aws

Guia do desenvolvedor de SQL

Guia do Desenvolvedor de Amazon Kinesis Data Analytics para aplicativos SQL

Guia do Desenvolvedor de Amazon Kinesis Data Analytics para aplicativos SQL: Guia do desenvolvedor de SQL

Table of Contents

Para novos projetos, recomendamos que você use o novo Managed Service for Apache Flink Studio em vez do Kinesis Data Analytics para aplicativos SQL. O Managed Service for Apache Flink Studio combina facilidade de uso com recursos analíticos avançados, permitindo que você crie aplicativos sofisticados de processamento de stream em minutos.

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.

O que é o Kinesis Data Analytics para aplicativos SQL?

Com o Amazon Kinesis Data Analytics para aplicativos SQL, você pode processar e analisar dados de streaming usando SQL padrão. O serviço permite que você crie e execute rapidamente um código SQL sofisticado com base nas origens de streaming para executar análises por tempo, alimentar painéis em tempo real e criar métricas em tempo real.

Para começar a usar o Kinesis Data Analytics, crie um aplicativo Kinesis Data Analytics que leia e processe continuamente o fluxo de dados. O serviço suporta a ingestão de dados das fontes de streaming Amazon Kinesis Data Streams e Amazon Data Firehose. Em seguida, você cria o código SQL usando o editor interativo e o testa com dados de streaming ao vivo. Você também pode configurar destinos para os quais o Kinesis Data Analytics enviará os resultados.

O Kinesis Data Analytics oferece suporte ao Amazon Data Firehose (Amazon S3, Amazon Redshift, Amazon Service e Splunk) e ao OpenSearch AWS LambdaAmazon Kinesis Data Streams como destinos.

Quando devo usar o Amazon Kinesis Data Analytics?

O Amazon Kinesis Data Analytics permite que você crie rapidamente o código SQL que lê, processa e armazena continuamente dados em tempo quase real. Usando consultas SQL padrão nos dados em streaming, você pode criar aplicativos que transformam e fornecem informações sobre seus dados. Veja a seguir alguns cenários de exemplo de uso do Kinesis Data Analytics:

- Gerar análise por tempo: você pode calcular métricas ao longo do tempo e, em seguida, fazer streaming de valores para o Amazon S3 ou o Amazon Redshift por meio de um stream de entrega de dados do Kinesis.
- Alimentar painéis em tempo real – Você pode enviar resultados de dados de streaming agregados e processados downstream para alimentar painéis em tempo real.
- Criar métricas em tempo real – Você pode criar métricas e triggers personalizados para uso em monitoramento em tempo real, notificações e alarmes.

Para obter mais informações sobre os elementos da linguagem SQL compatíveis com o Kinesis Data Analytics, consulte Referência SQL do Amazon Kinesis Data Analytics.

Você é um usuário iniciante do Amazon Kinesis Data Analytics?

Se você estiver usando o Amazon Kinesis Data Analytics pela primeira vez, recomendamos que leia as seções a seguir nesta ordem:

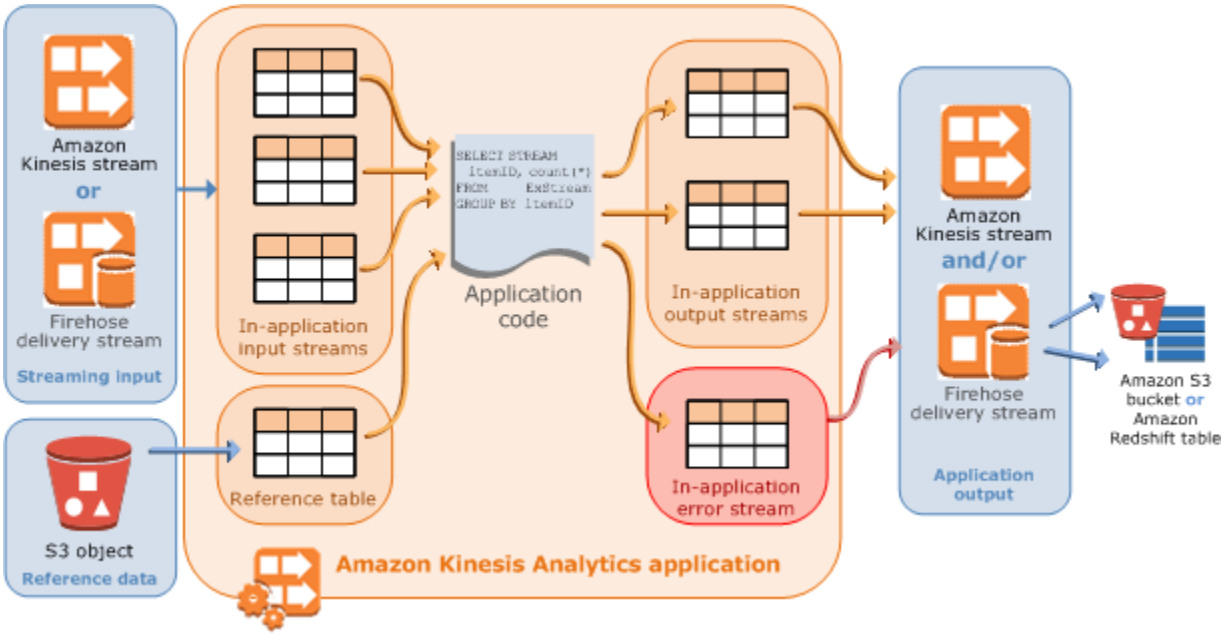
1. Leia a seção Como funciona deste guia. Esta seção apresenta vários componentes do Kinesis Data Analytics com os quais você trabalha para criar uma end-to-end experiência. Para ter mais informações, consulte Amazon Kinesis Data Analytics para aplicativos SQL: como funciona.
2. Tente fazer os exercícios de conceitos básicos. Para ter mais informações, consulte Conceitos básicos do Amazon Kinesis Data Analytics para aplicativos SQL.
3. Explore os conceitos de SQL de streaming. Para ter mais informações, consulte Conceitos de streaming com SQL.
4. Tente exemplos adicionais. Para ter mais informações, consulte Exemplos de Kinesis Data Analytics para SQL.

Amazon Kinesis Data Analytics para aplicativos SQL: como funciona

> **Note**
>
> Depois de 12 de setembro de 2023, você não poderá criar novos aplicativos usando o Kinesis Data Firehose como fonte se ainda não estiver usando o Kinesis Data Analytics for SQL. Para obter mais informações, consulte Limites.

Um aplicativo é o recurso principal do Amazon Kinesis Data Analytics que você pode criar na conta. Você pode criar e gerenciar aplicativos usando a API Kinesis Data Analytics AWS Management Console ou a Kinesis Data Analytics. O Kinesis Data Analytics fornece operações de API para gerenciar aplicativos. Para obter uma lista das operações de API, consulte Ações.

Os aplicativos do Kinesis Data Analytics leem e processam o streaming de dados continuamente e em tempo real. Escreva o código do aplicativo usando o SQL para processar os dados em streaming de entrada e produzir a saída. Depois, o Kinesis Data Analytics grava a saída em um destino configurado. O diagrama a seguir ilustra uma arquitetura típica de aplicativo.



Cada aplicativo tem um nome, uma descrição, o ID da versão e o status. O Amazon Kinesis Data Analytics atribui um ID de versão ao criar um aplicativo pela primeira vez. Esse ID de versão é atualizado quando você atualiza qualquer configuração de aplicativo. Por exemplo, se você adicionar

uma configuração de entrada, adicionar ou excluir uma fonte de dados de referência, adicionar ou excluir uma configuração de saída ou atualizar o código do aplicativo, o Kinesis Data Analytics atualizará o ID da versão de aplicativo atual. O Kinesis Data Analytics também mantém timestamps de quando um aplicativo foi criado e atualizado pela última vez.

Além dessas propriedades básicas, cada aplicativo consiste no seguinte:

- Entrada: a origem do streaming do aplicativo. Você pode selecionar um stream de dados do Kinesis ou um stream de entrega de dados do Firehose como fonte de streaming. Na configuração de entrada, você mapeia a origem de streaming para um fluxo de entrada de aplicativo. O fluxo no aplicativo funciona como uma tabela em contínua atualização em que você pode executar as operações `SELECT` e `INSERT SQL`. No seu código da aplicação, é possível criar fluxos no aplicativo adicionais para armazenar resultados de consulta intermediários.

  Se desejar, você pode particionar uma única origem de streaming em vários fluxos de entrada de aplicativo para melhorar a taxa de transferência. Para obter mais informações, consulte Configuração de entrada do aplicativo e Limites.

  O Amazon Kinesis Data Analytics fornece uma coluna de timestamp em cada stream de aplicativo chamado Time stamps e a coluna ROWTIME. É possível usar essa coluna em consultas em janelas baseadas em tempo. Para ter mais informações, consulte Consultas em janelas.

  Se desejar, você pode configurar uma fonte de dados de referência para enriquecer o fluxo de dados de entrada no aplicativo. Isso resultará em uma tabela de referência de aplicativo. Você deve armazenar os dados de referência como um objeto no bucket do S3. Quando o aplicativo é iniciado, o Amazon Kinesis Data Analytics lê o objeto do Amazon S3 e cria uma tabela no aplicativo. Para ter mais informações, consulte Configuração de entrada do aplicativo.

- Código do aplicativo: uma série de instruções SQL que processam entrada e produzem saída. Grave instruções SQL contra fluxos no aplicativo e tabelas de referência. Você também pode gravar consultas JOIN para combinar dados de ambas as origens.

Para obter mais informações sobre os elementos da linguagem SQL compatíveis com o Kinesis Data Analytics, consulte Amazon Kinesis Data Analytics SQL Reference.

Em sua forma mais simples, o código do aplicativo pode ser uma instrução SQL única que faz a seleção de uma entrada de streaming e insere os resultados em uma saída de streaming. Ele também pode ser uma série de instruções SQL em que a saída de uma delas contribui para a entrada da próxima instrução SQL. Além disso, você pode gravar código de aplicativo para dividir um fluxo de entrada em vários fluxos. Em seguida, aplique consultas adicionais para processar esses fluxos. Para ter mais informações, consulte Código do aplicativo.

- Saída: no código do aplicativo, os resultados da consulta são passados para os fluxos de aplicativo. No código do aplicativo, é possível criar um ou mais fluxos de aplicativo para manter os resultados intermediários. Depois, se desejar, você pode configurar a saída do aplicativo para manter os dados dos fluxos de aplicativo, que retêm a saída do aplicativo (conhecida também como fluxos de saída do aplicativo) nos destinos externos. Os destinos externos podem ser um stream de entrega do Firehose ou um stream de dados do Kinesis. Observe o seguinte sobre esses destinos:
  - Você pode configurar um stream de entrega do Firehose para gravar resultados no Amazon S3, Amazon Redshift ou Amazon OpenSearch Service (Service). OpenSearch
  - Você também pode gravar a saída do aplicativo em um destino personalizado, em vez do Amazon S3 ou do Amazon Redshift. Para fazer isso, especifique um fluxo de dados do Kinesis como destino na configuração de saída. Em seguida, você configura AWS Lambda para pesquisar o stream e invocar sua função Lambda. Seu código da função do Lambda recebe os dados do stream como entrada. No seu código da função do Lambda, você pode gravar os dados de entrada no destino personalizado. Para obter mais informações, consulte Como usar AWS Lambda com o Amazon Kinesis Data Analytics.

  Para ter mais informações, consulte Configuração da saída do aplicativo.

Além disso, observe o seguinte:

- O Amazon Kinesis Data Analytics precisa de permissões para ler registros em uma origem de streaming e gravar a saída do aplicativo nos destinos externos. Use os perfis do IAM para conceder essas permissões.

- O Kinesis Data Analytics fornece automaticamente um stream de erros de aplicativo para cada aplicação. Se o aplicativo tiver problemas ao processar determinados registros (por exemplo, devido a uma incompatibilidade de tipo ou chegada em atraso), esse registro será gravado no fluxo de erros. Você pode configurar a saída do aplicativo para que o Kinesis Data Analytics mantenha os dados do stream de erros em um destino externo para avaliação posterior. Para ter mais informações, consulte Como tratar erros.

- O Amazon Kinesis Data Analytics garante que os registros de saída do aplicativo serão gravados no destino configurado. Ele usa um modelo de processamento e entrega "pelo menos uma vez", mesmo que ocorra uma interrupção do aplicativo. Para ter mais informações, consulte Modelo de entrega para manter a saída do aplicativo em um destino externo.

Tópicos

Configuração de entrada do aplicativo

Seu aplicativo Amazon Kinesis Data Analytics pode receber entrada de uma única fonte de streaming e, opcionalmente, usar uma fonte de dados de referência. Para ter mais informações, consulte Amazon Kinesis Data Analytics para aplicativos SQL: como funciona. As seções neste tópico descrevem as fontes de entrada do aplicativo.

Tópicos

Configuração de uma fonte de streaming

No momento em que você criar um aplicativo, especifique uma fonte de streaming. Também é possível modificar uma entrada depois de criar o aplicativo. O Amazon Kinesis Data Analytics é compatível com as seguintes fontes de streaming do aplicativo:

- Um fluxo de dados do Kinesis
- Um stream de entrega do Firehose

> **Note**
>
> Depois de 12 de setembro de 2023, você não poderá criar novos aplicativos usando o Kinesis Data Firehose como fonte se ainda não estiver usando o Kinesis Data Analytics for SQL. Os clientes atuais que usam o Kinesis Data Analytics para aplicativos SQL com `KinesisFirehoseInput` podem continuar adicionando aplicativos `KinesisFirehoseInput` em uma conta existente usando o Kinesis Data Analytics. Se você já é cliente e deseja criar uma nova conta com os aplicativos Kinesis Data Analytics para SQL com `KinesisFirehoseInput`, você pode criar um caso usando o formulário de aumento do limite de serviço. Para obter mais informações, consulte a CentralAWS Support. Recomendamos sempre testar os novos aplicativos antes de passá-los à produção.

> **Note**
>
> Se o fluxo de dados do Kinesis estiver criptografado, o Kinesis Data Analytics acessa os dados no fluxo criptografado perfeitamente sem precisar de outras configurações. O Kinesis Data Analytics não armazena dados não criptografados lidos dos fluxos de dados

Kinesis. Para obter mais informações, consulte O que é criptografia no lado do servidor para streamings de dados do Kinesis?.

O Kinesis Data Analytics apura continuamente a fonte de streaming para novos dados e os ingere em fluxos de aplicativos de acordo com a configuração de entrada.

> **Note**
>
> A adição de um fluxo do Kinesis como entrada do seu aplicativo não afeta os dados do fluxo. Se outro recurso, como um stream de entrega do Firehose, também acessasse o mesmo stream do Kinesis, tanto o stream de entrega do Firehose quanto o aplicativo Kinesis Data Analytics receberiam os mesmos dados. Entretanto, o throughput e o controle de utilização podem ser afetados.

O código de aplicativo pode consultar o stream de aplicativos. Como parte da configuração de entrada, forneça o seguinte:

- Fonte de streaming – forneça o nome do recurso da Amazon (ARN) do fluxo e um perfil do IAM que o Kinesis Data Analytics pode assumir para acessar o fluxo em seu nome.
- Prefixo do nome de fluxo no aplicativo – quando você inicia o aplicativo, o Kinesis Data Analytics cria o fluxo no aplicativo especificado. No código do aplicativo, acesse o stream no aplicativo usando o mesmo nome.

  Ou você pode mapear uma fonte de streaming para vários fluxos no aplicativo. Para ter mais informações, consulte Limites. Nesse caso, o Amazon Kinesis Data Analytics cria o número especificado de streams no aplicativo com nomes assim: *prefixo*_001, *prefixo*_002, e *prefixo*_003. Por padrão, o Kinesis Data Analytics mapeia a fonte de streaming para um fluxo no aplicativo chamado *prefixo*_001.

  Há um limite na velocidade que você pode inserir linhas em um stream no aplicativo. Portanto, o Kinesis Data Analytics oferece suporte a vários streams no aplicativo para que você possa trazer registros para seu aplicativo a uma velocidade muito mais alta. Se você considerar que o aplicativo não está acompanhando o ritmo dos dados na fonte de streaming, adicione unidades de paralelismo para melhorar o desempenho.

- Mapeamento de esquema – você descreve o formato de registro (JSON, CSV) na fonte de streaming. Você também descreve como cada registro no stream mapeia para colunas no stream do aplicativo criado. É aqui você fornece nomes de colunas e tipos de dados.

> **Note**
>
> O Kinesis Data Analytics adiciona aspas em torno dos identificadores (nome de stream e nomes de coluna) ao criar o stream no aplicativo de entrada. Ao consultar esse stream e as colunas, você deve especificá-los entre aspas usando a mesma capitalização (letras maiúsculas e minúsculas exatamente). Para obter mais informações sobre identificadores, consulte Identificadores no Amazon Managed Service for Apache Flink SQL Reference.

É possível criar um aplicativo e configurar entradas no console do Amazon Kinesis Data Analytics. O console então faz as chamadas de API necessárias. Você pode configurar a entrada do aplicativo ao criar uma nova API de aplicativo ou adicionar uma configuração de entrada a um aplicativo existente. Para obter mais informações, consulte AddApplicationInput e CreateApplication. Veja a seguir a parte da configuração de entrada do corpo da solicitação da API `Createapplication`:

```
"Inputs": [
       {
           "InputSchema": {
               "RecordColumns": [
                   {
                       "Mapping": "string",
                       "Name": "string",
                       "SqlType": "string"
                   }
               ],
               "RecordEncoding": "string",
               "RecordFormat": {
                   "MappingParameters": {
                       "CSVMappingParameters": {
                           "RecordColumnDelimiter": "string",
                           "RecordRowDelimiter": "string"
                       },
                       "JSONMappingParameters": {
                           "RecordRowPath": "string"
                       }
                   },
```

```
                    "RecordFormatType": "string"
                }
            },
            "KinesisFirehoseInput": {
                "ResourceARN": "string",
                "RoleARN": "string"
            },
            "KinesisStreamsInput": {
                "ResourceARN": "string",
                "RoleARN": "string"
            },
            "Name": "string"
        }
    ]
```

Configuração de uma fonte de referência

Como opção, você pode adicionar uma fonte de dados de referência a um aplicativo existente para enriquecer os dados que chegam de fontes de streaming. Você deve armazenar os dados de referência como um objeto no bucket do Amazon S3. Quando o aplicativo é iniciado, o Amazon Kinesis Data Analytics lê o objeto do Amazon S3 e cria uma tabela de referência no aplicativo. O código do aplicativo pode, então, associar-se a um streaming no aplicativo.

Armazene os dados de referência no objeto do Amazon S3 usando formatos compatíveis (CSV, JSON). Por exemplo, suponha que o seu aplicativo realiza análises em pedidos de ações. Assuma o seguinte formato de registro na fonte de streaming:

```
Ticker, SalePrice, OrderId

AMZN     $700        1003
XYZ      $250        1004
...
```

Neste caso, você pode considerar a manutenção de uma fonte de dados de referência para fornecer detalhes para cada marcador de ações, como nome da empresa.

```
Ticker, Company
AMZN, Amazon
XYZ, SomeCompany
...
```

Você pode adicionar uma fonte de dados de referência do aplicativo com a API ou com o console. O Amazon Kinesis Data Analytics oferece as seguintes ações de API para gerenciar fontes de dados de referência:

- AddApplicationReferenceDataSource
- UpdateApplication

Para obter informações sobre como adicionar dados de referência usando o console, consulte Exemplo: adição de dados de referência a um aplicativo do Kinesis Data Analytics.

Observe o seguinte:

- Se o aplicativo estiver em execução, o Kinesis Data Analytics criará uma tabela de referência no aplicativo e, em seguida, carregará os dados de referência imediatamente.
- Se o aplicativo não estiver em execução (por exemplo, no estado de prontidão), o Kinesis Data Analytics salvará apenas a configuração de entrada atualizada. Quando o aplicativo começa a executar, o Kinesis Data Analytics carrega os dados de referência no aplicativo como uma tabela.

Suponha que você deseje atualizar os dados depois que o Kinesis Data Analytics criar a tabela de referência no aplicativo. Talvez você tenha atualizado o objeto do Amazon S3 ou queira usar outro objeto do Amazon S3. Nesse caso, você pode chamar explicitamente UpdateApplication ou selecionar Ações, Sincronizar tabela de dados de referência no console. O Kinesis Data Analytics não atualiza a tabela de referência no aplicativo automaticamente.

Há um limite para o tamanho do objeto do Amazon S3 que você pode criar como uma fonte de dados de referência. Para ter mais informações, consulte Limites. Se o tamanho do objeto exceder o limite, o Kinesis Data Analytics não poderá carregar os dados. O estado do aplicativo aparecerá como em execução, mas os dados não serão lido.

Quando você adiciona uma fonte de referência de dados, fornece as seguintes informações:

- Nome da chave do objeto e bucket do S3 – além do nome do bucket e da chave do objeto, forneça também um perfil do IAM que o Kinesis Data Analytics possa assumir para ler o objeto em seu nome.
- Nome da tabela de referência no aplicativo – o Kinesis Data Analytics cria essa tabela no aplicativo e a preenche lendo o objeto do Amazon S3. Esse é o nome da tabela que você especifica no código do aplicativo.

- Mapeamento de esquema – descreva o formato de registro (JSON, CSV), a codificação de dados armazenados no objeto do Amazon S3. Descreva também como cada elemento de dados é mapeado para colunas na tabela de referência no aplicativo.

A tabela a seguir mostra o corpo de solicitação na solicitação da API `AddApplicationReferenceDataSource`.

```
{
    "applicationName": "string",
    "CurrentapplicationVersionId": number,
    "ReferenceDataSource": {
        "ReferenceSchema": {
            "RecordColumns": [
                {
                    "IsDropped": boolean,
                    "Mapping": "string",
                    "Name": "string",
                    "SqlType": "string"
                }
            ],
            "RecordEncoding": "string",
            "RecordFormat": {
                "MappingParameters": {
                    "CSVMappingParameters": {
                        "RecordColumnDelimiter": "string",
                        "RecordRowDelimiter": "string"
                    },
                    "JSONMappingParameters": {
                        "RecordRowPath": "string"
                    }
                },
                "RecordFormatType": "string"
            }
        },
        "S3ReferenceDataSource": {
            "BucketARN": "string",
            "FileKey": "string",
            "ReferenceRoleARN": "string"
        },
        "TableName": "string"
    }
}
```

Trabalho com JSONPath

> **Note**
>
> Depois de 12 de setembro de 2023, você não poderá criar novos aplicativos usando o Kinesis Data Firehose como fonte se ainda não estiver usando o Kinesis Data Analytics for SQL. Para obter mais informações, consulte Limites.

JSONPath é uma maneira padronizada de consultar elementos de um objeto JSON. O JSONPath usa expressões de caminho para navegar elementos, elementos aninhados e matrizes em um documento JSON. Para obter mais informações sobre JSON, consulte Introdução ao JSON.

O Amazon Kinesis Data Analytics usa expressões JSONPath no esquema de origem do aplicativo para identificar elementos de dados em uma fonte de streaming que contém dados no formato JSON.

Para obter mais informações sobre como mapear dados de streaming para o fluxo de entrada do aplicativo, consulte the section called “Mapeamento de elementos de fonte de streaming para colunas de entrada do SQL”.

Acesso a elementos JSON com JSONPath

A seguir, veja como usar expressões JSONPath para acessar diversos elementos em dados formatados pelo JSON. Para os exemplos nesta seção, suponha que o stream de origem contenha o seguinte registro JSON:

```
{
  "customerName":"John Doe",
  "address":
  {
    "streetAddress":
    [
      "number":"123",
      "street":"AnyStreet"
    ],
    "city":"Anytown"
  }
  "orders":
  [
```

```
    { "orderId":"23284", "itemName":"Widget", "itemPrice":"33.99" },
    { "orderId":"63122", "itemName":"Gadget", "itemPrice":"22.50" },
    { "orderId":"77284", "itemName":"Sprocket", "itemPrice":"12.00" }
  ]
}
```

Acesso a elementos JSON

Para consultar um elemento em dados JSON usando JSONPath, use a seguinte sintaxe. Aqui, $ representa a raiz da hierarquia de dados e `elementName` é o nome do nó do elemento de consulta.

```
$.elementName
```

A expressão a seguir consulta o elemento `customerName` no exemplo de JSON anterior.

```
$.customerName
```

A expressão anterior retorna o seguinte do registro de JSON anterior.

```
John Doe
```

> **Note**
>
> As expressões do caminho diferenciam maiúsculas e minúsculas. A expressão `$.customername` retorna `null` do exemplo de JSON anterior.

> **Note**
>
> Se nenhum elemento aparecer no local onde a expressão do caminho especifica, a expressão retornará `null`. A expressão a seguir retorna `null` do exemplo de JSON anterior, porque não há elemento correspondente.
>
> ```
> $.customerId
> ```

Acesso a elementos JSON aninhados

Para consultar um elemento JSON aninhado, use a seguinte sintaxe.

```
$.parentElement.element
```

A expressão a seguir consulta o elemento `city` no exemplo de JSON anterior.

```
$.address.city
```

A expressão anterior retorna o seguinte do registro de JSON anterior.

```
Anytown
```

Você pode consultar mais níveis de subelementos usando a sintaxe a seguir.

```
$.parentElement.element.subElement
```

A expressão a seguir consulta o elemento `street` no exemplo de JSON anterior.

```
$.address.streetAddress.street
```

A expressão anterior retorna o seguinte do registro de JSON anterior.

```
AnyStreet
```

Acesso a matrizes

Você pode acessar os dados em uma matriz JSON das seguintes formas:

- Recupere todos os elementos na matriz como uma única linha.
- Recupere cada elemento na matriz como uma linha separada.

Recuperar todos os elementos em uma matriz em uma única linha

Para consultar todo o conteúdo de uma matriz como uma única linha, use a sintaxe a seguir.

```
$.arrayObject[0:]
```

A expressão a seguir consulta todo o conteúdo do elemento `orders` no exemplo de JSON anterior usado nesta seção. Ela retorna o conteúdo da matriz em uma única coluna em uma única linha.

```
$.orders[0:]
```

A expressão anterior retorna o seguinte do registro JSON de exemplo usado nesta seção.

```
[{"orderId":"23284","itemName":"Widget","itemPrice":"33.99"},
{"orderId":"61322","itemName":"Gadget","itemPrice":"22.50"},
{"orderId":"77284","itemName":"Sprocket","itemPrice":"12.00"}]
```

Recuperar todos os elementos em uma matriz em linhas separadas

Para consultar os elementos individuais em uma matriz como linhas separadas, use a seguinte sintaxe.

```
$.arrayObject[0:].element
```

A expressão a seguir consulta os elementos `orderId` no exemplo de JSON anterior e retorna cada elemento de matriz como uma linha separada.

```
$.orders[0:].orderId
```

A expressão anterior retorna o seguinte do registro de JSON anterior, com cada item de dados retornado como uma linha separada.

23284
63122
77284

> **Note**
>
> Se expressões que consultam elementos não matriz estiverem incluídos em um esquema que consulta elementos de matriz individuais, os elementos não matriz serão repetidos para

cada elemento na matriz. Por exemplo, suponha que um esquema para o exemplo de JSON anterior inclua as seguintes expressões:

- $.customerName
- $.orders[0:].orderId

Nesse caso, as linhas de dados retornados do elemento de stream de entrada de amostra se parecem com o seguinte, com o elemento `name` repetido para cada elemento `orderId`.

John Doe	23284
John Doe	63122
John Doe	77284

Note

As seguintes limitações se aplicam a expressões em matriz no Amazon Kinesis Data Analytics:

- Apenas um nível de deferência conta com suporte em uma expressão. O formato de expressão a seguir não conta com suporte.

```
$.arrayObject[0:].element[0:].subElement
```

- Apenas uma matriz pode ser achatada em um esquema. Várias matrizes podem ser referenciadas-retornadas como uma linha contendo todos os elementos na matriz. No entanto, apenas uma matriz pode ter cada um de seus elementos retornados como linhas individuais.

  Um esquema que contém elementos no seguinte formato é válido. Este formato retorna o conteúdo da segunda matriz como uma única coluna, repetido para cada elemento na primeira matriz.

```
$.arrayObjectOne[0:].element
$.arrayObjectTwo[0:]
```

Um esquema que contém elementos no seguinte formato não é válido.

```
$.arrayObjectOne[0:].element
$.arrayObjectTwo[0:].element
```

Outras considerações

As considerações adicionais para trabalhar com JSONPath são as seguintes:

- Se nenhuma matriz for acessada por um elemento individual nas expressões JSONPath no esquema do aplicativo, uma única linha será criada no fluxo de entrada de aplicativo para cada registro JSON processado.
- Quando uma matriz é achatada (ou seja, seus elementos são retornados como linhas individuais), quaisquer elementos ausentes resultam na criação de um valor nulo no stream no aplicativo.
- Uma matriz é sempre achatada para pelo menos uma linha. Se nenhum valor for retornado (ou seja, a matriz estiver vazia ou nenhum de seus elementos for consultado), uma única linha com todos os valores nulos será retornada.

  A expressão a seguir retorna registros com valores nulos de exemplo de JSON anterior, porque não há elemento correspondente no caminho especificado.

  ```
  $.orders[0:].itemId
  ```

  A expressão anterior retorna o seguinte do registro de exemplo de JSON anterior.

nulo
nulo
nulo

Related Topics

- Apresentação do JSON

Mapeamento de elementos de fonte de streaming para colunas de entrada do SQL

> **Note**
>
> Depois de 12 de setembro de 2023, você não poderá criar novos aplicativos usando o Kinesis Data Firehose como fonte se ainda não estiver usando o Kinesis Data Analytics for SQL. Para obter mais informações, consulte Limites.

Com o Amazon Kinesis Data Analytics, é possível processar e analisar dados de streaming nos formatos JSON ou CSV usando o SQL padrão.

- Para processar e analisar dados CSV de streaming, atribua nomes de colunas e tipos de dados às colunas do stream de entrada. O aplicativo importa uma coluna do stream de entrada por definição de coluna, em ordem.

  Não é necessário incluir todas as colunas do stream de entrada de aplicativo, mas você não pode ignorar colunas do stream de origem. Por exemplo, é possível importar as três primeiras colunas de um stream de entrada que contém cinco elementos, mas não importar apenas as colunas 1, 2 e 4.
- Para processar e analisar os dados JSON do streaming, use as expressões JSONPath para mapear elementos JSON de uma fonte de streaming para colunas SQL em um stream de entrada. Para obter mais informações sobre JSONPath com Amazon Kinesis Data Analytics, consulte Trabalho com JSONPath. As colunas na tabela SQL têm tipos de dados que são mapeados de tipos de JSON. Para obter os tipos de dados compatíveis, consulte Tipos de dados. Para obter detalhes sobre como converter dados JSON em dados SQL, consulte Mapeamento de tipos de dados JSON para tipos de dados SQL.

Para obter mais informações sobre como configurar fluxos de entrada, consulte Configuração de entrada do aplicativo.

Mapeamento de dados JSON para colunas SQL

Você pode mapear elementos JSON para colunas de entrada usando a API Kinesis Data AWS Management Console Analytics ou a Kinesis Data Analytics.

- Para mapear elementos para colunas usando o console, consulte Trabalho com o editor de esquema.
- Para mapear elementos para colunas usando a API do Kinesis Data Analytics, consulte a seção a seguir.

Para mapear elementos JSON para colunas no stream de entrada no aplicativo, você precisa de um esquema com as seguintes informações para cada coluna:

- Expressão de origem: a expressão JSONPath que identifica o local dos dados para a coluna.
- Nome da coluna: o nome que as consultas do SQL usam para referenciar os dados.
- Tipo de dados: o tipo de dados do SQL para a coluna.

Uso da API

Para mapear elementos de uma fonte de streaming para colunas de entrada, use a ação CreateApplication da API do Kinesis Data Analytics. Para criar o stream no aplicativo, especifique um esquema para transformar seus dados em uma versão esquematizada usada no SQL. A ação CreateApplication configura o aplicativo para receber a entrada de uma única fonte de streaming. Para mapear elementos JSON ou colunas CSV para colunas SQL, crie um objeto RecordColumn na matriz SourceSchema `RecordColumns`. O objeto RecordColumn tem o seguinte esquema:

```
{
    "Mapping": "String",
    "Name": "String",
    "SqlType": "String"
}
```

Os campos do objeto RecordColumn têm os seguintes valores:

- `Mapping`: a expressão JSONPath que identifica o local dos dados no registro de stream de entrada. Esse valor não está presente em um esquema de entrada para um stream de origem no formato CSV.
- `Name`: o nome da coluna no streaming de dados SQL no aplicativo.
- `SqlType`: o tipo de dados dos dados no stream de dados SQL no aplicativo.

Exemplo de esquema de entrada JSON

O exemplo a seguir demonstra o formato do valor `InputSchema` para um esquema JSON.

```
"InputSchema": {
    "RecordColumns": [
        {
            "SqlType": "VARCHAR(4)",
            "Name": "TICKER_SYMBOL",
            "Mapping": "$.TICKER_SYMBOL"
        },
        {
            "SqlType": "VARCHAR(16)",
            "Name": "SECTOR",
            "Mapping": "$.SECTOR"
        },
        {
            "SqlType": "TINYINT",
            "Name": "CHANGE",
            "Mapping": "$.CHANGE"
        },
        {
            "SqlType": "DECIMAL(5,2)",
            "Name": "PRICE",
            "Mapping": "$.PRICE"
        }
    ],
    "RecordFormat": {
        "MappingParameters": {
            "JSONMappingParameters": {
                "RecordRowPath": "$"
            }
        },
        "RecordFormatType": "JSON"
    },
    "RecordEncoding": "UTF-8"
}
```

Exemplo de esquema de entrada CSV

O exemplo a seguir demonstra o formato do valor `InputSchema` para um esquema no formato CSV (valores separados por vírgulas).

```
"InputSchema": {
    "RecordColumns": [
        {
            "SqlType": "VARCHAR(16)",
            "Name": "LastName"
        },
        {
            "SqlType": "VARCHAR(16)",
            "Name": "FirstName"
        },
        {
            "SqlType": "INTEGER",
             "Name": "CustomerId"
        }
    ],
    "RecordFormat": {
        "MappingParameters": {
            "CSVMappingParameters": {
                "RecordColumnDelimiter": ",",
                "RecordRowDelimiter": "\n"
            }
        },
        "RecordFormatType": "CSV"
    },
    "RecordEncoding": "UTF-8"
}
```

Mapeamento de tipos de dados JSON para tipos de dados SQL

Os tipos de dados JSON são convertidos em tipos de dados SQL correspondentes de acordo com o esquema de entrada do aplicativo. Para obter informações sobre os tipos de dados SQL com suporte, consulte Tipos de dados. O Amazon Kinesis Data Analytics converte tipos de dados JSON em tipos de dados SQL de acordo com as seguintes regras.

Nulo literal

Um nulo literal no stream de entrada JSON (ou seja, `"City":null`) é convertido em um SQL nulo, independentemente do tipo de dados de destino.

Booleano literal

Um Booleano literal no stream de entrada JSON (ou seja, `"Contacted":true`) é convertido nos seguintes dados SQL:

- Numérico (DECIMAL, INT e assim por diante): `true` é convertido em 1; `false` é convertido em 0.
- Binário (BINARY ou VARBINARY):
  - `true`: o resultado tem o menor conjunto de bits e os bits restantes são desmarcados.
  - `false`: o resultado tem todos os bits desmarcados.

  A conversão em VARBINARY resulta em um valor de 1 byte de comprimento.
- BOOLEANO: é convertido no valor SQL BOOLEAN correspondente.
- Caractere (CHAR ou VARCHAR): é convertido no valor da string correspondente (`true` ou `false`). O valor é truncado para se ajustar ao tamanho do campo.
- Data e hora (DATE, TIME ou TIMESTAMP): a conversão falha e um erro de coerção é gravado no stream de erro.

Número

Um número literal no stream de entrada JSON (ou seja, `"CustomerId":67321`) é convertido nos seguintes dados SQL:

- Numérico (DECIMAL, INT e assim por diante): é convertido diretamente. Se o valor convertido exceder o tamanho ou a precisão do tipo de dados de destino (ou seja, convertendo `123.4` em INT), a conversão falha e um erro de coerção é gravado no stream de erro.
- Binário (BINARY ou VARBINARY): a conversão falha e um erro de coerção é gravado no stream de erro.
- BOOLEAN:
  - `0`: é convertido em `false`.
  - Todos os outros números: são convertidos em `true`.
- Caractere (CHAR ou VARCHAR): é convertido em uma representação do número em string.
- Data e hora (DATE, TIME ou TIMESTAMP): a conversão falha e um erro de coerção é gravado no stream de erro.

String

O valor de uma string no stream de entrada JSON (ou seja, `"CustomerName":"John Doe"`) é convertido nos seguintes dados SQL:

- Numérico (DECIMAL, INT e assim por diante): o Amazon Kinesis Data Analytics tenta converter o valor no tipo de dados de destino. Se o valor não puder ser convertido, a conversão falha e um erro de coerção será gravado no stream de erro.
- rio (BINARY ou VARBINARY): se a string de origem for um binário literal (ou seja, `X'3F67A23A'`, com um número par de f), o valor será convertido no tipo de dados de destino. Caso contrário, a conversão falha e um erro de coerção será gravado no stream de erro.
- BOOLEANO: se a string de origem for `"true"`, será convertida em `true`. Essa comparação diferencia maiúsculas de minúsculas. Caso contrário, será convertida em `false`.
- Caractere (CHAR ou VARCHAR): é convertido no valor da string na entrada. Se o valor for maior do que o tipo de dados de destino, ele estará truncado e nenhum erro será gravado no stream de erro.
- Data e hora (DATE, TIME ou TIMESTAMP): se a string de origem estiver em um formato que pode ser convertido no valor de destino, o valor será convertido. Caso contrário, a conversão falha e um erro de coerção será gravado no stream de erro.

  Os formatos de data e hora válidos incluem:

  - "1992-02-14"
  - "1992-02-14 18:35:44.0"

Matriz ou objeto

Uma matriz ou objeto no stream de entrada JSON é convertido em dados SQL da seguinte forma:

- Caractere (CHAR ou VARCHAR): é convertido no texto de origem da matriz ou objeto. Consulte Acesso a matrizes.
- Todos os outros tipos de dados: a conversão falha e um erro de coerção é gravado no stream de erro.

Para obter um exemplo de uma matriz JSON, consulte Trabalho com JSONPath.

Related Topics

- Configuração de entrada do aplicativo
- Tipos de dados
- Trabalho com o editor de esquema
- CreateApplication
- RecordColumn
- SourceSchema

Usar o recurso de descoberta de esquema em dados em streaming

> **Note**
>
> Depois de 12 de setembro de 2023, você não poderá criar novos aplicativos usando o Kinesis Data Firehose como fonte se ainda não estiver usando o Kinesis Data Analytics for SQL. Para obter mais informações, consulte Limites.

Fornecer um esquema de entrada que descreve como os registros na entrada de streaming são mapeados para um stream no aplicativo pode ser trabalhoso e propenso a erros. Use a API DiscoverInputSchema (chamada de API de descoberta) para inferir um esquema. Usando amostras aleatórias de registros na fonte de streaming, a API pode inferir um esquema (ou seja, nomes de colunas, tipos de dados e posição do elemento de dados nos dados recebidos).

> **Note**
>
> Para usar a API de descoberta para gerar um esquema de um arquivo armazenado no Amazon S3, consulte Usar o recurso de descoberta de esquema em dados estáticos.

O console usa a API de descoberta para gerar um esquema para uma fonte de streaming especificada. Usando o console, é possível também atualizar o esquema, inclusive adicionar ou remover colunas, alterar nomes de colunas ou tipos de dados e assim por diante. No entanto, faça alterações cuidadosamente para garantir que um esquema inválido não seja criado.

Após finalizar um esquema para o stream no aplicativo, será possível usar funções para manipular strings e valores de data e hora. Use essas funções no código do aplicativo ao trabalhar com linhas

no stream no aplicativo resultante. Para ter mais informações, consulte Exemplo: Transformação de valores DateTime .

Nomeação de colunas durante a descoberta de esquema

Durante a descoberta de esquema, o Amazon Kinesis Data Analytics tenta reter o máximo possível do nome da coluna original da fonte de entrada de streaming, exceto nos seguintes casos:

- O nome do stream de origem é uma palavra-chave reservada do SQL, como `TIMESTAMP`, `USER`, `VALUES` ou `YEAR`.
- O nome da coluna de stream de origem contém caracteres não suportados. Somente letras, números e o caractere de sublinhado (_) são suportados.
- O nome da coluna de stream de origem começa com um número.
- O nome da coluna de stream de origem tem mais de 100 caracteres.

Se uma coluna for renomeada, o nome da coluna do esquema renomeada começará com `COL_`. Em alguns casos, nenhum nome de coluna original pode ser retido, por exemplo, se todo o nome tiver caracteres não suportados. Nesse caso, a coluna será chamada `COL_#`, com # sendo um número que indica o local da coluna na ordem da coluna.

Após a conclusão da descoberta, você poderá atualizar o esquema usando o console para adicionar ou remover colunas ou alterar nomes de coluna, tipos de dados ou tamanho de dados.

Exemplos de nomes de coluna sugeridos pela descoberta

Nome da coluna de stream de origem	Nome da coluna sugerido pela descoberta
USER	COL_USER
USER@DOMAIN	COL_USERDOMAIN
@@	COL_0

Problemas de descoberta de esquema

O que acontecerá se o Kinesis Data Analytics não inferir um esquema para uma determinada fonte de streaming?

O Kinesis Data Analytics infere o esquema para formatos comuns, como CSV e JSON, que são codificados em UTF-8. O Kinesis Data Analytics oferece suporte a quaisquer registros codificados em UTF-8 (incluindo texto bruto, como logs e registros de aplicativos) com colunas personalizadas e delimitadores de linhas. Se o Kinesis Data Analytics não inferir um esquema, você poderá definir um esquema manualmente usando o editor de esquema no console (ou usando a API).

Se os dados não seguirem um padrão (que você pode especificar usando o editor de esquema), defina um esquema como uma única coluna do tipo VARCHAR(N), em que N é o maior número de caracteres que você espera que seu registro inclua. A partir daí, será possível usar a manipulação de strings e de data e hora para estruturar os dados a partir do momento que estiverem no stream no aplicativo. Para ver exemplos, consulte Exemplo: Transformação de valores DateTime .

Usar o recurso de descoberta de esquema em dados estáticos

> **Note**
>
> Depois de 12 de setembro de 2023, você não poderá criar novos aplicativos usando o Kinesis Data Firehose como fonte se ainda não estiver usando o Kinesis Data Analytics for SQL. Para obter mais informações, consulte Limites.

O recurso de descoberta de esquema pode gerar um esquema dos dados em um stream ou em um arquivo estático armazenado em um bucket do Amazon S3. Suponha que você deseje gerar um esquema para um aplicativo do Kinesis Data Analytics para fins de referência ou quando os dados em streaming ao vivo não estiverem disponíveis. Você pode usar o atributo de descoberta de esquema em um arquivo estático que contém uma amostra dos dados no formato esperado de dados em streaming ou de referência. O Kinesis Data Analytics pode executar a descoberta de esquema em dados de amostra de um arquivo JSON ou CSV armazenado em um bucket do Amazon S3. Executar a descoberta de esquema em um arquivo de dados usa o console ou a API DiscoverInputSchema com o parâmetro `S3Configuration` especificado.

Execução de descoberta de esquema usando o console

Para executar a descoberta em um arquivo estático usando o console, faça o seguinte:

1. Adicione um objeto de dados de referência a um bucket do S3.
2. Selecione Conectar dados de referência na página principal do aplicativo no console do Kinesis Data Analytics.

3. Forneça os dados do bucket, do caminho e do perfil do IAM para acessar o objeto do Amazon S3 que contém os dados de referência.
4. Escolha Discover schema (Descobrir esquema).

Para obter mais informações sobre como adicionar dados de referência e descobrir o esquema no console, consulte Exemplo: adição de dados de referência a um aplicativo do Kinesis Data Analytics.

Execução de descoberta de esquema usando a API

Para executar a descoberta em um arquivo estático usando a API, forneça à API uma estrutura `S3Configuration` com as seguintes informações:

- `BucketARN`: o nome do recurso da Amazon (ARN) do bucket do Amazon S3 que contém o arquivo. Para saber o formato do ARN de um bucket do Amazon S3, consulte Nomes do recurso da Amazon (ARN) e namespaces do serviço da Amazon: Amazon Simple Storage Service (Amazon S3).
- `RoleARN`: o ARN de um perfil do IAM com a política `AmazonS3ReadOnlyAccess`. Para obter mais informações sobre como adicionar uma política a um perfil, consulte Modificação de um perfil.
- `FileKey`: o nome de arquivo do objeto.

Para gerar um esquema de um objeto do Amazon S3 usando a API **`DiscoverInputSchema`**

1. Verifique se você tem a AWS CLI configuração. Para obter mais informações, consulte Etapa 2: configurar o AWS Command Line Interface (AWS CLI) na seção Conceitos básicos.
2. Crie um arquivo denominado `data.csv` com o conteúdo a seguir:

```
year,month,state,producer_type,energy_source,units,consumption
2001,1,AK,TotalElectricPowerIndustry,Coal,ShortTons,47615
2001,1,AK,ElectricGeneratorsElectricUtilities,Coal,ShortTons,16535
2001,1,AK,CombinedHeatandPowerElectricPower,Coal,ShortTons,22890
2001,1,AL,TotalElectricPowerIndustry,Coal,ShortTons,3020601
2001,1,AL,ElectricGeneratorsElectricUtilities,Coal,ShortTons,2987681
```

3. Faça login no Amazon S3 em https://console.aws.amazon.com/s3/.
4. Crie um bucket do Amazon S3 e faça upload do arquivo `data.csv` que você criou. Anote o ARN do bucket criado. Para obter informações sobre como criar um bucket do Amazon S3 e fazer upload de um arquivo, consulte Conceitos básicos do Amazon Simple Storage Service.

5. Abra o console IAM em https://console.aws.amazon.com/iam/. Crie uma função com a política `AmazonS3ReadOnlyAccess`. Anote o ARN da nova função. Para obter informações sobre como criar um perfil, consulte Criação de um perfil para delegar permissões a um serviço Amazon. Para obter mais informações sobre como adicionar uma política a um perfil, consulte Modificação de um perfil.
6. Execute o seguinte `DiscoverInputSchema` comando no AWS CLI, substituindo os ARNs pelo bucket do Amazon S3 e pela função do IAM:

```
$aws kinesisanalytics discover-input-schema --s3-configuration '{ "RoleARN":
 "arn:aws:iam::123456789012:role/service-role/your-IAM-role", "BucketARN":
 "arn:aws:s3:::your-bucket-name", "FileKey": "data.csv" }'
```

7. A resposta é semelhante ao seguinte:

```
{
    "InputSchema": {
        "RecordEncoding": "UTF-8",
        "RecordColumns": [
            {
                "SqlType": "INTEGER",
                "Name": "COL_year"
            },
            {
                "SqlType": "INTEGER",
                "Name": "COL_month"
            },
            {
                "SqlType": "VARCHAR(4)",
                "Name": "state"
            },
            {
                "SqlType": "VARCHAR(64)",
                "Name": "producer_type"
            },
            {
                "SqlType": "VARCHAR(4)",
                "Name": "energy_source"
            },
            {
                "SqlType": "VARCHAR(16)",
                "Name": "units"
            },
```

```
            {
                "SqlType": "INTEGER",
                "Name": "consumption"
            }
        ],
        "RecordFormat": {
            "RecordFormatType": "CSV",
            "MappingParameters": {
                "CSVMappingParameters": {
                    "RecordRowDelimiter": "\r\n",
                    "RecordColumnDelimiter": ","
                }
            }
        }
    },
    "RawInputRecords": [
        "year,month,state,producer_type,energy_source,units,consumption
\r\n2001,1,AK,TotalElectricPowerIndustry,Coal,ShortTons,47615\r
\n2001,1,AK,ElectricGeneratorsElectricUtilities,Coal,ShortTons,16535\r
\n2001,1,AK,CombinedHeatandPowerElectricPower,Coal,ShortTons,22890\r
\n2001,1,AL,TotalElectricPowerIndustry,Coal,ShortTons,3020601\r
\n2001,1,AL,ElectricGeneratorsElectricUtilities,Coal,ShortTons,2987681"
    ],
    "ParsedInputRecords": [
        [
            null,
            null,
            "state",
            "producer_type",
            "energy_source",
            "units",
            null
        ],
        [
            "2001",
            "1",
            "AK",
            "TotalElectricPowerIndustry",
            "Coal",
            "ShortTons",
            "47615"
        ],
        [
            "2001",
```

```
            "1",
            "AK",
            "ElectricGeneratorsElectricUtilities",
            "Coal",
            "ShortTons",
            "16535"
        ],
        [
            "2001",
            "1",
            "AK",
            "CombinedHeatandPowerElectricPower",
            "Coal",
            "ShortTons",
            "22890"
        ],
        [
            "2001",
            "1",
            "AL",
            "TotalElectricPowerIndustry",
            "Coal",
            "ShortTons",
            "3020601"
        ],
        [
            "2001",
            "1",
            "AL",
            "ElectricGeneratorsElectricUtilities",
            "Coal",
            "ShortTons",
            "2987681"
        ]
    ]
}
```

Pré-processar dados usando uma função do Lambda

> **Note**
>
> Depois de 12 de setembro de 2023, você não poderá criar novos aplicativos usando o Kinesis Data Firehose como fonte se ainda não estiver usando o Kinesis Data Analytics for SQL. Para obter mais informações, consulte Limites.

Se os dados em seu stream precisarem de conversão, transformação, enriquecimento ou filtragem de formato, você poderá pré-processar os dados usando uma função. AWS Lambda Você pode fazer isso antes que o código SQL do aplicativo seja executado ou antes que seu aplicativo crie um esquema para seu streaming de dados.

Usar uma função do Lambda para pré-processamento de registros é útil nos seguintes cenários:

- Transformar registros de outros formatos (como KPL ou GZIP) em formatos que o Kinesis Data Analytics pode analisar. O Kinesis Data Analytics atualmente é compatível com formatos de dados JSON ou CSV.
- Expandir dados em um formato mais acessível para operações como agregação ou detecção de anomalias. Por exemplo, se vários valores de dados são armazenados juntos em uma string, você pode expandir os dados em colunas separadas.
- Enriquecer dados com outros serviços da Amazon, como extrapolação ou correção de erros.
- Aplicar transformação complexa de string em campos de registro.
- Filtrar dados para limpar os dados.

Usar uma função do Lambda para pré-processamento de registros

Ao criar o aplicativo do Kinesis Data Analytics, você habilita o pré-processamento do Lambda na página Conectar a uma fonte.

Para usar uma função do Lambda para pré-processar registros em um aplicativo do Kinesis Data Analytics

1. Faça login AWS Management Console e abra o console do Managed Service for Apache Flink em https://console.aws.amazon.com/kinesisanalytics.

2. Na página Conectar a uma fonte do seu aplicativo, escolha Habilitada na seção Registrar pré-processamento com AWS Lambda.
3. Para usar uma função do Lambda que você já tenha criado, escolha a função na lista suspensa Função do Lambda.
4. Para criar uma nova função do Lambda de um dos modelos de pré-processamento do Lambda, escolha o modelo na lista suspensa. Em seguida, escolha View template name > in Lambda (Exibir <nome do modelo> no Lambda) para editar a função.
5. Para criar uma nova função do Lambda, selecione Criar nova. Para obter informações sobre a criação de uma função Lambda, consulte Criar uma função HelloWorld Lambda e explorar o console no Guia do desenvolvedor.AWS Lambda
6. Escolha a versão da função do Lambda a ser usada. Para usar a versão mais recente, escolha $LATEST.

