTTP corpo vuoto.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Il servizio non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

Impossibile trovare la risorsa richiesta.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3

- AWS SDKper Python
- AWS SDKper Ruby V3

DescribeAccountSettings
Servizio: Amazon Timestream Query

Descrive le impostazioni dell'account che includono il modello di determinazione dei prezzi delle query e il numero massimo configurato che TCUs il servizio può utilizzare per il carico di lavoro relativo alle query.

Ti viene addebitata solo la durata delle unità di calcolo utilizzate per i tuoi carichi di lavoro.

Sintassi della risposta

```
{
   "MaxQueryTCU": number,
   "QueryPricingModel": "string"
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

MaxQueryTCU

Il numero massimo di unità di calcolo Timestream (TCUs) che il servizio utilizzerà in qualsiasi momento per rispondere alle tue domande.

Tipo: integer

QueryPricingModel

Il modello di prezzo per le domande nel tuo account.

Tipo: stringa

Valori validi: `BYTES_SCANNED | COMPUTE_UNITS`

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Il servizio non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

DescribeEndpoints
Servizio: Amazon Timestream Query

DescribeEndpoints restituisce un elenco di endpoint disponibili su cui effettuare chiamate TimestreamAPI. Questo API è disponibile sia tramite Write che Query.

Poiché i Timestream SDKs sono progettati per funzionare in modo trasparente con l'architettura del servizio, inclusa la gestione e la mappatura degli endpoint del servizio, non è consigliabile utilizzarli a meno che: API

- Stai usando VPC endpoints () con Timestream AWS PrivateLink
- L'applicazione utilizza un linguaggio di programmazione che non supporta ancora SDK
- È necessario un controllo migliore sull'implementazione lato client

Per informazioni dettagliate su come e quando utilizzare e implementare DescribeEndpoints, consulta The Endpoint Discovery Pattern.

Sintassi della risposta

```
{
   "Endpoints": [
      {
         "Address": "string",
         "CachePeriodInMinutes": number
      }
   ]
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

Endpoints

Un `Endpoints` oggetto viene restituito quando viene effettuata una `DescribeEndpoints` richiesta.

Tipo: matrice di oggetti Endpoint

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

InternalServerException

Il servizio non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

DescribeScheduledQuery
Servizio: Amazon Timestream Query

Fornisce informazioni dettagliate su un'interrogazione pianificata.

Sintassi della richiesta

```
{
   "ScheduledQueryArn": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

ScheduledQueryArn

L'ARN della query pianificata.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: sì

Sintassi della risposta

```
{
   "ScheduledQuery": {
      "Arn": "string",
      "CreationTime": number,
      "ErrorReportConfiguration": {
         "S3Configuration": {
            "BucketName": "string",
            "EncryptionOption": "string",
            "ObjectKeyPrefix": "string"
         }
      },
      "KmsKeyId": "string",
      "LastRunSummary": {
         "ErrorReportLocation": {
```

```
            "S3ReportLocation": {
               "BucketName": "string",
               "ObjectKey": "string"
            }
         },
         "ExecutionStats": {
            "BytesMetered": number,
            "CumulativeBytesScanned": number,
            "DataWrites": number,
            "ExecutionTimeInMillis": number,
            "QueryResultRows": number,
            "RecordsIngested": number
         },
         "FailureReason": "string",
         "InvocationTime": number,
         "QueryInsightsResponse": {
            "OutputBytes": number,
            "OutputRows": number,
            "QuerySpatialCoverage": {
               "Max": {
                  "PartitionKey": [ "string" ],
                  "TableArn": "string",
                  "Value": number
               }
            },
            "QueryTableCount": number,
            "QueryTemporalRange": {
               "Max": {
                  "TableArn": "string",
                  "Value": number
               }
            }
         },
         "RunStatus": "string",
         "TriggerTime": number
      },
      "Name": "string",
      "NextInvocationTime": number,
      "NotificationConfiguration": {
         "SnsConfiguration": {
            "TopicArn": "string"
         }
      },
      "PreviousInvocationTime": number,
```

```
      "QueryString": "string",
      "RecentlyFailedRuns": [
         {
            "ErrorReportLocation": {
               "S3ReportLocation": {
                  "BucketName": "string",
                  "ObjectKey": "string"
               }
            },
            "ExecutionStats": {
               "BytesMetered": number,
               "CumulativeBytesScanned": number,
               "DataWrites": number,
               "ExecutionTimeInMillis": number,
               "QueryResultRows": number,
               "RecordsIngested": number
            },
            "FailureReason": "string",
            "InvocationTime": number,
            "QueryInsightsResponse": {
               "OutputBytes": number,
               "OutputRows": number,
               "QuerySpatialCoverage": {
                  "Max": {
                     "PartitionKey": [ "string" ],
                     "TableArn": "string",
                     "Value": number
                  }
               },
               "QueryTableCount": number,
               "QueryTemporalRange": {
                  "Max": {
                     "TableArn": "string",
                     "Value": number
                  }
               }
            },
            "RunStatus": "string",
            "TriggerTime": number
         }
      ],
      "ScheduleConfiguration": {
         "ScheduleExpression": "string"
      },
```

```
      "ScheduledQueryExecutionRoleArn": "string",
      "State": "string",
      "TargetConfiguration": {
         "TimestreamConfiguration": {
            "DatabaseName": "string",
            "DimensionMappings": [
               {
                  "DimensionValueType": "string",
                  "Name": "string"
               }
            ],
            "MeasureNameColumn": "string",
            "MixedMeasureMappings": [
               {
                  "MeasureName": "string",
                  "MeasureValueType": "string",
                  "MultiMeasureAttributeMappings": [
                     {
                        "MeasureValueType": "string",
                        "SourceColumn": "string",
                        "TargetMultiMeasureAttributeName": "string"
                     }
                  ],
                  "SourceColumn": "string",
                  "TargetMeasureName": "string"
               }
            ],
            "MultiMeasureMappings": {
               "MultiMeasureAttributeMappings": [
                  {
                     "MeasureValueType": "string",
                     "SourceColumn": "string",
                     "TargetMultiMeasureAttributeName": "string"
                  }
               ],
               "TargetMultiMeasureName": "string"
            },
            "TableName": "string",
            "TimeColumn": "string"
         }
      }
   }
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

ScheduledQuery

L'interrogazione pianificata.

Tipo: oggetto ScheduledQueryDescription

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Il servizio non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

Impossibile trovare la risorsa richiesta.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

ExecuteScheduledQuery
Servizio: Amazon Timestream Query

È possibile utilizzarlo API per eseguire manualmente una query pianificata.

Se abilitato`QueryInsights`, restituisce API anche informazioni e metriche relative alla query che hai eseguito come parte di una SNS notifica Amazon. `QueryInsights`aiuta a ottimizzare le prestazioni della tua query. Per ulteriori informazioni su`QueryInsights`, consulta Usare le informazioni sulle query per ottimizzare le query in Amazon Timestream.

Sintassi della richiesta

```
{
   "ClientToken": "string",
   "InvocationTime": number,
   "QueryInsights": {
      "Mode": "string"
   },
   "ScheduledQueryArn": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel formato. JSON

ClientToken

Non utilizzato.

Tipo: stringa

Vincoli di lunghezza: lunghezza minima di 32. La lunghezza massima è 128 caratteri.

Campo obbligatorio: no

InvocationTime

Il timestamp in. UTC La query verrà eseguita come se fosse stata richiamata in questo timestamp.

Tipo: Timestamp

Campo obbligatorio: sì

QueryInsights

Incapsula le impostazioni per l'attivazione. `QueryInsights`

Attivazione di approfondimenti e metriche sui `QueryInsights` resi come parte della SNS notifica Amazon per la query che hai eseguito. Puoi utilizzarla `QueryInsights` per ottimizzare le prestazioni e i costi delle query.

Tipo: oggetto ScheduledQueryInsights

Campo obbligatorio: no

ScheduledQueryArn

ARNdella query pianificata.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: sì

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200 con un HTTP corpo vuoto.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Il servizio non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

Impossibile trovare la risorsa richiesta.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Esempi

Messaggio di notifica di interrogazione pianificata per la CONTROL modalità ENABLED WITH _ RATE _ _

L'esempio seguente mostra un messaggio di notifica di interrogazione pianificata riuscita per la `ENABLED_WITH_RATE_CONTROL` modalità del `QueryInsights` parametro.

```
"SuccessNotificationMessage": {
    "type": "MANUAL_TRIGGER_SUCCESS",
    "arn": "arn:aws:timestream:<Region>:<Account>:scheduled-query/sq-test-49c6ed55-
c2e7-4cc2-9956-4a0ecea13420-80e05b035236a4c3",
    "scheduledQueryRunSummary": {
        "invocationEpochSecond": 1723710546,
        "triggerTimeMillis": 1723710547490,
        "runStatus": "MANUAL_TRIGGER_SUCCESS",
        "executionStats": {
            "executionTimeInMillis": 17343,
            "dataWrites": 1024,
```

```
            "bytesMetered": 0,
            "cumulativeBytesScanned": 600,
            "recordsIngested": 1,
            "queryResultRows": 1
        },
        "queryInsightsResponse": {
            "querySpatialCoverage": {
                "max": {
                    "value": 1.0,
                    "tableArn": "arn:aws:timestream:<Region>:<Account>:database/BaseDb/
table/BaseTable",
                    "partitionKey": [
                        "measure_name"
                    ]
                }
            },
            "queryTemporalRange": {
                "max": {
                    "value": 2399999999999,
                    "tableArn": "arn:aws:timestream:<Region>:<Account>:database/BaseDb/
table/BaseTable"
                }
            },
            "queryTableCount": 1,
            "outputRows": 1,
            "outputBytes": 59
        }
    }
}
```

Messaggio di notifica delle interrogazioni pianificate per la DISABLED modalità

L'esempio seguente mostra un messaggio di notifica di interrogazione pianificata riuscita per la `DISABLED` modalità del `QueryInsights` parametro.

```
"SuccessNotificationMessage": {
    "type": "MANUAL_TRIGGER_SUCCESS",
    "arn": "arn:aws:timestream:<Region>:<Account>:scheduled-query/sq-test-
fa109d9e-6528-4a0d-ac40-482fa05e657f-140faaeecdc5b2a7",
    "scheduledQueryRunSummary": {
        "invocationEpochSecond": 1723711401,
        "triggerTimeMillis": 1723711402144,
        "runStatus": "MANUAL_TRIGGER_SUCCESS",
```

```
        "executionStats": {
            "executionTimeInMillis": 17992,
            "dataWrites": 1024,
            "bytesMetered": 0,
            "cumulativeBytesScanned": 600,
            "recordsIngested": 1,
            "queryResultRows": 1
        }
    }
}
```

Messaggio di notifica di errore per la CONTROL modalità ENABLED WITH _ RATE _ _

L'esempio seguente mostra un messaggio di notifica di interrogazione pianificata non riuscita per la `ENABLED_WITH_RATE_CONTROL` modalità del `QueryInsights` parametro.

```
"FailureNotificationMessage": {
    "type": "MANUAL_TRIGGER_FAILURE",
    "arn": "arn:aws:timestream:<Region>:<Account>:scheduled-query/sq-test-
b261670d-790c-4116-9db5-0798071b18b1-b7e27a1d79be226d",
    "scheduledQueryRunSummary": {
        "invocationEpochSecond": 1727915513,
        "triggerTimeMillis": 1727915513894,
        "runStatus": "MANUAL_TRIGGER_FAILURE",
        "executionStats": {
            "executionTimeInMillis": 10777,
            "dataWrites": 0,
            "bytesMetered": 0,
            "cumulativeBytesScanned": 0,
            "recordsIngested": 0,
            "queryResultRows": 4
        },
        "errorReportLocation": {
            "s3ReportLocation": {
                "bucketName": "my-amzn-s3-demo-bucket",
                "objectKey": "4my-organization-f7a3c5d065a1a95e/1727915513/
MANUAL/1727915513894/5e14b3df-b147-49f4-9331-784f749b68ae"
            }
        },
        "failureReason": "Schedule encountered some errors and is incomplete. Please
 take a look at error report for further details"
    }
}
```

Messaggio di notifica di errore per la DISABLED modalità

L'esempio seguente mostra un messaggio di notifica di interrogazione pianificata non riuscita per la `DISABLED` modalità del `QueryInsights` parametro.

```
"FailureNotificationMessage": {
    "type": "MANUAL_TRIGGER_FAILURE",
    "arn": "arn:aws:timestream:<Region>:<Account>:scheduled-query/sq-test-
b261670d-790c-4116-9db5-0798071b18b1-b7e27a1d79be226d",
    "scheduledQueryRunSummary": {
        "invocationEpochSecond": 1727915194,
        "triggerTimeMillis": 1727915195119,
        "runStatus": "MANUAL_TRIGGER_FAILURE",
        "executionStats": {
            "executionTimeInMillis": 10777,
            "dataWrites": 0,
            "bytesMetered": 0,
            "cumulativeBytesScanned": 0,
            "recordsIngested": 0,
            "queryResultRows": 4
        },
        "errorReportLocation": {
            "s3ReportLocation": {
                "bucketName": "my-amzn-s3-demo-bucket",
                "objectKey": "4my-organization-b7e27a1d79be226d/1727915194/
MANUAL/1727915195119/08dea9f5-9a0a-4e63-a5f7-ded23247bb98"
            }
        },
        "failureReason": "Schedule encountered some errors and is incomplete. Please
 take a look at error report for further details"
    }
}
```

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API codice in una delle lingue specifiche AWS SDKs, consultate quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2

- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

ListScheduledQueries
Servizio: Amazon Timestream Query

Ottiene un elenco di tutte le domande pianificate nell'account Amazon e nella regione del chiamante. `ListScheduledQueries`alla fine è coerente.

Sintassi della richiesta

```
{
   "MaxResults": number,
   "NextToken": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

MaxResults

Il numero massimo di elementi da restituire nell'output. Se il numero totale di articoli disponibili è superiore al valore specificato, nell'output `NextToken` viene fornito un. Per riprendere l'impaginazione, fornite il `NextToken` valore come argomento per la successiva chiamata a. `ListScheduledQueriesRequest`

Tipo: integer

Intervallo valido: valore minimo di 1. Valore massimo pari a 1000.

Campo obbligatorio: no

NextToken

Un token di impaginazione per riprendere l'impaginazione.

Tipo: string

Campo obbligatorio: no

Sintassi della risposta

```
{
```

```
   "NextToken": "string",
   "ScheduledQueries": [
      {
         "Arn": "string",
         "CreationTime": number,
         "ErrorReportConfiguration": {
            "S3Configuration": {
               "BucketName": "string",
               "EncryptionOption": "string",
               "ObjectKeyPrefix": "string"
            }
         },
         "LastRunStatus": "string",
         "Name": "string",
         "NextInvocationTime": number,
         "PreviousInvocationTime": number,
         "State": "string",
         "TargetDestination": {
            "TimestreamDestination": {
               "DatabaseName": "string",
               "TableName": "string"
            }
         }
      }
   ]
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una HTTP risposta di 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

NextToken

Token per specificare dove iniziare l'impaginazione. Si tratta NextToken di una risposta precedentemente troncata.

Tipo: stringa

ScheduledQueries

Un elenco di interrogazioni pianificate.

Tipo: matrice di oggetti ScheduledQuery

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Il servizio non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2

- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

ListTagsForResource
Servizio: Amazon Timestream Query

Elenca tutti i tag su una risorsa di query Timestream.

Sintassi della richiesta

```
{
   "MaxResults": number,
   "NextToken": "string",
   "ResourceARN": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

MaxResults

Il numero massimo di tag da restituire.

Tipo: integer

Intervallo valido: valore minimo di 1. valore massimo pari a 200.

Campo obbligatorio: no

NextToken

Un token di impaginazione per riprendere l'impaginazione.

Tipo: string

Campo obbligatorio: no

ResourceARN

La risorsa Timestream con i tag da elencare. Questo valore è un Amazon Resource Name (ARN).

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: sì

Sintassi della risposta

```
{
   "NextToken": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

NextToken

Un token di impaginazione per riprendere l'impaginazione con una successiva chiamata a. `ListTagsForResourceResponse`

Tipo: stringa

Tags

I tag attualmente associati alla risorsa Timestream.

Tipo: matrice di oggetti Tag

Membri dell'array: numero minimo di 0 elementi. Numero massimo di 200 elementi.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

Impossibile trovare la risorsa richiesta.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

PrepareQuery
Servizio: Amazon Timestream Query

Un'operazione sincrona che consente di inviare una query con parametri da archiviare da Timestream per un'esecuzione successiva. Timestream supporta l'utilizzo di questa operazione solo con set to. `ValidateOnly` `true`

Sintassi della richiesta

```
{
   "QueryString": "string",
   "ValidateOnly": boolean
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

QueryString

La stringa di query Timestream che si desidera utilizzare come istruzione preparata. I nomi dei parametri possono essere specificati nella stringa query dal carattere @ seguito da un identificatore.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. Lunghezza massima di 262144.

Campo obbligatorio: sì

ValidateOnly

Impostando questo valore su`true`, Timestream convaliderà solo che la stringa di query sia una query Timestream valida e non memorizzerà la query preparata per un uso successivo.

Tipo: Booleano

Campo obbligatorio: no

Sintassi della risposta

```
{
   "Columns": [
      {
         "Aliased": boolean,
         "DatabaseName": "string",
         "Name": "string",
         "TableName": "string",
         "Type": {
            "ArrayColumnInfo": {
               "Name": "string",
               "Type": "Type"
            },
            "RowColumnInfo": [
               {
                  "Name": "string",
                  "Type": "Type"
               }
            ],
            "ScalarType": "string",
            "TimeSeriesMeasureValueColumnInfo": {
               "Name": "string",
               "Type": "Type"
            }
         }
      }
   ],
   "Parameters": [
      {
         "Name": "string",
         "Type": {
            "ArrayColumnInfo": {
               "Name": "string",
               "Type": "Type"
            },
            "RowColumnInfo": [
               {
                  "Name": "string",
                  "Type": "Type"
               }
            ],
            "ScalarType": "string",
            "TimeSeriesMeasureValueColumnInfo": {
```

```
                "Name": "string",
                "Type": "Type"
            }
        }
    }
  ],
  "QueryString": "string"
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di 200. HTTP

I seguenti dati vengono restituiti in JSON formato dal servizio.

Columns

Un elenco di colonne di SELECT clausole della stringa di query inviata.

Tipo: matrice di oggetti SelectColumn

Parameters

Un elenco di parametri utilizzati nella stringa di query inviata.

Tipo: matrice di oggetti ParameterMapping

QueryString

La stringa di query che si desidera preparare.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. Lunghezza massima di 262144.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Il servizio non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

Query

Servizio: Amazon Timestream Query

`Query`è un'operazione sincrona che consente di eseguire una query sui dati di Amazon Timestream.

Se abilitata`QueryInsights`, restituisce API anche informazioni e metriche relative alla query che hai eseguito. `QueryInsights`saiuta a ottimizzare le prestazioni della query. Per ulteriori informazioni su`QueryInsights`, consulta Usare le informazioni sulle query per ottimizzare le query in Amazon Timestream.

> **Note**
>
> Il numero massimo di `Query` API richieste che puoi effettuare con `QueryInsights` enabled è 1 query al secondo (). QPS Se superi questa frequenza di query, potrebbe verificarsi una limitazione.

`Query`scadrà dopo 60 secondi. È necessario aggiornare il timeout predefinito in SDK per supportare un timeout di 60 secondi. Per i dettagli, consulta l'esempio di codice.

La richiesta di interrogazione avrà esito negativo nei seguenti casi:

- Se invii una `Query` richiesta con lo stesso token client al di fuori della finestra di idempotenza di 5 minuti.
- Se invii una `Query` richiesta con lo stesso token client, ma modifichi altri parametri, entro la finestra di idempotenza di 5 minuti.
- Se la dimensione della riga (inclusi i metadati della query) supera 1 MB, la query avrà esito negativo e verrà visualizzato il seguente messaggio di errore:

  ```
  Query aborted as max page response size has been exceeded by the output result row
  ```
- Se il IAM principale dell'iniziatore della query e del lettore dei risultati non coincidono e/o l'iniziatore della query e il lettore dei risultati non hanno la stessa stringa di query nelle richieste di query, la query avrà esito negativo e restituirà un errore. `Invalid pagination token`

Sintassi della richiesta

```
{
```

```
   "ClientToken": "string",
   "MaxRows": number,
   "NextToken": "string",
   "QueryInsights": {
      "Mode": "string"
   },
   "QueryString": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

ClientToken

Stringa univoca con distinzione tra maiuscole e minuscole, composta da un massimo di 64 ASCII caratteri, specificata quando viene effettuata una `Query` richiesta. L'immissione di un *`ClientToken`* rende la chiamata idempotente. *`Query`* Ciò significa che l'esecuzione ripetuta della stessa query produrrà lo stesso risultato. In altre parole, fare più `Query` richieste identiche ha lo stesso effetto di fare una singola richiesta. Quando si utilizza `ClientToken` in una query, tenete presente quanto segue:

- Se la Query API viene istanziata senza a`ClientToken`, la Query SDK genera un'istanza per `ClientToken` conto dell'utente.
- Se la `Query` chiamata contiene solo `ClientToken` ma non include a`NextToken`, si presume che tale invocazione di `Query` sia l'esecuzione di una nuova query.
- Se la chiamata contiene`NextToken`, si presume che quella particolare invocazione sia una chiamata successiva di una precedente chiamata alla Query e viene restituito un set di risultati. API
- Dopo 4 ore, qualsiasi richiesta con lo stesso valore `ClientToken` viene trattata come una nuova richiesta.

Tipo: stringa

Vincoli di lunghezza: lunghezza minima di 32. La lunghezza massima è 128 caratteri.

Campo obbligatorio: no

MaxRows

Il numero totale di righe da restituire nell'`Query`output. L'esecuzione iniziale di `Query` con un `MaxRows` valore specificato restituirà il set di risultati della query in due casi:

- La dimensione del risultato è inferiore a`1MB`.
- Il numero di righe nel set di risultati è inferiore al valore di`maxRows`.

In caso contrario, l'invocazione iniziale di restituisce `Query` solo a`NextToken`, che può quindi essere utilizzato nelle chiamate successive per recuperare il set di risultati. Per riprendere l'impaginazione, fornite il `NextToken` valore nel comando successivo.

Se la dimensione della riga è grande (ad esempio una riga ha molte colonne), Timestream può restituire un numero inferiore di righe per evitare che la dimensione della risposta superi il limite di 1 MB. Se non `MaxRows` viene fornito, Timestream invierà il numero di righe necessario per soddisfare il limite di 1 MB.

Tipo: integer

Intervallo valido: valore minimo di 1. Valore massimo pari a 1000.

Campo obbligatorio: no

NextToken

Un token di impaginazione utilizzato per restituire una serie di risultati. Quando `Query` API viene richiamato utilizzando`NextToken`, si presume che quella particolare invocazione sia una chiamata successiva di una chiamata precedente a e viene restituito un set di `Query` risultati. Tuttavia, se la `Query` chiamata contiene solo il, si presume che tale invocazione sia una nuova esecuzione di `Query` una query. `ClientToken`

Quando si utilizza NextToken in una query, tenete presente quanto segue:

- Un token di impaginazione può essere utilizzato per un massimo di cinque `Query` invocazioni, OPPURE per una durata massima di 1 ora, a seconda dell'evento che si verifica per primo.
- L'utilizzo dello stesso `NextToken` restituirà lo stesso set di record. Per continuare a sfogliare il set di risultati, è necessario utilizzare i più recenti. `nextToken`
- Supponiamo che una `Query` chiamata restituisca due valori e. `NextToken` `TokenA` `TokenB` Se `TokenB` viene utilizzato in una `Query` chiamata successiva, viene invalidato e non può essere `TokenA` riutilizzato.
- Per richiedere un set di risultati precedente da una query dopo l'inizio dell'impaginazione, è necessario richiamare nuovamente la Query. API

- L'ultimo `NextToken` dovrebbe essere usato per impaginare fino a quando non `null` viene restituito, a quel punto dovrebbe essere usato un nuovo`NextToken`.
- Se il IAM principale dell'iniziatore della query e del lettore dei risultati non sono gli stessi e/o l'iniziatore della query e il lettore dei risultati non hanno la stessa stringa di query nelle richieste di query, la query avrà esito negativo e verrà generato un errore. `Invalid pagination token`

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: no

QueryInsights

Incapsula le impostazioni per l'attivazione. `QueryInsights`

L'attivazione `QueryInsights` restituisce approfondimenti e metriche oltre ai risultati delle query per la query che hai eseguito. È possibile utilizzarlo `QueryInsights` per ottimizzare le prestazioni delle query.

Tipo: oggetto QueryInsights

Campo obbligatorio: no

QueryString

La query che deve essere eseguita da Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. Lunghezza massima di 262144.

Campo obbligatorio: sì

Sintassi della risposta

```
{
   "ColumnInfo": [
      {
         "Name": "string",
         "Type": {
            "ArrayColumnInfo": "ColumnInfo",
            "RowColumnInfo": [
```

```
            "ColumnInfo"
         ],
         "ScalarType": "string",
         "TimeSeriesMeasureValueColumnInfo": "ColumnInfo"
      }
   }
],
"NextToken": "string",
"QueryId": "string",
"QueryInsightsResponse": {
   "OutputBytes": number,
   "OutputRows": number,
   "QuerySpatialCoverage": {
      "Max": {
         "PartitionKey": [ "string" ],
         "TableArn": "string",
         "Value": number
      }
   },
   "QueryTableCount": number,
   "QueryTemporalRange": {
      "Max": {
         "TableArn": "string",
         "Value": number
      }
   },
   "UnloadPartitionCount": number,
   "UnloadWrittenBytes": number,
   "UnloadWrittenRows": number
},
"QueryStatus": {
   "CumulativeBytesMetered": number,
   "CumulativeBytesScanned": number,
   "ProgressPercentage": number
},
"Rows": [
   {
      "Data": [
         {
            "ArrayValue": [
               "Datum"
            ],
            "NullValue": boolean,
            "RowValue": "Row",
```

```
                "ScalarValue": "string",
                "TimeSeriesValue": [
                   {
                      "Time": "string",
                      "Value": "Datum"
                   }
                ]
             }
          ]
       }
    ]
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

ColumnInfo

I tipi di dati delle colonne del set di risultati restituito.

Tipo: matrice di oggetti ColumnInfo

NextToken

Un token di impaginazione che può essere riutilizzato durante una `Query` chiamata per ottenere il successivo set di risultati.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

QueryId

Un ID univoco per la query specificata.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 64 caratteri.

Modello: `[a-zA-Z0-9]+`

QueryInsightsResponse

Incapsula `QueryInsights` contenente informazioni e metriche relative alla query che hai eseguito.

Tipo: oggetto QueryInsightsResponse

QueryStatus

Informazioni sullo stato della query, inclusi lo stato di avanzamento e i byte scansionati.

Tipo: oggetto QueryStatus

Rows

Le righe del set di risultati restituite dalla query.

Tipo: matrice di oggetti Row

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

ConflictException

Impossibile visualizzare i risultati del sondaggio per una query annullata.

HTTPCodice di stato: 400

InternalServerException

Il servizio non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

QueryExecutionException

Timestream non è riuscito a eseguire la query con successo.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

TagResource
Servizio: Amazon Timestream Query

Associa un set di tag a una risorsa Timestream. È quindi possibile attivare questi tag definiti dall'utente in modo che vengano visualizzati nella console di Billing and Cost Management per il monitoraggio dell'allocazione dei costi.

Sintassi della richiesta

```
{
   "ResourceARN": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel formato. JSON

ResourceARN

Identifica la risorsa Timestream a cui aggiungere i tag. Questo valore è un Amazon Resource Name (ARN).

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: sì

Tags

I tag da assegnare alla risorsa Timestream.

Tipo: matrice di oggetti Tag

Membri dell'array: numero minimo di 0 elementi. Numero massimo di 200 elementi.

Campo obbligatorio: sì

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200 con un corpo vuotoHTTP.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

Impossibile trovare la risorsa richiesta.

HTTPCodice di stato: 400

ServiceQuotaExceededException

Hai superato la quota di servizio.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

UntagResource
Servizio: Amazon Timestream Query

Rimuove l'associazione di tag da una risorsa di query Timestream.

Sintassi della richiesta

```
{
   "ResourceARN": "string",
   "TagKeys": [ "string" ]
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

ResourceARN

La risorsa Timestream da cui verranno rimossi i tag. Questo valore è un Amazon Resource Name (ARN).

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: sì

TagKeys

Un elenco di tag e chiavi. I tag esistenti della risorsa le cui chiavi sono membri di questo elenco verranno rimossi dalla risorsa Timestream.

Tipo: matrice di stringhe

Membri dell'array: numero minimo di 0 elementi. Numero massimo di 200 elementi.

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 128 caratteri.

Campo obbligatorio: sì

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta HTTP 200 con un corpo vuotoHTTP.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

Impossibile trovare la risorsa richiesta.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3

- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

UpdateAccountSettings
Servizio: Amazon Timestream Query

Trasforma il tuo account in modo da utilizzarlo TCUs per la determinazione dei prezzi delle query e modifica il numero massimo di unità di calcolo per le query che hai configurato. Se riduci il valore di `MaxQueryTCU` alla configurazione desiderata, il nuovo valore può richiedere fino a 24 ore per essere efficace.

> **Note**
>
> Dopo aver trasferito il tuo account all'utilizzo TCUs per la determinazione dei prezzi delle query, non puoi passare all'utilizzo dei byte scansionati per la determinazione dei prezzi delle query.

Sintassi della richiesta

```
{
   "MaxQueryTCU": number,
   "QueryPricingModel": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel formato. JSON

MaxQueryTCU

Il numero massimo di unità di calcolo che il servizio utilizzerà in qualsiasi momento per rispondere alle richieste dell'utente. Per eseguire le query, è necessario impostare una capacità minima di 4. TCU È possibile impostare il numero massimo di TCU in multipli di 4, ad esempio 4, 8, 16, 32 e così via.

Il valore massimo supportato per `MaxQueryTCU` è 1000. Per richiedere un aumento di questo limite flessibile, contatta l' AWS assistenza. Per informazioni sulla quota predefinita per maxQueryTCU, consulta Quote predefinite.

Tipo: integer

Campo obbligatorio: no

QueryPricingModel

Il modello di prezzo per le richieste in un account.

> **Note**
>
> Il `QueryPricingModel` parametro viene utilizzato da diverse operazioni di Timestream; tuttavia, l'`UpdateAccountSettings`APIoperazione non riconosce alcun valore diverso da. `COMPUTE_UNITS`

Tipo: stringa

Valori validi: `BYTES_SCANNED | COMPUTE_UNITS`

Campo obbligatorio: no

Sintassi della risposta

```
{
   "MaxQueryTCU": number,
   "QueryPricingModel": "string"
}
```

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200.

I seguenti dati vengono restituiti in JSON formato dal servizio.

MaxQueryTCU

Il numero massimo configurato di unità di calcolo che il servizio utilizzerà in qualsiasi momento per rispondere alle richieste dell'utente.

Tipo: integer

QueryPricingModel

Il modello di prezzo per un account.

Tipo: stringa

Valori validi: `BYTES_SCANNED | COMPUTE_UNITS`

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Il servizio non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS

- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

UpdateScheduledQuery
Servizio: Amazon Timestream Query

Aggiorna una query pianificata.

Sintassi della richiesta

```
{
   "ScheduledQueryArn": "string",
   "State": "string"
}
```

Parametri della richiesta

Per informazioni sui parametri comuni per tutte le azioni, consulta Parametri comuni.

La richiesta accetta i seguenti dati nel JSON formato.

ScheduledQueryArn

ARNdella query pianificata.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: sì

State

Stato dell'interrogazione pianificata.

Tipo: stringa

Valori validi: `ENABLED | DISABLED`

Campo obbligatorio: sì

Elementi di risposta

Se l'azione ha esito positivo, il servizio restituisce una risposta di HTTP 200 con un HTTP corpo vuoto.

Errori

Per informazioni sugli errori comuni a tutte le operazioni, consultare Errori comuni.

AccessDeniedException

Non sei autorizzato a eseguire questa azione.

HTTPCodice di stato: 400

InternalServerException

Il servizio non è stato in grado di elaborare completamente questa richiesta a causa di un errore interno del server.

HTTPCodice di stato: 400

InvalidEndpointException

L'endpoint richiesto non era valido.

HTTPCodice di stato: 400

ResourceNotFoundException

Impossibile trovare la risorsa richiesta.

HTTPCodice di stato: 400

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationException

Richiesta non valida o non valida.

HTTPCodice di stato: 400

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- Interfaccia a riga di comando AWS
- AWS SDKper. NET
- AWS SDKper C++
- AWS SDKper Go v2
- AWS SDKper Java V2
- AWS SDKper JavaScript V3
- AWS SDKper PHP V3
- AWS SDKper Python
- AWS SDKper Ruby V3

Tipi di dati

I seguenti tipi di dati sono supportati da Amazon Timestream Write:

- BatchLoadProgressReport
- BatchLoadTask
- BatchLoadTaskDescription
- CsvConfiguration
- Database
- DataModel
- DataModelConfiguration
- DataModelS3Configuration
- DataSourceConfiguration
- DataSourceS3Configuration
- Dimension
- DimensionMapping
- Endpoint
- MagneticStoreRejectedDataLocation
- MagneticStoreWriteProperties
- MeasureValue
- MixedMeasureMapping
- MultiMeasureAttributeMapping

- MultiMeasureMappings
- PartitionKey
- Record
- RecordsIngested
- RejectedRecord
- ReportConfiguration
- ReportS3Configuration
- RetentionProperties
- S3Configuration
- Schema
- Table
- Tag

I seguenti tipi di dati sono supportati da Amazon Timestream Query:

- ColumnInfo
- Datum
- DimensionMapping
- Endpoint
- ErrorReportConfiguration
- ErrorReportLocation
- ExecutionStats
- MixedMeasureMapping
- MultiMeasureAttributeMapping
- MultiMeasureMappings
- NotificationConfiguration
- ParameterMapping
- QueryInsights
- QueryInsightsResponse
- QuerySpatialCoverage
- QuerySpatialCoverageMax

- QueryStatus
- QueryTemporalRange
- QueryTemporalRangeMax
- Row
- S3Configuration
- S3ReportLocation
- ScheduleConfiguration
- ScheduledQuery
- ScheduledQueryDescription
- ScheduledQueryInsights
- ScheduledQueryInsightsResponse
- ScheduledQueryRunSummary
- SelectColumn
- SnsConfiguration
- Tag
- TargetConfiguration
- TargetDestination
- TimeSeriesDataPoint
- TimestreamConfiguration
- TimestreamDestination
- Type

Amazon Timestream Write

I seguenti tipi di dati sono supportati da Amazon Timestream Write:

- BatchLoadProgressReport
- BatchLoadTask
- BatchLoadTaskDescription
- CsvConfiguration
- Database
- DataModel

- DataModelConfiguration
- DataModelS3Configuration
- DataSourceConfiguration
- DataSourceS3Configuration
- Dimension
- DimensionMapping
- Endpoint
- MagneticStoreRejectedDataLocation
- MagneticStoreWriteProperties
- MeasureValue
- MixedMeasureMapping
- MultiMeasureAttributeMapping
- MultiMeasureMappings
- PartitionKey
- Record
- RecordsIngested
- RejectedRecord
- ReportConfiguration
- ReportS3Configuration
- RetentionProperties
- S3Configuration
- Schema
- Table
- Tag

BatchLoadProgressReport
Servizio: Amazon Timestream Write

Dettagli sullo stato di avanzamento di un'operazione di caricamento in batch.

Indice

BytesMetered

Tipo: long

Campo obbligatorio: no

FileFailures

Tipo: long

Campo obbligatorio: no

ParseFailures

Tipo: long

Campo obbligatorio: no

RecordIngestionFailures

Tipo: long

Campo obbligatorio: no

RecordsIngested

Tipo: long

Campo obbligatorio: no

RecordsProcessed

Tipo: long

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, consultate quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

BatchLoadTask

Servizio: Amazon Timestream Write

Dettagli su un'operazione di caricamento in batch.

Indice

CreationTime

L'ora in cui è stata creata l'attività di caricamento in batch di Timestream.

Tipo: Timestamp

Campo obbligatorio: no

DatabaseName

Nome del database in cui un'operazione di caricamento in batch carica i dati.

Tipo: string

Campo obbligatorio: no

LastUpdatedTime

L'ora dell'ultimo aggiornamento dell'attività di caricamento in batch di Timestream.

Tipo: Timestamp

Campo obbligatorio: no

ResumableUntil

Tipo: Timestamp

Campo obbligatorio: no

TableName

Nome della tabella in cui un'operazione di caricamento in batch carica i dati.

Tipo: string

Campo obbligatorio: no

TaskId

L'ID dell'operazione di caricamento in batch.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 32 caratteri.

Modello: `[A-Z0-9]+`

Campo obbligatorio: no

TaskStatus

Stato dell'operazione di caricamento in batch.

Tipo: stringa

Valori validi: `CREATED | IN_PROGRESS | FAILED | SUCCEEDED | PROGRESS_STOPPED | PENDING_RESUME`

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, consultate quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

BatchLoadTaskDescription
Servizio: Amazon Timestream Write

Dettagli su un'operazione di caricamento in batch.

Indice

CreationTime

L'ora in cui è stata creata l'attività di caricamento in batch di Timestream.

Tipo: Timestamp

Campo obbligatorio: no

DataModelConfiguration

Configurazione del modello di dati per un'operazione di caricamento in batch. Contiene dettagli sulla posizione in cui è archiviato un modello di dati per un'operazione di caricamento in batch.

Tipo: oggetto DataModelConfiguration

Campo obbligatorio: no

DataSourceConfiguration

Dettagli di configurazione sull'origine dati per un'operazione di caricamento in batch.

Tipo: oggetto DataSourceConfiguration

Required: No

ErrorMessage

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: no

LastUpdatedTime

L'ora dell'ultimo aggiornamento dell'attività di caricamento in batch di Timestream.

Tipo: Timestamp

Campo obbligatorio: no

ProgressReport

Tipo: oggetto BatchLoadProgressReport

Campo obbligatorio: no

RecordVersion

Tipo: long

Campo obbligatorio: no

ReportConfiguration

Configurazione del report per un'attività di caricamento in batch. Contiene dettagli sulla posizione in cui vengono archiviate le segnalazioni di errori.

Tipo: oggetto ReportConfiguration

Campo obbligatorio: no

ResumableUntil

Tipo: Timestamp

Required: No

TargetDatabaseName

Tipo: string

Required: No

TargetTableName

Tipo: string

Campo obbligatorio: no

TaskId

L'ID dell'operazione di caricamento in batch.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 32 caratteri.

Modello: `[A-Z0-9]+`

Campo obbligatorio: no

TaskStatus

Stato dell'operazione di caricamento in batch.

Tipo: stringa

Valori validi: `CREATED | IN_PROGRESS | FAILED | SUCCEEDED | PROGRESS_STOPPED | PENDING_RESUME`

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, consultate quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

CsvConfiguration
Servizio: Amazon Timestream Write

Un formato di dati delimitato in cui il separatore di colonna può essere una virgola e il separatore di record è un carattere di nuova riga.

Indice

ColumnSeparator

Il separatore di colonna può essere composto da virgola (','), pipe ('|), punto e virgola (';'), tab ('/t') o spazio vuoto (").

Tipo: stringa

Vincoli di lunghezza: lunghezza fissa pari a 1.

Campo obbligatorio: no

EscapeChar

Il personaggio di fuga può essere uno dei

Tipo: stringa

Vincoli di lunghezza: lunghezza fissa pari a 1.

Campo obbligatorio: no

NullValue

Può essere uno spazio vuoto (").

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 256 caratteri.

Campo obbligatorio: no

QuoteChar

Può essere tra virgolette singole (') o doppie («).

Tipo: stringa

Vincoli di lunghezza: lunghezza fissa pari a 1.

Campo obbligatorio: no

TrimWhiteSpace

Specifica di tagliare gli spazi bianchi iniziali e finali.

Tipo: Booleano

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo valore API in una delle lingue AWS SDKs specifiche, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Database

Servizio: Amazon Timestream Write

Un contenitore di primo livello per una tabella. I database e le tabelle sono i concetti di gestione fondamentali in Amazon Timestream. Tutte le tabelle di un database sono crittografate con la stessa AWS KMS chiave.

Indice

Arn

L'Amazon Resource Name che identifica in modo univoco questo database.

Tipo: string

Campo obbligatorio: no

CreationTime

L'ora in cui è stato creato il database, calcolata a partire dall'epoca Unix.

Tipo: Timestamp

Campo obbligatorio: no

DatabaseName

Nome del database Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: no

KmsKeyId

L'identificatore della AWS KMS chiave utilizzata per crittografare i dati memorizzati nel database.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: no

LastUpdatedTime

L'ultima volta che questo database è stato aggiornato.

Tipo: Timestamp

Campo obbligatorio: no

TableCount

Il numero totale di tabelle trovate all'interno di un database Timestream.

Tipo: long

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo codice API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

DataModel
Servizio: Amazon Timestream Write

Modello di dati per un'attività di caricamento in batch.

Indice

DimensionMappings

Mappature da origine a destinazione per le dimensioni.

Tipo: matrice di oggetti DimensionMapping

Membri dell'array: numero minimo di 1 elemento.

Campo obbligatorio: sì

MeasureNameColumn

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 256 caratteri.

Campo obbligatorio: no

MixedMeasureMappings

Mappature da origine a destinazione per le misure.

Tipo: matrice di oggetti MixedMeasureMapping

Membri dell'array: numero minimo di 1 elemento.

Campo obbligatorio: no

MultiMeasureMappings

Mappature da origine a destinazione per record multimisura.

Tipo: oggetto MultiMeasureMappings

Campo obbligatorio: no

TimeColumn

Colonna di origine da mappare al tempo.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 256 caratteri.

Campo obbligatorio: no

TimeUnit

La granularità dell'unità timestamp. Indica se il valore temporale è in secondi, millisecondi, nanosecondi o altri valori supportati. Il valore predefinito è `MILLISECONDS`.

Tipo: stringa

Valori validi: `MILLISECONDS | SECONDS | MICROSECONDS | NANOSECONDS`

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo valore API in una delle lingue AWS SDKs specifiche, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

DataModelConfiguration
Servizio: Amazon Timestream Write

Indice

DataModel

Tipo: oggetto DataModel

Campo obbligatorio: no

DataModelS3Configuration

Tipo: oggetto DataModelS3Configuration

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

DataModelS3Configuration
Servizio: Amazon Timestream Write

Indice

BucketName

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 63 caratteri.

Modello: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

Required: No

ObjectKey

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 1024 caratteri.

Modello: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

DataSourceConfiguration
Servizio: Amazon Timestream Write

Definisce i dettagli di configurazione sull'origine dati.

Indice

DataFormat

Questo è attualmenteCSV.

Tipo: stringa

Valori validi: CSV

Campo obbligatorio: sì

DataSourceS3Configuration

Configurazione di una posizione S3 per un file che contiene dati da caricare.

Tipo: oggetto DataSourceS3Configuration

Campo obbligatorio: sì

CsvConfiguration

Un formato di dati delimitato in cui il separatore di colonna può essere una virgola e il separatore di record è un carattere di nuova riga.

Tipo: oggetto CsvConfiguration

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo codice API in una delle lingue AWS SDKs specifiche, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

DataSourceS3Configuration
Servizio: Amazon Timestream Write

Indice

BucketName

Il nome bucket del bucket S3 del cliente.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 63 caratteri.

Modello: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

Campo obbligatorio: sì

ObjectKeyPrefix

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 1024 caratteri.

Modello: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Dimension
Servizio: Amazon Timestream Write

Rappresenta gli attributi dei metadati delle serie temporali. Ad esempio, il nome e la zona di disponibilità di un'EC2istanza o il nome del produttore di una turbina eolica sono dimensioni.

Indice

Name

La dimensione rappresenta gli attributi dei metadati delle serie temporali. Ad esempio, il nome e la zona di disponibilità di un'EC2istanza o il nome del produttore di una turbina eolica sono dimensioni.

Per i vincoli sui nomi delle dimensioni, vedere Vincoli di denominazione.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. Lunghezza massima di 60.

Campo obbligatorio: sì

Value

Il valore della dimensione.

Tipo: stringa

Campo obbligatorio: sì

DimensionValueType

Il tipo di dati della dimensione per il punto dati della serie temporale.

Tipo: stringa

Valori validi: `VARCHAR`

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

DimensionMapping
Servizio: Amazon Timestream Write

Indice

DestinationColumn

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1.

Required: No

SourceColumn

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1.

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Endpoint

Servizio: Amazon Timestream Write

Rappresenta un endpoint disponibile verso il quale effettuare API chiamate, oltre a quello TTL per quell'endpoint.

Indice

Address

Un indirizzo di endpoint.

Tipo: stringa

Campo obbligatorio: sì

CachePeriodInMinutes

Il TTL per l'endpoint, in pochi minuti.

Tipo: long

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

MagneticStoreRejectedDataLocation
Servizio: Amazon Timestream Write

La posizione in cui scrivere i report degli errori per i record rifiutati, in modo asincrono, durante le scritture dell'archivio magnetico.

Indice

S3Configuration

Configurazione di una posizione S3 in cui scrivere i report degli errori per i record rifiutati, in modo asincrono, durante le scritture dell'archivio magnetico.

Tipo: oggetto S3Configuration

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

MagneticStoreWriteProperties
Servizio: Amazon Timestream Write

Il set di proprietà in una tabella per la configurazione di scritture dell'archivio magnetico.

Indice

EnableMagneticStoreWrites

Un contrassegno per abilitare scritture dell'archivio magnetico.

Tipo: Booleano

Campo obbligatorio: sì

MagneticStoreRejectedDataLocation

La posizione in cui scrivere i report degli errori per i record rifiutati, in modo asincrono, durante le scritture dell'archivio magnetico.

Tipo: oggetto MagneticStoreRejectedDataLocation

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

MeasureValue
Servizio: Amazon Timestream Write

Rappresenta l'attributo di dati della serie temporale. Ad esempio, l'CPUutilizzo di un'EC2istanza o RPM di una turbina eolica sono misure. MeasureValue ha sia un nome che un valore.

MeasureValue è consentito solo per tipo`MULTI`. Utilizzando `MULTI` type, è possibile passare più attributi di dati associati alla stessa serie temporale in un singolo record

Indice

Name

Il nome di MeasureValue.

Per i vincoli sui MeasureValue nomi, consulta Naming Constraints nella Amazon Timestream Developer Guide.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1.

Campo obbligatorio: sì

Type

Contiene il tipo di dati del punto dati della serie temporale. MeasureValue

Tipo: stringa

Valori validi: `DOUBLE | BIGINT | VARCHAR | BOOLEAN | TIMESTAMP | MULTI`

Campo obbligatorio: sì

Value

Il valore per. MeasureValue Per informazioni, consulta Tipi di dati.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

MixedMeasureMapping
Servizio: Amazon Timestream Write

Indice

MeasureValueType

Tipo: stringa

Valori validi: `DOUBLE | BIGINT | VARCHAR | BOOLEAN | TIMESTAMP | MULTI`

Campo obbligatorio: sì

MeasureName

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1.

Campo obbligatorio: no

MultiMeasureAttributeMappings

Tipo: matrice di oggetti MultiMeasureAttributeMapping

Membri dell'array: numero minimo di 1 elemento.

Required: No

SourceColumn

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1.

Required: No

TargetMeasureName

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1.

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

MultiMeasureAttributeMapping
Servizio: Amazon Timestream Write

Indice

SourceColumn

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1.

Campo obbligatorio: sì

MeasureValueType

Tipo: stringa

Valori validi: `DOUBLE | BIGINT | BOOLEAN | VARCHAR | TIMESTAMP`

Campo obbligatorio: no

TargetMultiMeasureAttributeName

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1.

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

MultiMeasureMappings
Servizio: Amazon Timestream Write

Indice

MultiMeasureAttributeMappings

Tipo: matrice di oggetti MultiMeasureAttributeMapping

Membri dell'array: numero minimo di 1 elemento.

Campo obbligatorio: sì

TargetMultiMeasureName

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1.

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

PartitionKey
Servizio: Amazon Timestream Write

Un attributo utilizzato per il partizionamento dei dati in una tabella. Una chiave di dimensione partiziona i dati utilizzando i valori della dimensione specificata dal nome della dimensione come chiave di partizione, mentre una chiave di misura partiziona i dati utilizzando i nomi delle misure (valori della colonna «measure_name»).

Indice

Type

Il tipo di chiave di partizione. Le opzioni sono DIMENSION (chiave di dimensione) e MEASURE (chiave di misura).

Tipo: stringa

Valori validi: `DIMENSION | MEASURE`

Campo obbligatorio: sì

EnforcementInRecord

Il livello di applicazione per la specificazione di una chiave di dimensione nei record importati. Le opzioni sono REQUIRED (la chiave della dimensione deve essere specificata) e OPTIONAL (la chiave della dimensione non deve essere specificata).

Tipo: stringa

Valori validi: `REQUIRED | OPTIONAL`

Campo obbligatorio: no

Name

Il nome dell'attributo utilizzato per una chiave di dimensione.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1.

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Record
Servizio: Amazon Timestream Write

Rappresenta un punto dati di serie temporali che viene scritto in Timestream. Ogni record contiene una serie di dimensioni. Le dimensioni rappresentano gli attributi dei metadati di un punto dati di una serie temporale, ad esempio il nome dell'istanza o la zona di disponibilità di un'EC2istanza. Un record contiene anche il nome della misura, che è il nome della misura raccolta (ad esempio, l'CPUutilizzo di un'EC2istanza). Inoltre, un record contiene il valore della misura e il tipo di valore, che è il tipo di dati del valore della misura. Inoltre, il record contiene il timestamp di quando la misura è stata raccolta e l'unità di timestamp, che rappresenta la granularità del timestamp.