Quando você escolhe ou cria uma função do Lambda para pré-processamento de registros, os registros são pré-processados antes da execução do código SQL do seu aplicativo ou o aplicativo gera um esquema dos registros.

Permissões de pré-processamento do Lambda

Para usar o pré-processamento do Lambda, o perfil do IAM do aplicativo requer a seguinte política de permissões:

```
    {
      "Sid": "UseLambdaFunction",
      "Effect": "Allow",
      "Action": [
          "lambda:InvokeFunction",
          "lambda:GetFunctionConfiguration"
      ],
      "Resource": "<FunctionARN>"
  }
```

Métricas de pré-processamento do Lambda

Você pode usar CloudWatch a Amazon para monitorar o número de invocações do Lambda, bytes processados, sucessos e falhas, e assim por diante. Para obter informações sobre CloudWatch métricas emitidas pelo pré-processamento do Kinesis Data Analytics Lambda, consulte Métricas do Amazon Kinesis Analytics.

Usando AWS Lambda com a Kinesis Producer Library

A Kinesis Producer Library (KPL) agrega pequenos registros formatados pelo usuário em registros maiores de até 1 MB para usar melhor a throughput do Amazon Kinesis Data Streams. A Kinesis Client Library (KCL) para Java é compatível com a desagregação desses registros. No entanto, você deve usar um módulo especial para desagregar os registros ao usar AWS Lambda como consumidor de seus streams.

Para obter o código e as instruções do projeto necessários, consulte os Módulos de desagregação da Kinesis Producer Library para saber mais. AWS LambdaGitHub Você pode usar os componentes desse projeto para processar dados serializados KPL AWS Lambda em Java, Node.js e Python. Você também pode usar esses componentes como parte de um aplicativo da KCL de várias linguagens.

Modelo de dados de entrada de eventos de pré-processamento de dados ou modelo de resposta de registros

Para pré-processar registros, sua função do Lambda precisa estar em conformidade com os modelos de dados de entrada de eventos e modelos de resposta de registros exigidos.

Modelo de dados de entrada de eventos

O Kinesis Data Analytics lê continuamente os dados do seu stream de dados do Kinesis ou do stream de entrega do Firehose. Para cada lote de registros recuperado, o serviço gerencia como esse lote é enviado à sua função do Lambda. Sua função recebe uma lista de registros como entrada. Dentro da função, você segue a lista e aplica a lógica de negócios para cumprir requisitos de pré-processamento (como enriquecimento ou conversão de formato de dados).

O modelo de entrada para sua função de pré-processamento varia um pouco, dependendo se os dados foram recebidos de um stream de dados do Kinesis ou de um stream de entrega do Firehose.

Se a fonte for um stream de entrega do Firehose, o modelo de dados de entrada do evento será o seguinte:

Modelo de solicitação de dados do Kinesis Data Firehose

Campo	Descrição
`invocationId`	O Id de invocação do Lambda (GUID aleatório).

Campo	Descrição
`applicationArn`	O nome do recurso da Amazon (ARN) do aplicativo Kinesis Data Analytics
`streamArn`	ARN do fluxo de entrega
registros	

Campo	Descrição	
`recordId`	ID de registro (GUID aleatório)	
`kinesisFirehoseRecordMetadata`		

Campo	Descrição
`approximateArrivalTimestamp`	Tempo de chegada aproximado do registro do fluxo de entrega

`data`	Carga útil de registros de origem codificada em Base64	

O exemplo a seguir mostra a entrada de um fluxo de entrega do Firehose:

```
{
   "invocationId":"00540a87-5050-496a-84e4-e7d92bbaf5e2",
   "applicationArn":"arn:aws:kinesisanalytics:us-east-1:12345678911:application/lambda-
test",
   "streamArn":"arn:aws:firehose:us-east-1:AAAAAAAAAAAA:deliverystream/lambda-test",
   "records":[
      {
         "recordId":"49572672223665514422805246926656954630972486059535892482",
         "data":"aGVsbG8gd29ybGQ=",
         "kinesisFirehoseRecordMetadata":{
            "approximateArrivalTimestamp":1520280173
         }
      }
```

```
    ]
}
```

Se a origem for um fluxo de dados do Kinesis, o modelo de dados de entrada de eventos será o seguinte:

Modelo de solicitação de dados do Kinesis Streams

Campo	Descrição
`invocationId`	O Id de invocação do Lambda (GUID aleatório).
`applicationArn`	ARN do aplicativo Kinesis Data Analytics
`streamArn`	ARN do fluxo de entrega
registros	

Campo	Descrição
`recordId`	ID de registro baseado no número de sequência de registro do Kinesis
`kinesisStreamRecordMetadata`	

Campo	Descrição
`sequenceNumber`	Número de sequência do registro do stream do Kinesis
`partitionKey`	Chave de partição do registro do stream do Kinesis
`shardId`	`ShardId` do registro do fluxo do Kinesis

Campo	Descrição
Campo	Descrição
Campo	Descrição
Campo	Descrição
approximateArrivalTimestamp	Tempo de chegada aproximado do registro do fluxo de entrega
data	Carga útil de registros de origem codificada em Base64

O exemplo a seguir mostra a entrada de um streaming de dados do Kinesis:

```
{
  "invocationId": "00540a87-5050-496a-84e4-e7d92bbaf5e2",
  "applicationArn": "arn:aws:kinesisanalytics:us-east-1:12345678911:application/lambda-
test",
  "streamArn": "arn:aws:kinesis:us-east-1:AAAAAAAAAAAA:stream/lambda-test",
  "records": [
    {
      "recordId": "49572672223665514422805246926656954630972486059535892482",
      "data": "aGVsbG8gd29ybGQ=",
      "kinesisStreamRecordMetadata":{
            "shardId" :"shardId-000000000003",
            "partitionKey":"7400791606",

 "sequenceNumber":"49572672223665514422805246926656954630972486059535892482",
            "approximateArrivalTimestamp":1520280173
        }
    }
  ]
}
```

Modelo de resposta de registros

Todos os registros retornados da função de pré-processamento do Lambda (com IDs de registro) que são enviados para a função do Lambda devem ser retornados. Eles devem conter os seguintes parâmetros, ou o Kinesis Data Analytics os rejeita e aponta uma falha de pré-processamento de dados. A parte de carga útil de dados do registro pode ser transformada para cumprir requisitos de pré-processamento.

Modelo de resposta de dados

registros

Campo	Descrição
`recordId`	O ID do registro é transmitido do Kinesis Data Analytics para o Lambda durante a invocação. O registro transformado deve conter o mesmo ID de registro. Qualquer incompatibilidade entre o ID do registro original e o ID do registro transformado é considerada uma falha de pré-processamento de dados.
`result`	O status da transformação de dados do registro. Os valores possíveis são: • `Ok`: o registro foi transformado com êxito. O Kinesis Data Analytics ingere o registro para processamento de SQL. • `Dropped`: o registro foi eliminado intencionalmente por sua lógica de processamento. O Kinesis Data Analytics elimina o registro para processamento de SQL. O campo de carga útil de dados é opcional para um registro `Dropped`. • `ProcessingFailed` : não foi possível transformar o registro. O Kinesis Data Analytics considera uma falha no processamento pela função do Lambda e grava um erro no stream de erros. Para obter mais informações sobre o stream de erros, consulte Como tratar erros. O campo de carga útil de dados é opcional para um registro `ProcessingFailed` .

Campo	Descrição
`data`	A carga útil dos dados transformados, após a codificação base64. Cada carga de dados pode conter vários documento s JSON se o formato de dados de ingestão do aplicativo for JSON. Ou cada carga pode conter várias linhas CSV (com um delimitador de linha especificado em cada linha) se o formato de dados de ingestão do aplicativo for CSV. O serviço Kinesis Data Analytics analisa e processa dados com êxito com vários documentos JSON ou linhas CSV na mesma carga útil de dados.

O exemplo a seguir mostra a saída de uma função Lambda:

```
{
  "records": [
    {
      "recordId": "49572672223665514422805246926656954630972486059535892482",
      "result": "Ok",
      "data": "SEVMTE8gV09STEQ="
    }
  ]
}
```

Falhas de pré-processamento de dados comuns

Veja a seguir os motivos comuns pelos quais o pré-processamento pode falhar.

- Nem todos os registros (com IDs de registro) em um lote que são enviados para a função do Lambda são retornados para o serviço do Kinesis Data Analytics.
- A resposta não tem o ID do registro, o status ou o campo de carga útil de dados. O campo de carga útil de dados é opcional para um registro `Dropped` ou `ProcessingFailed`.
- O tempo limite da função do Lambda não é suficiente para pré-processar os dados.
- A resposta da função do Lambda excede os limites de resposta impostos pelo serviço AWS Lambda .

Para falhas de processamento de dados, o Kinesis Data Analytics continua a fazer novas tentativas de invocação do Lambda para o mesmo conjunto de registros até que tenha êxito. Você pode monitorar as CloudWatch métricas a seguir para obter informações sobre falhas.

- Aplicativo do Kinesis Data Analytics `MillisBehindLatest`: indica o tempo de atraso do aplicativo na leitura da origem de streaming.
- Métricas do `InputPreprocessing` CloudWatch aplicativo Kinesis Data Analytics: indica o número de sucessos e fracassos, entre outras estatísticas. Para obter mais informações, consulte Métricas do Amazon Kinesis Analytics.
- AWS Lambda CloudWatch métricas e registros de funções.

Criar funções do Lambda para pré-processamento

O aplicativo Amazon Kinesis Data Analytics pode usar funções do Lambda para pré-processamento de registros à medida que são ingeridos no aplicativo. O Kinesis Data Analytics fornece os modelos a seguir no console para usar como ponto de partida para pré-processamento de seus dados.

Tópicos

- Criar uma função do Lambda de pré-processamento em Node.js
- Criar uma função do Lambda de pré-processamento em Python
- Criar uma função do Lambda de pré-processamento em Java
- Criar uma função do Lambda de pré-processamento em .NET

Criar uma função do Lambda de pré-processamento em Node.js

Os modelos a seguir para criar uma função do Lambda de pré-processamento em Node.js estão disponíveis no console do Kinesis Data Analytics:

Lambda Blueprint	Linguagem e versão	Descrição
Processamento de entrada geral do Kinesis Data Analytics	Node.js 6.10	Um pré-processador de registros do Kinesis Data Analytics que recebe registros JSON ou CSV como entrada e os retorna com um status de processamento. Use esse processad or como ponto de partida para a lógica de transformação personalizada.

Lambda Blueprint	Linguagem e versão	Descrição
Processamento de entrada compactado	Node.js 6.10	Um processador de registros do Kinesis Data Analytics que recebe registros JSON ou CSV compactados (GZIP ou Deflate) como entrada e retorna registros descompactados com um status de processamento.

Criar uma função do Lambda de pré-processamento em Python

Os modelos a seguir para criar uma função do Lambda de pré-processamento em Python estão disponíveis no console:

Lambda Blueprint	Linguagem e versão	Descrição
Processamento de entrada geral do Kinesis Analytics	Python 2.7	Um pré-processador de registros do Kinesis Data Analytics que recebe registros JSON ou CSV como entrada e os retorna com um status de processamento. Use esse processad or como ponto de partida para a lógica de transformação personalizada.
Processamento de entrada do KPL	Python 2.7	Um processador de registros do Kinesis Data Analytics que recebe agregados de registros JSON ou CSV do Kinesis Producer Library (KPL) como entrada e retorna registros desagregados com um status de processam ento.

Criar uma função do Lambda de pré-processamento em Java

Para criar uma função do Lambda em Java para pré-processar registros, use as classes de eventos Java.

O código a seguir demonstra um exemplo de função do Lambda que pré-processa registros usando Java:

```
public class LambdaFunctionHandler implements
        RequestHandler<KinesisAnalyticsStreamsInputPreprocessingEvent,
 KinesisAnalyticsInputPreprocessingResponse> {

    @Override
    public KinesisAnalyticsInputPreprocessingResponse handleRequest(
            KinesisAnalyticsStreamsInputPreprocessingEvent event, Context context) {
        context.getLogger().log("InvocatonId is : " + event.invocationId);
        context.getLogger().log("StreamArn is : " + event.streamArn);
        context.getLogger().log("ApplicationArn is : " + event.applicationArn);

        List<KinesisAnalyticsInputPreprocessingResponse.Record> records = new
 ArrayList<KinesisAnalyticsInputPreprocessingResponse.Record>();
        KinesisAnalyticsInputPreprocessingResponse response = new
 KinesisAnalyticsInputPreprocessingResponse(records);

        event.records.stream().forEach(record -> {
            context.getLogger().log("recordId is : " + record.recordId);
            context.getLogger().log("record aat is :" +
 record.kinesisStreamRecordMetadata.approximateArrivalTimestamp);
             // Add your record.data pre-processing logic here.

            // response.records.add(new Record(record.recordId,
 KinesisAnalyticsInputPreprocessingResult.Ok, <preprocessedrecordData>));
        });
        return response;
    }

}
```

Criar uma função do Lambda de pré-processamento em .NET

Para criar uma função do Lambda em .NET para pré-processar registros, use as classes de eventos .NET.

O código a seguir demonstra um exemplo de função do Lambda que pré-processa registros usando C#:

```
public class Function
    {
        public KinesisAnalyticsInputPreprocessingResponse
 FunctionHandler(KinesisAnalyticsStreamsInputPreprocessingEvent evnt, ILambdaContext
 context)
```

```
        {
            context.Logger.LogLine($"InvocationId: {evnt.InvocationId}");
            context.Logger.LogLine($"StreamArn: {evnt.StreamArn}");
            context.Logger.LogLine($"ApplicationArn: {evnt.ApplicationArn}");

            var response = new KinesisAnalyticsInputPreprocessingResponse
            {
                Records = new List<KinesisAnalyticsInputPreprocessingResponse.Record>()
            };

            foreach (var record in evnt.Records)
            {
                context.Logger.LogLine($"\tRecordId: {record.RecordId}");
                context.Logger.LogLine($"\tShardId: {record.RecordMetadata.ShardId}");
                context.Logger.LogLine($"\tPartitionKey:
{record.RecordMetadata.PartitionKey}");
                context.Logger.LogLine($"\tRecord ApproximateArrivalTime:
{record.RecordMetadata.ApproximateArrivalTimestamp}");
                context.Logger.LogLine($"\tData: {record.DecodeData()}");

                // Add your record preprocessig logic here.

                var preprocessedRecord = new
KinesisAnalyticsInputPreprocessingResponse.Record
                {
                    RecordId = record.RecordId,
                    Result = KinesisAnalyticsInputPreprocessingResponse.OK
                };
                preprocessedRecord.EncodeData(record.DecodeData().ToUpperInvariant());
                response.Records.Add(preprocessedRecord);
            }
            return response;
        }
    }
```

Para obter mais informações sobre como criar funções do Lambda para pré-processamento e destinos em .NET, consulte Amazon.Lambda.KinesisAnalyticsEvents.

Paralelização dos streams de entrada para aumentar a taxa de transferência

> **Note**
>
> Depois de 12 de setembro de 2023, você não poderá criar novos aplicativos usando o Kinesis Data Firehose como fonte se ainda não estiver usando o Kinesis Data Analytics for SQL. Para obter mais informações, consulte Limites.

Os aplicativos do Amazon Kinesis Data Analytics podem oferecer suporte a vários streams de entrada no aplicativo para escalar um aplicativo além do throughput de um único stream de entrada no aplicativo. Para obter mais informações sobre streams de entrada no aplicativo, consulte Amazon Kinesis Data Analytics para aplicativos SQL: como funciona.

Em quase todos os casos, o Amazon Kinesis Data Analytics escala seu aplicativo para lidar com a capacidade dos streams do Kinesis ou dos streams de origem do Firehose que alimentam seu aplicativo. No entanto, se a taxa de transferência do stream de origem exceder a taxa de transferência de um único stream de entrada no aplicativo, aumente explicitamente o número de streams de entrada no aplicativo usado pelo aplicativo. Isso é feito com o parâmetro `InputParallelism`.

Quando o parâmetro `InputParallelism` for maior que um, o Amazon Kinesis Data Analytics divide uniformemente as partições do stream de origem entre os streams no aplicativo. Por exemplo, se o fluxo de origem tiver 50 estilhaços e você definiu `InputParallelism` para 2, cada fluxo de entrada no aplicativo receberá a entrada de 25 estilhaços do fluxo de origem.

Quando você aumentar o número de streams no aplicativo, este deverá acessar os dados em cada stream explicitamente. Para obter informações sobre como acessar vários fluxos no aplicativo no seu código, consulte Acesso a streams no aplicativo separados no aplicativo do Amazon Kinesis Data Analytics.

Embora os fragmentos de stream do Kinesis Data Streams e do Firehose estejam divididos entre os streams no aplicativo da mesma forma, eles diferem na forma como aparecem no seu aplicativo:

- Os registros de um fluxo de dados do Kinesis incluem um campo `shard_id` que pode ser usado para identificar o fragmento de origem do registro.

- Os registros de um stream de entrega do Firehose não incluem um campo que identifique o fragmento ou a partição de origem do registro. Isso ocorre porque o Firehose abstrai essas informações do seu aplicativo.

Avaliações para saber se o número de streams de entrada no aplicativo deve ser aumentado

Na maioria dos casos, um único stream de entrada no aplicativo pode lidar com a taxa de transferência de um único stream de origem, de acordo com a complexidade e o volume de dados dos streams de entrada. Para determinar se você precisa aumentar o número de fluxos de entrada no aplicativo, você pode monitorar as `MillisBehindLatest` métricas `InputBytes` e na Amazon. CloudWatch

Se a métrica `InputBytes` for maior que 100 MB/s (ou se você antecipar que ela será maior que essa taxa), isso poderá causar um aumento em `MillisBehindLatest` e ampliar o impacto de problemas do aplicativo. Para resolver essa situação, recomendamos que sejam feitas as seguintes escolhas de linguagem para o aplicativo:

- Use vários fluxos e aplicativos Kinesis Data Analytics para SQL se o aplicativo tiver necessidades de escalar além de 100 MB/segundo.
- Use Kinesis Data Analytics para aplicativos Java se quiser usar um único fluxo e aplicativo.

Se a métrica `MillisBehindLatest` tiver uma das seguintes características, aumente a configuração `InputParallelism` do seu aplicativo:

- A métrica `MillisBehindLatest` aumentar gradualmente, indicando que o aplicativo está atrasado em relação aos dados mais recentes no stream.
- A métrica `MillisBehindLatest` estiver consistentemente acima de 1.000 (um segundo).

Não será preciso aumentar a configuração `InputParallelism` do aplicativo se:

- A métrica `MillisBehindLatest` diminuir gradualmente, indicando que o aplicativo está alcançando os dados mais recentes no stream.
- A métrica `MillisBehindLatest` estiver abaixo de 1.000 (um segundo).

Para obter mais informações sobre o uso CloudWatch, consulte o Guia CloudWatch do usuário.

Implementação de vários streams de entrada no aplicativo

Defina o número de streams de entrada no aplicativo quando um aplicativo for criado usando CreateApplication. Defina esse número após criar um aplicativo usando UpdateApplication.

> **Note**
>
> Você só pode definir a configuração `InputParallelism` usando a API do Amazon Kinesis Data Analytics ou o. AWS CLIVocê não pode definir essa configuração usando AWS Management Consoleo. Para obter informações sobre como configurar o AWS CLI, consulteEtapa 2: configurar o AWS Command Line Interface (AWS CLI).

Configuração da contagem do stream de entrada de um aplicativo novo

O exemplo a seguir demonstra como usar a ação da API `CreateApplication` para definir a contagem do stream de entrada de um aplicativo novo para 2.

Para obter mais informações sobre o `CreateApplication`, consulte CreateApplication.

```
{
   "ApplicationCode": "<The SQL code the new application will run on the input
 stream>",
   "ApplicationDescription": "<A friendly description for the new application>",
   "ApplicationName": "<The name for the new application>",
   "Inputs": [
    {
      "InputId": "ID for the new input stream",
      "InputParallelism": {
        "Count": 2
    }],
   "Outputs": [ ... ],
 }]
}
```

Configuração da contagem do stream de entrada de um aplicativo existente

O exemplo a seguir demonstra como usar a ação da API `UpdateApplication` para definir a contagem do stream de entrada de um aplicativo existente para 2.

Para obter mais informações sobre o `Update_Application`, consulte UpdateApplication.

```
{
   "InputUpdates": [
      {
         "InputId": "yourInputId",
         "InputParallelismUpdate": {
            "CountUpdate": 2
         }
      }
   ],
}
```

Acesso a streams no aplicativo separados no aplicativo do Amazon Kinesis Data Analytics

Para usar vários streams de entrada no aplicativo em seu aplicativo, selecione explicitamente entre streams diferentes. O exemplo de código a seguir demonstra como consultar vários streams de entrada no aplicativo criado no tutorial de Conceitos básicos.

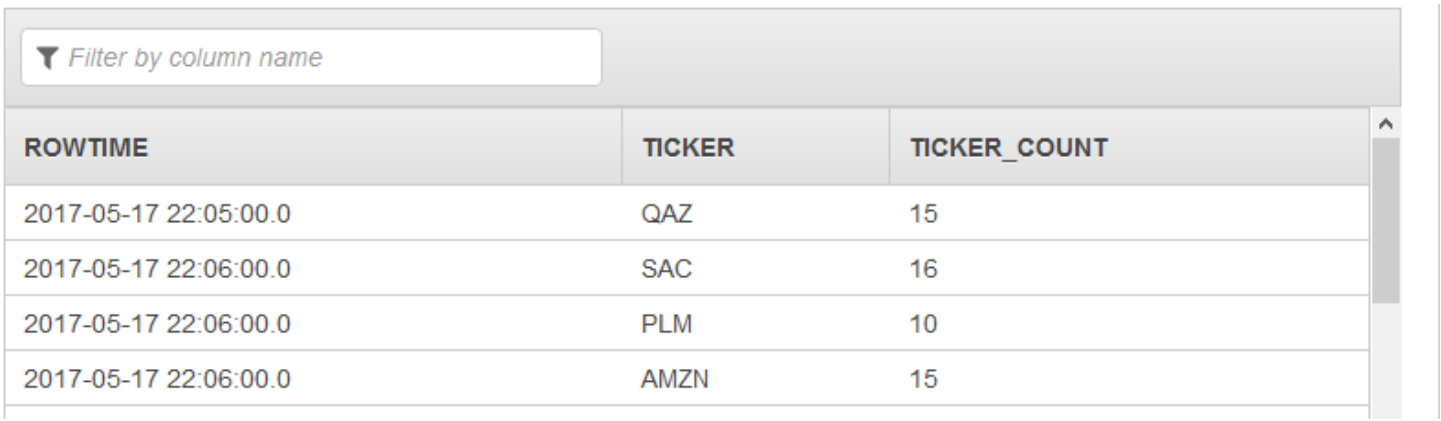
No exemplo a seguir, cada fluxo de origem é agregado usando COUNT antes de serem combinados em um único fluxo no aplicativo chamado `in_application_stream001`. Agregar streams de origem de antemão ajuda a garantir que o stream no aplicativo combinado lide com o tráfego de vários streams sem ficar sobrecarregado.

> **Note**
>
> Para executar esse exemplo e obter resultados de ambos os fluxos de entrada no aplicativo, atualize o número de estilhaços do fluxo de origem e o parâmetro `InputParallelism` do aplicativo.

```
CREATE OR REPLACE STREAM in_application_stream_001 (
    ticker VARCHAR(64),
    ticker_count INTEGER
);

CREATE OR REPLACE PUMP pump001 AS
INSERT INTO in_application_stream_001
SELECT STREAM ticker_symbol, COUNT(ticker_symbol)
FROM source_sql_stream_001
GROUP BY STEP(source_sql_stream_001.rowtime BY INTERVAL '60' SECOND),
```

```
    ticker_symbol;

CREATE OR REPLACE PUMP pump002 AS
INSERT INTO in_application_stream_001
SELECT STREAM ticker_symbol, COUNT(ticker_symbol)
FROM source_sql_stream_002
GROUP BY STEP(source_sql_stream_002.rowtime BY INTERVAL '60' SECOND),
    ticker_symbol;
```

O exemplo de código anterior produz saída em `in_application_stream001` semelhante a:



Considerações adicionais

Ao usar vários streams de entrada, esteja ciente do seguinte:

- O número máximo de streams de entrada no aplicativo é 64.
- Os streams de entrada no aplicativo são distribuídos uniformemente entre os estilhaços do stream de entrada do aplicativo.
- Os ganhos de desempenho da adição de streams no aplicativo não escalam de forma linear. Ou seja, dobrar o número de streams no aplicativo não dobra a taxa de transferência. Com um tamanho de linha típico, cada stream no aplicativo pode alcançar a taxa de transferência de aproximadamente 5.000 a 15.000 linhas por segundo. Ao aumentar a contagem de streams no aplicativo para 10, é possível alcançar uma taxa de transferência de 20.000 a 30.000 linhas por segundo. A velocidade da taxa de transferência depende da contagem, tipos de dados e volume de dados de campos no fluxo de entrada.
- Algumas funções de agregação (como AVG) podem produzir resultados inesperados quando aplicadas a fluxos de entrada particionados em diferentes estilhaços. Como é necessário executar

a operação agregada em estilhaços individuais combinando-os em um stream agregado, os resultados podem ser ponderados em relação ao stream que contiver mais registros.

- Se o aplicativo continua a apresentar performance insatisfatória (indicado por uma métrica `MillisBehindLatest` elevada) depois que você aumentou o número de streams de entrada, você pode ter atingido o limite de unidades de processamento do Kinesis (KPUs). Para ter mais informações, consulte Escalabilidade automática de aplicativos para aumentar a taxa de transferência.

Código do aplicativo

Código do aplicativo é uma série de instruções SQL que processam entrada e produzem saída. Essas instruções SQL operam em streamings no aplicativo e tabelas de referência. Para obter mais informações, consulte Amazon Kinesis Data Analytics para aplicativos SQL: como funciona.

Para obter mais informações sobre os elementos da linguagem SQL compatíveis com o Kinesis Data Analytics, consulte Referência SQL do Amazon Kinesis Data Analytics.

Em bancos de dados relacionais, você trabalha com tabelas, usando instruções INSERT para adicionar os registros e a instrução SELECT para consultar os dados. No Amazon Kinesis Data Analytics, você trabalha com streams. É possível gravar uma instrução SQL para consultar esses streams. Os resultados da consulta de um stream no aplicativo são sempre enviados para outro aplicativo no stream. Ao executar análises complexas, você pode criar vários streams no aplicativo para manter os resultados de análises intermediárias. E, por fim, você configura a saída do aplicativo para manter os resultados da análise final (de um ou mais streams no aplicativo) em destinos externos. Resumindo, o padrão a seguir é comum na gravação do código do aplicativo:

- A instrução SELECT é sempre usada no contexto de uma instrução INSERT. Ou seja, ao selecionar linhas, você insere resultados em outro stream no aplicativo.
- A instrução INSERT é sempre usada no contexto de uma bomba. Ou seja, você usa bombas para gravar em um stream no aplicativo.

O código do aplicativo de exemplo a seguir lê registros de um stream no aplicativo (SOURCE_SQL_STREAM_001) e o grava em outro stream no aplicativo (DESTINATION_SQL_STREAM no aplicativo). É possível inserir registros em streams no aplicativo usando bombas, como mostrado a seguir:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (ticker_symbol VARCHAR(4),
                                                   change DOUBLE,
                                                   price DOUBLE);
-- Create a pump and insert into output stream.
CREATE OR REPLACE PUMP "STREAM_PUMP" AS

  INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM ticker_symbol, change,price
    FROM   "SOURCE_SQL_STREAM_001";
```

> **Note**
>
> Os identificadores que você especifica para nomes de stream e nomes de coluna seguem as convenções padrão do SQL. Por exemplo, se você colocar um identificador entre aspas, ele diferenciará maiúsculas de minúsculas. Caso contrário, o identificador será padronizado com maiúsculas. Para obter mais informações sobre identificadores, consulte Identificadores no Amazon Managed Service for Apache Flink SQL Reference.

O código do aplicativo pode consistir em muitas instruções SQL. Por exemplo:

- É possível gravar consultas SQL de forma sequencial com o resultado de uma instrução SQL contribuindo para a próxima instrução SQL.
- Você também pode gravar consultas SQL para execução independente uma da outra. Por exemplo, gravar duas instruções SQL que consultam o mesmo stream no aplicativo, mas que enviam a saída para streams no aplicativo diferentes. Você pode então consultar os streams no aplicativo recém-criados de modo independente.

É possível criar streams no aplicativo para salvar resultados intermediários. Insira os dados em streams no aplicativo usando bombas. Para obter mais informações, consulte Bombas e fluxos no aplicativo.

Se você adicionar uma tabela de referência no aplicativo, poderá gravar no SQL para associar dados nos fluxos de aplicativo e tabelas de referência. Para obter mais informações, consulte Exemplo: adição de dados de referência a um aplicativo do Kinesis Data Analytics.

De acordo com a configuração de saída do aplicativo, o Amazon Kinesis Data Analytics grava dados de streams no aplicativo específicos para o destino externo conforme a configuração de saída do

aplicativo. Certifique-se de gravar o código do aplicativo nos streams no aplicativo especificados na configuração de saída.

Para obter mais informações, consulte os tópicos a seguir:

- Conceitos de streaming com SQL
- Referência SQL do Amazon Kinesis Data Analytics

Configuração da saída do aplicativo

No código do aplicativo, você grava a saída das instruções SQL em um ou mais fluxos de aplicativo. Opcionalmente, você pode adicionar uma configuração de saída ao seu aplicativo para manter tudo gravado em um stream no aplicativo em um destino externo, como um stream de dados do Amazon Kinesis, um stream de entrega do Firehose ou uma função. AWS Lambda

Há um limite para o número de destinos externos que você pode usar para manter uma saída de aplicativo. Para ter mais informações, consulte Limites.

> **Note**
>
> É recomendável que você use um destino externo no qual manterá os dados do fluxo de erros de aplicativo, para que você possa investigar os erros.

Em cada uma dessas configurações de saída, você fornece o seguinte:

- Nome do fluxo de aplicativo – O fluxo que você deseja manter em um destino externo.

  O Kinesis Data Analytics busca o fluxo no aplicativo especificado na configuração de saída. (O nome do fluxo faz distinção entre maiúsculas e minúsculas e deve corresponder exatamente.) Verifique se o código do aplicativo está criando esse fluxo de aplicativo.
- Destino externo — você pode persistir os dados em um stream de dados do Kinesis, um stream de entrega do Firehose ou uma função Lambda. Forneça o nome de recurso da Amazon (ARN) do fluxo ou da função. Forneça também um perfil do IAM que o Kinesis Data Analytics possa assumir para gravar no fluxo ou na função em seu nome. Descreva o formato do registro (JSON, CSV) para que o Kinesis Data Analytics use durante a gravação no destino externo.

Se o serviço do Kinesis Data Analytics não conseguir gravar no destino de streaming ou do Lambda, ele continuará tentando por tempo indeterminado. Isso cria uma pressão contrária, fazendo com que o aplicativo fique para trás. Se isso problema não for resolvido, o aplicativo acabará interrompendo o processamento de novos dados. Você pode monitorar as métricas do Kinesis Data Analytics e definir alarmes para falhas. Para obter mais informações sobre métricas e alarmes, consulte Usando CloudWatch métricas da Amazon e Criação de CloudWatch alarmes da Amazon.

Você pode configurar a saída do aplicativo usando o AWS Management Console. O console faz com que a chamada de API salve a configuração.

Criando uma saída usando o AWS CLI

Esta seção descreve como criar a seção `Outputs` do corpo de uma solicitação para uma operação `CreateApplication` ou `AddApplicationOutput`.

Criação de uma saída do Kinesis Stream

O fragmento JSON a seguir mostra a seção `Outputs` no corpo da solicitação de `CreateApplication` para a criação de um destino de fluxo de dados do Amazon Kinesis.

```
"Outputs": [
   {
       "DestinationSchema": {
           "RecordFormatType": "string"
       },
       "KinesisStreamsOutput": {
           "ResourceARN": "string",
           "RoleARN": "string"
       },
       "Name": "string"
   }

]
```

Criando uma saída de fluxo de entrega do Firehose

O fragmento JSON a seguir mostra a `Outputs` seção no corpo da `CreateApplication` solicitação para criar um destino de stream de entrega do Amazon Data Firehose.

```
"Outputs": [
   {
```

```
        "DestinationSchema": {
            "RecordFormatType": "string"
        },
        "KinesisFirehoseOutput": {
            "ResourceARN": "string",
            "RoleARN": "string"
        },
        "Name": "string"
    }
]
```

Criação de saída de uma função do Lambda

O fragmento JSON a seguir mostra a `Outputs` seção no corpo da `CreateApplication` solicitação para criar um destino de AWS Lambda função.

```
"Outputs": [
    {
        "DestinationSchema": {
            "RecordFormatType": "string"
        },
        "LambdaOutput": {
            "ResourceARN": "string",
            "RoleARN": "string"
        },
        "Name": "string"
    }
]
```

Como usar a função do Lambda como saída

Usar AWS Lambda como destino permite que você realize mais facilmente o pós-processamento de seus resultados de SQL antes de enviá-los para um destino final. Algumas tarefas comuns de pós-processamento são:

- Agregação de várias linhas em um único registro
- Combinação de resultados atuais com resultados passados para o tratamento de dados retardatários
- Entrega em destinos diferentes com base no tipo de informações
- Conversão do formato de registros (como a conversão para Protobuf)

- Manipulação ou transformação de strings
- Enriquecimento de dados após o processamento analítico
- Processamento personalizado para casos de uso geoespacial
- Criptografia de dados

As funções Lambda podem fornecer informações analíticas para uma variedade de AWS serviços e outros destinos, incluindo os seguintes:

- Amazon Simple Storage Service (Amazon S3)
- APIs personalizadas
- Amazon DynamoDB
- Apache Aurora
- Amazon Redshift
- Amazon Simple Notification Service (Amazon SNS)
- Amazon Simple Queue Service (Amazon SQS)
- Amazon CloudWatch

Para obter mais informações sobre como criar aplicativos do Lambda, consulte Conceitos básicos do AWS Lambda.

Tópicos

- Permissões para o Lambda como saída
- Lambda como métrica de saída
- Modelo de dados de entrada de eventos de saída do Lambda e modelo de resposta de registros
- Frequência de invocação de saída do Lambda
- Adição de uma função do Lambda para usar como uma saída
- Falhas comuns de Lambda como saída
- Como criar funções do Lambda para destinos de aplicativos

Permissões para o Lambda como saída

Para usar o Lambda como saída, o perfil do IAM de saída do Lambda do aplicativo requer a seguinte política de permissões:

```
{
   "Sid": "UseLambdaFunction",
   "Effect": "Allow",
   "Action": [
       "lambda:InvokeFunction",
       "lambda:GetFunctionConfiguration"
   ],
   "Resource": "FunctionARN"
}
```

Lambda como métrica de saída

Você usa CloudWatch a Amazon para monitorar o número de bytes enviados, sucessos e falhas, e assim por diante. Para obter informações sobre CloudWatch métricas emitidas pelo Kinesis Data Analytics usando o Lambda como saída, consulte Métricas do Amazon Kinesis Analytics.

Modelo de dados de entrada de eventos de saída do Lambda e modelo de resposta de registros

Para enviar registros de saída do Kinesis Data Analytics, sua função do Lambda precisa ser compatível com os modelos de dados de entrada de eventos e modelos de resposta de registros exigidos.

Modelo de dados de entrada de eventos

O Kinesis Data Analytics envia continuamente os registros de saída do aplicativo para o Lambda como uma função de saída com o seguinte modelo de solicitação. Dentro da função, você percorre a lista e aplica a lógica de negócios para cumprir os requisitos de saída (como a transformação dos dados antes de enviá-los para um destino final).

Campo	Descrição
`invocationId`	ID de invocação do Lambda (GUID aleatório)
`applicationArn`	O nome do recurso da Amazon (ARN) do aplicativo Kinesis Data Analytics
registros	

Campo	Descrição
Campo	Descrição
`recordId`	ID de registro (GUID aleatório)
`lambdaDeliveryRecordMetadata`	Campo: `retryHint` — Descrição: Número de novas tentativas de entrega
data	Carga útil de registros de saída codificada em Base64

> **Note**
>
> O `retryHint` é um valor que aumenta a cada falha de entrega. Esse valor não é mantido permanentemente, e será redefinido se o aplicativo for interrompido.

Modelo de resposta de registros

Cada registro enviado para o Lambda como uma função de saída (com IDs de registro) deve ser reconhecido com `Ok` ou `DeliveryFailed`, e deve conter os seguintes parâmetros. Caso contrário, o Kinesis Data Analytics os tratará como uma falha de entrega.

registros

Campo	Descrição
`recordId`	O ID do registro é transmitido do Kinesis Data Analytics para o Lambda durante a invocação. Qualquer discrepância entre o ID do registro original e o ID do registro reconhecido é considerada como uma falha de entrega.

Campo	Descrição
`result`	O status da entrega do registro. Os valores possíveis são: • `Ok`: o registro foi transformado com êxito e enviado para o destino final. O Kinesis Data Analytics ingere o registro para processamento de SQL. • `DeliveryFailed` : o registro não foi entregue ao destino final pelo Lambda como função de saída. O Kinesis Data Analytics faz continuamente novas tentativas de entrega dos registros com falha para o Lambda como função de saída.

Frequência de invocação de saída do Lambda

Um aplicativo Kinesis Data Analytics armazena os registros de saída em buffers e invoca a função de destino do AWS Lambda com frequência.

- Se os registros forem emitidos para o fluxo de destino no aplicativo de análise de dados como uma janela suspensa, a função de AWS Lambda destino será invocada por gatilho da janela intermitente. Por exemplo, se uma janela em cascata de 60 segundos é usada para emitir os registros para o fluxo no aplicativo de destino, a função do Lambda é invocada uma vez a cada 60 segundos.
- Se os registros são emitidos para o fluxo no aplicativo de destino dentro do aplicativo como uma consulta contínua ou uma janela deslizante, a função de destino do Lambda é invocada aproximadamente uma vez por segundo.

> **Note**
>
> Os limites de tamanho da carga de solicitação por cada invocação da função do Lambda se aplicam. Exceder esses limites resulta em registros de saída divididos e enviados em várias chamadas da função do Lambda.

Adição de uma função do Lambda para usar como uma saída

O procedimento a seguir demonstra como adicionar uma função do Lambda como saída em um aplicativo Kinesis Data Analytics.

1. Faça login AWS Management Console e abra o console do Managed Service for Apache Flink em https://console.aws.amazon.com/kinesisanalytics.
2. Escolha o aplicativo na lista e, em seguida, escolha Application details.
3. Na seção Destination, escolha Connect new destination.
4. Para o item Destination (Destino), escolha Function (Função do)AWS Lambda .
5. Na seção Entregar registros para AWS Lambda, escolha uma função e versão existentes do Lambda ou selecione Criar novo.
6. Se você estiver criando uma nova função do Lambda, faça o seguinte:
   a. Escolha um dos modelos fornecidos. Para obter mais informações, Como criar funções do Lambda para destinos de aplicativos.
   b. A página Create Function (Criar função) é aberta em uma nova guia do navegador. Na caixa Name (Nome), atribua um nome significativo à função (por exemplo, **`myLambdaFunction`**).
   c. Atualize o modelo com a funcionalidade de pós-processamento para o seu aplicativo. Para obter informações sobre como criar uma função do Lambda, consulte Conceitos básicos no Guia do desenvolvedor doAWS Lambda .
   d. No console do Kinesis Data Analytics, na lista de funções do Lambda, escolha a função do Lambda que você acabou de criar. Selecione $LATEST para a versão da função do Lambda.
7. Na seção In-application stream, escolha Choose an existing in-application stream. Em In-application stream name, escolha o fluxo de saída do seu aplicativo. Os resultados do fluxo de saída selecionado são enviadas para a função de saída do Lambda.
8. Deixe o restante do formulário com os valores padrão e escolha Save and continue.

O aplicativo agora envia registros do fluxo do aplicativo para a sua função do Lambda. Você pode ver os resultados do modelo padrão no CloudWatch console da Amazon. Monitore a métrica `AWS/KinesisAnalytics/LambdaDelivery.OkRecords` para ver o número de registros que estão sendo entregues à função do Lambda.

Falhas comuns de Lambda como saída

Veja a seguir os motivos comuns pelos quais a entrega a uma função do Lambda pode falhar.

- Nem todos os registros (com IDs de registro) em um lote que são enviados para a função do Lambda são retornados para o serviço do Kinesis Data Analytics.
- A resposta não tem o ID do registro ou o campo de status.
- Os tempos limite da função do Lambda não são suficientes para executar a lógica de negócios na função do Lambda.
- A lógica de negócios da função do Lambda não detecta todos os erros, o que resulta na extrapolação do tempo limite e em uma pressão contrária devido à ocorrência de exceções não processadas. Essas são geralmente conhecidas como mensagens "poison pill".

Para falhas de entrega, o Kinesis Data Analytics continua a fazer novas tentativas de invocação do Lambda para o mesmo conjunto de registros até que tenha êxito. Para obter informações sobre falhas, você pode monitorar as seguintes CloudWatch métricas:

- Lambda do aplicativo Kinesis Data Analytics como métricas CloudWatch de saída: indica o número de sucessos e falhas, entre outras estatísticas. Para obter mais informações, consulte Métricas do Amazon Kinesis Analytics.
- AWS Lambda CloudWatch métricas e registros de funções.

Como criar funções do Lambda para destinos de aplicativos

Seu aplicativo Kinesis Data Analytics pode AWS Lambda usar funções como saída. O Kinesis Data Analytics fornece modelos para a criação de funções do Lambda a serem usadas como um destino para seus aplicativos. Use esses modelos como ponto de partida para o pós-processamento da saída de seu aplicativo.

Tópicos

- Criar uma função do Lambda de destino em Node.js
- Criar uma função do Lambda de destino em Python
- Criar uma função do Lambda de destino em Java
- Criar uma função do Lambda de destino em .NET

Criar uma função do Lambda de destino em Node.js

O modelo a seguir para criar uma função do Lambda de destino em Node.js está disponível no console:

Lambda como esquema de saída	Linguagem e versão	Descrição
`kinesis-analytics-output`	Node.js 12.x	Entrega registros de saída de um aplicativo do Kinesis Data Analytics em um destino personalizado.

Criar uma função do Lambda de destino em Python

Os modelos a seguir para criar uma função do Lambda de destino em Python estão disponíveis no console:

Lambda como esquema de saída	Linguagem e versão	Descrição
`kinesis-analytics-output-sns`	Python 2.7	Entrega registros de saída de um aplicativo Kinesis Data Analytics para o Amazon SNS.
`kinesis-analytics-output-ddb`	Python 2.7	Entrega registros de saída de um aplicativo Kinesis Data Analytics para o Amazon DynamoDB.

Criar uma função do Lambda de destino em Java

Para criar uma função do Lambda de destino em Java, use as classes de Eventos Java.

O código a seguir demonstra um exemplo de função do Lambda de destino usando Java:

```
public class LambdaFunctionHandler
```

```
        implements RequestHandler<KinesisAnalyticsOutputDeliveryEvent,
 KinesisAnalyticsOutputDeliveryResponse> {

    @Override
    public KinesisAnalyticsOutputDeliveryResponse
 handleRequest(KinesisAnalyticsOutputDeliveryEvent event,
            Context context) {
        context.getLogger().log("InvocatonId is : " + event.invocationId);
        context.getLogger().log("ApplicationArn is : " + event.applicationArn);

        List<KinesisAnalyticsOutputDeliveryResponse.Record> records = new
 ArrayList<KinesisAnalyticsOutputDeliveryResponse.Record>();
        KinesisAnalyticsOutputDeliveryResponse response = new
 KinesisAnalyticsOutputDeliveryResponse(records);

        event.records.stream().forEach(record -> {
            context.getLogger().log("recordId is : " + record.recordId);
            context.getLogger().log("record retryHint is :" +
 record.lambdaDeliveryRecordMetadata.retryHint);
            // Add logic here to transform and send the record to final destination of
 your choice.
            response.records.add(new Record(record.recordId,
 KinesisAnalyticsOutputDeliveryResponse.Result.Ok));
        });
        return response;
    }

}
```

Criar uma função do Lambda de destino em .NET

Para criar uma função do Lambda de destino em .NET, use as classes de eventos .NET.

O código a seguir demonstra um exemplo de função do Lambda de destino usando C#:

```
public class Function
    {
        public KinesisAnalyticsOutputDeliveryResponse
 FunctionHandler(KinesisAnalyticsOutputDeliveryEvent evnt, ILambdaContext context)
        {
            context.Logger.LogLine($"InvocationId: {evnt.InvocationId}");
            context.Logger.LogLine($"ApplicationArn: {evnt.ApplicationArn}");
```

```
            var response = new KinesisAnalyticsOutputDeliveryResponse
            {
                Records = new List<KinesisAnalyticsOutputDeliveryResponse.Record>()
            };

            foreach (var record in evnt.Records)
            {
                context.Logger.LogLine($"\tRecordId: {record.RecordId}");
                context.Logger.LogLine($"\tRetryHint:
{record.RecordMetadata.RetryHint}");
                context.Logger.LogLine($"\tData: {record.DecodeData()}");

                // Add logic here to send to the record to final destination of your
choice.

                var deliveredRecord = new KinesisAnalyticsOutputDeliveryResponse.Record
                {
                    RecordId = record.RecordId,
                    Result = KinesisAnalyticsOutputDeliveryResponse.OK
                };
                response.Records.Add(deliveredRecord);
            }
            return response;
        }
    }
```

Para obter mais informações sobre como criar funções do Lambda para pré-processamento e destinos em .NET, consulte [Amazon.Lambda.KinesisAnalyticsEvents].

Modelo de entrega para manter a saída do aplicativo em um destino externo

O Amazon Kinesis Data Analytics usa o modelo de entrega "ao menos uma vez" para a saída de aplicativo nos destinos configurados. Quando um aplicativo está em execução, o Kinesis Data Analytics usa pontos de verificação internos. Esses pontos de verificação são momentos específicos nos quais os registros de saída foram entregues aos destinos sem perda de dados. O serviço usa os pontos de verificação, conforme forem necessários, para garantir que a saída do seu aplicativo seja entregue pelo menos uma vez aos destinos configurados.

Em uma situação normal, seu aplicativo processa os dados recebidos continuamente. O Kinesis Data Analytics grava a saída nos destinos configurados, como um stream de dados do Kinesis ou um

stream de entrega do Firehose. No entanto, o aplicativo pode ser interrompido ocasionalmente; por exemplo:

- Você opta por interromper o aplicativo e reiniciá-lo mais tarde.
- Você exclui a função do perfil do IAM de que o Kinesis Data Analytics precisa para gravar a saída do aplicativo no destino configurado. Sem a função do perfil do IAM, o Kinesis Data Analytics não tem permissões para gravar conteúdo no destino externo em seu nome.
- Uma falha da rede ou outra falha de serviço interna faz com que a execução do aplicativo seja interrompida momentaneamente.

Quando o aplicativo for reiniciado, o Kinesis Data Analytics garantirá que ele continuará processando e gravando a saída de um ponto anterior ou igual ao momento em que a falha ocorreu. Isso ajuda a garantir que ele não perderá a entrega de qualquer saída do aplicativo para os destinos configurados.

Suponha que você tenha configurado vários destinos do mesmo fluxo de aplicativo. Depois que o aplicativo se recuperar da falha, o Kinesis Data Analytics continuará mantendo a saída nos destinos configurados a partir do último registro entregue no destino mais lento. Isso pode fazer com que o mesmo registro de saída seja entregue mais de uma vez nos outros destinos. Nesse caso, você deverá lidar com as duplicações potenciais no destino externamente.

Como tratar erros

O Amazon Kinesis Data Analytics retorna erros de API ou SQL diretamente para você. Para obter mais informações sobre as operações de API, consulte Ações. Para obter mais informações sobre o tratamento de erros do SQL, consulte Referência SQL do Amazon Kinesis Data Analytics.

O Amazon Kinesis Data Analytics reporta erros de runtime usando um stream de erros de aplicativo chamado `error_stream`.

Geração de relatórios de erros por meio de um fluxo de erros de aplicativo

O Amazon Kinesis Data Analytics reporta erros de runtime para o stream de erros de aplicativo chamado `error_stream`. Estes são exemplos de erros que podem ocorrer:

- A leitura de registro na origem do streaming não está de acordo com o esquema de entrada.
- O código do aplicativo especifica a divisão por zero.

- As linhas estão fora de ordem (por exemplo, um registro aparece no fluxo com o valor `ROWTIME` modificado por um usuário, o que faz com que o registro fique fora de ordem).
- Os dados no fluxo de origem não podem ser convertidos no tipo de dados especificado no esquema (erro de coerção). Para obter informações sobre quais tipos de dados podem ser convertidos, consulte Mapeamento de tipos de dados JSON para tipos de dados SQL

É recomendável que você trate esses erros de modo programático no código SQL ou mantenha os dados do fluxo de erros em um destino externo. Isso requer que você adicione uma configuração de saída (consulte Configuração da saída do aplicativo) ao aplicativo. Para obter um exemplo de como funciona o fluxo de erros de aplicativo, consulte Exemplo: exploração do fluxo de erros no aplicativo.

> **Note**
>
> Seu aplicativo Kinesis Data Analytics não pode acessar nem modificar o fluxo de erro programaticamente porque o fluxo de erro é criado usando a conta do sistema. Você precisa usar a saída de erro para determinar quais erros seu aplicativo poderá encontrar. Em seguida, escreva o código SQL do seu aplicativo para lidar com condições de erro previstas.

Esquema do fluxo de erros

O fluxo de erros tem o seguinte esquema:

Campo	Tipo de dados	Observações
ERROR_TIME	TIMESTAMP	A hora em que ocorreu o erro
ERROR_LEVEL	VARCHAR (10)	
ERROR_NAME	VARCHAR (32)	
MESSAGE	VARCHAR (4096)	
DATA_ROWTIME	TIMESTAMP	O rowtime do registro de entrada
DATA_ROW	VARCHAR (49152)	Os dados codificados em hexadecimal na linha original.

		É possível usar bibliotecas padrão para decodificar esse valor em decimal ou usar recursos da web, como este conversor de hexadecimal em string.
PUMP_NAME	VARCHAR (128)	A bomba de origem, conforme definido com `CREATE PUMP`

Escalabilidade automática de aplicativos para aumentar a taxa de transferência

O Amazon Kinesis Data Analytics escala seu aplicativo de maneira elástica para acomodar o throughput do stream de origem e a complexidade de consultas da maioria dos cenários. O Kinesis Data Analytics fornece capacidade na forma de unidades de processamento do Kinesis (KPU). Uma única KPU fornece a memória (4 GB) e a computação e a rede correspondentes.

O limite padrão para KPUs para seu aplicativo é 64. Para obter instruções sobre como solicitar um aumento desse limite, consulte Para solicitar um aumento de limite em Limites de serviço da Amazon.

Uso de tags

Esta seção descreve como adicionar tags de metadados de chave-valor para aplicativos do Kinesis Data Analytics. Essas tags podem ser usadas para as seguintes finalidades:

- Determinar o faturamento de aplicativos individuais do Kinesis Data Analytics. Para obter mais informações, consulte Como usar tags de alocação de custos no AWSGuia de Gerenciamento de custos e faturamento.
- Controlar o acesso a recursos de aplicativos com base em tags. Para obter mais informações, consulte Controlar o acesso usando tags no Guia do usuário.
- Finalidades definidas pelo usuário. Você pode definir a funcionalidade do aplicativo com base na presença de tags do usuário.

Observe as seguintes informações sobre a marcação:

- O número máximo de tags do aplicativo inclui tags de sistema. O número máximo de tags do aplicativo definidas pelo usuário é de 50.
- Se uma ação inclui uma lista de tags que tem valores `Key` duplicados, o serviço lançará um `InvalidArgumentException`.

Este tópico contém as seguintes seções:

Adicionar tags quando um aplicativo é criado

Você pode adicionar tags ao criar um aplicativo usando o parâmetro `tags` da ação CreateApplication.

O exemplo de solicitação a seguir mostra o nó `Tags` para uma solicitação `CreateApplication`:

```
"Tags": [
    {
        "Key": "Key1",
        "Value": "Value1"
    },
    {
        "Key": "Key2",
        "Value": "Value2"
    }
]
```

Adicionar ou atualizar tags para um aplicativo existente

Você pode adicionar tags a um aplicativo usando a ação TagResource. Você não pode adicionar tags a um aplicativo usando a ação UpdateApplication.

Para atualizar uma tag existente, adicione uma tag com a mesma chave da tag existente.

O exemplo de solicitação a seguir para a ação `TagResource` adiciona novas tags ou atualiza tags existentes:

```
{
   "ResourceARN": "string",
   "Tags": [
      {
         "Key": "NewTagKey",
         "Value": "NewTagValue"
      },
      {
         "Key": "ExistingKeyOfTagToUpdate",
         "Value": "NewValueForExistingTag"
      }
   ]
}
```

Listar tags para um aplicativo

Para listar tags existentes, use a ação ListTagsForResource.

O exemplo de solicitação a seguir para a ação `ListTagsForResource` lista as tags de um aplicativo:

```
{
   "ResourceARN": "arn:aws:kinesisanalytics:us-west-2:012345678901:application/
MyApplication"
}
```

Remover tags de um aplicativo

Para remover tags de um aplicativo, use a ação UntagResource.

O exemplo de solicitação a seguir para a ação `UntagResource` remove tags de um aplicativo:

```
{
   "ResourceARN": "arn:aws:kinesisanalytics:us-west-2:012345678901:application/
MyApplication",
   "TagKeys": [ "KeyOfFirstTagToRemove", "KeyOfSecondTagToRemove" ]
}
```

Conceitos básicos do Amazon Kinesis Data Analytics para aplicativos SQL

A seguir, você pode encontrar tópicos para ajudar você a começar a usar o Amazon Kinesis Data Analytics para aplicativos SQL. Se você é novo no Kinesis Data Analytics para aplicativos SQL, é recomendável verificar os conceitos e a terminologia apresentados em Amazon Kinesis Data Analytics para aplicativos SQL: como funciona antes de executar as etapas da seção de Conceitos básicos.

Tópicos

Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga as instruções on-line.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica e digitar um código de verificação no teclado do telefone.

   Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWSé criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, é possível visualizar as atividades da conta atual e gerenciar sua conta acessando https://aws.amazon.com/ e selecionando Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu Usuário raiz da conta da AWS AWS IAM Identity Center, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login AWS Management Consolecomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, digite sua senha.

   Para obter ajuda ao fazer login usando o usuário-raiz, consulte Signing in as the root user (Fazer login como usuário-raiz) no Guia do usuário do Início de Sessão da AWS .
2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

   Para obter instruções, consulte Habilitar um dispositivo de MFA virtual para seu usuário Conta da AWS raiz (console) no Guia do usuário do IAM.

Criar um usuário com acesso administrativo

1. Habilitar o IAM Identity Center.

   Para obter instruções, consulte Habilitar AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center .
2. No Centro de Identidade do IAM, conceda o acesso administrativo para um usuário.

   Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM no Guia AWS IAM Identity Center do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com seu usuário do Centro de Identidade do IAM, use a URL de login que foi enviada ao seu endereço de e-mail quando você criou o usuário do Centro do Usuário do IAM.

Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte Como fazer login no portal de AWS acesso no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

   Para obter instruções, consulte Create a permission set no Guia do usuário do AWS IAM Identity Center .
2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

   Para obter instruções, consulte Add groups no Guia do usuário do AWS IAM Identity Center .

Etapa 1: Configurar uma conta da e criar um usuário administrador

Antes de usar o Amazon Kinesis Data Analytics pela primeira vez, realize as tarefas a seguir:

1. Inscreva-se para AWS
2. Criar um usuário do IAM

Inscreva-se para AWS

Quando você se inscreve no Amazon Web Services, você Conta da AWS se inscreve automaticamente em todos os serviços AWS, incluindo o Amazon Kinesis Data Analytics. Você será cobrado apenas pelos serviços que usar.

Com o Kinesis Data Analytics, você paga apenas pelos recursos que usa. Se você é um novo AWS cliente, pode começar a usar o Kinesis Data Analytics gratuitamente. Para mais informações, consulte Nível de uso gratuito da AWS.

Se você já tiver uma Conta da AWS, vá para a próxima tarefa. Se você não tiver um Conta da AWS, execute as etapas no procedimento a seguir para criar um.

Para criar um Conta da AWS

1. Abra https://portal.aws.amazon.com/billing/signup.

2. Siga as instruções on-line.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica e digitar um código de verificação no teclado do telefone.

   Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWSé criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

Anote seu Conta da AWS ID porque você precisará dele para a próxima tarefa.

Criar um usuário do IAM

Serviços em AWS, como o Amazon Kinesis Data Analytics, exigem que você forneça credenciais ao acessá-los para que o serviço possa determinar se você tem permissões para acessar os recursos de propriedade desse serviço. O console requer sua senha. Você pode criar chaves de acesso Conta da AWS para acessar a API AWS CLI ou. No entanto, não recomendamos que você acesse AWS usando as credenciais do seu Conta da AWS. Em vez disso, recomendamos usar o AWS Identity and Access Management (IAM) Crie um usuário do IAM, adicione o usuário a um grupo do IAM com permissões administrativas e, em seguida, conceda permissões administrativas ao usuário do IAM criado. Em seguida, você pode acessar AWS usando uma URL especial e as credenciais desse usuário do IAM.

Se você se inscreveu AWS, mas não criou um usuário do IAM para si mesmo, você pode criar um usando o console do IAM.

Os exercícios de conceitos básicos deste guia pressupõem que você tenha um usuário (`adminuser`) com privilégios de administrador. Siga o procedimento para criar `adminuser` na conta.

Para criar um usuário administrador e fazer login no console

1. Crie um usuário administrador chamado `adminuser`, em sua Conta da AWS. Para obter instruções, consulte Criar seu primeiro grupo de administradores e usuário do IAM no Guia do usuário do IAM.
2. Um usuário pode fazer login no AWS Management Console usando um URL especial. Para obter mais informações, consulte Como os usuários fazem login em sua conta no Guia do usuário do IAM.

Para mais informações sobre IAM, consulte o seguinte:

- AWS Identity and Access Management (IAM)
- Conceitos básicos
- Guia do usuário do IAM

Próxima etapa

Etapa 2: configurar o AWS Command Line Interface (AWS CLI)

Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga as instruções on-line.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica e digitar um código de verificação no teclado do telefone.

   Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWSé criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, é possível visualizar as atividades da conta atual e gerenciar sua conta acessando https://aws.amazon.com/ e selecionando Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu Usuário raiz da conta da AWS AWS IAM Identity Center, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login AWS Management Consolecomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, digite sua senha.

   Para obter ajuda ao fazer login usando o usuário-raiz, consulte Signing in as the root user (Fazer login como usuário-raiz) no Guia do usuário do Início de Sessão da AWS .
2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

   Para obter instruções, consulte Habilitar um dispositivo de MFA virtual para seu usuário Conta da AWS raiz (console) no Guia do usuário do IAM.

Criar um usuário com acesso administrativo

1. Habilitar o IAM Identity Center.

   Para obter instruções, consulte Habilitar AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center .
2. No Centro de Identidade do IAM, conceda o acesso administrativo para um usuário.

   Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM no Guia AWS IAM Identity Center do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com seu usuário do Centro de Identidade do IAM, use a URL de login que foi enviada ao seu endereço de e-mail quando você criou o usuário do Centro do Usuário do IAM.

  Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte Como fazer login no portal de AWS acesso no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

Para obter instruções, consulte Create a permission set no Guia do usuário do AWS IAM Identity Center .

2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

   Para obter instruções, consulte Add groups no Guia do usuário do AWS IAM Identity Center .

Etapa 2: configurar o AWS Command Line Interface (AWS CLI)

Siga as etapas para baixar e configurar o AWS Command Line Interface (AWS CLI).

> **Important**
>
> Você não precisa AWS CLI executar as etapas do exercício de introdução. No entanto, alguns dos exercícios neste guia usam a AWS CLI. Você pode pular essa etapa Etapa 3: Criar o aplicativo de análise inicial Amazon Kinesis Data Analytics, acessar e configurar a AWS CLI mais tarde quando precisar.

Para configurar o AWS CLI

1. Faça download e configure a AWS CLI. Para obter instruções, consulte os seguintes tópicos no Manual do usuário do AWS Command Line Interface :

   - Configurando o AWS Command Line Interface
   - Configurando o AWS Command Line Interface

2. Adicione um perfil nomeado para o usuário administrador no arquivo de AWS CLI configuração. Você usa esse perfil ao executar os AWS CLI comandos. Para obter mais informações sobre perfis nomeados, consulte Perfis nomeados no Guia do usuário da AWS Command Line Interface .

   ```
   [profile adminuser]
   aws_access_key_id = adminuser access key ID
   aws_secret_access_key = adminuser secret access key
   region = aws-region
   ```

   Para obter uma lista dos disponíveis Regiões da AWS, consulte Regiões e endpoints no Referência geral da Amazon Web Services.

3. Verifique a configuração digitando o seguinte comando no prompt de comando:

```
aws help
```

Próxima etapa

Etapa 3: Criar o aplicativo de análise inicial Amazon Kinesis Data Analytics

Etapa 3: Criar o aplicativo de análise inicial Amazon Kinesis Data Analytics

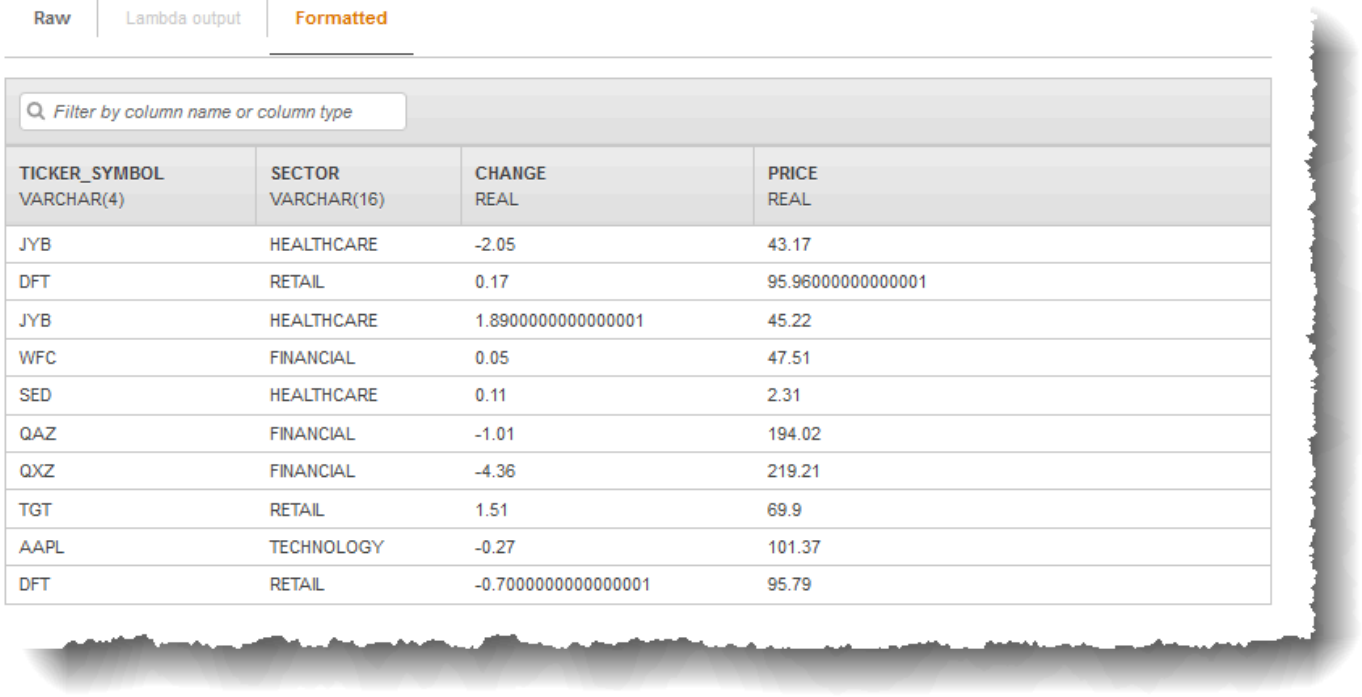
Seguindo as etapas nesta seção, você poderá criar seu primeiro aplicativo de análise de dados do Kinesis Data Analytics usando o console.

> **Note**
>
> Sugerimos que você examine Amazon Kinesis Data Analytics para aplicativos SQL: como funciona antes de tentar fazer o exercício de conceitos básicos.

Para este exercício de conceitos básicos, é possível usar o console para trabalhar com o fluxo de demonstração ou modelos com código de aplicativo.

- Se você optar por usar o fluxo de demonstração, o console criará um fluxo de dados do Kinesis em sua conta que é chamado `kinesis-analytics-demo-stream`.

  Um aplicativo Kinesis Data Analytics requer uma origem de streaming. Para essa origem, vários exemplos de SQL neste guia usam o fluxo de demonstração `kinesis-analytics-demo-stream`. O console também executa um script que adiciona continuamente dados de exemplo (registros simulados de negociações de ações) a esse fluxo, conforme mostrado a seguir.



Você pode usar `kinesis-analytics-demo-stream` como a origem de streaming para seu aplicativo neste exercício.

> **Note**
>
> O fluxo de demonstração permanece na conta. Você pode usá-lo para testar outros exemplos neste guia. No entanto, quando você sai do console, o script usado pelo console para de preencher os dados. Quando necessário, o console oferece a opção de começar a preencher o fluxo novamente.

- Se optar por usar os modelos com código de aplicativo de exemplo, você usará o código do modelo fornecido pelo console para executar análises simples no fluxo de demonstração.

Use esses recursos para configurar rapidamente o primeiro aplicativo da seguinte forma:

1. Criar um aplicativo: você só precisa fornecer um nome. O console cria o aplicativo e o serviço define o estado do aplicativo para `READY`.

2. Configurar a entrada: primeiro você adiciona uma origem de streaming, o fluxo de demonstração. Você deve criar um fluxo de demonstração no console para que possa usá-lo. Em seguida, o console usa uma amostra aleatória dos registros no fluxo de demonstração e infere um esquema para o fluxo de entrada de aplicativo criado. O console nomeia o stream no aplicativo `SOURCE_SQL_STREAM_001`.

   O console usa a API de descoberta para inferir o esquema. Se necessário, você pode editar o esquema inferido. Para ter mais informações, consulte DiscoverInputSchema. O Kinesis Data Analytics usa esse esquema para criar um fluxo de aplicativo.

   Quando você inicia o aplicativo, o Kinesis Data Analytics lê o fluxo de demonstração continuamente em seu nome e insere linhas no fluxo de entrada de aplicativo `SOURCE_SQL_STREAM_001`.

3. Especificar o código do aplicativo: você usa um modelo (chamado Continuous filter) que fornece o código a seguir:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM"
  (symbol VARCHAR(4), sector VARCHAR(12), CHANGE DOUBLE, price DOUBLE);

-- Create pump to insert into output.
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM ticker_symbol, sector, CHANGE, price
      FROM "SOURCE_SQL_STREAM_001"
      WHERE sector SIMILAR TO '%TECH%';
```

   O código de aplicativo consulta o stream no aplicativo `SOURCE_SQL_STREAM_001`. Em seguida, o código insere as linhas resultantes em outro stream no aplicativo `DESTINATION_SQL_STREAM`, usando bombas. Para obter mais informações sobre esse padrão de codificação, consulte Código do aplicativo.

   Para obter mais informações sobre os elementos de idioma SQL compatíveis com o Kinesis Data Analytics, consulte Amazon Kinesis Data Analytics SQL Reference.

4. Configuração de saída: neste exercício, você não configura nenhuma saída. Ou seja, você não persiste dados no stream no aplicativo criado pelo aplicativo em nenhum destino externo. Em vez disso, você verifica os resultados da consulta no console. Exemplos adicionais neste guia mostram como configurar a saída. Para obter um exemplo, consulte Exemplo: criar alertas simples.

> **Important**
>
> O exercício usa a região Leste dos EUA (N. da Virgínia) (us-east-1) para configurar o aplicativo. Você pode usar qualquer um dos compatíveis Regiões da AWS.

Próxima etapa

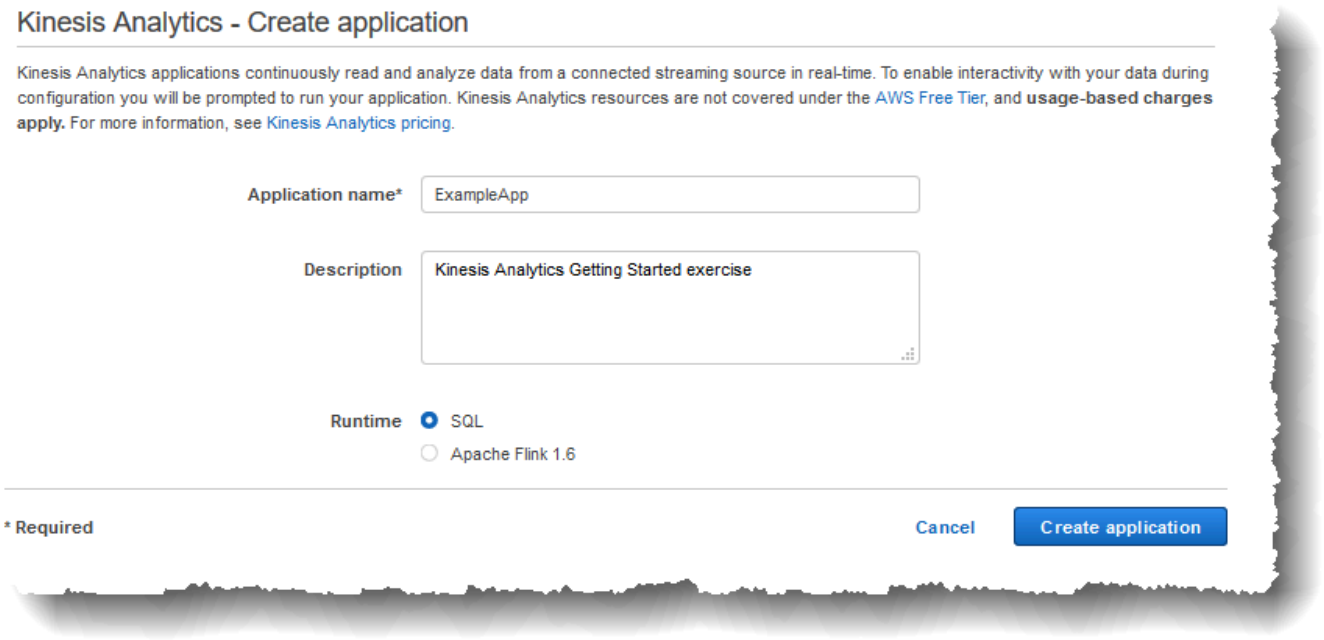
Etapa 3.1: criar um aplicativo

Etapa 3.1: criar um aplicativo

Nesta seção, você cria um aplicativo Amazon Kinesis Data Analytics. Configure a entrada do aplicativo na próxima etapa.

Para criar um aplicativo de análise de dados

1. Faça login AWS Management Console e abra o console do Managed Service for Apache Flink em https://console.aws.amazon.com/kinesisanalytics.
2. Escolha Criar aplicativo.
3. Na página Create application (Criar aplicativo), digite o nome de um aplicativo, digite uma descrição, selecione SQL para a configuração de Runtime (Tempo de execução) do aplicativo e selecione Create application (Criar aplicativo).



Ao fazer isso, você cria um aplicativo Kinesis Data Analytics com o status READY. O console mostra o hub de aplicativo em que você poderá configurar a entrada e a saída.

> **Note**
>
> Para criar um aplicativo, a operação CreateApplication requer apenas o nome do aplicativo. Você pode adicionar a configuração de entrada e saída após a criação de um aplicativo no console.

Na próxima etapa, configure a entrada do aplicativo. Na configuração de entrada, adicione uma fonte de dados de streaming ao aplicativo e descubra um esquema para um fluxo de entrada de aplicativo fazendo a amostragem de dados na origem de streaming.

Próxima etapa

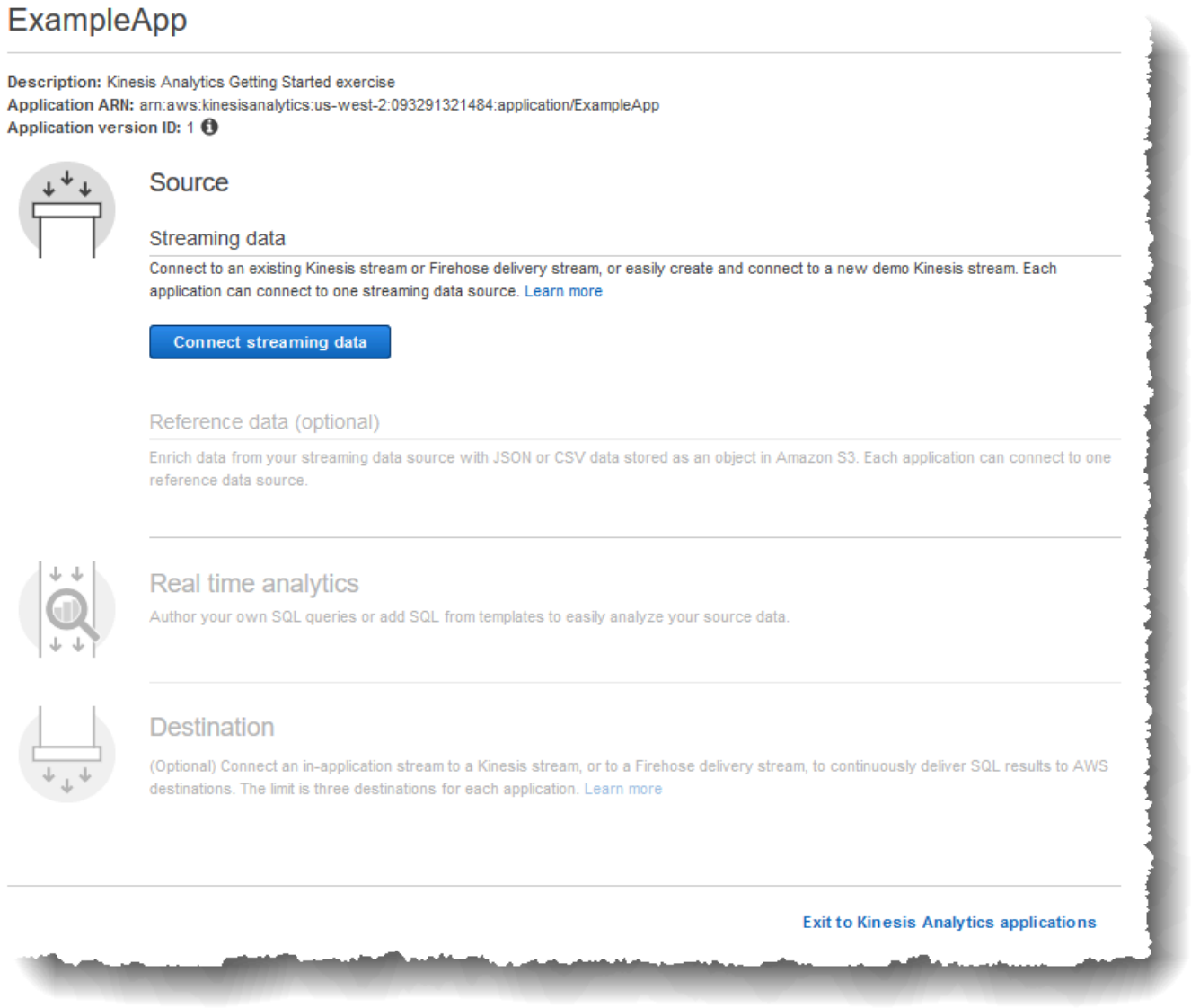
Etapa 3.2: Configurar a entrada

Etapa 3.2: Configurar a entrada

O aplicativo precisa de uma origem de streaming. Para ajudá-lo a começar, o console pode criar um fluxo de demonstração (chamado `kinesis-analytics-demo-stream`). O console também executa um script que preenche registros no fluxo.

Para adicionar uma origem de streaming ao seu aplicativo

1. Na página de hub do aplicativo no console, selecione Connect streaming data (Conectar dados de streaming).



2. Na página exibida, analise o seguinte:

- A seção Source, na qual você especifica uma origem de streaming para o aplicativo. É possível selecionar uma origem de streaming existente ou criar uma nova. Neste exercício, você cria um novo fluxo, o fluxo de demonstração.

  Por padrão, o console nomeia o fluxo de entrada de aplicativo criado como `INPUT_SQL_STREAM_001`. Neste exercício, mantenha esse nome como ele aparece.

- Nome de referência do stream: esta opção mostra o nome do stream de entrada no aplicativo criado, `SOURCE_SQL_STREAM_001`. Você pode alterar o nome, mas, neste exercício, mantenha esse nome.

  Na configuração de entrada, mapeie o fluxo de demonstração para um fluxo de entrada de aplicativo criado. Quando você inicia o aplicativo, o Amazon Kinesis Data Analytics lê o fluxo de demonstração continuamente e insere linhas no fluxo de entrada de aplicativo . Consulte esse fluxo de entrada de aplicativo no código do aplicativo.

- Pré-processamento de registros com AWS Lambda: Essa opção é onde você especifica uma AWS Lambda expressão que modifica os registros no fluxo de entrada antes da execução do código do aplicativo. Neste exercício, deixe a opção Disabled selecionada. Para obter mais informações sobre o pré-processamento do Lambda, consulte Pré-processar dados usando uma função do Lambda.

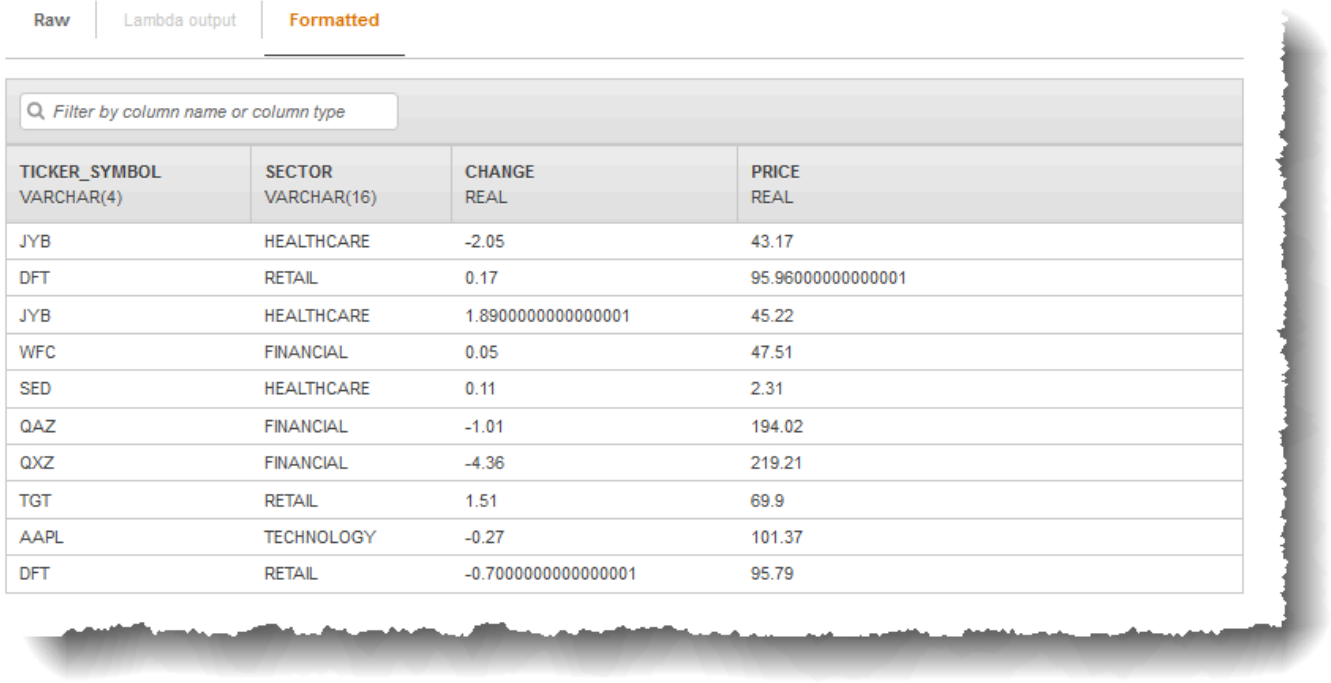
Após fornecer todas as informações desta página, o console enviará uma solicitação de atualização (consulte UpdateApplication) para adicionar a configuração de entrada ao aplicativo.

3. Na página Source, escolha Configure a new stream.
4. Escolha Create demo stream. O console configura a entrada do aplicativo fazendo o seguinte:

   - O console cria um fluxo de dados do Kinesis chamado `kinesis-analytics-demo-stream`.
   - O console preenche o fluxo com dados de exemplo de códigos das ações.

- Usando a ação de entrada DiscoverInputSchema, o console infere um esquema lendo os registros de exemplo no fluxo. O esquema inferido é o esquema para o stream de entrada no aplicativo criado. Para ter mais informações, consulte Configuração de entrada do aplicativo.
- O console mostra o esquema inferido e os dados de exemplo que ele leu na origem de streaming para inferir o esquema.

O console exibe os registros de exemplo na origem de streaming.



TICKER_SYMBOL VARCHAR(4)	SECTOR VARCHAR(16)	CHANGE REAL	PRICE REAL
JYB	HEALTHCARE	-2.05	43.17
DFT	RETAIL	0.17	95.96000000000001
JYB	HEALTHCARE	1.8900000000000001	45.22
WFC	FINANCIAL	0.05	47.51
SED	HEALTHCARE	0.11	2.31
QAZ	FINANCIAL	-1.01	194.02
QXZ	FINANCIAL	-4.36	219.21
TGT	RETAIL	1.51	69.9
AAPL	TECHNOLOGY	-0.27	101.37
DFT	RETAIL	-0.7000000000000001	95.79

O seguinte é exibido na página do console Stream sample:

- A guia Raw stream sample exibe uma amostra dos registros de fluxo brutos feita pela ação da API DiscoverInputSchema para inferir o esquema.
- A guia Formatted stream sample mostra a versão tabular dos dados na guia Raw stream sample.
- Se selecionar Edit schema, você poderá editar o esquema inferido. Neste exercício, não altere o esquema inferido. Para obter mais informações sobre a edição de um esquema, consulte Trabalho com o editor de esquema.

  Se selecionar Rediscover schema, você poderá solicitar que o console execute DiscoverInputSchema novamente e infira o esquema.

5. Escolha Save and continue.

   Agora, você tem um aplicativo com a configuração de entrada adicionada a ele. Na próxima etapa, você adiciona o código SQL para executar algumas análises no fluxo de dados entrada de aplicativo.

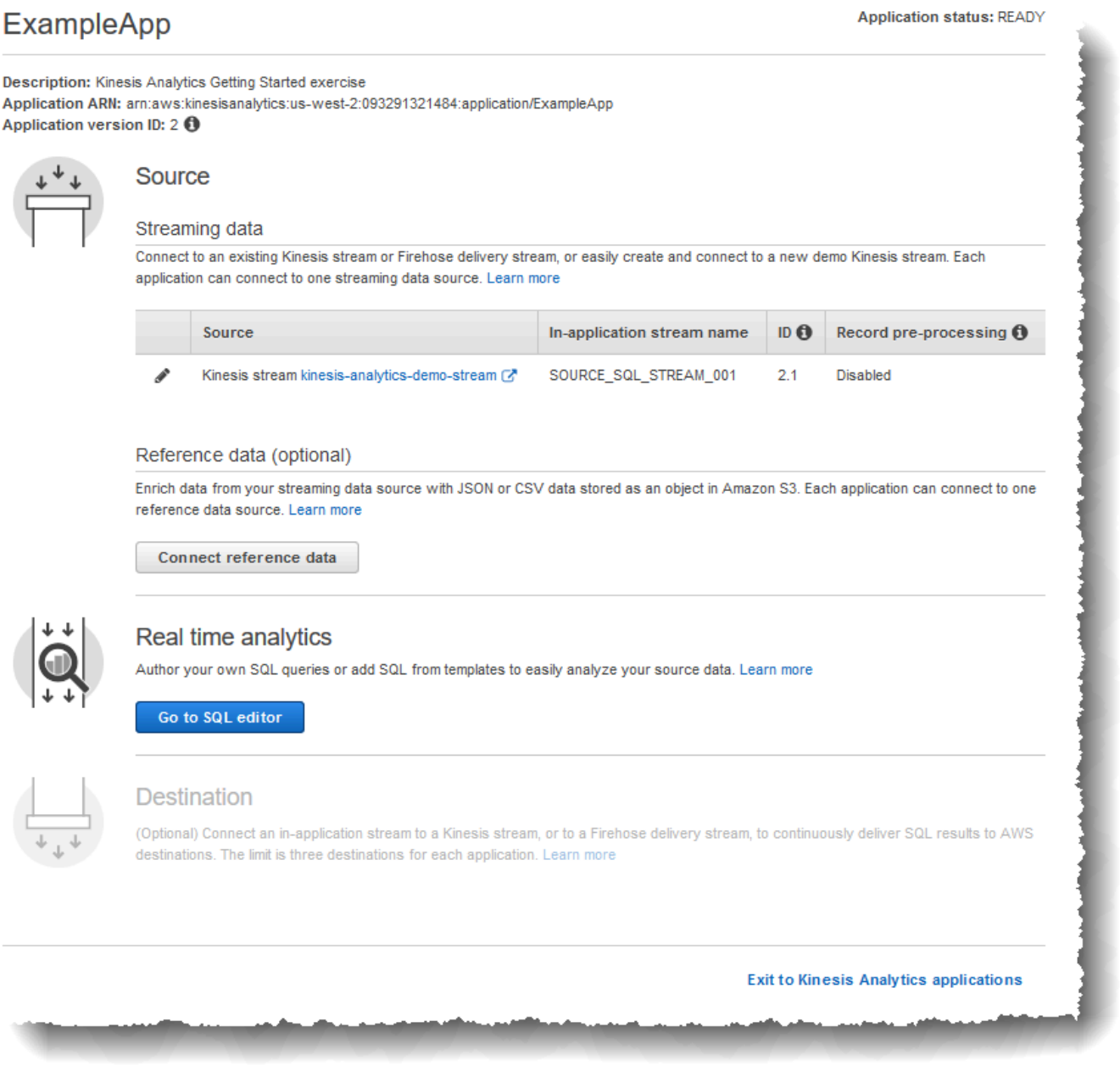
Próxima etapa

Etapa 3.3: Adicionar análise em tempo real (adicionar código de aplicativo)

Etapa 3.3: Adicionar análise em tempo real (adicionar código de aplicativo)

Você pode criar suas próprias consultas SQL em relação ao stream no aplicativo, mas, para a etapa a seguir, você usará um dos modelos que fornece código de exemplo.

1. Na página do hub de aplicativo, escolha Go to SQL editor.



2. Na seção Você gostaria de começar a executar "ExampleApp“? caixa de diálogo, escolha Sim, iniciar aplicativo.

   O console envia uma solicitação de inicialização do aplicativo (consulte StartApplication) e, em seguida, a página do editor SQL é exibida.

3. O console abre a página do editor SQL. Analise a página, incluindo os botões (Add SQL from templates, Save and run SQL) e várias guias.
4. No editor SQL, escolha Add SQL from templates.

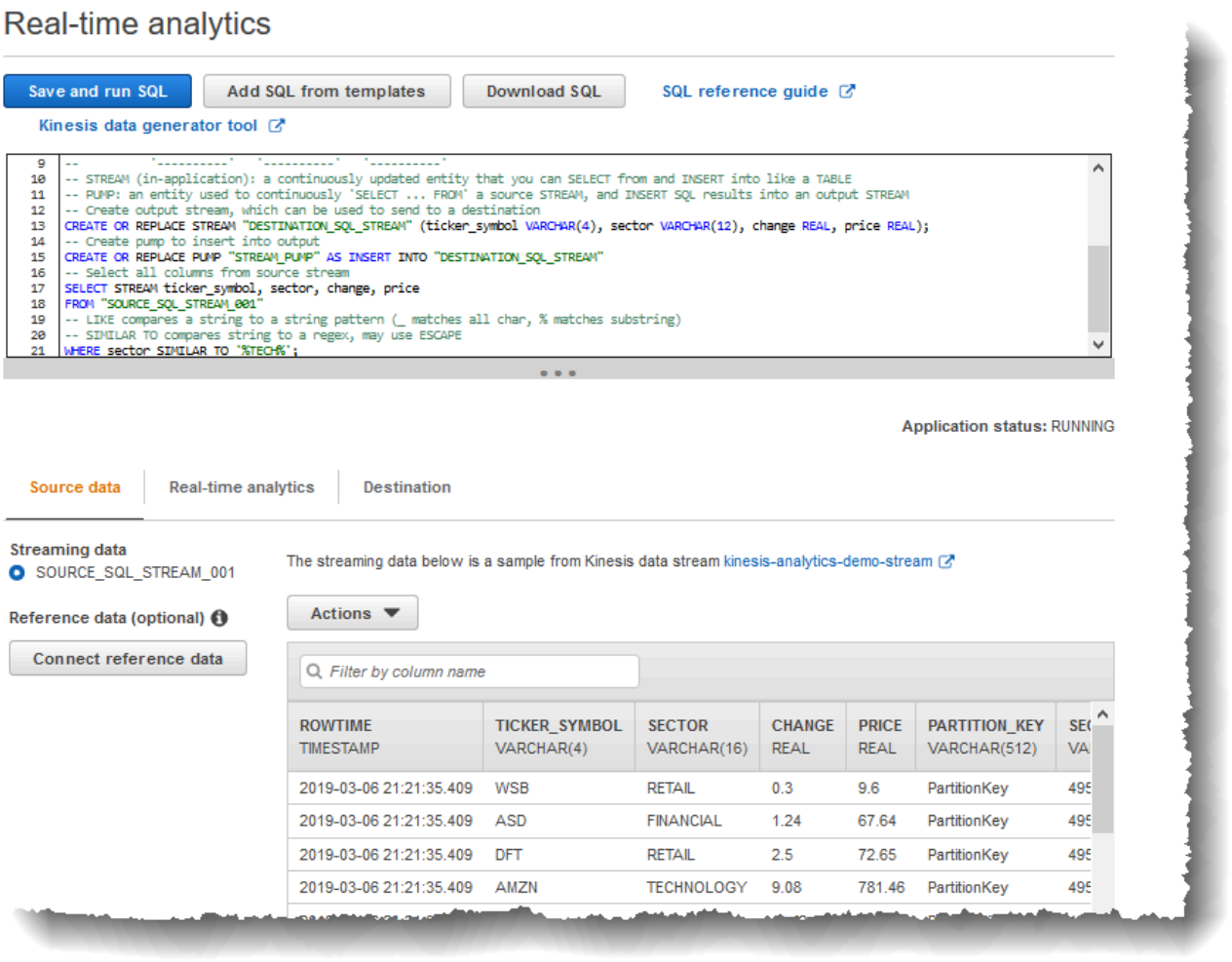
5. Na lista de modelos disponíveis, escolha Continuous filter. O código de exemplo lê dados de um stream no aplicativo (a cláusula `WHERE` filtra as linhas) e os insere em outro stream no aplicativo, da seguinte forma:

   - Ele cria o stream no aplicativo `DESTINATION_SQL_STREAM`.
   - Ele cria uma bomba `STREAM_PUMP` e a usa para selecionar linhas do `SOURCE_SQL_STREAM_001` e inseri-las no `DESTINATION_SQL_STREAM`.

6. Escolha Add this SQL to editor.
7. Teste o código do aplicativo da seguinte forma:

   Lembre-se, você já iniciou o aplicativo (o status é RUNNING). Portanto, o Amazon Kinesis Data Analytics já está lendo continuamente a origem de streaming e adicionando linhas ao stream no aplicativo `SOURCE_SQL_STREAM_001`.

   a. No Editor SQL, selecione Save and run SQL. Primeiro, o console envia a solicitação de atualização para salvar o código do aplicativo. Em seguida, o código é executado continuamente.
   b. Você pode ver os resultados na guia Real-time analytics.



O Editor SQL tem as seguintes guias:

- A guia Source data mostra um fluxo de entrada de aplicativo que é mapeado para a origem de streaming. Escolha o fluxo de aplicativo e você poderá ver a entrada dos dados em ação. Observe as colunas adicionais do stream de entrada no aplicativo que não foram especificadas na configuração de entrada. Elas incluem as seguintes colunas de timestamp:

  - ROWTIME: cada linha de um stream no aplicativo tem uma coluna especial chamada `ROWTIME`. Essa coluna é o timestamp quando o Amazon Kinesis Data Analytics inseriu a linha no primeiro stream no aplicativo (o stream de entrada no aplicativo mapeado para a origem de streaming).

- Approximate_Arrival_Time: cada registro do Kinesis Data Analytics inclui um valor chamado `Approximate_Arrival_Time`. Esse valor é o timestamp de chegada aproximado definido quando a origem de streaming recebe e armazena o registro com êxito. Quando o Kinesis Data Analytics lê registros em uma origem de streaming, ele obtém essa coluna no fluxo de entrada de aplicativo.

Esses valores de timestamp são úteis nas consultas em janelas baseadas em tempo. Para ter mais informações, consulte Consultas em janelas.

- A guia Real-time analytics mostra todos os outros fluxos de aplicativo criados pelo código de aplicativo. Ela também inclui o stream de erros. O Kinesis Data Analytics envia quaisquer linhas que não consegue processar no fluxo de erros. Para ter mais informações, consulte Como tratar erros.

  Selecione `DESTINATION_SQL_STREAM` para visualizar as linhas inseridas pelo código do aplicativo. Observe as colunas adicionais que o código do aplicativo não criou. Essas colunas incluem a coluna de timestamp `ROWTIME`. O Kinesis Data Analytics simplesmente copia esses valores da origem (`SOURCE_SQL_STREAM_001`).

- A guia Destination mostra o destino externo em que o Kinesis Data Analytics grava os resultados da consulta. Você ainda não configurou um destino externo para a saída do aplicativo.

Próxima etapa

Etapa 3.4: (Opcional) Atualizar o código do aplicativo

Etapa 3.4: (Opcional) Atualizar o código do aplicativo

Nesta etapa, você explorará como atualizar o código do aplicativo.

Para atualizar o código do aplicativo

1. Crie outro stream no aplicativo, da seguinte forma:

- Crie outro stream no aplicativo chamado `DESTINATION_SQL_STREAM_2`.
- Crie uma bomba e, em seguida, a utilize para inserir linhas no fluxo recém-criado, selecionando linhas no `DESTINATION_SQL_STREAM`.

No Editor SQL, anexe o seguinte código ao código de aplicativo existente:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM_2"
           (ticker_symbol VARCHAR(4),
            change        DOUBLE,
            price         DOUBLE);

CREATE OR REPLACE PUMP "STREAM_PUMP_2" AS
   INSERT INTO "DESTINATION_SQL_STREAM_2"
      SELECT STREAM ticker_symbol, change, price
      FROM   "DESTINATION_SQL_STREAM";
```

Salve e execute o código. Os fluxos de aplicativo adicionais aparecem na guia Real-time analytics.

2. Crie dois fluxos de aplicativo. Filtre as linhas no `SOURCE_SQL_STREAM_001` com base no índice de ações e, em seguida, insira as linhas nesses fluxos separados.

   Anexe as seguintes instruções SQL ao código do aplicativo:

```
CREATE OR REPLACE STREAM "AMZN_STREAM"
           (ticker_symbol VARCHAR(4),
            change        DOUBLE,
            price         DOUBLE);

CREATE OR REPLACE PUMP "AMZN_PUMP" AS
   INSERT INTO "AMZN_STREAM"
      SELECT STREAM ticker_symbol, change, price
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  ticker_symbol SIMILAR TO '%AMZN%';

CREATE OR REPLACE STREAM "TGT_STREAM"
           (ticker_symbol VARCHAR(4),
            change        DOUBLE,
            price         DOUBLE);