I record hanno un `Version` campo, a 64 bit`long`, che è possibile utilizzare per aggiornare i punti dati. La scrittura di un record duplicato con la stessa dimensione, timestamp e nome di misura ma un valore di misura diverso avrà esito positivo solo se l'`Version`attributo del record nella richiesta di scrittura è superiore a quello del record esistente. Il valore predefinito di Timestream è quello dei record senza il campo. `Version` 1 `Version`

Indice

Dimensions

Contiene l'elenco delle dimensioni per i punti dati delle serie temporali.

Tipo: matrice di oggetti Dimension

Membri dell'array: numero massimo di 128 elementi.

Campo obbligatorio: no

MeasureName

La misura rappresenta l'attributo di dati della serie temporale. Ad esempio, l'CPUutilizzo di un'EC2istanza o RPM di una turbina eolica sono misure.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 256 caratteri.

Campo obbligatorio: no

MeasureValue

Contiene il valore di misura per il punto dati della serie temporale.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: no

MeasureValues

Contiene l'elenco di quattro punti dati MeasureValue delle serie temporali.

È consentito solo per il tipo`MULTI`. Per i valori scalari, usa direttamente l'`MeasureValue`attributo del record.

Tipo: matrice di oggetti MeasureValue

Campo obbligatorio: no

MeasureValueType

Contiene il tipo di dati del valore di misura per il punto dati della serie temporale. Il tipo predefinito è`DOUBLE`. Per ulteriori informazioni, consulta Tipi di dati.

Tipo: stringa

Valori validi: `DOUBLE | BIGINT | VARCHAR | BOOLEAN | TIMESTAMP | MULTI`

Campo obbligatorio: no

Time

Contiene l'ora in cui è stato raccolto il valore di misura per il punto dati. Il valore temporale più l'unità fornisce il tempo trascorso dall'epoca. Ad esempio, se il valore temporale è `12345` e l'unità è`ms`, allora `12345 ms` sono trascorsi dall'epoca.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 256 caratteri.

Campo obbligatorio: no

TimeUnit

La granularità dell'unità timestamp. Indica se il valore temporale è in secondi, millisecondi, nanosecondi o altri valori supportati. Il valore predefinito è `MILLISECONDS`.

Tipo: stringa

Valori validi: `MILLISECONDS | SECONDS | MICROSECONDS | NANOSECONDS`

Campo obbligatorio: no

Version

Attributo a 64 bit utilizzato per gli aggiornamenti dei record. Le richieste di scrittura per dati duplicati con un numero di versione superiore aggiorneranno il valore e la versione della misura esistenti. Nei casi in cui il valore di misura è lo stesso, `Version` verrà comunque aggiornato. Il valore predefinito è 1.

> **Note**
>
> `Version`deve essere uguale 1 o superiore, altrimenti riceverai un `ValidationException` errore.

Tipo: long

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

RecordsIngested
Servizio: Amazon Timestream Write

Informazioni sui record acquisiti con questa richiesta.

Indice

MagneticStore

Numero di record ingeriti nel magazzino magnetico.

Tipo: integer

Campo obbligatorio: no

MemoryStore

Numero di record inseriti nell'archivio di memoria.

Tipo: integer

Campo obbligatorio: no

Total

Numero totale di record ingeriti con successo.

Tipo: integer

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo codice API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

RejectedRecord
Servizio: Amazon Timestream Write

Rappresenta i record che non sono stati inseriti correttamente in Timestream a causa di problemi di convalida dei dati che devono essere risolti prima di reinserire i dati delle serie temporali nel sistema.

Indice

ExistingVersion

La versione esistente del record. Questo valore viene compilato negli scenari in cui esiste un record identico con una versione superiore a quella nella richiesta di scrittura.

Tipo: long

Campo obbligatorio: no

Reason

Il motivo per cui un record non è stato inserito correttamente in Timestream. Le possibili cause di errore includono:

- Record con dati duplicati in cui sono presenti più record con le stesse dimensioni, timestamp e nomi di misure ma:
  - I valori di misura sono diversi
  - La versione non è presente nella richiesta oppure il valore della versione nel nuovo record è uguale o inferiore al valore esistente

  Se Timestream rifiuta i dati per questo caso, il `ExistingVersion` campo nella `RejectedRecords` risposta indicherà la versione del record corrente. Per forzare un aggiornamento, puoi inviare nuovamente la richiesta con una versione del record impostata su un valore maggiore di. `ExistingVersion`
- Record con timestamp che non rientrano nella durata di conservazione dell'archivio di memoria.

> **Note**
>
> Quando la finestra di conservazione viene aggiornata, riceverai un'`RejectedRecords`eccezione se tenti immediatamente di inserire dati all'interno della nuova finestra. Per evitare un'`RejectedRecords`eccezione, attendi la durata della nuova finestra per inserire nuovi dati. Per ulteriori informazioni, consulta Best

Practices for Configuring Timestream e la spiegazione di come funziona lo storage in Timestream.

- Record con dimensioni o misure che superano i limiti definiti dal Timestream.

Per ulteriori informazioni, vedere Gestione degli accessi nella guida per gli sviluppatori Timestream.

Tipo: string

Campo obbligatorio: no

RecordIndex

L'indice del record nella richiesta di input per. WriteRecords Gli indici iniziano con 0.

Tipo: integer

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

ReportConfiguration
Servizio: Amazon Timestream Write

Configurazione del report per un'attività di caricamento in batch. Contiene dettagli sulla posizione in cui vengono archiviate le segnalazioni di errori.

Indice

ReportS3Configuration

Configurazione di una posizione S3 per scrivere report ed eventi di errore per un caricamento in batch.

Tipo: oggetto ReportS3Configuration

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

ReportS3Configuration
Servizio: Amazon Timestream Write

Indice

BucketName

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 63 caratteri.

Modello: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

Campo obbligatorio: sì

EncryptionOption

Tipo: stringa

Valori validi: `SSE_S3 | SSE_KMS`

Campo obbligatorio: no

KmsKeyId

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Required: No

ObjectKeyPrefix

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. Lunghezza massima di 928.

Modello: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

RetentionProperties
Servizio: Amazon Timestream Write

Le proprietà di conservazione contengono la durata per la quale i dati di serie temporali devono essere memorizzati nell'archivio magnetico e nell'archivio della memoria.

Indice

MagneticStoreRetentionPeriodInDays

La durata per la quale i dati devono essere memorizzati nell'archivio magnetico.

Tipo: long

Intervallo valido: valore minimo di 1. Valore massimo di 73000.

Campo obbligatorio: sì

MemoryStoreRetentionPeriodInHours

La durata per la quale i dati devono essere memorizzati nell'archivio della memoria.

Tipo: long

Intervallo valido: valore minimo di 1. Valore massimo di 8766.

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

S3Configuration

Servizio: Amazon Timestream Write

La configurazione che specifica una posizione S3.

Indice

BucketName

Il nome bucket del bucket S3 del cliente.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 63 caratteri.

Modello: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

Campo obbligatorio: no

EncryptionOption

L'opzione di crittografia per la posizione S3 del cliente. Le opzioni sono la crittografia lato server S3 con una chiave gestita o una chiave gestita S3. AWS

Tipo: stringa

Valori validi: `SSE_S3 | SSE_KMS`

Campo obbligatorio: no

KmsKeyId

L'ID della AWS KMS chiave per la posizione S3 del cliente durante la crittografia con una chiave gestita. AWS

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: no

ObjectKeyPrefix

L'anteprima della chiave oggetto per la posizione S3 del cliente.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. Lunghezza massima di 928.

Modello: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Schema

Servizio: Amazon Timestream Write

Uno schema specifica il modello di dati previsto per la tabella.

Indice

CompositePartitionKey

Un elenco non vuoto di chiavi di partizione che definiscono gli attributi utilizzati per partizionare i dati della tabella. L'ordine dell'elenco determina la gerarchia delle partizioni. Il nome e il tipo di ciascuna chiave di partizione e l'ordine delle chiavi di partizione non possono essere modificati dopo la creazione della tabella. Tuttavia, è possibile modificare il livello di applicazione di ciascuna chiave di partizione.

Tipo: matrice di oggetti PartitionKey

Membri dell'array: numero minimo di 1 elemento.

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Table
Servizio: Amazon Timestream Write

Rappresenta una tabella di database in Timestream. Le tabelle contengono una o più serie temporali correlate. È possibile modificare la durata di conservazione dell'archivio di memoria e dell'archivio magnetico per una tabella.

Indice

Arn

Il nome di risorsa Amazon che identifica in modo univoco questa tabella.

Tipo: string

Campo obbligatorio: no

CreationTime

L'ora in cui è stata creata la tabella Timestream.

Tipo: Timestamp

Campo obbligatorio: no

DatabaseName

Nome del database Timestream che contiene questa tabella.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: no

LastUpdatedTime

L'ora dell'ultimo aggiornamento della tabella Timestream.

Tipo: Timestamp

Campo obbligatorio: no

MagneticStoreWriteProperties

Contiene le proprietà da impostare nella tabella quando si abilitano le scritture dello store magnetico.

Tipo: oggetto MagneticStoreWriteProperties

Campo obbligatorio: no

RetentionProperties

La durata di conservazione per lo store di memoria e lo store magnetico.

Tipo: oggetto RetentionProperties

Campo obbligatorio: no

Schema

Lo schema della tabella.

Tipo: oggetto Schema

Campo obbligatorio: no

TableName

Nome della tabella Timestream.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 256 caratteri.

Campo obbligatorio: no

TableStatus

Lo stato attuale della tabella:

- `DELETING`- La tabella viene eliminata.
- `ACTIVE`- Il tavolo è pronto per l'uso.

Tipo: stringa

Valori validi: `ACTIVE | DELETING | RESTORING`

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Tag
Servizio: Amazon Timestream Write

Un tag è un'etichetta che si assegna alle and/or table. Each tag consists of a key and an optional value, both of which you define. With tags, you can categorize databases and/or tabelle di un database Timestream, ad esempio per scopo, proprietario o ambiente.

Indice

Key

La chiave del tag. Le chiavi dei tag prevedono una distinzione tra lettere maiuscole e minuscole.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 128 caratteri.

Campo obbligatorio: sì

Value

Il valore del tag. I valori dei tag fanno distinzione tra maiuscole e minuscole e possono essere nulli.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 0. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo codice API in una delle lingue AWS SDKs specifiche, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Interrogazione su Amazon Timestream

I seguenti tipi di dati sono supportati da Amazon Timestream Query:

- ColumnInfo
- Datum
- DimensionMapping
- Endpoint
- ErrorReportConfiguration
- ErrorReportLocation
- ExecutionStats
- MixedMeasureMapping
- MultiMeasureAttributeMapping
- MultiMeasureMappings
- NotificationConfiguration
- ParameterMapping
- QueryInsights
- QueryInsightsResponse
- QuerySpatialCoverage
- QuerySpatialCoverageMax
- QueryStatus
- QueryTemporalRange
- QueryTemporalRangeMax
- Row
- S3Configuration
- S3ReportLocation
- ScheduleConfiguration
- ScheduledQuery
- ScheduledQueryDescription
- ScheduledQueryInsights
- ScheduledQueryInsightsResponse
- ScheduledQueryRunSummary
- SelectColumn
- SnsConfiguration

- Tag
- TargetConfiguration
- TargetDestination
- TimeSeriesDataPoint
- TimestreamConfiguration
- TimestreamDestination
- Type

ColumnInfo
Servizio: Amazon Timestream Query

Contiene i metadati per i risultati delle query, ad esempio i nomi delle colonne, i tipi di dati e altri attributi.

Indice

Type

Il tipo di dati della colonna del set di risultati. Il tipo di dati può essere scalare o complesso. I tipi di dati scalari sono numeri interi, stringhe, doppi, booleani e altri. I tipi di dati complessi sono tipi come matrici, righe e altri.

Tipo: oggetto Type

Campo obbligatorio: sì

Name

Il nome della colonna del set di risultati. Il nome del set di risultati è disponibile per le colonne di tutti i tipi di dati ad eccezione degli array.

Tipo: string

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo codice API in una delle lingue specifiche AWS SDKs, consultate quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Datum

Servizio: Amazon Timestream Query

Datum rappresenta un singolo punto dati nel risultato di una query.

Indice

ArrayValue

Indica se il punto dati è una matrice.

Tipo: matrice di oggetti Datum

Campo obbligatorio: no

NullValue

Indica se il punto dati è nullo.

Tipo: Booleano

Campo obbligatorio: no

RowValue

Indica se il punto dati è una riga.

Tipo: oggetto Row

Campo obbligatorio: no

ScalarValue

Indica se il punto dati è un valore scalare come intero, stringa, double o booleano.

Tipo: string

Campo obbligatorio: no

TimeSeriesValue

Indica se il punto dati è un tipo di dati di una serie temporale.

Tipo: matrice di oggetti TimeSeriesDataPoint

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

DimensionMapping
Servizio: Amazon Timestream Query

Questo tipo viene utilizzato per mappare colonne dal risultato della query a una dimensione nella tabella di destinazione.

Indice

DimensionValueType

Tipo per la dimensione.

Tipo: stringa

Valori validi: `VARCHAR`

Campo obbligatorio: sì

Name

Nome della colonna dal risultato della query.

Tipo: stringa

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Endpoint

Servizio: Amazon Timestream Query

Rappresenta un endpoint disponibile verso il quale effettuare API chiamate, oltre a quello TTL per quell'endpoint.

Indice

Address

Un indirizzo di endpoint.

Tipo: stringa

Campo obbligatorio: sì

CachePeriodInMinutes

Il TTL per l'endpoint, in pochi minuti.

Tipo: long

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

ErrorReportConfiguration
Servizio: Amazon Timestream Query

Configurazione richiesta per la segnalazione degli errori.

Indice

S3Configuration

La configurazione S3 per la segnalazione degli errori.

Tipo: oggetto S3Configuration

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

ErrorReportLocation
Servizio: Amazon Timestream Query

Contiene la posizione della segnalazione degli errori per una singola chiamata di interrogazione pianificata.

Indice

S3ReportLocation

La posizione S3 in cui vengono scritte le segnalazioni di errore.

Tipo: oggetto S3ReportLocation

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

ExecutionStats

Servizio: Amazon Timestream Query

Statistiche per una singola esecuzione di query pianificata.

Indice

BytesMetered

Byte misurati per una singola esecuzione di query pianificata.

Tipo: long

Campo obbligatorio: no

CumulativeBytesScanned

Byte analizzati per una singola esecuzione di una query pianificata.

Tipo: long

Campo obbligatorio: no

DataWrites

Le scritture dei dati vengono misurate per i record inseriti in una singola esecuzione di query pianificata.

Tipo: long

Campo obbligatorio: no

ExecutionTimeInMillis

Tempo totale, misurato in millisecondi, necessario per il completamento dell'esecuzione della query pianificata.

Tipo: long

Campo obbligatorio: no

QueryResultRows

Numero di righe presenti nell'output dell'esecuzione di una query prima dell'inserimento nell'origine dati di destinazione.

Tipo: long

Campo obbligatorio: no

RecordsIngested

Il numero di record acquisiti per una singola esecuzione di query pianificata.

Tipo: long

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo codice API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

MixedMeasureMapping
Servizio: Amazon Timestream Query

MixedMeasureMappings sono mappature che possono essere utilizzate per inserire dati in una combinazione di misure ristrette e multiple nella tabella derivata.

Indice

MeasureValueType

Tipo di valore da cui leggere. sourceColumn Se la mappatura è perMULTI, usa MeasureValueType. MULTI.

Tipo: stringa

Valori validi: `BIGINT | BOOLEAN | DOUBLE | VARCHAR | MULTI`

Campo obbligatorio: sì

MeasureName

Fa riferimento al valore measure_name in una riga dei risultati. Questo campo è obbligatorio se MeasureNameColumn fornito.

Tipo: string

Campo obbligatorio: no

MultiMeasureAttributeMappings

Richiesto quando measureValueType èMULTI. Mappature degli attributi per le misure di MULTI valore.

Tipo: matrice di oggetti MultiMeasureAttributeMapping

Membri dell'array: numero minimo di 1 elemento.

Campo obbligatorio: no

SourceColumn

Questo campo fa riferimento alla colonna di origine da cui deve essere letto measure-value per la materializzazione del risultato.

Tipo: string

Campo obbligatorio: no

TargetMeasureName

Nome della misura di destinazione da utilizzare. Se non viene fornito, il nome della misura di destinazione per impostazione predefinita sarebbe il nome della misura, se fornito o in altro modo. sourceColumn

Tipo: string

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

MultiMeasureAttributeMapping
Servizio: Amazon Timestream Query

Mappatura degli attributi per le misure di MULTI valore.

Indice

MeasureValueType

Tipo dell'attributo da leggere dalla colonna di origine.

Tipo: stringa

Valori validi: `BIGINT | BOOLEAN | DOUBLE | VARCHAR | TIMESTAMP`

Campo obbligatorio: sì

SourceColumn

Colonna di origine da cui deve essere letto il valore dell'attributo.

Tipo: stringa

Campo obbligatorio: sì

TargetMultiMeasureAttributeName

Nome personalizzato da utilizzare per il nome dell'attributo nella tabella derivata. Se non viene fornito, verrà utilizzato il nome della colonna di origine.

Tipo: string

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

MultiMeasureMappings
Servizio: Amazon Timestream Query

Deve essere fornito solo uno dei MixedMeasureMappings o MultiMeasureMappings deve essere fornito. MultiMeasureMappings può essere utilizzato per inserire dati come misure multiple nella tabella derivata.

Indice

MultiMeasureAttributeMappings

Obbligatorio. Mappature di attributi da utilizzare per mappare i risultati delle query per l'acquisizione di dati per attributi con più misure.

Tipo: matrice di oggetti MultiMeasureAttributeMapping

Membri dell'array: numero minimo di 1 elemento.

Campo obbligatorio: sì

TargetMultiMeasureName

Il nome del nome con più misure di destinazione nella tabella derivata. Questo input è obbligatorio quando non measureNameColumn viene fornito. Se MeasureNameColumn viene fornito, il valore di quella colonna verrà utilizzato come nome multimisura.

Tipo: string

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo codice API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

NotificationConfiguration
Servizio: Amazon Timestream Query

Configurazione della notifica per una query pianificata. Viene inviata una notifica da Timestream quando una query pianificata viene creata, il suo stato viene aggiornato o quando viene eliminata.

Indice

SnsConfiguration

Dettagli sulla SNS configurazione.

Tipo: oggetto SnsConfiguration

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

ParameterMapping
Servizio: Amazon Timestream Query

Mappatura per parametri denominati.

Indice

Name

Nome del parametro.

Tipo: stringa

Campo obbligatorio: sì

Type

Contiene il tipo di dati di una colonna in un set di risultati di query. Il tipo di dati può essere scalare o complesso. I tipi di dati scalari supportati sono numeri interi, booleani, string, double, timestamp, date, time e intervalli. I tipi di dati complessi supportati sono matrici, righe e serie temporali.

Tipo: oggetto Type

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue AWS SDKs specifiche, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

QueryInsights
Servizio: Amazon Timestream Query

`QueryInsights`è una funzionalità di ottimizzazione delle prestazioni che consente di ottimizzare le query, ridurre i costi e migliorare le prestazioni. Con`QueryInsights`, è possibile valutare l'efficienza di ottimizzazione delle query e identificare le aree di miglioramento per migliorare le prestazioni delle query. Con`QueryInsights`, puoi anche analizzare l'efficacia delle tue query in termini di riduzione temporale e spaziale e identificare opportunità per migliorare le prestazioni. In particolare, è possibile valutare in che modo le query utilizzano strategie di indicizzazione basate sul tempo e su chiavi di partizione per ottimizzare il recupero dei dati. Per ottimizzare le prestazioni delle query, è essenziale ottimizzare i parametri temporali e spaziali che regolano l'esecuzione delle query.

Le metriche chiave fornite da sono e. `QueryInsights` `QuerySpatialCoverage` `QueryTemporalRange` `QuerySpatialCoverage`indica la parte dell'asse spaziale scansionata dalla query, mentre i valori più bassi sono più efficienti. `QueryTemporalRange`mostra l'intervallo di tempo scansionato, con intervalli più stretti che offrono prestazioni migliori.

Vantaggi di QueryInsights

Di seguito sono riportati i principali vantaggi dell'utilizzo`QueryInsights`:

- Identificazione delle query inefficienti: `QueryInsights` fornisce informazioni sull'eliminazione basata sul tempo e sugli attributi delle tabelle a cui accede la query. Queste informazioni consentono di identificare le tabelle a cui si accede in modo non ottimale.
- Ottimizzazione del modello di dati e del partizionamento: è possibile utilizzare `QueryInsights` le informazioni per accedere e perfezionare il modello di dati e la strategia di partizionamento.
- Ottimizzazione delle query: evidenzia le opportunità di utilizzare gli indici in modo più efficace. `QueryInsights`

> **Note**
>
> Il numero massimo di `Query` API richieste che puoi effettuare con `QueryInsights` enabled è 1 query al secondo (). QPS Se superi questa frequenza di query, potrebbe verificarsi una limitazione.

Indice

Mode

Fornisce le seguenti modalità da abilitare: `QueryInsights`

- `ENABLED_WITH_RATE_CONTROL`— Abilita `QueryInsights` l'elaborazione delle interrogazioni. Questa modalità include anche un meccanismo di controllo della velocità, che limita la `QueryInsights` funzionalità a 1 query al secondo (QPS).
- `DISABLED`— Disabilita. `QueryInsights`

Tipo: stringa

Valori validi: `ENABLED_WITH_RATE_CONTROL | DISABLED`

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, vedere quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

QueryInsightsResponse
Servizio: Amazon Timestream Query

Fornisce varie informazioni e metriche relative alla query che hai eseguito.

Indice

OutputBytes

Indica la dimensione del set di risultati della query in byte. È possibile utilizzare questi dati per verificare se il set di risultati è stato modificato nell'ambito dell'esercizio di ottimizzazione delle query.

Tipo: long

Campo obbligatorio: no

OutputRows

Indica il numero totale di righe restituite come parte del set di risultati della query. È possibile utilizzare questi dati per verificare se il numero di righe nel set di risultati è cambiato nell'ambito dell'esercizio di ottimizzazione delle query.

Tipo: long

Campo obbligatorio: no

QuerySpatialCoverage

Fornisce informazioni sulla copertura spaziale dell'interrogazione, inclusa la tabella con una potatura spaziale non ottimale (massima). Queste informazioni possono aiutarvi a identificare le aree da migliorare nella vostra strategia di partizionamento per migliorare la potatura spaziale.

Tipo: oggetto QuerySpatialCoverage

Campo obbligatorio: no

QueryTableCount

Indica il numero di tabelle nell'interrogazione.

Tipo: long

Campo obbligatorio: no

QueryTemporalRange

Fornisce informazioni sull'intervallo temporale della query, inclusa la tabella con l'intervallo di tempo più ampio (massimo). Di seguito sono riportate alcune delle potenziali opzioni per ottimizzare la potatura basata sul tempo:

- Aggiungi i predicati temporali mancanti.
- Rimuovi le funzioni relative ai predicati temporali.
- Aggiungi predicati temporali a tutte le sottoquery.

Tipo: oggetto QueryTemporalRange

Campo obbligatorio: no

UnloadPartitionCount

Indica le partizioni create dall'operazione. `Unload`

Tipo: long

Campo obbligatorio: no

UnloadWrittenBytes

Indica la dimensione, in byte, scritta dall'`Unload`operazione.

Tipo: long

Campo obbligatorio: no

UnloadWrittenRows

Indica le righe scritte dall'`Unload`interrogazione.

Tipo: long

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++

- AWS SDKper Java V2
- AWS SDKper Ruby V3

QuerySpatialCoverage
Servizio: Amazon Timestream Query

Fornisce informazioni sulla copertura spaziale della query, inclusa la tabella con una potatura spaziale non ottimale (massima). Queste informazioni possono aiutarvi a identificare le aree da migliorare nella vostra strategia di partizionamento per migliorare la potatura spaziale

Ad esempio, puoi fare quanto segue con le informazioni: `QuerySpatialCoverage`

- Aggiungi measure_name o usa i predicati della chiave di partizione () definiti dal cliente. CDPK
- Se hai già eseguito l'azione precedente, rimuovi le funzioni o le clausole che li circondano, ad esempio. `LIKE`

Indice

Max

Fornisce informazioni sulla copertura spaziale della query eseguita e della tabella con la riduzione spaziale più inefficiente.

- `Value`— Il rapporto massimo di copertura spaziale.
- `TableArn`— L'Amazon Resource Name (ARN) della tabella con potatura spaziale non ottimale.
- `PartitionKey`— La chiave di partizione utilizzata per il partizionamento, che può essere predefinita o una. `measure_name` CDPK

Tipo: oggetto QuerySpatialCoverageMax

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue AWS SDKs specifiche, vedere quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

QuerySpatialCoverageMax
Servizio: Amazon Timestream Query

Fornisce informazioni dettagliate sulla tabella con l'intervallo spaziale meno ottimale analizzato dalla query.

Indice

PartitionKey

La chiave di partizione utilizzata per il partizionamento, che può essere una chiave di partizione predefinita `measure_name` o definita dal cliente.

Tipo: matrice di stringhe

Campo obbligatorio: no

TableArn

L'Amazon Resource Name (ARN) della tabella con la potatura spaziale meno ottimale.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: no

Value

Il rapporto massimo di copertura spaziale.

Tipo: double

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo codice API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

QueryStatus
Servizio: Amazon Timestream Query

Informazioni sullo stato della query, inclusi lo stato di avanzamento e i byte scansionati.

Indice

CumulativeBytesMetered

La quantità di dati analizzati dalla query, espressa in byte, per la quale ti verrà addebitato il costo. Si tratta di una somma cumulativa e rappresenta la quantità totale di dati che ti verrà addebitata dall'avvio della query. L'addebito viene applicato una sola volta e viene applicato al termine dell'esecuzione della query o quando la query viene annullata.

Tipo: long

Campo obbligatorio: no

CumulativeBytesScanned

La quantità di dati analizzati dalla query, espressa in byte. Si tratta di una somma cumulativa e rappresenta la quantità totale di byte analizzati dall'avvio della query.

Tipo: long

Campo obbligatorio: no

ProgressPercentage

L'avanzamento della query, espresso in percentuale.

Tipo: double

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

QueryTemporalRange
Servizio: Amazon Timestream Query

Fornisce informazioni sull'intervallo temporale della query, inclusa la tabella con l'intervallo di tempo più ampio (massimo).

Indice

Max

Incapsula le seguenti proprietà che forniscono informazioni sulla tabella con prestazioni meno ottimali sull'asse temporale:

- `Value`— La durata massima in nanosecondi tra l'inizio e la fine della query.
- `TableArn`— L'Amazon Resource Name (ARN) della tabella su cui viene eseguita la query con l'intervallo di tempo più ampio.

Tipo: oggetto QueryTemporalRangeMax

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

QueryTemporalRangeMax
Servizio: Amazon Timestream Query

Fornisce informazioni dettagliate sulla tabella con la riduzione temporale meno ottimale analizzata dalla query.

Indice

TableArn

L'Amazon Resource Name (ARN) della tabella su cui viene eseguita la query con l'intervallo di tempo più ampio.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: no

Value

La durata massima in nanosecondi tra l'inizio e la fine della query.

Tipo: long

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Row
Servizio: Amazon Timestream Query

Rappresenta una singola riga nei risultati della query.

Indice

Data

Elenco di punti dati in una singola riga del set di risultati.

Tipo: matrice di oggetti Datum

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

S3Configuration

Servizio: Amazon Timestream Query

Dettagli sulla posizione S3 per le segnalazioni di errore risultanti dall'esecuzione di una query.

Indice

BucketName

Nome del bucket S3 in cui verranno creati le segnalazioni di errore.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 63 caratteri.

Modello: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

Campo obbligatorio: sì

EncryptionOption

Opzioni di crittografia dei dati inattivi per le segnalazioni di errore. Se non viene specificata alcuna opzione di crittografia, Timestream sceglierà SSE _S3 come impostazione predefinita.

Tipo: stringa

Valori validi: `SSE_S3 | SSE_KMS`

Campo obbligatorio: no

ObjectKeyPrefix

Prefisso per la chiave di segnalazione degli errori. Timestream per impostazione predefinita aggiunge il seguente prefisso al percorso della segnalazione degli errori.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. Lunghezza massima di 896.

Modello: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

S3ReportLocation
Servizio: Amazon Timestream Query

Posizione del report S3 per l'esecuzione della query pianificata.

Indice

BucketName

Nome bucket S3.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 3. La lunghezza massima è 63 caratteri.

Modello: `[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]`

Campo obbligatorio: no

ObjectKey

Chiave S3.

Tipo: string

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

ScheduleConfiguration
Servizio: Amazon Timestream Query

Configurazione della pianificazione della query.

Indice

ScheduleExpression

Un'espressione che indica quando attivare l'esecuzione pianificata della query. Può essere un'espressione cron o un'espressione rate.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

ScheduledQuery
Servizio: Amazon Timestream Query

Query pianificata

Indice

Arn

Il nome della risorsa Amazon.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: sì

Name

Il nome della query pianificata.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 64 caratteri.

Modello: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Campo obbligatorio: sì

State

Stato dell'interrogazione pianificata.

Tipo: stringa

Valori validi: `ENABLED | DISABLED`

Campo obbligatorio: sì

CreationTime

L'ora di creazione dell'interrogazione pianificata.

Tipo: Timestamp

Campo obbligatorio: no

ErrorReportConfiguration

Configurazione per la segnalazione degli errori delle query pianificate.

Tipo: oggetto ErrorReportConfiguration

Campo obbligatorio: no

LastRunStatus

Stato dell'ultima esecuzione pianificata della query.

Tipo: stringa

Valori validi: `AUTO_TRIGGER_SUCCESS | AUTO_TRIGGER_FAILURE | MANUAL_TRIGGER_SUCCESS | MANUAL_TRIGGER_FAILURE`

Campo obbligatorio: no

NextInvocationTime

La prossima volta che verrà eseguita la query pianificata.

Tipo: Timestamp

Campo obbligatorio: no

PreviousInvocationTime

L'ultima volta che è stata eseguita la query pianificata.

Tipo: Timestamp

Campo obbligatorio: no

TargetDestination

Fonte di dati di destinazione in cui verrà scritto il risultato finale della query pianificata.

Tipo: oggetto TargetDestination

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

ScheduledQueryDescription
Servizio: Amazon Timestream Query

Struttura che descrive l'interrogazione pianificata.

Indice

Arn

Interrogazione pianificataARN.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: sì

Name

Nome dell'interrogazione pianificata.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 64 caratteri.

Modello: `[a-zA-Z0-9|!\-_*'\(\)]([a-zA-Z0-9]|[!\-_*'\(\)\/.])+`

Campo obbligatorio: sì

NotificationConfiguration

Configurazione delle notifiche.

Tipo: oggetto NotificationConfiguration

Campo obbligatorio: sì

QueryString

L'interrogazione da eseguire.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. Lunghezza massima di 262144.

Campo obbligatorio: sì

ScheduleConfiguration

Configurazione della pianificazione.

Tipo: oggetto ScheduleConfiguration

Campo obbligatorio: sì

State

Stato dell'interrogazione pianificata.

Tipo: stringa

Valori validi: `ENABLED | DISABLED`

Campo obbligatorio: sì

CreationTime

Ora di creazione dell'interrogazione pianificata.

Tipo: Timestamp

Campo obbligatorio: no

ErrorReportConfiguration

Configurazione per la segnalazione degli errori per l'interrogazione pianificata.

Tipo: oggetto ErrorReportConfiguration

Campo obbligatorio: no

KmsKeyId

Una KMS chiave fornita dal cliente utilizzata per crittografare la risorsa di interrogazione pianificata.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: no

LastRunSummary

Riepilogo del runtime per l'ultima esecuzione pianificata della query.

Tipo: oggetto ScheduledQueryRunSummary

Campo obbligatorio: no

NextInvocationTime

La prossima volta che viene pianificata l'esecuzione della query pianificata.

Tipo: Timestamp

Campo obbligatorio: no

PreviousInvocationTime

L'ultima volta che è stata eseguita la query.

Tipo: Timestamp

Campo obbligatorio: no

RecentlyFailedRuns

Riepilogo del runtime per le ultime cinque interrogazioni pianificate non riuscite.

Tipo: matrice di oggetti ScheduledQueryRunSummary

Campo obbligatorio: no

ScheduledQueryExecutionRoleArn

IAMruolo utilizzato da Timestream per eseguire la query di pianificazione.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: no

TargetConfiguration

Configurazione dell'archivio di destinazione delle query pianificate.

Tipo: oggetto TargetConfiguration

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

ScheduledQueryInsights
Servizio: Amazon Timestream Query

Incapsula le impostazioni per l'attivazione `QueryInsights` su un. `ExecuteScheduledQueryRequest`

Indice

Mode

Fornisce le seguenti modalità da abilitare: `ScheduledQueryInsights`

- `ENABLED_WITH_RATE_CONTROL`— Abilita `ScheduledQueryInsights` l'elaborazione delle interrogazioni. Questa modalità include anche un meccanismo di controllo della velocità, che limita la `QueryInsights` funzionalità a 1 query al secondo (QPS).
- `DISABLED`— Disabilita. `ScheduledQueryInsights`

Tipo: stringa

Valori validi: `ENABLED_WITH_RATE_CONTROL | DISABLED`

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, vedere quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

ScheduledQueryInsightsResponse
Servizio: Amazon Timestream Query

Fornisce varie informazioni e metriche relative a `ExecuteScheduledQueryRequest` ciò che è stato eseguito.

Indice

OutputBytes

Indica la dimensione del set di risultati della query in byte. È possibile utilizzare questi dati per verificare se il set di risultati è stato modificato nell'ambito dell'esercizio di ottimizzazione delle query.

Tipo: long

Campo obbligatorio: no

OutputRows

Indica il numero totale di righe restituite come parte del set di risultati della query. È possibile utilizzare questi dati per verificare se il numero di righe nel set di risultati è cambiato nell'ambito dell'esercizio di ottimizzazione delle query.

Tipo: long

Campo obbligatorio: no

QuerySpatialCoverage

Fornisce informazioni sulla copertura spaziale dell'interrogazione, inclusa la tabella con una potatura spaziale non ottimale (massima). Queste informazioni possono aiutarvi a identificare le aree da migliorare nella vostra strategia di partizionamento per migliorare la potatura spaziale.

Tipo: oggetto QuerySpatialCoverage

Campo obbligatorio: no

QueryTableCount

Indica il numero di tabelle nell'interrogazione.

Tipo: long

Campo obbligatorio: no

QueryTemporalRange

Fornisce informazioni sull'intervallo temporale della query, inclusa la tabella con l'intervallo di tempo più ampio (massimo). Di seguito sono riportate alcune delle potenziali opzioni per ottimizzare la potatura basata sul tempo:

- Aggiungi i predicati temporali mancanti.
- Rimuovi le funzioni relative ai predicati temporali.
- Aggiungi predicati temporali a tutte le sottoquery.

Tipo: oggetto QueryTemporalRange

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo codice API in una delle lingue AWS SDKs specifiche, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

ScheduledQueryRunSummary
Servizio: Amazon Timestream Query

Esegui il riepilogo per la query pianificata

Indice

ErrorReportLocation

Posizione S3 per la segnalazione degli errori.

Tipo: oggetto ErrorReportLocation

Campo obbligatorio: no

ExecutionStats

Statistiche di runtime per un'esecuzione pianificata.

Tipo: oggetto ExecutionStats

Campo obbligatorio: no

FailureReason

Messaggio di errore per l'interrogazione pianificata in caso di errore. Potrebbe essere necessario consultare il rapporto sugli errori per ottenere motivi di errore più dettagliati.

Tipo: string

Campo obbligatorio: no

InvocationTime

InvocationTime per questa corsa. Questa è l'ora in cui è pianificata l'esecuzione della query. Il parametro `@scheduled_runtime` può essere utilizzato nella query per ottenere il valore.

Tipo: Timestamp

Campo obbligatorio: no

QueryInsightsResponse

Fornisce varie informazioni e metriche relative al riepilogo dell'esecuzione della query pianificata.

Tipo: oggetto ScheduledQueryInsightsResponse

Campo obbligatorio: no

RunStatus

Lo stato dell'esecuzione di una query pianificata.

Tipo: stringa

Valori validi: `AUTO_TRIGGER_SUCCESS | AUTO_TRIGGER_FAILURE | MANUAL_TRIGGER_SUCCESS | MANUAL_TRIGGER_FAILURE`

Campo obbligatorio: no

TriggerTime

L'ora effettiva in cui è stata eseguita la query.

Tipo: Timestamp

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

SelectColumn
Servizio: Amazon Timestream Query

Dettagli della colonna restituita dalla query.

Indice

Aliased

Ha valore True, se la query ha utilizzato un alias per il nome della colonna. In caso contrario, falso.

Tipo: Booleano

Campo obbligatorio: no

DatabaseName

Database che contiene questa colonna.

Tipo: string

Campo obbligatorio: no

Name

Nome della colonna.

Tipo: string

Campo obbligatorio: no

TableName

Tabella all'interno del database che contiene questa colonna.

Tipo: string

Campo obbligatorio: no

Type

Contiene il tipo di dati di una colonna in un set di risultati di query. Il tipo di dati può essere scalare o complesso. I tipi di dati scalari supportati sono numeri interi, booleani, string, double, timestamp, date, time e intervalli. I tipi di dati complessi supportati sono matrici, righe e serie temporali.

Tipo: oggetto Type

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue AWS SDKs specifiche, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

SnsConfiguration
Servizio: Amazon Timestream Query

I relativi SNS dettagli sono necessari per inviare la notifica.

Indice

TopicArn

SNSargomento a ARN cui verranno inviate le notifiche sullo stato delle interrogazioni pianificate.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 1. La lunghezza massima è 2048 caratteri.

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa API opzione in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Tag

Servizio: Amazon Timestream Query

Un tag è un'etichetta che si assegna alle and/or table. Each tag consists of a key and an optional value, both of which you define. Tags enable you to categorize databases and/or tabelle di un database Timestream, ad esempio per scopo, proprietario o ambiente.

Indice

Key

La chiave del tag. Le chiavi dei tag prevedono una distinzione tra lettere maiuscole e minuscole.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima pari a 1. La lunghezza massima è 128 caratteri.

Campo obbligatorio: sì

Value

Il valore del tag. I valori dei tag fanno distinzione tra maiuscole e minuscole e possono essere nulli.

Tipo: stringa

Limitazioni di lunghezza: lunghezza minima di 0. La lunghezza massima è 256 caratteri.

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo codice API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

TargetConfiguration
Servizio: Amazon Timestream Query

Configurazione utilizzata per scrivere l'output di una query.

Indice

TimestreamConfiguration

Configurazione necessaria per scrivere dati nel database e nella tabella Timestream.

Tipo: oggetto TimestreamConfiguration

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

TargetDestination
Servizio: Amazon Timestream Query

Dettagli sulla destinazione per scrivere dati per un'origine dati di destinazione. La fonte di dati attualmente supportata è Timestream.

Indice

TimestreamDestination

Esegui una query sui dettagli della destinazione dei risultati per l'origine dati Timestream.

Tipo: oggetto TimestreamDestination

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

TimeSeriesDataPoint
Servizio: Amazon Timestream Query

Il tipo di dati timeseries rappresenta i valori di una misura nel tempo. Una serie temporale è una matrice di righe di timestamp e valori di misura, con righe ordinate in ordine crescente di tempo. A TimeSeriesDataPoint è un singolo punto dati nella serie temporale. Rappresenta una tupla di (tempo, valore di misura) in una serie temporale.

Indice

Time

Il timestamp in cui è stato raccolto il valore della misura.

Tipo: stringa

Campo obbligatorio: sì

Value

Il valore di misura per il punto dati.

Tipo: oggetto Datum

Campo obbligatorio: sì

Vedi anche

Per ulteriori informazioni sull'utilizzo di questo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

TimestreamConfiguration
Servizio: Amazon Timestream Query

Configurazione per scrivere dati nel database e nella tabella Timestream. Questa configurazione consente all'utente di mappare le colonne di selezione dei risultati della query nelle colonne della tabella di destinazione.

Indice

DatabaseName

Nome del database Timestream in cui verrà scritto il risultato della query.

Tipo: stringa

Campo obbligatorio: sì

DimensionMappings

Consente di mappare le colonne dal risultato della query a una dimensione nella tabella di destinazione.

Tipo: matrice di oggetti DimensionMapping

Campo obbligatorio: sì

TableName

Nome della tabella Timestream in cui verrà scritto il risultato della query. La tabella deve trovarsi all'interno dello stesso database fornito nella configurazione Timestream.

Tipo: stringa

Campo obbligatorio: sì

TimeColumn

Colonna dal risultato della query che deve essere utilizzata come colonna TIME nella tabella di destinazione. Il tipo di colonna per questo dovrebbe essereTIMESTAMP.

Tipo: stringa

Campo obbligatorio: sì

MeasureNameColumn

Nome della colonna delle misure.

Tipo: string

Campo obbligatorio: no

MixedMeasureMappings

Specifica come mappare le misure ai record di più misure.

Tipo: matrice di oggetti MixedMeasureMapping

Membri dell'array: numero minimo di 1 elemento.

Campo obbligatorio: no

MultiMeasureMappings

Mappatura a più misure.

Tipo: oggetto MultiMeasureMappings

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche AWS SDKs, consulta quanto segue:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

TimestreamDestination
Servizio: Amazon Timestream Query

Destinazione per l'interrogazione pianificata.

Indice

DatabaseName

Nome del database Timestream.

Tipo: string

Campo obbligatorio: no

TableName

Nome della tabella Timestream.

Tipo: string

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Type
Servizio: Amazon Timestream Query

Contiene il tipo di dati di una colonna in un set di risultati di query. Il tipo di dati può essere scalare o complesso. I tipi di dati scalari supportati sono numeri interi, booleani, string, double, timestamp, date, time e intervalli. I tipi di dati complessi supportati sono matrici, righe e serie temporali.

Indice

ArrayColumnInfo

Indica se la colonna è una matrice.

Tipo: oggetto ColumnInfo

Campo obbligatorio: no

RowColumnInfo

Indica se la colonna è una riga.

Tipo: matrice di oggetti ColumnInfo

Campo obbligatorio: no

ScalarType

Indica se la colonna è di tipo string, integer, boolean, double, timestamp, date, time. Per ulteriori informazioni, consulta Tipi di dati supportati.

Tipo: stringa

Valori validi: `VARCHAR | BOOLEAN | BIGINT | DOUBLE | TIMESTAMP | DATE | TIME | INTERVAL_DAY_TO_SECOND | INTERVAL_YEAR_TO_MONTH | UNKNOWN | INTEGER`

Campo obbligatorio: no

TimeSeriesMeasureValueColumnInfo

Indica se la colonna è un tipo di dati di una serie temporale.

Tipo: oggetto ColumnInfo

Campo obbligatorio: no

Vedi anche

Per ulteriori informazioni sull'utilizzo di questa opzione API in una delle lingue specifiche, consulta quanto segue AWS SDKs:

- AWS SDKper C++
- AWS SDKper Java V2
- AWS SDKper Ruby V3

Errori comuni

Questa sezione elenca gli errori comuni alle API azioni di tutti i AWS servizi. Per gli errori specifici relativi a un'APIazione per questo servizio, consulta l'argomento relativo a tale API azione.

AccessDeniedException

Non disponi dell'autorizzazione di accesso sufficiente per eseguire questa operazione.

HTTPCodice di stato: 400

IncompleteSignature

La firma della richiesta non è conforme agli AWS standard.

HTTPCodice di stato: 400

InternalFailure

L'elaborazione della richiesta non è riuscita a causa di un errore, un'eccezione o un guasto interno sconosciuto.

HTTPCodice di stato: 500

InvalidAction

L'azione o l'operazione richiesta non è valida. Verifica che l'operazione sia digitata correttamente.

HTTPCodice di stato: 400

InvalidClientTokenId

Il certificato X.509 o AWS l'ID della chiave di accesso fornito non esiste nei nostri archivi.

HTTPCodice di stato: 403

NotAuthorized

Non disponi delle autorizzazioni per eseguire questa azione.

HTTPCodice di stato: 400

OptInRequired

L'ID della chiave di AWS accesso richiede un abbonamento al servizio.

HTTPCodice di stato: 403

RequestExpired

La richiesta ha raggiunto il servizio più di 15 minuti dopo la data indicata sulla richiesta o più di 15 minuti dopo la data di scadenza della richiesta (ad esempio nel caso di pre-firmataURLs), oppure la data indicata sulla richiesta è tra più di 15 minuti.

HTTPCodice di stato: 400

ServiceUnavailable

La richiesta non è riuscita a causa di un errore temporaneo del server.

HTTPCodice di stato: 503

ThrottlingException

La richiesta è stata negata a causa del throttling della richiesta.

HTTPCodice di stato: 400

ValidationError

L'input non soddisfa i vincoli specificati da un AWS servizio.

HTTPCodice di stato: 400

Parametri comuni

L'elenco seguente contiene i parametri utilizzati da tutte le azioni per firmare le richieste di Signature Version 4 con una stringa di query. Qualsiasi parametro specifico di un'operazione è riportato

nell'argomento relativo all'operazione. Per ulteriori informazioni sulla versione 4 di Signature, consulta la AWS APIsezione Richieste di firma nella Guida IAM per l'utente.

Action

azione da eseguire.

Tipo: stringa

Campo obbligatorio: sì

Version

La API versione per cui è scritta la richiesta, espressa nel formato YYYY-MM-DD.

Tipo: stringa

Campo obbligatorio: sì

X-Amz-Algorithm

Algoritmo hash utilizzato per creare la firma della richiesta.

Condizione: specifica questo parametro quando includi le informazioni di autenticazione in una stringa di query anziché nell'intestazione di HTTP autorizzazione.

Tipo: stringa

Valori validi: `AWS4-HMAC-SHA256`

Obbligatorio: condizionale

X-Amz-Credential

Il valore dell'ambito delle credenziali, che è una stringa che include la chiave di accesso, la data, la regione di destinazione, il servizio richiesto e una stringa di terminazione ("aws4_request"). Il valore è espresso nel seguente formato: access_key//region YYYYMMDD/service /aws4_request.

Per ulteriori informazioni, consulta Creare una richiesta firmata nella Guida per l'utente. AWS API IAM

Condizione: Specificate questo parametro quando includete le informazioni di autenticazione in una stringa di query anziché nell'intestazione di HTTP autorizzazione.

Tipo: stringa

Obbligatorio: condizionale

X-Amz-Date

La data utilizzata per creare la firma. Il formato deve essere il formato di base ISO 8601 (YYYYMMDD'T' 'ZHHMMSS'). Ad esempio, la data e l'ora seguenti è un X-Amz-Date valore valido:`20120325T120000Z`.

Condizione: X-Amz-Date è facoltativa per tutte le richieste; può essere utilizzata per sovrascrivere la data utilizzata per la firma delle richieste. Se l'intestazione Date è specificata nel formato base ISO 8601, non X-Amz-Date è obbligatoria. Quando X-Amz-Date viene utilizzata, sovrascrive sempre il valore dell'intestazione Date. Per ulteriori informazioni, consulta Elementi di una firma di AWS API richiesta nella Guida per l'IAMutente.

Tipo: stringa

Obbligatorio: condizionale

X-Amz-Security-Token

Il token di sicurezza temporaneo ottenuto tramite una chiamata a AWS Security Token Service (AWS STS). Per un elenco dei servizi che supportano le credenziali di sicurezza temporanee di AWS STS, consulta Servizi AWS that work with IAM nella Guida per l'IAMutente.

Condizione: se utilizzi credenziali di sicurezza temporanee di AWS STS, devi includere il token di sicurezza.

Tipo: stringa

Obbligatorio: condizionale

X-Amz-Signature

Specifica la firma con codifica esadecimale calcolata dalla stringa da firmare e dalla chiave di firma derivata.

Condizione: specifica questo parametro quando includi le informazioni di autenticazione in una stringa di query anziché nell'HTTPintestazione di autorizzazione.

Tipo: stringa

Obbligatorio: condizionale

X-Amz-SignedHeaders

Speciifica tutte le HTTP intestazioni che sono state incluse come parte della richiesta canonica. Per ulteriori informazioni sulla specificazione delle intestazioni firmate, consulta Creare una richiesta firmata AWS API nella Guida per l'utente. IAM

Condizione: Specificate questo parametro quando includete le informazioni di autenticazione in una stringa di query anziché nell'intestazione di HTTP autorizzazione.