CREATE OR REPLACE PUMP "TGT_PUMP" AS
```

```
INSERT INTO "TGT_STREAM"
   SELECT STREAM ticker_symbol, change, price
   FROM   "SOURCE_SQL_STREAM_001"
   WHERE  ticker_symbol SIMILAR TO '%TGT%';
```

Salve e execute o código. Observe os fluxos de aplicativo adicionais na guia Real-time analytics.

Agora você tem seu primeiro aplicativo Amazon Kinesis Data Analytics em funcionamento. Neste exercício, você fez o seguinte:

- Criou a seu primeiro aplicativo Kinesis Data Analytics.

- Configurou a entrada do aplicativo que identificou o fluxo de demonstração como origem de streaming e a mapeou para um stream no aplicativo (`SOURCE_SQL_STREAM_001`) criado. O Kinesis Data Analytics lê continuamente o fluxo de demonstração e insere registros no fluxo de aplicativo.

- O código de aplicativo consultou o `SOURCE_SQL_STREAM_001` e gravou a saída em outro stream no aplicativo chamado `DESTINATION_SQL_STREAM`.

Agora, se desejar, você pode configurar a saída do aplicativo para gravá-la em um destino externo. Ou seja, você pode configurar a saída do aplicativo para gravar registros no `DESTINATION_SQL_STREAM` para um destino externo. Neste exercício, essa etapa é opcional. Para saber como configurar o destino, vá para a próxima etapa.

Próxima etapa

Etapa 4 (opcional) Editar o esquema e código SQL usando o console.

Etapa 4 (opcional) Editar o esquema e código SQL usando o console

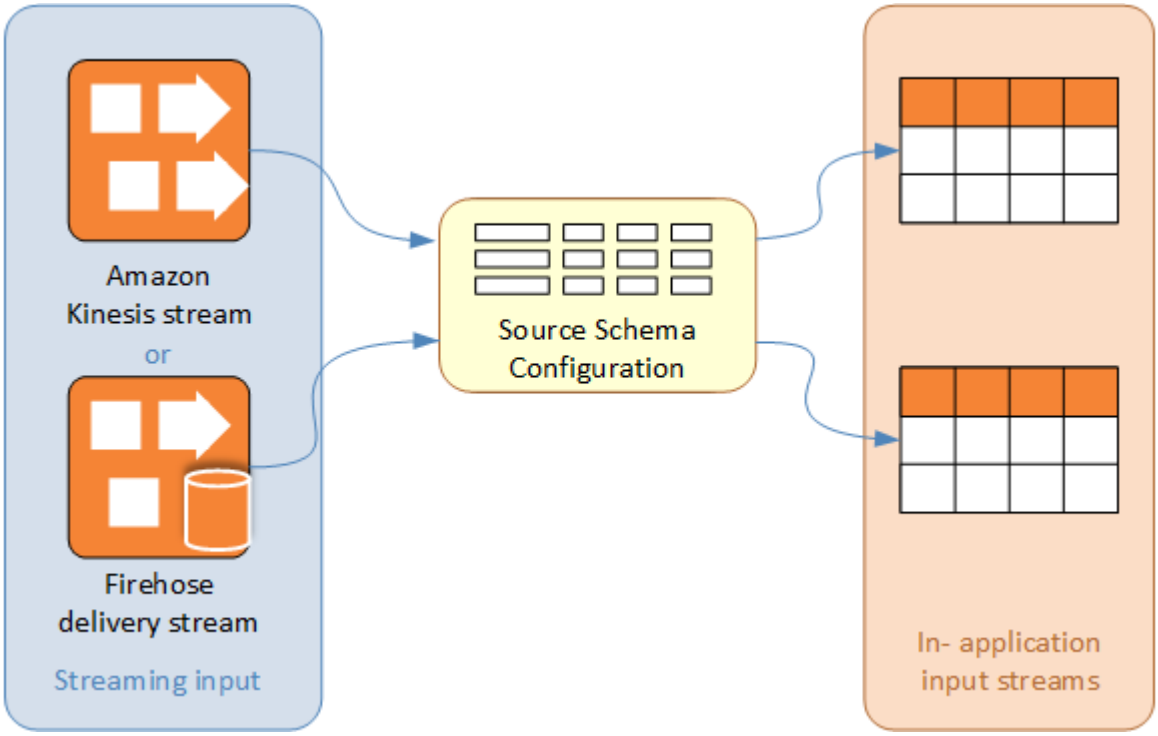
A seguir, você pode encontrar informações sobre como editar um esquema inferido e como editar código SQL para o Amazon Kinesis Data Analytics. Isso é feito trabalhando com o editor de esquema e o editor SQL que fazem parte do console do Kinesis Data Analytics.

> **Note**
>
> Para acessar ou criar amostra de dados no console, a função do usuário de login deve ter a permissão `kinesisanalytics:GetApplicationState`. Para obter mais informações sobre as permissões de aplicativo do Kinesis Data Analytics, consulte Visão geral do gerenciamento de acesso.

Tópicos

Trabalho com o editor de esquema

O esquema do fluxo de entrada de um aplicativo Amazon Kinesis Data Analytics define a forma como os dados do fluxo são disponibilizados nas consultas SQL no aplicativo.



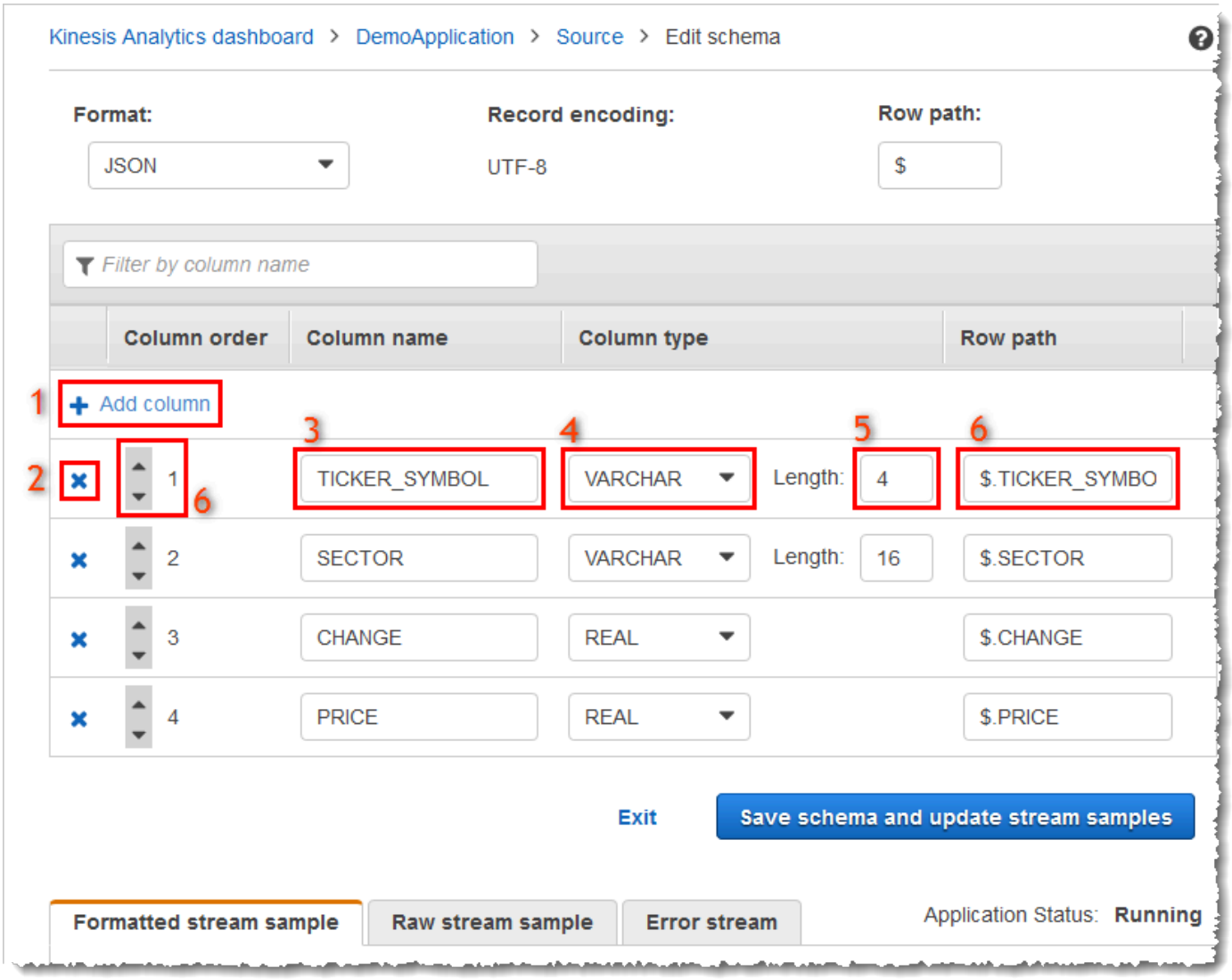
O esquema contém critérios de seleção para determinar qual parte da entrada de streaming será transformada em uma coluna de dados no stream de entrada no aplicativo. Essa entrada pode ser um dos seguintes itens:

- Uma expressão JSONPath para fluxos de entrada JSON. JSONPath é uma ferramenta para consultar dados JSON.
- Um número de coluna para fluxos de entrada em formato CSV (valores separados por vírgulas).
- Um nome de coluna e um tipo de dados SQL para apresentar os dados no fluxo de dados do aplicativo. O tipo de dados também contém um tamanho de caractere ou dados binários.

O console tenta gerar o esquema usando DiscoverInputSchema. Se a descoberta do esquema apresentar falha ou retornar um esquema incorreto ou incompleto, você deverá editá-lo manualmente usando o editor de esquema.

Tela principal do editor de esquema

A imagem a seguir mostra a tela principal do editor de esquema.



Você pode aplicar as seguintes edições ao esquema:

- Adicionar uma coluna (1): talvez seja necessário adicionar uma coluna de dados se um item de dados não for detectado automaticamente.
- Excluir uma coluna (2): você pode excluir dados do fluxo de origem se eles não forem necessários ao aplicativo. Essa exclusão não afeta os dados no fluxo de origem. Se os dados forem excluídos, eles simplesmente não serão disponibilizados no aplicativo.
- Renomear uma coluna (3). Um nome de coluna não pode ficar em branco, deve ter mais de um único caractere e não deve conter palavras-chave SQL reservadas. O nome também deve atender aos critérios de nomenclatura dos identificadores SQL comuns: o nome deve começar com uma letra e conter apenas letras, sublinhados e dígitos.

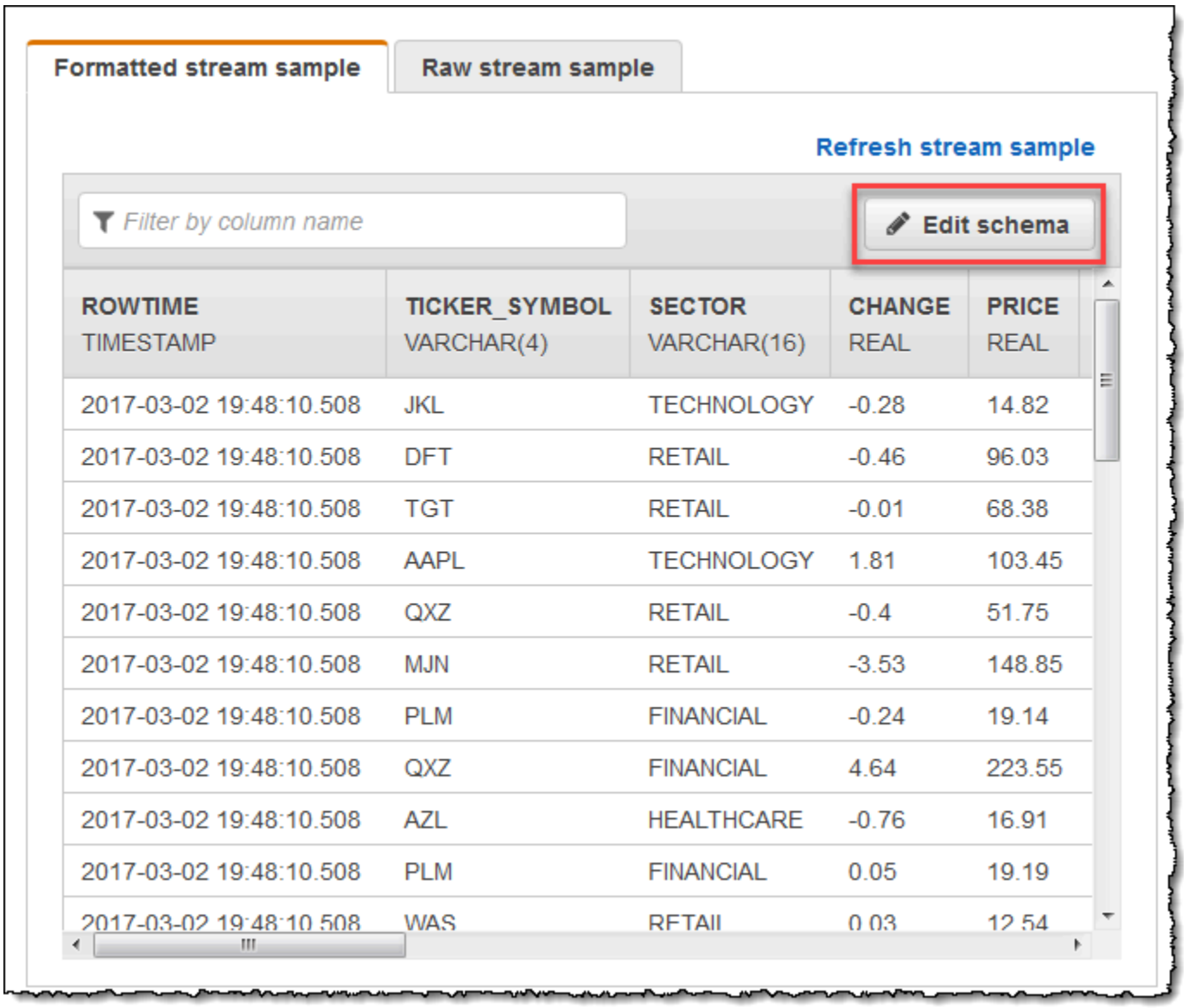
- Alterar o tipo de dados (4) ou o tamanho (5) de uma coluna: você pode especificar um tipo de dados compatível para uma coluna. Se você especificar um tipo de dados incompatível, a coluna será preenchida com NULL ou o fluxo do aplicativo não será preenchido. Nesse último caso, os erros serão gravados no fluxo de erros. Se você especificar um tamanho muito pequeno para uma coluna, os dados de entrada serão truncados.
- Alterar os critérios de seleção de uma coluna (6): você pode editar a expressão JSONPath ou a ordem da coluna CSV usada para determinar a origem dos dados em uma coluna. Para alterar os critérios de seleção de um schema JSON, insira um novo valor para a expressão de caminho de linha. Um esquema CSV usa a ordem das colunas como critérios de seleção. Para alterar os critérios de seleção de um esquema CSV, altere a ordem das colunas.

Edição de um esquema para uma origem de streaming

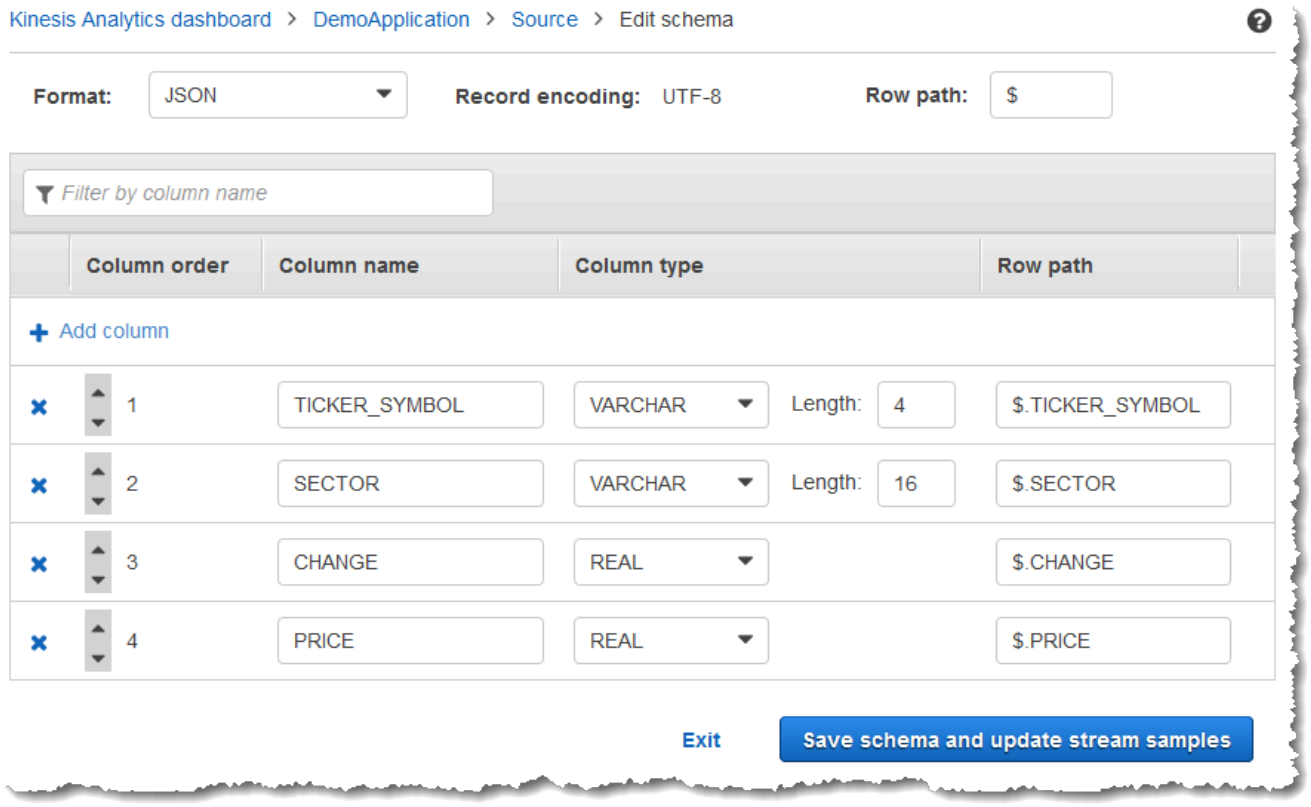
Se você precisar editar um esquema para uma origem de streaming, siga estas etapas.

Para editar o esquema de uma origem de streaming

1. Na página Source, escolha Edit schema.



2. Na página Edit schema, edite o esquema de origem.



3. Em Format, escolha JSON ou CSV. Para o formato JSON ou CSV, a codificação compatível é ISO 8859-1.

Para obter mais informações sobre como editar o esquema para o formato JSON ou CSV, consulte os procedimentos nas próximas seções.

Edição de um esquema JSON

Você pode editar um esquema JSON usando as etapas a seguir.

Para editar um esquema JSON

1. No editor de esquema, escolha Add column para adicionar uma coluna.

   Uma nova coluna aparece na primeira posição de coluna. Para alterar a ordem das colunas, selecione as setas para cima e para baixo ao lado do nome da coluna.

   No caso de uma nova coluna, forneça as seguintes informações:

   - Em Column name, digite um nome.

Um nome de coluna não pode ficar em branco, deve ter mais de um único caractere e não deve conter palavras-chave SQL reservadas. Ele também deve atender aos critérios dos identificadores SQL comuns: deve começar com uma letra e conter apenas letras, sublinhados e dígitos.

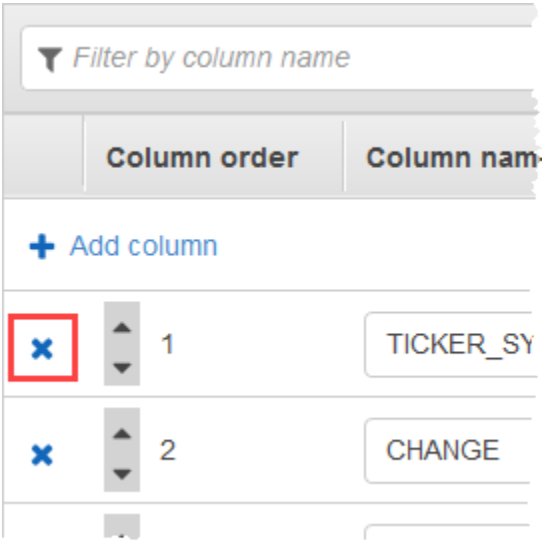
- Em Column type, digite um tipo de dados SQL.

  Um tipo de coluna pode ser qualquer tipo de dados SQL compatível. Se o novo tipo de dados for CHAR, VARBINARY ou VARCHAR, especifique um tamanho de dados em Length. Para obter mais informações, consulte Tipos de dados.

- Em Linha de linha, forneça um caminho de linha. Um caminho de linha é uma expressão JSONPath válida que é mapeada para um elemento JSON.

  > **Note**
  >
  > O valor base Row path é o caminho para o pai de nível superior que contém os dados a serem importados. Por padrão, esse valor é $. Para obter mais informações, consulte `RecordRowPath` em `JSONMappingParameters`.

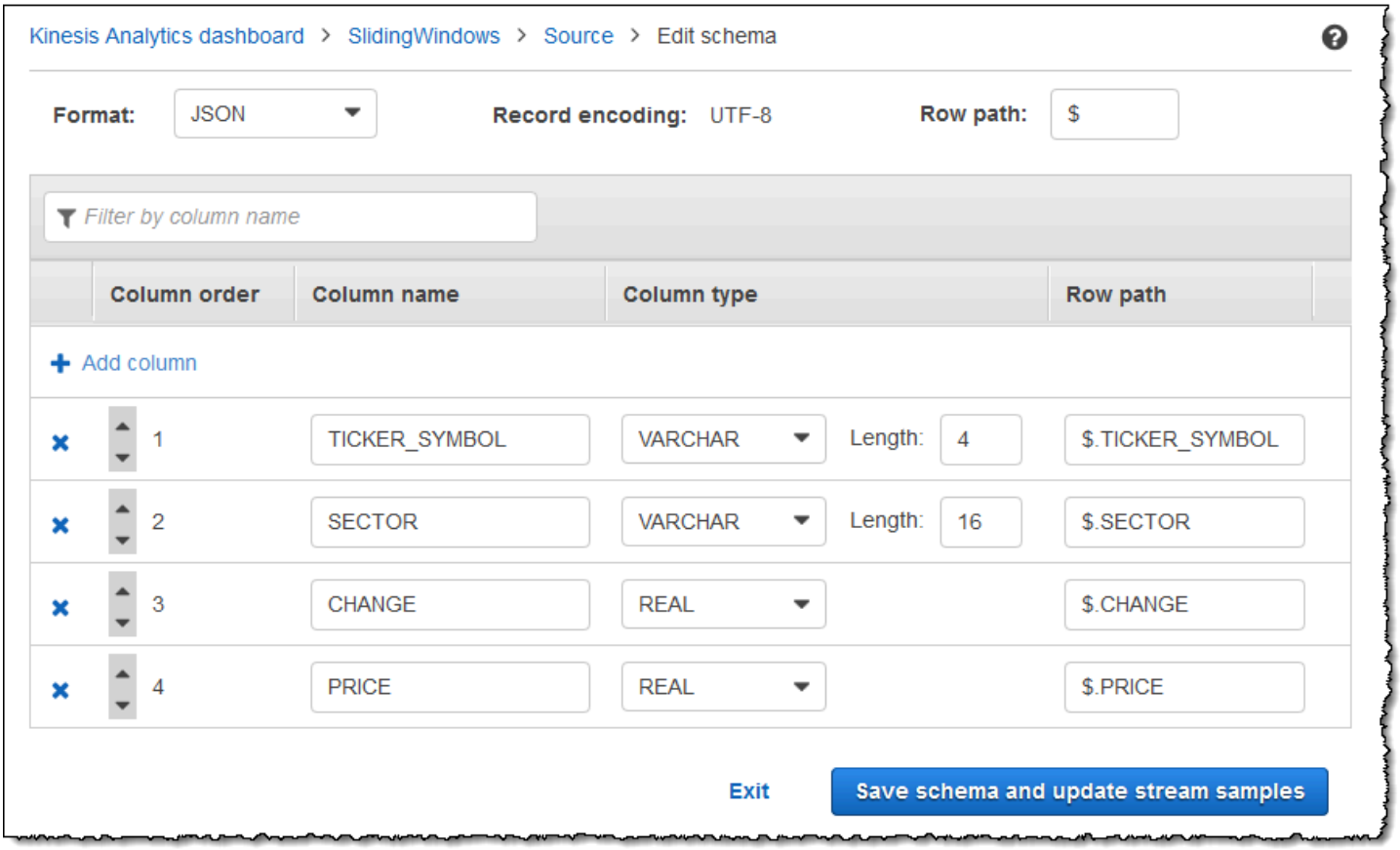
2. Para excluir uma coluna, escolha o ícone x ao lado do número da coluna.



3. Para renomear uma coluna, insira um novo nome em Column name (Nome da coluna). O novo nome de coluna não pode ficar em branco, deve ter mais de um único caractere e não deve conter palavras-chave SQL reservadas. Ele também deve atender aos critérios dos identificadores SQL comuns: deve começar com uma letra e conter apenas letras, sublinhados e dígitos.

4. Para alterar o tipo de dados de uma coluna, escolha um novo tipo de dados em Column type. Se o novo tipo de dados for `CHAR`, `VARBINARY` ou `VARCHAR`, especifique um tamanho de dados em Length (Comprimento). Para obter mais informações, consulte Tipos de dados.
5. Escolha Save schema and update stream para salvar as alterações.

O esquema modificado será exibido no editor e se parecerá com o seguinte.



Se o esquema tiver várias linhas, você poderá filtrar as linhas usando Filter by column name. Por exemplo, para editar nomes de coluna que começam com `P`, como uma coluna `Price`, insira `P` na caixa Filter by column name (Filtrar por nome de coluna).

Edição de um esquema CSV

Você pode editar um esquema CSV usando as etapas a seguir.

Para editar um esquema CSV

1. No editor de esquema, em Row delimitador, escolha o delimitador usado pelo fluxo de dados de entrada. Esse é o delimitador entre os registros de dados do fluxo, como um caractere de nova linha, por exemplo.
2. Em Column delimiter, escolha o delimitador usado pelo fluxo de dados de entrada. Esse é o delimitador entre os campos de dados do fluxo, como uma vírgula, por exemplo.
3. Para adicionar uma coluna, escolha Add column.

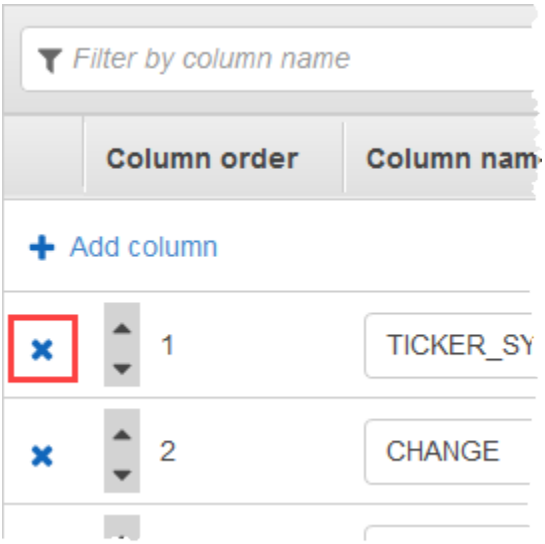
   Uma nova coluna aparece na primeira posição de coluna. Para alterar a ordem das colunas, selecione as setas para cima e para baixo ao lado do nome da coluna.

   No caso de uma nova coluna, forneça as seguintes informações:

   - Em Column name (Nome de coluna), insira um nome.

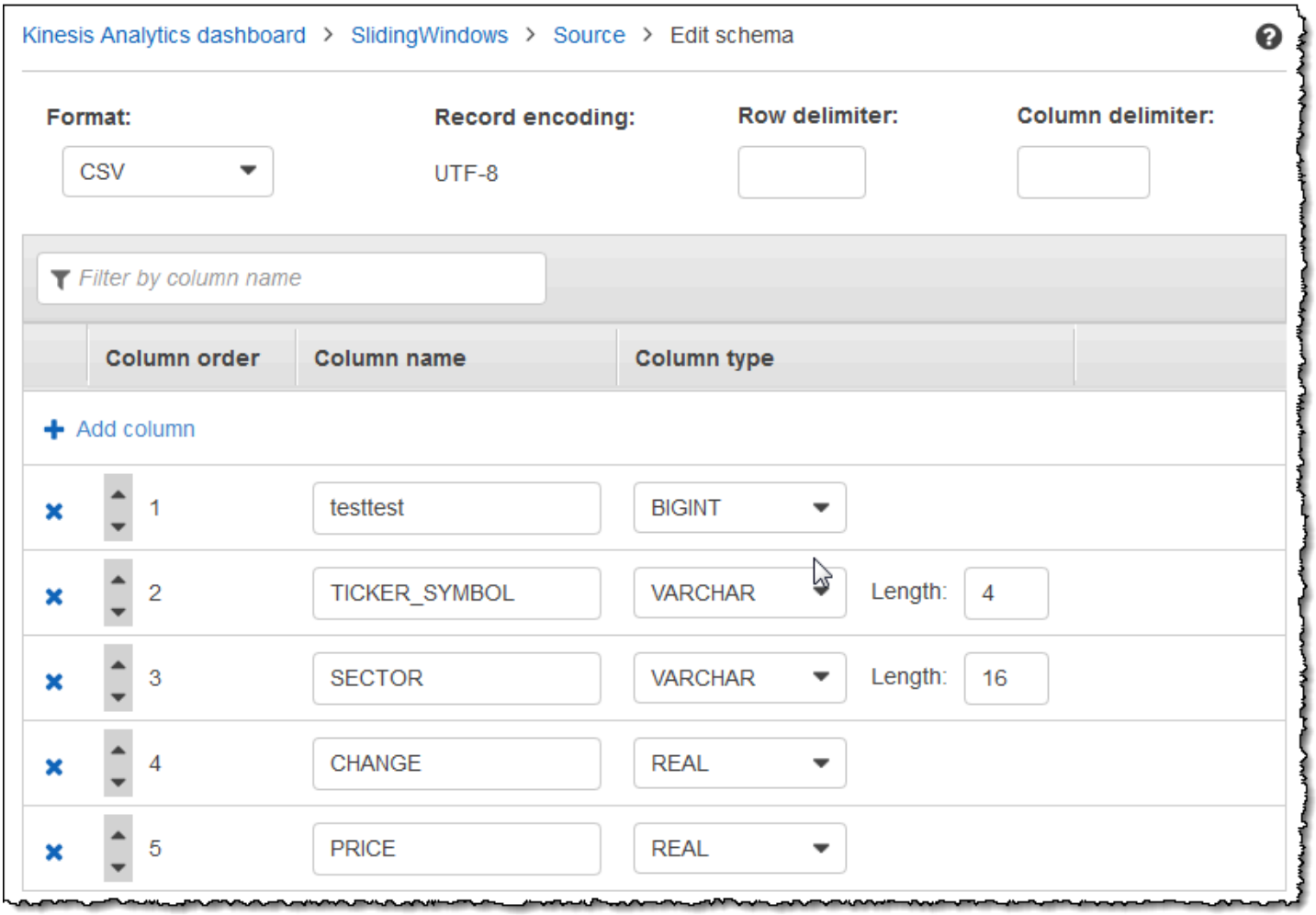
     Um nome de coluna não pode ficar em branco, deve ter mais de um único caractere e não deve conter palavras-chave SQL reservadas. Ele também deve atender aos critérios dos identificadores SQL comuns: deve começar com uma letra e conter apenas letras, sublinhados e dígitos.
   - Em Column type (Tipo de coluna), insira um tipo de dados SQL.

     Um tipo de coluna pode ser qualquer tipo de dados SQL compatível. Se o novo tipo de dados for CHAR, VARBINARY ou VARCHAR, especifique um tamanho de dados em Length. Para obter mais informações, consulte Tipos de dados.
4. Para excluir uma coluna, escolha o ícone x ao lado do número da coluna.



5. Para renomear uma coluna, insira um novo nome em Column name (Nome da coluna). O novo nome de coluna não pode ficar em branco, deve ter mais de um único caractere e não deve conter palavras-chave SQL reservadas. Ele também deve atender aos critérios dos identificadores SQL comuns: deve começar com uma letra e conter apenas letras, sublinhados e dígitos.
6. Para alterar o tipo de dados de uma coluna, escolha um novo tipo de dados em Column type. Se o novo tipo de dados for CHAR, VARBINARY ou VARCHAR, especifique um tamanho de dados em Length. Para obter mais informações, consulte Tipos de dados.
7. Escolha Save schema and update stream para salvar as alterações.

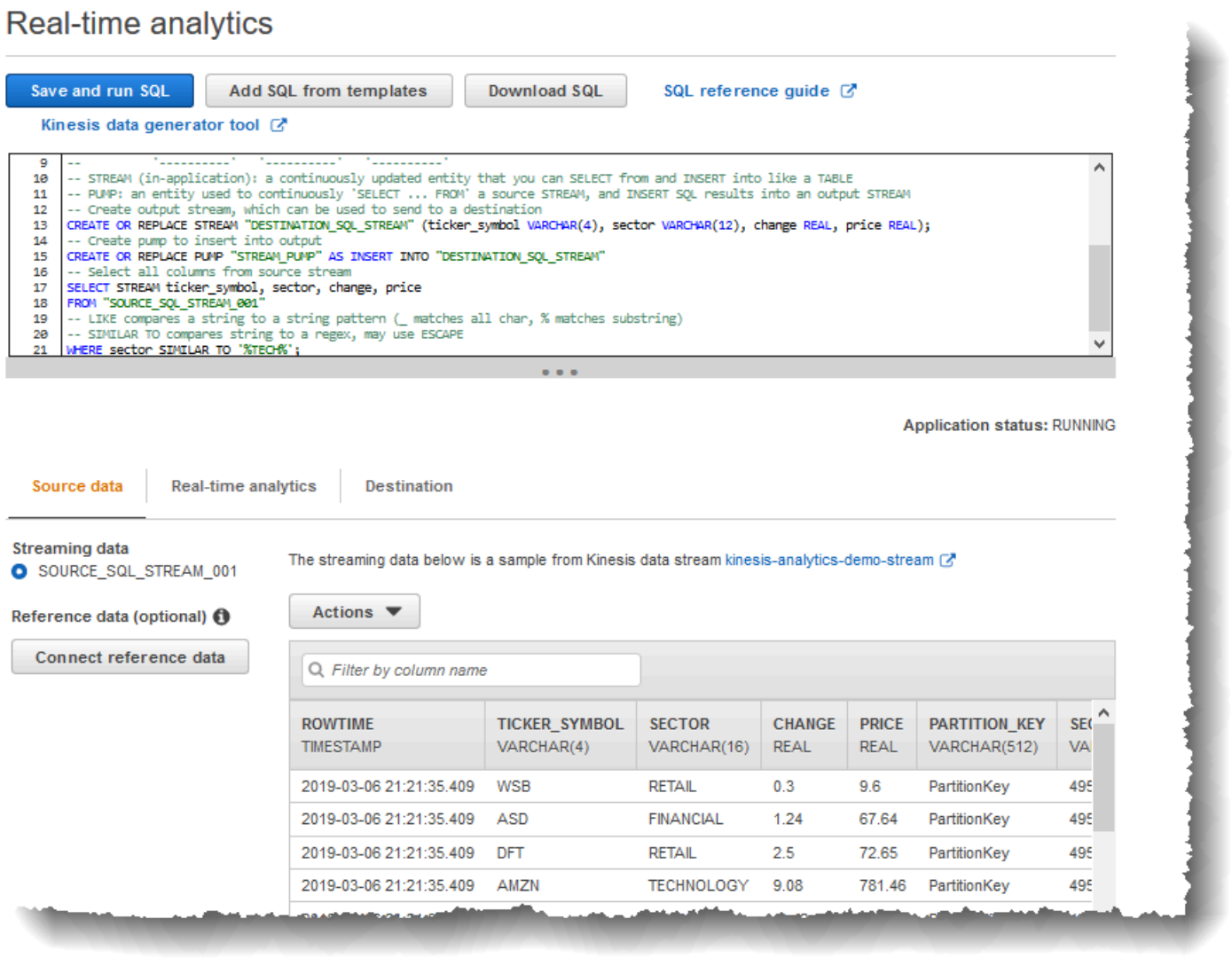
O esquema modificado será exibido no editor e se parecerá com o seguinte.



Se o esquema tiver várias linhas, você poderá filtrar as linhas usando Filter by column name. Por exemplo, para editar nomes de coluna que começam com `P`, como uma coluna `Price`, insira `P` na caixa Filter by column name (Filtrar por nome de coluna).

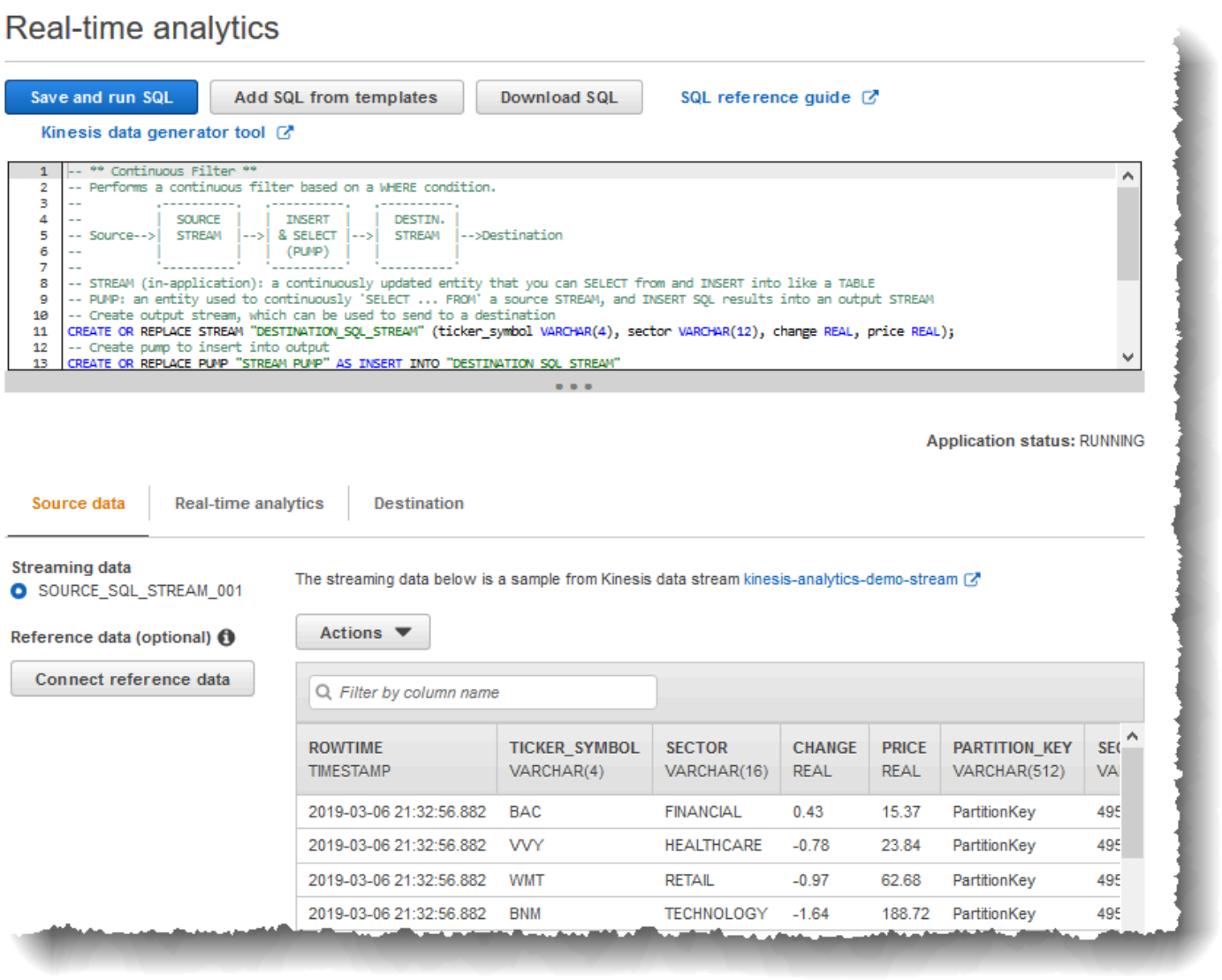
Trabalho com o Editor SQL

Veja a seguir informações sobre as seções do editor SQL e como cada uma delas funciona. No editor SQL, você pode criar seu próprio código ou escolher Add SQL from templates. Um modelo SQL fornece um exemplo de código SQL que pode ajudar você a escrever aplicativos Amazon Kinesis Data Analytics comuns. Os aplicativos de exemplo neste guia usam alguns desses modelos. Para ter mais informações, consulte Exemplos de Kinesis Data Analytics para SQL.



Guia Source data

A guia Source data identifica uma origem de streaming. Ela também identifica o stream de entrada no aplicativo que essa origem mapeia e que fornece a configuração de entrada do aplicativo.



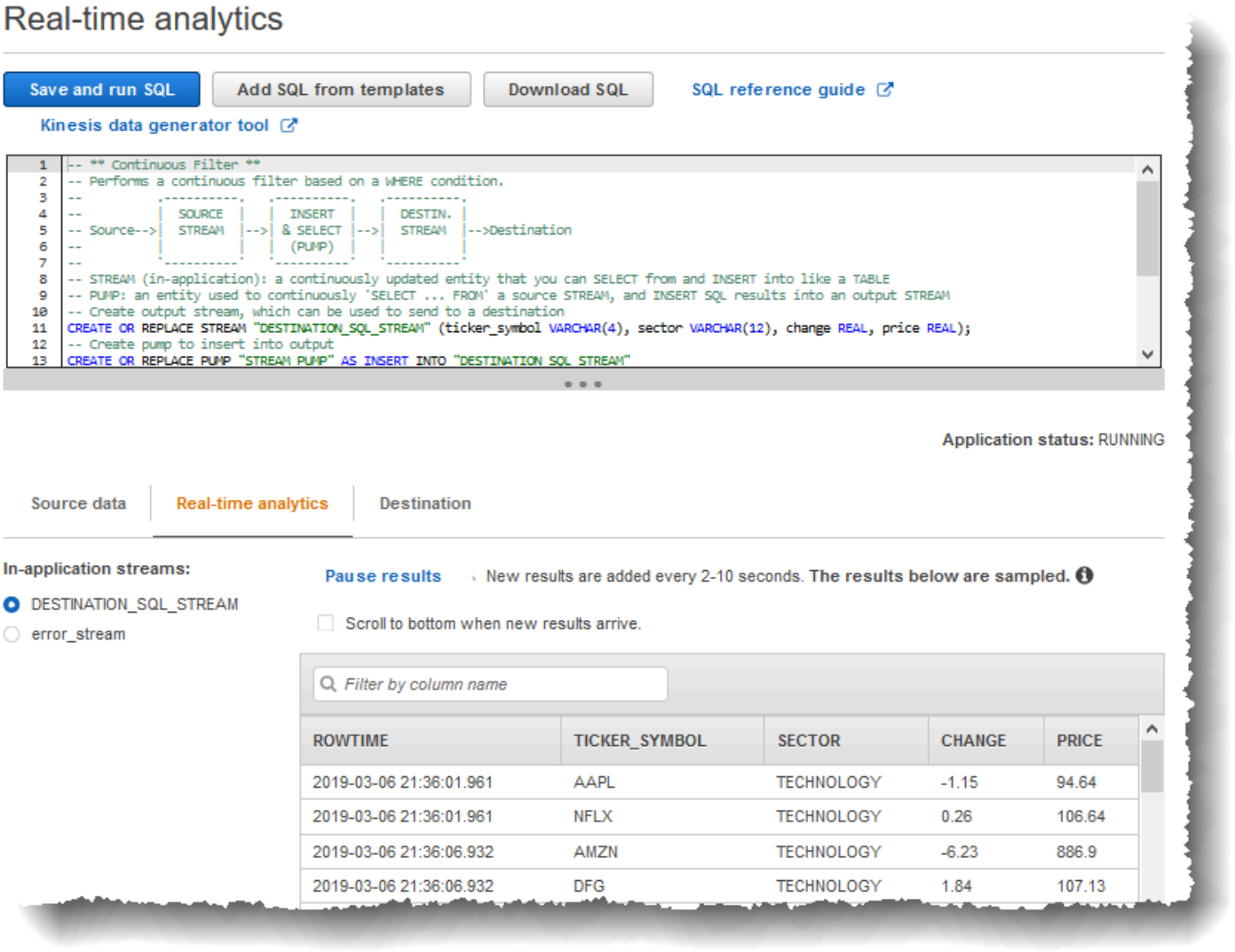
O Amazon Kinesis Data Analytics fornece as seguintes colunas de timestamp, para que você não precise fornecer mapeamento explícito na configuração de entrada:

- ROWTIME: cada linha de um stream no aplicativo tem uma coluna especial chamada `ROWTIME`. Essa coluna é o time stamp referente ao momento em que o Kinesis Data Analytics inseriu a linha no primeiro fluxo no aplicativo.
- Approximate_Arrival_Time: os registros da origem de streaming incluem a coluna `Approximate_Arrival_Timestamp`. Esse é o timestamp de chegada aproximado definido quando a origem de streaming recebe e armazena o registro relacionado com êxito. O Kinesis Data Analytics busca essa coluna no fluxo de entrada de aplicativo como `Approximate_Arrival_Time`. O Amazon Kinesis Data Analytics fornece essa coluna somente no fluxo de entrada de aplicativo mapeado para a origem de streaming.

Esses valores de timestamp são úteis nas consultas em janelas baseadas em tempo. Para ter mais informações, consulte Consultas em janelas.

Guia Real-time analytics

A guia Real-time analytics mostra todos os fluxos de aplicativo criados pelo código de aplicativo. Esse grupo de fluxos inclui o stream de erros (`error_stream`) fornecido pelo Amazon Kinesis Data Analytics para todos os aplicativos.



Guia Destination

A guia Destination (Destino) permite que você configure a saída do aplicativo para manter os fluxos no aplicativo nos destinos externos. Você pode configurar a saída para manter os dados em qualquer

um dos fluxos de aplicativo nos destinos externos. Para ter mais informações, consulte Configuração da saída do aplicativo.

Conceitos de streaming com SQL

O Amazon Kinesis Data Analytics implementa o padrão SQL ANSI 2008 com extensões. Essas extensões permitem o processamento de dados de streaming. Os tópicos a seguir abrangem os principais conceitos de streaming com SQL.

Tópicos

Bombas e fluxos no aplicativo

Ao configurar a entrada do aplicativo, mapeie uma origem de streaming para um fluxo no aplicativo que é criado. Os dados fluem continuamente da origem de streaming para o stream no aplicativo. Um stream no aplicativo funciona como uma tabela que você pode consultar usando instruções SQL, mas é chamado de stream porque representa o fluxo de dados contínuo.

> **Note**
>
> Não confunda streams no aplicativo com streams de dados do Amazon Kinesis e streams de entrega do Firehose. Os streams no aplicativo existem apenas no contexto de um aplicativo do Amazon Kinesis Data Analytics. Os fluxos de dados do Kinesis e os streams de entrega do Firehose existem independentemente do seu aplicativo. Você pode configurá-los como uma origem de streaming na configuração de entrada do aplicativo ou como um destino na configuração de saída.

Também é possível criar mais fluxos no aplicativo, conforme necessário, para armazenar resultados de consulta intermediários. A criação de um stream no aplicativo é um processo de duas etapas. Primeiro, você cria um stream no aplicativo e, em seguida, bomba dados nele. Por exemplo, suponha que a configuração de entrada do aplicativo crie um fluxo no aplicativo chamado `INPUTSTREAM`.

No exemplo a seguir, você cria outro stream (TEMPSTREAM) e, em seguida, bomba dados de INPUTSTREAM nele.

1. Crie um stream no aplicativo (TEMPSTREAM) com três colunas, como mostrado a seguir:

```
CREATE OR REPLACE STREAM "TEMPSTREAM" (
   "column1" BIGINT NOT NULL,
   "column2" INTEGER,
   "column3" VARCHAR(64));
```

   Os nomes das colunas são especificados entre aspas, diferenciando maiúsculas de minúsculas. Para obter mais informações, consulte Identificadores em Amazon Kinesis Data Analytics SQL Reference.

2. Insira dados no stream usando uma bomba. Uma bomba é uma consulta de inserção contínua em execução que insere dados de um stream no aplicativo em outro stream no aplicativo. A instrução a seguir cria uma bomba (SAMPLEPUMP) e insere dados em TEMPSTREAM selecionando registros de outro stream (INPUTSTREAM).

```
CREATE OR REPLACE PUMP "SAMPLEPUMP" AS
INSERT INTO "TEMPSTREAM" ("column1",
                          "column2",
                          "column3")
SELECT STREAM inputcolumn1,
              inputcolumn2,
              inputcolumn3
FROM "INPUTSTREAM";
```

Você pode ter várias inserções de gravadores em um stream no aplicativo, e vários leitores selecionados no stream. Pense em um fluxo no aplicativo como a implementação de um paradigma de mensagens de publicação/assinatura. Nesse paradigma, a linha de dados, incluindo o horário da criação e o horário do recebimento, pode ser processada, interpretada e encaminhada por uma cascata de instruções de streaming com SQL, sem a necessidade de ser armazenada em um RDBMS tradicional.

Após criar um stream no aplicativo, você poderá executar consultas SQL normais.

> **Note**
>
> Ao consultar fluxos, a maioria das instruções SQL são vinculados usando uma janela baseada em linha ou em horário. Para ter mais informações, consulte Consultas em janelas.

Você também pode associar streams. Para exemplos de associação, consulte Operações de dados de streaming: associações de streams.

Time stamps e a coluna ROWTIME

Os streams no aplicativo incluem uma coluna especial chamada `ROWTIME`. Ela armazena um timestamp quando o Amazon Kinesis Data Analytics insere uma linha no primeiro stream do aplicativo. `ROWTIME` reflete o timestamp no qual o Amazon Kinesis Data Analytics inseriu um registro no primeiro stream no aplicativo após ler a partir da fonte de streaming. Esse valor `ROWTIME` então é mantido em todo o aplicativo.

> **Note**
>
> Quando você bombeia registros de um stream no aplicativo para outro, não precisa copiar explicitamente a coluna `ROWTIME`, o Amazon Kinesis Data Analytics copia esta coluna para você.

O Amazon Kinesis Data Analytics garante que os valores `ROWTIME` sejam monotonicamente aumentados. É possível usar esse time stamp nas consultas em janelas baseadas em tempo. Para ter mais informações, consulte Consultas em janelas.

Você pode acessar a coluna ROWTIME na instrução `SELECT` como qualquer outra coluna no fluxo no aplicativo. Por exemplo: .

```
SELECT STREAM ROWTIME,
              some_col_1,
              some_col_2
FROM  SOURCE_SQL_STREAM_001
```

Compreensão de vários horários da análise de streaming

Além do `ROWTIME`, há outros tipos de horários em aplicativos de streaming em tempo real. Eles são:

- Horário do evento: o time stamp em que o evento ocorreu. Isso também é às vezes chamado de horário do cliente. Muitas vezes, é desejável usar esse horário em análises, porque é o momento em que um evento ocorreu. No entanto, muitas fontes de eventos, como celulares e clientes da Web, não têm relógios confiáveis, o que pode levar a horários imprecisos. Além disso, problemas de conectividade podem levar a registros que aparecem em um stream não na mesma ordem em que os eventos ocorreram.

- Horário de ingestão: o time stamp de quando o registro foi adicionada à origem de streaming. O Amazon Kinesis Data Streams inclui um campo chamado `APPROXIMATE_ARRIVAL_TIME` em cada registro que fornece esse time stamp. Isso também é às vezes chamado de horário do servidor. Esse horário de inclusão normalmente está muito próximo do horário do evento. Se houver qualquer tipo de atraso na inclusão do registro no stream, pode haver imprecisões, que são normalmente raras. Além disso, o horário de inclusão raramente está fora de ordem, mas pode ocorrer devido à natureza distribuída dos dados de streaming. Portanto, o horário de inclusão é na maioria das vezes preciso e reflete a ordem do horário do evento.

- Horário do processamento: o time stamp quando o Amazon Kinesis Data Analytics insere uma linha no primeiro stream no aplicativo. O Amazon Kinesis Data Analytics fornece esse time stamp na coluna `ROWTIME` existente em cada stream no aplicativo. O tempo de processamento está sempre aumentando monotonicamente. Mas ele não será preciso se o aplicativo ficar atrasado. (Se um aplicativo ficar atrasado, o tempo de processamento não refletirá com precisão o horário do evento.) Esse `ROWTIME` é preciso em relação ao relógio, mas pode não representar o horário em que o evento realmente ocorreu.

Usar cada um desses horários nas consultas em janelas baseadas em horário tem vantagens e desvantagens. Recomendamos que você escolha um ou mais desses horários e uma estratégia para lidar com as desvantagens relevantes de acordo com o caso de uso.

> **Note**
>
> Se você estiver usando janelas baseadas em linha, o horário não será um problema e você poderá ignorar esta seção.

Recomendamos uma estratégia de duas janelas que usam dois horários, o ROWTIME e um dos outros horários (de inclusão ou do evento).

- Use o ROWTIME como a primeira janela, que controla a frequência com que a consulta emite os resultados, como mostrado no exemplo a seguir. Ele não é usado como um horário lógico.
- Use um dos outros horários lógicos que deseja associar à sua análise. Esse horário representa quando o evento ocorreu. No exemplo a seguir, o objetivo da análise é agrupar os registros e retornar a contagem pelo marcador.

A vantagem dessa estratégia é que ela pode usar um horário que representa o momento em que o evento ocorreu. Ela pode lidar tranquilamente com atrasos de seu aplicativo ou com a chegada de eventos fora de ordem. Se o aplicativo atrasar ao colocar registros no stream no aplicativo, eles ainda estarão agrupados pelo horário lógico na segunda janela. A consulta usa o ROWTIME para garantir a ordem de processamento. Todos os registros atrasados (o time stamp de inclusão mostra um valor anterior comparado com o valor do ROWTIME) também são processados com êxito.

Considere a seguinte consulta em relação ao fluxo de demonstração usado no Exercício de conceitos básicos. A consulta usa a cláusula GROUP BY e emite uma contagem do marcador em uma janela em cascata de um minuto.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM"
    ("ingest_time"    timestamp,
    "APPROXIMATE_ARRIVAL_TIME" timestamp,
    "ticker_symbol"  VARCHAR(12),
    "symbol_count"        integer);


CREATE OR REPLACE PUMP "STREAM_PUMP" AS
    INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND) AS
 "ingest_time",
        STEP("SOURCE_SQL_STREAM_001".APPROXIMATE_ARRIVAL_TIME BY INTERVAL '60' SECOND)
 AS "APPROXIMATE_ARRIVAL_TIME",
        "TICKER_SYMBOL",
        COUNT(*) AS "symbol_count"
    FROM "SOURCE_SQL_STREAM_001"
    GROUP BY "TICKER_SYMBOL",
        STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND),
        STEP("SOURCE_SQL_STREAM_001".APPROXIMATE_ARRIVAL_TIME BY INTERVAL '60' SECOND);
```

No `GROUP BY`, primeiro agrupe os registros baseados no `ROWTIME` em uma janela de um minuto e, em seguida, pelo `APPROXIMATE_ARRIVAL_TIME`.

Os valores de time stamp no resultado são arredondados para baixo para o intervalo de 60 segundos mais próximo. O primeiro grupo de resultados emitidos pela consulta mostra registros no primeiro minuto. O segundo grupo de resultados emitidos mostra os registros nos próximos minutos com base no `ROWTIME`. O último registro indica que o aplicativo atrasou a entrega do registro no stream no aplicativo (ele mostra um valor `ROWTIME` atrasado em comparação com o time stamp de inclusão).

```
ROWTIME                 INGEST_TIME     TICKER_SYMBOL  SYMBOL_COUNT

--First one minute window.
2016-07-19 17:05:00.0    2016-07-19 17:05:00.0    ABC      10
2016-07-19 17:05:00.0    2016-07-19 17:05:00.0    DEF      15
2016-07-19 17:05:00.0    2016-07-19 17:05:00.0    XYZ      6
--Second one minute window.
2016-07-19 17:06:00.0    2016-07-19 17:06:00.0    ABC      11
2016-07-19 17:06:00.0    2016-07-19 17:06:00.0    DEF      11
2016-07-19 17:06:00.0    2016-07-19 17:05:00.0    XYZ      1  ***

***late-arriving record, instead of appearing in the result of the
first 1-minute windows (based on ingest_time, it is in the result
of the second 1-minute window.
```

É possível combinar os resultados de uma contagem final precisa por minuto enviando os resultados para um banco de dados de downstream. Por exemplo, você pode configurar a saída do aplicativo para manter os resultados em um stream de entrega do Firehose que pode ser gravado em uma tabela do Amazon Redshift. Depois que os resultados estiverem em uma tabela do Amazon Redshift, será possível consultar esta tabela para calcular o grupo de contagem total pelo `Ticker_Symbol`. No caso do XYZ, o total é preciso (6+1), mesmo que um registro chegue atrasado.

Consultas contínuas

Uma consulta por um stream é executada continuamente pelos dados de streaming. Essa execução contínua permite cenários, como a capacidade de aplicativos consultarem continuamente um stream e gerar alertas.

No exercício de Conceitos básicos, há um fluxo no aplicativo chamado `SOURCE_SQL_STREAM_001`. Ele recebe continuamente os preços de ações a partir de um fluxo de demonstração (um fluxo de dados do Kinesis). O esquema é o seguinte:

```
(TICKER_SYMBOL VARCHAR(4),
 SECTOR varchar(16),
 CHANGE REAL,
 PRICE REAL)
```

Suponha que você tenha interesse em alterações nos preços de ações acima de 15%. Você poderá usar a seguinte consulta no código do aplicativo. Essa consulta é executada continuamente e emite registros quando uma alteração acima de 15% é detectada no preço de uma ação.

```
SELECT STREAM TICKER_SYMBOL, PRICE
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  (ABS((CHANGE / (PRICE-CHANGE)) * 100)) > 15
```

Use o procedimento a seguir para configurar um aplicativo do Amazon Kinesis Data Analytics e testar essa consulta.