Tipo: stringa

Obbligatorio: condizionale

Cronologia dei documenti

Modifica	Descrizione	Data
Aggiornamento solo della documentazione	È stato aggiornato l'argomento Quote per separare le quote predefinite dai limiti di sistema.	22 ottobre 2024
Amazon Timestream ora supporta le analisi delle query	Timestream ora include il supporto per la funzionalità Query Insights che consente di ottimizzare le query, migliorarne le prestazioni e ridurre i costi.	22 ottobre 2024
Amazon Timestream for InfluxDB si aggiorna a una policy esistente.	Amazon Timestream per InfluxDB ha aggiunto `ec2:DescribeRouteTables` l'azione alla politica gestita `AmazonTimestreamInfluxDBFullAccess` esistente per descrivere le tabelle di routing.	8 ottobre 2024

AmazonTimestreamReadOnlyAccess — Aggiornamento a una politica esistente	Timestream for LiveAnalytics ha aggiunto l'DescribeAccountSettings autorizzazione alla politica AmazonTimestreamReadOnlyAccess gestita per la descrizione Account AWS delle impostazioni.	3 giugno 2024
Amazon Timestream LiveAnalytics per ora supporta le unità di calcolo Timestream () TCUs	Amazon Timestream LiveAnalytics per ora include il supporto per Timestream Compute Units TCUs () per misurare la capacità di calcolo allocata per le tue esigenze di query.	29 aprile 2024
Nuove politiche aggiunte	Amazon Timestream per InfluxDB ha aggiunto due nuove politiche: una che consente al servizio di gestire le interfacce di rete e i gruppi di sicurezza nel tuo account. Per ulteriori informazioni, consulta. AmazonTimestreamInfluxDBServiceRolePolicy Un altro che fornisce l'accesso amministrativo completo per creare, aggiornare, eliminare ed elencare istanze Amazon Timestream InfluxDB e creare ed elencare gruppi di parametri. Per ulteriori informazioni, consulta. AmazonTimestreamInfluxDBFullAccess	14 marzo 2024

Amazon Timestream per InfluxDB è ora disponibile a livello generale.	Questa documentazione riguarda la versione iniziale di Amazon Timestream per InfluxDB.	14 marzo 2024
Gli eventi di Amazon Timestream LiveAnalytics for Query sono disponibili in AWS CloudTrail	Amazon Timestream LiveAnalytics per ora pubblica API gli eventi dei dati di Query su. AWS CloudTrail I clienti possono controllare tutte le API richieste di Query effettuat e nei propri AWS account e visualizzare informazioni come l'IAMutente/ruolo che ha effettuato la richiesta, quando è stata effettuata la richiesta, quali database e tabelle sono state interrogate e l'ID di query della richiesta.	12 settembre 2023
Amazon Timestream per LiveAnalytics UNLOAD	Amazon Timestream LiveAnalytics per ora UNLOAD supporta l'esportazione dei risultati delle query in S3.	12 maggio 2023
Amazon Timestream LiveAnalytics per l'aggiorn amento a una politica esistente.	Autorizzazioni di caricamento in batch aggiunte a una policy gestita.	24 febbraio 2023
Amazon Timestream LiveAnalytics per il caricamen to in batch.	Amazon Timestream LiveAnalytics per ora supporta la funzionalità di caricamento in batch.	24 febbraio 2023

Amazon Timestream LiveAnalytics per ora supporta. AWS Backup	Amazon Timestream LiveAnalytics per ora supporta. AWS Backup	14 dicembre 2022
Amazon Timestream LiveAnalytics per gli aggiornamenti delle politiche gestite AWS	Nuove informazioni sulle politiche AWS gestite e Amazon Timestream LiveAnalytics for, inclusi gli aggiornamenti alle politiche gestite esistenti.	29 novembre 2021
Amazon Timestream LiveAnalytics per il supporto delle query pianificate	Amazon Timestream LiveAnalytics per ora supporta l'esecuzione di una query per tuo conto, in base a una pianificazione.	29 novembre 2021
Amazon Timestream LiveAnalytics for supporta l'archiviazione magnetica.	Amazon Timestream LiveAnalytics per ora supporta l'utilizzo dello storage magnetico per le scritture da tavolo.	29 novembre 2021
Amazon Timestream per record multimisura LiveAnalytics .	Amazon Timestream LiveAnalytics per ora supporta un formato più compatto per l'archiviazione dei dati delle serie temporali.	29 novembre 2021
Amazon Timestream LiveAnalytics per gli aggiornamenti delle politiche gestite AWS	Nuove informazioni sulle politiche AWS gestite e Amazon Timestream LiveAnalytics for, inclusi gli aggiornamenti alle politiche gestite esistenti.	24 maggio 2021

Amazon Timestream LiveAnalytics for è ora disponibile nella regione Europa (Francoforte).	Amazon Timestream LiveAnalytics for è ora disponibile a livello generale nella regione Europa (Francoforte) (). `eu-central-1`	23 Aprile 2021
Amazon Timestream LiveAnalytics per noi ora VPC supporta gli endpoint ().AWS PrivateLink	Amazon Timestream LiveAnalytics per ora supporta l'uso VPC degli endpoint ().AWS PrivateLink	23 marzo 2021
Amazon Timestream ora supporta le query tra tabelle.	Puoi utilizzare Amazon Timestream LiveAnalytics per eseguire query tra tabelle.	10 febbraio 2021
Amazon Timestream LiveAnalytics per ora supporta statistiche avanzate sull'esecuzione delle query.	Amazon Timestream LiveAnalytics per ora supporta statistiche avanzate sull'esecuzione delle query, come la quantità di dati scansionati.	10 febbraio 2021
Amazon Timestream LiveAnalytics per ora supporta funzioni avanzate di serie temporali.	Puoi utilizzare Amazon Timestream LiveAnalytics per SQL eseguire query con funzioni avanzate di serie temporali, come derivati, integrali e correlazioni.	10 febbraio 2021
Amazon Timestream LiveAnalytics per HIPAA ora è conformeISO. PCI	Ora puoi utilizzare Amazon Timestream LiveAnalytics per carichi di lavoro che HIPAA richiedono un'infrastruttura conformeISO. PCI	27 gennaio 2021

Amazon Timestream LiveAnalytics per ora supporta telegraf e grafana open source.	Ora puoi usare Telegraf, l'agente server open source basato su plug-in per la raccolta e il reporting dei parametri, e Grafana, la piattaforma di analisi e monitoraggio open source per i database, con Amazon Timestream per. LiveAnalytics	25 novembre 2020
Amazon Timestream LiveAnalytics for è ora disponibile a livello generale.	Questa documentazione riguarda la versione iniziale di Amazon LiveAnalytics Timestream per.	30 settembre 2020

Cos'è Timestream per InfluxDB?

Amazon Timestream per InfluxDB è un motore di database di serie temporali gestito che consente agli sviluppatori di applicazioni DevOps e ai team di eseguire facilmente database InfluxDB per applicazioni di serie temporali in tempo reale utilizzando l'open source. AWS APIs Con Amazon Timestream per InfluxDB, è facile configurare, gestire e scalare carichi di lavoro di serie temporali in grado di rispondere alle query con un tempo di risposta alle query di una sola cifra di millisecondi.

Amazon Timestream per InfluxDB ti dà accesso alle funzionalità della familiare versione open source di InfluxDB sulla sua filiale 2.x. Ciò significa che il codice, le applicazioni e gli strumenti che già utilizzi oggi con i tuoi database open source InfluxDB esistenti dovrebbero funzionare perfettamente con Amazon Timestream per InfluxDB. Amazon Timestream per InfluxDB può eseguire automaticamente il backup del database e mantenere il software del database aggiornato con la versione più recente. Inoltre, Amazon Timestream for InfluxDB semplifica l'utilizzo della replica per migliorare la disponibilità del database e migliorare la durabilità dei dati. Come per tutti i AWS servizi, non sono richiesti investimenti iniziali e paghi solo per le risorse che utilizzi.

Istanze DB

Un'istanza database è un ambiente di database isolato in esecuzione nel cloud. È l'elemento costitutivo di base di Amazon Timestream per InfluxDB. Un'istanza DB può contenere più database creati dall'utente (o organizzazioni e bucket nel caso dei database InfluxDb 2.x) ed è accessibile utilizzando gli stessi strumenti e applicazioni client che potresti utilizzare per accedere a un'istanza InfluxDB autonoma e autogestita. Le istanze DB sono semplici da creare e modificare con gli strumenti da riga di AWS comando, le operazioni Amazon API Timestream InfluxDB o il. AWS Management Console

> **Note**
>
> Amazon Timestream for InfluxDB supporta l'accesso ai database utilizzando le operazioni Influx e l'APIinterfaccia utente Influx. Amazon Timestream per InfluxDB non consente l'accesso diretto all'host.

Puoi avere fino a 40 istanze Amazon Timestream per InfluxDB.

Ogni istanza DB ha un nome di istanza DB. Questo nome fornito dal cliente identifica in modo univoco l'istanza DB quando interagisce con Amazon Timestream for InfluxDB e i comandi. API AWS CLI Il nome dell'istanza DB deve essere univoco per quel cliente in una regione. AWS

Il nome dell'istanza DB fa parte del DNS nome host assegnato all'istanza da Timestream for InfluxDB. Ad esempio, se specifichi influxdb1 come nome dell'istanza DB, Timestream allocherà automaticamente un endpoint per l'istanza. DNS Un endpoint di esempio è, dov'è il nome dell'istanza. `influxdb1-3ksj4dla5nfjhi.us-east-1.timestream-influxdb.amazonaws.com` `influxdb1`

Nell'endpoint di esempio`influxdb1-3ksj4dla5nfjhi.us-east-1.timestream-influxdb.amazonaws.com`, la stringa `3ksj4dla5nfjhi` è un identificatore di account univoco generato da. AWS L'identificatore `3ksj4dla5nfjhi` dell'esempio non cambia per l'account specificato in una determinata regione. Pertanto, tutte le istanze DB create da questo account condividono lo stesso identificatore fisso. Considera le seguenti funzionalità dell'identificatore fisso:

- Attualmente Timestream for InfluxDB non supporta la ridenominazione delle istanze DB.
- Se elimini e ricrei un'istanza database con lo stesso identificatore di istanza database, l'endpoint è lo stesso.
- Se utilizzi lo stesso account per creare un'istanza database in una regione diversa, l'identificatore generato internamente è diverso perché la regione è diversa, come in `influxdb2.4a3j5du5ks7md2.us-west-1.timestream-influxdb.amazonaws.com`.

Ogni istanza DB supporta solo un motore di database Timestream for InfluxDB.

Quando si crea un'istanza DB, InfluxDB richiede che venga specificato il nome di un'organizzazione. Un'istanza DB può ospitare più organizzazioni e più bucket associati a ciascuna organizzazione.

Amazon Timestream per InfluxDB ti consente di creare un account utente principale e una password per la tua istanza DB come parte del processo di creazione. Questo utente master dispone delle autorizzazioni per creare organizzazioni, bucket e per eseguire operazioni di lettura, scrittura, eliminazione e modifica dei dati. Potrai anche accedere a InfluxUI e recuperare il token dell'operatore al primo accesso. Da lì potrai gestire anche tutti i tuoi token di accesso. È necessario impostare la password dell'utente principale quando si crea un'istanza DB, ma è possibile modificarla in qualsiasi momento utilizzando InfluxAPI, Influx o CLI InfluxUI.

Classi di istanze database

La classe di istanza DB determina la capacità di calcolo e memoria di un'istanza database `fi UbfkyxDB` Amazon Timestream. La classe di istanza database di cui hai bisogno dipende dalla potenza di elaborazione e dai requisiti di memoria specifici.

Una classe di istanza database è costituita dal tipo di classe di istanza database e dalla dimensione. Ad esempio, `db.influx` è un tipo di classe di istanza DB ottimizzato per la memoria adatto ai requisiti di memoria ad alte prestazioni relativi ai carichi di lavoro in esecuzione. InfluxDb All'interno del tipo di classe di `db.influx` istanza, `db.influx.2xlarge` c'è una classe di istanza DB. La dimensione di questa classe è 2xlarge.

Per ulteriori informazioni sui prezzi delle classi di istanze, consulta i prezzi di Amazon Timestream for InfluxDB.

Tipi di classi di istanza database

Amazon Timestream for InfluxDB supporta classi di istanze DB per i seguenti casi d'uso ottimizzati per i casi d'uso di InfluxDB.

- **`db.influx`**—Queste classi di istanze sono ideali per eseguire carichi di lavoro che richiedono molta memoria in database InfluxDB open source

Specifiche hardware per le classi di istanze DB

La terminologia seguente descrive le specifiche hardware per le classi di istanze DB:

- v CPU

  Il numero di unità di elaborazione centrale virtuali (CPUs). Una virtuale CPU è un'unità di capacità che è possibile utilizzare per confrontare le classi di istanze DB.

- Memoria (GiB)

  LaRAM, in gibibyte, è allocata all'istanza DB. Spesso esiste un rapporto costante tra memoria e v. CPU Ad esempio, prendiamo la classe di istanza db.influx, che ha un CPU rapporto memoria/v simile alla classe di istanze EC2 r7g.

- Ottimizzata per Influx

L'istanza database utilizza uno stack di configurazione ottimizzato e offre capacità aggiuntiva dedicata per l'I/O di Amazon EBS. Questa ottimizzazione offre prestazioni ottimali ai volumi EBS, riducendo al minimo i conflitti tra l'I/O di Amazon EBS e altro traffico proveniente dall'istanza.

- Larghezza di banda di rete

  La velocità di rete relativa ad altre classi di istanza database. Nella tabella seguente, puoi trovare dettagli hardware sulle classi di istanze Amazon Timestream for InfluxDB.

Classe di istanze	v CPU	Memoria (GiB)	Storage Type (Tipo di storage)	Larghezza di banda di rete (Gbps)
db.influx.medium	1	8	Influx incluso IOPS	10
db.influx.large	2	16	Influx incluso IOPS	10
db.influx.xlarge	4	32	Influx incluso IOPS	10
db.influx.2xlarge	8	64	Influx incluso IOPS	10
db.influx.4xlarge	16	128	Influx incluso IOPS	10
db.influx.8xlarge	32	256	Influx incluso IOPS	12
db.influx .12xlarge	48	384	Influx incluso IOPS	20
db.influx .16xlarge	64	512	Influx incluso IOPS	25

Archiviazione di istanze InfluxDB

Le istanze DB per Amazon Timestream for InfluxDB IOPS utilizzano i volumi Influx Included per database e archiviazione dei log.

In alcuni casi, il carico di lavoro del database potrebbe non essere in grado di raggiungere il 100% di quello che hai fornito. IOPS Per ulteriori informazioni, consulta Fattori che influenzano le prestazioni di storage. Per ulteriori informazioni sui prezzi di storage di Timestream for InfluxDB, consulta i prezzi di Amazon Timestream.

Amazon Timestream per i tipi di storage InfluxDB

Amazon Timestream for InfluxDB fornisce supporto per un tipo di storage, Influx Included. IOPS Puoi creare Timestream per istanze InfluxDB con un massimo di 16 tebibyte (TiB) di spazio di archiviazione.

Ecco una breve descrizione del tipo di archiviazione disponibile:

- Influx IO Archiviazione inclusa: le prestazioni di archiviazione sono la combinazione delle operazioni di I/O al secondo (IOPS) e della velocità con cui il volume di archiviazione può eseguire letture e scritture (throughput di archiviazione). Sui volumi di storage IOPS inclusi in Influx, Amazon Timestream per InfluxDB offre 3 livelli di storage preconfigurati IOPS con un throughput ottimale e richiesto per diversi tipi di carichi di lavoro.

Dimensionamento delle istanze InfluxDB

La configurazione ottimale di un'istanza Timestream for InfluxDB dipende da molti fattori che includono la velocità di inserimento, le dimensioni dei batch, la cardinalità delle serie temporali, le query simultanee e i tipi di query. Nel tentativo di fornire consigli sul dimensionamento, ci stiamo concentrando su un carico di lavoro esemplare con le seguenti caratteristiche:

- I dati vengono raccolti e scritti da una flotta di agenti Telegraf che raccolgono sistema, memoria, discoCPU, I/O e così via da un data center.

  Ogni richiesta di scrittura contiene 5000 righe.
- I tipi di query eseguite sul sistema sono classificati come query di «complessità moderata». Questa categoria di interrogazioni presenta le seguenti caratteristiche:
  - Hanno più funzioni e una o due espressioni regolari

- Può anche avere clausole raggruppate per o campionare un intervallo di tempo di più settimane.
- L'esecuzione richiede in genere da alcune centinaia di millisecondi a un paio di migliaia di millisecondi.
- CPUfavorisce principalmente le prestazioni delle query.

Classe di istanza	Storage Type (Tipo di storage)	Scrive (righe al secondo)	Letture (interrogazioni al secondo)
db.influx.large	Influx IO incluso 3K	~50.000	<10
db.influx.2xlarge	Influx IO incluso 3K	~150.000	<25
db.influx.4xlarge	Influx IO incluso 3K	~200.000	~25
db.influx.4xlarge	Influx IO incluso 12K	~250.000	~35
db.influx.8xlarge	Influx IO incluso 12K	~500.000	~50
db.influx.12xlarge	Influx IO incluso 12K	<750.000	<100

AWS Regioni e zone di disponibilità

Le risorse di cloud computing Amazon sono ospitate in più ubicazioni in tutto il mondo. Queste località sono composte da AWS regioni e zone di disponibilità. Ogni AWS regione è un'area geografica separata. Ogni AWS regione ha più località isolate note come zone di disponibilità.

> **Note**
>
> Per informazioni su come trovare le zone di disponibilità per una AWS regione, consulta Regioni e zone nella Amazon EC2 User Guide.

Amazon Timestream for InfluxDB ti consente di collocare risorse, come istanze DB e dati, in più posizioni.

Amazon gestisce state-of-the-art data center ad alta disponibilità. Sebbene rari, possono verificarsi dei guasti che influiscano sulla disponibilità delle istanze database che si trovano nello stesso luogo.

Se tutte le istanze database sono ospitate in un singolo luogo interessato da questo errore, nessuna delle istanze database sarà disponibile.

È importante ricordare che ogni AWS regione è completamente indipendente. Qualsiasi attività di Amazon Timestream for InfluxDB che inizi (ad esempio, la creazione di istanze di database o l'elenco delle istanze di database disponibili) viene eseguita solo nella tua regione predefinita corrente. AWS Puoi modificare la Regione AWS predefinita nella console o impostando la variabile di ambiente `AWS_DEFAULT_REGION`. Oppure può essere sovrascritta utilizzando il parametro con (). `--region` AWS Command Line Interface AWS CLI Per ulteriori informazioni, consulta Configurazione di AWS Command Line Interface, in particolare le sezioni relative alle variabili di ambiente e alle opzioni della riga di comando.

Per creare o lavorare con un'istanza database Amazon Timestream for InfluxDB in una regione AWS specifica, utilizza l'endpoint di servizio regionale corrispondente.

AWS Disponibilità regionale

Per ulteriori informazioni sulle AWS regioni in cui è attualmente disponibile Amazon Timestream for InfluxDB e sugli endpoint per ciascuna regione, consulta Endpoint e quote Amazon Timestream.

AWS Progettazione delle regioni

Ogni AWS regione è progettata per essere isolata dalle altre AWS regioni. Questo progetto permette di raggiungere la maggiore stabilità e tolleranza ai guasti possibile.

Quando si visualizzano le risorse, vengono visualizzate solo le risorse legate alla AWS regione specificata. Questo perché AWS le regioni sono isolate l'una dall'altra e non replichiamo automaticamente le risorse tra le AWS regioni.

AWS Zone di disponibilità

Quando crei un'istanza DB, Amazon Timestream per InfluxDB ne sceglie una a caso in base alla configurazione della sottorete. Una zona di disponibilità è rappresentata da un codice AWS regionale seguito da una lettera identificativa (ad esempio,). `us-east-1a`

Utilizza il EC2 comando Amazon `describe-availability` -zones come segue per descrivere le zone di disponibilità all'interno della regione specificata che sono abilitate per il tuo account.

```
aws ec2 describe-availability-zones --region region-name
```

Ad esempio, per descrivere le zone di disponibilità all'interno della regione Stati Uniti orientali (Virginia settentrionale) (us-east-1) abilitate per il tuo account, esegui il comando seguente:

```
aws ec2 describe-availability-zones --region us-east-1
```

Non è possibile scegliere le zone di disponibilità per le istanze DB primarie e secondarie in una distribuzione DB Multi-AZ. Amazon Timesteram per InfluxDB li sceglie per te in modo casuale. Per ulteriori informazioni sulle implementazioni Multi-AZ, consulta.. Configurazione e gestione di un'implementazione Multi-AZ

Fatturazione tramite istanze DB per Amazon Timestream for InfluxDB

Le istanze di Amazon Timestream per InfluxDB vengono fatturate in base ai seguenti componenti:

- Ore dell'istanza DB (all'ora): in base alla classe di istanza DB dell'istanza DB, ad esempio db.influx.large. I prezzi sono calcolati in base a una tariffa oraria, mentre le fatture sono calcolate al secondo e mostrano i valori in formato decimale. L'utilizzo di Amazon Timestream per InfluxDB

viene fatturato in incrementi di 1 secondo, con un minimo di 10 minuti. Per ulteriori informazioni, consulta Classi di istanze DB. Classi di istanze database

- Storage (per GiB al mese): capacità di storage fornita all'istanza DB. Per ulteriori informazioni, consulta Archiviazione di istanze InfluxDB.
- Trasferimento dati (per GB): trasferimento di dati in entrata e in uscita dall'istanza DB da o verso Internet e altre AWS regioni.

Per informazioni sui prezzi di Amazon Timestream for InfluxDB, consulta la pagina dei prezzi di Amazon Timestream for InfluxDB.

Configurazione di Amazon Timestream per InfluxDB

Prima di utilizzare Amazon Timestream for InfluxDB per la prima volta, completa le seguenti attività:

Se disponi già di un AWS account, conosci i requisiti di Amazon Timestream for InfluxDB e preferisci utilizzare le impostazioni predefinite per Amazon Timestream for InfluxDB e IAM Guida VPC Guida introduttiva a Timestream for InfluxDB introduttiva ad Amazon Timestream for InfluxDB.

AWS Registrati per creare un account

Se non disponi di un AWS account, completa i seguenti passaggi per crearne uno.

Per creare un AWS account

- Vai alla pagina di AWS accesso.
- Scegli Crea un nuovo account e segui le istruzioni.

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

Quando si registra un AWS account, viene creato un AWS account utente root. L'utente root ha accesso a tutti i AWS servizi e le risorse dell'account. Come best practice di sicurezza, assegna l'accesso amministrativo a un utente amministrativo e utilizza solo l'utente root per eseguire attività che richiedono l'accesso di un utente root.

AWS ti invia un'email di conferma dopo il completamento della procedura di registrazione. In qualsiasi momento, puoi visualizzare l'attività corrente del tuo account e gestirlo accedendo a https://aws.amazon.com/e scegliendo Il mio account.

Gestione degli utenti

Crea un utente amministrativo

Creazione di un utente amministratore

Dopo aver creato un AWS account, crea un utente amministrativo in modo da non utilizzare l'utente root per le attività quotidiane.

Proteggi il tuo AWS account come utente root

Accedi alla Console di AWS gestione come proprietario dell'account scegliendo Utente root e inserendo l'indirizzo email AWS del tuo account. Nella pagina successiva, inserisci la password. Per informazioni sull'accesso tramite utente root, consulta Accesso come utente root nella Guida per l'utente di AWS accesso

Attiva l'autenticazione a più fattori (MFA) per il tuo utente root. Per istruzioni, consulta Abilitare un MFA dispositivo virtuale per l'utente root dell' AWS account (console) nella Guida per l'IAMutente.

Concedi l'accesso programmatico

Gli utenti necessitano di un accesso programmatico se desiderano interagire con l' AWS esterno di. AWS Management Console La modalità con cui concedere l'accesso programmatico dipende dal tipo di utente che accede ad AWS.

Per concedere agli utenti l'accesso programmatico, scegli una delle seguenti opzioni:

Quale utente necessita dell'accesso programmatico?	Per	Come
Identità della forza lavoro (utenti gestiti in IAM Identity Center)	Utilizza credenziali temporane e per firmare le richieste programmatiche a AWS CLI,, AWS SDKs o. AWS APIs	Seguire le istruzioni relative all'interfaccia che si desidera utilizzare.* Per la, vedere AWS CLI

Quale utente necessita dell'accesso programmatico?	Per	Come
		Configurazione dell'utilizzo di Identity AWS CLI Center AWS IAM nel Guida per l'utente dell'inte rfaccia a riga di comando di AWS .* Per AWS SDKs, strumenti e AWS APIs, vedi IAMAutenticazione Identity Center nel AWSSDKse Guida di riferimen to agli strumenti.
IAM	Utilizza credenziali temporane e per firmare le richieste programmatiche a AWS CLISDKs, e. APIs	Seguendo le istruzioni riportate in Utilizzo delle credenziali temporanee con le AWS risorse nella Guida per l'IAMutente.

Quale utente necessita dell'accesso programmatico?	Per	Come
IAM	(Non consigliato) Utilizzate credenziali a lungo termine per firmare richieste programma tiche a AWS CLI, SDKs e. APIs	Seguire le istruzioni relative all'interfaccia che si desidera utilizzare. Per la, vedere AWS CLI Autenticazione tramite credenziali utente IAM nel AWS Guida per l'utente dell'interfaccia a riga di comando .Per gli strumenti AWS SDKs e gli strumenti, vedere Effettua l'autenticazione utilizzando credenziali a lungo termine nel AWS SDKse guida di riferimen to agli strumenti .Per AWS APIs, vedi Gestione delle chiavi di accesso per gli IAM utenti nel Guida per l'utente di IAM.

Determinazione dei requisiti

L'elemento costitutivo di base di Amazon Timestream for Influx è l'istanza DB. In un'istanza DB, crei i tuoi bucket. Un'istanza database fornisce un indirizzo di rete che si chiama un endpoint. Le applicazioni utilizzano questo endpoint per connettersi all'istanza database. Accederai anche alla tua

InfluxUI utilizzando lo stesso endpoint dal tuo browser. Quando crei un'istanza DB, specifichi dettagli come archiviazione, memoria, motore e versione del database, configurazione di rete e sicurezza. Controlli l'accesso alla rete a un'istanza database tramite un gruppo di sicurezza.

Prima di creare un'istanza database e un gruppo di sicurezza, devi conoscere le necessità dell'istanza database e della rete. Ecco alcune cose importanti da considerare:

- Requisiti in termini di risorse: quali sono i requisiti di memoria e processore per l'applicazione o il servizio? Utilizzi queste impostazioni per aiutare a determinare quale classe di istanza database da utilizzare. Per le specifiche sulle classi di istanze DB, consulta Classi di istanze DB.
- VPCe gruppo di sicurezza: molto probabilmente l'istanza DB si troverà in un cloud privato virtuale (VPC). Per connetterti all'istanza database, devi configurare le regole del gruppo di sicurezza. Queste regole sono impostate in modo diverso a seconda del tipo di sistema VPC utilizzato e del modo in cui lo si utilizza. Ad esempio, è possibile utilizzare: un valore predefinito VPC o definito dall'utenteVPC.

  L'elenco seguente descrive le regole per ogni VPC opzione:

  - Predefinito VPC: se il tuo AWS account ha un valore predefinito VPC nella AWS regione corrente, questo VPC è configurato per supportare le istanze DB. Se specifichi l'impostazione predefinita VPC quando crei l'istanza DB, assicurati di creare un gruppo di VPC sicurezza che autorizzi le connessioni dall'applicazione o dal servizio all'istanza database Amazon Timestream for InfluxDB. Usa l'opzione Security Group sulla VPC console o per creare gruppi di sicurezza. AWS CLI VPC Per ulteriori informazioni, vedere Passaggio 3: Creare un gruppo VPC di sicurezza.
- Definito dall'utente VPC: se desideri specificare un elemento definito dall'utente VPC quando crei un'istanza DB, tieni presente quanto segue:
  - Assicurati di creare un gruppo VPC di sicurezza che autorizzi le connessioni dall'applicazione o dal servizio all'istanza DB Amazon Timestream for InfluxDB. Usa l'opzione Security Group sulla VPC console o per creare gruppi di sicurezza. AWS CLI VPC Per informazioni, consulta Fase 3: Creare un gruppo VPC di sicurezza.
  - VPCDevono soddisfare determinati requisiti per ospitare istanze DB, ad esempio avere almeno due sottoreti, ciascuna in una zona di disponibilità separata. Per informazioni, consulta Amazon VPC VPCs e Amazon Timestream for InfluxDB.
- Alta disponibilità: è necessario il supporto per il failover? Su Amazon Timestream for InfluxDB, una distribuzione Multi-AZ crea un'istanza DB primaria e un'istanza DB secondaria in standby in un'altra zona di disponibilità per il supporto del failover. Consigliamo implementazioni Multi-

AZ per carichi di lavoro di produzione per mantenere alta disponibilità. Per scopi di sviluppo e di test, puoi utilizzare un'implementazione che non è Multi-AZ. Per ulteriori informazioni, consulta Implementazioni dell'istanza database Multi-AZ.

- IAMpolitiche: il tuo AWS account dispone di politiche che concedono le autorizzazioni necessarie per eseguire le operazioni di Amazon Timestream per InfluxDB? Se ti connetti AWS tramite IAM credenziali, il tuo IAM account deve disporre di IAM politiche che concedano le autorizzazioni necessarie per eseguire le operazioni del piano di controllo di Amazon Timestream for InfluxDB. Per ulteriori informazioni, consulta Identity and Access Management per Amazon Timestream per InfluxDB.
- Porte aperte: su quale porta TCP /IP viene ascoltato il database? I firewall in alcune aziende possono bloccare le connessioni alla porta predefinita del motore di database. L'impostazione predefinita per Timestream for InfluxDB è 8086.
- AWS Regione: in quale AWS regione vuoi inserire il tuo database? Avere il database in prossimità ravvicinata all'applicazione o servizio web può ridurre la latenza di rete. Per ulteriori informazioni, consulta AWS Regioni e zone di disponibilità.
- Sottosistema di dischi DB: quali sono i requisiti di archiviazione? Amazon Timestream for InfluxDB fornisce tre configurazioni per il tipo di storage Influx incluso: IOPS
  - Influx Io incluso (3k) IOPS SSD
  - Influx Io incluso 12k () IOPS SSD
  - Influx Io incluso 25k () IOPS SSD

  Per ulteriori informazioni sullo storage Amazon Timestream per InfluxDB, consulta Amazon Timestream for InfluxDB Instance Storage. Quando disponi delle informazioni, devi creare un gruppo di sicurezza e istanza database, continua alla fase successiva.

Fornisci l'accesso alla tua istanza DB direttamente da te creando un gruppo di sicurezza VPC

VPCi gruppi di sicurezza forniscono l'accesso alle istanze DB in unVPC. Fungono da firewall per l'istanza database associata, controllando sia il traffico in entrata che in uscita a livello di istanza database. Le istanze database vengono create come impostazione predefinita con un firewall e un gruppo di sicurezza predefinito che protegge l'istanza database.

Prima di connetterti a un'istanza database, devi aggiungere regole al gruppo di sicurezza che consentono di connettersi. Utilizza le informazione di rete e di configurazione per creare regole per permettere l'accesso all'istanza database.

Ad esempio, supponiamo di avere un'applicazione che accede a un database sulla tua istanza DB in un. VPC In questo caso, è necessario aggiungere una TCP regola personalizzata che specifichi l'intervallo di porte e gli indirizzi IP utilizzati dall'applicazione per accedere al database. Se hai un'applicazione su un'EC2istanza Amazon, puoi utilizzare il gruppo di sicurezza che hai configurato per l'EC2istanza Amazon.

Creazione di un gruppo di sicurezza per VPC l'accesso

Per creare un gruppo VPC di sicurezza, accedi a AWS Management Console e scegli VPC.

> **Note**
>
> Assicurati di essere nella VPC console, non nella console Amazon Timesteam for InfluxDB.

- Nell'angolo in alto a destra di AWS Management Console, scegli la AWSregione in cui desideri creare il gruppo di sicurezza e l'istanza DB. VPC Nell'elenco delle VPC risorse Amazon per quella AWS regione, dovresti vedere almeno una VPC o più sottoreti. In caso contrario, non hai un'impostazione predefinita VPC in quella AWS regione. .
- Fare clic su Security Groups (Gruppi di sicurezza) nel pannello di navigazione.
- Scegliere Create Security Group (Crea gruppo di sicurezza).
- Nella sezione Dettagli di base della pagina del gruppo di sicurezza, inserisci il nome e la descrizione del gruppo di sicurezza. Per VPC, scegli l'oggetto VPC in cui vuoi creare la tua istanza DB.
- Per Inbound rules (Regole in entrata), scegli Add rule (Aggiungi regola).
  - Per Tipo, scegli Personalizzato TCP.
  - Per Origine, scegli il nome di un gruppo di sicurezza o inserisci l'intervallo di indirizzi IP (CIDRvalore) da cui accedi all'istanza DB. Se scegli My IP (Il mio IP), questo consente l'accesso all'istanza database dall'indirizzo IP rilevato nel browser.

  Per Source, scegli il nome di un gruppo di sicurezza o digita l'intervallo di indirizzi IP (CIDRvalore) da cui accedi all'istanza DB. Se scegli My IP (Il mio IP), questo consente l'accesso all'istanza database dall'indirizzo IP rilevato nel browser.

- (Facoltativo) In Outbound Rules (Regole in uscita), aggiungi regole per il traffico in uscita. Come impostazione predefinita, tutto il traffico in uscita è permesso.
- Scegliere Create Security Group (Crea gruppo di sicurezza).

È possibile utilizzare questo gruppo VPC di sicurezza come gruppo di sicurezza per l'istanza DB al momento della creazione.

> **Note**
>
> Se utilizzi un valore predefinitoVPC, viene creato automaticamente un gruppo di sottoreti predefinito che comprende tutte VPC le sottoreti. Quando crei un'istanza DB, puoi scegliere il valore predefinito VPC eiifccntf e scegliere quello predefinito per DB Subnet Group.

Quando hai completato i requisiti di configurazione, puoi creare un'istanza database utilizzando i requisiti e il gruppo di sicurezza. Per farlo, segui le istruzioni in Creazione di un'istanza database.

Guida introduttiva a Timestream for InfluxDB

Negli esempi seguenti, puoi scoprire come creare e connetterti a un'istanza DB utilizzando Amazon Timestream for InfluxDB Service.

> **Note**
>
> Assicurati di completare le attività indicate su Configurazione di Amazon Timestream per InfluxDB prima di poter creare o connettere a un'istanza database.

Argomenti

- Creazione e connessione a un'istanza Timestream for InfluxDB
- Creazione di un nuovo Operator Token per la tua istanza InfluxDB

Creazione e connessione a un'istanza Timestream for InfluxDB

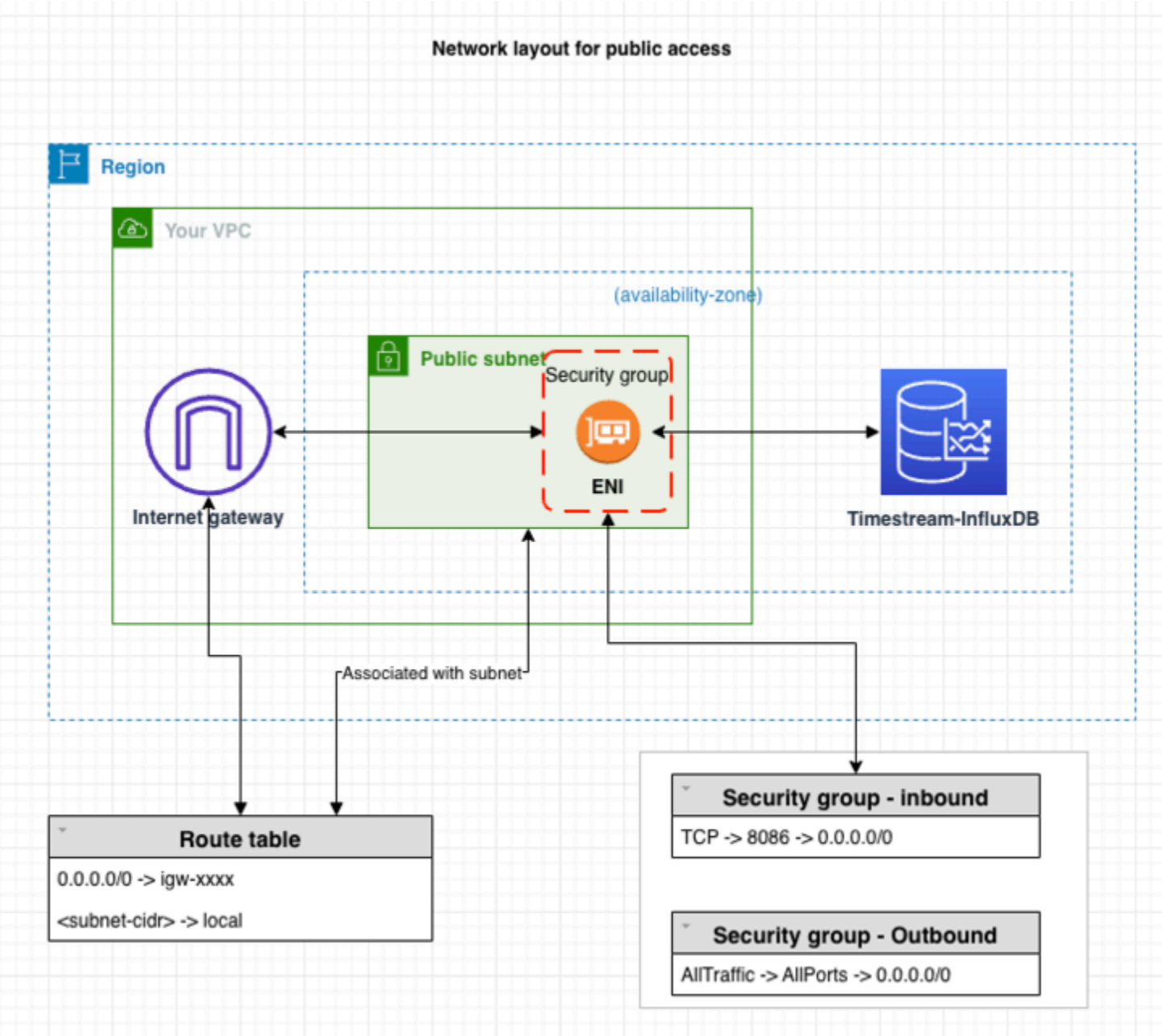
Questo tutorial crea un'EC2istanza Amazon e un'istanza Amazon Timestream per InfluxDB DB. Il tutorial mostra come scrivere dati sull'istanza DB dall'EC2istanza utilizzando il client Telegraf.

Come best practice, questo tutorial crea un'istanza DB privata in un cloud privato virtuale (). VPC Nella maggior parte dei casi, altre risorse della stessaVPC, come EC2 le istanze, possono accedere all'istanza DB, ma le risorse esterne non VPC possono accedervi.

Dopo aver completato il tutorial, c'è una sottorete pubblica e privata in ogni zona di disponibilità del tuo. VPC In una zona di disponibilità, l'EC2istanza si trova nella sottorete pubblica e l'istanza DB si trova nella sottorete privata.

> **Note**
>
> La creazione di un AWS account è gratuita. Tuttavia, completando questo tutorial, potresti incorrere in costi per le AWS risorse che utilizzi. È possibile eliminare queste risorse dopo aver completato l'esercitazione se non sono più necessarie.

Il diagramma seguente mostra la configurazione quando l'accessibilità è pubblica.



Warning

Non è consigliabile utilizzare 0.0.0.0/0 per l'HTTPaccesso, poiché consentite a tutti gli indirizzi IP di accedere alla vostra istanza pubblica di InfluxDB tramite. HTTP Questo approccio non è accettabile nemmeno per un breve periodo in un ambiente di test. Autorizza solo un indirizzo IP o un intervallo di indirizzi specifico per accedere alle tue istanze InfluxDB utilizzando being HTTP for WebUI o access. API

Questo tutorial crea un'istanza DB che esegue InfluxDB con. AWS Management Console Ci concentreremo solo sulla dimensione dell'istanza DB e sull'identificatore dell'istanza DB. Useremo le impostazioni predefinite per le altre opzioni di configurazione. L'istanza DB creata da questo esempio sarà privata.

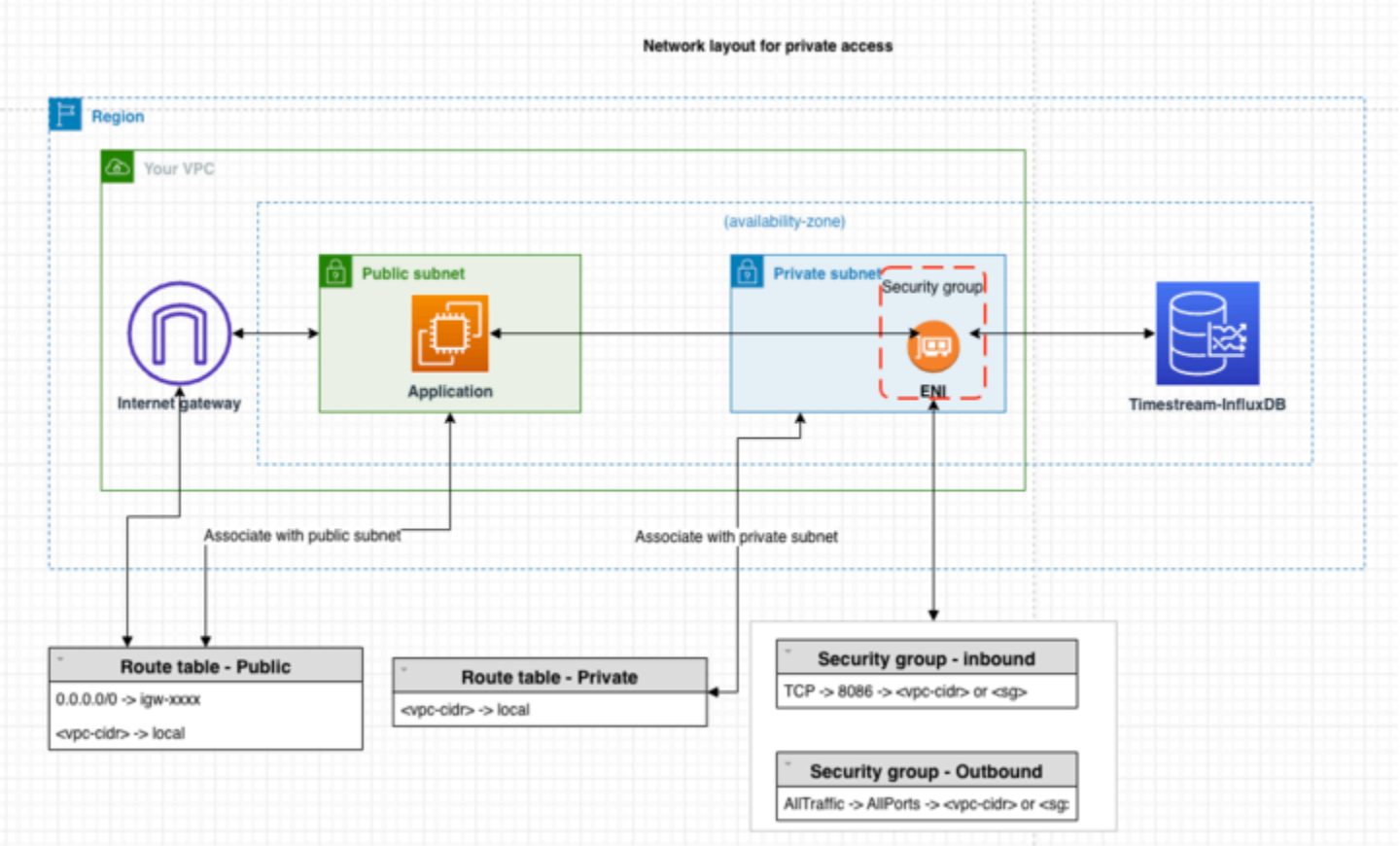
Altre impostazioni che è possibile configurare includono disponibilità, sicurezza e registrazione. Per creare un'istanza DB pubblica, devi scegliere di rendere l'istanza «accessibile pubblicamente» nella sezione Configurazione della connettività. Per informazioni sulla creazione di istanze DB, consulta.. Creazione di un'istanza database

Se la tua istanza non è accessibile pubblicamente, procedi come segue:

- Crea un host sull'istanza attraverso il quale effettuare il tunneling del traffico. VPC
- Imposta il tunneling ssh verso l'istanza. Per ulteriori informazioni, consulta Amazon EC2 Instance Port Forwarding with Systems Manager AWS
- Affinché il certificato funzioni, aggiungi la seguente riga al `/etc/hosts` file del tuo computer client:. `127.0.0.1` Questo è l'DNSindirizzo della tua istanza.
- Connect alla propria istanza utilizzando il nome di dominio completo, ad esempio https: //< DNS >:8086.

> **Note**
>
> Localhost non è in grado di convalidare il certificato perché localhost non fa parte del certificato. SAN

Il diagramma seguente mostra la configurazione quando l'accessibilità è privata:



Prerequisiti

Prima di iniziare, completa le fasi descritte in questa sezione:

- Crea un account AWS .
- Creazione di un utente amministratore.

Fase 1: creare un'EC2istanza Amazon

Crea un'EC2istanza Amazon da utilizzare per connetterti al tuo database.

1. Accedi a AWS Management Console e apri la EC2 console Amazon all'indirizzo https://console.aws.amazon.com/ec2/.
2. Nell'angolo in alto a destra di AWS Management Console, scegli la AWS regione in cui desideri creare l'istanza. EC2
3. Scegli EC2Dashboard, quindi scegli Launch instance.

4. Quando si apre la pagina Avvia un'istanza, scegli le seguenti impostazioni nella pagina Avvia un'istanza.

   a. In Nome e tag, per Nome, inserisci ec2-database-connect.

   b. In Immagini di applicazioni e sistemi operativi (Amazon Machine Image), scegli Amazon Linux, quindi scegli Amazon Linux 2023AMI. Mantieni le selezioni predefinite per le altre opzioni.

   c. In Instance type (Tipo di istanza), scegli t2.micro.

   d. In Key pair (login) (Coppia di chiavi (login), per Key pair name (Nome della coppia di chiavi), scegli una coppia di chiavi esistente. Per creare una nuova coppia di chiavi per l'EC2istanza Amazon, scegli Crea nuova coppia di chiavi e usa la finestra Crea key pair per crearla. Per ulteriori informazioni sulla creazione di una nuova coppia di chiavi, consulta Create a key pair nella Amazon EC2 User Guide for Linux Instances.

   e. Per Consenti SSH il traffico nelle impostazioni di rete, scegli l'origine delle SSH connessioni all'EC2istanza. Puoi scegliere My IP se l'indirizzo IP visualizzato è corretto per SSH le connessioni. Altrimenti, puoi determinare l'indirizzo IP da utilizzare per connetterti alle EC2 istanze VPC utilizzando Secure Shell (SSH). Per determinare il vostro indirizzo IP pubblico, in un'altra finestra o scheda del browser, potete utilizzare il servizio all' https://checkip.amazonaws.comindirizzo. Un esempio di indirizzo IP è 192.0.2.1/32. In molti casi, è possibile connettersi tramite un provider di servizi Internet (ISP) o proteggendo il firewall senza un indirizzo IP statico. In tal caso, accertati di determinare l'intervallo di indirizzi IP utilizzati dai computer client.

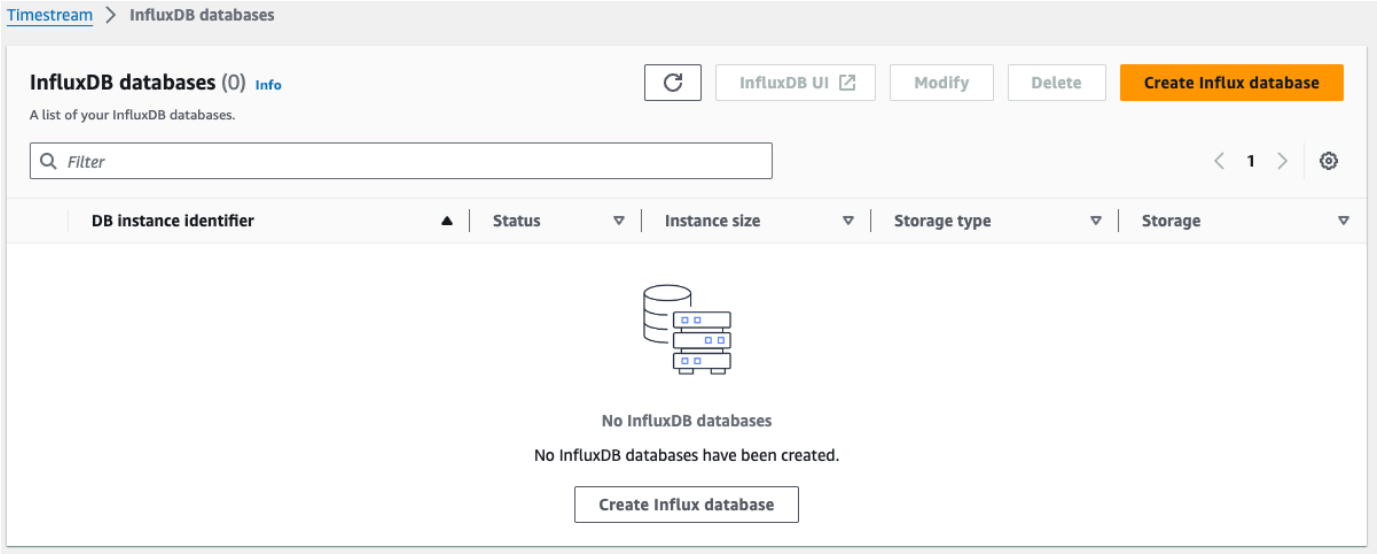
      > ⚠ Warning
      >
      > Non è consigliabile utilizzare 0.0.0.0/0 per SSH l'accesso, poiché consentite a tutti gli indirizzi IP di accedere alle istanze pubbliche utilizzando. EC2 SSH Questo approccio non è accettabile nemmeno per un breve periodo in un ambiente di test. Autorizzate solo uno specifico indirizzo IP o un intervallo di indirizzi da utilizzare per accedere alle istanze. EC2 SSH

Passaggio 2: creare un'istanza DB InfluxDB

L'elemento costitutivo di base di Amazon Timestream per InfluxDB è l'istanza DB. Questo ambiente è il luogo in cui vengono eseguiti i database InfluxDB.

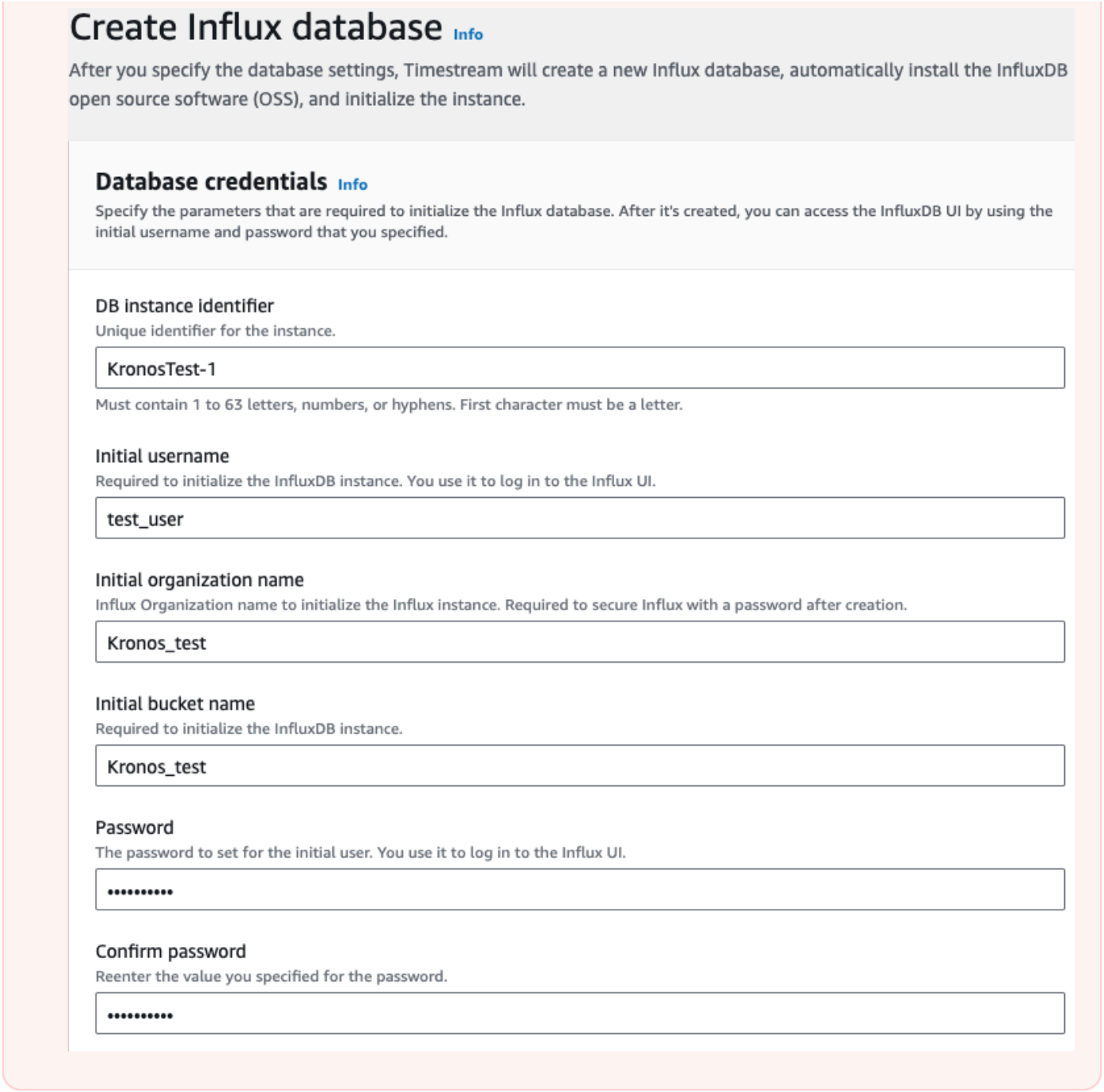
In questo esempio, creerai un'istanza DB che esegue il motore di database InfluxDB con una classe di istanza DB db.influx.large.

1. Accedi AWS Management Console e apri la console Amazon Timestream per InfluxDB all'indirizzo. https://console.aws.amazon.com/timestream/
2. Nell'angolo in alto a destra della console Amazon Timestream for InfluxDB, scegli la AWS regione in cui desideri creare l'istanza DB.
3. Nel pannello di navigazione, scegli InfluxDB Databases.
4. Scegli Crea database Influx.

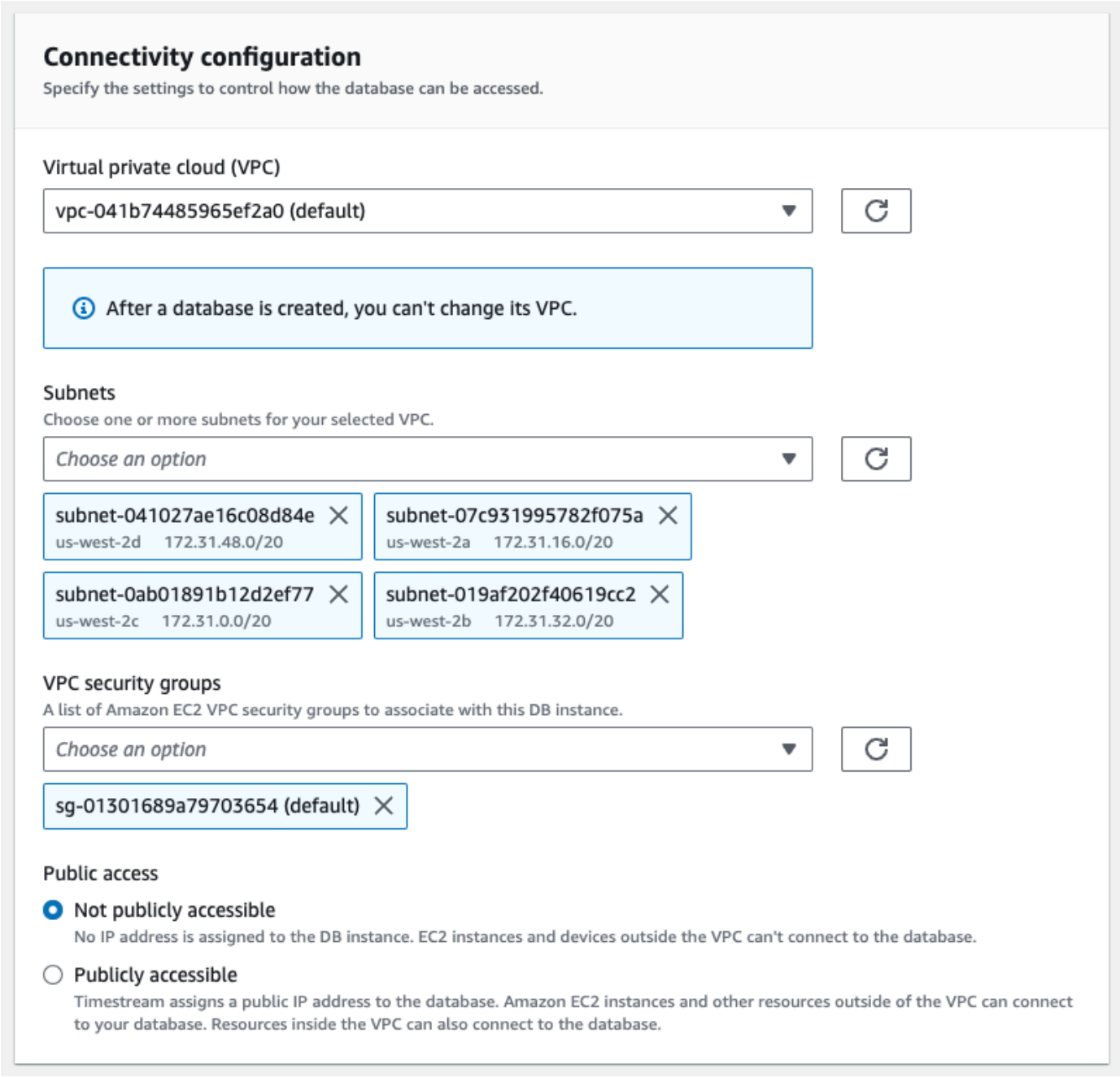


5. Per DB Instance Identifier, inserisci KronosTest -1.
6. Fornisci i parametri di configurazione di base di InfluxDB: nome utente, organizzazione, nome del bucket e password.

> **Important**
>
> Non sarai più in grado di visualizzare la password dell'utente. Non sarai in grado di accedere alla tua istanza e ottenere un token operatore senza la tua password. Se non la registri, potresti doverla modificare. Per informazioni, consulta Creazione di un nuovo Operator Token per la tua istanza InfluxDB.
>
> Se è necessario modificare la password utente dopo che l'istanza DB è disponibile, è possibile modificare l'istanza DB a tale scopo. Per ulteriori informazioni sulla modifica di un'istanza database, consulta Aggiornamento delle istanze database.



7. Per DB Instance Class, seleziona db.influx.large.
8. Per DB Storage Class, seleziona influx Included 3K. IOPS
9. Configura i tuoi log. Per ulteriori informazioni, consulta Configurazione per visualizzare i log di InfluxDB sulle istanze Timestream Influxdb.
10. Nella sezione Configurazione della connettività, assicurati che l'istanza InfluxDB si trovi nella stessa sottorete dell'istanza appena creata. EC2



11. Scegli Crea database Influx.
12. Nell'elenco Database, scegli il nome della tua nuova istanza InfluxDB per mostrarne i dettagli. L'istanza DB ha lo stato Creazione finché non è pronta per l'uso.

È possibile connettersi all'istanza DB quando lo stato cambia in Disponibile. A seconda della classe di istanza database e della quantità di storage, prima che la nuova istanza sia disponibile possono trascorrere fino a 20 minuti.

> **Important**
>
> Al momento, non è possibile modificare la configurazione di calcolo (tipi di istanza) e di archiviazione (tipi di archiviazione) delle istanze esistenti.

Passaggio 3: invia i dati di Telegraf alla tua istanza InfluxDB

Ora puoi iniziare a inviare dati di telemetria alla tua istanza DB InfluxDB utilizzando l'agente Telegraf. In questo esempio, installerai e configurerai un agente Telegraf per inviare le metriche delle prestazioni all'istanza DB InfluxDB.

1. Trova l'endpoint (DNSnome) e il numero di porta per la tua istanza DB.

   a. Accedi alla console di AWS gestione e apri la console Amazon Timestream all'indirizzo. https://console.aws.amazon.com/timestream/

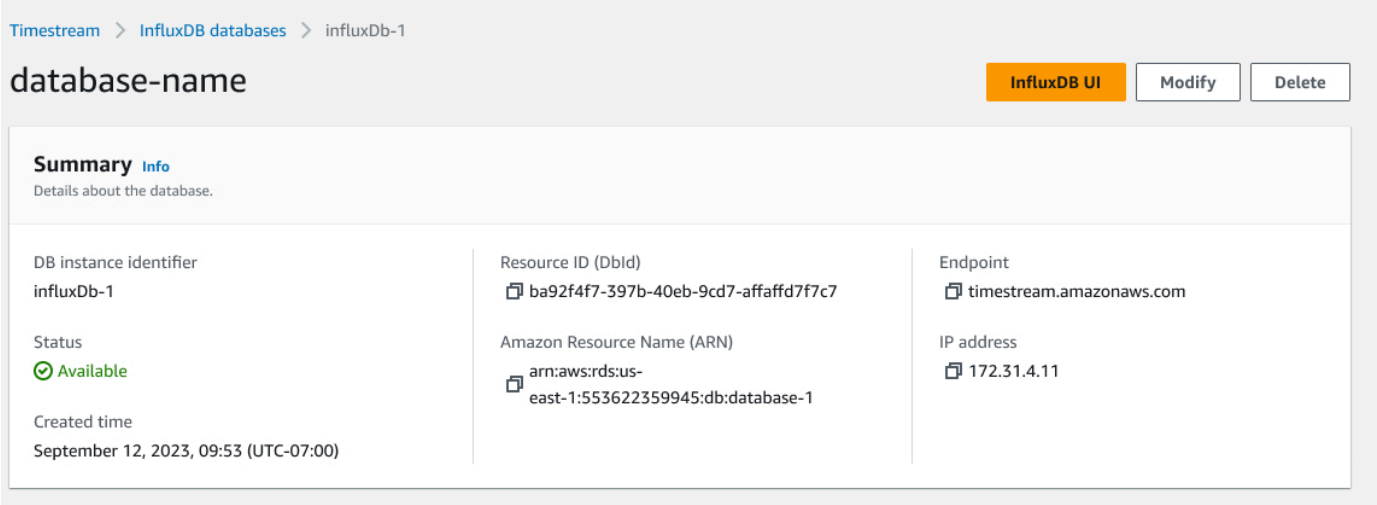
   b. Nell'angolo in alto a destra della console Amazon Timestream, scegli la regione per l'istanza DB. AWS

   c. Nel riquadro di navigazione, scegli InfluxDB Databases.

   d. Scegli il nome dell'istanza DB InfluxDB per visualizzarne i dettagli.

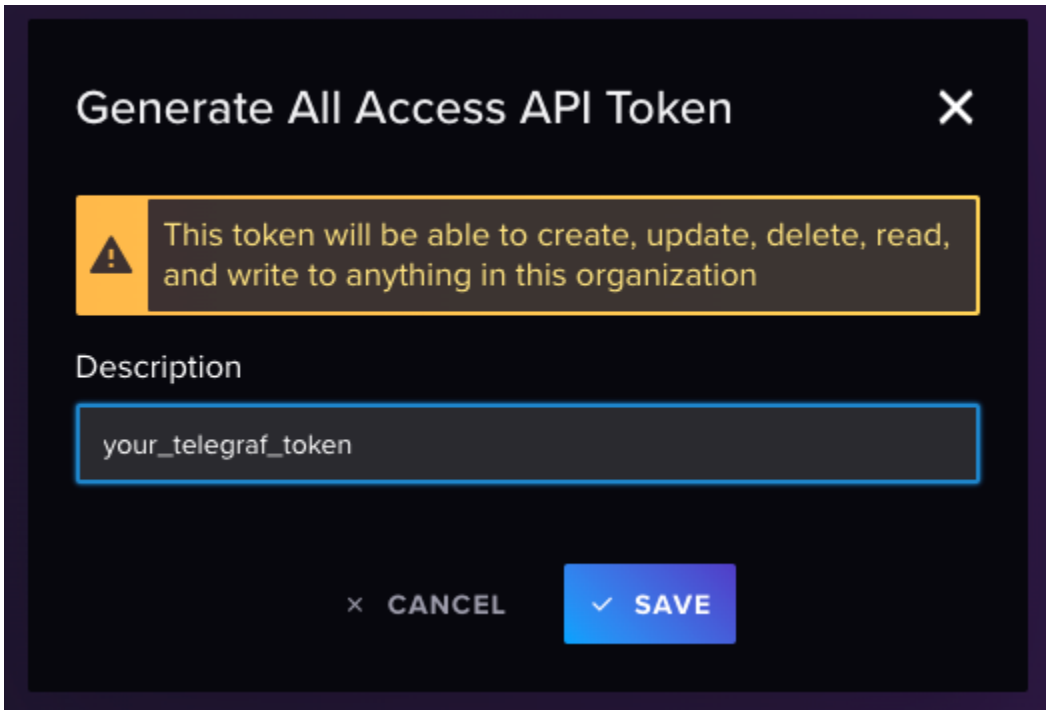
   e. Nella sezione Riepilogo, copia l'endpoint. Annotare anche il numero di porta. Sono necessari sia l'endpoint che il numero di porta per connettersi all'istanza DB (il numero di porta predefinito per InfluxDB è 8086).

2. Quindi, seleziona InfluxDB UI.



3. Si aprirà una nuova finestra del browser in cui dovresti vedere una richiesta di accesso. Inserisci le credenziali che hai usato in precedenza per creare la tua istanza Db InfluxDB.
4. Nel pannello di navigazione, fai clic sulla freccia e seleziona Token. API
5. Per questo test, genera un All Access Token.

> **Note**
>
> Per gli scenari di produzione, consigliamo di creare token con accesso specifico ai bucket richiesti, creati per esigenze specifiche di Telegraf.



6. Il tuo token apparirà sullo schermo.

> **Important**
>
> Assicurati di copiare e salvare il token poiché non potrai più visualizzarlo.

7. Connettiti all'EC2istanza che hai creato in precedenza seguendo i passaggi in Connetti alla tua istanza Linux nella Amazon EC2 User Guide for Linux Instances.

Ti consigliamo di connetterti alla tua EC2 istanza utilizzandoSSH. Se l'utilità SSH client è installata su Windows, Linux o Mac, puoi connetterti all'istanza utilizzando il seguente formato di comando:

```
ssh -i location_of_pem_file ec2-user@ec2-instance-public-dns-name
```

Ad esempio, supponiamo che `ec2-database-connect-key-pair.pem` sia archiviata `/dir1` in Linux e che il pubblico IPv4 DNS dell'EC2istanza lo sia`ec2-12-345-678-90.compute-1.amazonaws.com`. Il tuo SSH comando avrebbe il seguente aspetto:

```
ssh -i /dir1/ec2-database-connect-key-pair.pem ec2-
user@ec2-12-345-678-90.compute-1.amazonaws.com
```

8. Installa l'ultima versione di telegraf sulla tua istanza. Per fare ciò, usa il seguente comando:

```
cat <<EOF | sudo tee /etc/yum.repos.d/influxdata.repo
[influxdata]
name = InfluxData Repository - Stable
baseurl = https://repos.influxdata.com/stable/\$basearch/main
enabled = 1
gpgcheck = 1
gpgkey = https://repos.influxdata.com/influxdata-archive_compat.key
EOF

sudo yum install telegraf
```

9. Configura la tua istanza di Telegraf.

> **Note**
>
> Se telegraf.conf non esiste o contiene una sezione, puoi generarne una `contimestream`:
>
> ```
> telegraf –section-filter agent:inputs:outputs –input-filter cpu:mem –output-
> filter timestream config > telegraf.conf
> ```

a. Modifica il file di configurazione che di solito si trova in. `/etc/telegraf`

```
sudo nano /etc/telegraf/telegraf.conf
```

b. Configura gli input di base perCPU, MEM eDISK.

```
[[inputs.cpu]]
  percpu = true
  totalcpu = true
  collect_cpu_time = false
  report_active = false

[[inputs.mem]]

[[inputs.disk]]
  ignore_fs = ["tmpfs", "devtmpfs", "devfs"]
```

c. Configura il plug-in Output per inviare dati alla tua istanza DB InfluxDB e salvare le modifiche.

```
[[outputs.influxdb_v2]]
   urls = ["https://us-west-2-1.aws.cloud2.influxdata.com"]
   token = "<your_telegraf_token"
   organization = "your_org"
   bucket = "your_bucket"
   timeout = "5s"
```

d. Configura il target Timestream.

```
# Configuration for sending metrics to Amazon Timestream.
[[outputs.timestream]]

  ## Amazon Region and credentials
  region = "us-east-1"
  access_key = "<AWS key here>"
  secret_key = "<AWS secret key here>"
  database_name = "<timestream database name>" # needs to exist

  ## Specifies if the plugin should describe t start.
  describe_database_on_start = false
  mapping_mode = "multi-table" # allows multible tables for each input metrics

  create_table_if_not_exists = true
  create_table_magnetic_store_retention_period_in_days = 365
```

```
  create_table_memory_store_retention_period_in_hours = 24

  use_multi_measure_records = true # Important to use multi-measure records
  measure_name_for_multi_measure_records = "telegraf_measure"
  max_write_go_routines = 25
```

10. Abilita e avvia il servizio Telegraf.

```
$ sudo systemctl enable telegraf
$ sudo systemctl start telegraf
```

Passaggio 4: eliminare l'EC2istanza Amazon e l'istanza DB InfluxDB

Dopo aver esaminato i dati generati da Telegraf utilizzando l'istanza DB InfluxDB con InfluxUI, elimina sia la tua istanza DB che quella InfluxDB in modo da non dover EC2 più addebitare alcun costo.

Per eliminare l'istanza: EC2

1. Accedi a AWS Management Console e apri la EC2 console Amazon all'indirizzo https://console.aws.amazon.com/ec2/.
2. Nel pannello di navigazione, seleziona Instances (Istanze).
3. Seleziona l'EC2istanza, scegli Stato dell'istanza e Termina istanza.
4. Quando viene richiesta la conferma, seleziona Terminate (Interrompi).

Per ulteriori informazioni sull'eliminazione di un'EC2istanza, consulta Terminate your istance nella Amazon EC2 User Guide.

Per eliminare l'istanza DB senza uno snapshot DB finale:

1. Accedi AWS Management Console e apri la console Amazon Timestream per InfluxDB all'indirizzo. https://console.aws.amazon.com/timestream/
2. Nel riquadro di navigazione, scegli InfluxDB Databases.
3. Scegliere l'istanza database da eliminare.
4. In Actions (Azioni), selezionare Delete (Elimina).
5. Completa la conferma e scegli Elimina.

(Facoltativo) Connettiti alla tua istanza DB utilizzando Amazon Managed Grafana

Puoi usare Amazon Managed Grafana per creare dashboard e monitorare le prestazioni delle tue EC2 istanze utilizzando Amazon Timestream per InfluxDB. Amazon Managed Grafana è un servizio completamente gestito per Grafana, una popolare piattaforma di analisi open source che ti consente di interrogare, visualizzare e inviare avvisi su metriche, log e tracce.

Creazione di un nuovo Operator Token per la tua istanza InfluxDB

Se devi ottenere l'Operator Token per la tua nuova istanza InfluxDB, procedi nel seguente modo:

1. Per modificare il token dell'operatore, ti consigliamo di utilizzare Influx. CLI Per le istruzioni, consulta: Installazione e utilizzo di CLI influx.
2. Configura CLI il tuo `--username-password` da utilizzare per poter creare l'operatore:

```
influx config create --config-name CONFIG_NAME1  --host-url "https://
yourinstanceid.eu-central-1.timestream-influxdb.amazonaws.com:8086" --org [YOURORG]
  --username-password [YOURUSERNAME] --active
```

3. Crea il tuo nuovo token operatore. Ti verrà richiesta la password per confermare questo passaggio.

```
influx auth create --org [YOURORG] --operator
```

> **Important**
>
> Una volta creato un nuovo token operatore, dovrai aggiornare qualsiasi client che attualmente utilizza quello vecchio.

Migrazione dei dati da InfluxDB autogestito a Timestream for InfluxDB

Lo script di migrazione Influx è uno script Python che migra i dati tra istanze di OSS InfluxDB, indipendentemente dal fatto che tali istanze siano gestite da o meno. AWS

InfluxDB è un database di serie temporali. InfluxDB contiene punti, che contengono una serie di coppie chiave-valore e un timestamp. Quando i punti sono raggruppati per coppie chiave-valore, formano una serie. Una serie è raggruppata in base a un identificatore di stringa chiamato

misurazione. InfluxDB viene spesso utilizzato per il monitoraggio delle operazioni, IOT i dati e l'analisi. Un bucket è un tipo di contenitore all'interno di InfluxDB per archiviare i dati. AWS-managed InfluxDB è InfluxDB all'interno dell'ecosistema. AWS InfluxDB fornisce InfluxDB v2 API per accedere ai dati e apportare modifiche al database. InfluxDB v2 API è ciò che lo script di migrazione Influx utilizza per migrare i dati.

- Lo script di migrazione Influx può migrare i bucket e i relativi metadati, migrare tutti i bucket da tutte le organizzazioni o eseguire una migrazione completa, che sostituisce tutti i dati sull'istanza di destinazione.
- Lo script esegue il backup dei dati dall'istanza di origine localmente, su qualsiasi sistema esegua lo script, quindi ripristina i dati nell'istanza di destinazione. I dati vengono conservati nelle directory code>influxdb-backup-, una per ogni migrazione.<timestamp></timestamp>
- Lo script fornisce una serie di opzioni e configurazioni, tra cui il montaggio di bucket S3 per limitare l'utilizzo dello storage locale durante la migrazione e la scelta delle organizzazioni da utilizzare durante la migrazione.

Argomenti

Preparazione

La migrazione dei dati per InfluxDB viene eseguita con uno script Python che utilizza le funzionalità di InfluxDB e CLI InfluxDB v2. API L'esecuzione dello script di migrazione richiede la seguente configurazione di ambiente:

- Versioni supportate: è supportata una versione minima di 2.3 di InfluxDB e InfluxCLI.
- Variabili di ambiente dei token
  - Crea la variabile di ambiente `INFLUX_SRC_TOKEN` contenente il token per l'istanza di InfluxDB di origine.
  - Crea la variabile di ambiente `INFLUX_DEST_TOKEN` contenente il token per l'istanza InfluxDB di destinazione.
- Python 3
  - Controlla l'installazione:`python3 --version`.

  - Se non è installato, installalo dal sito Web di Python. È richiesta la versione minima 3.7. In Windows l'alias predefinito di Python 3 è semplicemente python.
  - Le richieste del modulo Python sono obbligatorie. Installalo con: `shell python3 -m pip install requests`
  - TTheÈ richiesto il modulo Python influxdb_client. Installalo con: `shell python3 -m pip install influxdb_client`
- InfluxDB CLI
  - Conferma l'installazione:. `influx version`
  - Se non è installato, segui la guida all'installazione nella documentazione di InfluxDB.

    Aggiungi influx al tuo $. PATH
- Strumenti di montaggio S3 (opzionali)

  Quando si utilizza il montaggio S3, tutti i file di backup vengono archiviati in un bucket S3 definito dall'utente. Il montaggio S3 può essere utile per risparmiare spazio sulla macchina in esecuzione o quando è necessario condividere i file di backup. Se non si utilizza il montaggio S3, omettendo l'`--s3-bucket`opzione, verrà creata una `influxdb-backup-<millisecond timestamp>` directory locale per archiviare i file di backup nella stessa directory in cui è stato eseguito lo script.

  Per Linux: mountpoint-s3.

  Per Windows: rclone (è necessaria una configurazione precedente di rclone).
- Spazio su disco
  - Il processo di migrazione crea automaticamente directory univoche per archiviare set di file di backup e conserva queste directory di backup in S3 o nel file system locale, a seconda degli argomenti del programma forniti.
  - Assicurati che ci sia abbastanza spazio su disco per il backup del database, idealmente il doppio delle dimensioni del database InfluxDB esistente se scegli di omettere l'opzione e utilizzare l'archiviazione locale per il backup e il `--s3-bucket` ripristino.
  - Controlla lo spazio con `df -h (UNIX/Linux)` o controllando le proprietà dell'unità su Windows.
- Connessione diretta

  Assicurati che esista una connessione di rete diretta tra il sistema che esegue lo script di migrazione e i sistemi di origine e destinazione. `influx ping --host <host>`è un modo per verificare una connessione diretta.

Come usare lo script

Un semplice esempio di esecuzione dello script è il comando:

```
python3 influx_migration.py --src-host <source host> --src-bucket <source bucket> --
dest-host <destination host>
```

Che migra un singolo bucket.

Tutte le opzioni possono essere visualizzate eseguendo:

```
python3 influx_migration.py -h
```

Utilizzo

```
shell   influx_migration.py [-h] [--src-bucket SRC_BUCKET] [--dest-bucket DEST_BUCKET]
 [--src-host SRC_HOST] --dest-host DEST_HOST [--full] [--confirm-full] [--src-org
 SRC_ORG] [--dest-org DEST_ORG] [--csv] [--retry-restore-dir RETRY_RESTORE_DIR] [--dir-
name DIR_NAME] [--log-level LOG_LEVEL] [--skip-verify] [--s3-bucket S3_BUCKET]
```

Opzioni

- -confirm-full (opzionale): l'utilizzo di `--full` without `--csv` sostituirà tutti i token, gli utenti, i bucket, i dashboard e qualsiasi altro dato chiave-valore nel database di destinazione con i token, gli utenti, i bucket, i dashboard e qualsiasi altro dato con valore-chiave nel database di origine. `--full`with migra solo tutti i metadati dei bucket e dei bucket, incluse le organizzazioni dei bucket. `--csv` Questa opzione (`--confirm-full`) confermerà una migrazione completa e procederà senza l'input dell'utente. Se questa opzione non viene fornita, ed è `--full` stata fornita e `--csv` non fornita, lo script verrà messo in pausa per l'esecuzione e attenderà la conferma dell'utente. Questa è un'azione critica, procedi con cautela. Il valore predefinito è false (falso).
- -csv (opzionale): indica se utilizzare i file csv per il backup e il ripristino. Se `--full` viene superato anche questo limite, verranno migrati tutti i bucket definiti dall'utente in tutte le organizzazioni, non i bucket di sistema, gli utenti, i token o i dashboard. Se si desidera un'unica organizzazione per tutti i bucket nel server di destinazione anziché per le organizzazioni di origine già esistenti, utilizzare. `--dest-org`
- -dest-bucket DEST _ BUCKET (opzionale): il nome del bucket InfluxDB nel server di destinazione non deve essere un bucket già esistente. Il valore predefinito è o se non viene fornito. `--src-bucket` `None` `--src-bucket`

- -dest-host DEST _HOST: l'host per il server di destinazione. Esempio: http://localhost:8086.
- -dest-org DEST _ ORG (opzionale): il nome dell'organizzazione in cui ripristinare i bucket nel server di destinazione. Se viene omesso, tutti i bucket migrati dal server di origine manterranno la loro organizzazione originale e i bucket migrati potrebbero non essere visibili nel server di destinazione senza creare e cambiare organizzazione. Questo valore verrà utilizzato in tutte le forme di ripristino, che si tratti di un singolo bucket, di una migrazione completa o di qualsiasi migrazione che utilizzi file csv per il backup e il ripristino.
- -dir-name DIR _ NAME (opzionale): il nome della directory di backup da creare. L'impostazione predefinita è `influxdb-backup-<timestamp>`. Non deve già esistere.
- -full (opzionale): indica se eseguire un ripristino completo, sostituendo tutti i dati sul server di destinazione con tutti i dati del server di origine di tutte le organizzazioni, inclusi tutti i dati con valori chiave come token, dashboard, utenti, ecc. Sostituisce e. `--src-bucket --dest-bucket` Se utilizzato con`--csv`, migra solo i dati e i metadati dei bucket. Il valore predefinito è false (falso).
- h, --help: mostra un messaggio di aiuto ed esce.
- -log-level LOG _ LEVEL (opzionale): il livello di registro da utilizzare durante l'esecuzione. Le opzioni sono debug, error e info. Il valore predefinito è info.
- - retry-restore-dir RETRY _ RESTORE _ DIR (opzionale): la directory da utilizzare per il ripristino quando un ripristino precedente non è riuscito, salterà il backup e la creazione della directory, fallirà se la directory non esiste, può essere una directory all'interno di un bucket S3. Se un ripristino fallisce, il percorso della directory di backup che può essere utilizzato per il ripristino verrà indicato in relazione alla directory corrente. I bucket S3 saranno disponibili nel modulo. `influxdb-backups/<s3 bucket>/<backup directory>` Il nome della directory di backup predefinita è. `influxdb-backup-<timestamp>`
- -s3-bucket S3_ BUCKET (opzionale): il nome del bucket S3 da utilizzare per archiviare i file di backup. Su Linux questo è semplicemente il nome del bucket S3, ad esempio, le variabili di ambiente sono state impostate o esistono`my-bucket`. `AWS_ACCESS_KEY_ID` `AWS_SECRET_ACCESS_KEY` `${HOME}/.aws/credentials` In Windows, questo è il nome del dispositivo remoto e del bucket `rclone` configurato, ad esempio. `my-remote:my-bucket` Tutti i file di backup verranno lasciati nel bucket S3 dopo la migrazione in una directory creata. `influxdb-backups-<timestamp>` Una directory di montaggio temporanea denominata `influx-backups` verrà creata nella directory da cui viene eseguito questo script. Se non viene fornita, tutti i file di backup verranno archiviati localmente in una `influxdb-backups-<timestamp>` directory creata da cui viene eseguito questo script.
- -skip-verify (opzionale): TLS ignora la verifica del certificato.

- -src-bucket SRC _ BUCKET (opzionale): il nome del bucket InfluxDB nel server di origine. Se non viene fornito, deve essere fornito. `--full`
- -src-host SRC _ HOST (opzionale): l'host per il server di origine. Il valore predefinito è http://localhost:8086.

Come indicato in precedenza, `mountpoint-s3` `rclone` sono necessari se `--s3-bucket` devono essere utilizzati, ma possono essere ignorati se l'utente non fornisce un valore per`--s3-bucket`, nel qual caso i file di backup verranno archiviati localmente in una directory unica.

Panoramica sulla migrazione

Dopo aver soddisfatto i prerequisiti:

1. Esegui script di migrazione: utilizzando un'app terminale a tua scelta, esegui lo script Python per trasferire i dati dall'istanza InfluxDB di origine all'istanza InfluxDB di destinazione.
2. Fornisci credenziali: fornisci gli indirizzi e le porte dell'host come opzioni. CLI
3. Verifica dei dati: assicurati che i dati vengano trasferiti correttamente mediante:
    a. Utilizzo dell'interfaccia utente di InfluxDB e ispezione dei bucket.
    b. Elencare i bucket con. `influx bucket list -t <destination token> --host <destination host address> --skip-verify`
    c. Utilizzo `influx v1 shell -t <destination token> --host <destination host address> --skip-verify` ed esecuzione `SELECT * FROM <migrated bucket>.<retention period>.<measurement name> LIMIT 100 to view contents of a bucket or SELECT COUNT(*) FROM <migrated bucket>.<retention period>.<measurment name>` per verificare che sia stato migrato il numero corretto di record.

Example Esempio di esecuzione

1. Apri un'app terminale a tua scelta e assicurati che i prerequisiti richiesti siano installati correttamente:

```
~ > python3 --version
Python 3.11.5
~ > influx version
Influx CLI 2.7.3 (git: 8b962c7e75) build_date: 2023-04-28T14:22:49Z
~ > s3fs --version
Amazon Simple Storage Service File System V1.92 (commit:unknown) with GnuTLS(gcrypt)
Copyright (C) 2010 Randy Rizun <rrizun@gmail.com>
License GPL2: GNU GPL version 2 <https://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
~ >
```

2. Vai allo script di migrazione:

```
~ > cd sample-code/influxdb-sample/migration/influxdb
~/sample-code/influxdb-sample/migration/influxdb > ls *.py
influx_migration.py
~/sample-code/influxdb-sample/migration/influxdb >
```

3. Prepara le seguenti informazioni:

   a. Nome del bucket di origine da migrare.

   b. (Facoltativo) Scegliete un nuovo nome di bucket per il bucket migrato nel server di destinazione.

   c. Token root per le istanze di flusso di origine e destinazione.

   d. Indirizzo host delle istanze di influsso di origine e destinazione.

   e. (Facoltativo) Nome e credenziali del bucket S3; le AWS Command Line Interface credenziali devono essere impostate nelle variabili di ambiente del sistema operativo.

```
# AWS credentials (for timestream testing)
    export AWS_ACCESS_KEY_ID="xxx"
    export AWS_SECRET_ACCESS_KEY="xxx"
```

   f. Costruisci il comando come:

```
python3 influx_migration.py --src-bucket [source-bucket-name]  --dest-bucket
 [dest-bucket-name] --src-host [source host] --dest-host [dest host] --s3-
bucket [s3 bucket name](optional) --log-level debug
```

   g. Esegui lo script:

```
~/sample-code/influxdb-sample/migration/influxdb > python3 influx_migration.py --src-bucket primary-bucket --src-host $INFLUXDB_1_HOST --dest-host $KRO
NOS_HOST --dest-bucket new-bucket-name
```

h. Attendi che lo script finisca l'esecuzione.

i. Controlla l'integrità dei dati nel bucket appena migrato,. `performance.txt` Questo file, che si trova nella stessa directory in cui è stato eseguito lo script, contiene alcune informazioni di base sulla durata di ogni passaggio.

Scenari di migrazione

Example Esempio 1: migrazione semplice tramite l'archiviazione locale

Vuoi migrare un singolo bucket, il bucket primario, dal server di origine a un server di destinazione. `(http://localhost:8086)` `(http://dest-server-address:8086)`

Dopo esserti assicurato di avere TCP accesso (per HTTP l'accesso) a entrambe le macchine che ospitano le istanze InfluxDB sulla porta 8086 e di disporre sia dei token di origine che di destinazione e li hai archiviati come variabili di ambiente e, rispettivamente, per una maggiore sicurezza: `INFLUX_SRC_TOKEN` `INFLUX_DEST_TOKEN`

```
python3 influx_migration.py --src-bucket primary-bucket --src-host http://
localhost:8086 --dest-host http://dest-server-address:8086
```

L'output visualizzato dovrebbe essere simile al seguente:

```
INFO: influx_migration.py: Backing up bucket data and metadata using the InfluxDB CLI
2023/10/26 10:47:15 INFO: Downloading metadata snapshot
2023/10/26 10:47:15 INFO: Backing up TSM for shard 1
2023/10/26 10:47:15 INFO: Backing up TSM for shard 8245
2023/10/26 10:47:15 INFO: Backing up TSM for shard 8263
[More shard backups . . .]
2023/10/26 10:47:20 INFO: Backing up TSM for shard 8240
2023/10/26 10:47:20 INFO: Backing up TSM for shard 8268
2023/10/26 10:47:20 INFO: Backing up TSM for shard 2
INFO: influx_migration.py: Restoring bucket data and metadata using the InfluxDB CLI
2023/10/26 10:47:20 INFO: Restoring bucket "96c11c8876b3c016" as "primary-bucket"
2023/10/26 10:47:21 INFO: Restoring TSM snapshot for shard 12772
2023/10/26 10:47:22 INFO: Restoring TSM snapshot for shard 12773
[More shard restores . . .]
2023/10/26 10:47:28 INFO: Restoring TSM snapshot for shard 12825
2023/10/26 10:47:28 INFO: Restoring TSM snapshot for shard 12826
INFO: influx_migration.py: Migration complete
```

La directory `influxdb-backup-<timestamp>` verrà creata e archiviata nella directory da cui è stato eseguito lo script, contenente i file di backup.

Example Esempio 2: migrazione completa utilizzando l'archiviazione locale e la registrazione di debug

Come sopra, tranne per il fatto che si desidera migrare tutti i bucket, i token, gli utenti e i dashboard, eliminare i bucket nel server di destinazione e procedere senza la conferma da parte dell'utente di una migrazione completa del database utilizzando l'opzione. `--confirm-full` Desideri anche vedere quali sono le misurazioni delle prestazioni in modo da abilitare la registrazione del debug.

```
python3 influx_migration.py --full --confirm-full --src-host http://localhost:8086 --
dest-host http://dest-server-address:8086 --log-level debug
```

L'output visualizzato dovrebbe essere simile al seguente:

```
INFO: influx_migration.py: Backing up bucket data and metadata using the InfluxDB CLI
2023/10/26 10:55:27 INFO: Downloading metadata snapshot
2023/10/26 10:55:27 INFO: Backing up TSM for shard 6952
2023/10/26 10:55:27 INFO: Backing up TSM for shard 6953
[More shard backups . . .]
2023/10/26 10:55:36 INFO: Backing up TSM for shard 8268
2023/10/26 10:55:36 INFO: Backing up TSM for shard 2
DEBUG: influx_migration.py: backup started at 2023-10-26 10:55:27 and took 9.41 seconds
 to run.
INFO: influx_migration.py: Restoring bucket data and metadata using the InfluxDB CLI
2023/10/26 10:55:36 INFO: Restoring KV snapshot
2023/10/26 10:55:38 WARN: Restoring KV snapshot overwrote the operator token, ensure
 following commands use the correct token
2023/10/26 10:55:38 INFO: Restoring SQL snapshot
2023/10/26 10:55:39 INFO: Restoring TSM snapshot for shard 6952
2023/10/26 10:55:39 INFO: Restoring TSM snapshot for shard 6953
[More shard restores . . .]
2023/10/26 10:55:49 INFO: Restoring TSM snapshot for shard 8268
2023/10/26 10:55:49 INFO: Restoring TSM snapshot for shard 2
DEBUG: influx_migration.py: restore started at 2023-10-26 10:55:36 and took 13.51
 seconds to run.
INFO: influx_migration.py: Migration complete
```

Example Esempio 3: utilizzo della migrazione completaCSV, organizzazione di destinazione e S3 Bucket

Come nell'esempio precedente, ma utilizzando Linux o Mac e archiviando i file nel bucket S3,. `my-s3-bucket` Questo evita che i file di backup sovraccarichino la capacità di archiviazione locale.

```
python3 influx_migration.py --full --src-host http://localhost:8086 --dest-host http://
dest-server-address:8086 --csv --dest-org MyOrg --s3-bucket my-s3-bucket
```

L'output visualizzato dovrebbe essere simile al seguente:

```
INFO: influx_migration.py: Creating directory influxdb-backups
INFO: influx_migration.py: Mounting influxdb-migration-bucket
INFO: influx_migration.py: Creating directory influxdb-backups/my-s3-bucket/influxdb-
backup-1698352128323
INFO: influx_migration.py: Backing up bucket data and metadata using the InfluxDB v2
 API
INFO: influx_migration.py: Restoring bucket data and metadata from csv
INFO: influx_migration.py: Restoring bucket some-bucket
INFO: influx_migration.py: Restoring bucket another-bucket
INFO: influx_migration.py: Restoring bucket primary-bucket
INFO: influx_migration.py: Migration complete
INFO: influx_migration.py: Unmounting influxdb-backups
INFO: influx_migration.py: Removing temporary mount directory
```

Configurazione di un'istanza database

Questa sezione mostra come configurare la tua istanza Amazon Timestream for InfluxDB DB. Prima di creare un'istanza database, decidere la classe di istanza database che eseguirà l'istanza database. Inoltre, decidi dove verrà eseguita l'istanza DB scegliendo una regione. AWS Quindi, creare l'istanza database.

È possibile configurare un'istanza DB con un gruppo di parametri DB. Un gruppo di parametri DB funge da contenitore per i valori di configurazione del motore applicati a una o più istanze DB.

I parametri disponibili dipendono dal motore DB e dalla versione del motore DB. È possibile specificare un gruppo di parametri DB quando si crea un'istanza DB. puoi modificare un'istanza database per specificarli.

> **Important**
>
> Al momento, non è possibile modificare la configurazione di calcolo (tipi di istanze) e di archiviazione (tipi di archiviazione) delle istanze esistenti.

Creazione di un'istanza database

Utilizzo della console

1. Accedi AWS Management Console e apri Amazon Timestream for InfluxDB.
2. Nell'angolo in alto a destra della console Amazon Timestream for InfluxDB, scegli la AWS regione in cui desideri creare l'istanza DB.
3. Nel pannello di navigazione, scegli InfluxDB Databases.
4. Scegli Crea database Influx.
5. Per DB Instance Identifier, inserisci un nome che identificherà la tua istanza.
6. Fornisci i parametri di configurazione di base di InfluxDB: nome utente, organizzazione, nome del bucket e password.

   > **Important**
   >
   > Il nome utente, l'organizzazione, il nome del bucket e la password verranno archiviati come AWS segreti in Secrets Manager che verrà creato per il tuo account.

   Se è necessario modificare la password utente dopo che l'istanza DB è disponibile, è possibile modificarla utilizzando CLIInflux.
7.
8. Per DB Instance Class, seleziona una dimensione dell'istanza più adatta alle tue esigenze di carico di lavoro.
9. Per DB Storage Class, seleziona una classe di storage adatta alle tue esigenze. In tutti i casi, dovrai solo configurare lo storage allocato.
10. Nella sezione Configurazione della connettività, assicurati che l'istanza di InfluxDB si trovi nella stessa sottorete dei nuovi client che richiedono la connettività all'istanza DB Timestream for InfluxDB. Puoi anche scegliere di rendere la tua istanza DB disponibile pubblicamente.

11. Scegli Crea database Influx.
12. Nell'elenco Database, scegli il nome della tua nuova istanza InfluxDB per mostrarne i dettagli. L'istanza DB ha lo stato Creazione finché non è pronta per l'uso.
13. Quando lo stato cambia in Available (Disponibile), puoi connettersi all'istanza database. A seconda della classe di istanza database e della quantità di storage, prima che la nuova istanza sia disponibile possono trascorrere fino a 20 minuti.

Utilizzando il CLI

Per creare un'istanza DB utilizzando il AWS Command Line Interface, chiamate il `create-db-instance` comando con i seguenti parametri:

```
--name
--vpc-subnet-ids
--vpc-security-group-ids
--db-instance-type
--db-storage-type
--username
--organization
--password
--allocated-storage
```

Per informazioni su ciascuna impostazione, consulta Impostazioni per istanze database.

Example Esempio: utilizzo delle configurazioni di motore predefinite

Per Linux, macOS o Unix:

```
aws timestream-influxdb create-db-instance \
    --name myinfluxDbinstance \
    --allocated-storage 400 \
    --db-instance-type db.influx.4xlarge \
    --vpc-subnet-ids subnetid1 subnetid2
    --vpc-security-group-ids mysecuritygroup \
    --username masterawsuser \
    --password \
    --db-storage-type InfluxIOIncludedT2
```

Per Windows:

```
aws timestream-influxdb create-db-instance \
    --name myinfluxDbinstance \
    --allocated-storage 400 \
    --db-instance-type db.influx.4xlarge \
    --vpc-subnet-ids subnetid1 subnetid2
    --vpc-security-group-ids mysecuritygroup \
    --username masterawsuser \
    --password \
    --db-storage-type InfluxIOIncludedT2
```

Usando il API

Per creare un'istanza DB utilizzando il AWS Command Line Interface, chiamate il `CreateDBInstance` comando con i seguenti parametri:

Per informazioni su ciascuna impostazione, consulta Impostazioni per istanze database.

> **Important**
>
> Parte dell'oggetto di DBInstance risposta ricevete un influxAuthParametersSecretArn. Questo manterrà un ARN account SecretsManager segreto nel tuo account. Verrà compilato solo dopo che le istanze DB di InfluxDB saranno disponibili. Il segreto contiene i parametri di autenticazione Influx forniti durante il processo. `CreateDbInstance` Questa è una READONLY copia in quanto qualsiasi updates/modifications/deletions di questo segreto non ha alcun impatto sull'istanza DB creata. Se elimini questo segreto, la nostra API risposta farà comunque riferimento al segreto eliminatoARN.

Una volta terminata la creazione dell'istanza database di Timestream for InfluxDB, ti consigliamo di scaricare, installare e configurare Influx. CLI

L'influx CLI fornisce un modo semplice per interagire con InfluxDB da una riga di comando. Per istruzioni dettagliate di installazione e configurazione, consulta Use the Influx. CLI

Impostazioni per istanze database

È possibile creare un'istanza DB utilizzando la console, il `create-db-instance` CLI comando o l'operazione `CreateDBInstance` Timestream for InfluxDB. API

La tabella seguente fornisce dettagli sulle impostazioni scelte quando si crea un'istanza DB.

Impostazione della console	Descrizione	CLIopzione e parametro Timestream API
Allocated storage (Storage allocato)	La quantità di archiviazione, in gigabyte, da allocare per l'istanza database. In alcuni casi, l'allocazione di una maggiore quantità di storage per l'istanza database rispetto alla dimensione del database può migliorare le prestazioni di I/O. Per ulteriori informazioni, consulta Archiviazione di istanze InfluxDB.	CLI: `allocated-storage` API: `allocated storage`
Bucket Name (Nome bucket)	Un nome per il bucket per inizializzare l'istanza InfluxDb	CLI: `bucket` API: `bucket`
Tipo di istanza database	La configurazione per l'istanza database. Ad esempio, una classe di istanza DB db.influx.large ha 16 GiB di memoria, 2 di memoria ottimizzata. vCPUs Se possibile, scegliete un tipo di istanza DB sufficien temente grande da contenere in memoria un tipico set di lavoro di interrogazione. Quando i set di lavoro sono conservati in memoria, il sistema può evitare di scrivere sul disco, migliorando le prestazioni. Per ulteriori informazioni, consulta Tipi di classi di istanza database.	CLI: `db-instance-type` API: `Dbinstancetype`
DB instance identifier (Identificatore istanze DB)	Il nome dell'istanza database. Assegna un nome alle istanze database nello stesso modo in cui assegni un nome ai server in locale. L'identificatore dell'istanza DB può contenere fino a 63 caratteri alfanumerici e deve essere univoco per il tuo account nella AWS regione che hai scelto.	CLI: `db-instance-identifier` API: `Dbinstanceidentifier`

Impostazione della console	Descrizione	CLIopzione e parametro Timestream API
DB parameter group (Gruppo di parametri database)	Un gruppo di parametri per l'istanza database. Puoi scegliere il gruppo di parametri predefiniti o puoi creare un gruppo di parametri personalizzato. Per ulteriori informazioni, consulta Utilizzo di gruppi di parametri di database.	CLI: `db-parameter-group-name` API: `DBParameterGroupName`
Impostazione di consegna dei log	Il nome del bucket S3 in cui verranno archiviati i log di InfluxDB.	CLI: `LogDeliveryConfiguration` API: `log-delivery-configuration`
Multi-AZ deployment (Implementazione Multi-AZ)	Create a standby instance (Crea un'istanza standby) per creare una replica secondaria passiva dell'istanza database in un'altra zona di disponibilità per il supporto per il failover. Consigliamo Multi-AZ per carichi di lavoro di produzione per mantenere alta disponibilità. Per lo sviluppo e il testing, è possibile scegliere Do not create a standby instance (Non creare un'istanza database). Per ulteriori informazioni, consulta Configurazione e gestione di un'implementazione Multi-AZ.	CLI: `MultiAz` API: `multi-az`
Password	Questa sarà la password d'uso principale utilizzata per inizializzare l'istanza Db di InfluxDB. Utilizzerai questa password per accedere a InfluxUI e ottenere il token dell'operatore.	CLI: `password` API: `password`

Impostazione della console	Descrizione	CLIopzione e parametro Timestream API
Accesso pubblico	Sì, assegna all'istanza DB un indirizzo IP pubblico, il che significa che è accessibile al di fuori diVPC. Per essere accessibile pubblicamente, l'istanza DB deve anche trovarsi in una sottorete pubblica diVPC. No, per rendere l'istanza DB accessibile solo dall'interno diVPC. Per connettersi a un'istanza DB dall'esternoVPC, l'istanza DB deve essere accessibile pubblicamente. Inoltre, l'accesso deve essere concesso utilizzando le regole in ingresso del gruppo di sicurezza dell'istanza database e devono essere soddisfatti altri requisiti.	CLI: `publicly-accessible` API: `PubliclyAccessible`
Storage Type (Tipo di storage)	Il tipo di storage per l'istanza DB Puoi scegliere tra 3 diversi tipi di storage IOPS incluso Provisioned influx in base ai requisiti dei tuoi carichi di lavoro: * Influx (incluso): 3.000 IOPS IOPS * Flusso incluso 12000 IOPS IOPS * Incluso 16000 INflux IOPS IOPS Per ulteriori informazioni, consulta Archiviazione di istanze InfluxDB.	CLI: `db-storage-type` API: `DbStorageType`
Nome utente iniziale	Questo sarà l'utente principale con cui inizializzare la tua istanza DB InfluxDB. Utilizzerai questo nome utente per accedere a InfluxUI e ottenere il token dell'operatore.	CLI: `username` API: `Username`

Impostazione della console	Descrizione	CLIopzione e parametro Timestream API
Sottoreti	Una sottorete vpc da associare a questa istanza DB.	CLI: `vpc-subnet-ids` API: `VPCSubnetIds`
VPCGruppo di sicurezza (firewall)	I gruppi di sicurezza da associare alle istanze database.	CLI: `vpc-security-group-ids` API: `VPCSecurityGroupIds`

Connessione a un'istanza database Amazon Timestream for InfluxDB

Prima di poter connettersi a un'istanza database, è necessario creare l'istanza database. Per informazioni, consultare Creazione di un'istanza database. Dopo che Amazon Timestream ha effettuato il provisioning della tua istanza DB, utilizza API InfluxDB, CLI Influx o qualsiasi client o utilità compatibile per InfluxDB per connetterti all'istanza DB.