Para testar a consulta

1. Crie um aplicativo seguindo as instruções em Exercício de conceitos básicos.
2. Substitua a instrução SELECT no código de aplicativo pela consulta SELECT anterior. O código de aplicativo resultante é mostrado a seguir:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (ticker_symbol VARCHAR(4),
                                                   price DOUBLE);
-- CREATE OR REPLACE PUMP to insert into output
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
  INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM TICKER_SYMBOL,
                    PRICE
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  (ABS((CHANGE / (PRICE-CHANGE)) * 100)) > 15;
```

Consultas em janelas

As consultas SQL no código de aplicativo são executadas continuamente em streams no aplicativo. Um fluxo no aplicativo representa dados não vinculados que fluem continuamente pelo aplicativo. Portanto, para que os conjuntos de resultados desta entrada sejam continuamente atualizados, as

consultas são frequentemente vinculadas usando uma janela definida em termos de tempo ou linhas. Elas são também chamadas de SQL em janelas.

Para obter uma consulta em janelas baseada em horário, especifique o tamanho da janela em termos de tempo (por exemplo, uma janela de um minuto). Isso requer uma coluna de time stamp no fluxo no aplicativo que aumente monotonicamente. (O time stamp de uma nova linha é maior ou igual ao da linha anterior.) O Amazon Kinesis Data Analytics fornece uma coluna de time stamp chamada `ROWTIME` para cada stream de aplicativo. É possível usar essa coluna especificando consultas baseadas em tempo. Para o seu aplicativo, escolha outra opção de time stamp. Para ter mais informações, consulte Time stamps e a coluna ROWTIME.

Para obter uma consulta em janela baseada em linha, especifique o tamanho da janela em termos do número de linhas.

É possível especificar uma consulta para processar registros em uma janela em cascata, janela deslizante ou janela de escalonar, dependendo das necessidades do aplicativo. O Kinesis Data Analytics é compatível com os seguintes tipos de janela:

- Janelas de escalonamento: uma consulta que agrega dados usando janelas baseadas em horário com chave que abre com a chegada dos dados. As chaves permitem várias janelas sobrepostas. Essa é a maneira recomendada de agregar dados usando janelas baseadas em tempo, porque as janelas escalonadas reduzem os atrasos ou os out-of-order dados em comparação com as janelas Tumbling.
- Janelas em cascata: uma consulta que agrega dados usando diferentes janelas baseadas em horário que abrem e fecham em intervalos regulares
- Janelas deslizantes: uma consulta que agrega dados continuamente, usando um horário fixo ou um intervalo de contagem de linhas.

Janelas de escalonamento

Usar janelas de escalonamento é um método de janelas adequado para analisar grupos de dados que chegam em horários inconsistentes. Ele é ideal para qualquer caso de uso de análise de séries temporais, como um conjunto de vendas relacionadas ou registros de log.

Por exemplo, os logs de fluxo da VPC têm uma janela de captura de cerca de dez minutos. No entanto, podem ter uma janela de captura de até 15 minutos se você estiver agregando dados no cliente. As janelas de escalonamento são ideais para agregar esses logs para análise.

As janelas de escalonamento resolvem o problema de registros relacionados sem cair na mesma janela de horário restrito, como quando as janelas em cascata eram usadas.

Resultados parciais com janelas em cascata

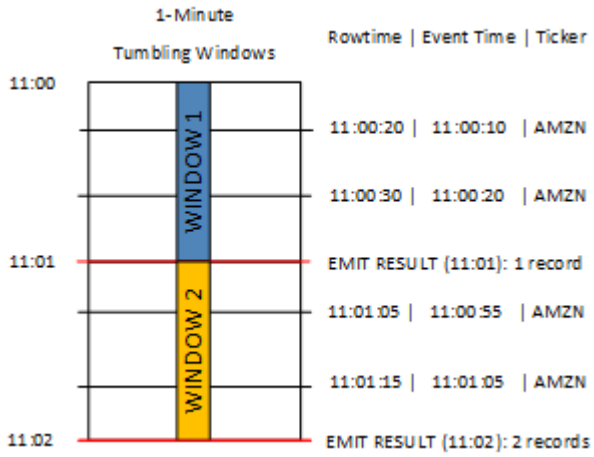
Há certas limitações no uso de Janelas em cascata para agregar dados atrasados ou fora de ordem.

Se as janelas em cascata são usados para analisar grupos de dados relacionados ao horário, os registros individuais podem ser colocados em janelas separadas. Portanto, os resultados parciais de cada janela devem ser combinados posteriormente a fim de produzir resultados completos para cada grupo de registros.

Na consulta de janela em cascata a seguir, os registros são agrupados em janelas por rowtime, horário do evento e símbolo de índice:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    TICKER_SYMBOL VARCHAR(4),
    EVENT_TIME timestamp,
    TICKER_COUNT     DOUBLE);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS
  INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM
        TICKER_SYMBOL,
        FLOOR(EVENT_TIME TO MINUTE),
        COUNT(TICKER_SYMBOL) AS TICKER_COUNT
    FROM "SOURCE_SQL_STREAM_001"
    GROUP BY ticker_symbol, FLOOR(EVENT_TIME TO MINUTE),
 STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '1' MINUTE);
```

No diagrama a seguir, um aplicativo está contando o número de transações que recebe, com base em quando elas ocorreram (horário do evento) com um minuto de granularidade. O aplicativo pode usar uma janela em cascata para agrupar dados com base em rowtime e horário do evento. O aplicativo recebe quatro registros de que tudo chegou com um minuto entre um e outro. Ele agrupa os registros por rowtime, horário do evento e símbolo de índice. Como alguns registros chegam depois que a primeira janela em cascata termina, os registros não ficam na mesma janela em cascata de um minuto.



O diagrama anterior tem os seguintes eventos.

ROWTIME	EVENT_TIME	TICKER_SYMBOL
11:00:20	11:00:10	AMZN
11:00:30	11:00:20	AMZN
11:01:05	11:00:55	AMZN
11:01:15	11:01:05	AMZN

O conjunto de resultados do aplicativo da janela em cascata é semelhante ao seguinte.

ROWTIME	EVENT_TIME	TICKER_SYMBOL	COUNT
11:01:00	11:00:00	AMZN	2
11:02:00	11:00:00	AMZN	1
11:02:00	11:01:00	AMZN	1

No conjunto de resultados anterior, três resultados foram retornados:

- Um registro com um `ROWTIME` de 11:01:00 que agrega os dois primeiros registros.

- Um registro em 11:02:00 que agrega somente o terceiro registro. Esse registro tem um `ROWTIME` na segunda janela, mas um `EVENT_TIME` na primeira janela.
- Um registro em 11:02:00 que agrega somente o quarto registro.

Para analisar o conjunto de resultados completo, os registros devem ser agregados no armazenamento de persistência. Isso adiciona complexidade e requisitos de processamento ao aplicativo.

Concluir resultados com janelas de escalonamento

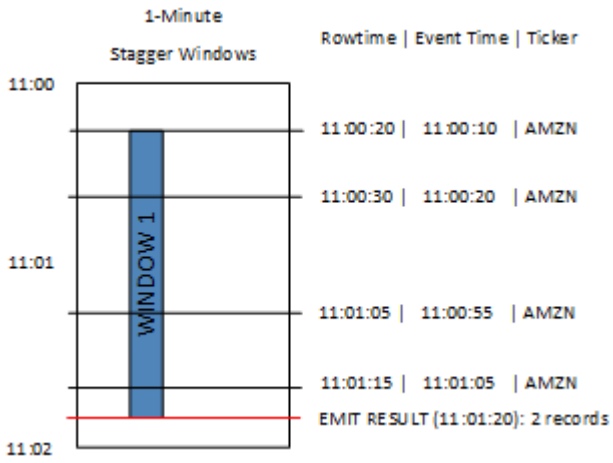
Para melhorar a precisão da análise dos registros de dados relacionados ao horário, o Kinesis Data Analytics oferece um novo tipo de janela chamado janelas de escalonamento. Nesse tipo de janela, as janelas abrem quando o primeiro evento relacionado à chave de partição chega e não em um intervalo de tempo fixo. As janelas fecham com base na idade especificada, que é medida a partir do horário em que a janela foi aberta.

Uma janela de escalonamento é uma janela de restrição de horário separada para cada grupo de chaves em uma cláusula de janela. O aplicativo agrega cada resultado da cláusula de janela em sua própria janela de horário, em vez de usar uma única janela para todos os resultados.

Na seguinte consulta em janela de escalonamento, os registros são agrupados em janelas por horário do evento e símbolo de índice:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    ticker_symbol    VARCHAR(4),
    event_time       TIMESTAMP,
    ticker_count     DOUBLE);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS
  INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM
        TICKER_SYMBOL,
        FLOOR(EVENT_TIME TO MINUTE),
        COUNT(TICKER_SYMBOL) AS ticker_count
    FROM "SOURCE_SQL_STREAM_001"
    WINDOWED BY STAGGER (
            PARTITION BY FLOOR(EVENT_TIME TO MINUTE), TICKER_SYMBOL RANGE INTERVAL '1'
 MINUTE);
```

No seguinte diagrama, os eventos são agregados por horário do evento e símbolo de índice em janelas de escalonamento.



O diagrama anterior tem os seguintes eventos, que são os mesmos eventos que o aplicativo da janela em cascata analisou:

ROWTIME	EVENT_TIME	TICKER_SYMBOL
11:00:20	11:00:10	AMZN
11:00:30	11:00:20	AMZN
11:01:05	11:00:55	AMZN
11:01:15	11:01:05	AMZN

O conjunto de resultados do aplicativo da janela de escalonamento é semelhante ao seguinte.

ROWTIME	EVENT_TIME	TICKER_SYMBOL	Contagem
11:01:20	11:00:00	AMZN	3
11:02:15	11:01:00	AMZN	1

O registro retornado agrega os primeiros três registros de entrada. Os registros são agrupados por janelas de escalonamento de um minuto. A janela de escalonamento inicia quando o aplicativo recebe o primeiro registro AMZN (com um `ROWTIME` de 11:00:20). Quando a janela de

escalonamento de um minuto expirar (às 11:01:20), um registro com os resultados que ficam na janela de escalonamento (com base em `ROWTIME` e `EVENT_TIME`) será gravado no fluxo de saída. Ao usar uma janela de escalonamento, todos os registros com `ROWTIME` e `EVENT_TIME` em uma janela de um minuto serão emitidos em um único resultado.

O último registro (com um `EVENT_TIME` fora da agregação de um minuto) será agregado separadamente. Isso ocorre porque `EVENT_TIME` é uma das chaves de partição usada para separar os registros em conjuntos de resultados e a chave de partição de `EVENT_TIME` para a primeira janela é `11:00`.

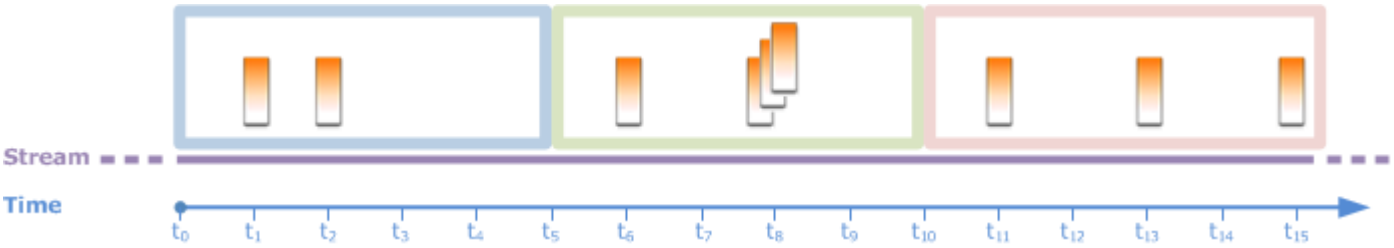
A sintaxe para uma janela de escalonamento é definida em uma cláusula especial, `WINDOWED BY`. Essa cláusula é usada em vez da cláusula `GROUP BY` para agregações de streaming. A cláusula aparece imediatamente após a cláusula opcional `WHERE` e antes da cláusula `HAVING`.

A janela de escalonamento é definida na cláusula `WINDOWED BY` e considera dois parâmetros: chaves de partição e tamanho da janela. As chaves de partição particionam o streaming de dados de entrada e definem quando a janela abre. Uma janela de escalonamento abre quando o primeiro evento com uma chave de partição exclusiva aparece no fluxo. A janela de escalonamento fecha após um período fixo definido pelo tamanho da janela. A sintaxe é mostrada no seguinte exemplo de código:

```
...
FROM <stream-name>
WHERE <... optional statements...>
WINDOWED BY STAGGER(
 PARTITION BY <partition key(s)>
 RANGE INTERVAL <window length, interval>
);
```

Janelas em cascata (Agregações usando GROUP BY)

Quando uma consulta em janela processa cada janela de forma não sobreposta, é chamada de janela em cascata. Nesse caso, cada registro em um fluxo no aplicativo pertence a uma janela específica. Ele é processado somente uma vez (quando a consulta processa a janela à qual o registro pertence).



Por exemplo, uma consulta de agregação que usa uma cláusula GROUP BY processa linhas em uma janela em cascata. O fluxo de demonstração no exercício de conceitos básicos recebe os dados de preços de ações que são mapeados no fluxo no aplicativo SOURCE_SQL_STREAM_001. O stream tem o seguinte esquema:

```
(TICKER_SYMBOL VARCHAR(4),
 SECTOR varchar(16),
 CHANGE REAL,
 PRICE REAL)
```

Em seu código de aplicativo, suponha que você queira encontrar preços agregados (mínimo, máximo) para cada marcador em uma janela de um minuto. É possível usar a seguinte consulta.

```
SELECT STREAM ROWTIME,
              Ticker_Symbol,
              MIN(Price) AS Price,
              MAX(Price) AS Price
FROM     "SOURCE_SQL_STREAM_001"
GROUP BY Ticker_Symbol,
         STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND);
```

Esse é um exemplo de consulta em janela baseada em horário. A consulta agrupa registros por valores de ROWTIME. Para relatórios por minuto, a função STEP arredonda os valores ROWTIME para baixo, para o minuto mais próximo.

> **Note**
>
> Você também pode usar a função FLOOR para agrupar registros em janelas. No entanto, FLOOR só pode arredondar os valores de tempo para uma unidade de tempo inteira (hora, minuto, segundo e assim por diante). STEP é recomendada para agrupar registros em janelas em cascata porque, assim, pode arredondar valores para baixo para um intervalo arbitrário, por exemplo, 30 segundos.

Essa consulta é um exemplo de janela não sobreposta (em cascata). Os registros de grupos da cláusula GROUP BY em uma janela de um minuto e cada registro pertence a uma janela específica (não sobreposta). A consulta emite um registro de saída por minuto, fornecendo o preço de marcador mín./máx. registrado no minuto específico. Esse tipo de consulta é útil para gerar relatórios periódicos do stream de dados de entrada. Neste exemplo, os relatórios são gerados a cada minuto.

Para testar a consulta

1. Configure um aplicativo seguindo as instruções no exercício de conceitos básicos.
2. Substitua a instrução SELECT no código de aplicativo pela consulta SELECT anterior. O código de aplicativo resultante é mostrado a seguir:

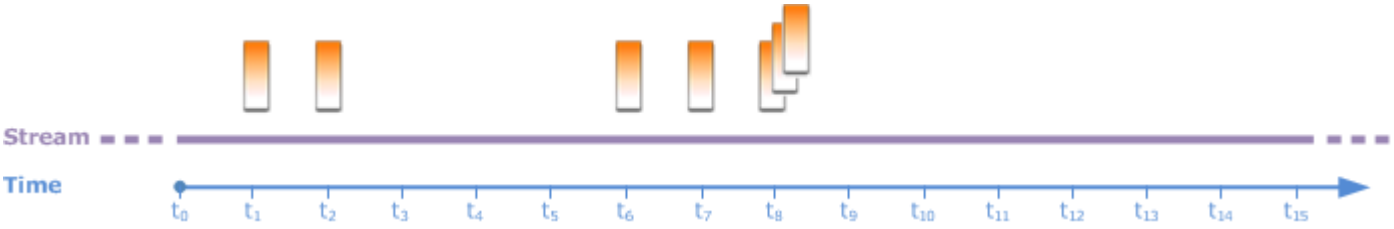
```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
                                    ticker_symbol VARCHAR(4),
                                    Min_Price     DOUBLE,
                                    Max_Price     DOUBLE);
-- CREATE OR REPLACE PUMP to insert into output
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
  INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM Ticker_Symbol,
                  MIN(Price) AS Min_Price,
                  MAX(Price) AS Max_Price
    FROM    "SOURCE_SQL_STREAM_001"
    GROUP BY Ticker_Symbol,
             STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND);
```

Janelas deslizantes

Em vez de agrupar registros usando GROUP BY, defina uma janela baseada em horário ou em linha. Para isso, adicione uma cláusula WINDOW explícita.

Nesse caso, como a janela desliza com o tempo, o Amazon Kinesis Data Analytics emite uma saída quando novos registros aparecem no stream. O Kinesis Data Analytics emite essa saída ao processar as linhas na janela. As janelas podem se sobrepor neste tipo de processamento, e um registro pode fazer parte de várias janelas e ser processado em cada janela. O exemplo a seguir ilustra uma janela deslizante.

Considere uma consulta simples que conta registros no stream. Este exemplo assume uma janela de cinco segundos. No stream de exemplo a seguir, novos registros chegam no horário t_1, t_2, t_6, and t_7, e três registros chegam no horário t_8 segundos.



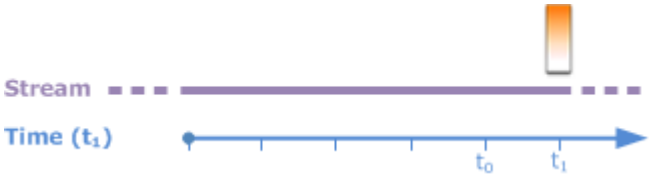
Lembre-se do seguinte:

- O exemplo assume uma janela de cinco segundos. A janela de 5 segundos desliza continuamente com o tempo.
- Para cada linha que entra em uma janela, uma linha de saída é emitida pela janela deslizante. Logo após o aplicativo iniciar, a consulta emitirá uma saída para cada novo registro que aparecer no stream, embora uma janela de 5 segundos não tenha passado ainda. Por exemplo, a consulta emite a saída quando um registro aparece no primeiro segundo e no segundo. Mais tarde, a consulta processará registros na janela de 5 segundos.
- As janelas deslizam com o tempo. Se um registro antigo no stream ficar fora da janela, a consulta não emitirá a saída, a menos que também haja um novo registro no stream que fique dentro da janela de 5 segundos.

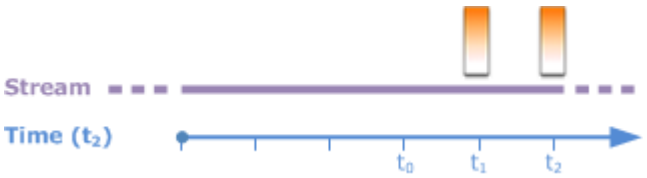
Suponha que a consulta comece a ser executada em t_0. Ocorrerá o seguinte:

1. No horário t_0, a consulta será iniciada. A consulta não emitirá a saída (valor da contagem), porque não haverá registros nesse horário.

2. No momento t_1, um novo registro aparece no stream e a consulta emite o valor de contagem 1.



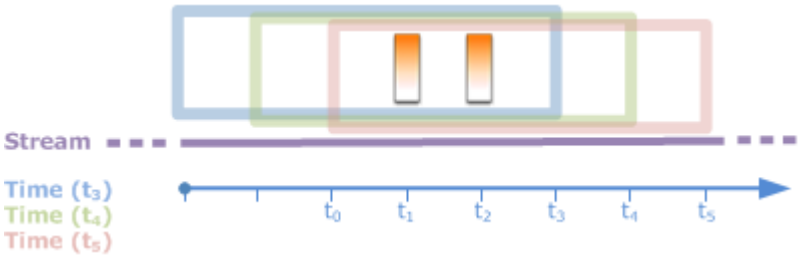
3. No momento t_2, outro registro aparece e a consulta emite a contagem 2.



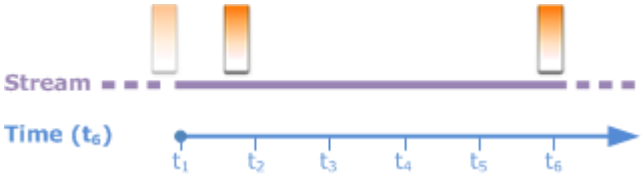
4. A janela de 5 segundos desliza com o tempo:

   - No t_3, a janela deslizante t_3 para t_0
   - No t4 (janela deslizante t_4 para t_0)
   - No t_5, a janela deslizante t_5 - t_0

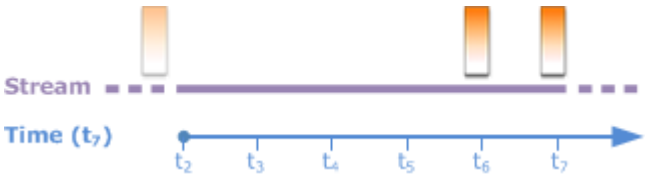
   Em todos esses horários, a janela de 5 segundos tem os mesmos registros - não há novos registros. Portanto, a consulta não emite saídas.



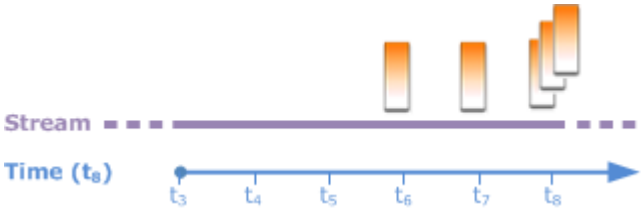
5. No momento t_6, a janela de 5 segundos é (t_6 para t_1). A consulta detecta um novo registro no t_6; assim, ele emite a saída 2. O registro no t_1 não está mais na janela e não conta.



6. No momento t_7, a janela de 5 segundos é t_7 para t_2. A consulta detecta um novo registro no t_7; assim, ele emite a saída 2. O registro no t_2 não está mais na janela de 5 segundos e, portanto, não é contado.



7. No momento t_8, a janela de 5 segundos é t_8 para t_3. A consulta detecta três novos registros e, portanto, emite a contagem de registro 5.



Resumindo, a janela tem um tamanho fixo e desliza com o tempo. A consulta emite a saída quando novos registros são exibidos.

> **Note**
>
> Recomendamos que você use uma janela deslizante por não mais de uma hora. Se você usar uma janela maior, o aplicativo levará mais tempo para reiniciar após a manutenção regular do sistema. Isso ocorre porque os dados de origem precisam ser lidos no fluxo novamente.

Os exemplos de consultas a seguir usam a cláusula `WINDOW` para definir janelas e executar agregados. Como as consultas não especificam o `GROUP BY`, a consulta usa a abordagem de janela deslizante para processar registros no stream.

Exemplo 1: processar um fluxo usando uma janela deslizante de um minuto

Considere o stream de demonstração no exercício de conceitos básicos que preenche o stream no aplicativo, `SOURCE_SQL_STREAM_001`. Este é o esquema.

```
(TICKER_SYMBOL VARCHAR(4),
 SECTOR varchar(16),
 CHANGE REAL,
 PRICE REAL)
```

Suponha que você queira que o aplicativo calcule os agregados usando uma janela deslizante de 1 minuto. Ou seja, para cada novo registro que aparecer no stream, você quer que o aplicativo emita uma saída aplicando agregados aos registros na janela de 1 minuto anterior.

É possível usar as seguintes consultas em janela baseadas em horário. A consulta usa a cláusula `WINDOW` para definir o intervalo de 1 minuto. O `PARTITION BY` na cláusula `WINDOW` agrupa os registros por valores do marcador dentro da janela deslizante.

```
SELECT STREAM ticker_symbol,
              MIN(Price) OVER W1 AS Min_Price,
              MAX(Price) OVER W1 AS Max_Price,
              AVG(Price) OVER W1 AS Avg_Price
FROM   "SOURCE_SQL_STREAM_001"
WINDOW W1 AS (
   PARTITION BY ticker_symbol
   RANGE INTERVAL '1' MINUTE PRECEDING);
```

Para testar a consulta

1. Configure um aplicativo seguindo as instruções em Exercício de conceitos básicos.
2. Substitua a instrução SELECT no código de aplicativo pela consulta SELECT anterior. O código de aplicativo resultante é o seguinte.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
                        ticker_symbol VARCHAR(10),
                        Min_Price     double,
                        Max_Price     double,
                        Avg_Price     double);
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
     SELECT STREAM ticker_symbol,
                   MIN(Price) OVER W1 AS Min_Price,
                   MAX(Price) OVER W1 AS Max_Price,
                   AVG(Price) OVER W1 AS Avg_Price
     FROM   "SOURCE_SQL_STREAM_001"
     WINDOW W1 AS (
        PARTITION BY ticker_symbol
        RANGE INTERVAL '1' MINUTE PRECEDING);
```

Exemplo 2: agregações de aplicação de consulta em uma janela deslizante

A consulta a seguir no stream de demonstração retorna a média de percentual de alteração no preço de cada marcador em uma janela de 10 segundos.

```
SELECT STREAM Ticker_Symbol,
              AVG(Change / (Price - Change)) over W1 as Avg_Percent_Change
FROM "SOURCE_SQL_STREAM_001"
WINDOW W1 AS (
```

```
        PARTITION BY ticker_symbol
        RANGE INTERVAL '10' SECOND PRECEDING);
```

Para testar a consulta

1. Configure um aplicativo seguindo as instruções em Exercício de conceitos básicos.
2. Substitua a instrução SELECT no código de aplicativo pela consulta SELECT anterior. O código de aplicativo resultante é o seguinte.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
                            ticker_symbol VARCHAR(10),
                            Avg_Percent_Change double);
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM Ticker_Symbol,
                    AVG(Change / (Price - Change)) over W1 as Avg_Percent_Change
      FROM "SOURCE_SQL_STREAM_001"
      WINDOW W1 AS (
              PARTITION BY ticker_symbol
              RANGE INTERVAL '10' SECOND PRECEDING);
```

Exemplo 3: dados de consulta de várias janelas deslizantes no mesmo stream

É possível gravar consultas para emitir saída em que cada valor da coluna que é calculado usando diferentes janelas deslizantes definidas no mesmo stream.

No exemplo a seguir, a consulta emite o marcador de saída, preço, a2 e a10. Ela emite a saída de símbolos de marcador cuja média de movimentação de duas cruza a média de movimentação de dez linhas. Os valores das colunas a2 e a10 são derivados de janelas deslizantes de duas linhas e dez linhas.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
                          ticker_symbol      VARCHAR(12),
                          price              double,
                          average_last2rows  double,
                          average_last10rows double);

CREATE OR REPLACE PUMP "myPump" AS INSERT INTO "DESTINATION_SQL_STREAM"
SELECT STREAM ticker_symbol,
              price,
```

```
            avg(price) over last2rows,
            avg(price) over last10rows
FROM SOURCE_SQL_STREAM_001
WINDOW
    last2rows AS (PARTITION BY ticker_symbol ROWS 2 PRECEDING),
    last10rows AS (PARTITION BY ticker_symbol ROWS 10 PRECEDING);
```

Para testar essa consulta em relação ao stream de demonstração, siga o procedimento de teste descrito no Exemplo 1.

Operações de dados de streaming: associações de streams

É possível ter vários streams no aplicativo em seu aplicativo. Você pode gravar consultas `JOIN` para correlacionar os dados que chegam nesses streams. Por exemplo, suponha que você tenha os seguintes fluxos no aplicativo:

- OrderStream— Recebe pedidos de estoque sendo feitos.

```
(orderId SqlType, ticker SqlType, amount SqlType, ROWTIME TimeStamp)
```

- TradeStream— Recebe as negociações de ações resultantes desses pedidos.

```
(tradeId SqlType, orderId SqlType, ticker SqlType, amount SqlType, ticker SqlType,
 amount SqlType, ROWTIME TimeStamp)
```

Estes são exemplos de consulta `JOIN` que correlacionam os dados desses streams.

Exemplo 1: relata pedidos com transações realizadas até um minuto após os pedidos serem feitos

Neste exemplo, a consulta associa o `OrderStream` e o `TradeStream`. No entanto, como queremos apenas as transações feitas um minuto após os pedidos, a consulta define a janela de 1 minuto pelo `TradeStream`. Para obter informações sobre consultas em janelas, veja Janelas deslizantes.

```
SELECT STREAM
     ROWTIME,
     o.orderId, o.ticker, o.amount AS orderAmount,
     t.amount AS tradeAmount
```

```
FROM OrderStream AS o
JOIN TradeStream OVER (RANGE INTERVAL '1' MINUTE PRECEDING) AS t
ON   o.orderId = t.orderId;
```

É possível definir as janelas explicitamente usando a cláusula `WINDOW` e gravando a consulta anterior da seguinte forma:

```
SELECT STREAM
    ROWTIME,
    o.orderId, o.ticker, o.amount AS orderAmount,
    t.amount AS tradeAmount
FROM OrderStream AS o
JOIN TradeStream OVER t
ON o.orderId = t.orderId
WINDOW t AS
    (RANGE INTERVAL '1' MINUTE PRECEDING)
```

Quando você incluir essa consulta no código do aplicativo, esse código será executado continuamente. O aplicativo emitirá uma saída para cada registro que chegar no `OrderStream`, se houver transações dentro da janela de 1 minuto após o pedido ser feito.

A associação na consulta anterior é uma associação interna, na qual a consulta emite registros no `OrderStream` com registros correspondentes no `TradeStream` (e vice-versa). Usando uma associação externa, crie outro cenário interessante. Suponha que você queira pedidos de ações para as quais não há transações em até um minuto da realização do pedido, e transações relatadas na mesma janela, mas para outros pedidos. Este é um exemplo de associação externa.

```
SELECT STREAM
    ROWTIME,
    o.orderId, o.ticker, o.amount AS orderAmount,
    t.ticker, t.tradeId, t.amount AS tradeAmount,
FROM OrderStream AS o
LEFT OUTER JOIN TradeStream OVER (RANGE INTERVAL '1' MINUTE PRECEDING) AS t
ON    o.orderId = t.orderId;
```

Exemplos de migração para o Managed Service for Apache Flink Studio

Os exemplos a seguir demonstram como migrar os aplicativos do Kinesis Data Analytics para SQL para o Managed Service for Apache Flink Studio.

Replicar as consultas do Kinesis Data Analytics para SQL no Managed Service para Apache Flink Studio

> **Warning**
>
> Para novos projetos, recomendamos que você use o Managed Service for Apache Flink Studio em vez do Kinesis Data Analytics para aplicativos SQL. O Managed Service for Apache Flink Studio combina facilidade de uso com recursos analíticos avançados, permitindo que você crie aplicativos sofisticados de processamento de stream em minutos.

Para migrar suas workloads para o Managed Service for Apache Flink Studio ou Managed Service for Apache Flink, esta seção fornece traduções de consultas que você pode usar para casos de uso comuns.

> **Note**
>
> O Managed Service for Apache Flink e o Managed Service for Apache Flink Studio oferecem atributos avançados de processamento de fluxo de dados não disponíveis nos aplicativos Kinesis Data Analytics baseados em SQL. Isso inclui semântica de processamento de uma única vez, janelas de horário de eventos, extensibilidade usando funções definidas pelo usuário e integrações personalizadas, suporte a idiomas imperativos, estado durável do aplicativo, escalabilidade horizontal, suporte para várias fontes de dados, integrações extensíveis e muito mais. Eles são essenciais para garantir a precisão, integridade, consistência e confiabilidade do processamento do fluxo de dados.

Antes de explorar esses exemplos, recomendamos que você primeiro leia Usando um caderno de notas Studio com um Managed Service for Apache Flink.

Tópicos

Recriar as consultas do Kinesis Data Analytics para SQL no Managed Service para Apache Flink Studio

A tabela a seguir fornece traduções de consultas comuns do aplicativo Kinesis Data Analytics baseado em SQL para o Managed Service for Apache Flink Studio.

Aplicativo em várias etapas

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "IN_APP_STREAM_001" (
   ingest_time TIMESTAMP,
   ticker_symbol VARCHAR(4),
   sector VARCHAR(16), price REAL, change REAL);
CREATE
OR REPLACE PUMP "STREAM_PUMP_001" AS
INSERT INTO
   "IN_APP_STREAM_001"
   SELECT
      STREAM APPROXIMATE_ARRIVAL_TIME,
      ticker_symbol,
      sector,
      price,
      change FROM "SOURCE_SQL_STREAM_001";
-- Second in-app stream and pump
CREATE
OR REPLACE STREAM "IN_APP_STREAM_02" (ingest_time TIMESTAMP,
   ticker_symbol VARCHAR(4),
   sector VARCHAR(16),
   price REAL,
   change REAL);
CREATE
OR REPLACE PUMP "STREAM_PUMP_02" AS
INSERT INTO
   "IN_APP_STREAM_02"
   SELECT
```

```
      STREAM ingest_time,
      ticker_symbol,
      sector,
      price,
      change FROM "IN_APP_STREAM_001";
-- Destination in-app stream and third pump
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (ingest_time TIMESTAMP,
   ticker_symbol VARCHAR(4),
   sector VARCHAR(16),
   price REAL,
   change REAL);
CREATE
OR REPLACE PUMP "STREAM_PUMP_03" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM ingest_time,
      ticker_symbol,
      sector,
      price,
      change FROM "IN_APP_STREAM_02";
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql DROP TABLE IF EXISTS SOURCE_SQL_STREAM_001;

CREATE TABLE SOURCE_SQL_STREAM_001 (TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(16),
   PRICE DOUBLE,
   CHANGE DOUBLE,
   APPROXIMATE_ARRIVAL_TIME TIMESTAMP(3) METADATA

FROM
   'timestamp' VIRTUAL,
   WATERMARK FOR APPROXIMATE_ARRIVAL_TIME AS APPROXIMATE_ARRIVAL_TIME - INTERVAL '1'
 SECOND )
   PARTITIONED BY (TICKER_SYMBOL) WITH (
      'connector' = 'kinesis',
      'stream' = 'kinesis-analytics-demo-stream',
      'aws.region' = 'us-east-1',
      'scan.stream.initpos' = 'LATEST',
      'format' = 'json',
```

```
      'json.timestamp-format.standard' = 'ISO-8601');
DROP TABLE IF EXISTS IN_APP_STREAM_001;

CREATE TABLE IN_APP_STREAM_001 (
   INGEST_TIME TIMESTAMP,
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(16),
   PRICE DOUBLE,
   CHANGE DOUBLE )
PARTITIONED BY (TICKER_SYMBOL) WITH (
      'connector' = 'kinesis',
      'stream' = 'IN_APP_STREAM_001',
      'aws.region' = 'us-east-1',
      'scan.stream.initpos' = 'LATEST',
      'format' = 'json',
      'json.timestamp-format.standard' = 'ISO-8601');

DROP TABLE IF EXISTS IN_APP_STREAM_02;

CREATE TABLE IN_APP_STREAM_02 (
   INGEST_TIME TIMESTAMP,
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(16),
   PRICE DOUBLE,
   CHANGE DOUBLE )
PARTITIONED BY (TICKER_SYMBOL) WITH (
   'connector' = 'kinesis',
   'stream' = 'IN_APP_STREAM_02',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');

DROP TABLE IF EXISTS DESTINATION_SQL_STREAM;

CREATE TABLE DESTINATION_SQL_STREAM (
   INGEST_TIME TIMESTAMP, TICKER_SYMBOL VARCHAR(4), SECTOR VARCHAR(16),
   PRICE DOUBLE, CHANGE DOUBLE )
PARTITIONED BY (TICKER_SYMBOL) WITH (
   'connector' = 'kinesis',
   'stream' = 'DESTINATION_SQL_STREAM',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
```

```
   'json.timestamp-format.standard' = 'ISO-8601');


Query 2 - % flink.ssql(type =
update
)
   INSERT INTO
      IN_APP_STREAM_001
      SELECT
         APPROXIMATE_ARRIVAL_TIME AS INGEST_TIME,
         TICKER_SYMBOL,
         SECTOR,
         PRICE,
         CHANGE
      FROM
         SOURCE_SQL_STREAM_001;


Query 3 - % flink.ssql(type =
update
)
   INSERT INTO
      IN_APP_STREAM_02
      SELECT
         INGEST_TIME,
         TICKER_SYMBOL,
         SECTOR,
         PRICE,
         CHANGE
      FROM
         IN_APP_STREAM_001;


Query 4 - % flink.ssql(type =
update
)
   INSERT INTO
      DESTINATION_SQL_STREAM
      SELECT
         INGEST_TIME,
         TICKER_SYMBOL,
         SECTOR,
         PRICE,
         CHANGE
```

```
      FROM
         IN_APP_STREAM_02;
```

Transformação de valores DateTime

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
   TICKER VARCHAR(4),
   event_time TIMESTAMP,
   five_minutes_before TIMESTAMP,
   event_unix_timestamp BIGINT,
   event_timestamp_as_char VARCHAR(50),
   event_second INTEGER);

CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM TICKER,
      EVENT_TIME,
      EVENT_TIME - INTERVAL '5' MINUTE,
      UNIX_TIMESTAMP(EVENT_TIME),
      TIMESTAMP_TO_CHAR('yyyy-MM-dd hh:mm:ss', EVENT_TIME),
      EXTRACT(SECOND
   FROM
      EVENT_TIME)
   FROM
      "SOURCE_SQL_STREAM_001"
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER VARCHAR(4),
   EVENT_TIME TIMESTAMP(3),
   FIVE_MINUTES_BEFORE TIMESTAMP(3),
   EVENT_UNIX_TIMESTAMP INT,
   EVENT_TIMESTAMP_AS_CHAR VARCHAR(50),
```

```
   EVENT_SECOND INT)

PARTITIONED BY (TICKER) WITH (
   'connector' = 'kinesis', 'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601')

Query 2 - % flink.ssql(type =
   update
)
      SELECT
         TICKER,
         EVENT_TIME,
         EVENT_TIME - INTERVAL '5' MINUTE AS FIVE_MINUTES_BEFORE,
         UNIX_TIMESTAMP() AS EVENT_UNIX_TIMESTAMP,
         DATE_FORMAT(EVENT_TIME, 'yyyy-MM-dd hh:mm:ss') AS EVENT_TIMESTAMP_AS_CHAR,
         EXTRACT(SECOND
      FROM
         EVENT_TIME) AS EVENT_SECOND
      FROM
         DESTINATION_SQL_STREAM;
```

Alertas simples

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM"(
   ticker_symbol VARCHAR(4),
   sector VARCHAR(12),
   change DOUBLE,
   price DOUBLE);

CREATE
OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
SELECT
   STREAM ticker_symbol,
   sector,
   change,
   price
```

```
FROM
   "SOURCE_SQL_STREAM_001"
WHERE
   (
      ABS(Change / (Price - Change)) * 100
   )
   > 1
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) DROP TABLE IF EXISTS DESTINATION_SQL_STREAM;

CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(4),
   CHANGE DOUBLE,
   PRICE DOUBLE )
PARTITIONED BY (TICKER_SYMBOL) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');

Query 2 - % flink.ssql(type =
update
)
   SELECT
      TICKER_SYMBOL,
      SECTOR,
      CHANGE,
      PRICE
   FROM
      DESTINATION_SQL_STREAM
   WHERE
      (
         ABS(CHANGE / (PRICE - CHANGE)) * 100
      )
      > 1;
```

Alertas controlados

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "CHANGE_STREAM"(
   ticker_symbol VARCHAR(4),
   sector VARCHAR(12),
   change DOUBLE,
   price DOUBLE);

CREATE
OR REPLACE PUMP "change_pump" AS INSERT INTO "CHANGE_STREAM"
SELECT
   STREAM ticker_symbol,
   sector,
   change,
   price
FROM "SOURCE_SQL_STREAM_001"
WHERE
   (
      ABS(Change / (Price - Change)) * 100
   )
   > 1;
-- ** Trigger Count and Limit **
-- Counts "triggers" or those values that evaluated true against the previous where
 clause
-- Then provides its own limit on the number of triggers per hour per ticker symbol
 to what is specified in the WHERE clause

CREATE
OR REPLACE STREAM TRIGGER_COUNT_STREAM (
   ticker_symbol VARCHAR(4),
   change REAL,
   trigger_count INTEGER);

CREATE
OR REPLACE PUMP trigger_count_pump AS
INSERT INTO
   TRIGGER_COUNT_STREAMSELECT STREAM ticker_symbol,
   change,
   trigger_count
FROM
   (
```

```
      SELECT
         STREAM ticker_symbol,
         change,
         COUNT(*) OVER W1 as trigger_countFROM "CHANGE_STREAM" --window to perform
 aggregations over last minute to keep track of triggers
         WINDOW W1 AS
         (
            PARTITION BY ticker_symbol RANGE INTERVAL '1' MINUTE PRECEDING
         )
   )
WHERE
   trigger_count >= 1;
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) DROP TABLE IF EXISTS DESTINATION_SQL_STREAM;

CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(4),
   CHANGE DOUBLE, PRICE DOUBLE,
   EVENT_TIME AS PROCTIME())
PARTITIONED BY (TICKER_SYMBOL)
WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');
DROP TABLE IF EXISTS TRIGGER_COUNT_STREAM;
CREATE TABLE TRIGGER_COUNT_STREAM (
   TICKER_SYMBOL VARCHAR(4),
   CHANGE DOUBLE,
   TRIGGER_COUNT INT)
PARTITIONED BY (TICKER_SYMBOL);

Query 2 - % flink.ssql(type =
update
)
   SELECT
```

```
      TICKER_SYMBOL,
      SECTOR,
      CHANGE,
      PRICE
   FROM
      DESTINATION_SQL_STREAM
   WHERE
      (
         ABS(CHANGE / (PRICE - CHANGE)) * 100
      )
      > 1;

Query 3 - % flink.ssql(type =
update
)
   SELECT *
   FROM(
         SELECT
            TICKER_SYMBOL,
            CHANGE,
            COUNT(*) AS TRIGGER_COUNT
         FROM
            DESTINATION_SQL_STREAM
         GROUP BY
            TUMBLE(EVENT_TIME, INTERVAL '1' MINUTE),
            TICKER_SYMBOL,
            CHANGE
      )
   WHERE
      TRIGGER_COUNT > 1;
```

Agregação dos resultados parciais de uma consulta

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "CALC_COUNT_SQL_STREAM"(
   TICKER VARCHAR(4),
   TRADETIME TIMESTAMP,
   TICKERCOUNT DOUBLE);

CREATE
```

```
OR REPLACE STREAM "DESTINATION_SQL_STREAM"(
   TICKER VARCHAR(4),
   TRADETIME TIMESTAMP,
   TICKERCOUNT DOUBLE);

CREATE PUMP "CALC_COUNT_SQL_PUMP_001" AS
INSERT INTO
   "CALC_COUNT_SQL_STREAM"(
   "TICKER",
   "TRADETIME",
   "TICKERCOUNT")
   SELECT
      STREAM "TICKER_SYMBOL",
      STEP("SOURCE_SQL_STREAM_001",
      "ROWTIME" BY INTERVAL '1' MINUTE) as "TradeTime",
      COUNT(*) AS "TickerCount "
   FROM
      "SOURCE_SQL_STREAM_001"
   GROUP BY
      STEP("SOURCE_SQL_STREAM_001". ROWTIME BY INTERVAL '1' MINUTE),
      STEP("SOURCE_SQL_STREAM_001"." APPROXIMATE_ARRIVAL_TIME" BY INTERVAL '1'
 MINUTE),
      TICKER_SYMBOL;
CREATE PUMP "AGGREGATED_SQL_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM" (
   "TICKER",
   "TRADETIME",
   "TICKERCOUNT")
   SELECT
      STREAM "TICKER",
      "TRADETIME",
      SUM("TICKERCOUNT") OVER W1 AS "TICKERCOUNT"
   FROM
      "CALC_COUNT_SQL_STREAM" WINDOW W1 AS
      (
         PARTITION BY "TRADETIME" RANGE INTERVAL '10' MINUTE PRECEDING
      )
;
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
```

```
update
) DROP TABLE IF EXISTS SOURCE_SQL_STREAM_001;
CREATE TABLE SOURCE_SQL_STREAM_001 (
   TICKER_SYMBOL VARCHAR(4),
   TRADETIME AS PROCTIME(),
   APPROXIMATE_ARRIVAL_TIME TIMESTAMP(3) METADATA
FROM
   'timestamp' VIRTUAL,
   WATERMARK FOR APPROXIMATE_ARRIVAL_TIME AS APPROXIMATE_ARRIVAL_TIME - INTERVAL '1'
 SECOND)
PARTITIONED BY (TICKER_SYMBOL) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');
DROP TABLE IF EXISTS CALC_COUNT_SQL_STREAM;
CREATE TABLE CALC_COUNT_SQL_STREAM (
   TICKER VARCHAR(4),
   TRADETIME TIMESTAMP(3),
   WATERMARK FOR TRADETIME AS TRADETIME - INTERVAL '1' SECOND,
   TICKERCOUNT BIGINT NOT NULL ) PARTITIONED BY (TICKER) WITH (
      'connector' = 'kinesis',
      'stream' = 'CALC_COUNT_SQL_STREAM',
      'aws.region' = 'us-east-1',
      'scan.stream.initpos' = 'LATEST',
      'format' = 'csv');
DROP TABLE IF EXISTS DESTINATION_SQL_STREAM;
CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER VARCHAR(4),
   TRADETIME TIMESTAMP(3),
   WATERMARK FOR TRADETIME AS TRADETIME - INTERVAL '1' SECOND,
   TICKERCOUNT BIGINT NOT NULL )
   PARTITIONED BY (TICKER) WITH ('connector' = 'kinesis',
      'stream' = 'DESTINATION_SQL_STREAM',
      'aws.region' = 'us-east-1',
      'scan.stream.initpos' = 'LATEST',
      'format' = 'csv');

Query 2 - % flink.ssql(type =
update
)
   INSERT INTO
```

```
      CALC_COUNT_SQL_STREAM
      SELECT
         TICKER,
         TO_TIMESTAMP(TRADETIME, 'yyyy-MM-dd HH:mm:ss') AS TRADETIME,
         TICKERCOUNT
      FROM
         (
            SELECT
               TICKER_SYMBOL AS TICKER,
               DATE_FORMAT(TRADETIME, 'yyyy-MM-dd HH:mm:00') AS TRADETIME,
               COUNT(*) AS TICKERCOUNT
            FROM
               SOURCE_SQL_STREAM_001
            GROUP BY
               TUMBLE(TRADETIME, INTERVAL '1' MINUTE),
               DATE_FORMAT(TRADETIME, 'yyyy-MM-dd HH:mm:00'),
               DATE_FORMAT(APPROXIMATE_ARRIVAL_TIME, 'yyyy-MM-dd HH:mm:00'),
               TICKER_SYMBOL
         )
;

Query 3 - % flink.ssql(type =
update
)
   SELECT
      *
   FROM
      CALC_COUNT_SQL_STREAM;

Query 4 - % flink.ssql(type =
update
)
   INSERT INTO
      DESTINATION_SQL_STREAM
      SELECT
         TICKER,
         TRADETIME,
         SUM(TICKERCOUNT) OVER W1 AS TICKERCOUNT
      FROM
         CALC_COUNT_SQL_STREAM WINDOW W1 AS
         (
            PARTITION BY TICKER
         ORDER BY
            TRADETIME RANGE INTERVAL '10' MINUTE PRECEDING
```

```
         )
;

Query 5 - % flink.ssql(type =
update
)
   SELECT
      *
   FROM
      DESTINATION_SQL_STREAM;
```

Transformação de valores de string

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM for cleaned up referrerCREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" ( "ingest_time" TIMESTAMP, "referrer"
 VARCHAR(32));
CREATE
OR REPLACE PUMP "myPUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
SELECT
   STREAM "APPROXIMATE_ARRIVAL_TIME",
   SUBSTRING("referrer", 12,
   (
      POSITION('.com' IN "referrer") - POSITION('www.' IN "referrer") - 4
   )
)
FROM
   "SOURCE_SQL_STREAM_001";
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   referrer VARCHAR(32),
   ingest_time AS PROCTIME() ) PARTITIONED BY (referrer)
WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
```

```
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601')

Query 2 - % flink.ssql(type =
   update
)
      SELECT
         ingest_time,
         substring(referrer, 12, 6) as referrer
      FROM
         DESTINATION_SQL_STREAM;
```

Substituição de uma substring usando regex

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM for cleaned up referrerCREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" ( "ingest_time" TIMESTAMP, "referrer"
 VARCHAR(32));
CREATE
OR REPLACE PUMP "myPUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
SELECT
   STREAM "APPROXIMATE_ARRIVAL_TIME",
   REGEX_REPLACE("REFERRER", 'http://', 'https://', 1, 0)
FROM
   "SOURCE_SQL_STREAM_001";
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   referrer VARCHAR(32),
   ingest_time AS PROCTIME())
PARTITIONED BY (referrer) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
```

```
   'json.timestamp-format.standard' = 'ISO-8601')

Query 2 - % flink.ssql(type =
   update
)
      SELECT
         ingest_time,
         REGEXP_REPLACE(referrer, 'http', 'https') as referrer
      FROM
         DESTINATION_SQL_STREAM;
```

Análise de log Regex

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM"(
   sector VARCHAR(24),
   match1 VARCHAR(24),
   match2 VARCHAR(24));
CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM T.SECTOR,
      T.REC.COLUMN1,
      T.REC.COLUMN2
   FROM
      (
         SELECT
            STREAM SECTOR,
            REGEX_LOG_PARSE(SECTOR, '.*([E].).*([R].*)') AS REC
         FROM
            SOURCE_SQL_STREAM_001
      )
      AS T;
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
```

```
) CREATE TABLE DESTINATION_SQL_STREAM (
   CHANGE DOUBLE, PRICE DOUBLE,
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(16))
PARTITIONED BY (SECTOR) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601')

Query 2 - % flink.ssql(type =
   update
)
SELECT
   *
FROM
   (
      SELECT
         SECTOR,
         REGEXP_EXTRACT(SECTOR, '.([E].).([R].)', 1) AS MATCH1,
         REGEXP_EXTRACT(SECTOR, '.([E].).([R].)', 2) AS MATCH2
      FROM
         DESTINATION_SQL_STREAM
   )
WHERE
   MATCH1 IS NOT NULL
   AND MATCH2 IS NOT NULL;
```

Transformação de valores DateTime

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
   TICKER VARCHAR(4),
   event_time TIMESTAMP,
   five_minutes_before TIMESTAMP,
   event_unix_timestamp BIGINT,
   event_timestamp_as_char VARCHAR(50),
   event_second INTEGER);
```

```
CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM TICKER,
      EVENT_TIME,
      EVENT_TIME - INTERVAL '5' MINUTE,
      UNIX_TIMESTAMP(EVENT_TIME),
      TIMESTAMP_TO_CHAR('yyyy-MM-dd hh:mm:ss', EVENT_TIME),
      EXTRACT(SECOND
   FROM
      EVENT_TIME)
   FROM
      "SOURCE_SQL_STREAM_001"
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER VARCHAR(4),
   EVENT_TIME TIMESTAMP(3),
   FIVE_MINUTES_BEFORE TIMESTAMP(3),
   EVENT_UNIX_TIMESTAMP INT,
   EVENT_TIMESTAMP_AS_CHAR VARCHAR(50),
   EVENT_SECOND INT) PARTITIONED BY (TICKER)
WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601')

Query 2 - % flink.ssql(type =
   update
)
      SELECT
         TICKER,
         EVENT_TIME,
         EVENT_TIME - INTERVAL '5' MINUTE AS FIVE_MINUTES_BEFORE,
         UNIX_TIMESTAMP() AS EVENT_UNIX_TIMESTAMP,
```

```
         DATE_FORMAT(EVENT_TIME, 'yyyy-MM-dd hh:mm:ss') AS EVENT_TIMESTAMP_AS_CHAR,
         EXTRACT(SECOND
      FROM
         EVENT_TIME) AS EVENT_SECOND
      FROM
         DESTINATION_SQL_STREAM;
```

Janelas e agregação

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
   event_time TIMESTAMP,
   ticker_symbol VARCHAR(4),
   ticker_count INTEGER);
CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM EVENT_TIME,
      TICKER,
      COUNT(TICKER) AS ticker_count
   FROM
      "SOURCE_SQL_STREAM_001" WINDOWED BY STAGGER ( PARTITION BY
         TICKER,
         EVENT_TIME RANGE INTERVAL '1' MINUTE);
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   EVENT_TIME TIMESTAMP(3),
   WATERMARK FOR EVENT_TIME AS EVENT_TIME - INTERVAL '60' SECOND,
   TICKER VARCHAR(4),
   TICKER_COUNT INT) PARTITIONED BY (TICKER)
WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
```

```
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json'

Query 2 - % flink.ssql(type =
   update
)
      SELECT
         EVENT_TIME,
         TICKER, COUNT(TICKER) AS ticker_count
      FROM
         DESTINATION_SQL_STREAM
      GROUP BY
         TUMBLE(EVENT_TIME,
         INTERVAL '60' second),
         EVENT_TIME, TICKER;
```