Argomenti

- Ricerca delle informazioni di connessione per un'istanza database Amazon Timestream for InfluxDB
- Opzioni di autenticazione del database
- Uso di gruppi di parametri

Ricerca delle informazioni di connessione per un'istanza database Amazon Timestream for InfluxDB

Le informazioni di connessione per un'istanza DB includono l'endpoint, la porta, il nome utente, la password e un token di accesso valido, ad esempio l'operatore o tutti i token di accesso. Ad esempio, per un'istanza DB InfluxDB, supponiamo che il valore dell'endpoint sia.

`influxdb1-123456789.us-east-1.timestream-influxdb.amazonaws.com` In questo caso, il valore della porta è 8086 e l'utente del database è admin. Date queste informazioni, è possibile specificare i seguenti valori in una stringa di connessione:

- Per l'host o il nome o DNS il nome dell'host, specificare`influxdb1-123456789.us-east-1.timestream-influxdb.amazonaws.com`.
- Per porta, specificare 8086.
- Per utente, specifica admin.
- Per la password, specifica quella che hai fornito durante la creazione dell'istanza DB.

> **Important**
>
> Quando hai creato l'istanza Timestream for InfluxDB Db, ricevi una parte dell'oggetto di DBInstance risposta. `influxAuthParametersSecretArn` Questo manterrà un messaggio segreto nel tuo account. SecretsManager Verrà compilato solo dopo che le istanze DB di InfluxDB saranno disponibili. Il segreto contiene i parametri di autenticazione Influx forniti durante il processo. `CreateDbInstance` Questa è una READONLY copia in quanto qualsiasi updates/modifications/deletions di questo segreto non ha alcun impatto sull'istanza DB creata. Se elimini questo segreto, la nostra API risposta farà comunque riferimento all'arn segreto eliminato.

L'endpoint è univoco per ogni istanza database e i valori della porta e dell'utente possono variare. Per connetterti a un'istanza DB, puoi utilizzare InfluxCLI, Influx o qualsiasi altro client compatibile con API InfluxDB.

Per trovare le informazioni di connessione per un'istanza DB, usa la console di gestione. AWS È inoltre possibile utilizzare il AWS comando Command Line Interface (AWS CLI) o l'`describe-db-instances`operazione API `GetDBInstance` Timestream-InfluxDB.

Usando il AWS Management Console

1. Accedi AWS Management Console e apri la console Amazon Timestream.
2. Nel riquadro di navigazione, scegli InfluxDB Databases per visualizzare un elenco delle tue istanze DB.
3. Scegliere il nome dell'istanza database per visualizzarne i dettagli.

4. Nella sezione Riepilogo, copia l'endpoint. Annotare anche il numero di porta. L'endpoint e il numero di porta sono necessari per la connessione all'istanza database.

Se hai bisogno di trovare le informazioni sul nome utente e sulla password, scegli la scheda Dettagli di configurazione e scegli `influxAuthParametersSecretArn` per accedere al tuo Secrets Manager.

Usando il CLI

- Per trovare le informazioni di connessione per un'istanza DB InfluxDB utilizzando AWS CLI, chiamate il get-db-instance comando. Nella chiamata, richiedi l'ID dell'istanza DB, l'endpoint, la porta e il segreto. influxAuthParameters

  Per Linux, macOS o Unix:

  ```
  aws timestream-influxdb get-db-instance --identifier id \
   --query "[name,endpoint,influxAuthParametersSecretArn]"
  ```

  Per Windows:

  ```
  aws timestream-influxdb get-db-instance --identifier id \
   --query "[name,endpoint,influxAuthParametersSecretArn]"
  ```

  L'output visualizzato dovrebbe essere simile al seguente. Per accedere alle informazioni sul nome utente dovrai controllare il`InfluxAuthParameterSecret`.

  ```
  [
      [
          "mydb",
          "mydb-123456789012.us-east-1.timestream-influxdb.amazonaws.com",
          8086,
      ]
  ]
  ```

Creazione di token di accesso

Con queste informazioni sarai in grado di connetterti e consentire alla tua istanza di recuperare o creare i tuoi token di accesso. Esistono diversi modi per raggiungere questo obiettivo:

Utilizzando il CLI

1. Se non l'hai già fatto, scarica, installa e configura l'influsso CLI.
2. Quando configuri la configurazione di Influx, usa per l'CLIautenticazione. `--username-password`

```
influx config create --config-name YOUR_CONFIG_NAME  --host-url "https://
yourinstance.timestream-influxdb.amazonaws.com:8086"  --org yourorg --username-
password admin --active
```

3. Usa il comando influx auth create per ricreare il tuo token operatore. Tieni presente che questo processo invaliderà il vecchio token dell'operatore.

```
influx auth create --org kronos --operator
```

4. Una volta ottenuto il token dell'operatore, puoi utilizzare il comando influx auth list per visualizzare i token di tutti i tuoi token. È possibile utilizzare il comando influx auth create per creare un token di accesso completo.

> **Important**
>
> Dovrai eseguire questo passaggio per ottenere prima il token dell'operatore, per poi poter creare nuovi token utilizzando InfluxDB o. API CLI

Utilizzo dell'interfaccia utente Influx

1. Accedi alla tua istanza di Timestream for InfluxDB utilizzando l'endpoint creato per accedere e accedere all'interfaccia utente di InfluxDB. Dovrai utilizzare il nome utente e la password utilizzati per creare l'istanza DB InfluxDB. È possibile recuperare queste informazioni da quanto specificato nell'oggetto di risposta di. `influxAuthParametersSecretArn` `CreateDbInstance`

   In alternativa, puoi aprire InfluxUI dalla console di gestione di Timestream for InfluxDB:

   a. Accedi AWS Management Console e apri la console Amazon Timestream per InfluxDB all'indirizzo. https://console.aws.amazon.com/timestream/

b. Nell'angolo in alto a destra della console Amazon Timestream for InfluxDB, scegli la regione in cui hai creato l' AWS istanza DB.

c. Nell'elenco Database, scegli il nome della tua istanza InfluxDB per mostrarne i dettagli. Nell'angolo in alto a destra, scegli Open Influx UI.

2. Una volta effettuato l'accesso a InfluxUI, vai su Carica dati e poi su APIToken utilizzando la barra di navigazione a sinistra.
3. Scegli + GENERATE API TOKEN e seleziona All Access Token. API
4. Inserisci una descrizione per il API token e scegli SAVE.
5. Copia il token generato e conservalo per conservarlo al sicuro.

> **Important**
>
> Quando si creano token da InfluxUI, i token appena creati verranno mostrati solo una volta. Assicurati di copiarli, altrimenti dovrai ricrearli.

Utilizzo di InfluxDB API

- Invia una richiesta all'API`/api/v2/authorizations`endpoint InfluxDB utilizzando il metodo di richiesta. POST

  Includi quanto segue nella tua richiesta:

  a. Intestazioni:

     i. Autorizzazione: Token < INFLUX _ OPERATOR _ TOKEN >

     ii. Tipo di contenuto: application/json

  b. Corpo della richiesta: JSON corpo con le seguenti proprietà:

     i. status: «attivo»

     ii. descrizione: descrizione del API token

     iii. OrgID: ID dell'organizzazione InfluxDB

     iv. autorizzazioni: matrice di oggetti in cui ogni oggetto rappresenta le autorizzazioni per un tipo di risorsa InfluxDB o una risorsa specifica. Ogni autorizzazione contiene le seguenti proprietà:

A. azione: «leggere» o «scrivere»

B. risorsa: JSON oggetto che rappresenta la risorsa InfluxDB a cui concedere l'autorizzazione. Ogni risorsa contiene almeno la seguente proprietà: OrgID: ID dell'organizzazione InfluxDB

C. tipo: Tipo di risorsa. Per informazioni sui tipi di risorse InfluxDB esistenti, usa the / api/v2/resources endpoint.

L'esempio seguente utilizza `curl` InfluxDB API per generare un token di accesso completo:

```
export INFLUX_HOST=https://influxdb1-123456789.us-east-1.timestream-
influxdb.amazonaws.com
export INFLUX_ORG_ID=<YOUR_INFLUXDB_ORG_ID>
export INFLUX_TOKEN=<YOUR_INFLUXDB_OPERATOR_TOKEN>

curl --request POST \
"$INFLUX_HOST/api/v2/authorizations" \
  --header "Authorization: Token $INFLUX_TOKEN" \
  --header "Content-Type: text/plain; charset=utf-8" \
  --data '{
    "status": "active",
    "description": "All access token for get started tutorial",
    "orgID": "'"$INFLUX_ORG_ID"'",
    "permissions": [
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "authorizations"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "authorizations"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "buckets"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "buckets"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "dashboards"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "dashboards"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "orgs"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "orgs"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "sources"}},
```

```
    {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"sources"}},
    {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "tasks"}},
    {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"tasks"}},
    {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"telegrafs"}},
    {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"telegrafs"}},
    {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "users"}},
    {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"users"}},
    {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"variables"}},
    {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"variables"}},
    {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"scrapers"}},
    {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"scrapers"}},
    {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"secrets"}},
    {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"secrets"}},
    {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"labels"}},
    {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"labels"}},
    {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "views"}},
    {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"views"}},
    {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"documents"}},
    {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"documents"}},
    {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"notificationRules"}},
    {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"notificationRules"}},
    {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"notificationEndpoints"}},
    {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
"notificationEndpoints"}},
```

```
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "checks"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "checks"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "dbrp"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type": "dbrp"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "notebooks"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "notebooks"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "annotations"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "annotations"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "remotes"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "remotes"}},
      {"action": "read", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "replications"}},
      {"action": "write", "resource": {"orgID": "'"$INFLUX_ORG_ID"'", "type":
 "replications"}}
    ]
  }
'
```

Opzioni di autenticazione del database

Amazon Timestream for InfluxDB supporta i seguenti modi per autenticare gli utenti del database:

- Autenticazione con password – La tua istanza database esegue tutte le attività di gestione degli account utente. Puoi creare utenti, specificare password e amministrare i token utilizzando InfluxUI, Influx o influx. CLI API
- Autenticazione con token: l'istanza DB esegue tutta l'amministrazione degli account utente. È possibile creare utenti, specificare la password e amministrare i token utilizzando il token operatore utilizzando Influx e InfluxCLI. API

Connessioni crittografate

È possibile utilizzare Secure Socket Layer (SSL) o Transport Layer Security (TLS) dall'applicazione per crittografare una connessione a un'istanza DB. I certificati necessari per l'TLShandshake tra

InfluxDB e le applicazioni create e gestite dal servizio Kronos. Quando il certificato viene rinnovato, l'istanza viene aggiornata automaticamente con la versione più recente senza richiedere alcun intervento da parte dell'utente.

Uso di gruppi di parametri

Parametri database specificano la modalità di configurazione del database. Ad esempio, i parametri del database possono specificare la quantità di risorse, come la memoria, da allocare a un database.

Puoi gestire la configurazione del database associando le tue istanze DB a gruppi di parametri. Amazon Timestream for InfluxDB definisce gruppi di parametri con impostazioni predefinite. Puoi definire i gruppi di parametri anche con le impostazioni personalizzate.

Panoramica dei gruppi di parametri

Un gruppo di parametri database agisce da container per i valori di configurazione del motore che si applicano a una o più istanze database.

Argomenti

Gruppi di parametri predefiniti e personalizzati

Le istanze database utilizzano gruppi di parametri database. Le sezioni seguenti descrivono la configurazione e la gestione dei gruppi di parametri dell’istanza database.

Creazione di un gruppo di parametri del database

Puoi creare un nuovo gruppo di parametri DB utilizzando AWS Management Console AWS Command Line Interface, the o Timestream. API

Le seguenti limitazioni si applicano al nome del gruppo di parametri database:

- Il nome deve contenere da 1 a 255 lettere, numeri o trattini.
- I nomi del gruppo di parametri predefiniti possono includere un punto, ad esempio `default.InfluxDB.2.7`. Tuttavia, i nomi del gruppo di parametri personalizzati non possono includere un punto.

- Il primo carattere deve essere una lettera.
- Il nome non può iniziare con «dbpg-»
- Il nome non può terminare con un trattino o contenere due trattini consecutivi.
- Se si crea un'istanza DB senza specificare un gruppo di parametri DB, l'istanza DB utilizza i valori predefiniti del motore InfluxDB.

Non puoi modificare le impostazioni dei parametri di un gruppo di parametri predefinito. Puoi invece procedere come descritto di seguito:

1. Crea un nuovo set di parametri.
2. Modifica le impostazioni dei parametri desiderati. Non tutti i parametri del motore di database presenti nel gruppo di parametri possono essere modificati.
3. Aggiorna l'istanza DB per utilizzare il gruppo di parametri personalizzato. Per informazioni sull'aggiornamento di un'istanza DB, consultaAggiornamento delle istanze database.

> **Note**
>
> Se hai modificato l'istanza DB per utilizzare un gruppo di parametri personalizzato e avvii l'istanza DB, Amazon Timestream for InfluxDB riavvia automaticamente l'istanza DB come parte del processo di avvio.
> Al momento, non sarai in grado di modificare i gruppi di parametri personalizzati una volta creati. Se è necessario modificare un parametro, è necessario creare un nuovo gruppo di parametri personalizzato e assegnarlo alle istanze che richiedono questa modifica della configurazione. Se aggiorni un'istanza DB esistente per assegnare un nuovo gruppo di parametri, questo verrà sempre applicato immediatamente e l'istanza verrà riavviata.

Parametri statici e dinamici dell'istanza database

I parametri dell'istanza DB InfluxDB sono sempre statici. Si comportano come segue:

Quando modificate un parametro statico, salvate il gruppo di parametri DB e lo assegnate a un'istanza, la modifica del parametro ha effetto automaticamente dopo il riavvio dell'istanza.

Quando si associa un nuovo gruppo di parametri DB a un'istanza DB, Timestream applica i parametri statici modificati solo dopo il riavvio dell'istanza DB. Attualmente l'unica opzione è applicare immediatamente.

Per informazioni sulla modifica del gruppo di parametri database, consulta Aggiornamento delle istanze database.

Parametri e valori dei parametri supportati

Per determinare i parametri supportati per la tua istanza DB, visualizza i parametri nel gruppo di parametri DB utilizzato dall'istanza DB. Per ulteriori informazioni, consulta Visualizzazione dei valori dei parametri per un gruppo di parametri DB.

Per ulteriori informazioni su tutti i parametri supportati dalla versione open source di InfluxDB, consulta le opzioni di configurazione di InfluxDB. Attualmente potrai modificare solo i seguenti parametri InfluxDB:

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
flux-log-enabled	Include l'opzione per mostrare i log dettagliati per le query Flux	FALSE	true, false	N/D	
a livello di registro	Livello di output del registro. InfluxDB emette voci di registro con livelli di gravità maggiori o uguali al livello specificato.	Info	debug, informazioni, errore	N/D	
nessun compito	Numero di interrogazioni che possono essere	FALSE	true, false	N/D	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
	eseguite contemporaneamente. L'impostazione su 0 consente un numero illimitato di interrogazioni simultanee.				
concorrenza tra interrogazioni	Disabilita il task scheduler. Se le attività problematiche impediscono l'avvio di InfluxDB, utilizzate questa opzione per avviare InfluxDB senza pianificare o eseguire attività.	1.024		N/D	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
query-queue-size	Numero massimo di query consentite nella coda di esecuzione. Quando viene raggiunto il limite di coda, le nuove interrogazioni vengono rifiutate. L'impostazione su 0 consente un numero illimitato di interrogazioni nella coda.	1.024		N/D	
tipo di tracciamento	Abilita il tracciamento in InfluxDB e specifica il tipo di tracciamento. Il tracciamento è disabilitato per impostazione predefinita.	""	log, jaeger	N/D	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
metriche disattivate	Disabilita l'endpoint HTTP / metrics che espone le metriche interne di InfluxDB.	FALSE		N/D	
http-idle-timeout	Durata massima: il server deve mantenere attive le connessioni stabilite in attesa di nuove richieste. Impostato su `0` per nessun timeout.	30 ms	Durata con unità`hours`,,,`milliseconds` `milliseconds` Esempio: `durationType=minutes,value=10`	Ore: -Minimo: 0 -Massimo: 256205 Minuti: -Minimo: 0 -Massimo: 15372286 Secondi: -Minimo: 0 -Massimo: 922337203 Millisecondi: -Minimo: 0 -Massimo: 9223372036685	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
http-read-header-timeout	Durata massima in cui il server deve cercare di leggere le intestazioni per le nuove richieste . HTTP Impostato su `0` «No timeout».	10 secondi	Durata con unità`hours`,, `minutes` `seconds` . Esempio: `durationType=minutes,value=10`	Ore: -Minimo: 0 -Massimo: 256205 Minuti: -Minimo: 0 -Massimo: 15372286 Secondi: -Minimo: 0 -Massimo: 922337203 Millisecondi: -Minimo: 0 -Massimo: 922337203685	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
http-read-timeout	Durata massima in cui il server deve cercare di leggere la totalità delle nuove richieste. Impostato su «No `0` timeout».	0	Durata con unità `hoursminutes nds` . Esempio: `durationType=minutes,value=10`	Ore: -Minimo: 0 -Massimo: 256205 Minuti: -Minimo: 0 -Massimo: 15372286 Secondi: -Minimo: 0 -Massimo: 922337203 Millisecondi: -Minimo: 0 -Massimo: 922337203685	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
http-write-timeout	Durata massima che il server deve dedicare all'elaborazione e alla risposta alle richieste di scrittura. Impostato su `0` «No timeout».	0	Durata con unità `hoursminutes nds` . Esempio: `durationType=minutes,value=10`	Ore: -Minimo: 0 -Massimo: 256205 Minuti: -Minimo: 0 -Massimo: 15372286 Secondi: -Minimo: 0 -Massimo: 922337203 Millisecondi: -Minimo: 0 -Massimo: 922337203685	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
influxql-max-select-buckets	Numero massimo di intervalli temporali raggruppati per periodo che un'istruzione può creare. `SELECT 0` consente un numero illimitato di bucket.	0	Long	Minimo: 0 Massimo: 9.223.372.036.854.775.807	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
influxql-max-select-point	Numero massimo di punti che una dichiarazione può elaborare. `SELECT 0` consente un numero illimitato di punti. InfluxDB controlla il conteggio dei punti ogni secondo (quindi le query che superano il massimo non vengono immediatamente interrotte).	0	Long	Minimo: 0 Massimo: 9.223.372.036.854.775.807	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
influxql-max-select-series	Numero massimo di serie che un'istruzione può restituire. `SELECT 0` consente un numero illimitato di serie.	0	Long	Minimo: 0 Massimo: 9.223.372.036.854.775.807	
pprof - disabilitato	Disabilita l'endpoint. `/debug/pprof` HTTP Questo endpoint fornisce dati di profilazione in fase di esecuzione e può essere utile durante il debug.	FALSE	Booleano	N/D	
query-initial-memory-bytes	Byte iniziali di memoria allocati per una query.	0	Long	Minimo: 0 Massimo: query-memory-bytes	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
query-max-memory-bytes	Numero massimo di byte di memoria totali consentiti per le query.	0	Long	Minimo: 0 Massimo: 9.223.372.036.854.775.807	
query-memory-bytes	Specifica il Time to Live () in minuti per le sessioni utente appena create. TTL	0	Long	Minimo: 0 Massimo: 2.147.483.647	Deve essere maggiore o uguale a. query-initial-memory-bytes
durata della sessione	Specifica il Time to Live (TTL) in minuti per le sessioni utente appena create.	60	Numero intero	Minimo: 0 Massimo: 2880	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
session-renew-disabled	Disabilita l'estensione automatica della sessione di un utente a ogni TTL richiesta. Per impostazione predefinita, ogni richiesta imposta la scadenza della sessione a cinque minuti da oggi. Se disabilitate, le sessioni scadono dopo la durata specificata e l'utente viene reindirizzato alla pagina di accesso, anche se attiva di recente.	FALSE	Booleano	N/D	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
storage-cache-max-memory-dimensione	Dimensione massima (in byte) che la cache di uno shard può raggiungere prima che inizi a rifiutare le scritture.	1073741824	Long	Minimo: 0 Massimo: 549755813888	Deve essere inferiore alla capacità di memoria totale dell'istanza. Si consiglia di impostarla su un valore inferiore al 15% della capacità di memoria totale.
storage-cache-snapshot-memory-dimensione	Dimensione (in byte) alla quale il motore di archiviazione eseguirà lo snapshot della cache e la scriverà in un TSM file per rendere disponibile più memoria.	26214400	Long	Minimo: 0 Massimo: 549755813888	Deve essere inferiore a -size. storage-cache-max-memory

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
storage-cache-snapshot-write-durata del freddo	Durata alla quale il motore di archiviazione eseguirà un'istantanea della cache e la scriverà in un nuovo TSM file se lo shard non ha ricevuto scritture o eliminazioni.	100 ms	Durata con unitàhours,,,milliseconds millisecondsEsempio: durationType=minutes,value=10	Ore: -Minimo: 0 -Massimo: 256205 Minuti: -Minimo: 0 -Massimo: 15372286 Secondi: -Minimo: 0 -Massimo: 922337203 Millisecondi: -Minimo: 0 -Massimo: 922337203685	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
storage-compact-full-write- durata a freddo	Durata alla quale il motore di archiviazione comprimer à tutti TSM i file in uno shard se non ha ricevuto scritture o eliminazioni.	40 ore 0 s	Durata con unità,,,. `hours` `minutes` `seconds` `milliseconds` Esempio: `durationType=minutes,value=10`	Ore: -Minimo: 0 -Massimo: 256205 Minuti: -Minimo: 0 -Massimo: 15372286 Secondi: -Minimo: 0 -Massimo: 922337203 Millisecondi: -Minimo: 0 -Massimo: 922337203 685	
storage-compact-throughput-burst	Limite di velocità (in byte al secondo) che le compattaz ioni possono scrivere su disco. TSM	50331648	Long	Minimo: 0 Massimo: 9.223.372 .036.854. 775.807	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
storage-max-concurrent-compactions	Numero massimo di compattazioni complete e livellate che possono essere eseguite contemporaneamente. Un valore pari al 50% dei `0` risultati `runtime.GOMAXPROCS(0)` utilizzati in fase di esecuzione. Qualsiasi numero superiore a zero limita le compattazioni a quel valore. Questa impostazione non si applica allo snapshot della cache.	0	Numero intero	Minimo: 0 Massimo: 64	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
storage-max-index-log-dimensione del file	Dimensione (in byte) alla quale un file write-ahead log (WAL) di indice verrà compattato in un file indice. Dimensioni inferiori faranno sì che i file di log vengano compattati più rapidamente e comporteranno un minore utilizzo dell'heap a scapito della velocità di scrittura.	1048576	Long	Minimo: 0 Massimo: 9.223.372.036.854.775.807	
storage-no-validate-field-dimensione	Salta la convalida della dimensione e dei campi nelle richieste di scrittura in entrata.	FALSE	Booleano	N/D	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
storage-retention-check-interval	Intervallo di verifica dell'applicazione della politica di conservazione.	300 ms	Durata con unità`hours`,`minutes`,`seconds`,`milliseconds` Esempio: `durationType=minutes,value=10`	N/D	Ore: -Minimo: 0 -Massimo: 256205 Minuti: -Minimo: 0 -Massimo: 15372286 Secondi: -Minimo: 0 -Massimo: 922337203 Millisecondi: -Minimo: 0 -Massimo: 9223372036685

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
storage-series-file-max-concurrent-snapshot-compactions	Numero massimo di compattazioni di istantanee che possono essere eseguite contemporaneamente su tutte le partizioni in serie di un database.	0	Numero intero	Minimo: 0 Massimo: 64	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
storage-series-id-set-dimensione della cache	Dimensione della cache interna utilizzata nell'TSIindice per memorizzare i risultati delle serie calcolate in precedenza. I risultati memorizzati nella cache vengono restituiti rapidamente anziché dover essere ricalcolati quando viene eseguita una query successiva con lo stesso predicato chiave/valore di tag. L'impostazione di questo valore su 0 disabilitera la cache	100	Long	Minimo: 0 Massimo: 9.223.372.036.854.775.807	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
	e potrebbe ridurre le prestazioni delle query.				
storage-wal-max-concurrent-scrive	Numero massimo di operazioni di scrittura WAL nella directory da effettuare contempor aneamente.	0	Numero intero	Minimo: 0 Massimo: 256	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
storage-wal-max-write-ritardo	Tempo massimo di attesa per una richiesta di scrittura nella WAL directory quando viene raggiunto il numero massimo di scritture attive simultanee nella WAL directory. Imposta su per `0` disabilitare il timeout.	10 m	Durata con unità`hours`, `minutes``seconds` . Esempio: `durationType=minutes,value=10`	Ore: -Minimo: 0 -Massimo: 256205 Minuti: -Minimo: 0 -Massimo: 15372286 Secondi: -Minimo: 0 -Massimo: 922337203 Millisecondi: -Minimo: 0 -Massimo: 922337203685	

Parametro	Descrizione	Valore predefinito	Valore	Intervallo valido	Nota
ui-disabilitato	Disabilita l'interfaccia utente (UI) di InfluxDB. L'interfaccia utente è abilitata per impostazione predefinita.	FALSE	Booleano	N/D	

Un'impostazione errata dei parametri in un gruppo di parametri può avere conseguenze negative impreviste, tra cui il peggioramento delle prestazioni e l'instabilità del sistema. Fai sempre attenzione quando modifichi i parametri del database. Prova a modificare le impostazioni dei gruppi di parametri su un'istanza DB di test prima di applicare tali modifiche al gruppo di parametri a un'istanza DB di produzione.

Utilizzo di gruppi di parametri di database

Le istanze database utilizzano gruppi di parametri database. Le sezioni seguenti descrivono la configurazione e la gestione dei gruppi di parametri dell'istanza database.

Argomenti

- Creazione di un gruppo di parametri del database
- Associazione di un gruppo di parametri database a un'istanza database
- Generazione di un elenco di gruppi di parametri del database
- Visualizzazione dei valori dei parametri per un gruppo di parametri del database

Creazione di un gruppo di parametri del database

Usando il AWS Management Console

1. Accedi AWS Management Console e apri la console Amazon Timestream per InfluxDB.
2. Nel pannello di navigazione, scegliParameter groups (Gruppi di parametri).

3. Scegli Create parameter group (Crea gruppo di parametri).
4. Nella casella Group name (Nome gruppo), inserire il nome del nuovo gruppo di parametri database.
5. Nella casella Description (Descrizione), inserire una descrizione per il nuovo gruppo di parametri database.
6. Scegli i parametri da modificare e applica i valori desiderati. Per ulteriori informazioni sui parametri supportati, vedereParametri e valori dei parametri supportati.
7. Seleziona Salva.

Utilizzo del AWS Command Line Interface

- Per creare un gruppo di parametri DB utilizzando il AWS CLI, chiamate il `create-db-parameter-group` comando con i seguenti parametri:

```
--db-parameter-group-name <value>
--description <value>
--endpoint_url <value>
--region <value>
--parameters (list) (string)
```

Example Esempio

Per informazioni su ciascuna impostazione, consulta Impostazioni per istanze database. Questo esempio utilizza le configurazioni di motore predefinite.

```
aws timestream-influxdb create-db-parameter-group
    --db-parameter-group-name YOUR_PARAM_GROUP_NAME\
    --endpoint-url YOUR_ENDPOINT
    --region YOUR_REGION \
    --parameters
 "InfluxDBv2={logLevel=debug,queryConcurrency=10,metricsDisabled=true}" \" \
    --debug
```

Associazione di un gruppo di parametri database a un'istanza database

Puoi creare i tuoi gruppi di parametri database con impostazioni personalizzate. È possibile associare un gruppo di parametri DB a un'istanza DB utilizzando AWS Management Console AWS Command

Line Interface, the o API Timestream-InfluxDB. Ciò è possibile quando crei o modifichi un'istanza database.

Per ulteriori informazioni sulla creazione di un gruppo di parametri database, consulta Creazione di un gruppo di parametri del database. Per informazioni sulla creazione di un'istanza database, consulta Creazione di un'istanza database. Per informazioni sulla modifica di un'istanza DB, consulta.. Aggiornamento delle istanze database

> **Note**
>
> Quando si associa un nuovo gruppo di parametri DB a un'istanza DB, i parametri statici modificati vengono applicati solo dopo il riavvio dell'istanza DB. Attualmente è supportata solo l'applicazione immediata. Timestream for InfluxDB supporta solo parametri statici.

Usando il AWS Management Console

1. Accedi AWS Management Console e apri la console Amazon Timestream per InfluxDB.
2. Nel riquadro di navigazione, scegli InfluxDB Databases, quindi scegli l'istanza DB che desideri modificare.
3. Scegli Aggiorna. Viene visualizzata la pagina Aggiorna istanza DB.
4. Modifica l'impostazione del gruppo di parametri database .
5. Scegliere Continue (Continua) e controllare il riepilogo delle modifiche.
6. Attualmente è supportato solo Applica immediatamente. Questa opzione può causare un'interruzione in alcuni casi poiché riavvierà l'istanza DB.
7. Nella pagina di conferma esaminare le modifiche. Se sono corrette, scegli Aggiorna istanza DB per salvare le modifiche e applicarle. Oppure scegliere Back (Indietro) per cambiare le modifiche o Cancel (Annulla) per annullare le modifiche.

Usando il AWS Command Line Interface

Per Linux, macOS o Unix:

```
aws timestream-influxdb update-db-instance
--identifier YOUR_DB_INSTANCE_ID \
--region YOUR_REGION \
--db-parameter-group-identifier YOUR_PARAM_GROUP_ID \
```

```
--log-delivery-configuration "{\"s3Configuration\": {\"bucketName\":
 \"${LOGGING_BUCKET}\", \"enabled\": false }}"
```

Per Windows:

```
aws timestream-influxdb update-db-instance
--identifier YOUR_DB_INSTANCE_ID ^
--region YOUR_REGION ^
--db-parameter-group-identifier YOUR_PARAM_GROUP_ID ^
--log-delivery-configuration "{\"s3Configuration\": {\"bucketName\":
 \"${LOGGING_BUCKET}\", \"enabled\": false }}"
```

Generazione di un elenco di gruppi di parametri del database

Puoi elencare i gruppi di parametri DB che hai creato per il tuo AWS account.

Usando il AWS Management Console

1. Accedi AWS Management Console e apri la console Amazon Timestream per InfluxDB.
2. Nel riquadro di navigazione scegliere Parameter groups (Gruppi di parametri).
3. I gruppi di parametri database vengono visualizzati in un elenco.

Usando il AWS Command Line Interface

Per elencare tutti i gruppi di parametri DB per un AWS account, utilizzare il AWS Command Line Interface `list-db-parameter-groups` comando.

```
aws timestream-influxdb list-db-parameter-groups --region region
```

Per restituire uno specifico gruppo di parametri DB per un AWS account, usa il AWS Command Line Interface `get-db-parameter-group` comando.

```
aws timestream-influxdb get-db-parameter-group --region region --identifier identifier
```

Visualizzazione dei valori dei parametri per un gruppo di parametri del database

Puoi ottenere un elenco di tutti i parametri in un gruppo di parametri del database e dei rispettivi valori.

Usando il AWS Management Console

1. Accedi AWS Management Console e apri la console Amazon Timestream per InfluxDB.
2. Nel riquadro di navigazione scegliere Parameter groups (Gruppi di parametri).
3. I gruppi di parametri database vengono visualizzati in un elenco.
4. Scegliere il nome del gruppo di parametri per visualizzarne l'elenco di parametri.

Usando il AWS Command Line Interface

Per visualizzare i valori dei parametri per un gruppo di parametri DB, utilizzate il AWS Command Line Interface `get-db-parameters` comando con il seguente parametro obbligatorio.

```
--db-parameter-group-name
```

Utilizzando il API

Per visualizzare i valori dei parametri per un gruppo di parametri DB, utilizzate il API `GetDBParameters` comando Timestream con il seguente parametro obbligatorio.

```
DBParameterGroupName
```

Gestione delle istanze DB

Questa sezione tratta vari aspetti della gestione di Timestream for InfluxDB per garantire prestazioni, disponibilità e capacità di monitoraggio ottimali. Fornisce indicazioni sull'aggiornamento delle configurazioni delle istanze di database, sulla gestione delle implementazioni Multi-AZ e sui processi di failover. Spiega inoltre come eliminare le istanze del database e configurare la visualizzazione dei log per le istanze InfluxDB.

Argomenti

- Aggiornamento delle istanze database
- Manutenzione di un'istanza database
- Eliminazione di un'istanza database
- Implementazioni dell'istanza database Multi-AZ
- Configurazione per visualizzare i log di InfluxDB sulle istanze Timestream Influxdb

Aggiornamento delle istanze database

Puoi aggiornare i seguenti parametri di configurazione dell'istanza Timestream for InfluxDB:

- Classe di istanza
- Il tipo di distribuzione
- Gruppo di parametri
- Configurazione della consegna dei log

> **Important**
>
> Ti consigliamo di testare tutte le modifiche su un'istanza di test prima di modificare l'istanza di produzione per comprenderne l'impatto, specialmente quando si aggiornano le versioni del database. Verifica l'impatto sul database e sulle applicazioni prima di aggiornare le impostazioni. Alcune modifiche richiedono il riavvio dell'istanza DB, con conseguenti tempi di inattività.

Utilizzando il AWS Management Console

1. Accedi AWS Management Console e apri la console Amazon Timestream per InfluxDB.
2. Nel riquadro di navigazione, scegli InfluxDB Databases, quindi scegli l'istanza DB che desideri modificare.
3. Scegli Modifica.
4. Nella pagina Modifica istanza DB, apporta le modifiche desiderate.
5. Scegliere Continue (Continua) e controllare il riepilogo delle modifiche.
6. Scegli Next (Successivo).
7. Rivedi le modifiche.
8. Scegli Modifica istanza per applicare le modifiche.

> **Note**
>
> Queste modifiche richiedono il riavvio dell'istanza Influx DB e in alcuni casi possono causare un'interruzione.

Utilizzando il AWS Command Line Interface

Per aggiornare un'istanza DB utilizzando AWS Command Line Interface, chiamate il `update-db-instance` comando. Specifica l'identificatore istanze DB e i valori per le impostazioni da modificare. Per ulteriori informazioni su ciascuna opzione, consulta Impostazioni per istanze database.

Example Esempio

Il codice seguente modifica mydbinstance impostando un dbparametergroup diverso. Le modifiche vengono applicate immediatamente.

Per Linux, macOS o Unix:

```
aws timestream-influxdb update-db-instance \
    --identifier mydbinstance \
    --db-instance-type desired-instance-type \
    --deployment-type desired-deployment-type \
    --db-parameter-group-name newparamgroup \
    --port 8086
```

Per Windows:

```
aws timestream-influxdb update-db-instance ^
    --identifier mydbinstance ^
    --db-instance-type desired-instance-type ^
    --deployment-type desired-deployment-type ^
    --db-parameter-group-name newparamgroup
    --port 8086
```

Manutenzione di un'istanza database

Periodicamente, Amazon Timestream for InfluxDB esegue la manutenzione sulle risorse Amazon Timestream for InfluxDB. La manutenzione prevede più spesso l'aggiornamento delle seguenti risorse nell'istanza DB:

- Hardware sottostante
- Sistema operativo (OS) sottostante
- Versione del motore del database

Gli aggiornamenti al sistema operativo si verificano generalmente per problemi di sicurezza.

Alcuni elementi di manutenzione richiedono che Amazon Timestream for InfluxDB metta offline l'istanza DB per un breve periodo. Tra le operazioni di manutenzione che richiedono l'impostazione offline di una risorsa si annovera l'applicazione delle patch necessarie al sistema operativo o al database. L'applicazione delle patch necessarie viene pianificata automaticamente solo per le patch correlate alla sicurezza e all'affidabilità dell'istanza. Tali patch si verificano raramente, in genere una volta ogni pochi mesi. Raramente richiedono più di una frazione del periodo di manutenzione.

- Le finestre di manutenzione sono configurate per essere eseguite tutti i giorni tra le 12:00 e le 4:00 ora locale per la regione in cui è ospitata l'istanza.
- Le risorse dei clienti potrebbero essere aggiornate una volta alla settimana in una qualsiasi delle sette finestre di manutenzione della settimana.

Eliminazione di un'istanza database

L'eliminazione di un'istanza DB ha un effetto sulla recuperabilità dell'istanza e sulla disponibilità delle snapshot. Considera le problematiche descritte di seguito:

- Se desideri eliminare tutte le risorse Timestream for InfluxDB, tieni presente che le risorse delle istanze DB comportano costi di fatturazione.
- Quando lo stato di un'istanza DB viene eliminato, il relativo valore del certificato CA non viene visualizzato nella console Timestream for InfluxDB o nell'output dei comandi o delle operazioni Timestream. AWS Command Line Interface API
- Il tempo necessario per eliminare un'istanza DB varia a seconda della quantità di dati eliminati e dell'eventuale acquisizione di un'istantanea finale.

È possibile eliminare un'istanza DB utilizzando il AWS Management Console AWS Command Line Interface, il o il API Timestream. È necessario fornire il nome dell'istanza DB:

Utilizzando il AWS Management Console

1. Accedi AWS Management Console e apri la console Amazon Timestream per InfluxDB.
2. Nel riquadro di navigazione, scegli InfluxDB Databases, quindi scegli l'istanza DB che desideri eliminare.
3. Scegli Elimina.

4. Inserisci conferma nella casella.
5. Scegli Elimina.

Usando il AWS Command Line Interface

Per trovare l'istanza IDs delle istanze DB nel tuo account, chiama il `list-db-instances` comando:

```
aws timestream-influxdb list-db-instances \
--endpoint-url YOUR_ENDPOINT \
--region YOUR_REGION
```

Per eliminare un'istanza DB utilizzando il AWSCLI, chiama il `delete-db-instance` comando con le seguenti opzioni:

```
aws timestream-influxdb list-db-instances \
--identifier YOUR_DB_INSTANCE \
```

Example Esempio

Per Linux, macOS o Unix:

```
aws timestream-influxdb delete-db-instance \
    --identifier mydbinstance
```

Per Windows:

```
aws timestream-influxdb delete-db-instance ^
    --identifier mydbinstance
```

Implementazioni dell'istanza database Multi-AZ

Amazon Timestream for InfluxDB offre elevata disponibilità e supporto di failover per le istanze DB utilizzando implementazioni Multi-AZ con una singola istanza DB in standby. Questo tipo di implementazione è chiamato una implementazione di istanza database Multi-AZ. Amazon Timestream for InfluxDB utilizza la tecnologia di failover di Amazon.

In una distribuzione di istanze DB Multi-AZ, Amazon Timestream effettua automaticamente il provisioning e mantiene una replica sincrona in standby in una zona di disponibilità diversa.

L'istanza database principale viene replicata in modo sincrono tra le zone di disponibilità in una replica in standby per garantire la ridondanza dei dati. L'esecuzione di un'istanza DB con elevata disponibilità può migliorare la disponibilità in caso di guasto dell'istanza DB e interruzione della zona di disponibilità. Per ulteriori informazioni sulle zone di disponibilità, consulta AWS Regioni e zone di disponibilità.

> **Note**
>
> L'opzione di disponibilità elevata non è una soluzione di dimensionamento per scenari di sola lettura. Non è possibile utilizzare una replica in standby per gestire il traffico di lettura.

Utilizzando la console Amazon Timestream, puoi creare una distribuzione di istanze DB Multi-AZ semplicemente specificando l'opzione Crea un'istanza di standby nella sezione Configurazione di disponibilità e durabilità durante la creazione di un'istanza DB. Puoi anche specificare una distribuzione di istanze DB Multi-AZ con Amazon AWS Command Line Interface TimestreamAPI. Usa il CLI comando `create-db-instance` o o l'operazione. `CreateDBInstance` API

Le istanze database che utilizzano implementazioni Multi-AZ possono avere una latenza di scrittura e di commit maggiore rispetto a un'implementazione Single-AZ. Ciò può accadere a causa della replica sincrona dei dati che si verifica. È possibile che si verifichi una modifica della latenza se la distribuzione fallisce nella replica in standby, sebbene AWS sia progettata con connettività di rete a bassa latenza tra le zone di disponibilità. Per i carichi di lavoro di produzione, consigliamo di utilizzare lo storage IOPS incluso da 12.000 o 16.000 rpm per prestazioni veloci e costanti. IOPS Per altre informazioni sulle classi di istanza database, consulta Classi di istanze database.

Configurazione e gestione di un'implementazione Multi-AZ

Le implementazioni Timestream for InfluxDB Multi-AZ possono avere un solo standby. Quando l'implementazione ha un'istanza DB in standby, viene chiamata una implementazione istanza database Multi-AZ. Un'implementazione di istanze database Multi-AZ ha un'istanza database in standby che fornisce supporto per il failover, ma non serve il traffico di lettura.

> **Important**
>
> L'istanza deve avere almeno due sottoreti associate per eseguire gli aggiornamenti da Single-AZ a Multi-AZ. Una volta creata l'istanza, non è possibile modificarne la modalità di distribuzione da Single-AZ a Multi-AZ.

È possibile utilizzare il AWS Management Console per determinare se l'istanza DB è una distribuzione Single-AZ o Multi-AZ.

Utilizzando il AWS Management Console

1. Accedi AWS Management Console e apri la console Amazon Timestream per InfluxDB.
2. Nel riquadro di navigazione, scegli Database InfluxDB, quindi scegli l'identificatore DB.

Un'implementazione di istanze database Multi-AZ presenta le seguenti caratteristiche:

- Esiste una sola riga per l'istanza database.
- Il valore di Role (Ruolo) è Instance (Istanza) o Primary (Prncipale).
- Il valore di Multi-AZ è Yes (Sì).

Processo di failover per Amazon Timestream

Se un'interruzione pianificata o non pianificata dell'istanza DB deriva da un difetto dell'infrastruttura, Amazon Timestream for InfluxDB passa automaticamente a una replica in standby in un'altra zona di disponibilità se hai attivato Multi-AZ. Il tempo necessario per il completamento del failover varia in base all'attività del database e ad altre condizioni presenti quando l'istanza database principale diventa non disponibile. Il failover richiede in genere da 60 a 120 secondi, tempo che può tuttavia aumentare in caso di transazioni di grandi dimensioni o di un processo di ripristino di lunga durata. Una volta completato il failover, può essere necessario più tempo prima che la console Timestream rifletta la nuova zona di disponibilità.

> **Note**
>
> Amazon Timestream gestisce automaticamente i failover in modo da poter riprendere le operazioni del database il più rapidamente possibile senza interventi amministrativi. L'istanza database principale passa automaticamente alla replica di standby qualora si verifichi una delle condizioni riportate nella seguente tabella.

Motivo del failover	Descrizione
Il sistema operativo alla base dell'istanza del database Timestream viene patchato durante un'operazione offline.	È stato attivato un failover durante la finestra di manutenzione per una patch del sistema operativo o un aggiornamento di sicurezza.
L'host principale dell'istanza Timestream Multi-AZ non è integro.	L'implementazione istanza database Multi-AZ ha rilevato un'istanza database primaria compromessa e ha attivato il failover.
L'host principale dell'istanza Timestream Multi-AZ non è raggiungibile a causa della perdita di connettività di rete.	Il monitoraggio Timestream ha rilevato un errore di raggiungibilità della rete nell'istanza DB principale e ha attivato un failover.
L'istanza Timestream è stata modificata dal cliente.	Una modifica dell'istanza DB di Timesteam for InfluxDB ha innescato un failover. Per ulteriori informazioni, consulta Aggiornamento delle istanze database.
L'istanza primaria di Timestream Multi-AZ è occupata e non risponde.	L'istanza database primaria non risponde. Si consiglia di effettuare le seguenti operazion i: * Esamina l'evento per verificare l'utilizzo eccessivo della memoria o dello CPU spazio di swap. * Valuta il tuo carico di lavoro per determinare se stai utilizzando la classe di istanza DB appropriata. Per ulteriori informazi oni, consulta le classi di istanza database.
Il volume di storage sottostante l'host principal e dell'istanza Timestream Multi-AZ ha subito un errore.	L'implementazione dell'istanza database Multi-AZ ha rilevato un problema di archiviaz ioine nell'istanza DB principale e ha avviato il failover.

Impostazione delle ricerche per i nomi JVM TTL DNS

Il meccanismo di failover modifica automaticamente il record Domain Name System (DNS) dell'istanza DB in modo che punti all'istanza DB in standby. Di conseguenza, sarà necessario

ristabilire le connessioni esistenti alla propria istanza database. In un ambiente Java virtual machine (JVM), a causa del funzionamento del meccanismo di DNS caching Java, potrebbe essere necessario riconfigurare le impostazioni. JVM

Le ricerche dei JVM nomi delle cache. DNS Quando JVM risolve un nome host in un indirizzo IP, memorizza l'indirizzo IP nella cache per un periodo di tempo specificato, noto come (). time-to-liveTTL

Poiché AWS le risorse utilizzano voci di DNS nome che cambiano di tanto in tanto, si consiglia di configurare il file JVM con un TTL valore non superiore a 60 secondi. In questo modo, quando l'indirizzo IP di una risorsa cambia, l'applicazione può ricevere e utilizzare il nuovo indirizzo IP della risorsa richiedendo il. DNS

In alcune configurazioni Java, l'JVMimpostazione predefinita TTL è impostata in modo che non aggiorni mai le DNS voci fino al riavvio di. JVM Pertanto, se l'indirizzo IP di una AWS risorsa cambia mentre l'applicazione è ancora in esecuzione, non può utilizzare quella risorsa finché non si riavvia manualmente JVM e le informazioni IP memorizzate nella cache non vengono aggiornate. In questo caso, è fondamentale impostare i in TTL modo che JVM aggiorni periodicamente le informazioni IP memorizzate nella cache.

Puoi ottenere l'JVMimpostazione predefinita TTL recuperando il valore della proprietà: `networkaddress.cache.ttl`

```
String ttl = java.security.Security.getProperty("networkaddress.cache.ttl");
```

> **Note**
>
> L'impostazione predefinita TTL può variare a seconda della versione JVM e dell'installazione o meno di un gestore della sicurezza. Molti JVMs forniscono un valore predefinito TTL inferiore a 60 secondi. Se utilizzi tale strumento JVM e non utilizzi un gestore della sicurezza, puoi ignorare il resto di questo argomento.
> Per modificare iTTL, impostate il JVM valore della proprietà networkaddress.cache.ttl. Utilizza uno dei seguenti metodi, a seconda delle esigenze:
>
> - Per impostare il valore della proprietà a livello globale per tutte le applicazioni che utilizzano la, impostatela nel file. JVM `networkaddress.cache.ttl` `$JAVA_HOME/jre/lib/security/java.security`
>
>   ```
>   networkaddress.cache.ttl=60
>   ```

- Per impostare la proprietà localmente solo per l'applicazione, imposta `networkaddress.cache.ttl` nel codice di inizializzazione dell'applicazione prima che venga stabilita qualsiasi connessione.

```
java.security.Security.setProperty("networkaddress.cache.ttl" , "60");
```

Configurazione per visualizzare i log di InfluxDB sulle istanze Timestream Influxdb

Per impostazione predefinita, InfluxDB genera log che vanno su stdout. Per ulteriori informazioni, consulta Gestire i log di InfluxDB

Per visualizzare i log di InfluxDB generati dall'istanza creata tramite Timestream InfluxDB, offriamo l'opportunità di fornire log orari. Questi log verranno inseriti in un bucket S3 specificato che devi creare prima di creare l'istanza.

- Prima di creare l'istanza, il bucket Amazon S3 fornito deve inoltre concedere a Timestream-InfluxDB l'autorizzazione a inviare i log a questo bucket fornendo una policy sui bucket con Timestream InfluxDB Service Principal come segue (sostituisci *{BUCKET_NAME}* con il nome effettivo del tuo bucket Amazon S3:

```
{
    "Version": "2012-10-17",
    "Id": "PolicyForInfluxLogs",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "timestream-influxdb.amazonaws.com"
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::{BUCKET_NAME}/InfluxLogs/*"
        }
    ]
}
```

- Il bucket fornito deve trovarsi nello stesso account e nella stessa regione dell'istanza Timestream InfluxDB creata

Ecco il comando che puoi chiamare per creare un'istanza per ricevere i log di influsso:

```
aws timestream-influxdb create-db-instance \
    --name myinfluxDbinstance \
    --allocated-storage 400 \
    --db-instance-type db.influx.4xlarge \
    --vpc-subnet-ids subnetid1 subnetid2
    --vpc-security-group-ids mysecuritygroup \
    --username masterawsuser \
    --password \
    --db-storage-type InfluxIOIncludedT2
```

Ecco il formato di questo parametro.

```
-- log-delivery-configuration
{
    "S3Configuration": {
      "BucketName": "string",
      "Enabled": true|false
    }
}
```

- Questo campo non è obbligatorio e la registrazione non è abilitata per impostazione predefinita.
- Non impostare questo campo equivale a non avere i log abilitati.
- I log verranno inviati al bucket specificato con il prefisso di. `InfluxLogs/`
- Dopo aver creato l'istanza, puoi modificare la configurazione di consegna dei log con il comando. `update-db-instance` API

InfluxDB offre diversi tipi di log. Questi possono essere configurati impostando i parametri InfluxDB. Utilizza flux-log-enabled i parametri a livello di registro per configurare il tipo di log emessi dall'istanza. Per ulteriori informazioni, consulta Parametri e valori dei parametri supportati.