Janela em cascata usando ROWTIME

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM"(
   TICKER VARCHAR(4),
   MIN_PRICE REAL,
   MAX_PRICE REAL);
CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM TICKER,
      MIN(PRICE),
      MAX(PRICE)
   FROM
      "SOURCE_SQL_STREAM_001"
   GROUP BY
      TICKER,
      STEP("SOURCE_SQL_STREAM_001".
            ROWTIME BY INTERVAL '60' SECOND);
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   ticker VARCHAR(4),
   price DOUBLE,
   event_time VARCHAR(32),
   processing_time AS PROCTIME())
PARTITIONED BY (ticker) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601')

Query 2 - % flink.ssql(type =
   update
)
      SELECT
         ticker,
         min(price) AS MIN_PRICE,
         max(price) AS MAX_PRICE
      FROM
         DESTINATION_SQL_STREAM
      GROUP BY
         TUMBLE(processing_time, INTERVAL '60' second),
         ticker;
```

Recuperação dos valores que ocorrem com mais frequência (TOP_K_ITEMS_TUMBLING)

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "CALC_COUNT_SQL_STREAM"(TICKER VARCHAR(4),
   TRADETIME TIMESTAMP,
   TICKERCOUNT DOUBLE);
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM"(
   TICKER VARCHAR(4),
   TRADETIME TIMESTAMP,
```

```
   TICKERCOUNT DOUBLE);
CREATE PUMP "CALC_COUNT_SQL_PUMP_001" AS INSERT INTO "CALC_COUNT_SQL_STREAM" (
   "TICKER",
   "TRADETIME",
   "TICKERCOUNT")
SELECT
   STREAM"TICKER_SYMBOL",
   STEP("SOURCE_SQL_STREAM_001"."ROWTIME" BY INTERVAL '1' MINUTE) as "TradeTime",
   COUNT(*) AS "TickerCount"
FROM
   "SOURCE_SQL_STREAM_001"
GROUP BY STEP("SOURCE_SQL_STREAM_001".
   ROWTIME BY INTERVAL '1' MINUTE),
   STEP("SOURCE_SQL_STREAM_001".
      "APPROXIMATE_ARRIVAL_TIME" BY INTERVAL '1' MINUTE),
   TICKER_SYMBOL;
CREATE PUMP "AGGREGATED_SQL_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM" (
   "TICKER",
   "TRADETIME",
   "TICKERCOUNT")
SELECT
   STREAM "TICKER",
   "TRADETIME",
   SUM("TICKERCOUNT") OVER W1 AS "TICKERCOUNT"
FROM
   "CALC_COUNT_SQL_STREAM" WINDOW W1 AS
   (
      PARTITION BY "TRADETIME" RANGE INTERVAL '10' MINUTE PRECEDING
   )
;
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) DROP TABLE IF EXISTS DESTINATION_SQL_STREAM;
CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER VARCHAR(4),
   EVENT_TIME TIMESTAMP(3),
   WATERMARK FOR EVENT_TIME AS EVENT_TIME - INTERVAL '1' SECONDS )
PARTITIONED BY (TICKER) WITH (
   'connector' = 'kinesis', 'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
```

```
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');

Query 2 - % flink.ssql(type =
update
)
   SELECT
      *
   FROM
      (
         SELECT
            TICKER,
            COUNT(*) as MOST_FREQUENT_VALUES,
            ROW_NUMBER() OVER (PARTITION BY TICKER
         ORDER BY
            TICKER DESC) AS row_num
         FROM
            DESTINATION_SQL_STREAM
         GROUP BY
            TUMBLE(EVENT_TIME, INTERVAL '1' MINUTE),
            TICKER
      )
   WHERE
      row_num <= 5;
```

Itens Top-K aproximados

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (ITEM VARCHAR(1024), ITEM_COUNT DOUBLE);
CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM ITEM,
      ITEM_COUNT
   FROM
      TABLE(TOP_K_ITEMS_TUMBLING(CURSOR(
      SELECT
```

```
         STREAM *
      FROM
         "SOURCE_SQL_STREAM_001"), 'column1', -- name of column in single quotes10,
 -- number of top items60 -- tumbling window size in seconds));
```

Managed Service for Apache Flink Studio

```
%flinkssql
DROP TABLE IF EXISTS SOURCE_SQL_STREAM_001
CREATE TABLE SOURCE_SQL_STREAM_001 ( TS TIMESTAMP(3), WATERMARK FOR TS as TS -
 INTERVAL '5' SECOND, ITEM VARCHAR(1024),
PRICE DOUBLE)
   WITH ( 'connector' = 'kinesis', 'stream' = 'SOURCE_SQL_STREAM_001',
'aws.region' = 'us-east-1', 'scan.stream.initpos' = 'LATEST', 'format' = 'json',
'json.timestamp-format.standard' = 'ISO-8601');


%flink.ssql(type=update)
SELECT
   *
FROM
   (
      SELECT
         *,
         ROW_NUMBER() OVER (PARTITION BY AGG_WINDOW
      ORDER BY
         ITEM_COUNT DESC) as rownum
      FROM
         (
            select
               AGG_WINDOW,
               ITEM,
               ITEM_COUNT
            from
               (
                  select
                     TUMBLE_ROWTIME(TS, INTERVAL '60' SECONDS) as AGG_WINDOW,
                     ITEM,
                     count(*) as ITEM_COUNT
                  FROM
                     SOURCE_SQL_STREAM_001
                  GROUP BY
```

```
                    TUMBLE(TS, INTERVAL '60' SECONDS),
                    ITEM
                )
        )
    )
where
    rownum <= 3
```

Análise de logs da web (função W3C_LOG_PARSE)

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" ( column1 VARCHAR(16),
    column2 VARCHAR(16),
    column3 VARCHAR(16),
    column4 VARCHAR(16),
    column5 VARCHAR(16),
    column6 VARCHAR(16),
    column7 VARCHAR(16));
CREATE
OR REPLACE PUMP "myPUMP" ASINSERT INTO "DESTINATION_SQL_STREAM"
SELECT
    STREAM l.r.COLUMN1,
    l.r.COLUMN2,
    l.r.COLUMN3,
    l.r.COLUMN4,
    l.r.COLUMN5,
    l.r.COLUMN6,
    l.r.COLUMN7
FROM
    (
        SELECT
            STREAM W3C_LOG_PARSE("log", 'COMMON')
        FROM
            "SOURCE_SQL_STREAM_001"
    )
    AS l(r);
```

Managed Service for Apache Flink Studio

```
%flink.ssql(type=update)
```

```
DROP TABLE IF EXISTS SOURCE_SQL_STREAM_001 CREATE TABLE SOURCE_SQL_STREAM_001 ( log
 VARCHAR(1024))
   WITH ( 'connector' = 'kinesis',
          'stream' = 'SOURCE_SQL_STREAM_001',
          'aws.region' = 'us-east-1',
          'scan.stream.initpos' = 'LATEST',
          'format' = 'json',
          'json.timestamp-format.standard' = 'ISO-8601');

% flink.ssql(type=update)
   select
      SPLIT_INDEX(log, ' ', 0),
      SPLIT_INDEX(log, ' ', 1),
      SPLIT_INDEX(log, ' ', 2),
      SPLIT_INDEX(log, ' ', 3),
      SPLIT_INDEX(log, ' ', 4),
      SPLIT_INDEX(log, ' ', 5),
      SPLIT_INDEX(log, ' ', 6)
   from
      SOURCE_SQL_STREAM_001;
```

Divisão de strings de caracteres em vários campos (função VARIABLE_COLUMN_LOG_PARSE)

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM"( "column_A" VARCHAR(16),
   "column_B" VARCHAR(16),
   "column_C" VARCHAR(16),
   "COL_1" VARCHAR(16),
   "COL_2" VARCHAR(16),
   "COL_3" VARCHAR(16));
CREATE
OR REPLACE PUMP "SECOND_STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
SELECT
   STREAM t."Col_A",
   t."Col_B",
   t."Col_C",
   t.r."COL_1",
   t.r."COL_2",
   t.r."COL_3"
FROM
```

```
   (
      SELECT
         STREAM "Col_A",
         "Col_B",
         "Col_C",
         VARIABLE_COLUMN_LOG_PARSE ("Col_E_Unstructured",
         'COL_1 TYPE VARCHAR(16),
         COL_2 TYPE VARCHAR(16),
         COL_3 TYPE VARCHAR(16)', ',') AS r
      FROM
         "SOURCE_SQL_STREAM_001"
   )
   as t;
```

Managed Service for Apache Flink Studio

```
%flink.ssql(type=update)
DROP TABLE IF EXISTS SOURCE_SQL_STREAM_001 CREATE TABLE SOURCE_SQL_STREAM_001 ( log
 VARCHAR(1024))
   WITH ( 'connector' = 'kinesis',
          'stream' = 'SOURCE_SQL_STREAM_001',
          'aws.region' = 'us-east-1',
          'scan.stream.initpos' = 'LATEST',
          'format' = 'json',
          'json.timestamp-format.standard' = 'ISO-8601');

% flink.ssql(type=update)
   select
      SPLIT_INDEX(log, ' ', 0),
      SPLIT_INDEX(log, ' ', 1),
      SPLIT_INDEX(log, ' ', 2),
      SPLIT_INDEX(log, ' ', 3),
      SPLIT_INDEX(log, ' ', 4),
      SPLIT_INDEX(log, ' ', 5)
)
from
   SOURCE_SQL_STREAM_001;
```

Junções

SQL-based Kinesis Data Analytics application

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
   ticker_symbol VARCHAR(4),
   "Company" varchar(20),
   sector VARCHAR(12),
   change DOUBLE,
   price DOUBLE);
CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM ticker_symbol,
      "c"."Company",
      sector,
      change,
      priceFROM "SOURCE_SQL_STREAM_001"
      LEFT JOIN
         "CompanyName" as "c"
         ON "SOURCE_SQL_STREAM_001".ticker_symbol = "c"."Ticker";
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) CREATE TABLE DESTINATION_SQL_STREAM (
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(12),
   CHANGE INT,
   PRICE DOUBLE )
PARTITIONED BY (TICKER_SYMBOL) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');

Query 2 - CREATE TABLE CompanyName (
```

```
   Ticker VARCHAR(4),
   Company VARCHAR(4)) WITH (
      'connector' = 'filesystem',
      'path' = 's3://kda-demo-sample/TickerReference.csv',
      'format' = 'csv' );

Query 3 - % flink.ssql(type =
update
)
   SELECT
      TICKER_SYMBOL,
      c.Company,
      SECTOR,
      CHANGE,
      PRICE
   FROM
      DESTINATION_SQL_STREAM
      LEFT JOIN
         CompanyName as c
         ON DESTINATION_SQL_STREAM.TICKER_SYMBOL = c.Ticker;
```

Erros

SQL-based Kinesis Data Analytics application

```
SELECT
   STREAM ticker_symbol,
   sector,
   change,
   (
      price / 0
   )
   as ProblemColumnFROM "SOURCE_SQL_STREAM_001"
WHERE
   sector SIMILAR TO '%TECH%';
```

Managed Service for Apache Flink Studio

```
Query 1 - % flink.ssql(type =
update
) DROP TABLE IF EXISTS DESTINATION_SQL_STREAM;
CREATE TABLE DESTINATION_SQL_STREAM (
```

```
   TICKER_SYMBOL VARCHAR(4),
   SECTOR VARCHAR(16),
   CHANGE DOUBLE,
   PRICE DOUBLE )
PARTITIONED BY (TICKER_SYMBOL) WITH (
   'connector' = 'kinesis',
   'stream' = 'kinesis-analytics-demo-stream',
   'aws.region' = 'us-east-1',
   'scan.stream.initpos' = 'LATEST',
   'format' = 'json',
   'json.timestamp-format.standard' = 'ISO-8601');

Query 2 - % flink.pyflink @udf(input_types = [DataTypes.BIGINT()],
   result_type = DataTypes.BIGINT()) def DivideByZero(price): try: price / 0
except
: return - 1 st_env.register_function("DivideByZero",
   DivideByZero)

   Query 3 - % flink.ssql(type =
update
)
   SELECT
      CURRENT_TIMESTAMP AS ERROR_TIME,
      *
   FROM
      (
         SELECT
            TICKER_SYMBOL,
            SECTOR,
            CHANGE,
            DivideByZero(PRICE) as ErrorColumn
         FROM
            DESTINATION_SQL_STREAM
         WHERE
            SECTOR SIMILAR TO '%TECH%'
      )
      AS ERROR_STREAM;
```

Migração de workloads do Random Cut Forest

Se você deseja mover workloads que usam Random Cut Forest do Kinesis Analytics para SQL para o Managed Service for Apache Flink, esta AWSpostagem do blog demonstra como usar o Managed Service for Apache Flink para executar um algoritmo RCF on-line para detecção de anomalias.

Substituição do Kinesis Data Firehose como fonte pelo Kinesis Data Streams

Veja Converting-KDASQL-KDAStudio/ para obter um tutorial completo.

No exercício a seguir, você alterará seu fluxo de dados para usar o Amazon Managed Service for Apache Flink. Isso também significa mudar do Amazon Kinesis Data Firehose para o Amazon Kinesis Data Streams.

Primeiro, compartilhamos uma arquitetura típica do KDA-SQL, antes de mostrar como você pode substituí-la usando o Amazon Managed Service for Apache Flink e o Amazon Kinesis Data Streams. Como alternativa, você pode iniciar o AWS CloudFormation modelo aqui:

Amazon Kinesis Data Analytics-SQL e Amazon Kinesis Data Firehose

Aqui está o fluxo arquitetônico SQL do Amazon Kinesis Data Analytics:

Primeiro, examinamos a configuração de um Amazon Kinesis Data Analytics-SQL e do Amazon Kinesis Data Firehose legados. O caso de uso é um mercado de negociação em que os dados de negociação, incluindo cotação e preço das ações, são transmitidos de fontes externas para os sistemas Amazon Kinesis. O Amazon Kinesis Data Analytics para SQL usa o stream de entrada para executar consultas em janelas, como a janela em cascada para determinar o volume e o preço da negociação `min`, `max` e `average` em uma janela de um minuto para cada cotador de ações.

O Amazon Kinesis Data Analytics-SQL está configurado para ingerir dados da API do Amazon Kinesis Data Firehose. Após o processamento, o Amazon Kinesis Data Analytics-SQL envia os dados processados para outro Amazon Kinesis Data Firehose, que então salva a saída em um bucket do Amazon S3.

Nesse caso, você usa o Amazon Kinesis Data Generator. O Amazon Kinesis Data Generator permite que você envie dados de teste para seus streams de entrega do Amazon Kinesis Data Streams ou do Amazon Kinesis Data Firehose. Para começar, siga as instruções aqui. Use o AWS CloudFormation modelo aqui no lugar do fornecido nas instruções:

Depois de executar o modelo AWS CloudFormation, a seção de saída fornecerá a URL do Amazon Kinesis Data Generator. Faça login no portal usando a ID de usuário e a senha do Cognito que você configurou aqui. Selecione a região e o nome do stream de destino. Para o estado atual, escolha os streams de entrega do Amazon Kinesis Data Firehose. Para o novo estado, escolha nome dos streams do Amazon Kinesis Data Firehose. Você pode criar vários modelos, dependendo dos seus requisitos, e testar o modelo usando o botão Testar modelo antes de enviá-lo para o stream de destino.

Veja a seguir um exemplo de carga útil usando o Amazon Kinesis Data Generator. O gerador de dados tem como alvo a entrada dos streams do Amazon Kinesis Firehose para transmitir os dados continuamente. O cliente SDK do Amazon Kinesis também pode enviar dados de outros produtores.

```
2023-02-17 09:28:07.763,"AAPL",5032023-02-17 09:28:07.763,
"AMZN",3352023-02-17 09:28:07.763,
"GOOGL",1852023-02-17 09:28:07.763,
"AAPL",11162023-02-17 09:28:07.763,
"GOOGL",1582
```

O JSON a seguir é usado para gerar uma série aleatória de data e hora da negociação, código de negociação da ação e preço da ação:

```
date.now(YYYY-MM-DD HH:mm:ss.SSS),
"random.arrayElement(["AAPL","AMZN","MSFT","META","GOOGL"])",
random.number(2000)
```

Depois de escolher Enviar dados, o gerador começará a enviar dados simulados.

Os sistemas externos transmitem os dados para o Amazon Kinesis Data Firehose. Usando o Amazon Kinesis Data Analytics para aplicativos SQL, você pode analisar dados de transmissão usando o SQL padrão. O serviço permite que você crie e execute código SQL em fontes de streaming para realizar análises de séries temporais, alimentar painéis em tempo real e criar métricas em tempo real. O Amazon Kinesis Data Analytics para aplicativos SQL pode criar um stream de destino a partir de consultas SQL no stream de entrada e enviar o stream de destino para

outro Amazon Kinesis Data Firehose. O Amazon Kinesis Data Firehose de destino pode enviar os dados analíticos para o Amazon S3 como estado final.

O código legado do Amazon Kinesis Data Analytics-SQL é baseado em uma extensão do padrão SQL.

Você usa a consulta a seguir no Amazon Kinesis Data Analytics-SQL. Primeiro, você cria um stream de destino para a saída da consulta. Então, você usa PUMP, que é um objeto de repositório do Amazon Kinesis Data Analytics (uma extensão do padrão SQL) que fornece uma função de consulta de execução contínua `INSERT INTO stream SELECT ... FROM`, permitindo assim que os resultados de uma consulta sejam inseridos continuamente em um fluxo nomeado.

```
CREATE
OR REPLACE STREAM "DESTINATION_SQL_STREAM" (EVENT_TIME TIMESTAMP,
INGEST_TIME TIMESTAMP,
TICKER VARCHAR(16),
VOLUME BIGINT,
AVG_PRICE DOUBLE,
MIN_PRICE DOUBLE,
MAX_PRICE DOUBLE);

CREATE
OR REPLACE PUMP "STREAM_PUMP" AS
INSERT INTO
   "DESTINATION_SQL_STREAM"
   SELECT
      STREAM STEP("SOURCE_SQL_STREAM_001"."tradeTimestamp" BY INTERVAL '60' SECOND) AS
 EVENT_TIME,
      STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND) AS
 "STREAM_INGEST_TIME",
      "ticker",
       COUNT(*) AS VOLUME,
      AVG("tradePrice") AS AVG_PRICE,
      MIN("tradePrice") AS MIN_PRICE,
      MAX("tradePrice") AS MAX_PRICEFROM "SOURCE_SQL_STREAM_001"
   GROUP BY
      "ticker",
      STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND),
      STEP("SOURCE_SQL_STREAM_001"."tradeTimestamp" BY INTERVAL '60' SECOND);
```

O SQL acima usa duas janelas de tempo — `tradeTimestamp` que vêm da carga útil do fluxo de entrada e também `ROWTIME.tradeTimestamp` são chamadas de `Event Time` ou `client-side`

`time`. Muitas vezes, é desejável usar esse horário em análises, porque é o momento em que um evento ocorreu. No entanto, muitas fontes de eventos, como celulares e clientes da Web, não têm relógios confiáveis, o que pode levar a horários imprecisos. Além disso, problemas de conectividade podem levar a registros que aparecem em um stream não na mesma ordem em que os eventos ocorreram.

Os streams no aplicativo também incluem uma coluna especial chamada `ROWTIME`. Ela armazena um timestamp quando o Amazon Kinesis Data Analytics insere uma linha no primeiro stream do aplicativo. `ROWTIME` reflete o timestamp no qual o Amazon Kinesis Data Analytics inseriu um registro no primeiro stream no aplicativo após ler a partir da fonte de streaming. Esse valor `ROWTIME` então é mantido em todo o aplicativo.

O SQL determina a contagem do código de negociação como preço `volume`, `min`, `max` e `average` em um intervalo de 60 segundos.

Usar cada um desses horários nas consultas em janelas baseadas em horário tem vantagens e desvantagens. Escolha um ou mais desses horários e uma estratégia para lidar com as relevantes desvantagens de acordo com o caso de uso.

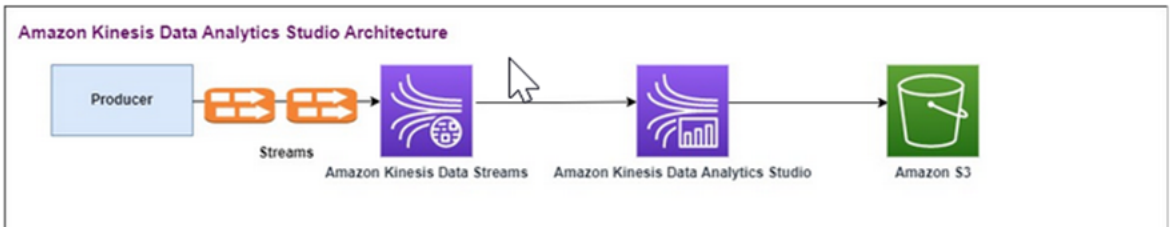
Uma estratégia de duas janelas que usam dois horários, o `ROWTIME` e um dos outros horários como a hora do evento.

- Use o `ROWTIME` como a primeira janela, que controla a frequência com que a consulta emite os resultados, como mostrado no exemplo a seguir. Ele não é usado como um horário lógico.
- Use um dos outros horários lógicos que deseja associar à sua análise. Esse horário representa quando o evento ocorreu. No exemplo a seguir, o objetivo da análise é agrupar os registros e retornar a contagem pelo marcador.

Amazon Managed Service for Apache Flink Studio

Na arquitetura atualizada, você substitui o Amazon Kinesis Data Firehose pelo Amazon Kinesis Data Streams. O Amazon Kinesis Data Analytics para aplicativos SQL foi substituído pelo Amazon Managed Service for Apache Flink Studio. O código do Apache Flink é executado interativamente em um caderno de notas Apache Zeppelin. O Amazon Managed Service for Apache Flink Studio envia os dados comerciais agregados em um bucket do Amazon S3 para armazenamento. As etapas são mostradas a seguir:

Aqui está o fluxo arquitetônico do Amazon Managed Service for Apache Flink Studio:



Criar um fluxo de dados Kinesis

Para criar um stream de dados usando o console

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Na barra de navegação, expanda o seletor de região e escolha uma região.
3. Selecione Create data stream (Criar stream de dados).
4. Na página Criar stream Kinesis, insira um nome para seu fluxo de dados e aceite o modo de capacidade sob demanda padrão.

   No modo sob demanda, você pode, em seguida, escolher Create stream do Kinesis para criar o fluxo de dados.

   Na página Kinesis streams (Streamings do Kinesis), o Status do streaming é Creating (Criando) enquanto o streaming está sendo criado. Quando o streaming estiver pronto para ser usado, o Status mudará para Active (Ativo).
5. Escolha o nome do fluxo. A página Stream Details (Detalhes do streaming) exibe um resumo da configuração do streaming com informações de monitoramento.
6. No Amazon Kinesis Data Generator, altere o stream/stream de entrega para o novo Amazon Kinesis Data Streams: TRADE_SOURCE_STREAM.

   O JSON e a carga útil serão os mesmos que você usou para o Amazon Kinesis Data Analytics-SQL. Use o Amazon Kinesis Data Generator para produzir alguns exemplos de dados de carga útil de negociação e direcionar o fluxo de dados TRADE_SOURCE_STREAM para este exercício:

```
{{date.now(YYYY-MM-DD HH:mm:ss.SSS)}},
"{{random.arrayElement(["AAPL","AMZN","MSFT","META","GOOGL"])}}",
{{random.number(2000)}}
```

7. No AWS Management Console ir ao Managed Service for Apache Flink e então escolher Criar aplicativo.
8. No painel de navegação à esquerda, escolha Cadernos de nota Studio e selecione Criar caderno de notas studio.
9. Insira um nome para o caderno de notas studio.
10. Em Banco de dados GlueAWS, forneça um banco de dados AWS Glue existente que definirá os metadados para suas fontes e destinos. Caso não tenha um banco de dados AWS Glue, escolha Criar e faça o seguinte:

    a. No console Glue AWS, escolha Bancos de dados em Catálogo de dados no menu à esquerda.

    b. Escolha Criar banco de dados

    c. Na página Criar banco de dados, insira um nome para o banco de dados. Na seção Localização – opcional, escolha Procurar no Amazon S3 e selecione o bucket do Amazon S3. Se ainda não tiver um bucket do Amazon S3 configurado, você pode pular essa etapa e retornar posteriormente.

    d. (Optional). Insira uma descrição para o banco de dados.

    e. Escolha Criar banco de dados.
11. Escolha Criar bloco de anotações.
12. Depois que seu caderno de notas for criado, escolha Executar.
13. Depois que o caderno de notas for iniciado com sucesso, inicie um caderno de notas Zeppelin escolhendo Abrir no Apache Zeppelin.
14. Na página do Caderno de notas Zeppelin, escolha Criar nova nota e chame-a de MarketDataFeed.

O código SQL do Flink é explicado a seguir, mas primeiro essa é a aparência da tela de um caderno de notas Zeppelin. Cada janela dentro do caderno de notas é um bloco de código separado e elas podem ser executadas uma de cada vez.

Código do Amazon Managed Service for Apache Flink Studio

O Amazon Managed Service for Apache Flink Studio usa os cadernos de nota Zeppelin para executar o código. O mapeamento é feito neste exemplo para código ssql baseado no Apache Flink 1.13. O código no caderno de notas Zeppelin é mostrado abaixo, um bloco por vez.

Antes de executar qualquer código em seu caderno de notas Zeppelin, os comandos de configuração do Flink devem ser executados. Se precisar alterar qualquer configuração após executar o código (ssql, Python ou Scala), você precisará parar e reiniciar seu caderno de notas. Neste exemplo, você precisará definir o ponto de verificação. É necessário um ponto de verificação para que você possa transmitir dados em um arquivo no Amazon S3. Isso permite que o fluxo de dados para o Amazon S3 seja transferido para um arquivo. A instrução abaixo define o intervalo para 5000 milissegundos.

```
%flink.conf
execution.checkpointing.interval 5000
```

`%flink.conf` indica que esse bloco são instruções de configuração. Para obter mais informações sobre a configuração do Flink, incluindo pontos de verificação, consulte Ponto de verificação do Apache Flink.

A tabela de entrada para o Amazon Kinesis Data Streams de origem é criada com o código ssql do Flink abaixo. Observe que o campo `TRADE_TIME` armazena a data/hora criada pelo gerador de dados.

```
%flink.ssql

DROP TABLE IF EXISTS TRADE_SOURCE_STREAM;
CREATE TABLE TRADE_SOURCE_STREAM (--`arrival_time` TIMESTAMP(3) METADATA FROM
 'timestamp' VIRTUAL,
TRADE_TIME TIMESTAMP(3),
WATERMARK FOR TRADE_TIME as TRADE_TIME - INTERVAL '5' SECOND,TICKER STRING,PRICE
 DOUBLE,
STATUS STRING)WITH ('connector' = 'kinesis','stream' = 'TRADE_SOURCE_STREAM',
'aws.region' = 'us-east-1','scan.stream.initpos' = 'LATEST','format' = 'csv');
```

Você pode visualizar o fluxo de entrada com esta instrução:

```
%flink.ssql(type=update)-- testing the source stream

select * from TRADE_SOURCE_STREAM;
```

Antes de enviar os dados agregados para o Amazon S3, você pode visualizá-los diretamente no Amazon Managed Service for Apache Flink com uma janela em cascata para selecionar uma consulta. Isso agrega os dados de negociação em uma janela de tempo de um minuto. Observe que a instrução %flink.ssql deve ter uma designação (tipo=atualizar):

```
%flink.ssql(type=update)

select TUMBLE_ROWTIME(TRADE_TIME,
INTERVAL '1' MINUTE) as TRADE_WINDOW,
TICKER, COUNT(*) as VOLUME,
AVG(PRICE) as AVG_PRICE,
MIN(PRICE) as MIN_PRICE,
MAX(PRICE) as MAX_PRICE FROM TRADE_SOURCE_STREAMGROUP BY TUMBLE(TRADE_TIME, INTERVAL
 '1' MINUTE), TICKER;
```

Em seguida, você poderá criar uma tabela para o destino no Amazon S3. Você precisa usar uma marca d'água. Uma marca d'água é uma métrica de progresso que indica um momento em que você tem certeza de que não haverá mais eventos atrasados. A marca d'água é para contabilizar chegadas tardias. O intervalo `‘5’ Second` permite que as negociações entrem no Amazon Kinesis Data Stream com 5 segundos de atraso e ainda sejam incluídas se tiverem um registro de data e hora na janela. Para obter mais informações, consulte Geração de marcas d´água.

```
%flink.ssql(type=update)

DROP TABLE IF EXISTS TRADE_DESTINATION_S3;
CREATE TABLE TRADE_DESTINATION_S3 (
TRADE_WINDOW_START TIMESTAMP(3),
WATERMARK FOR TRADE_WINDOW_START as TRADE_WINDOW_START - INTERVAL '5' SECOND,
TICKER STRING,
VOLUME BIGINT,
AVG_PRICE DOUBLE,
MIN_PRICE DOUBLE,
MAX_PRICE DOUBLE)
WITH ('connector' = 'filesystem','path' = 's3://trade-destination/','format' = 'csv');
```

Essa instrução insere os dados no `TRADE_DESTINATION_S3`. `TUMPLE_ROWTIME` é o time stamp do limite superior inclusivo da janela em cascata.

```
%flink.ssql(type=update)

insert into TRADE_DESTINATION_S3
select TUMBLE_ROWTIME(TRADE_TIME,
INTERVAL '1' MINUTE),
TICKER, COUNT(*) as VOLUME,
AVG(PRICE) as AVG_PRICE,
MIN(PRICE) as MIN_PRICE,
```

```
MAX(PRICE) as MAX_PRICE FROM TRADE_SOURCE_STREAM
GROUP BY TUMBLE(TRADE_TIME, INTERVAL '1' MINUTE), TICKER;
```

Deixe sua instrução ser executada por 10 a 20 minutos para acumular alguns dados no Amazon S3. Então aborte sua instrução.

Isso fecha o arquivo no Amazon S3 para que fique visível.

Aqui está a aparência do conteúdo:

Você pode usar o modelo AWS CloudFormation para criar a infraestrutura.

AWS CloudFormation criará os seguintes recursos na sua conta AWS:

- Amazon Kinesis Data Streams
- Amazon Managed Service for Apache Flink Studio
- Banco de dados Amazon Glue
- Bucket do Amazon S3
- Perfis e políticas do IAM para o Amazon Managed Service for Apache Flink Studio para acessar os recursos adequados

Importe o caderno de notas e altere o nome do bucket do Amazon S3 com o novo bucket do Amazon S3 criado por AWS CloudFormation.

```
%flink.ssql(type=update)

DROP TABLE IF EXISTS  TRADE_DESTINATION_S3;
CREATE TABLE TRADE_DESTINATION_S3 (
TRADE_WINDOW_START TIMESTAMP(3),
WATERMARK FOR TRADE_WINDOW_START as TRADE_WINDOW_START - INTERVAL '5' SECOND,
TICKER STRING,
VOLUME BIGINT,
AVG_PRICE DOUBLE,
MIN_PRICE DOUBLE,
MAX_PRICE DOUBLE)
WITH ('connector' = 'filesystem','path' = 's3://kda-studio-test-stack-markettradinganalyticsc[redacted]/','format' = 'csv');
```

Veja mais

Aqui estão alguns recursos adicionais que você pode usar para saber mais sobre o uso do Managed Service for Apache Flink Studio:

- Guia para desenvolvedores de serviços gerenciados para o caderno de notas Apache Flink Studio
- Documentação do Apache Flink 1.13
- Managed Service for Apache Flink Studio Workshop
- Janelas do Apache Flink
- Guia do desenvolvedor do Amazon Kinesis Data Analytics — Como escrever um stream do Kinesis Data Analytics para um bucket S3

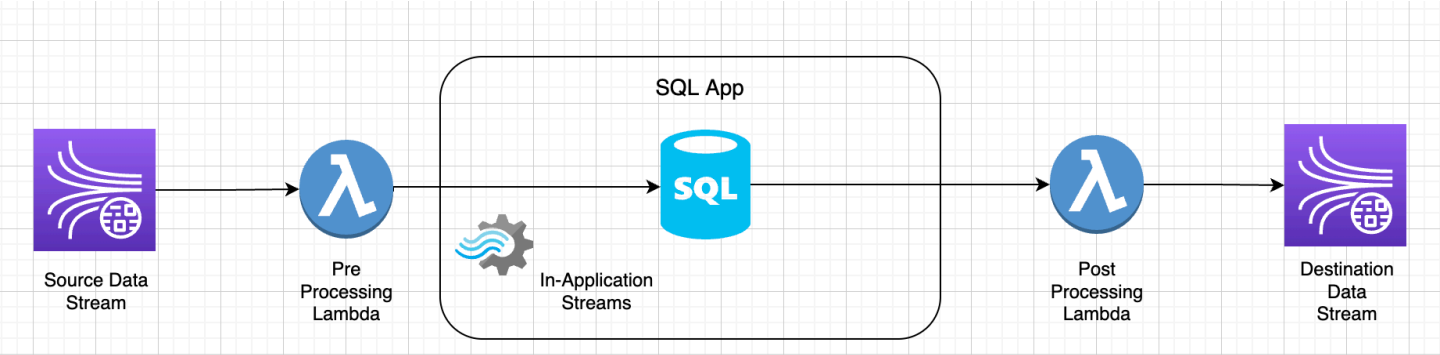
Aproveitar as funções definidas pelo usuário (UDFs)

O objetivo do padrão é demonstrar como aproveitar as UDFs nos cadernos de notas Kinesis Data Analytics-Studio Zeppelin para processar dados no stream do Kinesis. O Managed Service for Apache Flink Studio usa o Apache Flink para fornecer funcionalidades analíticas avançadas, incluindo semântica de processamento de uma única vez, janelas de horário de eventos, extensibilidade usando funções definidas pelo usuário e integrações personalizadas, suporte a idiomas imperativos, estado durável do aplicativo, escalabilidade horizontal, suporte para várias fontes de dados, integrações extensíveis e muito mais. Eles são essenciais para garantir a precisão, integridade, consistência e confiabilidade do processamento de fluxos de dados e não estão disponíveis com o Amazon Kinesis Data Analytics para SQL.

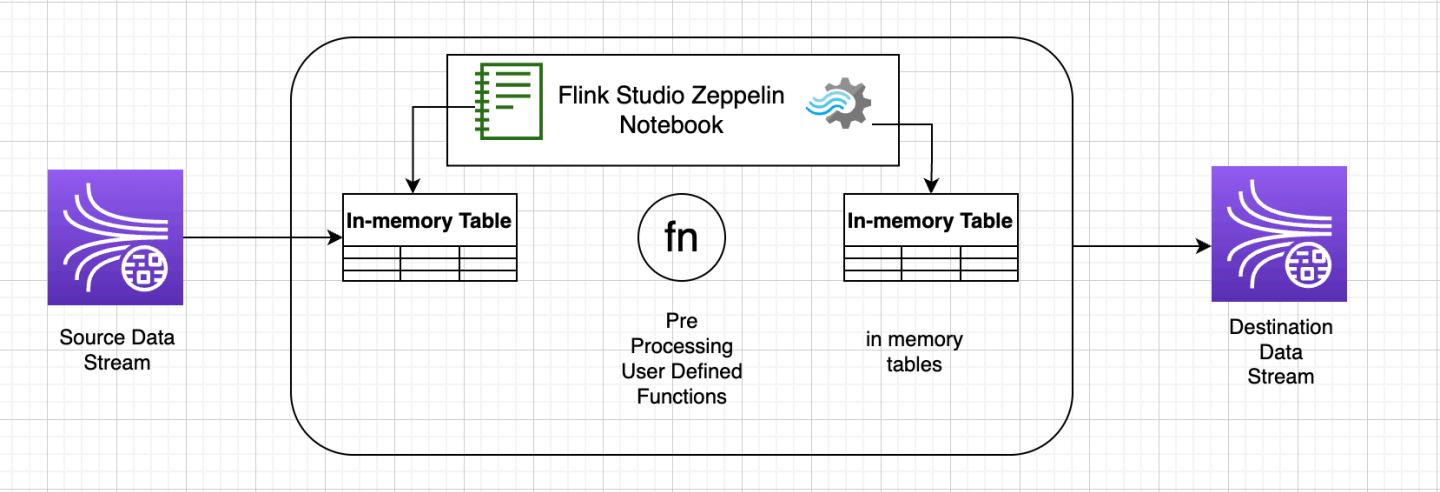
Neste aplicativo de exemplo, demonstraremos como aproveitar as UDFs no caderno de notas KDA-Studio Zeppelin para processar dados no stream do Kinesis. Os cadernos de notas Studio para Kinesis Data Analytics permitem que você consulte interativamente fluxos de dados em tempo real e crie e execute facilmente aplicativos de processamento de streams usando SQL, Python e Scala

padrão. Com alguns cliques no AWS Management Console, você pode iniciar um caderno de notas sem servidor para consultar fluxos de dados e obter resultados em segundos. Para obter mais informações, consulte Usar um caderno de notas Studio com o Kinesis Data Analytics para Apache Flink.

Funções do Lambda usadas para pré e pós-processamento de dados em aplicativos KDA-SQL:



Perfis definidos pelo usuário para pré- e pós-processamento de dados usando caderno de notas KDA-Studio Zeppelin



Funções definidas pelo usuário (UDFs)

Para reutilizar a lógica comercial comum em um operador, pode ser útil fazer referência a uma função definida pelo usuário para transformar seu fluxo de dados. Isso pode ser feito no caderno de notas Managed Service for Apache Flink Studio ou como um arquivo jar de aplicativo referenciado externamente. A utilização de funções definidas pelo usuário pode simplificar as transformações ou enriquecimentos de dados que você pode realizar em fluxo de dados.

Em seu caderno de notas, você fará referência a um simples jar de aplicativos Java que tem a funcionalidade de anonimizar números de telefone pessoais. Você também pode gravar UDFs em

Python ou Scala para uso no caderno de notas. Escolhemos um jar de aplicativo Java para destacar a funcionalidade de importar um jar de aplicativo em um caderno de notas Pyflink.

Configuração do ambiente

Para seguir este guia e interagir com seus dados de streaming, você usará um script AWS CloudFormation para iniciar os seguintes recursos:

- Kinesis Data Streams como origem e destino
- Banco de dados Glue
- Perfil do IAM
- Aplicativo do Managed Service for Apache Flink Studio
- Função do Lambda para iniciar o aplicativo Managed Service for Apache Flink Studio
- Função do Lambda para executar a função do Lambda acima
- Recurso personalizado para invocar a função do Lambda

Faça download do modelo do AWS CloudFormation aqui.

Crie a pilha AWS CloudFormation.

1. Vá até AWS Management Console e escolha CloudFormation na lista de serviços.
2. Na página CloudFormation, escolha Pilhas e depois escolha Criar pilha com novos recursos (padrão).
3. Na página Criar pilha, escolha Carregar um arquivo de modelo e, em seguida, escolha o `kda-flink-udf.yml` que você baixou anteriormente. Faça o upload do arquivo e escolha Próximo.
4. Dê um nome ao modelo, como `kinesis-UDF`, para que seja fácil de lembrar, e atualize os parâmetros de entrada, como fluxo de entrada, se quiser um nome diferente. Escolha Próximo.
5. Na página Configurar opções de pilha, adicione Tags se desejar e escolha Próximo.
6. Na página Revisão, marque as caixas que permitem a criação de recursos do IAM e escolha Enviar.

A pilha AWS CloudFormation pode levar de 10 a 15 minutos para ser lançada, dependendo da região em que você está lançando. Depois de ver o status `CREATE_COMPLETE` de toda a pilha, você está pronto para continuar.

Trabalhar com o caderno de notas Managed Service for Apache Flink Studio

Os cadernos de notas Studio para Kinesis Data Analytics permitem que você consulte interativamente fluxos de dados em tempo real e crie e execute facilmente aplicativos de processamento de streams usando SQL, Python e Scala padrão. Com alguns cliques no AWS Management Console, você pode iniciar um caderno de notas sem servidor para consultar fluxos de dados e obter resultados em segundos.

Um caderno de notas é um ambiente de desenvolvimento baseado na web. Com o caderno de notas, você obtém uma experiência simples de desenvolvimento interativo combinada com os recursos avançados de processamento de fluxo de dados fornecidos pelo Apache Flink. Os cadernos de notas Studio usam cadernos de notas equipados com Apache Zeppelin e usam o Apache Flink como mecanismo de processamento de streaming. Os cadernos de notas Studio combinam perfeitamente essas tecnologias para tornar a análise avançada em fluxos de dados acessível a desenvolvedores de todos os conjuntos de habilidades.

O Apache Zeppelin fornece aos seus cadernos de notas Studio um conjunto completo de ferramentas de análise, incluindo as seguintes:

- Visualização de dados
- Exportar dados para arquivos
- Controlar o formato da saída para análise mais fácil

Uso de caderno de notas

1. Vá até AWS Management Console e escolha Amazon Kinesis na lista de serviços.
2. Na página de navegação à esquerda, escolha Aplicativos Analytics e, em seguida, escolha Cadernos de notas Studio.
3. Verifique se o caderno de notas KinesisDataAnalyticsStudio está em execução.
4. Escolha o caderno de notas e, em seguida, escolha Abrir no Apache Zeppelin.
5. Faça o download do arquivo do Caderno de notas produtor de dados Zeppelin, que você usará para ler e carregar dados no Kinesis Stream.
6. Importe o caderno de notas Zeppelin `Data Producer`. Certifique-se de modificar a entrada `STREAM_NAME` e `REGION` o código do caderno de notas. O nome do fluxo de entrada pode ser encontrado na saída da pilhaAWS CloudFormation.

7. Execute o caderno de notas Produtor de dados escolhendo o botão Executar este parágrafo para inserir dados de amostra no Kinesis Data Stream de entrada.
8. Enquanto os dados de amostra são carregados, baixe o Caderno de notas interativo MaskPhoneNumber, que lerá os dados de entrada, anonimizará os números de telefone do fluxo de entrada e armazenará dados anônimos no fluxo de saída.
9. Importe o caderno de notas Zeppelin `MaskPhoneNumber-interactive`.
10. Execute cada parágrafo no caderno de notas.

    a. No parágrafo 1, você importa a Função Definida pelo Usuário para anonimizar os números de telefone.

    ```
    %flink(parallelism=1)
    import com.mycompany.app.MaskPhoneNumber
    stenv.registerFunction("MaskPhoneNumber", new MaskPhoneNumber())
    ```

    b. No próximo parágrafo, você cria uma tabela na memória para ler os dados do fluxo de entrada. Verifique se o nome e a região AWS estão corretos.

    ```
    %flink.ssql(type=update)

    DROP TABLE IF EXISTS customer_reviews;

    CREATE TABLE customer_reviews (
    customer_id VARCHAR,
    product VARCHAR,
    review VARCHAR,
    phone VARCHAR
    )
    WITH (
    'connector' = 'kinesis',
    'stream' = 'KinesisUDFSampleInputStream',
    'aws.region' = 'us-east-1',
    'scan.stream.initpos' = 'LATEST',
    'format' = 'json');
    ```

    c. Verifique se os dados estão carregados na tabela na memória.

    ```
    %flink.ssql(type=update)
    select * from customer_reviews
    ```

    d. Invoque a função definida pelo usuário para anonimizar o número de telefone.

```
%flink.ssql(type=update)
select customer_id, product, review, MaskPhoneNumber('mask_phone', phone) as
 phoneNumber from customer_reviews
```

e. Agora que os números de telefone estão mascarados, crie uma visualização com um número mascarado.

```
%flink.ssql(type=update)

DROP VIEW IF EXISTS sentiments_view;

CREATE VIEW
    sentiments_view
AS
  select customer_id, product, review, MaskPhoneNumber('mask_phone', phone) as
 phoneNumber from customer_reviews
```

f. Verifique os dados.

```
%flink.ssql(type=update)
select * from sentiments_view
```

g. Crie uma tabela na memória para a saída do Kinesis Stream. Verifique se o nome e a região AWS estão corretos.

```
%flink.ssql(type=update)

DROP TABLE IF EXISTS customer_reviews_stream_table;

CREATE TABLE customer_reviews_stream_table (
customer_id VARCHAR,
product VARCHAR,
review VARCHAR,
phoneNumber varchar
)
WITH (
'connector' = 'kinesis',
'stream' = 'KinesisUDFSampleOutputStream',
'aws.region' = 'us-east-1',
'scan.stream.initpos' = 'TRIM_HORIZON',
'format' = 'json');
```

h. Insira registros atualizados no Kinesis Stream de destino.

```
%flink.ssql(type=update)
INSERT INTO customer_reviews_stream_table
SELECT customer_id, product, review, phoneNumber
FROM sentiments_view
```

i. Visualize e verifique os dados do Kinesis Stream de destino.

```
%flink.ssql(type=update)
select * from customer_reviews_stream_table
```

Promover um caderno de notas como aplicativo

Agora que você testou o código do seu caderno de notas de forma interativa, você implantará o código como um aplicativo de streaming com estado durável. Você precisará primeiro modificar a configuração do aplicativo para especificar um local para seu código no Amazon S3.

1. Em AWS Management Console, escolha seu caderno de notas e, em Implantar como configuração do aplicativo - opcional, escolha Editar.
2. Em Destino para código no Amazon S3, escolha o bucket do Amazon S3 que foi criado pelos scripts AWS CloudFormation. O processo pode levar alguns minutos.
3. Não será possível promover a nota do jeito que está. Se você tentar, receberá um erro como as declarações `Select` não são suportadas. Para evitar esse problema, baixe o caderno de notas MaskPhoneNumber-streaming Zeppelin.
4. Importe o caderno de notas Zeppelin `MaskPhoneNumber-streaming`.
5. Abra a nota e escolha Ações para o KinesisDataAnalyticsStudio.
6. Escolha Build MaskPhoneNumber-streaming e exporte para o S3. Certifique-se de renomear o Nome do aplicativo e não incluir caracteres especiais.
7. Escolha Criar e exportar. Levará alguns minutos para configurar o aplicativo de streaming.
8. Quando a compilação estiver concluída, escolha Implantar usando o console AWS.
9. Na próxima página, revise as configurações e certifique-se de escolher o perfil do IAM adequado. Em seguida, escolha Criar aplicativo de streaming.
10. Depois de alguns minutos, você verá uma mensagem informando que o aplicativo de streaming foi criado com sucesso.

Para obter mais informações sobre a implantação de aplicativos com estado durável e limites, consulte Implantação como um aplicativo com estado durável.

Limpeza

Opcionalmente, agora você pode desinstalar a pilha AWS CloudFormation. Isso removerá todos os serviços que você configurou anteriormente.

Exemplos de Kinesis Data Analytics para SQL

Esta seção apresenta exemplos de como criar e trabalhar com aplicativos no Amazon Kinesis Data Analytics. Eles incluem código de exemplo e instruções passo a passo para ajudar você a criar aplicativos Kinesis Data Analytics e testar seus resultados.

Antes de explorar esses exemplos, recomendamos que você primeiro analise Amazon Kinesis Data Analytics para aplicativos SQL: como funciona e Conceitos básicos do Amazon Kinesis Data Analytics para aplicativos SQL.

Tópicos

Exemplos: transformação de dados

Às vezes o código de aplicativo precisa pré-processar registros de entrada antes de executar qualquer análise no Amazon Kinesis Data Analytics. Isso pode acontecer por vários motivos, como registros sem conformidade com os formatos de registro compatíveis, resultando em colunas não normalizadas em streams de entrada no aplicativo.

Esta seção fornece exemplos de como usar as funções de string disponíveis para normalizar dados, como extrair as informações necessárias de colunas de strings e assim por diante. A seção também aponta para funções de data e hora que podem ser úteis.

Pré-processar streamings com Lambda

Para obter informações sobre o pré-processamento de fluxos com AWS Lambda, consulte. Pré-processar dados usando uma função do Lambda

Tópicos

- Exemplo: Transformação de valores DateTime
- Exemplo: transformação de vários tipos de dados

Exemplos: transformação de valores de string

O Amazon Kinesis Data Analytics oferece suporte a formatos como JSON e CSV para registros em uma origem de streaming. Para obter mais detalhes, consulte RecordFormat. Em seguida, esses registros são mapeados para linhas no fluxo em um aplicativo de acordo com a configuração de entrada. Para obter mais detalhes, consulte Configuração de entrada do aplicativo. A configuração de entrada especifica como os campos de registro na origem de streaming são mapeados para colunas no fluxo de um aplicativo.

Esse mapeamento funciona quando os registros na origem de streaming têm os formatos compatíveis, o que resulta em um fluxo de aplicativo com dados normalizados. E se os dados na origem de streaming não estiver de acordo com os padrões compatíveis? Por exemplo, e se a origem de streaming contiver dados como dados de clickstream, sensores de IoT e logs de aplicativo?

Considere estes exemplos:

- A origem de streaming contém logs de aplicativo – Os logs de aplicativo têm o formato de log padrão do Apache e são gravados no stream com o formato JSON.

  ```
  {
     "Log":"192.168.254.30 - John [24/May/2004:22:01:02 -0700] "GET /icons/
  apache_pb.gif HTTP/1.1" 304 0"
  }
  ```

  Para obter mais informações sobre o formato de log padrão do Apache, consulte arquivos de log no site do Apache.

- A origem de streaming contém dados semiestruturados – O exemplo a seguir mostra dois registros. O valor de campo `Col_E_Unstructured` é uma série de valores separados por vírgulas. Há cinco colunas, as quatro primeiras têm valores de tipo de string e a última coluna contém valores separados por vírgulas.

  ```
  { "Col_A" : "string",
  ```

```
  "Col_B" : "string",
  "Col_C" : "string",
  "Col_D" : "string",
  "Col_E_Unstructured" : "value,value,value,value"}

{ "Col_A" : "string",
  "Col_B" : "string",
  "Col_C" : "string",
  "Col_D" : "string",
  "Col_E_Unstructured" : "value,value,value,value"}
```

- Os registros na origem de streaming contêm URLs, e você precisa de uma parte do nome de domínio da URL para fins de análise.

```
{ "referrer" : "http://www.amazon.com"}
{ "referrer" : "http://www.stackoverflow.com" }
```

Nesses casos, o seguinte processo de duas etapas geralmente funciona para a criação de fluxos de aplicativo que contêm dados normalizados:

1. Configure a entrada do aplicativo para mapear o campo não estruturado para uma coluna do tipo `VARCHAR(N)` no fluxo de entrada de aplicativo criado.
2. No código do aplicativo, use as funções de string para dividir esta única coluna em várias colunas e, em seguida, salve as linhas em outro stream no aplicativo. Este stream no aplicativo criado pelo código de aplicativo terá dados normalizados. Você poderá, então, executar análises nesse stream no aplicativo.

O Amazon Kinesis Data Analytics fornece as seguintes operações de string, funções SQL padrão e extensões para o padrão SQL para trabalhar com colunas de string:

- Operadores de string: Operadores como `LIKE` e `SIMILAR` são úteis nas strings de comparação. Para obter mais informações sobre identificadores, consulte Operadores de string em Referência SQL do Amazon Managed Service for Apache Flink.
- Funções SQL: as funções a seguir são úteis ao manipular strings individuais. Para obter mais informações, consulte Funções de string e pesquisa em Referência SQL do Amazon Managed Service for Apache Flink.
  - `CHAR_LENGTH` – Fornece o tamanho de uma string.