Aggiunta di tag ed etichette alle risorse

Puoi etichettare le risorse Amazon Timestream per InfluxDB utilizzando i tag. I tag consentono di classificare le risorse in diversi modi, ad esempio per scopo, proprietario, ambiente o altri criteri. I tag consentono di eseguire le seguenti operazioni:

- identificare rapidamente una risorsa in base ai tag a questa assegnati.
- Visualizza le AWS fatture suddivise per tag.

Il tagging è supportato da AWS servizi come Amazon Elastic Compute Cloud (AmazonEC2), Amazon Simple Storage Service (Amazon S3), Timestream for InfluxDB e molti altri. Un tagging efficiente può fornire informazioni dettagliate sui costi abilitando la creazione di report su servizi che recano tag specifici.

Infine, è buona norma seguire strategie di tagging ottimali. Per informazioni, consulta Strategie di tagging di AWS.

Restrizioni di tagging

Ogni tag consiste di una chiave e di un valore, entrambi personalizzabili. Le restrizioni si applicano come segue:

- Ogni istanza di Timestream for InfluxDB DB può avere un solo tag con la stessa chiave. Se si tenta di aggiungere un tag esistente, il valore del tag esistente viene aggiornato al nuovo valore.
- Un valore funge da descrittore all'interno di una categoria di tag. In Timestream for InfluxDB il valore non può essere vuoto o nullo.
- I valori e le chiavi dei tag rispettano la distinzione tra maiuscole e minuscole.
- La lunghezza massima della chiave è di 128 caratteri Unicode.
- La lunghezza massima del valore è di 256 caratteri Unicode.
- i caratteri consentiti sono lettere, spazi vuoti, numeri e i seguenti caratteri speciali: `+ - = . _ : /`
- Il numero massimo di tag per risorsa è 50.
- AWS-ai nomi e ai valori dei tag assegnati viene assegnato automaticamente il `aws:` prefisso, che non è possibile assegnare. AWS-i nomi dei tag assegnati non vengono conteggiati ai fini del limite di 50 tag. I nomi dei tag assegnati dall'utente hanno il prefisso `user:` nel report di allocazione dei costi;
- Non puoi retrodatare l'applicazione di un tag.

Le migliori pratiche di sicurezza per Timestream for InfluxDB

Ottimizza le scritture su InfluxDB

Come qualsiasi altro database di serie temporali, InfluxDB è progettato per essere in grado di importare ed elaborare i dati in tempo reale. Per mantenere le migliori prestazioni del sistema, consigliamo le seguenti ottimizzazioni durante la scrittura di dati su InfluxDB:

- Scritture in batch: quando scrivi dati su InfluxDB, scrivi i dati in batch per ridurre al minimo il sovraccarico di rete relativo a ogni richiesta di scrittura. La dimensione ottimale del batch è di 5000 righe di protocollo di linea per richiesta di scrittura. Per scrivere più righe in una richiesta, ogni riga del protocollo di riga deve essere delimitata da una nuova riga (\n).
- Ordina i tag per chiave: prima di scrivere punti dati su InfluxDB, ordina i tag per chiave in ordine lessicografico.

```
measurement,tagC=therefore,tagE=am,tagA=i,tagD=i,tagB=think fieldKey=fieldValue
 1562020262

# Optimized line protocol example with tags sorted by key
measurement,tagA=i,tagB=think,tagC=therefore,tagD=i,tagE=am fieldKey=fieldValue
 1562020262
```

- Usa la massima precisione temporale possibile: — InfluxDB scrive i dati con una precisione di nanosecondi, tuttavia se i dati non vengono raccolti in nanosecondi, non è necessario scrivere con tale precisione. Per prestazioni migliori, usa la massima precisione possibile per i timestamp. Puoi specificare la precisione di scrittura quando:
  - Quando si utilizza l'SDKattributo WritePrecision when setting time del punto è possibile specificare l'attributo time. Per ulteriori informazioni sulle librerie client di InfluxDB, consulta la documentazione di InfluxDB.
  - Quando si utilizza Telegraf, si configura la precisione temporale nella configurazione dell'agente Telegraf. La precisione è specificata come intervallo con un numero intero + (ad esempio 0s,10ms,2us,4s). Le unità di tempo valide sono «ns», «us», «ms» e «s».

```
[agent]
 interval ="10s"
 metric_batch_size="5000"
 precision = "0s"
```

- Usa la compressione gzip: — Usa la compressione gzip per velocizzare le scritture su InfluxDB e ridurre la larghezza di banda della rete. I benchmark hanno mostrato un miglioramento della velocità fino a 5 volte quando i dati vengono compressi.
  - Quando usi Telegraf, nella configurazione del plugin di output InfluxDB_v2 nel tuo telegraf.conf, imposta l'opzione content_encoding su gzip:

    ```
    [[outputs.influxdb_v2]]
      urls = ["http://localhost:8086"]
      # ...
      content_encoding = "gzip"
    ```

  - Quando si utilizzano librerie client, ogni libreria client InfluxDB fornisce opzioni per comprimere le richieste di scrittura o applica la compressione per impostazione predefinita. Il metodo per abilitare la compressione è diverso per ogni libreria. Per istruzioni specifiche, consulta la documentazione di InfluxDB
  - Quando usi l'API`/api/v2/write`endpoint InfluxDB per scrivere dati, comprimi i dati con gzip e imposta l'intestazione Content-Encoding su gzip.

Progettato per le prestazioni

Progetta il tuo schema per query più semplici e più performanti. Le seguenti linee guida garantiranno che lo schema sia facile da interrogare e massimizzi le prestazioni delle query:

- Progettazione per interrogare: scegli misure, chiavi di tag e chiavi di campo facili da interrogare. Per raggiungere questo obiettivo, segui questi principi:
  - Usa misurazioni con un nome semplice e descrivi accuratamente lo schema.
  - Evita di usare lo stesso nome per una chiave di tag e una chiave di campo all'interno dello stesso schema.
  - Evita di utilizzare parole chiave Flux riservate e caratteri speciali nelle chiavi di tag e campo.
  - I tag memorizzano i metadati che descrivono i campi e sono comuni a molti punti dati.
  - I campi memorizzano dati univoci o altamente variabili, in genere punti dati numerici.
  - Le misurazioni e le chiavi non devono contenere dati, ma devono essere utilizzate per aggregare o descrivere dati. I dati verranno archiviati nei valori dei tag e dei campi.
- Tieni sotto controllo la cardinalità delle serie temporali La cardinalità delle serie elevate è una delle cause principali della riduzione delle prestazioni di scrittura e lettura in InfluxDB. Nel contesto di InfluxDB, l'alta cardinalità si riferisce alla presenza di un numero molto elevato di valori di tag unici.

I valori dei tag sono indicizzati in InfluxDB, il che significa che un numero molto elevato di valori univoci genererà un indice più ampio che può rallentare l'inserimento dei dati e le prestazioni delle query.

Per comprendere e risolvere meglio i potenziali problemi relativi all'elevata cardinalità, puoi seguire questi passaggi:

- Comprendi le cause dell'alta cardinalità
- Misura la cardinalità dei tuoi bucket
- Agisci per risolvere la cardinalità elevata

- Cause dell'elevata cardinalità delle serie InfluxDB indicizza i dati in base a misurazioni e tag per velocizzare la lettura dei dati. Ogni set di elementi di dati indicizzati forma una chiave di serie. I tag contenenti informazioni altamente variabili come stringhe unicheIDs, hash e casuali generano un gran numero di serie, noto anche come cardinalità di serie elevata. La cardinalità di serie elevata è il fattore principale dell'elevato utilizzo della memoria in InfluxDB.
- Misurazione della cardinalità delle serie Se riscontri rallentamenti delle prestazioni o riscontri un utilizzo sempre maggiore della memoria nell'istanza di Timestream for InfluxDB, ti consigliamo di misurare la cardinalità delle serie dei tuoi bucket.

  InfluxDB fornisce funzioni che consentono di misurare la cardinalità delle serie sia in Flux che in InfluxQL.

  - In Flux usa la funzione `influxdb.cardinality()`
  - In FluxQL usa il comando `SHOW SERIES CARDINALITY`

  In entrambi i casi il motore restituirà il numero di chiavi di serie univoche nei dati. Tieni presente che non è consigliabile avere più di 10 milioni di chiavi di serie su nessuna delle tue istanze di Timestream for InfluxDB.
- Cause della cardinalità di serie elevata Se riscontri che uno dei tuoi bucket ha una cardinalità elevata, puoi eseguire alcuni passaggi correttivi per risolvere il problema:
  - Controlla i tag: assicurati che i tuoi carichi di lavoro non generino casi in cui i tag abbiano valori univoci per la maggior parte delle voci. Ciò potrebbe accadere nei casi in cui il numero di valori di tag univoci aumenta sempre nel tempo o se nel database vengono scritti messaggi di tipo registro in cui ogni messaggio avrebbe una combinazione unica di timestamp, tag ecc. Puoi usare il seguente codice Flux per aiutarti a capire quali tag stanno contribuendo maggiormente ai tuoi problemi di elevata cardinalità:

```
// Count unique values for each tag in a bucketimport "influxdata/influxdb/schema"
```

```
cardinalityByTag = (bucket) => schema.tagKeys(bucket: bucket)
    |> map(
        fn: (r) => ({
            tag: r._value,
            _value: if contains(set: ["_stop", "_start"], value: r._value) then
                0
            else
                (schema.tagValues(bucket: bucket, tag: r._value)
                    |> count()
                    |> findRecord(fn: (key) => true, idx: 0))._value,
        }),
    )
    |> group(columns: ["tag"])
    |> sum()

cardinalityByTag(bucket: "example-bucket")
```

Se riscontri una cardinalità molto elevata, la query precedente potrebbe scadere. Se si verifica un timeout, esegui le query seguenti, una alla volta.

Genera un elenco di tag:

```
// Generate a list of tagsimport "influxdata/influxdb/schema"

schema.tagKeys(bucket: "example-bucket")
```

Conta valori di tag univoci per ogni tag:

```
// Run the following for each tag to count the number of unique tag valuesimport
 "influxdata/influxdb/schema"

tag = "example-tag-key"

schema.tagValues(bucket: "my-bucket", tag: tag)
    |> count()
```

Ti consigliamo di eseguirli in momenti diversi per identificare quale tag sta crescendo più rapidamente.

- Migliora il tuo schema: segui i consigli di modellazione discussi nel nostroLe migliori pratiche di sicurezza per Timestream for InfluxDB.

- Rimuovi o aggrega i dati più vecchi per ridurre la cardinalità: valuta se i tuoi casi d'uso richiedono o meno tutti i dati che causano problemi di elevata cardinalità. Se questi dati non sono più necessari o non vengono più utilizzati frequentemente, puoi aggregarli, eliminarli o esportarli su un altro motore come Timestream for Live Analytics per l'archiviazione e l'analisi a lungo termine.

Risoluzione dei problemi

Avviso relativo alla versione «dev» non riconosciuta

L'avviso 'WARN: Impossibile analizzare la versione «dev» riportata dal server, supponendo che il backup/ripristino APIs più recente sia supportato' potrebbe essere visualizzato durante la migrazione. Questo avviso può essere ignorato.

Migrazione non riuscita durante la fase di ripristino

In caso di mancata migrazione durante la fase di ripristino, gli utenti possono utilizzare il `--retry-restore-dir` flag per ritentare il ripristino. Usa il `--retry-restore-dir` flag con il percorso di una directory di cui è stato precedentemente eseguito il backup per saltare la fase di backup e riprovare la fase di ripristino. La directory di backup creata utilizzata per una migrazione verrà indicata se una migrazione fallisce durante il ripristino.

Le possibili ragioni di un errore di ripristino includono:

- Token di destinazione InfluxDB non valido: un bucket esistente nell'istanza di destinazione con lo stesso nome dell'istanza di origine. Per le migrazioni di singoli bucket, utilizza l'`--dest-bucket`opzione per impostare un nome univoco per il bucket migrato
- Errore di connettività, con gli host di origine o di destinazione o con un bucket S3 opzionale.

Linee guida operative di base di Amazon Timestream per InfluxDB

Di seguito sono riportate le linee guida operative di base che tutti dovrebbero seguire quando lavorano con Amazon Timestream per InfluxDB. Tieni presente che il Service Level Agreement di Amazon Timestream for InfluxDB richiede di seguire queste linee guida:

- Utilizza le metriche per monitorare l'utilizzo della memoria e dello storage. CPU Puoi configurare Amazon in modo che CloudWatch ti avvisi quando i modelli di utilizzo cambiano o quando ti avvicini

alla capacità della tua implementazione. In questo modo, è possibile mantenere le prestazioni e la disponibilità del sistema.

- Incrementa la capacità dell'istanza database quando stai per raggiungere i limiti della capacità di storage. Avrai bisogno di memoria e storage aggiuntivi per soddisfare aumenti imprevisti della domanda delle tue applicazioni. Tieni presente che al momento dovrai creare una nuova istanza e migrare i tuoi dati per raggiungere questo obiettivo.
- Se il carico di lavoro del database richiede più operazioni di I/O rispetto a quelle che hai assegnato, il ripristino dopo un failover o un errore del database sarà lento. Per aumentare la capacità I/O di un'istanza di database, effettua una o più delle seguenti operazioni:
  - Esegui la migrazione a un'altra istanza DB con una maggiore capacità di I/O.
  - Se stai già utilizzando lo storage di archiviazione IOPS incluso Influx, fornisci un tipo di storage con una maggiore capacità di storage inclusa. IOPS
- Se l'applicazione client memorizza nella cache i dati del Domain Name Service (DNS) delle istanze DB, imposta un valore time-to-live (TTL) inferiore a 30 secondi. L'indirizzo IP sottostante di un'istanza DB può cambiare dopo un failover. La memorizzazione nella cache DNS dei dati per un periodo prolungato può quindi causare errori di connessione. L'applicazione potrebbe tentare di connettersi a un indirizzo IP non più in uso.

Consigli sulle istanze DB RAM

Una best practice in termini di prestazioni di Amazon Timestream for InfluxDB consiste nell'allocare una RAM quantità sufficiente in modo che il set di lavoro risieda quasi completamente in memoria. Il working set è formato dai dati e dagli indici che vengono utilizzati spesso nell'istanza. Più utilizzi l'istanza DB, più il working set crescerà.

Sicurezza in Timestream per InfluxDB

La sicurezza del cloud è la massima priorità. AWS In qualità di AWS cliente, puoi beneficiare di un data center e di un'architettura di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra AWS te e te. Il modello di responsabilità condivisa descrive questo come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud: AWS è responsabile della protezione dell'infrastruttura che gestisce AWS i servizi nel AWS cloud. AWS ti fornisce anche servizi che puoi utilizzare in modo sicuro. L'efficacia

della nostra sicurezza è regolarmente testata e verificata da revisori di terze parti come parte dei programmi di conformitàAWS. Per ulteriori informazioni sui programmi di conformità che si applicano a Timestream for InfluxDB, consulta AWS Services in Scope by Compliance Program.

- Sicurezza nel cloud: la tua responsabilità è determinata dal AWS servizio che utilizzi. L'utente è anche responsabile per altri fattori, tra cui la riservatezza dei dati, i requisiti dell'azienda, nonché le leggi e le normative applicabili.

Questa documentazione ti aiuterà a capire come applicare il modello di responsabilità condivisa quando usi Timestream for InfluxDB. I seguenti argomenti mostrano come configurare Timestream per InfluxDB per soddisfare i tuoi obiettivi di sicurezza e conformità. Imparerai anche come utilizzare altri AWS servizi che possono aiutarti a monitorare e proteggere le tue risorse Timestream for InfluxDB.

Argomenti

Panoramica

Questa documentazione ti aiuta a capire come applicare il modello di responsabilità condivisa quando usi Amazon Timestream per InfluxDB. I seguenti argomenti mostrano come configurare Amazon Timestream per InfluxDB per soddisfare i tuoi obiettivi di sicurezza e conformità. Scopri

anche come utilizzare altri AWS servizi che ti aiutano a monitorare e proteggere le tue risorse Amazon Timestream for InfluxDB.

Puoi gestire l'accesso alle tue risorse Amazon Timestream for InfluxDB e ai tuoi database su un'istanza DB. Il metodo utilizzato per gestire l'accesso dipende dal tipo di attività che l'utente deve eseguire con Amazon Timestream for InfluxDB:

- Esegui la tua istanza DB in un Virtual Private Cloud (VPC) basato sul VPC servizio Amazon per il controllo degli accessi alla rete.
- Utilizza le policy di AWS Identity and Access Management (IAM) per assegnare autorizzazioni che determinano chi è autorizzato a gestire le risorse Amazon Timestream for InfluxDB. Ad esempio, puoi utilizzarle per determinare chi è autorizzato IAM a creare, descrivere, modificare ed eliminare istanze DB, taggare risorse o modificare gruppi di sicurezza.
- Utilizza i gruppi di sicurezza per controllare quali indirizzi IP o EC2 istanze Amazon possono connettersi ai tuoi database su un'istanza DB. La prima volta che crei un'istanza DB, è accessibile solo tramite le regole specificate da un gruppo di sicurezza associato.
- Usa connessioni Secure Socket Layer (SSL) o Transport Layer Security (TLS) con le tue istanze DB.
- Utilizza le funzionalità di sicurezza del tuo motore InfluxDB per controllare chi può accedere ai database su un'istanza DB. Queste funzionalità funzionano come se il database si trovasse sulla rete locale. Per ulteriori informazioni, consulta Sicurezza in Timestream per InfluxDB.

> **Note**
>
> Devi configurare la sicurezza solo per i tuoi casi d'uso. Non è necessario configurare l'accesso di sicurezza per i processi gestiti da Amazon Timestream for InfluxDB. Questo include la creazione di backup, la replica di dati tra una istanza database primaria e una replica di lettura, nonché altri processi.

Argomenti

Sicurezza generale

Argomenti

- Autorizzazioni
- Accesso alla rete
- Dipendenze
- Bucket S3

Autorizzazioni

Agli utenti di InfluxDB dovrebbero essere concesse le autorizzazioni con privilegi minimi. Durante la migrazione devono essere utilizzati solo i token concessi a utenti specifici, anziché i token operatore.

Timestream for InfluxDB utilizza IAM le autorizzazioni per controllare le autorizzazioni degli utenti. Consigliamo di concedere agli utenti l'accesso alle azioni e alle risorse specifiche di cui hanno bisogno. Per ulteriori informazioni, consulta Concedere l'accesso con privilegi minimi.

Accesso alla rete

Lo script di migrazione Influx può funzionare localmente, migrando i dati tra due istanze InfluxDB sullo stesso sistema, ma si presume che il caso d'uso principale per le migrazioni sia la migrazione dei dati attraverso la rete, una rete locale o pubblica. A ciò si aggiungono considerazioni sulla sicurezza. Per impostazione predefinita, lo script di migrazione Influx verificherà TLS i certificati per le istanze TLS abilitate: consigliamo agli utenti di abilitarlo TLS nelle proprie istanze InfluxDB e di non utilizzare l'opzione per lo script. `--skip-verify`

Ti consigliamo di utilizzare una lista di autorizzazioni per limitare il traffico di rete proveniente da fonti attese. Puoi farlo limitando il traffico di rete alle istanze InfluxDB solo da istanze note. IPs

Dipendenze

È necessario utilizzare le ultime versioni principali di tutte le dipendenze, tra cui Influx, InfluxDBCLI, Python, il modulo Requests e dipendenze opzionali come e. `mountpoint-s3` `rclone`

Bucket S3

Se i bucket S3 vengono utilizzati come storage temporaneo per la migrazione, consigliamo di abilitareTLS, controllare le versioni e disabilitare l'accesso pubblico.

Utilizzo dei bucket S3 per la migrazione

1. Apri AWS Management Console, accedi ad Amazon Simple Storage Service e scegli Bucket.

2. Scegli il bucket che desideri utilizzare.
3. Scegli la scheda Autorizzazioni.
4. In Blocca accesso pubblico (impostazioni bucket), seleziona Modifica.
5. Seleziona Blocca tutti gli accessi pubblici.
6. Scegli Save changes (Salva modifiche).
7. In Bucket Policy (Policy del bucket) scegliere Edit (Modifica).
8. Inserisci quanto segue, sostituendolo **<example-bucket>**con il nome del tuo bucket, per imporre l'uso della TLS versione 1.2 o successiva per le connessioni:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EnforceTLSv12orHigher",
            "Principal": {
                "AWS": "*"
            },
            "Action": [
                "s3:*"
            ],
            "Effect": "Deny",
            "Resource": [
                "arn:aws:s3:::<example bucket>/*",
                "arn:aws:s3:::<example bucket>"
            ],
            "Condition": {
                "NumericLessThan": {
                    "s3:TlsVersion": 1.2
                }
            }
        }
    ]
}
```

9. Scegli Save changes (Salva modifiche).
10. Scegliere la scheda Properties (Proprietà).
11. In Bucket Versioning (Funzione Controllo delle versioni del bucket) scegliere Edit (Modifica).
12. Seleziona Abilita.
13. Scegli Save changes (Salva modifiche).

Per informazioni sulle migliori pratiche di sicurezza per i bucket Amazon S3, consulta Best practice di sicurezza per Amazon Simple Storage Service.

Autenticazione del database con Amazon Timestream per InfluxDB

Amazon Timestream for InfluxDB supporta due modi per autenticare gli utenti del database.

L'autenticazione del database con password e token di accesso utilizza diversi metodi di autenticazione nel database. Pertanto, un utente specifico può accedere a un database utilizzando un solo metodo di autenticazione. In entrambi i casi InfluxDB esegue tutta l'amministrazione degli account utente e dei token. API

Autenticazione password

Durante il processo di creazione dell'istanza DB InfluxDB, hai creato un'organizzazione, un utente e una password. L'utente dispone delle autorizzazioni per gestire tutto nell'istanza database di Timestream for InfluxDB. Con questa combinazione di nome utente e password potrai accedere alla LogIn tua istanza utilizzando InfluxUI e anche utilizzare Influx per generare un token operatore. CLI

È necessario un token operatore per creare utenti, eliminare bucket, organizzazioni ecc. Per ulteriori informazioni, consulta Opzioni di autenticazione del database.

APIgettoni

APII token InfluxDB garantiscono un'interazione sicura tra InfluxDB e strumenti esterni come client o applicazioni. Un API token appartiene a un utente specifico e identifica le autorizzazioni InfluxDB all'interno dell'organizzazione dell'utente.

Esistono tre tipi di token in InfluxDBAPI:

- Operator Token: concede l'accesso completo in lettura e scrittura a tutte le organizzazioni e a tutte le risorse organizzative in InfluxDB 2.x. OSS Alcune operazioni, ad esempio il recupero della configurazione del server, richiedono le autorizzazioni dell'operatore. Per creare manualmente un token operatore con l'interfaccia utente di InfluxDB o Influx CLI dopo il completamento del processo di configurazione, è necessario utilizzare un token operatore esistente o il nome utente e la password. `api/v2` API Per creare un nuovo token operatore senza utilizzarne uno esistente, consulta l'autenticazione di ripristino influxd. CLI

> **Important**
>
> Poiché i token operatore hanno accesso completo in lettura e scrittura a tutte le organizzazioni del database, consigliamo di creare un token All-Access per ogni organizzazione e di utilizzarlo per gestire InfluxDB. Questo aiuta a prevenire interazioni accidentali tra le organizzazioni.

- APIToken All-Access: garantisce l'accesso completo in lettura e scrittura a tutte le risorse di un'organizzazione.
- Token di lettura/scrittura: concede l'accesso in lettura, in scrittura o entrambi a bucket specifici di un'organizzazione.

Tutti i InfluxDb token sono token di lunga durata senza una data di scadenza prestabilita, quindi non è consigliabile utilizzare l'operatore o tutti i token di accesso per inviare dati di monitoraggio dai clienti o dagli agenti di Telegraf né incorporarli nelle applicazioni di dashboard. Per queste applicazioni, crea token di lettura/scrittura con solo le autorizzazioni necessarie per portare a termine il lavoro. Per ulteriori informazioni su come creare un token InfluxDB, consulta Creare un token.

Segreti

I token dell'operatore InfluxDB vengono generati durante la configurazione dell'istanza; altri tipi di token, come i token di accesso completo e i token di lettura/scrittura, possono essere creati utilizzando la funzione di rotazione multiutente Influx, Influx v2 o Timestream for CLI InfluxDB. API Vedi Gestire i token per sapere come generare, visualizzare, assegnare ed eliminare i token. API

Ti consigliamo di ruotare Timestream per i token InfluxDB utilizzando e archiviando spesso i token tramite variabili di ambiente. AWS Secrets Manager Vedi Usa i token per l'utilizzo dei token nelle variabili di ambiente e per come ruotare Timestream Rotazione del segreto per utenti e token di InfluxDB.

Consulta anche:

- Sicurezza dell'infrastruttura in Amazon Timestream per InfluxDB
- Le migliori pratiche di sicurezza per Timestream for InfluxDB

In che modo Amazon Timestream per InfluxDB utilizza i segreti

Timestream for InfluxDB supporta l'autenticazione di nome utente e password tramite l'interfaccia utente e le credenziali token per le connessioni a client e applicazioni con privilegi minimi. Gli utenti di Timestream for InfluxDB dispongono di `allAccess` autorizzazioni all'interno della propria organizzazione mentre i token possono avere qualsiasi set di autorizzazioni. Seguendo le migliori pratiche per una gestione sicura dei API token, è necessario creare utenti per gestire i token per un accesso mirato all'interno di un'organizzazione. Ulteriori informazioni sulle migliori pratiche di amministrazione con Timestream per InfluxDB sono disponibili nella documentazione di Influxdata.

AWS Secrets Manager è un servizio di archiviazione segreto che puoi utilizzare per proteggere le credenziali, API le chiavi e altre informazioni segrete del database. Quindi, nel codice, puoi sostituire le credenziali hardcoded con una API chiamata a Secrets Manager. Questo aiuta a garantire che il segreto non possa essere compromesso da qualcuno che esamina il codice, perché il segreto non è presente. Per una panoramica di Secrets Manager, vedi Cos'è AWS Secrets Manager.

Quando crei un'istanza di database, Timestream for InfluxDB crea automaticamente un segreto di amministrazione da utilizzare con la funzione di rotazione multiutente. AWS Lambda Per ruotare Timestream for InfluxDB utenti e token, devi creare manualmente un nuovo segreto per ogni utente o token che desideri ruotare. Ogni segreto può essere configurato per ruotare secondo una pianificazione con l'uso di una funzione Lambda. Il processo per impostare un nuovo segreto rotante consiste nel caricare il codice della funzione Lambda, configurare il ruolo Lambda, definire il nuovo segreto e configurare la pianificazione di rotazione segreta.

Cosa c'è nel segreto

Quando memorizzate le credenziali utente di Timestream for InfluxDB nel segreto, utilizzate il seguente formato.

Utente singolo:

```
{
  "engine": "<required: must be set to 'timestream-influxdb'>",
  "username": "<required: username>",
  "password": "<required: password>",
  "dbIdentifier": "<required: DB identifier>"
}
```

Quando crei un'istanza Timestream for InfluxDB, un segreto di amministrazione viene automaticamente archiviato in Secrets Manager con le credenziali da utilizzare con la funzione

Lambda multiutente. Imposta il `adminSecretArn` valore sul `Authentication Properties Secret Manager ARN` valore trovato nella pagina di riepilogo dell'istanza DB o sul segreto dell'amministratore. ARN Per creare un nuovo segreto di amministrazione è necessario disporre già delle credenziali associate e le credenziali devono avere i privilegi di amministratore.

Quando memorizzi le credenziali del token Timestream for InfluxDB nel segreto, usa il seguente formato.

Multiutente:

```
{
  "engine": "<required: must be set to 'timestream-influxdb'>",
  "org": "<required: organization to associate token with>",
  "adminSecretArn": "<required: ARN of the admin secret>",
  "type": "<required: allAccess or operator or custom>",
  "dbIdentifier": "<required: DB identifier>",
  "token": "<required unless generating a new token: token being rotated>",
  "writeBuckets": "<optional: list of bucketIDs for custom type token, must be input
 within plaintext panel, for example ['id1','id2']>",
  "readBuckets": "<optional: list of bucketIDs for custom type token, must be input
 within plaintext panel, for example ['id1','id2']>",
  "permissions": "<optional: list of permissions for custom type token, must be input
 within plaintext panel, for example ['write-tasks','read-tasks']>"
}
```

Quando memorizzi le credenziali di amministratore di Timestream for InfluxDB in modo segreto, utilizza il seguente formato:

Segreto amministrativo:

```
{
  "engine": "<required: must be set to 'timestream-influxdb'>",
  "username": "<required: username>",
  "password": "<required: password>",
  "dbIdentifier": "<required: DB identifier>",
  "organization": "<optional: initial organization>",
  "bucket": "<optional: initial bucket>"
}
```

Per attivare la rotazione automatica del segreto, il segreto deve trovarsi nella JSON struttura corretta. Scopri come ruotare Timestream Rotazione del segreto for InfluxDB secret.

Modifica del segreto

Le credenziali generate durante il processo di creazione dell'istanza Timestream for InfluxDB vengono archiviate in un segreto di Secrets Manager nel tuo account. L'oggetto di GetDbInstancerisposta contiene un oggetto `influxAuthParametersSecretArn` che contiene Amazon Resource Name (ARN) per tale segreto. Il segreto verrà compilato solo dopo che l'istanza di Timestream for InfluxDB sarà disponibile. Questa è una READONLY copia poiché qualsiasi updates/modifications/deletions di questo segreto non ha alcun impatto sull'istanza DB creata. Se elimini questo segreto, la APIrisposta farà comunque riferimento al segreto eliminatoARN.

Per creare un nuovo token nell'istanza Timestream for InfluxDB anziché archiviare le credenziali del token esistenti, puoi creare token non operatori lasciando il `token` valore vuoto nel campo segreto e utilizzando la funzione di rotazione multiutente con la variabile di ambiente Lambda impostata su. `AUTHENTICATION_CREATION_ENABLED` `true` Se crei un nuovo token, le autorizzazioni definite nel segreto vengono assegnate al token e non possono essere modificate dopo la prima rotazione riuscita. Per ulteriori informazioni sulla rotazione dei segreti, vedere Rotating AWS Secrets Manager Secrets.

Se un segreto viene eliminato, l'utente o il token associato nell'istanza Timestream for InfluxDB non verrà eliminato.

Rotazione del segreto

Si utilizzano le funzioni Lambda di rotazione Timestream for InfluxDB a rotazione singola e multiutente per ruotare le credenziali utente e token di Timestream for InfluxDB. Usa la funzione Lambda a utente singolo per ruotare le credenziali utente per la tua istanza Timestream for InfluxDB e usa la funzione Lambda multiutente per ruotare le credenziali del token per la tua istanza Timestream for InfluxDB.

La rotazione di utenti e token con le funzioni Lambda a utente singolo e multiutente è opzionale. Le credenziali di Timestream for InfluxDB non scadono mai e tutte le credenziali esposte rappresentano un rischio di azioni dannose contro l'istanza DB. Il vantaggio della rotazione delle credenziali di Timestream for InfluxDB con Secrets Manager è un ulteriore livello di sicurezza che limita il vettore di attacco delle credenziali esposte alla finestra temporale fino al ciclo di rotazione successivo. Se non è in atto alcun meccanismo di rotazione per l'istanza DB, tutte le credenziali esposte saranno valide fino a quando non verranno eliminate manualmente.

È possibile configurare Secrets Manager in modo che ruoti automaticamente i segreti in base a una pianificazione specificata. In questo modo puoi sostituire i segreti a lungo termine con altri a breve

termine, contribuendo a ridurre notevolmente il rischio di compromissione. Per ulteriori informazioni sulla rotazione dei segreti con Secrets Manager, consulta Rotate AWS Secrets Manager Secrets.

Rotazione degli utenti

Quando ruotate gli utenti con la funzione Lambda per utente singolo, all'utente verrà assegnata una nuova password casuale dopo ogni rotazione. Per ulteriori informazioni su come abilitare la rotazione automatica, vedere Configurare la rotazione automatica per AWS i segreti di Secrets Manager non appartenenti al database.

Segreti di amministrazione a rotazione

Per ruotare un segreto di amministrazione si utilizza la funzione di rotazione per utente singolo. È necessario aggiungere `dbIdentifier` i valori `engine` and al segreto poiché tali valori non vengono compilati automaticamente durante l'inizializzazione del DB. Vedi Cosa c'è nel segreto il modello segreto completo.

Per individuare un segreto di amministrazione per un'istanza di Timestream for InfluxDB, utilizza il segreto di amministrazione della pagina di riepilogo dell'istanza ARN di Timestream for InfluxDB. Si consiglia di ruotare tutti i segreti di amministrazione di Timestream for InfluxDB poiché gli utenti amministratori dispongono di autorizzazioni elevate per l'istanza Timestream for InfluxDB.

Funzione di rotazione Lambda

Puoi ruotare un utente Timestream for InfluxDB con la funzione di rotazione per utente singolo utilizzando con un nuovo segreto e aggiungendo i campi richiesti per il tuo utente Timestream for InfluxDB. Cosa c'è nel segreto Per ulteriori informazioni sulle funzioni Lambda a rotazione segreta, vedere Rotazione per funzione Lambda.

La funzione di rotazione per utente singolo si autentica con l'istanza DB Timestream for InfluxDB utilizzando le credenziali definite nel segreto, quindi genera una nuova password casuale e imposta la nuova password per l'utente. Per ulteriori informazioni sulle funzioni Lambda a rotazione segreta, vedere Rotazione per funzione Lambda.

Autorizzazioni del ruolo di esecuzione della funzione Lambda

Utilizza la seguente IAM politica come ruolo per la funzione Lambda per utente singolo. La policy fornisce alla funzione Lambda le autorizzazioni necessarie per eseguire una rotazione segreta per Timestream per gli utenti di InfluxDB.

Sostituisci tutti gli elementi elencati di seguito nella IAM politica con i valori del tuo account: AWS

- {rotating_secret_arn} — I dettagli segreti del ARN segreto che viene ruotato si trovano nei dettagli segreti di Secrets Manager.
- {db_instance_arn} — L'istanza Timestream for InfluxDB è disponibile nella pagina di riepilogo dell'istanza Timestream for InfluxDB. ARN

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage"
            ],
     "Resource": "{rotating_secret_arn}"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword"
            ],
            "Resource": "*"
        },
        {
            "Action": [
                "timestream-influxdb:GetDbInstance"
            ],
     "Resource": "{db_instance_arn}",
            "Effect": "Allow"
        }
    ]
}
```

Token rotanti

Puoi ruotare un token Timestream for InfluxDB con la funzione di rotazione multiutente utilizzando with a new secret e aggiungendo i campi richiesti per il Cosa c'è nel segreto token Timestream for InfluxDB. Per ulteriori informazioni sulle funzioni Lambda a rotazione segreta, vedere Rotazione per funzione Lambda.

È possibile ruotare un token Timestream for InfluxDB utilizzando la funzione Lambda multiutente Timestream for InfluxDB. Imposta la variabile di `AUTHENTICATION_CREATION_ENABLED` ambiente su `true` nella configurazione Lambda per abilitare la creazione di token. Per creare un nuovo token, usa il valore Cosa c'è nel segreto per il tuo segreto. Ometti la coppia `token` chiave-valore nel nuovo segreto e imposta `suallAccess`, oppure definisci `type` i permessi specifici e imposta il tipo su. `custom` La funzione di rotazione creerà un nuovo token durante il primo ciclo di rotazione. Non è possibile modificare i permessi del token modificando il segreto dopo la rotazione e tutte le rotazioni successive utilizzeranno i permessi impostati nell'istanza DB.

Funzione di rotazione Lambda

La funzione di rotazione multiutente ruota le credenziali del token creando un nuovo token identico all'autorizzazione utilizzando le credenziali di amministratore nel segreto di amministrazione. La funzione Lambda convalida il valore del token nel segreto prima di creare il token sostitutivo, memorizzare il nuovo valore del token nel segreto ed eliminare il vecchio token. Se la funzione Lambda sta creando un nuovo token, verificherà innanzitutto che la variabile di `AUTHENTICATION_CREATION_ENABLED` ambiente sia impostata su`true`, che non vi sia alcun valore di token nel segreto e che il tipo di token non sia un operatore di tipo.

Autorizzazioni del ruolo di esecuzione della funzione Lambda

Utilizza la seguente IAM politica come ruolo per la funzione Lambda multiutente. La policy fornisce alla funzione Lambda le autorizzazioni necessarie per eseguire una rotazione segreta per i token Timestream for InfluxDB.

Sostituisci tutti gli elementi elencati di seguito nella politica con i valori del IAM tuo account: AWS

- {rotating_secret_arn} — I dettagli segreti del ARN segreto che viene ruotato si trovano nei dettagli segreti di Secrets Manager.
- {authentication_properties_admin_secret_arn} — Il segreto di amministrazione di Timestream for InfluxDB è disponibile nella pagina di riepilogo dell'istanza di Timestream for InfluxDB. ARN
- {db_instance_arn} — L'istanza Timestream for InfluxDB è disponibile nella pagina ARN di riepilogo dell'istanza Timestream for InfluxDB.

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

```
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage"
            ],
     "Resource": "{rotating_secret_arn}"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
     "Resource": "{authentication_properties_admin_secret_arn}"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetRandomPassword"
            ],
            "Resource": "*"
        },
        {
            "Action": [
                "timestream-influxdb:GetDbInstance"
            ],
     "Resource": "{db_instance_arn}",
            "Effect": "Allow"
        }
    ]
}
```

Protezione dei dati in Timestream per InfluxDB

Il modello di responsabilità AWS condivisa Modello si applica alla protezione dei dati in Amazon Timestream per InfluxDB. Come descritto in questo modello, AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutti i. Cloud AWS L'utente è responsabile del controllo dei contenuti ospitati su questa infrastruttura. L'utente è inoltre responsabile della configurazione della protezione e delle attività di gestione per i Servizi AWS utilizzati. Per ulteriori informazioni sulla privacy dei dati, consulta la sezione Privacy dei dati FAQ. Per informazioni sulla protezione dei dati in Europa, consulta il Modello di responsabilitàAWS condivisa e GDPR il post sul blog sulla AWS sicurezza.

Ai fini della protezione dei dati, ti consigliamo di proteggere Account AWS le credenziali e di configurare i singoli utenti con AWS IAM Identity Center o AWS Identity and Access Management (IAM). In tal modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere i suoi compiti. Ti suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Utilizza l'autenticazione a più fattori (MFA) con ogni account.
- UsaSSL/TLSper comunicare con AWS le risorse. Richiediamo TLS 1.2 e consigliamo TLS 1.3.
- Configurazione API e registrazione delle attività degli utenti con AWS CloudTrail. Per informazioni sull'uso dei CloudTrail percorsi per registrare AWS le attività, consulta Lavorare con i CloudTrail percorsi nella Guida per l'AWS CloudTrail utente.
- Utilizza soluzioni di AWS crittografia, insieme a tutti i controlli di sicurezza predefiniti all'interno Servizi AWS.
- Utilizza i servizi di sicurezza gestiti avanzati, come Amazon Macie, che aiutano a individuare e proteggere i dati sensibili archiviati in Amazon S3.
- Se hai bisogno di FIPS 140-3 moduli crittografici convalidati per accedere AWS tramite un'interfaccia a riga di comando o unAPI, usa un endpoint. FIPS Per ulteriori informazioni sugli FIPS endpoint disponibili, vedere Federal Information Processing Standard () 140-3. FIPS

Ti consigliamo vivamente di non inserire mai informazioni riservate o sensibili, ad esempio gli indirizzi e-mail dei clienti, nei tag o nei campi di testo in formato libero, ad esempio nel campo Nome. Ciò include quando lavori con Timestream for InfluxDB o altro Servizi AWS utilizzando la console,, o. API AWS CLI AWS SDKs I dati inseriti nei tag o nei campi di testo in formato libero utilizzati per i nomi possono essere utilizzati per i la fatturazione o i log di diagnostica. Se fornisci un URL a un server esterno, ti consigliamo vivamente di non includere le informazioni sulle credenziali URL per convalidare la tua richiesta a quel server.

Per informazioni più dettagliate su argomenti di protezione dei dati di Timestream for InfluxDB come Encryption at Rest e Key Management, seleziona uno degli argomenti disponibili di seguito.

Argomenti

- Crittografia dei dati inattivi
- Crittografia in transito

Crittografia dei dati inattivi

La crittografia a riposo di Timestream for InfluxDB offre una maggiore sicurezza crittografando tutti i dati inattivi utilizzando le chiavi di crittografia archiviate in ().AWS Key Management ServiceAWS KMS Questa funzionalità consente di ridurre gli oneri operativi e la complessità associati alla protezione dei dati sensibili. La crittografia dei dati inattivi consente di creare applicazioni ad alto livello di sicurezza che rispettano rigorosi requisiti normativi e di conformità per la crittografia.

- La crittografia è attivata per impostazione predefinita sull'istanza DB di Timestream for InfluxDB e non può essere disattivata. L'algoritmo di crittografia standard del settore AES -256 è l'algoritmo di crittografia predefinito utilizzato.
- AWS KMS viene utilizzato per la crittografia a riposo in Timestream for InfluxDB.
- Non è necessario modificare le applicazioni client dell'istanza DB per utilizzare la crittografia.

Crittografia in transito

Tutti i dati di Timestream for InfluxDB sono crittografati in transito. Per impostazione predefinita, tutte le comunicazioni da e verso Timestream for InfluxDB sono protette utilizzando la crittografia Transport Layer Security (). TLS

Il traffico da e verso Amazon Timestream for InfluxDB è protetto utilizzando le versioni supportate 1.2 o 1.3. TLS

Identity and Access Management per Amazon Timestream per InfluxDB

AWS Identity and Access Management (IAM) è un programma Servizio AWS che aiuta un amministratore a controllare in modo sicuro l'accesso alle risorse. AWS IAMgli amministratori controllano chi può essere autenticato (effettuato l'accesso) e autorizzato (disporre delle autorizzazioni) a utilizzare Timestream per le risorse InfluxDB. IAMè un file Servizio AWS che puoi utilizzare senza costi aggiuntivi.

Argomenti

- Autenticazione con identità
- Gestione dell'accesso con policy
- Come funziona Amazon Timestream per InfluxDB con IAM

- Esempi di policy basate sull'identità per Amazon Timestream for InfluxDB
- Risoluzione dei problemi relativi all'identità e all'accesso di Amazon Timestream per InfluxDB
- Controllo dell'accesso a un'istanza DB in un VPC
- Utilizzo di ruoli collegati ai servizi per Amazon Timestream for InfluxDB
- AWS politiche gestite per Amazon Timestream for InfluxDB
- Connessione a Timestream for InfluxDB tramite un endpoint VPC

Autenticazione con identità

L'autenticazione è il modo in cui accedi AWS utilizzando le tue credenziali di identità. È necessario autenticarsi (accedere a AWS) come Utente root dell'account AWS, come IAM utente o assumendo un ruolo. IAM

È possibile accedere AWS come identità federata utilizzando le credenziali fornite tramite una fonte di identità. AWS IAM Identity Center Gli utenti (IAMIdentity Center), l'autenticazione Single Sign-On della tua azienda e le tue credenziali di Google o Facebook sono esempi di identità federate. Quando accedi come identità federata, l'amministratore aveva precedentemente configurato la federazione delle identità utilizzando i ruoli. IAM Quando si accede AWS utilizzando la federazione, si assume indirettamente un ruolo.

A seconda del tipo di utente, puoi accedere al AWS Management Console o al portale di AWS accesso. Per ulteriori informazioni sull'accesso a AWS, vedi Come accedere al tuo Account AWS nella Guida per l'Accedi ad AWS utente.

Se accedi a AWS livello di codice, AWS fornisce un kit di sviluppo software (SDK) e un'interfaccia a riga di comando () per firmare crittograficamente le tue richieste utilizzando le tue credenziali. CLI Se non utilizzi AWS strumenti, devi firmare tu stesso le richieste. Per ulteriori informazioni sull'utilizzo del metodo consigliato per firmare autonomamente le richieste, consulta AWS Signature Version 4 per API le richieste nella Guida per l'IAMutente.

A prescindere dal metodo di autenticazione utilizzato, potrebbe essere necessario specificare ulteriori informazioni sulla sicurezza. Ad esempio, ti AWS consiglia di utilizzare l'autenticazione a più fattori (MFA) per aumentare la sicurezza del tuo account. Per ulteriori informazioni, consulta Autenticazione a più fattori nella Guida per l'AWS IAM Identity Center utente e Autenticazione a AWS più fattori IAM nella Guida per l'IAMutente.

IAM users and groups

Un IAMutente è un'identità interna all'utente Account AWS che dispone di autorizzazioni specifiche per una singola persona o applicazione. Laddove possibile, consigliamo di fare affidamento su credenziali temporanee anziché creare IAM utenti con credenziali a lungo termine come password e chiavi di accesso. Tuttavia, se hai casi d'uso specifici che richiedono credenziali a lungo termine con IAM gli utenti, ti consigliamo di ruotare le chiavi di accesso. Per ulteriori informazioni, consulta Ruotare regolarmente le chiavi di accesso per i casi d'uso che richiedono credenziali a lungo termine nella Guida per l'utente. IAM

Un IAMgruppo è un'identità che specifica un insieme di utenti. IAM Non è possibile eseguire l'accesso come gruppo. È possibile utilizzare gruppi per specificare le autorizzazioni per più utenti alla volta. I gruppi semplificano la gestione delle autorizzazioni per set di utenti di grandi dimensioni. Ad esempio, potresti avere un gruppo denominato IAMAdminse concedere a quel gruppo le autorizzazioni per IAM amministrare le risorse.

Gli utenti sono diversi dai ruoli. Un utente è associato in modo univoco a una persona o un'applicazione, mentre un ruolo è destinato a essere assunto da chiunque ne abbia bisogno. Gli utenti dispongono di credenziali a lungo termine permanenti, mentre i ruoli forniscono credenziali temporanee. Per ulteriori informazioni, consulta Casi d'uso per IAM gli utenti nella Guida per l'IAMutente.

IAMruoli

Un IAMruolo è un'identità interna all'utente Account AWS che dispone di autorizzazioni specifiche. È simile a un IAM utente, ma non è associato a una persona specifica. Per assumere temporaneamente un IAM ruolo in AWS Management Console, puoi passare da un utente a un IAM ruolo (console). È possibile assumere un ruolo chiamando un' AWS APIoperazione AWS CLI or o utilizzando un'operazione personalizzataURL. Per ulteriori informazioni sui metodi di utilizzo dei ruoli, vedere Metodi per assumere un ruolo nella Guida per l'IAMutente.

IAMi ruoli con credenziali temporanee sono utili nelle seguenti situazioni:

- Accesso utente federato: per assegnare le autorizzazioni a una identità federata, è possibile creare un ruolo e definire le autorizzazioni per il ruolo. Quando un'identità federata viene autenticata, l'identità viene associata al ruolo e ottiene le autorizzazioni da esso definite. Per informazioni sui ruoli per la federazione, vedere Creazione di un ruolo per un provider di identità di terze parti nella Guida per l'IAMutente. Se utilizzi IAM Identity Center, configuri un set di autorizzazioni. Per controllare a cosa possono accedere le identità dopo l'autenticazione, IAM Identity Center correla

il set di autorizzazioni a un ruolo in. IAM Per informazioni sui set di autorizzazioni, consulta Set di autorizzazioni nella Guida per l'utente di AWS IAM Identity Center .

- Autorizzazioni IAM utente temporanee: un IAM utente o un ruolo può assumere il IAM ruolo di assumere temporaneamente autorizzazioni diverse per un'attività specifica.
- Accesso su più account: puoi utilizzare un IAM ruolo per consentire a qualcuno (un responsabile fidato) di un altro account di accedere alle risorse del tuo account. I ruoli sono lo strumento principale per concedere l'accesso multi-account. Tuttavia, con alcuni Servizi AWS, è possibile allegare una policy direttamente a una risorsa (anziché utilizzare un ruolo come proxy). Per conoscere la differenza tra ruoli e politiche basate sulle risorse per l'accesso tra account diversi, consulta la sezione Accesso alle risorse su più account IAM nella Guida per l'utente. IAM
- Accesso tra servizi: alcuni Servizi AWS utilizzano funzionalità in altri. Servizi AWS Ad esempio, quando effettui una chiamata in un servizio, è normale che quel servizio esegua applicazioni in Amazon EC2 o archivi oggetti in Amazon S3. Un servizio può eseguire questa operazione utilizzando le autorizzazioni dell'entità chiamante, utilizzando un ruolo di servizio o utilizzando un ruolo collegato al servizio.
  - Sessioni di accesso inoltrato (FAS): quando utilizzi un IAM utente o un ruolo per eseguire azioni AWS, sei considerato un principale. Quando si utilizzano alcuni servizi, è possibile eseguire un'operazione che attiva un'altra operazione in un servizio diverso. FASutilizza le autorizzazioni del principale che chiama an Servizio AWS, in combinazione con la richiesta di effettuare richieste Servizio AWS ai servizi downstream. FASIe richieste vengono effettuate solo quando un servizio riceve una richiesta che richiede interazioni con altri Servizi AWS o risorse per essere completata. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le azioni. Per i dettagli FAS delle politiche relative alle richieste, consulta Forward access sessions.
  - Ruolo di servizio: un ruolo di servizio è un IAMruolo che un servizio assume per eseguire azioni per conto dell'utente. Un IAM amministratore può creare, modificare ed eliminare un ruolo di servizio dall'internoIAM. Per ulteriori informazioni, vedere Creazione di un ruolo per delegare le autorizzazioni a un utente Servizio AWS nella Guida per l'IAMutente.
  - Ruolo collegato al servizio: un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un. Servizio AWS Il servizio può assumere il ruolo per eseguire un'azione per tuo conto. I ruoli collegati al servizio vengono visualizzati nel tuo account Account AWS e sono di proprietà del servizio. Un IAM amministratore può visualizzare, ma non modificare le autorizzazioni per i ruoli collegati al servizio.
- Applicazioni in esecuzione su Amazon EC2: puoi utilizzare un IAM ruolo per gestire le credenziali temporanee per le applicazioni in esecuzione su un'EC2istanza e che effettuano AWS CLI o richiedono AWS API. Ciò è preferibile alla memorizzazione delle chiavi di accesso all'interno

dell'EC2istanza. Per assegnare un AWS ruolo a un'EC2istanza e renderlo disponibile per tutte le sue applicazioni, create un profilo di istanza collegato all'istanza. Un profilo di istanza contiene il ruolo e consente ai programmi in esecuzione sull'EC2istanza di ottenere credenziali temporanee. Per ulteriori informazioni, consulta Usare un IAM ruolo per concedere le autorizzazioni alle applicazioni in esecuzione su EC2 istanze Amazon nella Guida per l'IAMutente.