- `INITCAP` – Retorna uma versão convertida da string de entrada, de forma que o primeiro caractere de cada palavra delimitada por espaço é em maiúscula e todos os outros caracteres em minúscula.
- `LOWER/UPPER` – Converte uma string em minúsculas ou maiúsculas.
- `OVERLAY` – Substitua uma parte do primeiro argumento de string (a string original) pelo segundo argumento de string (a string substituta).
- `POSITION` – Procura uma string em outra string.
- `REGEX_REPLACE` – Substitui uma substring com uma substring alternativa.
- `SUBSTRING` – Extrai uma parte de uma string de origem a partir de uma posição específica.
- `TRIM` – Remove instâncias do caractere especificado do começo ou no fim da string de origem.

- Extensões SQL: são úteis para trabalhar com strings não estruturadas, como logs e URIs. Para obter mais informações, consulte Funções de análise de log em Amazon Managed Service for Apache Flink SQL Reference.
  - `FAST_REGEX_LOG_PARSER` – Funciona como o analisador regex, mas usa vários atalhos para garantir resultados mais rápidos. Por exemplo, o analisador regex rápido para na primeira correspondência encontrada (conhecida como semântica preguiçosa).
  - `FIXED_COLUMN_LOG_PARSE` – Analisa campos de largura fixa e os converte automaticamente nos tipos de SQL fornecidos.
  - `REGEX_LOG_PARSE` – Analisa uma string com base em padrões de expressão regular Java padrão.
  - `SYS_LOG_PARSE` – Analisa entradas normalmente encontradas nos logs de sistema do UNIX/Linux.
  - `VARIABLE_COLUMN_LOG_PARSE` – Divide uma string de entrada em campos separados por um caractere delimitador ou uma string delimitadora.
  - `W3C_LOG_PARSE` – Pode ser usado para formatação rápida de logs do Apache.

Para obter exemplos do uso dessas funções, consulte os seguintes tópicos:

Tópicos

- Exemplo: extração de parte de uma string (função SUBSTRING)
- Exemplo: substituição de uma substring usando regex (função REGEX_REPLACE)
- Exemplo: análise de strings de log com base em expressões regulares (função REGEX_LOG_PARSE)

- Exemplo: análise de logs da web (função W3C_LOG_PARSE)
- Exemplo: divisão de strings de caracteres em vários campos (função VARIABLE_COLUMN_LOG_PARSE)

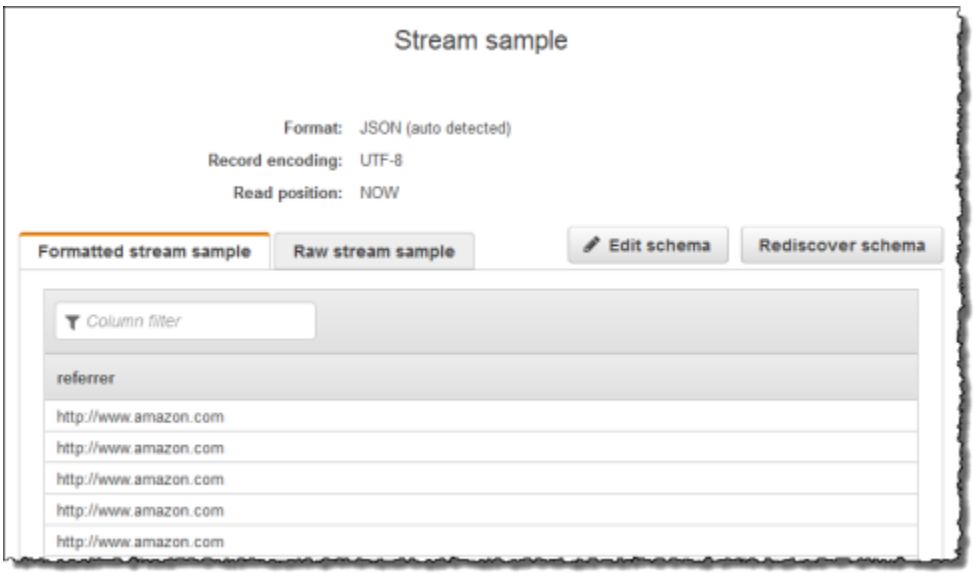
Exemplo: extração de parte de uma string (função SUBSTRING)

Este exemplo usa a função `SUBSTRING` para transformar uma string no Amazon Kinesis Data Analytics. A função `SUBSTRING` extrai uma parte de uma string de origem a partir de uma posição específica. Para obter mais informações, consulte SUBSTRING em Referência SQL do Amazon Managed Service for Apache Flink.

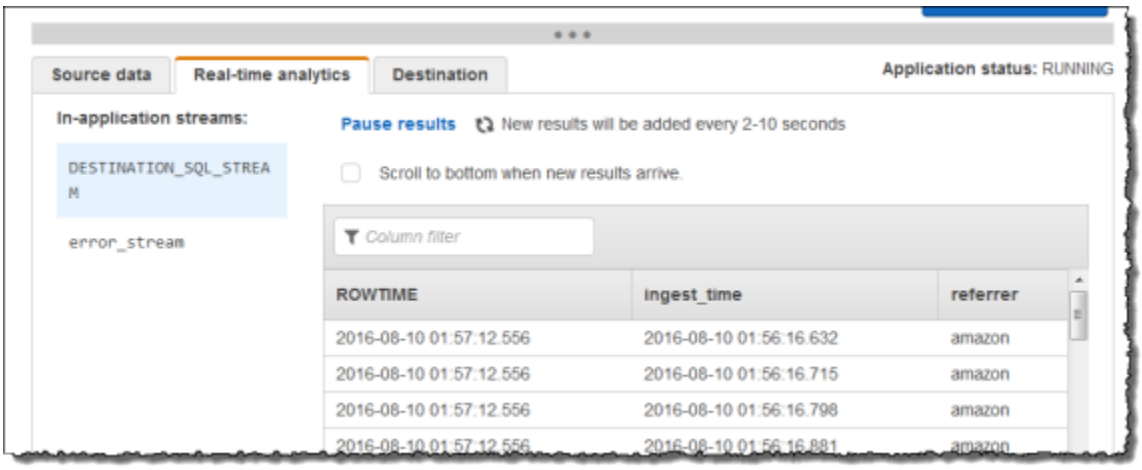
Neste exemplo, você grava os registros a seguir em um fluxo de dados do Amazon Kinesis.

```
{ "REFERRER" : "http://www.amazon.com" }
{ "REFERRER" : "http://www.amazon.com"}
{ "REFERRER" : "http://www.amazon.com"}
...
```

Em seguida, você criará um aplicativo Kinesis Data Analytics no console usando o fluxo de dados Kinesis como origem de streaming. O processo de descoberta lê registros de exemplo na origem de streaming e infere um esquema de aplicativo com uma coluna (`REFERRER`), como mostrado.



Em seguida, você usa o código do aplicativo com a função `SUBSTRING` para analisar a string de URL para recuperar o nome da empresa. Em seguida, insira os dados resultantes em outro fluxo de aplicativo, como mostramos a seguir:



Tópicos

- Etapa 1: Criar um fluxo de dados Kinesis
- Etapa 4: Criar o aplicativo Kinesis Data Analytics

Etapa 1: Criar um fluxo de dados Kinesis

Crie um Amazon Kinesis Data Streams e preencha registros de log da seguinte forma:

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Selecione Data Streams (Fluxos de dados) no painel de navegação.
3. Escolha Create Kinesis stream (Criar fluxo do Kinesis) e crie um fluxo com um estilhaço. Para obter mais informações, consulte Criar um stream no Guia do desenvolvedor do Amazon Kinesis Data Streams.
4. Execute o seguinte código Python para preencher os registros de log de exemplo. Esse código simples grava continuamente o mesmo registro de log no fluxo.

```
import json
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {"REFERRER": "http://www.amazon.com"}
```

```
def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Etapa 4: Criar o aplicativo Kinesis Data Analytics

Em seguida, crie um aplicativo Kinesis Data Analytics, da seguinte maneira:

1. Abra o console do Managed Service for Apache Flink em https://console.aws.amazon.com/kinesisanalytics.
2. Escolha Create application (Criar aplicativo), digite um nome para o aplicativo e selecione Create application (Criar aplicativo).
3. Na página de detalhes do aplicativo, escolha Connect streaming data (Conectar dados de streaming).
4. Na página Connect to source (Conectar com a fonte), faça o seguinte:
   a. Escolha o fluxo criado na seção anterior.
   b. Escolha a opção para criar uma função do IAM.
   c. Escolha Discover schema (Descobrir esquema). Aguarde o console mostrar o esquema inferido e os registros de exemplo usados para inferir o esquema do fluxo do aplicativo criado. O esquema inferido tem apenas uma coluna.
   d. Escolha Save and continue.
5. Na página de detalhes de aplicativo, escolha Go to SQL editor (Ir para o editor de SQL). Para iniciar o aplicativo, escolha Yes, start application (Sim, iniciar o aplicativo) na caixa de diálogo exibida.
6. No editor SQL, escreva o código do aplicativo e verifique os resultados da seguinte forma:

a. Copie o código de aplicativo a seguir e cole-o no editor.

```
-- CREATE OR REPLACE STREAM for cleaned up referrer
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    "ingest_time" TIMESTAMP,
    "referrer" VARCHAR(32));

CREATE OR REPLACE PUMP "myPUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM
         "APPROXIMATE_ARRIVAL_TIME",
         SUBSTRING("referrer", 12, (POSITION('.com' IN "referrer") -
 POSITION('www.' IN "referrer") - 4))
      FROM "SOURCE_SQL_STREAM_001";
```

b. Escolha Save and run SQL. Na guia Real-time analytics (Análise em tempo real), você pode ver todos os fluxos de aplicativo criados pelo aplicativo e verificar os dados.

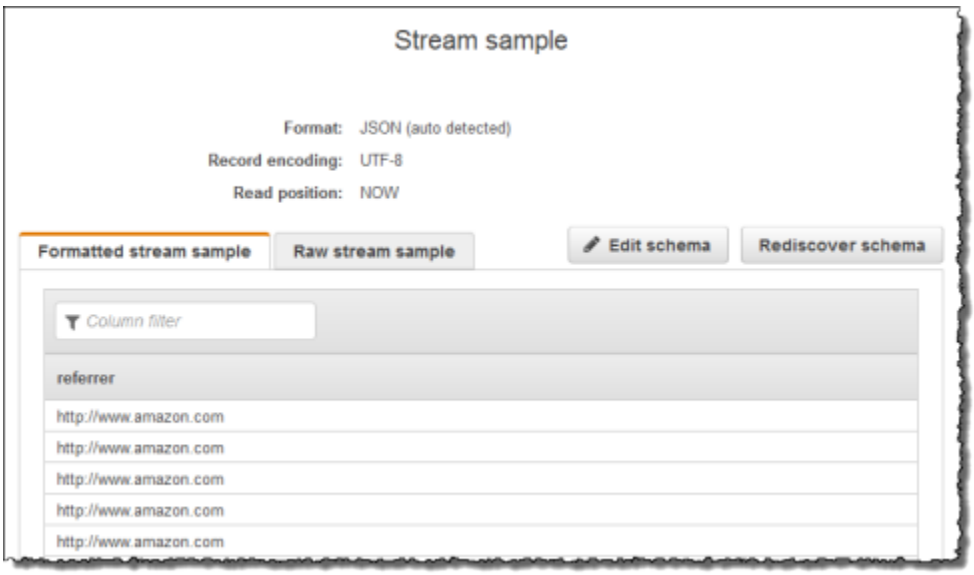
Exemplo: substituição de uma substring usando regex (função REGEX_REPLACE)

Este exemplo usa a função REGEX_REPLACE para transformar uma string no Amazon Kinesis Data Analytics. O REGEX_REPLACE substitui uma substring com uma substring alternativa. Para obter mais informações, consulte REGEX_REPLACE em Amazon Managed Service for Apache Flink SQL Reference.

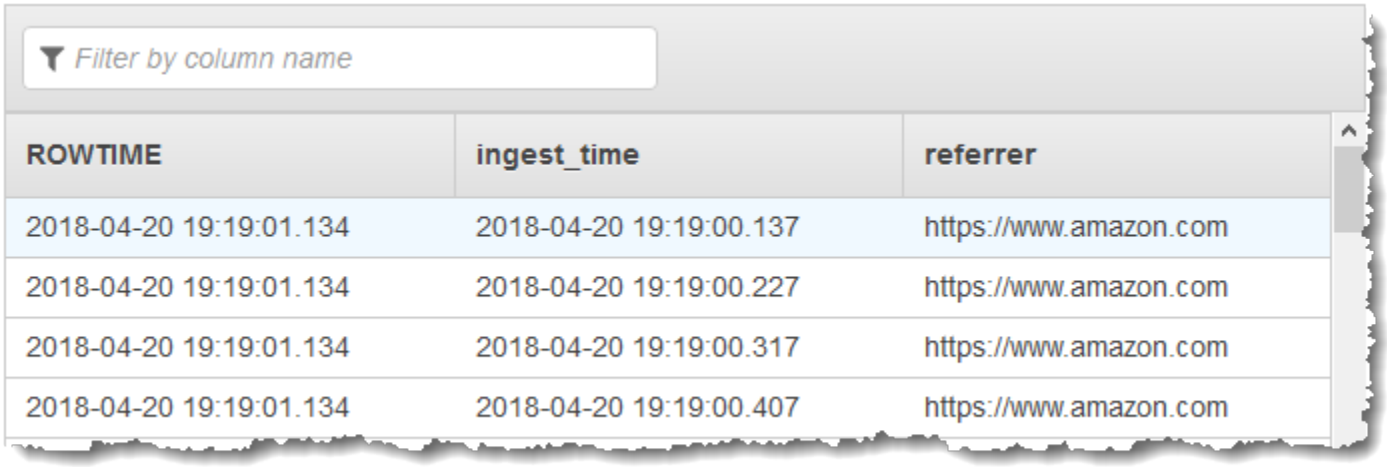
Neste exemplo, você grava os registros a seguir em um fluxo de dados do Amazon Kinesis.

```
{ "REFERRER" : "http://www.amazon.com" }
{ "REFERRER" : "http://www.amazon.com"}
{ "REFERRER" : "http://www.amazon.com"}
...
```

Em seguida, você criará um aplicativo Kinesis Data Analytics no console com o fluxo de dados Kinesis como origem de streaming. O processo de descoberta lê registros de exemplo na origem de streaming e infere um esquema de aplicativo com uma coluna (REFERRER), como mostrado.



Em seguida, use o código do aplicativo com a função `REGEX_REPLACE` para converter o URL e usar `https://` em vez de `http://`. Insira dos dados resultantes em outro fluxo de aplicativo, como mostramos a seguir:



Tópicos

- Etapa 1: Criar um fluxo de dados Kinesis
- Etapa 2: Criar o aplicativo Kinesis Data Analytics

Etapa 1: Criar um fluxo de dados Kinesis

Crie um fluxo de dados do Amazon Kinesis e preencha registros de log da seguinte forma:

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Selecione Data Streams (Fluxos de dados) no painel de navegação.
3. Escolha Create Kinesis stream (Criar fluxo do Kinesis) e crie um fluxo com um estilhaço. Para obter mais informações, consulte Criar um stream no Guia do desenvolvedor do Amazon Kinesis Data Streams.
4. Execute o seguinte código Python para preencher os registros de log de exemplo. Esse código simples grava continuamente o mesmo registro de log no fluxo.

```
import json
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {"REFERRER": "http://www.amazon.com"}


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Etapa 2: Criar o aplicativo Kinesis Data Analytics

Em seguida, crie um aplicativo Kinesis Data Analytics, da seguinte maneira:

1. Abra o Managed Service for Apache Flink console em https://console.aws.amazon.com/kinesisanalytics.

2. Escolha Create application (Criar aplicativo), digite um nome para o aplicativo e selecione Create application (Criar aplicativo).
3. Na página de detalhes do aplicativo, escolha Connect streaming data (Conectar dados de streaming).
4. Na página Connect to source (Conectar com a fonte), faça o seguinte:

   a. Escolha o fluxo criado na seção anterior.

   b. Escolha a opção para criar uma função do IAM.

   c. Escolha Discover schema (Descobrir esquema). Aguarde o console mostrar o esquema inferido e os registros de exemplo usados para inferir o esquema do fluxo do aplicativo criado. O esquema inferido tem apenas uma coluna.

   d. Escolha Save and continue.

5. Na página de detalhes de aplicativo, escolha Go to SQL editor (Ir para o editor de SQL). Para iniciar o aplicativo, escolha Yes, start application (Sim, iniciar o aplicativo) na caixa de diálogo exibida.
6. No editor SQL, escreva o código do aplicativo e verifique os resultados da seguinte forma:

   a. Copie o código de aplicativo a seguir e cole-o no editor:

```
-- CREATE OR REPLACE STREAM for cleaned up referrer
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    "ingest_time" TIMESTAMP,
    "referrer" VARCHAR(32));

CREATE OR REPLACE PUMP "myPUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM
         "APPROXIMATE_ARRIVAL_TIME",
         REGEX_REPLACE("REFERRER", 'http://', 'https://', 1, 0)
      FROM "SOURCE_SQL_STREAM_001";
```

   b. Escolha Save and run SQL. Na guia Real-time analytics (Análise em tempo real), você pode ver todos os fluxos de aplicativo criados pelo aplicativo e verificar os dados.

Exemplo: análise de strings de log com base em expressões regulares (função REGEX_LOG_PARSE)

Esse exemplo usa a função `REGEX_LOG_PARSE` para transformar uma string no Amazon Kinesis Data Analytics. O `REGEX_LOG_PARSE` analisa uma string baseada nos padrões de expressão comum de Java padrão. Para obter mais informações, consulte [REGEX_LOG_PARSE](#) em Referência SQL do Amazon Managed Service for Apache Flink.

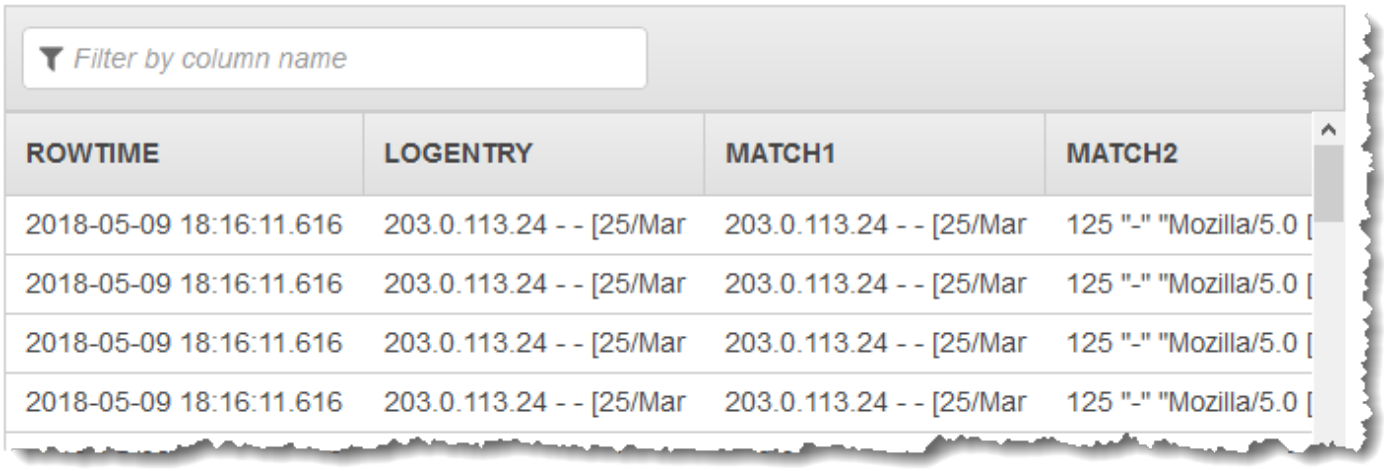
Neste exemplo, você grava os registros a seguir em um stream do Amazon Kinesis:

```
{"LOGENTRY": "203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] \"GET /index.php HTTP/1.1\"
 200 125 \"-\" \"Mozilla/5.0 [en] Gecko/20100101 Firefox/52.0\""}
{"LOGENTRY": "203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] \"GET /index.php HTTP/1.1\"
 200 125 \"-\" \"Mozilla/5.0 [en] Gecko/20100101 Firefox/52.0\""}
{"LOGENTRY": "203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] \"GET /index.php HTTP/1.1\"
 200 125 \"-\" \"Mozilla/5.0 [en] Gecko/20100101 Firefox/52.0\""}
...
```

Em seguida, você criará um aplicativo Kinesis Data Analytics no console com o fluxo de dados Kinesis como origem de streaming. O processo de descoberta lê registros de exemplo na origem de streaming e infere um esquema de aplicativo com uma coluna (LOGENTRY), como mostrado a seguir.

ROWTIME TIMESTAMP	LOGENTRY VARCHAR(256)
2018-05-09 18:12:18.552	203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] "GET /index.php HTTP/1.1"
2018-05-09 18:12:18.552	203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] "GET /index.php HTTP/1.1"
2018-05-09 18:12:18.552	203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] "GET /index.php HTTP/1.1"
2018-05-09 18:12:18.552	203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] "GET /index.php HTTP/1.1"

Em seguida, você usa o código do aplicativo com a função `REGEX_LOG_PARSE` para analisar a cadeia de log e recuperar os elementos de dados. Insira dos dados resultantes em outro fluxo de aplicativo, como mostramos na captura de tela a seguir:



Tópicos

- Etapa 1: Criar um fluxo de dados Kinesis
- Etapa 2: Criar o aplicativo Kinesis Data Analytics

Etapa 1: Criar um fluxo de dados Kinesis

Crie um fluxo de dados do Amazon Kinesis e preencha registros de log da seguinte forma:

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Selecione Data Streams (Fluxos de dados) no painel de navegação.
3. Escolha Create Kinesis stream (Criar fluxo do Kinesis) e crie um fluxo com um estilhaço. Para obter mais informações, consulte Criar um fluxo no Guia do desenvolvedor do Amazon Kinesis Data Streams.
4. Execute o seguinte código Python para preencher os registros de log de exemplo. Esse código simples grava continuamente o mesmo registro de log no fluxo.

```
import json
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
```

```
    return {
        "LOGENTRY": "203.0.113.24 - - [25/Mar/2018:15:25:37 -0700] "
        '"GET /index.php HTTP/1.1" 200 125 "-" '
        '"Mozilla/5.0 [en] Gecko/20100101 Firefox/52.0"'
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Etapa 2: Criar o aplicativo Kinesis Data Analytics

Em seguida, crie um aplicativo Kinesis Data Analytics, da seguinte maneira:

1. Abra o Managed Service for Apache Flink console em https://console.aws.amazon.com/kinesisanalytics.
2. Escolha Create application e especifique um nome de aplicativo.
3. Na página de detalhes do aplicativo, escolha Connect streaming data (Conectar dados de streaming).
4. Na página Connect to source (Conectar com a fonte), faça o seguinte:
    a. Escolha o fluxo criado na seção anterior.
    b. Escolha a opção para criar uma função do IAM.
    c. Escolha Discover schema (Descobrir esquema). Aguarde o console mostrar o esquema inferido e os registros de exemplo usados para inferir o esquema do fluxo do aplicativo criado. O esquema inferido tem apenas uma coluna.
    d. Escolha Save and continue.

5. Na página de detalhes de aplicativo, escolha Go to SQL editor (Ir para o editor de SQL). Para iniciar o aplicativo, escolha Yes, start application (Sim, iniciar o aplicativo) na caixa de diálogo exibida.
6. No editor SQL, escreva o código do aplicativo e verifique os resultados da seguinte forma:
   a. Copie o código de aplicativo a seguir e cole-o no editor.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (logentry VARCHAR(24), match1
 VARCHAR(24), match2 VARCHAR(24));

CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM T.LOGENTRY, T.REC.COLUMN1, T.REC.COLUMN2
    FROM
         (SELECT STREAM LOGENTRY,
             REGEX_LOG_PARSE(LOGENTRY, '(\w.+) (\d.+) (\w.+) (\w.+)') AS REC
             FROM SOURCE_SQL_STREAM_001) AS T;
```

   b. Escolha Save and run SQL. Na guia Real-time analytics (Análise em tempo real), você pode ver todos os fluxos de aplicativo criados pelo aplicativo e verificar os dados.

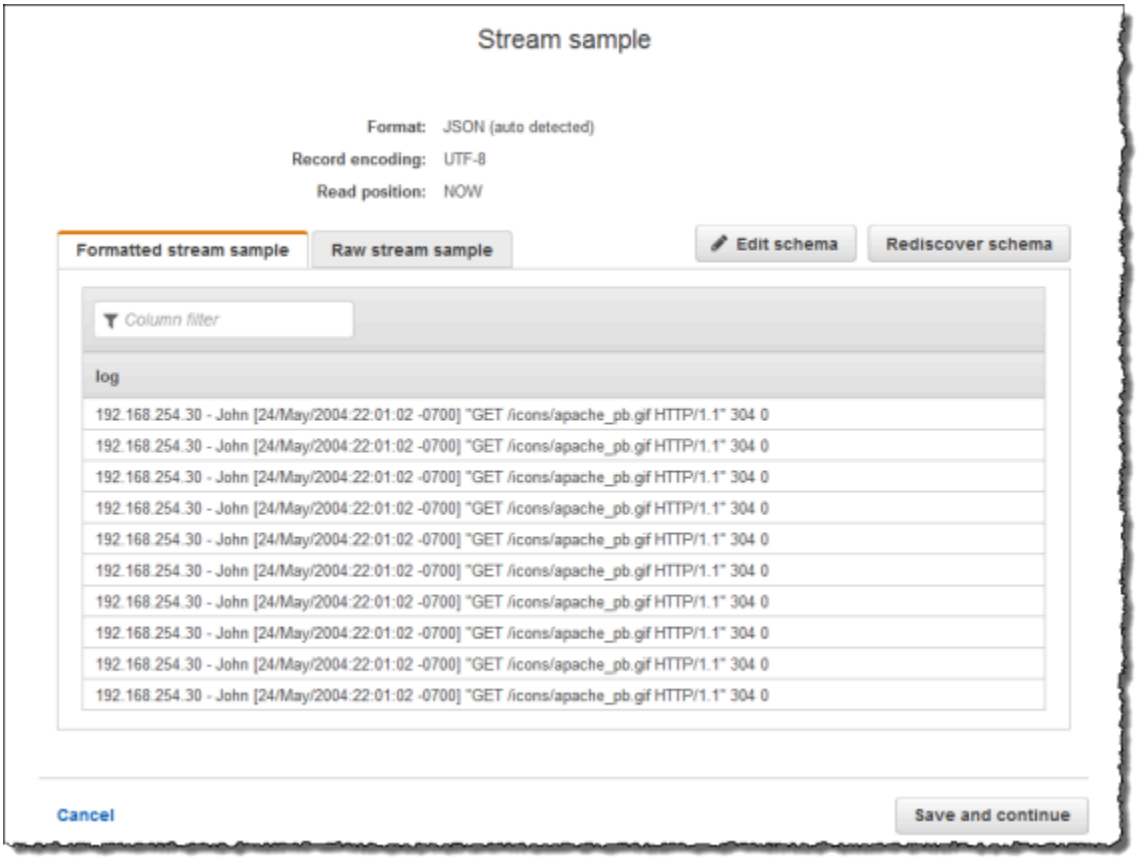
Exemplo: análise de logs da web (função W3C_LOG_PARSE)

Este exemplo usa a função `W3C_LOG_PARSE` para transformar uma string no Amazon Kinesis Data Analytics. Use `W3C_LOG_PARSE` para formatar rapidamente os logs do Apache. Para obter mais informações, consulte W3C_LOG_PARSE em Amazon Managed Service for Apache Flink SQL Reference.

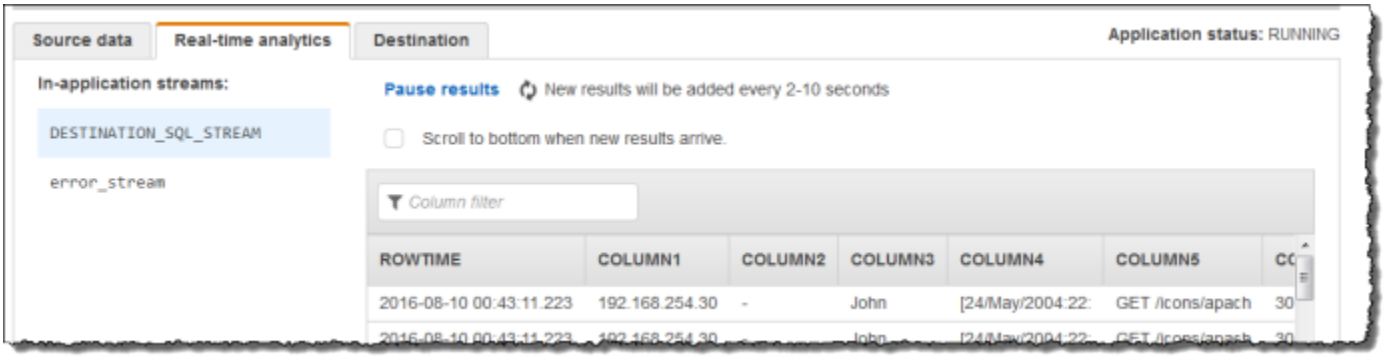
Neste exemplo, você grava registros de log em um Amazon Kinesis Data Streams. Os logs de exemplo são mostrados a seguir:

```
{"Log":"192.168.254.30 - John [24/May/2004:22:01:02 -0700] "GET /icons/apache_pba.gif
 HTTP/1.1" 304 0"}
{"Log":"192.168.254.30 - John [24/May/2004:22:01:03 -0700] "GET /icons/apache_pbb.gif
 HTTP/1.1" 304 0"}
{"Log":"192.168.254.30 - John [24/May/2004:22:01:04 -0700] "GET /icons/apache_pbc.gif
 HTTP/1.1" 304 0"}
...
```

Em seguida, você criará um aplicativo Kinesis Data Analytics no console com o fluxo de dados Kinesis como origem de streaming. O processo de descoberta lê registros de exemplo na origem de streaming e infere um esquema de aplicativo com uma coluna (log), como mostrado a seguir:



Em seguida, você usa o código do aplicativo com a função `W3C_LOG_PARSE` para analisar o log e cria outro stream no aplicativo com vários campos de log em colunas separadas, como mostrado a seguir:



Tópicos

- Etapa 1: Criar um fluxo de dados Kinesis

- Etapa 2: Criar o aplicativo Kinesis Data Analytics

Etapa 1: Criar um fluxo de dados Kinesis

Crie um streaming de dados do Amazon Kinesis Data Streams e preencha registros de log da seguinte forma:

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Selecione Data Streams (Fluxos de dados) no painel de navegação.
3. Escolha Create Kinesis stream (Criar fluxo do Kinesis) e crie um fluxo com um estilhaço. Para obter mais informações, consulte Criar um fluxo no Guia do desenvolvedor do Amazon Kinesis Data Streams.
4. Execute o seguinte código Python para preencher os registros de log de exemplo. Esse código simples grava continuamente o mesmo registro de log no fluxo.

```
import json
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {
        "log": "192.168.254.30 - John [24/May/2004:22:01:02 -0700] "
        '"GET /icons/apache_pb.gif HTTP/1.1" 304 0'
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )
```

```
if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Etapa 2: Criar o aplicativo Kinesis Data Analytics

Crie um aplicativo de análise de dados do Kinesis Data Analytics, da seguinte maneira:

1. Abra o Managed Service for Apache Flink console em https://console.aws.amazon.com/kinesisanalytics.
2. Escolha Create application (Criar aplicativo), digite um nome para o aplicativo e selecione Create application (Criar aplicativo).
3. Na página de detalhes do aplicativo, escolha Connect streaming data (Conectar dados de streaming).
4. Na página Connect to source (Conectar com a fonte), faça o seguinte:
   a. Escolha o fluxo criado na seção anterior.
   b. Escolha a opção para criar uma função do IAM.
   c. Escolha Discover schema (Descobrir esquema). Aguarde o console mostrar o esquema inferido e os registros de exemplo usados para inferir o esquema do fluxo do aplicativo criado. O esquema inferido tem apenas uma coluna.
   d. Escolha Save and continue.
5. Na página de detalhes de aplicativo, escolha Go to SQL editor (Ir para o editor de SQL). Para iniciar o aplicativo, escolha Yes, start application (Sim, iniciar o aplicativo) na caixa de diálogo exibida.
6. No editor SQL, escreva o código do aplicativo e verifique os resultados da seguinte forma:
   a. Copie o código de aplicativo a seguir e cole-o no editor.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
column1 VARCHAR(16),
column2 VARCHAR(16),
column3 VARCHAR(16),
column4 VARCHAR(16),
column5 VARCHAR(16),
column6 VARCHAR(16),
column7 VARCHAR(16));
```

```
CREATE OR REPLACE PUMP "myPUMP" AS
INSERT INTO "DESTINATION_SQL_STREAM"
        SELECT STREAM
            l.r.COLUMN1,
            l.r.COLUMN2,
            l.r.COLUMN3,
            l.r.COLUMN4,
            l.r.COLUMN5,
            l.r.COLUMN6,
            l.r.COLUMN7
        FROM (SELECT STREAM W3C_LOG_PARSE("log", 'COMMON')
              FROM "SOURCE_SQL_STREAM_001") AS l(r);
```

b. Escolha Save and run SQL. Na guia Real-time analytics (Análise em tempo real), você pode ver todos os fluxos de aplicativo criados pelo aplicativo e verificar os dados.

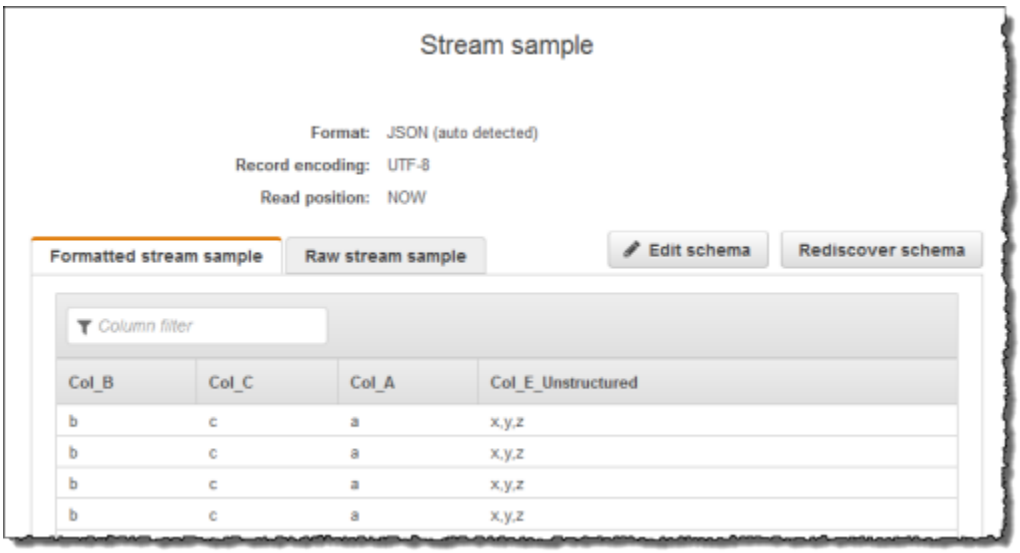
Exemplo: divisão de strings de caracteres em vários campos (função VARIABLE_COLUMN_LOG_PARSE)

Este exemplo usa a função `VARIABLE_COLUMN_LOG_PARSE` para manipular strings no Kinesis Data Analytics. O `VARIABLE_COLUMN_LOG_PARSE` divide uma string de entrada nos campos separados por um caractere delimitador ou uma string delimitadora. Para obter mais informações, consulte VARIABLE_COLUMN_LOG_PARSE em Amazon Managed Service for Apache Flink SQL Reference.

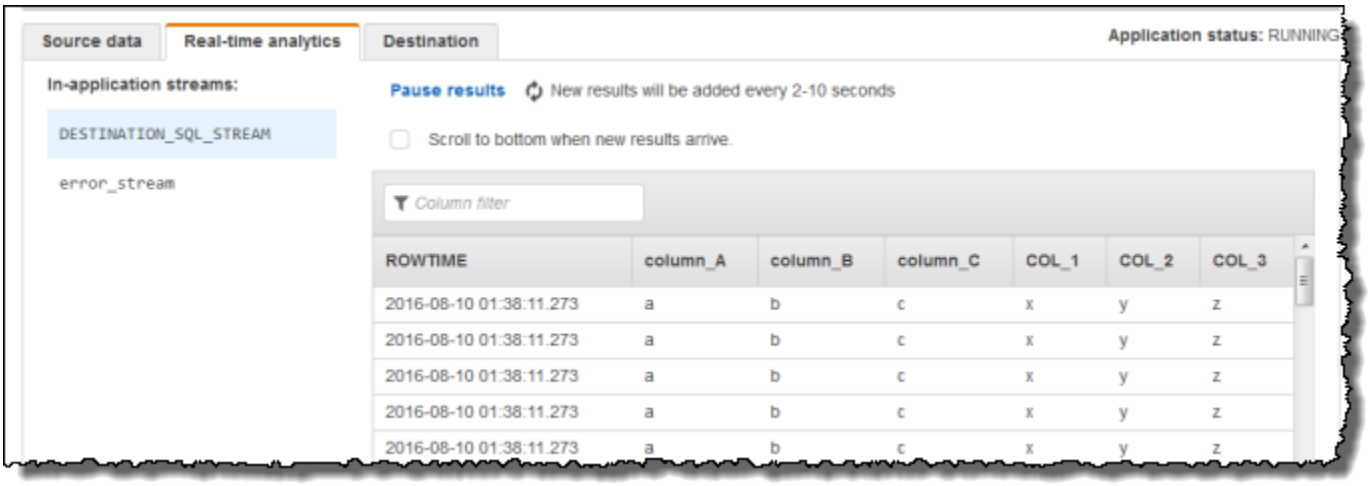
Neste exemplo, você grava registros semiestruturados em um fluxo de dados do Amazon Kinesis Data Streams. Esta é a aparência dos registros de exemplo:

```
{ "Col_A" : "string",
  "Col_B" : "string",
  "Col_C" : "string",
  "Col_D_Unstructured" : "value,value,value,value"}
{ "Col_A" : "string",
  "Col_B" : "string",
  "Col_C" : "string",
  "Col_D_Unstructured" : "value,value,value,value"}
```

Em seguida, você criará um aplicativo Kinesis Data Analytics no console usando o stream Kinesis como origem de streaming. O processo de descoberta lê registros de exemplo na origem de streaming e infere um esquema de aplicativo com quatro colunas, como mostrado a seguir:



Em seguida, você usa o código do aplicativo com a função `VARIABLE_COLUMN_LOG_PARSE` para analisar os valores separados por vírgulas e insere linhas normalizadas em outro stream no aplicativo, como mostrado a seguir:



Tópicos

- Etapa 1: Criar um fluxo de dados Kinesis
- Etapa 2: Criar o aplicativo Kinesis Data Analytics

Etapa 1: Criar um fluxo de dados Kinesis

Crie um fluxo de dados do Amazon Kinesis e preencha registros de log da seguinte forma:

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Selecione Data Streams (Fluxos de dados) no painel de navegação.
3. Escolha Create Kinesis stream (Criar fluxo do Kinesis) e crie um fluxo com um estilhaço. Para obter mais informações, consulte Criar um fluxo no Guia do desenvolvedor do Amazon Kinesis Data Streams.
4. Execute o seguinte código Python para preencher os registros de log de exemplo. Esse código simples grava continuamente o mesmo registro de log no fluxo.

```
import json
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {"Col_A": "a", "Col_B": "b", "Col_C": "c", "Col_E_Unstructured":
 "x,y,z"}


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Etapa 2: Criar o aplicativo Kinesis Data Analytics

Crie um aplicativo de análise de dados do Kinesis Data Analytics, da seguinte maneira:

1. Abra o Managed Service for Apache Flink console em https://console.aws.amazon.com/kinesisanalytics.
2. Escolha Create application (Criar aplicativo), digite um nome para o aplicativo e selecione Create application (Criar aplicativo).
3. Na página de detalhes do aplicativo, escolha Connect streaming data (Conectar dados de streaming).
4. Na página Connect to source (Conectar com a fonte), faça o seguinte:
   a. Escolha o fluxo criado na seção anterior.
   b. Escolha a opção para criar uma função do IAM.
   c. Escolha Discover schema (Descobrir esquema). Aguarde o console mostrar o esquema inferido e os registros de exemplo usados para inferir o esquema do fluxo do aplicativo criado. Observe que o esquema inferido tem apenas uma coluna.
   d. Escolha Save and continue.
5. Na página de detalhes de aplicativo, escolha Go to SQL editor (Ir para o editor de SQL). Para iniciar o aplicativo, escolha Yes, start application (Sim, iniciar o aplicativo) na caixa de diálogo exibida.
6. No editor SQL, escreva o código do aplicativo e verifique os resultados:
   a. Copie o código de aplicativo a seguir e cole-o no editor:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM"(
            "column_A" VARCHAR(16),
            "column_B" VARCHAR(16),
            "column_C" VARCHAR(16),
            "COL_1" VARCHAR(16),
            "COL_2" VARCHAR(16),
            "COL_3" VARCHAR(16));

CREATE OR REPLACE PUMP "SECOND_STREAM_PUMP" AS
INSERT INTO "DESTINATION_SQL_STREAM"
   SELECT STREAM  t."Col_A", t."Col_B", t."Col_C",
                  t.r."COL_1", t.r."COL_2", t.r."COL_3"
   FROM (SELECT STREAM
           "Col_A", "Col_B", "Col_C",
           VARIABLE_COLUMN_LOG_PARSE ("Col_E_Unstructured",
```

```
                                        'COL_1 TYPE VARCHAR(16), COL_2 TYPE
VARCHAR(16), COL_3 TYPE VARCHAR(16)',
                                        ',') AS r
        FROM "SOURCE_SQL_STREAM_001") as t;
```

b. Escolha Save and run SQL. Na guia Real-time analytics (Análise em tempo real), você pode ver todos os fluxos de aplicativo criados pelo aplicativo e verificar os dados.

Exemplo: Transformação de valores DateTime

O Amazon Kinesis Data Analytics suporta a conversão de colunas em time stamps. Por exemplo, use seu próprio time stamp como parte de uma cláusula `GROUP BY` como outra janela baseado em tempo, além da coluna `ROWTIME`. O Kinesis Data Analytics fornece operações e funções SQL para trabalhar com campos de data e hora.

- Operadores de data e hora: você pode executar operações aritméticas em datas, horas e tipos de dados de intervalo. Para obter mais informações, consulte Operadores de data, time stamp e intervalo em Amazon Managed Service for Apache Flink SQL Reference.

- Funções SQL: incluem o seguinte. Para obter mais informações, consulte Funções de data e hora em Amazon Managed Service for Apache Flink SQL Reference.
  - `EXTRACT()` - Extrai um campo a partir de uma data, hora, time stamp ou expressão de intervalo.
  - `CURRENT_TIME` – Retorna a hora em que a consulta é executada (UTC).
  - `CURRENT_DATE` – Retorna a data em que a consulta é executada (UTC).
  - `CURRENT_TIMESTAMP` – Retorna o time stamp em que a consulta é executada (UTC).
  - `LOCALTIME` – Retorna a hora atual em que a consulta é executada, conforme definido pelo ambiente no qual o Kinesis Data Analytics está sendo executado (UTC).
  - `LOCALTIMESTAMP` - Retorna o time stamp atual, conforme definido pelo ambiente no qual o Kinesis Data Analytics está sendo executado (UTC).

- Extensões SQL: incluem o seguinte. Para obter mais informações, consulte Funções de data e hora e Funções de conversão de data e hora em Amazon Managed Service for Apache Flink SQL Reference.

- `CURRENT_ROW_TIMESTAMP` - Retorna um novo time stamp para cada linha no fluxo.
- `TSDIFF` - Retorna a diferença de dois time stamps em milissegundos.
- `CHAR_TO_DATE` - Converte uma string em data.
- `CHAR_TO_TIME` – Converte uma string em hora.
- `CHAR_TO_TIMESTAMP` – Converte uma string em time stamp.
- `DATE_TO_CHAR` – Converte uma data em string.
- `TIME_TO_CHAR` – Converte uma hora em string.
- `TIMESTAMP_TO_CHAR` – Converte um time stamp em uma string.

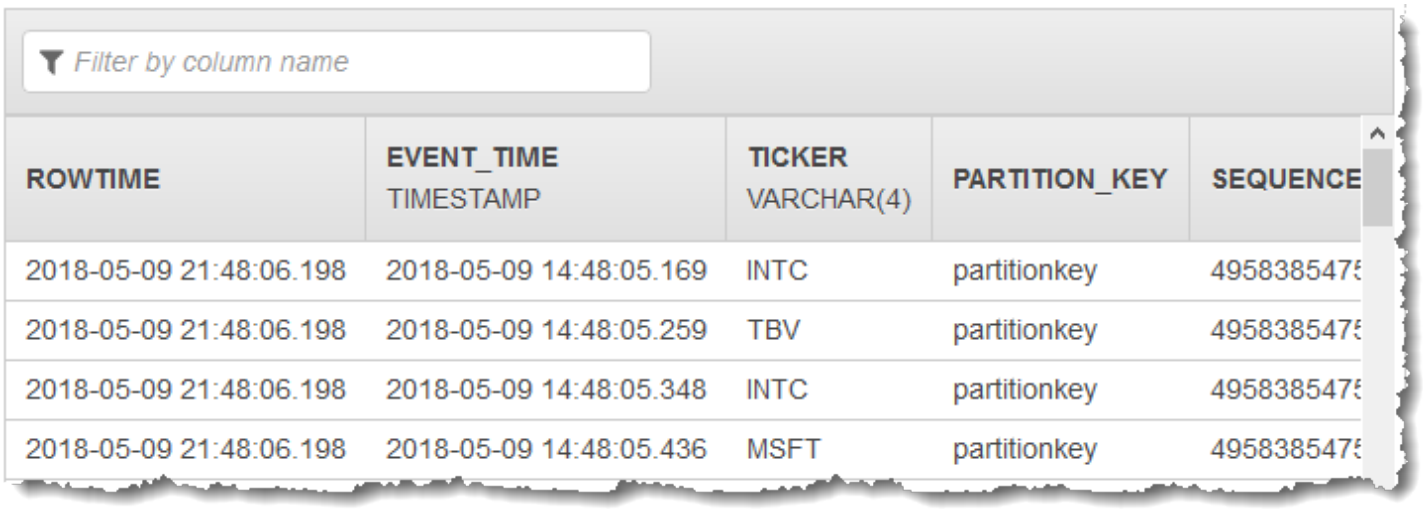
A maioria das funções SQL anteriores usam um formato para converter as colunas. O formato é flexível. Por exemplo, você pode especificar o formato `yyyy-MM-dd hh:mm:ss` para converter a string de entrada `2009-09-16 03:15:24` em time stamp. Para obter mais informações, consulte Char To Timestamp (Sys) em Amazon Managed Service for Apache Flink SQL Reference.

Exemplo: transformação de datas

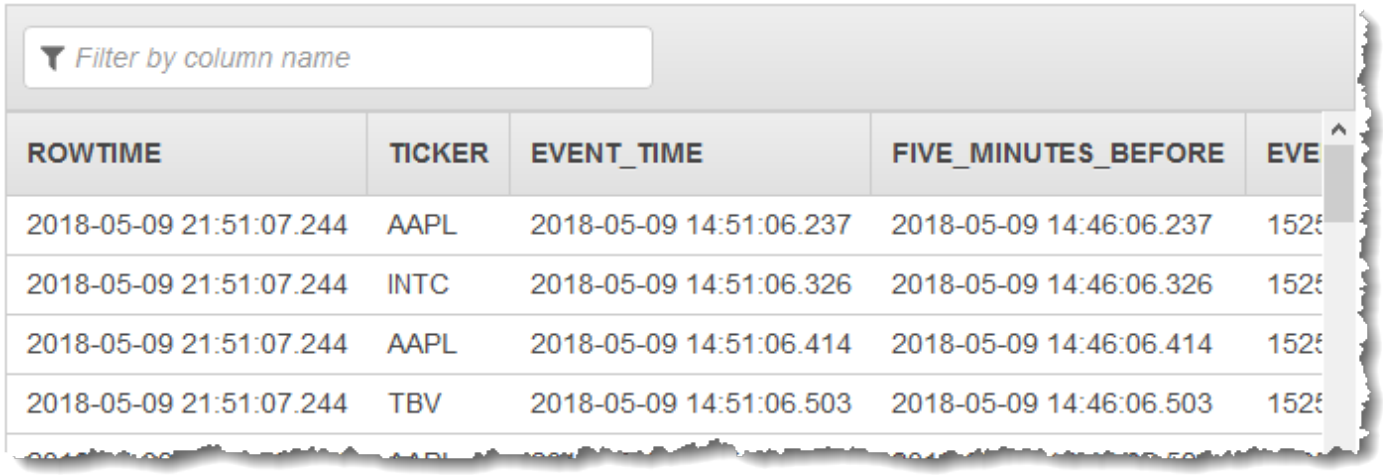
Neste exemplo, você grava os registros a seguir em um fluxo de dados do Amazon Kinesis.

```
{"EVENT_TIME": "2018-05-09T12:50:41.337510", "TICKER": "AAPL"}
{"EVENT_TIME": "2018-05-09T12:50:41.427227", "TICKER": "MSFT"}
{"EVENT_TIME": "2018-05-09T12:50:41.520549", "TICKER": "INTC"}
{"EVENT_TIME": "2018-05-09T12:50:41.610145", "TICKER": "MSFT"}
{"EVENT_TIME": "2018-05-09T12:50:41.704395", "TICKER": "AAPL"}
...
```

Em seguida, você criará um aplicativo Kinesis Data Analytics no console com o stream Kinesis como origem de streaming. O processo de descoberta lê registros de exemplo na origem de streaming e infere um esquema de aplicativo com duas colunas (`EVENT_TIME` e `TICKER`), como mostrado.



Em seguida, use o código do aplicativo com funções SQL para converter o campo de timestamp `EVENT_TIME` de várias maneiras. Insira dos dados resultantes em outro fluxo de aplicativo, como mostramos na captura de tela a seguir:



Etapa 1: Criar um fluxo de dados Kinesis

Crie um Amazon Kinesis Data Streams e preencha com hora do evento e os registros do marcador, da seguinte maneira:

1. Faça login AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Selecione Data Streams (Fluxos de dados) no painel de navegação.
3. Escolha Create Kinesis stream (Criar fluxo do Kinesis) e crie um fluxo com um estilhaço.

4. Execute o seguinte código Python para preencher o fluxo com dados de amostra. Esse código simples grava continuamente um registro com um símbolo de marcador aleatório e o timestamp atual no fluxo.

```
import datetime
import json
import random
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {
        "EVENT_TIME": datetime.datetime.now().isoformat(),
        "TICKER": random.choice(["AAPL", "AMZN", "MSFT", "INTC", "TBV"]),
        "PRICE": round(random.random() * 100, 2),
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Etapa 2: Criar o aplicativo Amazon Kinesis Data Analytics

Crie um aplicativo da seguinte maneira:

1. Abra o Managed Service for Apache Flink console em https://console.aws.amazon.com/kinesisanalytics.

2. Escolha Create application (Criar aplicativo), digite um nome para o aplicativo e selecione Create application (Criar aplicativo).
3. Na página de detalhes do aplicativo, escolha Connect streaming data (Conectar dados de streaming) para se conectar com a fonte.
4. Na página Connect to source (Conectar com a fonte), faça o seguinte:

   a. Escolha o stream criado na seção anterior.

   b. Escolha para criar uma função do IAM.

   c. Escolha Discover schema (Descobrir esquema). Aguarde o console mostrar o esquema inferido e os registros de exemplo usados para inferir o esquema do fluxo do aplicativo criado. O esquema inferido tem duas colunas.

   d. Escolha Edit Schema (Editar esquema). Mude Column type (Tipo de coluna) da coluna EVENT_TIME para `TIMESTAMP`.

   e. Escolha Save schema and update stream samples. Depois que o console salvar o esquema, escolha Exit (Sair).

   f. Escolha Save and continue.

5. Na página de detalhes de aplicativo, escolha Go to SQL editor (Ir para o editor de SQL). Para iniciar o aplicativo, escolha Yes, start application (Sim, iniciar o aplicativo) na caixa de diálogo exibida.
6. No editor SQL, escreva o código do aplicativo e verifique os resultados da seguinte forma:

   a. Copie o código de aplicativo a seguir e cole-o no editor.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    TICKER VARCHAR(4),
    event_time TIMESTAMP,
    five_minutes_before TIMESTAMP,
    event_unix_timestamp BIGINT,
    event_timestamp_as_char VARCHAR(50),
    event_second INTEGER);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"

SELECT STREAM
    TICKER,
    EVENT_TIME,
```

```
    EVENT_TIME - INTERVAL '5' MINUTE,
    UNIX_TIMESTAMP(EVENT_TIME),
    TIMESTAMP_TO_CHAR('yyyy-MM-dd hh:mm:ss', EVENT_TIME),
    EXTRACT(SECOND FROM EVENT_TIME)
FROM "SOURCE_SQL_STREAM_001"
```

b. Escolha Save and run SQL. Na guia Real-time analytics (Análise em tempo real), você pode ver todos os fluxos de aplicativo criados pelo aplicativo e verificar os dados.