Gestione dell'accesso con policy

Puoi controllare l'accesso AWS creando policy e associandole a AWS identità o risorse. Una policy è un oggetto AWS che, se associato a un'identità o a una risorsa, ne definisce le autorizzazioni. AWS valuta queste politiche quando un principale (utente, utente root o sessione di ruolo) effettua una richiesta. Le autorizzazioni nelle policy determinano l'approvazione o il rifiuto della richiesta. La maggior parte delle politiche viene archiviata AWS come JSON documenti. Per ulteriori informazioni sulla struttura e il contenuto dei documenti relativi alle JSON politiche, vedere Panoramica delle JSON politiche nella Guida per l'IAMutente.

Gli amministratori possono utilizzare AWS JSON le politiche per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire azioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Per concedere agli utenti l'autorizzazione a eseguire azioni sulle risorse di cui hanno bisogno, un IAM amministratore può creare IAM politiche. L'amministratore può quindi aggiungere le IAM politiche ai ruoli e gli utenti possono assumerli.

IAMle politiche definiscono le autorizzazioni per un'azione indipendentemente dal metodo utilizzato per eseguire l'operazione. Ad esempio, supponiamo di disporre di una policy che consente l'operazione `iam:GetRole`. Un utente con tale criterio può ottenere informazioni sul ruolo da AWS Management Console, da o da. AWS CLI AWS API

Policy basate su identità

I criteri basati sull'identità sono documenti relativi alle politiche di JSON autorizzazione che è possibile allegare a un'identità, ad esempio un IAM utente, un gruppo di utenti o un ruolo. Tali policy definiscono le azioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una politica basata sull'identità, consulta Definire le IAM autorizzazioni personalizzate con le politiche gestite dal cliente nella Guida per l'utente. IAM

Le policy basate su identità possono essere ulteriormente classificate come policy inline o policy gestite. Le policy inline sono integrate direttamente in un singolo utente, gruppo o ruolo. Le politiche

gestite sono politiche autonome che puoi allegare a più utenti, gruppi e ruoli all'interno del tuo. Account AWS Le politiche gestite includono politiche AWS gestite e politiche gestite dai clienti. Per informazioni su come scegliere tra una politica gestita o una politica in linea, consulta Scegliere tra politiche gestite e politiche in linea nella Guida per l'IAMutente.

Policy basate su risorse

Le politiche basate sulle risorse sono documenti di JSON policy allegati a una risorsa. Esempi di politiche basate sulle risorse sono le policy di trust dei IAM ruoli e le policy dei bucket di Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarli per controllare l'accesso a una risorsa specifica. Quando è collegata a una risorsa, una policy definisce le azioni che un principale può eseguire su tale risorsa e a quali condizioni. È necessario specificare un principale in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o. Servizi AWS

Le policy basate sulle risorse sono policy inline che si trovano in tale servizio. Non è possibile utilizzare le politiche AWS gestite IAM in una politica basata sulle risorse.

Elenchi di controllo degli accessi () ACLs

Le liste di controllo degli accessi (ACLs) controllano quali principali (membri dell'account, utenti o ruoli) dispongono delle autorizzazioni per accedere a una risorsa. ACLssono simili alle politiche basate sulle risorse, sebbene non utilizzino il formato del documento di policy. JSON

Amazon S3 e Amazon VPC sono esempi di servizi che supportano. AWS WAF ACLs Per ulteriori informazioniACLs, consulta la panoramica di Access control list (ACL) nella Amazon Simple Storage Service Developer Guide.

Altri tipi di policy

AWS supporta tipi di policy aggiuntivi e meno comuni. Questi tipi di policy possono impostare il numero massimo di autorizzazioni concesse dai tipi di policy più comuni.

- Limiti delle autorizzazioni: un limite di autorizzazioni è una funzionalità avanzata in cui si impostano le autorizzazioni massime che una politica basata sull'identità può concedere a un'entità (utente o ruolo). IAM IAM È possibile impostare un limite delle autorizzazioni per un'entità. Le autorizzazioni risultanti sono l'intersezione delle policy basate su identità dell'entità e i relativi limiti delle autorizzazioni. Le policy basate su risorse che specificano l'utente o il ruolo nel campo `Principal`sono condizionate dal limite delle autorizzazioni. Un rifiuto esplicito in una qualsiasi di

queste policy sostituisce l'autorizzazione. Per ulteriori informazioni sui limiti delle autorizzazioni, consulta Limiti delle autorizzazioni per le entità nella Guida per l'utente. IAM IAM

- Politiche di controllo del servizio (SCPs): SCPs sono JSON politiche che specificano le autorizzazioni massime per un'organizzazione o un'unità organizzativa (OU) in. AWS Organizations AWS Organizations è un servizio per il raggruppamento e la gestione centralizzata di più Account AWS di proprietà dell'azienda. Se abiliti tutte le funzionalità di un'organizzazione, puoi applicare le politiche di controllo del servizio (SCPs) a uno o tutti i tuoi account. SCPLimita le autorizzazioni per le entità negli account dei membri, inclusa ciascuna Utente root dell'account AWS. Per ulteriori informazioni su Organizations andSCPs, consulta le politiche di controllo dei servizi nella Guida AWS Organizations per l'utente.
- Policy di sessione: le policy di sessione sono policy avanzate che vengono trasmesse come parametro quando si crea in modo programmatico una sessione temporanea per un ruolo o un utente federato. Le autorizzazioni della sessione risultante sono l'intersezione delle policy basate su identità del ruolo o dell'utente e le policy di sessione. Le autorizzazioni possono anche provenire da una policy basata su risorse. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni, consulta le politiche di sessione nella Guida IAM per l'utente.

Più tipi di policy

Quando più tipi di policy si applicano a una richiesta, le autorizzazioni risultanti sono più complicate da comprendere. Per informazioni su come AWS determinare se consentire una richiesta quando sono coinvolti più tipi di policy, consulta Logica di valutazione delle politiche nella Guida per l'IAMutente.

Come funziona Amazon Timestream per InfluxDB con IAM

IAMfunzionalità che puoi usare con Amazon Timestream per InfluxDB

IAMfunzionalità	Timestream per il supporto di InfluxDB
Policy basate su identità	Sì
Policy basate su risorse	No
Azioni di policy	Sì

IAMfunzionalità	Timestream per il supporto di InfluxDB
Risorse relative alle policy	Sì
Chiavi di condizione delle policy	No
ACLs	No
ABAC(tag nelle politiche)	Sì
Credenziali temporanee	Sì
Autorizzazioni del principale	Sì
☻Ruoli di servizio	No
Ruoli collegati al servizio	Sì

Per avere una visione di alto livello di come Timestream for InfluxDB e altri AWS servizi funzionano con la maggior parte delle IAM funzionalità, consulta i AWS servizi con cui funzionano nella Guida per l'utente. IAM IAM

Politiche basate sull'identità per Timestream for InfluxDB

Supporta le policy basate su identità: sì

Le politiche basate sull'identità sono documenti relativi alle politiche di JSON autorizzazione che è possibile allegare a un'identità, ad esempio un utente, un gruppo di utenti o un ruolo. IAM Tali policy definiscono le azioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una politica basata sull'identità, consulta Definire le IAM autorizzazioni personalizzate con le politiche gestite dal cliente nella Guida per l'utente. IAM

Con le politiche IAM basate sull'identità, puoi specificare azioni e risorse consentite o negate, nonché le condizioni in base alle quali le azioni sono consentite o negate. Non è possibile specificare l'entità principale in una policy basata sull'identità perché si applica all'utente o al ruolo a cui è associato. Per ulteriori informazioni su tutti gli elementi che è possibile utilizzare in una JSON politica, vedere il riferimento agli elementi IAM JSON della politica nella Guida per l'IAMutente.

Esempi di policy basate sull'identità per Timestream for InfluxDB

Per visualizzare esempi di politiche basate sull'identità di Timestream for InfluxDB, consulta.. Esempi di policy basate sull'identità per Amazon Timestream for InfluxDB

Politiche basate sulle risorse all'interno di Timestream for InfluxDB

Supporta le policy basate su risorse: no

Le politiche basate sulle risorse sono documenti politici allegati a una risorsa. JSON Esempi di politiche basate sulle risorse sono le policy di trust dei IAM ruoli e le policy dei bucket di Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarli per controllare l'accesso a una risorsa specifica. Quando è collegata a una risorsa, una policy definisce le azioni che un principale può eseguire su tale risorsa e a quali condizioni. È necessario specificare un principale in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o. Servizi AWS

Per abilitare l'accesso tra più account, puoi specificare un intero account o IAM entità in un altro account come principale in una politica basata sulle risorse. L'aggiunta di un principale multi-account a una policy basata sulle risorse rappresenta solo una parte della relazione di trust. Quando il principale e la risorsa sono diversi Account AWS, un IAM amministratore dell'account fidato deve inoltre concedere all'entità principale (utente o ruolo) l'autorizzazione ad accedere alla risorsa. L'autorizzazione viene concessa collegando all'entità una policy basata sull'identità. Tuttavia, se una policy basata su risorse concede l'accesso a un principale nello stesso account, non sono richieste ulteriori policy basate su identità. Per ulteriori informazioni, consulta la sezione Cross Account Resource Access IAM nella Guida IAM per l'utente.

Azioni politiche per Timestream for InfluxDB

Supporta le operazioni di policy: si

Gli amministratori possono utilizzare AWS JSON le politiche per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'`Action`elemento di una JSON policy descrive le azioni che è possibile utilizzare per consentire o negare l'accesso a una policy. Le azioni politiche in genere hanno lo stesso nome dell' AWS APIoperazione associata. Esistono alcune eccezioni, come le azioni basate solo sulle autorizzazioni che non hanno un'operazione corrispondente. API Esistono anche alcune operazioni che richiedono più operazioni in una policy. Queste operazioni aggiuntive sono denominate operazioni dipendenti.

Includi le operazioni in una policy per concedere le autorizzazioni a eseguire l'operazione associata.

Per visualizzare un elenco delle azioni di Timestream for InfluxDB, consulta Actions Defined by Amazon Timestream for InfluxDB nel Service Authorization Reference.

Le azioni politiche in Timestream for InfluxDB utilizzano il seguente prefisso prima dell'azione:

```
timestream-influxdb
```

Per specificare più operazioni in una sola istruzione, occorre separarle con la virgola.

```
"Action": [
      "timestream-influxdb:action1",
      "timestream-influxdb:action2"
         ]
```

È possibile specificare più azioni tramite caratteri jolly (*). Ad esempio, per specificare tutte le azioni che iniziano con la parola `Describe`, includi la seguente azione:

```
"Action": "timestream-influxdb:Describe*"
```

Risorse politiche per Timestream for InfluxDB

Supporta le risorse di policy: sì

Gli amministratori possono utilizzare AWS JSON le politiche per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento `Resource` JSON policy specifica l'oggetto o gli oggetti a cui si applica l'azione. Le istruzioni devono includere un elemento `Resource`o un elemento `NotResource`. Come best practice, specifica una risorsa utilizzando il relativo Amazon Resource Name (ARN). Puoi eseguire questa operazione per azioni che supportano un tipo di risorsa specifico, note come autorizzazioni a livello di risorsa.

Per le azioni che non supportano le autorizzazioni a livello di risorsa, ad esempio le operazioni di elenco, utilizza un carattere jolly (*) per indicare che l'istruzione si applica a tutte le risorse.

```
"Resource": "*"
```

Per visualizzare un elenco dei tipi di risorse Timestream for InfluxDB e relativiARNs, consulta Resources Defined by Amazon Timestream for InfluxDB nel Service Authorization Reference. Per

sapere con quali azioni puoi specificare le caratteristiche ARN di ciascuna risorsa, consulta Azioni definite da Amazon Timestream per InfluxDB.

Chiavi delle condizioni politiche per Timestream for InfluxDB

Supporta le chiavi delle condizioni delle politiche specifiche del servizio: No

Gli amministratori possono utilizzare AWS JSON le policy per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire azioni su quali risorse, e in quali condizioni.

L'elemento `Condition`(o blocco `Condition`) consente di specificare le condizioni in cui un'istruzione è in vigore. L'elemento `Condition`è facoltativo. Puoi compilare espressioni condizionali che utilizzano operatori di condizione, ad esempio uguale a o minore di, per soddisfare la condizione nella policy con i valori nella richiesta.

Se specifichi più elementi `Condition`in un'istruzione o più chiavi in un singolo elemento `Condition`, questi vengono valutati da AWS utilizzando un'operazione `AND`logica. Se si specificano più valori per una singola chiave di condizione, AWS valuta la condizione utilizzando un'operazione logica`OR`. Tutte le condizioni devono essere soddisfatte prima che le autorizzazioni dell'istruzione vengano concesse.

Puoi anche utilizzare variabili segnaposto quando specifichi le condizioni. Ad esempio, è possibile concedere a un IAM utente l'autorizzazione ad accedere a una risorsa solo se è contrassegnata con il suo nome IAM utente. Per ulteriori informazioni, consulta gli elementi IAM della politica: variabili e tag nella Guida IAM per l'utente.

AWS supporta chiavi di condizione globali e chiavi di condizione specifiche del servizio. Per visualizzare tutte le chiavi di condizione AWS globali, consulta le chiavi di contesto delle condizioni AWS globali nella Guida per l'IAMutente.

Accedi agli elenchi di controllo (ACLs) in Timestream per InfluxDB

ACLsSupporti: No

Le liste di controllo degli accessi (ACLs) controllano quali principali (membri dell'account, utenti o ruoli) dispongono delle autorizzazioni per accedere a una risorsa. ACLssono simili alle politiche basate sulle risorse, sebbene non utilizzino il formato del documento di policy. JSON

Controllo degli accessi basato sugli attributi () con Timestream per InfluxDB ABAC

Supporti ABAC (tag nelle politiche): Sì

Il controllo degli accessi basato sugli attributi (ABAC) è una strategia di autorizzazione che definisce le autorizzazioni in base agli attributi. In AWS, questi attributi sono chiamati tag. È possibile allegare tag a IAM entità (utenti o ruoli) e a molte AWS risorse. L'etichettatura di entità e risorse è il primo passo diABAC. Quindi si progettano ABAC politiche per consentire le operazioni quando il tag del principale corrisponde al tag sulla risorsa a cui sta tentando di accedere.

ABACè utile in ambienti in rapida crescita e aiuta in situazioni in cui la gestione delle politiche diventa complicata.

Per controllare l'accesso basato su tag, fornisci informazioni sui tag nell'elemento condizione di una policy utilizzando le chiavi di condizione `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`o `aws:TagKeys`.

Se un servizio supporta tutte e tre le chiavi di condizione per ogni tipo di risorsa, il valore per il servizio è Yes (Sì). Se un servizio supporta tutte e tre le chiavi di condizione solo per alcuni tipi di risorsa, allora il valore sarà Parziale.

Per ulteriori informazioni in meritoABAC, vedere Definizione delle autorizzazioni con ABAC autorizzazione nella Guida per l'IAMutente. Per visualizzare un tutorial con i passaggi per la configurazioneABAC, consulta Use Attribute-based access control (ABAC) nella Guida per l'utente. IAM

Utilizzo di credenziali temporanee con Timestream per InfluxDB

Supporta le credenziali temporanee: sì

Alcune Servizi AWS non funzionano quando si accede utilizzando credenziali temporanee. Per ulteriori informazioni, incluse quelle che Servizi AWS funzionano con credenziali temporanee, consulta la sezione Servizi AWS relativa alla funzionalità IAM nella Guida per l'IAMutente.

Si utilizzano credenziali temporanee se si accede AWS Management Console utilizzando qualsiasi metodo tranne il nome utente e la password. Ad esempio, quando accedete AWS utilizzando il link Single Sign-on (SSO) della vostra azienda, tale processo crea automaticamente credenziali temporanee. Le credenziali temporanee vengono create in automatico anche quando accedi alla console come utente e poi cambi ruolo. Per ulteriori informazioni sul cambio di ruolo, consulta Passare da un utente a un IAM ruolo (console) nella Guida per l'IAMutente.

È possibile creare manualmente credenziali temporanee utilizzando AWS CLI o AWS API. È quindi possibile utilizzare tali credenziali temporanee per accedere. AWS AWS consiglia di generare dinamicamente credenziali temporanee anziché utilizzare chiavi di accesso a lungo termine. Per ulteriori informazioni, vedere Credenziali di sicurezza temporanee in. IAM

Autorizzazioni principali multiservizio per Timestream for InfluxDB

Supporta sessioni di accesso diretto (): Sì FAS

Quando utilizzi un IAM utente o un ruolo per eseguire azioni AWS, sei considerato un principale. Quando si utilizzano alcuni servizi, è possibile eseguire un'operazione che attiva un'altra operazione in un servizio diverso. FASutilizza le autorizzazioni del principale che chiama an Servizio AWS, in combinazione con la richiesta Servizio AWS per effettuare richieste ai servizi downstream. FASle richieste vengono effettuate solo quando un servizio riceve una richiesta che richiede interazioni con altri Servizi AWS o risorse per essere completata. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le azioni. Per i dettagli FAS delle politiche relative alle richieste, consulta Forward access sessions.

Ruoli di servizio per Timestream for InfluxDB

Supporta i ruoli di servizio: No

Un ruolo di servizio è un IAMruolo che un servizio assume per eseguire azioni per conto dell'utente. Un IAM amministratore può creare, modificare ed eliminare un ruolo di servizio dall'internoIAM. Per ulteriori informazioni, vedere Creazione di un ruolo per delegare le autorizzazioni a un utente Servizio AWS nella Guida per l'IAMutente.

> **Warning**
>
> La modifica delle autorizzazioni per un ruolo di servizio potrebbe interrompere la funzionalità di Timestream for InfluxDB. Modifica i ruoli di servizio solo quando Timestream for InfluxDB fornisce indicazioni in tal senso.

Ruoli collegati ai servizi per Timestream for InfluxDB

Supporta ruoli collegati ai servizi: Sì

Un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un. Servizio AWS Il servizio può assumere il ruolo per eseguire un'azione per tuo conto. I ruoli collegati al servizio vengono visualizzati nel tuo account Account AWS e sono di proprietà del servizio. Un IAM amministratore può visualizzare, ma non modificare le autorizzazioni per i ruoli collegati al servizio.

Per informazioni dettagliate sulla creazione o la gestione di ruoli collegati ai servizi, consulta AWS Servizi compatibili con. IAM Trova un servizio nella tabella che include un `Yes` nella colonna Service-

linked role (Ruolo collegato ai servizi). Scegli il collegamento Sì per visualizzare la documentazione relativa al ruolo collegato ai servizi per tale servizio.

Esempi di policy basate sull'identità per Amazon Timestream for InfluxDB

Per impostazione predefinita, gli utenti e i ruoli non sono autorizzati a creare o modificare le risorse Timestream for InfluxDB. Inoltre, non possono eseguire attività utilizzando AWS Management Console, AWS Command Line Interface ()AWS CLI o. AWS API Per concedere agli utenti il permesso di eseguire azioni sulle risorse di cui hanno bisogno, un IAM amministratore può creare IAM policy. L'amministratore può quindi aggiungere le IAM politiche ai ruoli e gli utenti possono assumerli.

Per informazioni su come creare una politica IAM basata sull'identità utilizzando questi documenti di esempio, consulta Create JSON IAM policy (console) nella Guida per l'IAMutente.

Per dettagli sulle azioni e sui tipi di risorse definiti da Timestream per InfluxDB, incluso il formato di ARNs per ogni tipo di risorsa, consulta Actions, Resources and Condition Keys for Amazon Timestream for InfluxDB nel Service Authorization Reference.

Argomenti

Best practice per le policy

Le politiche basate sull'identità determinano se qualcuno può creare, accedere o eliminare le risorse di Timestream for InfluxDB nel tuo account. Queste azioni possono comportare costi aggiuntivi per l' Account AWS. Quando crei o modifichi policy basate su identità, segui queste linee guida e raccomandazioni:

- Inizia con le politiche AWS gestite e passa alle autorizzazioni con privilegi minimi: per iniziare a concedere autorizzazioni a utenti e carichi di lavoro, utilizza le politiche gestite che concedono le autorizzazioni per molti casi d'uso comuni.AWS Sono disponibili nel tuo. Account AWS Ti

consigliamo di ridurre ulteriormente le autorizzazioni definendo politiche gestite dai AWS clienti specifiche per i tuoi casi d'uso. Per ulteriori informazioni, consulta le politiche AWS gestite o le politiche AWS gestite per le funzioni lavorative nella Guida per l'IAMutente.

- Applica le autorizzazioni con privilegi minimi: quando imposti le autorizzazioni con le IAM politiche, concedi solo le autorizzazioni necessarie per eseguire un'attività. Puoi farlo definendo le azioni che possono essere intraprese su risorse specifiche in condizioni specifiche, note anche come autorizzazioni con privilegi minimi. Per ulteriori informazioni sull'utilizzo per applicare le autorizzazioni, consulta Politiche e autorizzazioni nella Guida IAM per l'utente. IAM IAM
- Utilizza le condizioni nelle IAM politiche per limitare ulteriormente l'accesso: puoi aggiungere una condizione alle tue politiche per limitare l'accesso ad azioni e risorse. Ad esempio, puoi scrivere una condizione di policy per specificare che tutte le richieste devono essere inviate utilizzandoSSL. È inoltre possibile utilizzare condizioni per concedere l'accesso alle azioni di servizio se vengono utilizzate tramite uno specifico Servizio AWS, ad esempio AWS CloudFormation. Per ulteriori informazioni, consulta Elementi IAM JSON della politica: Condizione nella Guida IAM per l'utente.
- Usa IAM Access Analyzer per convalidare IAM le tue policy e garantire autorizzazioni sicure e funzionali: IAM Access Analyzer convalida le policy nuove ed esistenti in modo che aderiscano al linguaggio delle IAM policy () e alle best practice. JSON IAM IAMAccess Analyzer fornisce più di 100 controlli delle politiche e consigli pratici per aiutarti a creare policy sicure e funzionali. Per ulteriori informazioni, consulta Convalida delle politiche con IAM Access Analyzer nella Guida per l'utente. IAM
- Richiedi l'autenticazione a più fattori (MFA): se hai uno scenario che richiede l'utilizzo di IAM utenti o di un utente root Account AWS, attiva questa opzione MFA per una maggiore sicurezza. Per richiedere MFA quando vengono richiamate API le operazioni, aggiungi MFA delle condizioni alle tue politiche. Per ulteriori informazioni, consulta Secure API access with MFA nella Guida IAM per l'utente.

Per ulteriori informazioni sulle best practice inIAM, consulta la sezione Procedure consigliate in materia di sicurezza IAM nella Guida IAM per l'utente.

Utilizzo della console Timestream for InfluxDB

Per accedere alla console Amazon Timestream for InfluxDB, devi disporre di un set minimo di autorizzazioni. Queste autorizzazioni devono consentirti di elencare e visualizzare i dettagli sulle risorse Timestream for InfluxDB presenti nel tuo. Account AWS Se crei una policy basata sull'identità più restrittiva rispetto alle autorizzazioni minime richieste, la console non funzionerà nel modo previsto per le entità (utenti o ruoli) associate a tale policy.

Non è necessario consentire le autorizzazioni minime della console per gli utenti che effettuano chiamate solo verso il o il. AWS CLI AWS API Consenti invece l'accesso solo alle azioni che corrispondono all'APIoperazione che stanno cercando di eseguire.

Per garantire che utenti e ruoli possano ancora utilizzare la console Timestream for InfluxDB, collega anche Timestream for InfluxDB o la policy gestita alle entità`ConsoleAccess`. `ReadOnly` AWS Per ulteriori informazioni, consulta Aggiungere autorizzazioni a un utente nella Guida per l'utente. IAM

Consentire agli utenti di visualizzare le loro autorizzazioni

Questo esempio mostra come è possibile creare una politica che consenta IAM agli utenti di visualizzare le politiche in linea e gestite allegate alla loro identità utente. Questa politica include le autorizzazioni per completare questa azione sulla console o utilizzando o a livello di codice. AWS CLI AWS API

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

Accesso a un bucket Amazon S3

In questo esempio, vuoi concedere a un IAM utente del tuo AWS account l'accesso a uno dei tuoi bucket Amazon S3,. `examplebucket` Si vuole anche consentire all'utente di aggiungere, aggiornare ed eliminare oggetti.

Oltre ad assegnare le autorizzazioni `s3:PutObject`, `s3:GetObject` e `s3:DeleteObject` all'utente, la policy assegna anche le autorizzazioni `s3:ListAllMyBuckets`, `s3:GetBucketLocation` e `s3:ListBucket`. Queste sono le autorizzazioni aggiuntive richieste dalla console. Inoltre, le operazioni `s3:PutObjectAcl` e `s3:GetObjectAcl` sono necessarie per essere in grado di copiare, tagliare e incollare gli oggetti nella console. Per un esempio di procedura dettagliata che concede le autorizzazioni agli utenti e li verifica utilizzando la console, consulta Un esempio di procedura dettagliata: Using user policy to control access to your bucket.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Sid":"ListBucketsInConsole",
         "Effect":"Allow",
         "Action":[
            "s3:ListAllMyBuckets"
         ],
         "Resource":"arn:aws:s3:::*"
      },
      {
         "Sid":"ViewSpecificBucketInfo",
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket",
            "s3:GetBucketLocation"
         ],
         "Resource":"arn:aws:s3:::examplebucket"
      },
      {
         "Sid":"ManageBucketContents",
         "Effect":"Allow",
```

```
         "Action":[
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:GetObject",
            "s3:GetObjectAcl",
            "s3:DeleteObject"
         ],
         "Resource":"arn:aws:s3:::examplebucket/*"
      }
   ]
}
```

Consentire tutte le operazioni

Di seguito è riportato un esempio di politica che consente tutte le operazioni in Timestream for InfluxDB.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream-influxdb:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Crea, descrivi, elimina e aggiorna un'istanza DB

La seguente policy di esempio consente a un utente di creare, descrivere, eliminare e aggiornare un'istanza DB`sampleDB`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "timestream-influxdb:CreateDbInstance",
```

```
                "timestream-influxdb:GetDbInstance",
                "timestream-influxdb:DeleteDbInstance",
                "timestream-influxdb:UpdateDbInstance"
            ],
            "Resource": "arn:aws:timestream-influxdb:us-east-1:<account_ID>:dbinstance/
sampleDB"
        }
    ]
}
```

Risoluzione dei problemi relativi all'identità e all'accesso di Amazon Timestream per InfluxDB

Utilizza le seguenti informazioni per aiutarti a diagnosticare e risolvere i problemi più comuni che potresti riscontrare lavorando con Timestream for InfluxDB e. IAM

Argomenti

- Non sono autorizzato a eseguire un'azione in Timestream for InfluxDB
- Voglio consentire a persone esterne al mio AWS account di accedere alle mie risorse Timestream for InfluxDB

Non sono autorizzato a eseguire un'azione in Timestream for InfluxDB

Se ti AWS Management Console dice che non sei autorizzato a eseguire un'azione, devi contattare l'amministratore per ricevere assistenza. L'amministratore è la persona da cui si sono ricevuti il nome utente e la password.

Il seguente esempio di errore si verifica quando l'utente `mateojackson` prova a utilizzare la console per visualizzare i dettagli relativi a una risorsa *`my-example-widget`* fittizia, ma non dispone di autorizzazioni `timestream-influxdb:`*`GetWidget`* fittizie.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 timestream-influxdb:GetWidget on resource: my-example-widget
```

In questo caso, Mateo richiede al suo amministratore di aggiornare le policy per poter accedere alla risorsa *`my-example-widget`* utilizzando l'operazione `timestream-influxdb:`*`GetWidget`*.

Voglio consentire a persone esterne al mio AWS account di accedere alle mie risorse Timestream for InfluxDB

È possibile creare un ruolo con il quale utenti in altri account o persone esterne all'organizzazione possono accedere alle tue risorse. È possibile specificare chi è attendibile per l'assunzione del ruolo. Per i servizi che supportano politiche basate sulle risorse o liste di controllo degli accessi (ACLs), puoi utilizzare tali politiche per concedere alle persone l'accesso alle tue risorse.

Per ulteriori informazioni, consulta gli argomenti seguenti:

- Controllo dell'accesso a un'istanza DB in un VPC
- Per sapere se Timestream for InfluxDB supporta queste funzionalità, consulta Come funziona Amazon Timestream for InfluxDB. IAM
- Per sapere come fornire l'accesso alle tue risorse su più AWS account di tua proprietà, consulta Fornire l'accesso a un IAM utente in un altro AWS account di tua proprietà nella Guida per l'utente. IAM
- Per informazioni su come fornire l'accesso alle tue risorse ad AWS account di terze parti, consulta Fornire l'accesso agli AWS account di proprietà di terzi nella Guida per l'IAMutente.
- Per informazioni su come fornire l'accesso tramite la federazione delle identità, consulta Fornire l'accesso agli utenti autenticati esternamente (federazione delle identità) nella Guida per l'IAMutente.
- Per conoscere la differenza tra l'utilizzo dei ruoli e delle politiche basate sulle risorse per l'accesso tra account diversi, consulta In che modo i IAM ruoli differiscono dalle politiche basate sulle risorse nella Guida per l'utente. IAM

Controllo dell'accesso a un'istanza DB in un VPC

Utilizzando Amazon Virtual Private Cloud (AmazonVPC), puoi avviare AWS risorse, come Amazon Timestream per istanze DB InfluxDB, in un cloud privato virtuale (). VPC Quando usi AmazonVPC, hai il controllo del tuo ambiente di rete virtuale. Puoi scegliere il tuo intervallo di indirizzi IP, creare sottoreti e configurare liste di routing e di controllo accessi.

Un gruppo VPC di sicurezza controlla l'accesso alle istanze DB all'interno di unVPC. Ogni regola del gruppo di VPC sicurezza consente a una fonte specifica di accedere a un'VPCistanza DB associata a quel gruppo VPC di sicurezza. L'origine può essere un intervallo di indirizzi (ad esempio, 203.0.113.0/24) o un altro gruppo di sicurezza. VPC Specificando un gruppo VPC di sicurezza come origine, consentite il traffico in entrata da tutte le istanze (in genere server di applicazioni)

che utilizzano il gruppo di sicurezza di origine. VPC Prima di provare a connetterti all'istanza DB, configurala VPC per il tuo caso d'uso. Di seguito sono riportati gli scenari più comuni per l'accesso a un'istanza DB in unVPC:

Un'istanza DB in un'istanza a VPC cui accede un'EC2istanza Amazon nella stessa VPC

Un uso comune di un'istanza DB in a VPC consiste nel condividere dati con un server di applicazioni in esecuzione in un'EC2istanza della stessaVPC. L'EC2istanza potrebbe eseguire un server Web con un'applicazione che interagisce con l'istanza DB.

Un'istanza DB in un'istanza VPC accessibile da un'EC2istanza in un'altra VPC

In alcuni casi, l'istanza DB si trova in un'istanza diversa VPC dall'EC2istanza utilizzata per accedervi. In tal caso, puoi utilizzare il VPC peering per accedere all'istanza DB.

Un'istanza DB in un'applicazione client VPC accessibile tramite Internet

Per accedere a un'istanza DB in un'VPCapplicazione client tramite Internet, è necessario configurare un'istanza DB VPC con una singola sottorete pubblica e utilizzare le sottoreti pubbliche per creare l'istanza DB. È inoltre possibile configurare un gateway Internet VPC per abilitare la comunicazione su Internet. Per connettersi a un'istanza DB dall'esternoVPC, l'istanza DB deve essere accessibile pubblicamente. Inoltre, l'accesso deve essere concesso utilizzando le regole in ingresso del gruppo di sicurezza dell'istanza database e devono essere soddisfatti altri requisiti.

Per ulteriori informazioni sui gruppi VPC di sicurezza, consulta la sezione Gruppi di sicurezza nella Amazon Virtual Private Cloud User Guide.

Per i dettagli su come connettersi a un'istanza di Timestream for InfluxDB DB, consulta. Connessione a un'istanza database Amazon Timestream for InfluxDB

Scenario del gruppo di sicurezza

Un uso comune di un'istanza DB in a VPC consiste nel condividere i dati con un server delle applicazioni in esecuzione in un'EC2istanza Amazon nella stessaVPC, a cui si accede da un'applicazione client esterna aVPC. In questo scenario, utilizzi Timestream per InfluxDB e le VPC pagine su AWS Management Console o Timestream per InfluxDB e le EC2 API operazioni per creare le istanze e i gruppi di sicurezza necessari:

1. Create un gruppo VPC di sicurezza (ad esempio`sg-0123ec2example`) e definite regole in entrata che utilizzano gli indirizzi IP dell'applicazione client come origine. Questo gruppo di

sicurezza consente all'applicazione client di connettersi alle EC2 istanze in un ambiente VPC che utilizza questo gruppo di sicurezza.

2. Create un'EC2istanza per l'applicazione e aggiungete l'EC2istanza al gruppo di VPC sicurezza (`sg-0123ec2example`) creato nel passaggio precedente.
3. Create un secondo gruppo di VPC sicurezza (ad esempio`sg-6789rdsexample`) e create una nuova regola specificando come origine il gruppo di VPC sicurezza creato nel passaggio 1 (`sg-0123ec2example`).
4. Crea una nuova istanza DB e aggiungi l'istanza DB al gruppo di VPC sicurezza (`sg-6789rdsexample`) creato nel passaggio precedente. Quando create il DB, utilizzate lo stesso numero di porta specificato per la regola del gruppo di VPC sicurezza (`sg-6789rdsexample`) creata nel passaggio 3.

Creazione di un gruppo VPC di sicurezza

È possibile creare un gruppo VPC di sicurezza per un'istanza DB utilizzando la VPC console. Per informazioni sulla creazione di un gruppo di sicurezza, consulta Security groups nella Amazon Virtual Private Cloud User Guide.

Associazione di un gruppo di sicurezza a un istanza database

Puoi associare un gruppo di sicurezza a un'istanza DB utilizzando Update sulla console Timestream for InfluxDB, `UpdateDBInstance` Timestream for InfluxDB o il comando. API `update-db-instance` AWS CLI

L'CLIesempio seguente associa un gruppo di VPC sicurezza specifico e rimuove i gruppi di sicurezza DB dall'istanza DB

```
aws timestream-influxdb update-db-instance --identifier dbName --vpc-security-group-
ids sg-ID
```

Per ulteriori informazioni sulla modifica di un'istanza database, consulta Aggiornamento delle istanze database.

Utilizzo di ruoli collegati ai servizi per Amazon Timestream for InfluxDB

Amazon Timestream for InfluxDB AWS Identity and Access Management utilizza () ruoli collegati al servizio. IAM Un ruolo collegato al servizio è un tipo unico di IAM ruolo collegato direttamente a un AWS servizio, come Amazon Timestream per InfluxDB. I ruoli collegati ai servizi di Amazon Timestream for InfluxDB sono predefiniti da Amazon Timestream per InfluxDB. Includono tutte le

autorizzazioni richieste dal servizio per chiamare i servizi per conto delle tue istanze di database. AWS

Un ruolo collegato al servizio semplifica la configurazione di Amazon Timestream per InfluxDB perché non è necessario aggiungere manualmente le autorizzazioni necessarie. I ruoli esistono già nel tuo AWS account ma sono collegati ai casi d'uso di Amazon Timestream for InfluxDB e dispongono di autorizzazioni predefinite. Solo Amazon Timestream for InfluxDB può assumere questi ruoli e solo questi ruoli possono utilizzare la politica di autorizzazioni predefinita. È possibile eliminare i ruoli solo dopo aver eliminato le risorse correlate. Questo protegge le tue risorse Amazon Timestream for InfluxDB perché non puoi rimuovere inavvertitamente le autorizzazioni necessarie per accedere alle risorse.

Per informazioni su altri servizi che supportano ruoli collegati ai servizi, consulta Services That Work with IAM e cerca AWS i servizi con Sì nella colonna Service-Linked Role. Scegli Sì in corrispondenza di un link per visualizzare la documentazione relativa al ruolo collegato ai servizi per tale servizio.

Indice

Autorizzazioni di ruolo collegate ai servizi per Amazon Timestream for InfluxDB

Amazon Timestream for InfluxDB utilizza il ruolo collegato al servizio AmazonTimestreamInfluxDBServiceRolePolicydenominato: questa politica consente a Timestream for InfluxDB di gestire le risorse per tuo conto, se necessario AWS per la gestione dei tuoi cluster.

La politica AmazonTimestreamInflux DBServiceRolePolicy di autorizzazione dei ruoli collegati al servizio consente ad Amazon Timestream for InfluxDB di completare le seguenti azioni sulle risorse specificate:

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "DescribeNetworkStatement",
   "Effect": "Allow",
   "Action": [
    "ec2:DescribeSubnets",
    "ec2:DescribeVpcs",
    "ec2:DescribeNetworkInterfaces"
   ],
   "Resource": "*"
  },
  {
   "Sid": "CreateEniInSubnetStatement",
   "Effect": "Allow",
   "Action": [
    "ec2:CreateNetworkInterface"
   ],
   "Resource": [
    "arn:aws:ec2:*:*:subnet/*",
    "arn:aws:ec2:*:*:security-group/*"
   ]
  },
  {
   "Sid": "CreateEniStatement",
   "Effect": "Allow",
   "Action": [
    "ec2:CreateNetworkInterface"
   ],
   "Resource": "arn:aws:ec2:*:*:network-interface/*",
   "Condition": {
    "Null": {
     "aws:RequestTag/AmazonTimestreamInfluxDBManaged": "false"
    }
   }
  },
  {
   "Sid": "CreateTagWithEniStatement",
```

```
    "Effect": "Allow",
    "Action": [
     "ec2:CreateTags"
    ],
    "Resource": "arn:aws:ec2:*:*:network-interface/*",
    "Condition": {
     "Null": {
      "aws:RequestTag/AmazonTimestreamInfluxDBManaged": "false"
     },
     "StringEquals": {
      "ec2:CreateAction": [
       "CreateNetworkInterface"
      ]
     }
    }
   },
   {
    "Sid": "ManageEniStatement",
    "Effect": "Allow",
    "Action": [
     "ec2:CreateNetworkInterfacePermission",
     "ec2:DeleteNetworkInterface"
    ],
    "Resource": "arn:aws:ec2:*:*:network-interface/*",
    "Condition": {
     "Null": {
      "aws:ResourceTag/AmazonTimestreamInfluxDBManaged": "false"
     }
    }
   },
   {
    "Sid": "PutCloudWatchMetricsStatement",
    "Effect": "Allow",
    "Action": [
     "cloudwatch:PutMetricData"
    ],
    "Condition": {
     "StringEquals": {
      "cloudwatch:namespace": [
       "AWS/Timestream/InfluxDB",
       "AWS/Usage"
      ]
     }
    },
```

```
   "Resource": [
    "*"
   ]
  },
  {
   "Sid": "ManageSecretStatement",
   "Effect": "Allow",
   "Action": [
    "secretsmanager:CreateSecret",
    "secretsmanager:DeleteSecret"
   ],
   "Resource": [
    "arn:aws:secretsmanager:*:*:secret:READONLY-InfluxDB-auth-parameters-*"
   ],
   "Condition": {
    "StringEquals": {
     "aws:ResourceAccount": "${aws:PrincipalAccount}"
    }
   }
  }
 ]
}
```

Per consentire a un'entità di creare ruoli collegati ai servizi IAM AmazonTimestreamInflux DBServiceRolePolicy

Aggiungi la seguente dichiarazione politica alle autorizzazioni per quell'entità: IAM

```
{
    "Effect": "Allow",
    "Action": [
        "iam:CreateServiceLinkedRole",
        "iam:PutRolePolicy"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/
timestreamforinfluxdb.amazonaws.com/AmazonTimestreamInfluxDBServiceRolePolicy*",
    "Condition": {"StringLike": {"iam:AWS ServiceName":
 "timestreamforinfluxdb.amazonaws.com"}}
}
```

Per consentire a un'IAMentità di eliminare i ruoli collegati al AmazonTimestreamInflux DBServiceRolePolicy servizio

Aggiungi la seguente dichiarazione politica alle autorizzazioni per quell'entità: IAM

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/
timestreamforinfluxdb.amazonaws.com/AmazonTimestreamInfluxDBServiceRolePolicy*",
    "Condition": {"StringLike": {"iam:AWS ServiceName":
 "timestreamforinfluxdb.amazonaws.com"}}
}
```

In alternativa, puoi utilizzare una policy AWS gestita per fornire l'accesso completo ad Amazon Timestream for InfluxDB.

Creazione di un ruolo collegato ai servizi () IAM

Non hai bisogno di creare manualmente un ruolo collegato ai servizi. Quando crei un'istanza DB, Amazon Timestream per InfluxDB crea il ruolo collegato al servizio per te.

Se elimini questo ruolo collegato ai servizi, puoi ricrearlo seguendo lo stesso processo utilizzato per ricreare il ruolo nell'account. Quando crei un'istanza DB, Amazon Timestream per InfluxDB crea nuovamente il ruolo collegato al servizio per te.

Modifica della descrizione di un ruolo collegato ai servizi per Amazon Timestream for InfluxDB

Amazon Timestream for InfluxDB non consente di modificare il ruolo collegato al servizio. AmazonTimestreamInflux DBServiceRolePolicy Dopo aver creato un ruolo collegato al servizio, non potrai modificarne il nome perché varie entità potrebbero farvi riferimento. Tuttavia, puoi modificare la descrizione del ruolo utilizzando. IAM

Modifica della descrizione di un ruolo collegato ai servizi (console) IAM

È possibile utilizzare la IAM console per modificare la descrizione di un ruolo collegato al servizio.

Per modificare la descrizione di un ruolo collegato ai servizi (console)

1. Nel riquadro di navigazione a sinistra della IAM console, scegli Ruoli.
2. Scegliere il nome del ruolo da modificare.
3. Nella parte destra di Role description (Descrizione ruolo), scegliere Edit (Modifica).

4. Digita una nuova descrizione nella casella e scegli Save (Salva).

Modifica della descrizione di un ruolo collegato al servizio () IAM CLI

È possibile utilizzare IAM le operazioni di AWS Command Line Interface per modificare la descrizione di un ruolo collegato al servizio.

Per modificare la descrizione di un ruolo collegato al servizio () CLI

1. (Facoltativo) Per visualizzare la descrizione corrente di un ruolo, utilizzate l'operazione AWS CLI forIAM. `get-role`

   Example

   ```
   $ aws iam get-role --role-name AmazonTimestreamInfluxDBServiceRolePolicy
   ```

   Utilizzate il nome del ruolo, non ilARN, per fare riferimento ai ruoli con le CLI operazioni. Ad esempio, se un ruolo presenta le seguenti caratteristicheARN:`arn:aws:iam::123456789012:role/myrole`, fate riferimento al ruolo come**`myrole`**.
2. Per aggiornare la descrizione di un ruolo collegato al servizio, usa l'operazione AWS CLI forIAM. `update-role-description`

   Linux e macOS

   ```
   $ aws iam update-role-description \
       --role-name AmazonTimestreamInfluxDBServiceRolePolicy \
       --description "new description"
   ```

   Windows

   ```
   $ aws iam update-role-description ^
       --role-name AmazonTimestreamInfluxDBServiceRolePolicy ^
       --description "new description"
   ```

Modifica della descrizione di un ruolo collegato a un servizio () IAM API

È possibile utilizzare il IAM API per modificare la descrizione di un ruolo collegato al servizio.

Per modificare la descrizione di un ruolo collegato al servizio () API

1. (Facoltativo) Per visualizzare la descrizione corrente di un ruolo, utilizzate l'operazione IAM API GetRole.

   Example

   ```
   https://iam.amazonaws.com/
      ?Action=GetRole
      &RoleName=AmazonTimestreamInfluxDBServiceRolePolicy
      &Version=2010-05-08
      &AUTHPARAMS
   ```

2. Per aggiornare la descrizione di un ruolo, usa l'IAMAPIoperazione UpdateRoleDescription.

   Example

   ```
   https://iam.amazonaws.com/
      ?Action=UpdateRoleDescription
      &RoleName=AmazonTimestreamInfluxDBServiceRolePolicy
      &Version=2010-05-08
      &Description="New description"
   ```

Eliminazione di un ruolo collegato al servizio per Amazon Timestream for InfluxDB

Se non è più necessario utilizzare una funzionalità o un servizio che richiede un ruolo collegato al servizio, ti consigliamo di eliminare il ruolo. In questo modo non sarà più presente un'entità non utilizzata che non viene monitorata e gestita attivamente. Tuttavia, è necessario effettuare la pulizia delle risorse associate al ruolo collegato ai servizi prima di poterlo eliminare.

Amazon Timestream for InfluxDB non elimina automaticamente il ruolo collegato al servizio.

Pulizia di un ruolo collegato ai servizi

Prima di poter eliminare un ruolo collegato IAM al servizio, verifica innanzitutto che al ruolo non siano associate risorse (cluster).

Per verificare se il ruolo collegato al servizio ha una sessione attiva nella console IAM

1. Accedi AWS Management Console e apri la IAM console all'indirizzo. https://console.aws.amazon.com/iam/

2. Nel riquadro di navigazione a sinistra della IAM console, scegli Ruoli. Quindi scegli il nome (non la casella di controllo) del AmazonTimestreamInflux DBServiceRolePolicy ruolo.
3. Nella pagina Summary (Riepilogo) per il ruolo selezionato, scegliere la scheda Access Advisor (Consulente accessi).
4. Nella scheda Access Advisor (Consulente accessi), esamina l'attività recente per il ruolo collegato ai servizi.

Eliminazione di un ruolo collegato al servizio (console) IAM

È possibile utilizzare la IAM console per eliminare un ruolo collegato al servizio.

Per eliminare un ruolo collegato ai servizi (console)

1. Accedi AWS Management Console e apri la IAM console all'indirizzo. https://console.aws.amazon.com/iam/
2. Nel riquadro di navigazione a sinistra della IAM console, scegli Ruoli. Quindi, seleziona la casella di controllo accanto al nome del ruolo che desideri eliminare, non il nome o la riga stessa.
3. In operazioni Role (Ruolo) nella parte superiore della pagina, seleziona Delete (Elimina) ruolo.
4. Nella pagina di conferma, esamina i dati dell'ultimo accesso al servizio, che mostrano l'ultima volta che ciascuno dei ruoli selezionati ha effettuato l'ultimo accesso a un AWS servizio. In questo modo potrai verificare se il ruolo è attualmente attivo. Se desideri procedere, seleziona Yes, Delete (Sì, elimina) per richiedere l'eliminazione del ruolo collegato ai servizi.
5. Guarda le notifiche della IAM console per monitorare lo stato di avanzamento dell'eliminazione del ruolo collegato al servizio. Poiché l'eliminazione del ruolo IAM collegato al servizio è asincrona, dopo aver inviato il ruolo per l'eliminazione, l'operazione di eliminazione può avere esito positivo o negativo. Se il task non viene eseguito correttamente, puoi scegliere View details (Visualizza dettagli) o View Resources (Visualizza risorse) dalle notifiche per capire perché l'eliminazione non è stata effettuata.

Eliminazione di un ruolo collegato al servizio () IAM CLI

È possibile utilizzare IAM le operazioni di AWS Command Line Interface per eliminare un ruolo collegato al servizio.

Per eliminare un ruolo collegato al servizio () CLI

1. Se non conosci il nome del ruolo collegato ai servizi da eliminare, inserisci il comando seguente: Questo comando elenca i ruoli e i relativi Amazon Resource Names (ARNs) nel tuo account.

   ```
   $ aws iam get-role --role-name role-name
   ```

   Usa il nome del ruolo, non ilARN, per fare riferimento ai ruoli con le CLI operazioni. Ad esempio, se un ruolo ha il ARN`arn:aws:iam::123456789012:role/myrole`, si fa riferimento al ruolo come**`myrole`**.

2. Poiché un ruolo collegato ai servizi non può essere eliminato se è in uso o se a esso sono associate delle risorse, occorre inviare una richiesta di eliminazione. Se queste condizioni non sono soddisfatte, la richiesta può essere rifiutata. Acquisisci il valore di `deletion-task-id`dalla risposta per controllare lo stato del task di eliminazione. Per inviare una richiesta di eliminazione per un ruolo collegato ai servizi, inserire quanto segue:

   ```
   $ aws iam delete-service-linked-role --role-name role-name
   ```

3. Inserire quanto segue per verificare lo stato del processo di eliminazione:

   ```
   $ aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
   ```

   Lo stato di un task di eliminazione può essere `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED`o `FAILED`. Se l'eliminazione non viene eseguita correttamente, la chiamata restituisce il motivo dell'errore per consentire all'utente di risolvere il problema.

Eliminazione di un ruolo collegato al servizio () IAM API

È possibile utilizzare il IAM API per eliminare un ruolo collegato al servizio.

Per eliminare un ruolo collegato al servizio () API

1. Per inviare una richiesta di cancellazione per un ruolo collegato al servizio, chiama DeleteServiceLinkedRole. Nella richiesta, specificare il nome del ruolo.

   Poiché un ruolo collegato ai servizi non può essere eliminato se è in uso o se a esso sono associate delle risorse, occorre inviare una richiesta di eliminazione. Se queste condizioni non

sono soddisfatte, la richiesta può essere rifiutata. Acquisisci il valore di `DeletionTaskId`dalla risposta per controllare lo stato del task di eliminazione.

2. Per verificare lo stato dell'eliminazione, chiama GetServiceLinkedRoleDeletionStatus. Nella richiesta, specificare il`DeletionTaskId`.

   Lo stato di un task di eliminazione può essere `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED`o `FAILED`. Se l'eliminazione non viene eseguita correttamente, la chiamata restituisce il motivo dell'errore per consentire all'utente di risolvere il problema.

Regioni supportate per i ruoli collegati al servizio Amazon Timestream for InfluxDB

Amazon Timestream for InfluxDB supporta l'utilizzo di ruoli collegati al servizio in tutte le regioni in cui il servizio è disponibile. Per ulteriori informazioni, consulta Endpoint del servizio AWS.

AWS politiche gestite per Amazon Timestream for InfluxDB

Per aggiungere autorizzazioni a utenti, gruppi e ruoli, è più facile utilizzare le politiche AWS gestite che scrivere le politiche da soli. Ci vogliono tempo ed esperienza per creare politiche gestite dai IAM clienti che forniscano al team solo le autorizzazioni di cui ha bisogno. Per iniziare rapidamente, puoi utilizzare le nostre politiche AWS gestite. Queste politiche coprono casi d'uso comuni e sono disponibili nel tuo AWS account. Per ulteriori informazioni sulle politiche AWS gestite, consulta le politiche AWS gestite nella Guida IAM per l'utente.

AWS i servizi mantengono e aggiornano le politiche AWS gestite. Non è possibile modificare le autorizzazioni nelle politiche AWS gestite. I servizi occasionalmente aggiungono altre autorizzazioni a una policy gestita da AWS per supportare nuove funzionalità. Questo tipo di aggiornamento interessa tutte le identità (utenti, gruppi e ruoli) a cui è collegata la policy. È più probabile che i servizi aggiornino una policy gestita da AWS quando viene avviata una nuova funzionalità o quando diventano disponibili nuove operazioni. I servizi non rimuovono le autorizzazioni da una policy AWS gestita, quindi gli aggiornamenti delle policy non comprometteranno le autorizzazioni esistenti.

Inoltre, AWS supporta politiche gestite per le funzioni lavorative che si estendono su più servizi. Ad esempio, la policy ReadOnlyAccess AWS gestita fornisce l'accesso in sola lettura a tutti i AWS servizi e le risorse. Quando un servizio lancia una nuova funzionalità, AWS aggiunge autorizzazioni di sola lettura per nuove operazioni e risorse. Per un elenco e le descrizioni delle politiche relative alle funzioni lavorative, consulta le politiche AWS gestite per le funzioni lavorative nella Guida per l'utente. IAM

AWS politica gestita: AmazonTimestreamInflux DBServiceRolePolicy

Non puoi allegare la politica AmazonTimestreamInflux DBServiceRolePolicy AWS gestita alle identità del tuo account. Questa politica fa parte del ruolo collegato al servizio AWS TimestreamforInflux DB. Questo ruolo consente al servizio di gestire le interfacce di rete e i gruppi di sicurezza nell'account.

Timestream for InfluxDB utilizza le autorizzazioni di questa politica per gestire EC2 i gruppi di sicurezza e le interfacce di rete. Ciò è necessario per gestire Timestream per le istanze DB di InfluxDB.

Per rivedere il formato di questa politica, consulta. JSON AmazonTimestreamInfluxDBServiceRolePolicy

AWS-politiche gestite per Amazon Timestream for InfluxDB

AWS affronta molti casi d'uso comuni fornendo IAM politiche autonome create e amministrate da. AWS Le policy gestite concedono le autorizzazioni necessarie per i casi di utilizzo comune in modo da non dover cercare quali sono le autorizzazioni richieste. Per ulteriori informazioni, vedere AWS Managed Policies nella Guida per l'IAMutente.

Le seguenti politiche AWS gestite, che puoi allegare agli utenti del tuo account, sono specifiche di Timestream for InfluxDB:

AmazonTimestreamInfluxDBFullAccess

Puoi allegare la `AmazonTimestreamInfluxDBFullAccess` policy alle tue identità. IAM Questa politica concede autorizzazioni amministrative che consentono l'accesso completo a tutte le risorse Timestream for InfluxDB.

Puoi anche creare IAM politiche personalizzate per consentire le autorizzazioni per le azioni di Amazon API Timestream for InfluxDB. Puoi allegare queste politiche personalizzate IAM agli utenti o ai gruppi che richiedono tali autorizzazioni.

Per rivedere il JSON formato di questa politica, consulta AmazonTimestreamInfluxDBFullAccess.

Timestream per gli aggiornamenti di InfluxDB alle politiche gestite AWS

Visualizza i dettagli sugli aggiornamenti alle politiche AWS gestite per Timestream for InfluxDB da quando questo servizio ha iniziato a tenere traccia di queste modifiche. Per avvisi automatici sulle modifiche a questa pagina, iscriviti al RSS feed nella pagina di cronologia dei documenti di Timestream for InfluxDB.

Modifica	Descrizione	Data
AmazonTimestreamInfluxDBFullAccess: aggiornamento a una policy esistente	È stata aggiunta l'`ec2:DescribeRouteTables` azione alla politica gestita esistente . `AmazonTimestreamInfluxDBFullAccess` Questa azione viene utilizzata per descrivere le tabelle dei percorsi	10/08/2024
AWS politica gestita: AmazonTimestreamInfluxDBServiceRolePolicy: nuova policy	Amazon Timestream per InfluxDB ha aggiunto una nuova policy che consente al servizio di gestire interfacce di rete e gruppi di sicurezza nel tuo account.	14/03/2024
AmazonTimestreamInfluxDBFullAccess: nuova policy	Amazon Timestream for InfluxDB ha aggiunto una nuova policy per fornire l'accesso amministrativo completo per creare, aggiornare, eliminare ed elencare istanze Amazon Timestream InfluxDB e creare ed elencare gruppi di parametri.	14/03/2024

Connessione a Timestream for InfluxDB tramite un endpoint VPC

Puoi connetterti direttamente a Timestream for InfluxDB tramite un endpoint di interfaccia privata nel tuo cloud privato virtuale (). VPC Quando utilizzi un VPC endpoint di interfaccia, la comunicazione tra te VPC e Timestream for InfluxDB viene condotta interamente all'interno della rete. AWS

Timestream for InfluxDB supporta gli endpoint Amazon Virtual Private Cloud (AmazonVPC) con tecnologia. AWS PrivateLink Ogni VPC endpoint è rappresentato da una o più interfacce di rete elastiche (ENIs) con indirizzi IP privati nelle sottoreti. VPC

L'VPCendpoint di interfaccia si connette VPC direttamente a Timestream for InfluxDB senza un gateway Internet, un dispositivo, una connessione o una connessione. NAT VPN AWS Direct Connect Le istanze presenti VPC non necessitano di indirizzi IP pubblici per comunicare con Timestream for InfluxDB.

Regioni

Timestream for InfluxDB supporta gli endpoint e VPC le policy degli VPC endpoint in tutti i paesi in cui è supportato Timestream for InfluxDB. Regioni AWS

Argomenti

- Considerazioni per Timestream for InfluxDB endpoint VPC
- Creazione di un endpoint per Timestream for InfluxDB VPC
- Connessione a un endpoint Timestream for InfluxDB VPC
- Controllo dell'accesso a un VPC endpoint
- Utilizzo di un VPC endpoint in una dichiarazione politica
- Registrazione dell'endpoint VPC

Considerazioni per Timestream for InfluxDB endpoint VPC

Prima di configurare un VPC endpoint di interfaccia per Timestream for InfluxDB, consulta l'argomento Proprietà e limitazioni dell'endpoint dell'interfaccia nella Guida.AWS PrivateLink

Il supporto di Timestream for InfluxDB per un endpoint include quanto segue. VPC

- Puoi usare il tuo VPC endpoint per chiamare tutte le operazioni di Timestream for InfluxDB dal tuo. API VPC
- Puoi utilizzare AWS CloudTrail i log per verificare l'utilizzo di Timestream for InfluxDB risorse tramite l'endpoint. VPC Per informazioni dettagliate, consultare Registrazione dell'endpoint VPC.

Creazione di un endpoint per Timestream for InfluxDB VPC

Puoi creare un VPC endpoint per Timestream for InfluxDB utilizzando la console VPC Amazon o Amazon. VPC API Per ulteriori informazioni, consulta la sezione Creazione di un endpoint di interfaccia nella Guida per l'utente di AWS PrivateLink .

- Per creare un VPC endpoint per Timestream for InfluxDB, usa il seguente nome di servizio:

  ```
  com.amazonaws.region.timestream-influxdb
  ```

  Ad esempio, nella regione Stati Uniti occidentali (Oregon) (`us-west-2`), il nome del servizio sarebbe:

  ```
  com.amazonaws.us-west-2.timestream-influxdb
  ```

Per semplificare l'utilizzo dell'VPCendpoint, puoi abilitare un nome privato per il tuo endpoint. DNS VPC Se selezioni l'opzione Enable DNS Name, il nome host standard di Timestream for InfluxDB DNS viene risolto sul tuo endpoint. VPC Ad esempio, si `https://timestream-influxdb.us-west-2.amazonaws.com` risolverebbe in un endpoint connesso al nome del servizio. VPC `com.amazonaws.us-west-2.timestream-influxdb`

Questa opzione semplifica l'utilizzo dell'VPCendpoint. Per impostazione predefinita, AWS CLI utilizza lo standard Timestream for InfluxDB DNS hostname, quindi non è necessario specificare l'endpoint nelle applicazioni e nei VPC comandi. AWS SDKs URL

Per ulteriori informazioni, consulta la sezione Accesso a un servizio tramite un endpoint di interfaccia nella Guida di AWS PrivateLink .

Connessione a un endpoint Timestream for InfluxDB VPC

È possibile connettersi a Timestream for InfluxDB tramite l'VPCendpoint utilizzando un, o. AWS SDK AWS CLI AWS Tools for PowerShell Per specificare l'VPCendpoint, usa il suo nome. DNS

Se hai abilitato i nomi host privati al momento della creazione dell'VPCendpoint, non è necessario specificare l'VPCendpoint URL nei CLI comandi o nella configurazione dell'applicazione. Il nome host standard di Timestream for DNS InfluxDB si risolve nel tuo endpoint. VPC Impostate AWS CLI e SDKs utilizzate questo nome host per impostazione predefinita, in modo da poter iniziare a utilizzare l'VPCendpoint per connettervi a un endpoint regionale di Timestream for InfluxDB senza modificare nulla negli script e nelle applicazioni.

Per utilizzare i nomi di host privati, gli attributi e del `enableDnsHostnames` tuo devono essere impostati su. `enableDnsSupport` VPC `true` Per impostare questi attributi, usa l'ModifyVpcAttributeoperazione. Per i dettagli, consulta Visualizza e aggiorna DNS gli attributi per i tuoi VPC nella Amazon VPC User Guide.

Controllo dell'accesso a un VPC endpoint

Per controllare l'accesso al tuo VPC endpoint per Timestream for InfluxDB, allega una VPC policy sugli endpoint al tuo endpoint. VPC La policy dell'endpoint determina se i responsabili possono utilizzare l'endpoint per chiamare le operazioni di Timestream for InfluxDB sulle risorse VPC Timestream for InfluxDB.

Puoi creare una policy per gli endpoint quando crei l'VPCendpoint e puoi modificare la policy degli endpoint in qualsiasi momento. VPC Utilizza la console VPC di gestione o le CreateVpcEndpointoperazioni. ModifyVpcEndpoint È inoltre possibile creare e modificare una policy per gli VPC endpoint utilizzando un AWS CloudFormation modello. Per informazioni sull'utilizzo della console di VPC gestione, consulta Creare un endpoint di interfaccia e Modificare un endpoint di interfaccia nella Guida.AWS PrivateLink

> **Note**
>
> Timestream for InfluxDB supporta VPC le politiche degli endpoint a partire da luglio 2020. VPCgli endpoint per Timestream for InfluxDB che sono stati creati prima di tale data hanno la policy di VPCendpoint predefinita, ma puoi modificarla in qualsiasi momento.

Argomenti

- Informazioni sulle politiche degli endpoint VPC
- Politica predefinita per VPC gli endpoint
- Creazione di una policy sugli endpoint VPC
- Visualizzazione di una policy sull'endpoint VPC

Informazioni sulle politiche degli endpoint VPC

Affinché una richiesta Timestream for InfluxDB che utilizza un VPC endpoint abbia esito positivo, il principale richiede le autorizzazioni da due fonti:

- Una IAMpolicy deve fornire l'autorizzazione principale per chiamare l'operazione sulla risorsa.

- Una policy sull'VPCendpoint deve fornire l'autorizzazione principale per utilizzare l'endpoint per effettuare la richiesta.

Politica predefinita per VPC gli endpoint

Ogni VPC endpoint ha una policy relativa agli VPC endpoint, ma non è necessario specificarla. Se non specifichi una policy, la policy di endpoint predefinita consente tutte le operazioni effettuate da tutte i principali su tutte le risorse dell'endpoint.

Tuttavia, per le risorse Timestream for InfluxDB, il principale deve disporre anche dell'autorizzazione a richiamare l'operazione da una IAMpolicy. Pertanto, in pratica, la politica predefinita afferma che se un principale è autorizzato a chiamare un'operazione su una risorsa, può richiamarla anche utilizzando l'endpoint.

```
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Principal": "*",
      "Resource": "*"
    }
  ]
}
```

Per consentire ai responsabili di utilizzare l'VPCendpoint solo per un sottoinsieme delle operazioni consentite, crea o aggiorna la policy dell'endpoint. VPC

Creazione di una policy sugli endpoint VPC

Una policy sull'VPCendpoint determina se un principale è autorizzato a utilizzare l'VPCendpoint per eseguire operazioni su una risorsa. Per le risorse Timestream for InfluxDB, il principale deve inoltre disporre del permesso di eseguire le operazioni previste da una policy, IAM

Ogni dichiarazione di policy VPC sugli endpoint richiede i seguenti elementi:

- Il principale che può eseguire operazioni.
- Le azioni che possono essere eseguite
- Le risorse sui cui si possono eseguire le azioni

La dichiarazione politica non specifica l'VPCendpoint. Si applica invece a qualsiasi VPC endpoint a cui è allegata la policy. Per ulteriori informazioni, consulta Controllare l'accesso ai servizi con VPC endpoint nella Amazon VPC User Guide.

AWS CloudTrail registra tutte le operazioni che utilizzano l'VPCendpoint.

Visualizzazione di una policy sull'endpoint VPC

Per visualizzare la policy degli VPC endpoint per un endpoint, utilizza la console di VPC gestione o l'operazione. DescribeVpcEndpoints

Il AWS CLI comando seguente ottiene la policy per l'endpoint con l'ID endpoint specificatoVPC.

Prima di eseguire questo comando, sostituisci l'ID endpoint dell'esempio con un ID valido del tuo account.

```
$ aws ec2 describe-vpc-endpoints \
--query 'VpcEndpoints[?VpcEndpointId==`vpc-endpoint-id`].[PolicyDocument]'
--output text
```

Utilizzo di un VPC endpoint in una dichiarazione politica

È possibile controllare l'accesso alle risorse e alle operazioni di Timestream for InfluxDB quando la richiesta proviene VPC o utilizza un endpoint. VPC A tale scopo, utilizza una delle seguenti chiavi di condizione globali in una politica. IAM

- Usa la chiave `aws:sourceVpce` condizionale per concedere o limitare l'accesso in base all'VPCendpoint.
- Usa la chiave `aws:sourceVpc` condizionale per concedere o limitare l'accesso in base a VPC quello che ospita l'endpoint privato.

> **Note**
>
> Fai attenzione quando crei policy e IAM policy chiave basate sul tuo VPC endpoint. Se una dichiarazione di policy richiede che le richieste provengano da un particolare VPC o da un VPC endpoint, le richieste provenienti da AWS servizi integrati che utilizzano una risorsa Timestream for InfluxDB per tuo conto potrebbero non riuscire.
> Inoltre, la chiave di `aws:sourceIP` condizione non è efficace quando la richiesta proviene da un VPCendpoint Amazon. Per limitare le richieste a un VPC endpoint, usa le chiavi di `aws:sourceVpc` condizione `aws:sourceVpce` o. Per ulteriori informazioni, consulta

Gestione delle identità e degli accessi per VPC endpoint e servizi VPC endpoint nella Guida.AWS PrivateLink

Puoi utilizzare queste chiavi di condizione globali per controllare l'accesso a operazioni come queste CreateDbInstanceche non dipendono da alcuna risorsa particolare.

Registrazione dell'endpoint VPC

AWS CloudTrail registra tutte le operazioni che utilizzano l'endpoint. VPC Quando una richiesta a Timestream for InfluxDB utilizza un VPC endpoint, l'ID dell'endpoint viene visualizzato nella voce di VPC registro che registra la richiesta.AWS CloudTrail Puoi utilizzare l'ID dell'endpoint per verificare l'uso del tuo endpoint Timestream for InfluxDB. VPC

Tuttavia, i tuoi CloudTrail log non includono le operazioni richieste dai responsabili in altri account o le richieste di operazioni di Timestream for InfluxDB sulle risorse e gli alias di Timestream for InfluxDB in altri account. Inoltre, per proteggere le vostreVPC, le richieste che vengono rifiutate da una policy sugli VPC endpoint, ma che altrimenti sarebbero state consentite, non vengono registrate in. AWS CloudTrail

Registrazione e monitoraggio in Timestream per InfluxDB

Il monitoraggio è una parte importante per mantenere l'affidabilità, la disponibilità e le prestazioni di Timestream for InfluxDB e le tue soluzioni. AWS È necessario raccogliere i dati di monitoraggio da tutte le parti della AWS soluzione in modo da poter eseguire più facilmente il debug di un errore multipunto, se si verifica uno. Tuttavia, prima di iniziare a monitorare Timestream for InfluxDB, è necessario creare un piano di monitoraggio che includa le risposte alle seguenti domande:

- Quali sono gli obiettivi del monitoraggio?
- Di quali risorse si intende eseguire il monitoraggio?
- Con quale frequenza sarà eseguito il monitoraggio di queste risorse?
- Quali strumenti di monitoraggio verranno utilizzati?
- Chi eseguirà i processi di monitoraggio?
- Chi deve ricevere una notifica quando si verifica un problema?

Il passaggio successivo consiste nello stabilire una linea di base per le normali prestazioni di Timestream for InfluxDB nel vostro ambiente, misurando le prestazioni in vari momenti e in diverse

condizioni di carico. Durante il monitoraggio di Timestream for InfluxDB, archivia i dati di monitoraggio storici in modo da poterli confrontare con i dati sulle prestazioni correnti, identificare i normali modelli di prestazioni e le anomalie delle prestazioni e ideare metodi per risolvere i problemi.

Per stabilire una baseline, devi monitorare almeno gli elementi seguenti:

- Errori di sistema, in modo da poter determinare se le richieste hanno generato un errore.

Argomenti

Strumenti di monitoraggio

AWS fornisce vari strumenti che è possibile utilizzare per monitorare Timestream for InfluxDB. Alcuni di questi strumenti possono essere configurati in modo che eseguano automaticamente il monitoraggio, mentre altri richiedono l'intervento manuale. Si consiglia di automatizzare il più possibile i processi di monitoraggio.

Argomenti

Strumenti di monitoraggio automatici

Puoi utilizzare i seguenti strumenti di monitoraggio automatizzato per guardare Timestream for InfluxDB e segnalare quando qualcosa non va:

- Amazon CloudWatch Alarms: monitora una singola metrica in un periodo di tempo specificato ed esegui una o più azioni in base al valore della metrica rispetto a una determinata soglia in diversi periodi di tempo. L'azione è una notifica inviata a un argomento di Amazon Simple Notification Service (AmazonSNS) o a una politica di Amazon EC2 Auto Scaling. CloudWatch gli allarmi non richiamano azioni semplicemente perché si trovano in uno stato particolare; lo stato deve essere cambiato e mantenuto per un determinato numero di periodi. Per ulteriori informazioni, consulta Monitoraggio con Amazon CloudWatch.

Strumenti di monitoraggio manuali

Un'altra parte importante del monitoraggio di Timestream for InfluxDB riguarda il monitoraggio manuale degli elementi che gli allarmi non coprono. CloudWatch Il Timestream per InfluxDB e altri AWS Management Console dashboard CloudWatch forniscono una Trusted Advisor visione dello stato dell'ambiente. at-a-glance AWS

- La CloudWatch home page mostra quanto segue:
  - Stato e allarmi attuali
  - Grafici degli allarmi e delle risorse
  - Stato di integrità dei servizi

  Inoltre, è possibile utilizzare CloudWatch per effettuare le seguenti operazioni:
  - Crea pannelli di controllo personalizzati per monitorare i servizi di interesse.
  - Crea grafici dei dati dei parametri per la risoluzione di problemi e il rilevamento di tendenze.
  - Cerca e sfoglia tutte le metriche AWS delle tue risorse
  - Crea e modifica gli allarmi per ricevere le notifiche dei problemi.

Registrazione del timestream per le chiamate InfluxDB con API AWS CloudTrail

Timestream for InfluxDB è integrato con AWS CloudTrail, un servizio che fornisce una registrazione delle azioni intraprese da un utente, ruolo o servizio in Timestream for InfluxDB. AWS CloudTrail acquisisce le chiamate Data Definition Language () DDL per Timestream for InfluxDB come eventi. API Le chiamate acquisite includono chiamate dalla console Timestream for InfluxDB e chiamate di codice a Timestream for InfluxDB operations. API Se crei un trail, puoi abilitare la distribuzione continua di CloudTrail eventi a un bucket Amazon Simple Storage Service (Amazon S3), inclusi gli eventi per Timestream for InfluxDB. Se non configuri un percorso, puoi comunque visualizzare gli eventi più recenti sulla console nella cronologia degli eventi. CloudTrail Utilizzando le informazioni raccolte da CloudTrail, puoi determinare la richiesta effettuata a Timestream for InfluxDB, l'indirizzo IP da cui è stata effettuata la richiesta, chi ha effettuato la richiesta, quando è stata effettuata e dettagli aggiuntivi.

Per ulteriori informazioni CloudTrail, consulta la Guida per l'utente.AWS CloudTrail

Informazioni su Timestream for InfluxDB in CloudTrail

CloudTrail è abilitato sul tuo AWS account al momento della creazione dell'account. Quando si verifica un'attività in Timestream for InfluxDB, tale attività viene registrata in un CloudTrail evento insieme ad altri eventi di AWS servizio nella cronologia degli eventi. È possibile visualizzare, cercare e scaricare gli eventi recenti nell'account AWS . Per ulteriori informazioni, consulta Visualizzazione degli eventi con la cronologia degli eventi. CloudTrail

Per una registrazione continua degli eventi nel tuo AWS account, inclusi gli eventi per Timestream for InfluxDB, crea un percorso. Un trail consente di CloudTrail inviare file di log a un bucket Amazon S3. Per impostazione predefinita, quando crei un percorso nella console, il percorso si applica a tutte le AWS regioni. Il trail registra gli eventi di tutte le regioni della AWS partizione e consegna i file di log al bucket Amazon S3 specificato. Inoltre, puoi configurare altri AWS servizi per analizzare ulteriormente e agire in base ai dati sugli eventi raccolti nei log. CloudTrail

Per ulteriori informazioni, consulta gli argomenti seguenti nella Guida per l'utente di AWS CloudTrail :

- Panoramica della creazione di un trail
- CloudTrail Servizi e integrazioni supportati
- Configurazione di Amazon SNS Notifications per CloudTrail
- Ricezione di file di CloudTrail registro da più regioni
- Ricezione di file di CloudTrail registro da più account
- Registrazione degli eventi relativi ai dati

Ogni evento o voce di log contiene informazioni sull'utente che ha generato la richiesta. Le informazioni di identità consentono di determinare quanto segue:

- Se la richiesta è stata effettuata con credenziali utente root o AWS Identity and Access Management (IAM)
- Se la richiesta è stata effettuata con le credenziali di sicurezza temporanee per un ruolo o un utente federato.
- Se la richiesta è stata effettuata da un altro AWS servizio

Per ulteriori informazioni, consulta l'CloudTrail userIdentityelemento.

Convalida della conformità per Amazon Timestream for InfluxDB

I revisori di terze parti valutano la sicurezza e la conformità di Amazon Timestream for InfluxDB come parte di più programmi di conformità. AWS Questi sono i seguenti:

- GDPR
- HIPAA
- PCI
- SOC

Per sapere se un Servizio AWS programma rientra nell'ambito di specifici programmi di conformità, consulta Servizi AWS la sezione Scope by Compliance Program Servizi AWS e scegli il programma di conformità che ti interessa. Per informazioni generali, consulta Programmi di AWS conformità Programmi di .

È possibile scaricare report di audit di terze parti utilizzando AWS Artifact. Per ulteriori informazioni, consulta Scaricamento dei report in AWS Artifact .

La vostra responsabilità di conformità durante l'utilizzo Servizi AWS è determinata dalla sensibilità dei dati, dagli obiettivi di conformità dell'azienda e dalle leggi e dai regolamenti applicabili. AWS fornisce le seguenti risorse per contribuire alla conformità:

- Guide introduttive su sicurezza e conformità: queste guide all'implementazione illustrano considerazioni sull'architettura e forniscono passaggi per implementare ambienti di base incentrati sulla AWS sicurezza e la conformità.
- Architettura per la HIPAA sicurezza e la conformità su Amazon Web Services: questo white paper descrive in che modo le aziende possono utilizzare AWS per creare applicazioni idonee. HIPAA

> **Note**
>
> Non tutte sono idonee. Servizi AWS HIPAA Per ulteriori informazioni, consulta la Guida ai servizi HIPAA idonei.

- AWS Risorse per AWS per la conformità: questa raccolta di cartelle di lavoro e guide potrebbe riguardare il tuo settore e la tua località.
- AWS Guide alla conformità dei clienti: comprendi il modello di responsabilità condivisa attraverso la lente della conformità. Le guide riassumono le migliori pratiche per la protezione Servizi AWS e

mappano le linee guida per i controlli di sicurezza su più framework (tra cui il National Institute of Standards and Technology (NIST), il Payment Card Industry Security Standards Council (PCI) e l'International Organization for Standardization ()). ISO

- Evaluating Resources with Rules nella Guida per gli AWS Config sviluppatori: il AWS Config servizio valuta la conformità delle configurazioni delle risorse alle pratiche interne, alle linee guida del settore e alle normative.
- AWS Security Hub— Ciò Servizio AWS fornisce una visione completa dello stato di sicurezza interno. AWS La Centrale di sicurezza utilizza i controlli di sicurezza per valutare le risorse AWS e verificare la conformità agli standard e alle best practice del settore della sicurezza. Per un elenco dei servizi e dei controlli supportati, consulta la pagina Documentazione di riferimento sui controlli della Centrale di sicurezza.
- Amazon GuardDuty: Servizio AWS rileva potenziali minacce ai tuoi carichi di lavoro Account AWS, ai contenitori e ai dati monitorando l'ambiente alla ricerca di attività sospette e dannose. GuardDuty può aiutarti a soddisfare vari requisiti di conformità, ad esempio PCI DSS soddisfacendo i requisiti di rilevamento delle intrusioni imposti da determinati framework di conformità.
- AWS Audit Manager— Ciò Servizio AWS consente di verificare continuamente AWS l'utilizzo per semplificare la gestione del rischio e la conformità alle normative e agli standard di settore.

Resilienza in Amazon Timestream per InfluxDB

L'infrastruttura AWS globale è costruita attorno a regioni e zone di disponibilità. AWS AWS Le regioni forniscono più zone di disponibilità fisicamente separate e isolate, collegate con reti a bassa latenza, ad alto throughput e altamente ridondanti. Con le zone di disponibilità, puoi progettare e gestire applicazioni e database che eseguono automaticamente il failover tra zone di disponibilità senza interruzioni. Le zone di disponibilità sono più disponibili, tolleranti ai guasti e scalabili rispetto alle infrastrutture a data center singolo o multiplo tradizionali.

Per ulteriori informazioni su AWS regioni e zone di disponibilità, consulta Global Infrastructure.AWS

Amazon Timestream for InfluxDB esegue periodicamente backup interni e li conserva per 24 ore per supportare la disponibilità e la durabilità. Le istantanee vengono scattate durante le eliminazioni e conservate per 30 giorni per supportare i ripristini. Per accedervi o utilizzarli, invia un ticket all'assistenza.AWS

Puoi creare la tua istanza con funzionalità di ripristino Multi-AZ. Per ulteriori informazioni, consulta Implementazioni di istanze DB Multi-AZ.

Sicurezza dell'infrastruttura in Amazon Timestream per InfluxDB

In quanto servizio gestito, Amazon Timestream for InfluxDB è protetto dalle procedure di sicurezza di rete globali descritte AWS nel white paper Amazon Web Services: Overview of Security Processes.

Utilizzi le API chiamate AWS pubblicate sul piano di controllo per accedere a Timestream for InfluxDB attraverso la rete. Per ulteriori informazioni, consulta Piani di controllo e piani dati. I client devono supportare Transport Layer Security (TLS) 1.2 o versione successiva. Consigliamo TLS 1.2 o 1.3. I client devono inoltre supportare suite di crittografia con Perfect Forward Secrecy (PFS) come Ephemeral Diffie-Hellman () o Elliptic Curve Ephemeral Diffie-Hellman (). DHE ECDHE La maggior parte dei sistemi moderni, come Java 7 e versioni successive, supporta tali modalità.

Inoltre, le richieste devono essere firmate utilizzando un ID chiave di accesso e una chiave di accesso segreta associata a un principale. IAM O puoi utilizzare AWS Security Token Service (AWS STS) per generare credenziali di sicurezza temporanee per sottoscrivere le richieste.

Timestream for InfluxDB è progettato in modo tale che il traffico sia isolato AWS nella regione specifica in cui risiede l'istanza di Timestream for InfluxDB.

Gruppi di sicurezza

I gruppi di sicurezza controllano l'accesso che il traffico ha in entrata e in uscita di un'istanza database. Per impostazione predefinita, l'accesso alla rete è disattivato per un'istanza database. Puoi specificare delle norme in un gruppo di sicurezza che consente l'accesso da un intervallo di indirizzi IP, dalla porta o da un gruppo di sicurezza. Una volta configurate le norme per l'ingresso, si applicano le stesse norme a tutte le istanze database che sono associate a tale gruppo di sicurezza.

Per ulteriori informazioni, consulta Controllo dell'accesso a un'istanza DB in un VPC.

Analisi della configurazione e della vulnerabilità in Timestream for InfluxDB

La configurazione e i controlli IT sono una responsabilità condivisa tra voi e voi, nostri clienti. AWS Per ulteriori informazioni, consulta il modello di responsabilità AWS condivisa. Oltre al modello di responsabilità condivisa, gli utenti di Timestream for InfluxDB devono essere consapevoli di quanto segue:

- È responsabilità del cliente applicare patch alle applicazioni client con le relative dipendenze lato client.
- I clienti dovrebbero prendere in considerazione i test di penetrazione, se appropriato (vedi penetration-testing/.) https://aws.amazon.com/security/

Risposta agli incidenti in Timestream for InfluxDB

Gli incidenti del servizio Amazon Timestream for InfluxDB sono segnalati nella Personal Health Dashboard. Puoi saperne di più sulla dashboard e qui. AWS Health

Timestream for InfluxDB supporta il reporting utilizzando. AWS CloudTrail Per ulteriori informazioni, consulta Registrazione del timestream per le chiamate InfluxDB con API AWS CloudTrail.

Amazon Timestream per API InfluxDB e endpoint di interfaccia () VPC AWS PrivateLink

Puoi stabilire una connessione privata tra i tuoi endpoint del API piano di controllo Amazon Timestream for InfluxDB VPC e Amazon Timestream for InfluxDB creando un endpoint di interfaccia. VPC Gli endpoint dell'interfaccia sono alimentati da. AWS PrivateLink AWS PrivateLink consente di accedere in modo privato alle operazioni di Amazon Timestream API for InfluxDB senza un NAT gateway Internet, un dispositivoVPN, una connessione o una connessione Direct Connect. AWS

Le istanze nel tuo VPC non hanno bisogno di indirizzi IP pubblici per comunicare con gli endpoint Amazon API Timestream for InfluxDB. Inoltre, le tue istanze non necessitano di indirizzi IP pubblici per utilizzare nessuno dei Timestream disponibili per le operazioni InfluxDB. API Il traffico tra te VPC e Amazon Timestream for InfluxDB non esce dalla rete Amazon. Ogni endpoint di interfaccia è rappresentato da una o più interfacce di rete elastiche nelle sottoreti. Per ulteriori informazioni sulle interfacce di rete elastiche, consulta Interfacce di rete elastiche nella Amazon EC2 User Guide.

- Per ulteriori informazioni sugli VPC endpoint, consulta Interface VPC endpoints (AWS PrivateLink) nella Amazon VPC User Guide.
- Per ulteriori informazioni sulle operazioni Timestream for InfluxDB, consulta Timestream for InfluxDB API operations. API

Dopo aver creato un endpoint di interfaccia, se abiliti i DNS nomi di host privati per l'VPCendpoint, l'endpoint Timestream for InfluxDB predefinito (https://timestream-influxb).*Region*VPC.amazonaws.com) si risolve sul tuo endpoint. Se non abiliti DNS i nomi host privati, Amazon VPC fornisce un nome di DNS endpoint che puoi utilizzare nel seguente formato:

```
VPC_Endpoint_ID.timestream-influxb.Region.vpce.amazonaws.com
```

Per ulteriori informazioni, consulta Interface VPC Endpoints (AWS PrivateLink) nella Amazon VPC User Guide. Timestream for InfluxDB supporta l'effettuazione di chiamate a tutte le sue azioni all'interno del tuo. API VPC

> **Note**
>
> DNSI nomi di host privati possono essere abilitati per un solo endpoint in. VPC VPC Se si desidera creare un VPC endpoint aggiuntivo, il DNS nome host privato deve essere disabilitato.

Considerazioni per gli endpoint VPC

Prima di configurare un endpoint di interfaccia per gli VPC endpoint Amazon Timestream API for InfluxDB, assicurati di esaminare le proprietà e le limitazioni degli endpoint dell'interfaccia nella Amazon User Guide. VPC Tutte le API operazioni di Timestream for InfluxDB rilevanti per la gestione delle risorse di Amazon Timestream for InfluxDB sono disponibili da te. VPC AWS PrivateLink VPCLe politiche degli endpoint sono supportate per gli endpoint Timestream for InfluxDB. API Per impostazione predefinita, l'accesso completo alle operazioni di Timestream for InfluxDB è consentito tramite l'endpoint. API Per ulteriori informazioni, consulta Controllare l'accesso ai servizi con VPC endpoint nella Amazon VPC User Guide.

Creazione di un VPC endpoint di interfaccia per Timestream for InfluxDB API

Puoi creare un VPC endpoint per Amazon Timestream API for InfluxDB utilizzando la console Amazon o VPC il. AWS CLI Per ulteriori informazioni, consulta la sezione Creazione di un endpoint di interfaccia nella Amazon VPC User Guide.

Dopo aver creato un VPC endpoint di interfaccia, puoi abilitare i nomi DNS host privati per l'endpoint. Quando lo fai, l'endpoint Amazon Timestream for InfluxDB predefinito (https://timestream-influxb.*Region*.amazonaws.com) si risolve nel tuo endpoint. VPC Per ulteriori informazioni, consulta Accedere a un servizio tramite un endpoint di interfaccia nella Amazon VPC User Guide.

Creazione di una policy VPC sugli endpoint per Amazon Timestream for InfluxDB API

Puoi allegare una policy sugli endpoint al tuo VPC endpoint che controlli l'accesso a Timestream for InfluxDB. API La policy specifica quanto segue:

- Il principale che può eseguire azioni.

- Le azioni che possono essere eseguite.
- Le risorse sui cui si possono eseguire azioni.

Per ulteriori informazioni, consulta Controllare l'accesso ai servizi con VPC endpoint nella Amazon VPC User Guide.

Example VPCpolitica degli endpoint per le azioni di Timestream for InfluxDB API

Di seguito è riportato un esempio di policy sugli endpoint per Timestream for InfluxDB. API Se collegata a un endpoint, questa politica consente l'accesso alle azioni Timestream for InfluxDB elencate per tutti i principali su tutte le risorse. API

```
{
 "Statement": [{
  "Principal": "*",
  "Effect": "Allow",
  "Action": [
   "timestream-influxb:CreateDbInstance",
   "timestream-influxb:UpdateDbInstance"
  ],
  "Resource": "*"
 }]
}
```

Example VPCpolitica dell'endpoint che nega tutti gli accessi da un account specificato AWS

La seguente politica sugli VPC endpoint nega l'account AWS *123456789012* tutti gli accessi alle risorse tramite l'endpoint. La policy consente tutte le operazioni da altri account.

```
{
 "Statement": [{
   "Action": "*",
   "Effect": "Allow",
   "Resource": "*",
   "Principal": "*"
  },
  {
   "Action": "*",
   "Effect": "Deny",
   "Resource": "*",
   "Principal": {
```

```
    "AWS": [
     "123456789012"
    ]
   }
  }
 ]
}
```

Le migliori pratiche di sicurezza per Timestream for InfluxDB

Amazon Timestream per InfluxDB offre una serie di funzionalità di sicurezza da considerare durante lo sviluppo e l'implementazione delle proprie politiche di sicurezza. Le seguenti best practice sono linee guida generali e non rappresentano una soluzione di sicurezza completa. Poiché queste best practice potrebbero non essere appropriate o sufficienti per l'ambiente, gestiscile come considerazioni utili anziché prescrizioni.

Implementazione dell'accesso con privilegi minimi

Quando concedi le autorizzazioni, sei tu a decidere chi ottiene quali autorizzazioni per quali risorse di Timestream for InfluxDB. È possibile abilitare operazioni specifiche che si desidera consentire su tali risorse. Pertanto è necessario concedere solo le autorizzazioni necessarie per eseguire un'attività. L'implementazione dell'accesso con privilegi minimi è fondamentale per ridurre i rischi di sicurezza e l'impatto risultante da errori o intenzioni dannose.

Usa i ruoli IAM

Le applicazioni Producer e Client devono disporre di credenziali valide per accedere a Timestream for InfluxDB DB. Non è necessario archiviare AWS le credenziali direttamente in un'applicazione client o in un bucket Amazon S3. Si tratta di credenziali a lungo termine che non vengono automaticamente ruotate e potrebbero avere un impatto aziendale significativo se vengono compromesse.

Dovreste invece utilizzare un IAM ruolo per gestire le credenziali temporanee per le vostre applicazioni di produzione e client per accedere a Timestream for InfluxDB istanze DB. Quando utilizzi un ruolo, non devi necessariamente usare credenziali a lungo termine (ad esempio, nome utente e password o chiavi di accesso) per accedere ad altre risorse.

Per ulteriori informazioni, consulta i seguenti argomenti nella Guida per l'utente: IAM

- IAMRuoli
- Scenari comuni per ruoli: utenti, applicazioni e servizi

Usa gli account AWS Identity and Access Management (IAM) per controllare l'accesso alle operazioni di Amazon Timestream for API InfluxDB, in particolare le operazioni che creano, modificano o eliminano le risorse Amazon Timestream for InfluxDB. Tali risorse includono istanze DB, gruppi di sicurezza e gruppi di parametri.

- Crea un utente individuale per ogni persona che gestisce le risorse Amazon Timestream for InfluxDB, incluso te stesso. Non utilizzare le credenziali di AWS root per gestire le risorse Amazon Timestream for InfluxDB.
- Assegna a ciascun utente un set minimo di autorizzazioni richieste per eseguire le proprie mansioni.
- Usa IAM i gruppi per gestire efficacemente le autorizzazioni per più utenti.
- Ruota periodicamente le credenziali IAM.
- Configura AWS Secrets Manager per ruotare automaticamente i segreti per Amazon Timestream for InfluxDB. Per ulteriori informazioni, consulta Rotazione dei AWS segreti di Secrets Manager nella Guida per l'utente di AWS Secrets Manager. È inoltre possibile recuperare le credenziali da AWS Secrets Manager a livello di codice. Per ulteriori informazioni, vedere Recupero del valore segreto nella Guida per l'utente di AWS Secrets Manager.
- Proteggi il tuo Timestream per i token Influx di InfluxDB utilizzando. API APIgettoni

Implementazione della crittografia lato server in risorse dipendenti

I dati inattivi e i dati in transito possono essere crittografati in Timestream for InfluxDB. Per ulteriori informazioni, consulta Crittografia in transito.

Utilizzare per monitorare le chiamate CloudTrail API

Timestream for InfluxDB è integrato con AWS CloudTrail, un servizio che fornisce un registro delle azioni intraprese da un utente, ruolo o AWS servizio in Timestream for InfluxDB.

Utilizzando le informazioni raccolte da CloudTrail, è possibile determinare la richiesta effettuata a Timestream for InfluxDB, l'indirizzo IP da cui è stata effettuata la richiesta, chi ha effettuato la richiesta, quando è stata effettuata e dettagli aggiuntivi.

Per ulteriori informazioni, consulta the section called “Registrazione del timestream per le chiamate con LiveAnalytics API AWS CloudTrail”.

Amazon Timestream per InfluxDB supporta gli eventi del piano di controllo, ma non il CloudTrail piano dati. Per ulteriori informazioni, consulta Piani di controllo e piani dati.

Public accessibility (Accesso pubblico)

Quando avvii un'istanza DB all'interno di un cloud privato virtuale (VPC) basato sul VPC servizio Amazon, puoi attivare o disattivare l'accessibilità pubblica per quell'istanza DB. Per indicare se l'istanza DB che crei ha un DNS nome che si risolve in un indirizzo IP pubblico, utilizzi il parametro Public accessibility. Utilizzando questo parametro, è possibile indicare se è disponibile l'accesso pubblico all'istanza DB

Se la tua istanza DB si trova in un database VPC ma non è accessibile pubblicamente, puoi anche utilizzare una AWS Site-to-Site VPN connessione o una connessione AWS Direct Connect per accedervi da una rete privata.

Se la tua istanza DB è accessibile al pubblico, assicurati di adottare misure per prevenire o contribuire a mitigare le minacce legate alla negazione del servizio. Per ulteriori informazioni, consulta Introduzione agli attacchi Denial of Service e Protezione delle reti.

APIriferimento

Per un elenco completo e i dettagli di Amazon Timestream for InfluxDBAPIs, consulta Amazon Timestream for InfluxDB. APIs

Per i codici di errore comuni a tutti i AWS servizi, consulta la sezione AWS Support.

Cronologia dei documenti

Modifica	Descrizione	Data
Aggiornamento solo della documentazione	È stato aggiornato l'argomento Quote per separare le quote predefinite dai limiti di sistema.	22 ottobre 2024
Amazon Timestream ora supporta le analisi delle query	Timestream ora include il supporto per la funzionalità Query Insights che ti aiuta a ottimizzare le query, migliorarne le prestazioni e ridurre i costi.	22 ottobre 2024

Amazon Timestream for InfluxDB si aggiorna a una policy esistente.	Amazon Timestream per InfluxDB ha aggiunto `ec2:DescribeRouteTables` l'azione alla politica gestita `AmazonTimestreamInfluxDBFullAccess` esistente per descrivere le tabelle di routing.	8 ottobre 2024
`AmazonTimestreamReadOnlyAccess` — Aggiornamento a una politica esistente	Timestream for LiveAnalytics ha aggiunto l'`DescribeAccountSettings` autorizzazione alla politica `AmazonTimestreamReadOnlyAccess` gestita per la descrizione Account AWS delle impostazioni.	3 giugno 2024
Amazon Timestream LiveAnalytics per ora supporta le unità di calcolo Timestream () TCUs	Amazon Timestream LiveAnalytics per ora include il supporto per Timestream Compute Units TCUs () per misurare la capacità di calcolo allocata per le tue esigenze di query.	29 aprile 2024

Nuove politiche aggiunte	Amazon Timestream per InfluxDB ha aggiunto due nuove politiche: una che consente al servizio di gestire le interfacce di rete e i gruppi di sicurezza nel tuo account. Per ulteriori informazioni, consulta. AmazonTimestreamInfluxDBServiceRolePolicy Un altro che fornisce l'accesso amministrativo completo per creare, aggiornare, eliminare ed elencare istanze Amazon Timestream InfluxDB e creare ed elencare gruppi di parametri. Per ulteriori informazioni, consulta. AmazonTimestreamInfluxDBFullAccess	14 marzo 2024
Amazon Timestream per InfluxDB è ora disponibile a livello generale.	Questa documentazione riguarda la versione iniziale di Amazon Timestream per InfluxDB.	14 marzo 2024

Gli eventi di Amazon Timestream LiveAnalytics for Query sono disponibili in AWS CloudTrail	Amazon Timestream LiveAnalytics per ora pubblica API gli eventi dei dati di Query su. AWS CloudTrail I clienti possono controllare tutte le API richieste di Query effettuat e nei propri AWS account e visualizzare informazioni come l'IAMutente/ruolo che ha effettuato la richiesta, quando è stata effettuata la richiesta, quali database e tabelle sono state interrogate e l'ID di query della richiesta.	12 settembre 2023
Amazon Timestream per LiveAnalytics UNLOAD	Amazon Timestream LiveAnalytics per ora UNLOAD supporta l'esportazione dei risultati delle query in S3.	12 maggio 2023
Amazon Timestream LiveAnalytics per l'aggiorn amento a una politica esistente.	Autorizzazioni di caricamento in batch aggiunte a una policy gestita.	24 febbraio 2023
Amazon Timestream LiveAnalytics per il caricamen to in batch.	Amazon Timestream LiveAnalytics per ora supporta la funzionalità di caricamento in batch.	24 febbraio 2023
Amazon Timestream LiveAnalytics per ora supporta. AWS Backup	Amazon Timestream LiveAnalytics per ora supporta. AWS Backup	14 dicembre 2022

Amazon Timestream LiveAnalytics per gli aggiornamenti delle politiche gestite AWS	Nuove informazioni sulle politiche AWS gestite e Amazon Timestream LiveAnalytics for, inclusi gli aggiornamenti alle politiche gestite esistenti.	29 novembre 2021
Amazon Timestream LiveAnalytics per il supporto delle query pianificate	Amazon Timestream LiveAnalytics per ora supporta l'esecuzione di una query per tuo conto, in base a una pianificazione.	29 novembre 2021
Amazon Timestream LiveAnalytics for supporta l'archiviazione magnetica.	Amazon Timestream LiveAnalytics per ora supporta l'utilizzo dello storage magnetico per le scritture da tavolo.	29 novembre 2021
Amazon Timestream per record multimisura LiveAnalytics .	Amazon Timestream LiveAnalytics per ora supporta un formato più compatto per l'archiviazione dei dati delle serie temporali.	29 novembre 2021
Amazon Timestream LiveAnalytics per gli aggiornamenti delle politiche gestite AWS	Nuove informazioni sulle politiche AWS gestite e Amazon Timestream LiveAnalytics for, inclusi gli aggiornamenti alle politiche gestite esistenti.	24 maggio 2021

Amazon Timestream LiveAnalytics for è ora disponibile nella regione Europa (Francoforte).	Amazon Timestream LiveAnalytics for è ora disponibile a livello generale nella regione Europa (Francoforte) (). `eu-central-1`	23 Aprile 2021
Amazon Timestream LiveAnalytics per noi ora VPC supporta gli endpoint ().AWS PrivateLink	Amazon Timestream LiveAnalytics per ora supporta l'uso VPC degli endpoint ().AWS PrivateLink	23 marzo 2021
Amazon Timestream ora supporta le query tra tabelle.	Puoi utilizzare Amazon Timestream LiveAnalytics per eseguire query tra tabelle.	10 febbraio 2021
Amazon Timestream LiveAnalytics per ora supporta statistiche avanzate sull'esecuzione delle query.	Amazon Timestream LiveAnalytics per ora supporta statistiche avanzate sull'esecuzione delle query, come la quantità di dati scansionati.	10 febbraio 2021
Amazon Timestream LiveAnalytics per ora supporta funzioni avanzate di serie temporali.	Puoi utilizzare Amazon Timestream LiveAnalytics per SQL eseguire query con funzioni avanzate di serie temporali, come derivati, integrali e correlazioni.	10 febbraio 2021
Amazon Timestream LiveAnalytics per HIPAA ora è conformeISO. PCI	Ora puoi utilizzare Amazon Timestream LiveAnalytics per carichi di lavoro che HIPAA richiedono un'infrastruttura conformeISO. PCI	27 gennaio 2021

Amazon Timestream LiveAnalytics per ora supporta telegraf e grafana open source.	Ora puoi usare Telegraf, l'agente server open source basato su plug-in per la raccolta e il reporting dei parametri, e Grafana, la piattaforma di analisi e monitoraggio open source per i database, con Amazon Timestream per. LiveAnalytics	25 novembre 2020
Amazon Timestream LiveAnalytics for è ora disponibile a livello generale.	Questa documentazione riguarda la versione iniziale di Amazon LiveAnalytics Timestream per.	30 settembre 2020

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.