Exemplo: transformação de vários tipos de dados

Um requisito comum nos aplicativos ETL (extração, transformação e carregamento) é o processamento de vários tipos de registro em uma origem de streaming. Você pode criar aplicativos Kinesis Data Analytics para processar esses tipos de origens de streaming. O processo é o seguinte:

1. Primeiro, mapeie a origem de streaming para um fluxo de entrada de aplicativo, semelhante a todos os outros aplicativos Kinesis Data Analytics.
2. Em seguida, no código do seu aplicativo, grave instruções SQL para recuperar linhas de tipos específicos do fluxo de entrada no aplicativo. Em seguida, insira-as em fluxos separados no aplicativo. (Você pode criar fluxos adicionais no aplicativo no código do seu aplicativo.)

Neste exercício, há uma origem de streaming que recebe registros de dois tipos (`Order` e `Trade`). Esses são pedidos de ações e suas negociações correspondentes. Para cada pedido, pode haver zero ou mais negociações. Veja a seguir registros de exemplo de cada tipo:

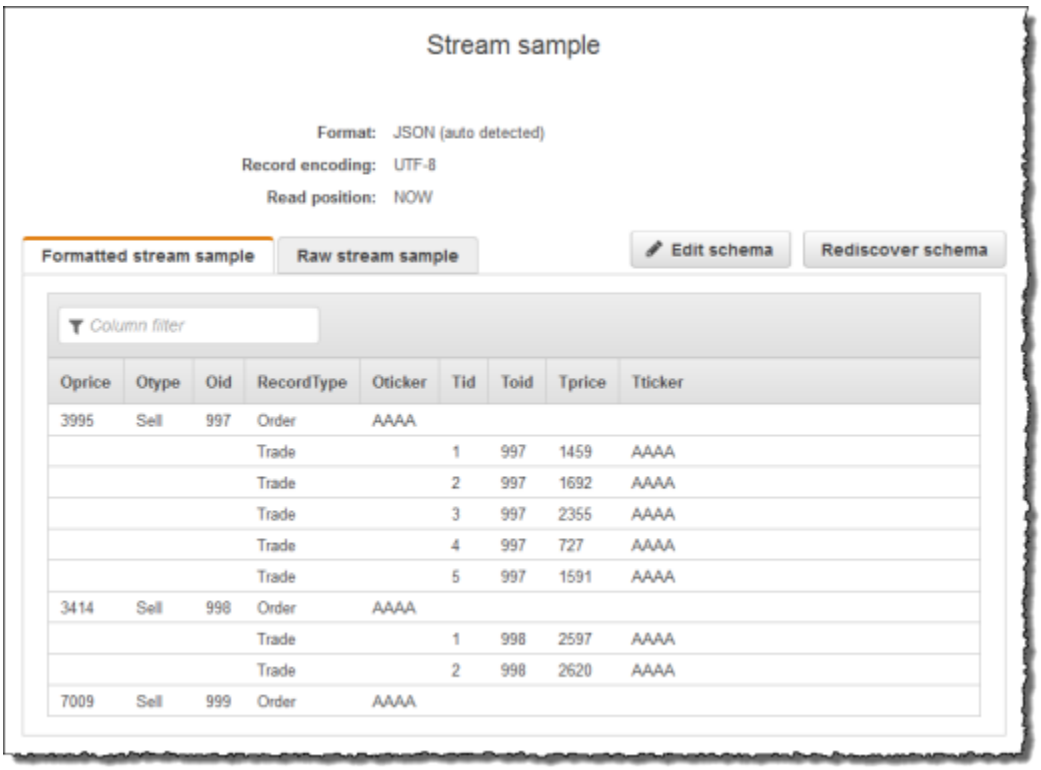
Order record

```
{"RecordType": "Order", "Oprice": 9047, "Otype": "Sell", "Oid": 3811, "Oticker":
 "AAAA"}
```

Trade record

```
{"RecordType": "Trade", "Tid": 1, "Toid": 3812, "Tprice": 2089, "Tticker": "BBBB"}
```

Quando você cria um aplicativo usando o AWS Management Console, o console exibe o seguinte esquema inferido para o fluxo de entrada no aplicativo criado. Por padrão, o console nomeia esse fluxo de aplicativo como `SOURCE_SQL_STREAM_001`.



Quando você salvar a configuração, o Amazon Kinesis Data Analytics lerá continuamente dados na origem de streaming e inserirá linhas no stream no aplicativo. Agora você pode fazer a análise de dados no stream no aplicativo.

Neste exemplo, primeiro você cria dois fluxos de aplicativo adicionais no código do aplicativo, que são `Order_Stream` e `Trade_Stream`. Depois, filtra as linhas do fluxo `SOURCE_SQL_STREAM_001` com base no tipo de registro e as insere nos fluxos recém-criados usando bombas. Para obter informações sobre esse padrão de codificação, consulte Código do aplicativo.

1. Filtragem de linhas de pedidos e negociações em fluxos de aplicativo separados:

   a. Filtre os registros de pedidos no `SOURCE_SQL_STREAM_001` e salve os pedidos no `Order_Stream`.

```
--Create Order_Stream.
CREATE OR REPLACE STREAM "Order_Stream"
           (
            order_id     integer,
            order_type   varchar(10),
            ticker       varchar(4),
            order_price  DOUBLE,
            record_type  varchar(10)
```

```
            );

CREATE OR REPLACE PUMP "Order_Pump" AS
   INSERT INTO "Order_Stream"
      SELECT STREAM oid, otype,oticker, oprice, recordtype
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  recordtype = 'Order';
```

b. Filtre os registros de negociações no `SOURCE_SQL_STREAM_001` e salve os pedidos no `Trade_Stream`.

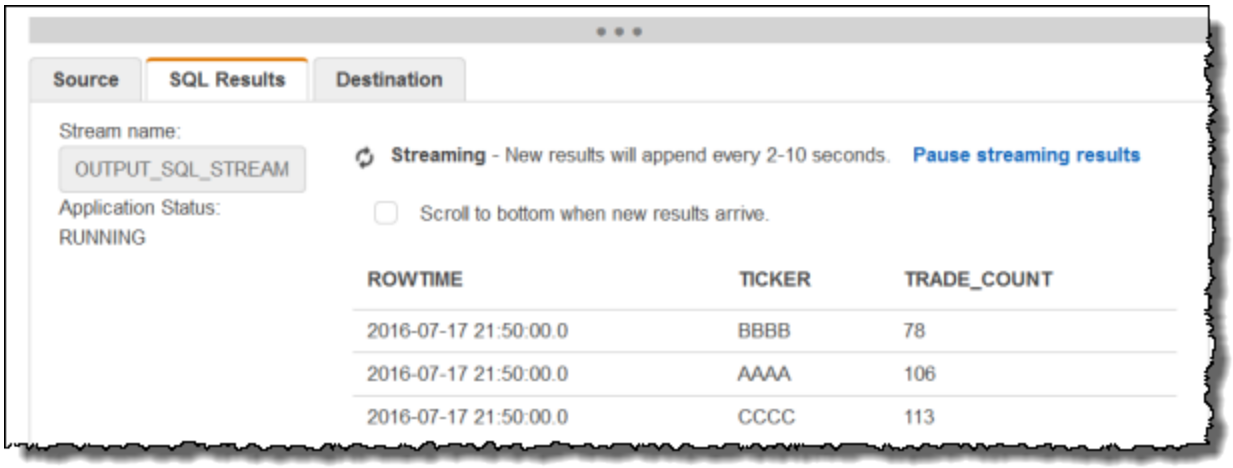
```
--Create Trade_Stream.
CREATE OR REPLACE STREAM "Trade_Stream"
           (trade_id     integer,
            order_id     integer,
            trade_price  DOUBLE,
            ticker       varchar(4),
            record_type  varchar(10)
            );

CREATE OR REPLACE PUMP "Trade_Pump" AS
   INSERT INTO "Trade_Stream"
      SELECT STREAM tid, toid, tprice, tticker, recordtype
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  recordtype = 'Trade';
```

2. Agora você pode executar análises adicionais nesses fluxos. Neste exemplo, conte o número de negociações pelo índice em uma janela em cascata de um minuto e salve os resultados em outro fluxo, `DESTINATION_SQL_STREAM`.

```
--do some analytics on the Trade_Stream and Order_Stream.
-- To see results in console you must write to OPUT_SQL_STREAM.

CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
            ticker  varchar(4),
            trade_count   integer
            );

CREATE OR REPLACE PUMP "Output_Pump" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM ticker, count(*) as trade_count
      FROM   "Trade_Stream"
      GROUP BY ticker,
```

```
                FLOOR("Trade_Stream".ROWTIME TO MINUTE);
```

Você verá o resultado a seguir:



Tópicos

- Etapa 1: Preparar os dados
- Etapa 2: Criar o aplicativo

Próxima etapa

Etapa 1: Preparar os dados

Etapa 1: Preparar os dados

Nesta seção, você cria um fluxo de dados do Kinesis e, em seguida, preenche os registros de pedidos e negociações no fluxo. Esta é a origem de streaming do aplicativo criado na próxima etapa.

Tópicos

- Etapa 1.1: Criar uma origem de streaming
- Etapa 1.2: Preencher a origem de streaming

Etapa 1.1: Criar uma origem de streaming

Você pode criar esse fluxo de dados do Kinesis usando o console ou a AWS CLI. O exemplo assume `OrdersAndTradesStream` como nome do fluxo.

- Usando o console — Faça login AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis. Escolha Data Streams e crie um fluxo com um estilhaço. Para obter mais informações, consulte Criar um stream no Guia do desenvolvedor do Amazon Kinesis Data Streams.
- Usando o AWS CLI — Use o seguinte `create-stream` AWS CLI comando do Kinesis para criar o stream:

```
$ aws kinesis create-stream \
--stream-name OrdersAndTradesStream \
--shard-count 1 \
--region us-east-1 \
--profile adminuser
```

Etapa 1.2: Preencher a origem de streaming

Execute o seguinte script Python para preencher os registros de exemplo no `OrdersAndTradesStream`. Se você tiver criado o fluxo com outro nome, atualize o código Python adequadamente.

1. Instale o Python e o `pip`.

   Para obter informações sobre como instalar o Python, consulte o site do Python.

   Você pode instalar dependências usando o pip. Para obter informações sobre como instalar o pip, consulte Installation no site do pip.

2. Execute o código do Python a seguir. O comando `put-record` no código grava os registros JSON no fluxo.

```
import json
import random
import boto3

STREAM_NAME = "OrdersAndTradesStream"
PARTITION_KEY = "partition_key"


def get_order(order_id, ticker):
    return {
        "RecordType": "Order",
```

```
        "Oid": order_id,
        "Oticker": ticker,
        "Oprice": random.randint(500, 10000),
        "Otype": "Sell",
    }


def get_trade(order_id, trade_id, ticker):
    return {
        "RecordType": "Trade",
        "Tid": trade_id,
        "Toid": order_id,
        "Tticker": ticker,
        "Tprice": random.randint(0, 3000),
    }


def generate(stream_name, kinesis_client):
    order_id = 1
    while True:
        ticker = random.choice(["AAAA", "BBBB", "CCCC"])
        order = get_order(order_id, ticker)
        print(order)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(order),
 PartitionKey=PARTITION_KEY
        )
        for trade_id in range(1, random.randint(0, 6)):
            trade = get_trade(order_id, trade_id, ticker)
            print(trade)
            kinesis_client.put_record(
                StreamName=stream_name,
                Data=json.dumps(trade),
                PartitionKey=PARTITION_KEY,
            )
        order_id += 1


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Próxima etapa

Etapa 2: Criar o aplicativo

Etapa 2: Criar o aplicativo

Nesta seção, você cria um aplicativo Kinesis Data Analytics. Em seguida, você atualiza o aplicativo adicionando a configuração de entrada que mapeia a origem de streaming criada na seção anterior para um fluxo de entrada de aplicativo.

1. Abra o Managed Service for Apache Flink console em https://console.aws.amazon.com/kinesisanalytics.
2. Escolha Criar aplicação. Esse exemplo usa o nome de aplicativo **`ProcessMultipleRecordTypes`**.
3. Na página de detalhes do aplicativo, escolha Connect streaming data (Conectar dados de streaming) para se conectar com a fonte.
4. Na página Connect to source (Conectar com a fonte), faça o seguinte:
   a. Selecione o fluxo que você criou em Etapa 1: Preparar os dados.
   b. Escolha para criar uma função do IAM.
   c. Aguarde o console mostrar o esquema inferido e os registros de exemplos usados para inferir o esquema do stream do aplicativo criado.
   d. Escolha Save and continue.
5. No hub de aplicativo, escolha Go to SQL editor. Para iniciar o aplicativo, escolha Yes, start application (Sim, iniciar o aplicativo) na caixa de diálogo exibida.
6. No editor SQL, escreva o código do aplicativo e verifique os resultados:
   a. Copie o código de aplicativo a seguir e cole-o no editor.

```
--Create Order_Stream.
CREATE OR REPLACE STREAM "Order_Stream"
           (
            "order_id"     integer,
            "order_type"   varchar(10),
            "ticker"       varchar(4),
            "order_price"  DOUBLE,
            "record_type"  varchar(10)
            );

CREATE OR REPLACE PUMP "Order_Pump" AS
```

```
   INSERT INTO "Order_Stream"
      SELECT STREAM "Oid", "Otype","Oticker", "Oprice", "RecordType"
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  "RecordType" = 'Order';
--***************************************
--Create Trade_Stream.
CREATE OR REPLACE STREAM "Trade_Stream"
          ("trade_id"     integer,
           "order_id"     integer,
           "trade_price"  DOUBLE,
           "ticker"       varchar(4),
           "record_type"  varchar(10)
           );

CREATE OR REPLACE PUMP "Trade_Pump" AS
   INSERT INTO "Trade_Stream"
      SELECT STREAM "Tid", "Toid", "Tprice", "Tticker", "RecordType"
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  "RecordType" = 'Trade';
--*************************************************************
--do some analytics on the Trade_Stream and Order_Stream.
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
           "ticker"  varchar(4),
           "trade_count"   integer
           );

CREATE OR REPLACE PUMP "Output_Pump" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM "ticker", count(*) as trade_count
      FROM   "Trade_Stream"
      GROUP BY "ticker",
               FLOOR("Trade_Stream".ROWTIME TO MINUTE);
```

b. Escolha Save and run SQL. Escolha a guia Real-time analytics (Análise em tempo real) para ver todos os fluxos de aplicativo criados pelo aplicativo e verifique os dados.

Próxima etapa

Você pode configurar a saída do aplicativo para manter os resultados em um destino externo, como outro stream do Kinesis ou um stream de entrega de dados do Firehose.

Exemplos: janelas e agregação

Esta seção fornece exemplos de aplicativos do Amazon Kinesis Data Analytics que usam consultas em janela e de agregação. (Para ter mais informações, consulte Consultas em janelas.) Cada exemplo fornece instruções passo a passo e código de exemplo para configurar o aplicativo Kinesis Data Analytics.

Tópicos

Exemplo: janela de escalonamento

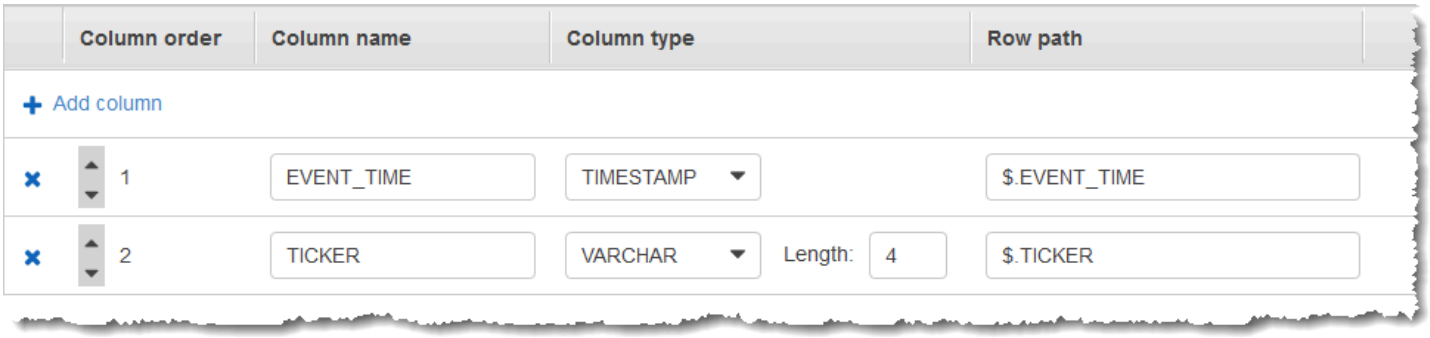
Quando uma consulta em janela processa janelas separadas para cada chave de partição exclusiva, a partir da chegada dos dados com a chave correspondente, a janela é chamada de janela de escalonamento. Para obter mais detalhes, consulte Janelas de escalonamento. Esse exemplo do Amazon Kinesis Data Analytics usa as colunas EVENT_TIME e TICKER para criar janelas de escalonamento. O stream de origem contém grupos de seis registros com valores EVENT_TIME e TICKER idênticos, que chegam em um período de um minuto, mas não necessariamente com o mesmo valor de minutos (por exemplo, `18:41:xx`).

Neste exemplo, você grava os registros a seguir em um fluxo de dados do Kinesis nos seguintes tempos. O script não grava os horários no fluxo, mas o horário em que o registro é consumido pelo aplicativo é gravado no campo `ROWTIME`:

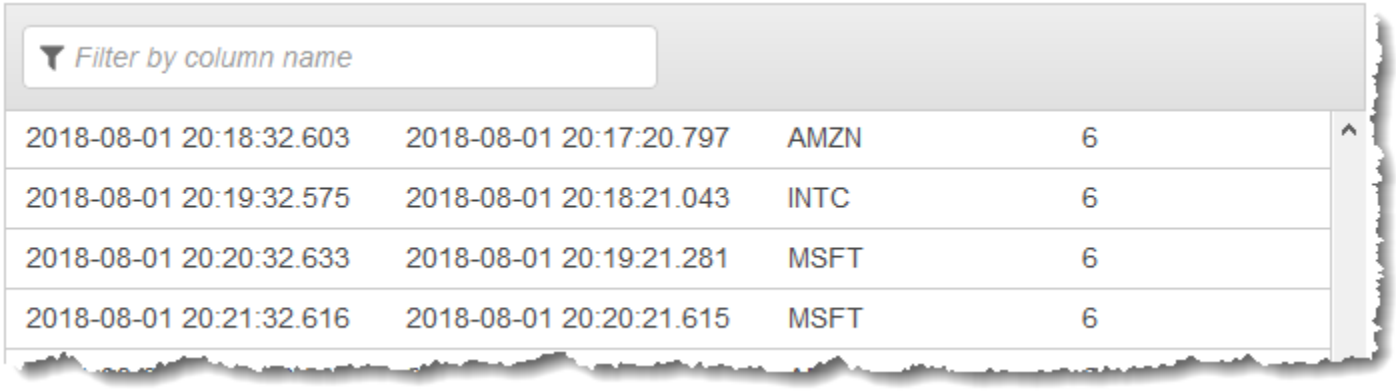
```
{"EVENT_TIME": "2018-08-01T20:17:20.797945", "TICKER": "AMZN"}   20:17:30
{"EVENT_TIME": "2018-08-01T20:17:20.797945", "TICKER": "AMZN"}   20:17:40
{"EVENT_TIME": "2018-08-01T20:17:20.797945", "TICKER": "AMZN"}   20:17:50
{"EVENT_TIME": "2018-08-01T20:17:20.797945", "TICKER": "AMZN"}   20:18:00
{"EVENT_TIME": "2018-08-01T20:17:20.797945", "TICKER": "AMZN"}   20:18:10
{"EVENT_TIME": "2018-08-01T20:17:20.797945", "TICKER": "AMZN"}   20:18:21
{"EVENT_TIME": "2018-08-01T20:18:21.043084", "TICKER": "INTC"}   20:18:31
{"EVENT_TIME": "2018-08-01T20:18:21.043084", "TICKER": "INTC"}   20:18:41
{"EVENT_TIME": "2018-08-01T20:18:21.043084", "TICKER": "INTC"}   20:18:51
```

```
{"EVENT_TIME": "2018-08-01T20:18:21.043084", "TICKER": "INTC"}   20:19:01
{"EVENT_TIME": "2018-08-01T20:18:21.043084", "TICKER": "INTC"}   20:19:11
{"EVENT_TIME": "2018-08-01T20:18:21.043084", "TICKER": "INTC"}   20:19:21
...
```

Em seguida, crie um aplicativo do Kinesis Data Analytics no AWS Management Console, com o streaming de dados do Kinesis como a origem de streaming. O processo de descoberta lê os registros de exemplo na origem de streaming e infere um esquema no aplicativo com duas colunas (`EVENT_TIME` e `TICKER`) conforme mostrado a seguir.



Você usa o código do aplicativo com a função `COUNT` para criar uma agregação em janelas dos dados. Em seguida, insira os dados resultantes em outro stream no aplicativo, conforme mostrado na captura de tela a seguir:



No procedimento a seguir, você cria um aplicativo do Kinesis Data Analytics que agrega valores no streaming de entrada em uma janela de escalonamento com base em EVENT_TIME e TICKER.

Tópicos

- Etapa 1: Criar um Kinesis Data Stream
- Etapa 2: Criar um aplicativo Kinesis Data Analytics

Etapa 1: Criar um Kinesis Data Stream

Crie um fluxo de dados do Amazon Kinesis e preencha registros da seguinte forma:

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Selecione Data Streams (Streams de dados) no painel de navegação.
3. Selecione Create Kinesis stream (Criar stream do Kinesis) e crie um stream com um estilhaço. Para obter mais informações, consulte Criar um stream no Guia do desenvolvedor de Amazon Kinesis Data Streams.
4. Para gravar registros em um streaming de dados do Kinesis em um ambiente de produção, recomendamos usar o Kinesis Producer Library ou a API do Kinesis Data Streams. Para simplificar, este exemplo usa o script Python a seguir para gerar registros. Execute o código para preencher os registros de marcador de exemplo. Esse código simples grava continuamente um grupo de seis registros com o mesmo símbolo de índice e `EVENT_TIME` aleatórios no stream durante o período de um minuto. Mantenha o script em execução para que você possa gerar o esquema do aplicativo em uma etapa posterior.

```
import datetime
import json
import random
import time
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    event_time = datetime.datetime.utcnow() - datetime.timedelta(seconds=10)
    return {
        "EVENT_TIME": event_time.isoformat(),
        "TICKER": random.choice(["AAPL", "AMZN", "MSFT", "INTC", "TBV"]),
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        # Send six records, ten seconds apart, with the same event time and ticker
```

```
        for _ in range(6):
            print(data)
            kinesis_client.put_record(
                StreamName=stream_name,
                Data=json.dumps(data),
                PartitionKey="partitionkey",
            )
            time.sleep(10)


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Etapa 2: Criar um aplicativo Kinesis Data Analytics

Crie um aplicativo Kinesis Data Analytics, da seguinte maneira:

1. Abra o Managed Service for Apache Flink console em https://console.aws.amazon.com/kinesisanalytics.
2. Escolha Create application (Criar aplicativo), digite um nome para o aplicativo e selecione Create application (Criar aplicativo).
3. Na página de detalhes do aplicativo, escolha Connect streaming data (Conectar dados de streaming) para se conectar com a fonte.
4. Na página Connect to source (Conectar com a fonte), faça o seguinte:

    a. Escolha o stream criado na seção anterior.

    b. Selecione Discover schema (Descobrir esquema). Aguarde o console mostrar o esquema inferido e os registros de exemplos usados para inferir o esquema do stream do aplicativo criado. O esquema inferido tem duas colunas.

    c. Escolha Edit Schema (Editar esquema). Mude Column type (Tipo de coluna) da coluna EVENT_TIME para `TIMESTAMP`.

    d. Escolha Save schema and update stream samples. Depois que o console salvar o esquema, escolha Exit (Sair).

    e. Escolha Save and continue.

5. Na página de detalhes de aplicativo, escolha Go to SQL editor (Ir para o editor de SQL). Para iniciar o aplicativo, escolha Yes, start application (Sim, iniciar o aplicativo) na caixa de diálogo exibida.
6. No editor SQL, escreva o código do aplicativo e verifique os resultados da seguinte forma:

   a. Copie o código de aplicativo a seguir e cole-o no editor.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    event_time TIMESTAMP,
    ticker_symbol    VARCHAR(4),
    ticker_count     INTEGER);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS
  INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM
        EVENT_TIME,
        TICKER,
        COUNT(TICKER) AS ticker_count
    FROM "SOURCE_SQL_STREAM_001"
    WINDOWED BY STAGGER (
            PARTITION BY TICKER, EVENT_TIME RANGE INTERVAL '1' MINUTE);
```

   b. Escolha Save and run SQL.

      Na guia Real-time analytics (Análise em tempo real), você pode ver todos os fluxos de aplicativo criados pelo aplicativo e verificar os dados.

Exemplo: janela em cascata usando ROWTIME

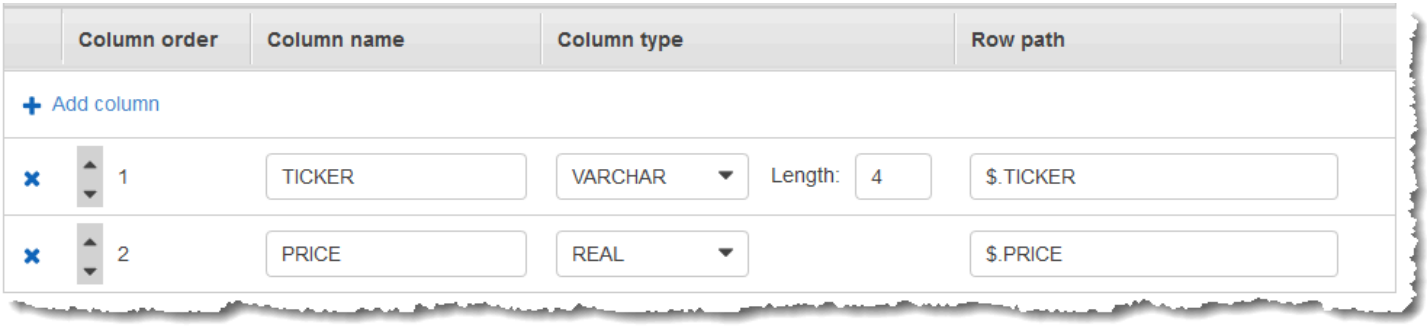
Quando uma consulta em janela processa cada janela de forma não sobreposta, é chamada de janela em cascata. Para obter mais detalhes, consulte Janelas em cascata (Agregações usando GROUP BY). Este exemplo do Amazon Kinesis Data Analytics usa a coluna `ROWTIME` para criar janelas em cascata. A coluna `ROWTIME` representa o momento em que o registro foi lido pelo aplicativo.

Neste exemplo, você grava os registros a seguir em um streaming de dados do Kinesis.

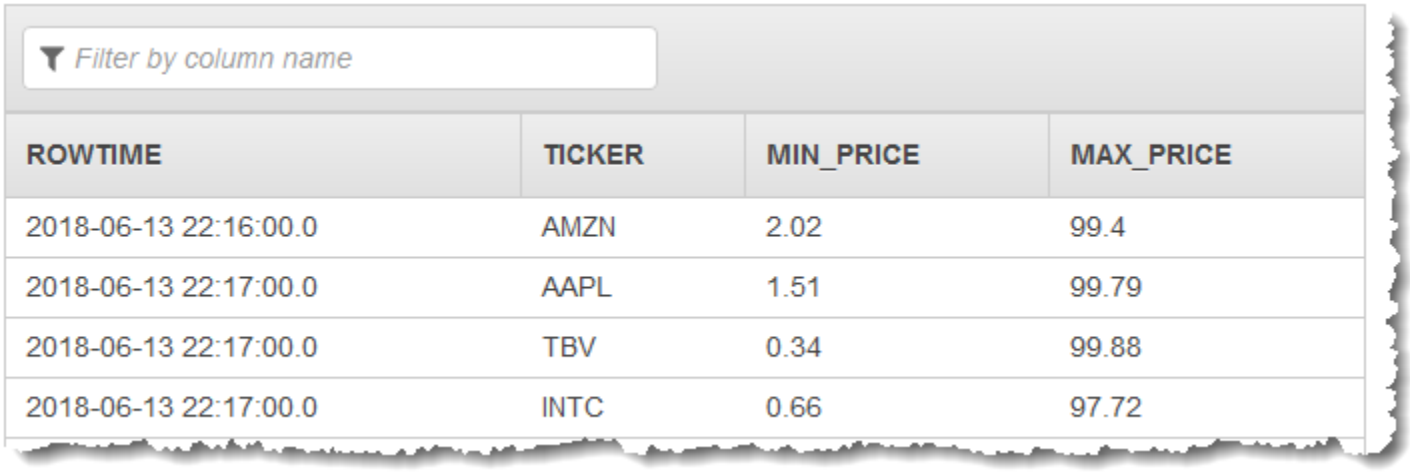
```
{"TICKER": "TBV", "PRICE": 33.11}
{"TICKER": "INTC", "PRICE": 62.04}
{"TICKER": "MSFT", "PRICE": 40.97}
{"TICKER": "AMZN", "PRICE": 27.9}
```

```
...
```

Em seguida, crie um aplicativo do Kinesis Data Analytics no AWS Management Console, com o streaming de dados do Kinesis como a origem de streaming. O processo de descoberta lê os registros de exemplo na origem de streaming e infere um esquema no aplicativo com duas colunas (`TICKER` e `PRICE`) conforme mostrado a seguir.



Você usa o código do aplicativo com as funções `MIN` e `MAX` para criar uma agregação em janela dos dados. Em seguida, insira os dados resultantes em outro stream no aplicativo, conforme mostrado na captura de tela a seguir:



No procedimento a seguir, você cria um aplicativo do Kinesis Data Analytics que agrega valores no stream de entrada em uma janela em cascata com base em ROWTIME.

Tópicos

- Etapa 1: Criar um Kinesis Data Stream
- Etapa 2: Criar um aplicativo Kinesis Data Analytics

Etapa 1: Criar um Kinesis Data Stream

Crie um fluxo de dados do Amazon Kinesis e preencha registros da seguinte forma:

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Selecione Data Streams (Streams de dados) no painel de navegação.
3. Selecione Create Kinesis stream (Criar stream do Kinesis) e crie um stream com um estilhaço. Para obter mais informações, consulte Criar um stream no Guia do desenvolvedor de Amazon Kinesis Data Streams.
4. Para gravar registros em um streaming de dados do Kinesis em um ambiente de produção, recomendamos usar o Kinesis Client Library ou a API do Kinesis Data Streams. Para simplificar, este exemplo usa o script Python a seguir para gerar registros. Execute o código para preencher os registros de marcador de exemplo. Esse código simples grava continuamente um registro de marcador aleatório no stream. Mantenha o script em execução para que você possa gerar o esquema do aplicativo em uma etapa posterior.

```
import datetime
import json
import random
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {
        "EVENT_TIME": datetime.datetime.now().isoformat(),
        "TICKER": random.choice(["AAPL", "AMZN", "MSFT", "INTC", "TBV"]),
        "PRICE": round(random.random() * 100, 2),
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
```

```
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Etapa 2: Criar um aplicativo Kinesis Data Analytics

Crie um aplicativo Kinesis Data Analytics, da seguinte maneira:

1. Abra o Managed Service for Apache Flink console em https://console.aws.amazon.com/kinesisanalytics.
2. Escolha Create application (Criar aplicativo), insira um nome para o aplicativo e selecione Create application (Criar aplicativo).
3. Na página de detalhes do aplicativo, escolha Connect streaming data (Conectar dados de streaming) para se conectar com a fonte.
4. Na página Connect to source (Conectar com a fonte), faça o seguinte:

   a. Escolha o stream criado na seção anterior.

   b. Selecione Discover schema (Descobrir esquema). Aguarde o console mostrar o esquema inferido e os registros de exemplos usados para inferir o esquema do stream do aplicativo criado. O esquema inferido tem duas colunas.

   c. Escolha Save schema and update stream samples. Depois que o console salvar o esquema, escolha Exit (Sair).

   d. Escolha Save and continue.
5. Na página de detalhes de aplicativo, escolha Go to SQL editor (Ir para o editor de SQL). Para iniciar o aplicativo, escolha Yes, start application (Sim, iniciar o aplicativo) na caixa de diálogo exibida.
6. No editor SQL, escreva o código do aplicativo e verifique os resultados da seguinte forma:

   a. Copie o código de aplicativo a seguir e cole-o no editor.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (TICKER VARCHAR(4), MIN_PRICE
 REAL, MAX_PRICE REAL);
```

```
CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM TICKER, MIN(PRICE), MAX(PRICE)
        FROM "SOURCE_SQL_STREAM_001"
        GROUP BY TICKER,
            STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND);
```

b. Escolha Save and run SQL.

   Na guia Real-time analytics (Análise em tempo real), você pode ver todos os fluxos de aplicativo criados pelo aplicativo e verificar os dados.

Exemplo: janela em cascata usando um time stamp do evento

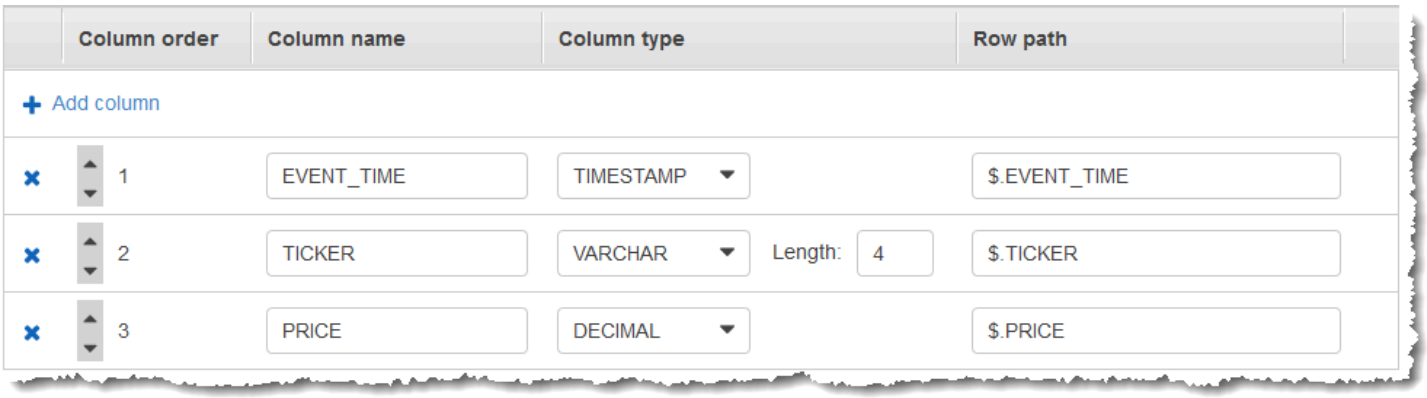
Quando uma consulta em janela processa cada janela de forma não sobreposta, é chamada de janela em cascata. Para obter mais detalhes, consulte Janelas em cascata (Agregações usando GROUP BY). Este exemplo do Amazon Kinesis Data Analytics demonstra uma janela em cascata que usa um time stamp do evento, que é um time stamp criado pelo usuário que está incluído no streaming de dados. Ele usa essa abordagem, em vez de apenas usar ROWTIME, que é um time stamp criado pelo Kinesis Data Analytics quando o aplicativo recebe o registro. Use um time stamp do evento no streaming de dados se quiser criar uma agregação com base em quando um evento ocorreu, em vez de quando ele foi recebido pelo aplicativo. Neste exemplo, o valor de `ROWTIME` aciona a agregação a cada minuto, e os registros são agregados pelo `ROWTIME` e pelo horário do evento incluído.

Neste exemplo, você grava os registros a seguir em um stream do Amazon Kinesis. O valor `EVENT_TIME` é definido como 5 segundos no passado, para simular o atraso no processamento e na transmissão que pode criar um atraso de quando o evento ocorreu para quando o registro foi recebido no Kinesis Data Analytics.

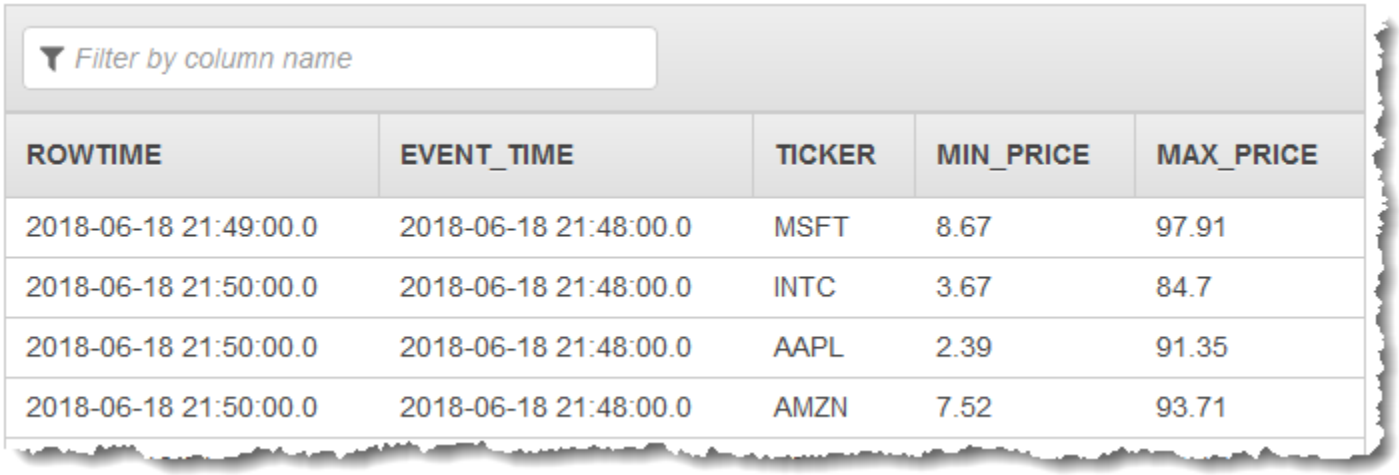
```
{"EVENT_TIME": "2018-06-13T14:11:05.766191", "TICKER": "TBV", "PRICE": 43.65}
{"EVENT_TIME": "2018-06-13T14:11:05.848967", "TICKER": "AMZN", "PRICE": 35.61}
{"EVENT_TIME": "2018-06-13T14:11:05.931871", "TICKER": "MSFT", "PRICE": 73.48}
{"EVENT_TIME": "2018-06-13T14:11:06.014845", "TICKER": "AMZN", "PRICE": 18.64}
...
```

Em seguida, crie um aplicativo do Kinesis Data Analytics no AWS Management Console, com o streaming de dados do Kinesis como a origem de streaming. O processo de descoberta lê os

registros de exemplo na origem de streaming e infere um esquema no aplicativo com três colunas (`EVENT_TIME`, `TICKER` e `PRICE`) conforme mostrado a seguir.



Você usa o código do aplicativo com as funções `MIN` e `MAX` para criar uma agregação em janela dos dados. Em seguida, insira os dados resultantes em outro stream no aplicativo, conforme mostrado na captura de tela a seguir:



No procedimento a seguir, você cria um aplicativo do Kinesis Data Analytics que agrega valores no stream de entrada em uma janela em cascata com base em um horário de evento.

Tópicos

- Etapa 1: Criar um Kinesis Data Stream
- Etapa 2: Criar um aplicativo Kinesis Data Analytics

Etapa 1: Criar um Kinesis Data Stream

Crie um fluxo de dados do Amazon Kinesis e preencha registros da seguinte forma:

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Selecione Data Streams (Streams de dados) no painel de navegação.
3. Selecione Create Kinesis stream (Criar stream do Kinesis) e crie um stream com um estilhaço. Para obter mais informações, consulte Criar um stream no Guia do desenvolvedor de Amazon Kinesis Data Streams.
4. Para gravar registros em um streaming de dados do Kinesis em um ambiente de produção, recomendamos usar o Kinesis Client Library ou a API do Kinesis Data Streams. Para simplificar, este exemplo usa o script Python a seguir para gerar registros. Execute o código para preencher os registros de marcador de exemplo. Esse código simples grava continuamente um registro de marcador aleatório no stream. Mantenha o script em execução para que você possa gerar o esquema do aplicativo em uma etapa posterior.

```
import datetime
import json
import random
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {
        "EVENT_TIME": datetime.datetime.now().isoformat(),
        "TICKER": random.choice(["AAPL", "AMZN", "MSFT", "INTC", "TBV"]),
        "PRICE": round(random.random() * 100, 2),
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
```

```
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Etapa 2: Criar um aplicativo Kinesis Data Analytics

Crie um aplicativo Kinesis Data Analytics, da seguinte maneira:

1. Abra o Managed Service for Apache Flink console em https://console.aws.amazon.com/kinesisanalytics.
2. Escolha Create application (Criar aplicativo), insira um nome para o aplicativo e selecione Create application (Criar aplicativo).
3. Na página de detalhes do aplicativo, escolha Connect streaming data (Conectar dados de streaming) para se conectar com a fonte.
4. Na página Connect to source (Conectar com a fonte), faça o seguinte:

    a. Escolha o stream criado na seção anterior.

    b. Selecione Discover schema (Descobrir esquema). Aguarde o console mostrar o esquema inferido e os registros de exemplos usados para inferir o esquema do stream do aplicativo criado. O esquema inferido tem três colunas.

    c. Escolha Edit Schema (Editar esquema). Mude Column type (Tipo de coluna) da coluna EVENT_TIME para `TIMESTAMP`.

    d. Escolha Save schema and update stream samples. Depois que o console salvar o esquema, escolha Exit (Sair).

    e. Escolha Save and continue.
5. Na página de detalhes de aplicativo, escolha Go to SQL editor (Ir para o editor de SQL). Para iniciar o aplicativo, escolha Yes, start application (Sim, iniciar o aplicativo) na caixa de diálogo exibida.
6. No editor SQL, escreva o código do aplicativo e verifique os resultados da seguinte forma:

    a. Copie o código de aplicativo a seguir e cole-o no editor.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (EVENT_TIME timestamp, TICKER
 VARCHAR(4), min_price REAL, max_price REAL);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS
  INSERT INTO "DESTINATION_SQL_STREAM"
```

```
   SELECT STREAM STEP("SOURCE_SQL_STREAM_001".EVENT_TIME BY INTERVAL '60'
SECOND),
       TICKER,
        MIN(PRICE) AS MIN_PRICE,
        MAX(PRICE) AS MAX_PRICE
   FROM    "SOURCE_SQL_STREAM_001"
   GROUP BY TICKER,
            STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '60' SECOND),
            STEP("SOURCE_SQL_STREAM_001".EVENT_TIME BY INTERVAL '60' SECOND);
```

b. Escolha Save and run SQL.

   Na guia Real-time analytics (Análise em tempo real), você pode ver todos os fluxos de aplicativo criados pelo aplicativo e verificar os dados.

Exemplo: recuperação dos valores que ocorrem com mais frequência (TOP_K_ITEMS_TUMBLING)

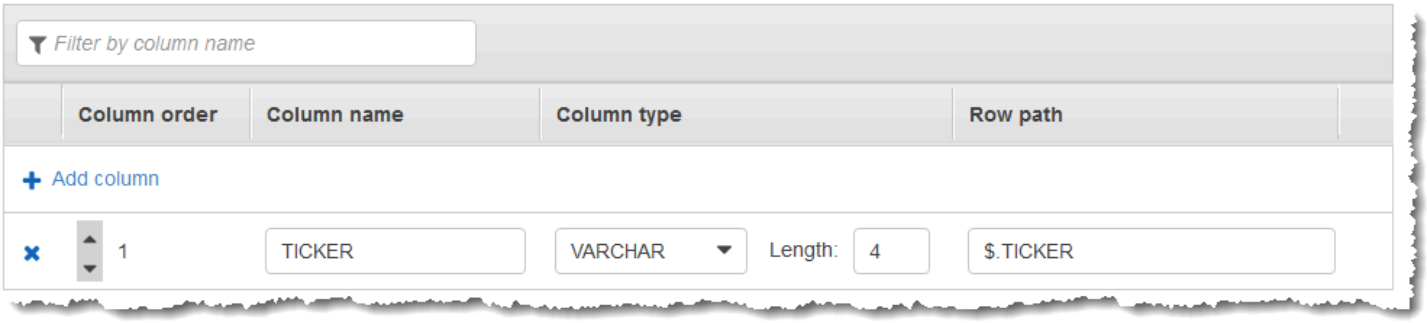
Este exemplo do Amazon Kinesis Data Analytics demonstra como usar a função `TOP_K_ITEMS_TUMBLING` para recuperar os valores que ocorrem com mais frequência em uma janela em cascata. Para obter mais informações, consulte a `TOP_K_ITEMS_TUMBLING`função em Amazon Managed Service for Apache Flink SQL Reference.

A função `TOP_K_ITEMS_TUMBLING` é útil quando agrega dezenas ou centenas de milhares de chaves e você deseja reduzir o uso de recursos. A função gera o mesmo resultado que a agregação com as cláusulas `GROUP BY` e `ORDER BY`.

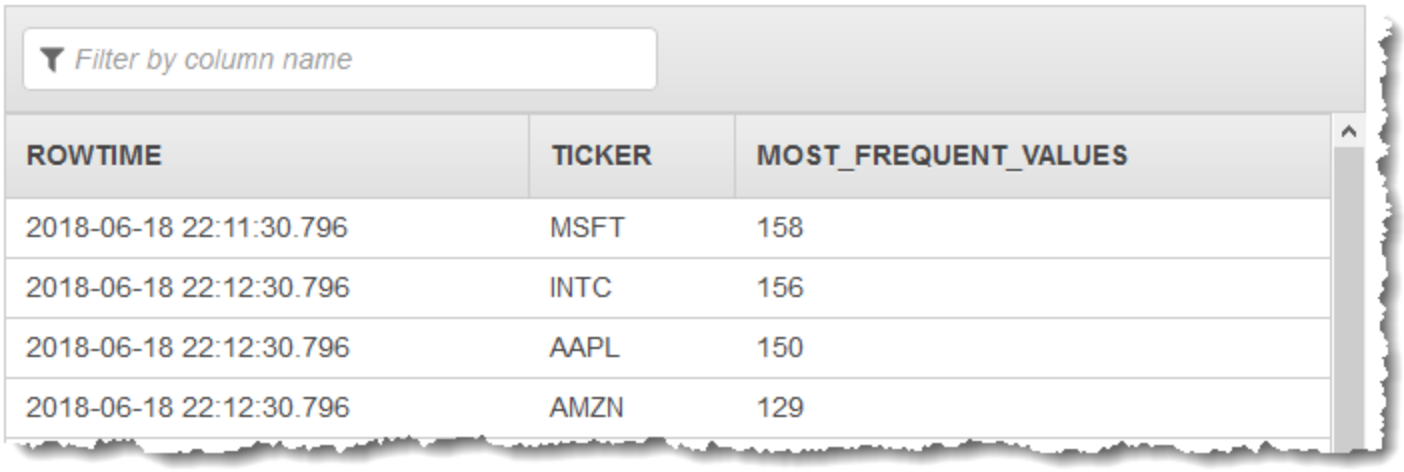
Neste exemplo, você grava os registros a seguir em um streaming de dados do Amazon Kinesis:

```
{"TICKER": "TBV"}
{"TICKER": "INTC"}
{"TICKER": "MSFT"}
{"TICKER": "AMZN"}
...
```

Em seguida, crie um aplicativo do Kinesis Data Analytics no AWS Management Console, com o streaming de dados do Kinesis como a origem de streaming. O processo de descoberta lê os registros de exemplo na origem de streaming e infere um esquema no aplicativo com uma coluna (`TICKER`) conforme mostrado a seguir.



Você usa o código do aplicativo com a função `TOP_K_VALUES_TUMBLING` para criar uma agregação em janelas dos dados. Em seguida, insira os dados resultantes em outro stream no aplicativo, conforme mostrado na captura de tela a seguir:



No procedimento a seguir, você criará um aplicativo do Kinesis Data Analytics que recupera os valores que ocorrem com mais frequência no stream de entrada.

Tópicos

- Etapa 1: Criar um Kinesis Data Stream
- Etapa 2: Criar um aplicativo Kinesis Data Analytics

Etapa 1: Criar um Kinesis Data Stream

Crie um fluxo de dados do Amazon Kinesis e preencha registros da seguinte forma:

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Selecione Data Streams (Streams de dados) no painel de navegação.

3. Selecione Create Kinesis stream (Criar stream do Kinesis) e crie um stream com um estilhaço. Para obter mais informações, consulte Criar um stream no Guia do desenvolvedor de Amazon Kinesis Data Streams.

4. Para gravar registros em um streaming de dados do Kinesis em um ambiente de produção, recomendamos usar o Kinesis Client Library ou a API do Kinesis Data Streams. Para simplificar, este exemplo usa o script Python a seguir para gerar registros. Execute o código para preencher os registros de marcador de exemplo. Esse código simples grava continuamente um registro de marcador aleatório no stream. Deixe o script em execução para que você possa gerar o esquema do aplicativo em uma etapa posterior.

```
import datetime
import json
import random
import boto3

STREAM_NAME = "ExampleInputStream"


def get_data():
    return {
        "EVENT_TIME": datetime.datetime.now().isoformat(),
        "TICKER": random.choice(["AAPL", "AMZN", "MSFT", "INTC", "TBV"]),
        "PRICE": round(random.random() * 100, 2),
    }


def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
        kinesis_client.put_record(
            StreamName=stream_name, Data=json.dumps(data),
 PartitionKey="partitionkey"
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Etapa 2: Criar um aplicativo Kinesis Data Analytics

Crie um aplicativo Kinesis Data Analytics, da seguinte maneira:

1. Abra o Managed Service for Apache Flink console em https://console.aws.amazon.com/kinesisanalytics.
2. Escolha Create application (Criar aplicativo), digite um nome para o aplicativo e selecione Create application (Criar aplicativo).
3. Na página de detalhes do aplicativo, escolha Connect streaming data (Conectar dados de streaming) para se conectar com a fonte.
4. Na página Connect to source (Conectar com a fonte), faça o seguinte:

   a. Escolha o stream criado na seção anterior.

   b. Selecione Discover schema (Descobrir esquema). Aguarde o console mostrar o esquema inferido e os registros de exemplos usados para inferir o esquema do stream do aplicativo criado. O esquema inferido tem uma coluna.

   c. Escolha Save schema and update stream samples. Depois que o console salvar o esquema, escolha Exit (Sair).

   d. Escolha Save and continue.
5. Na página de detalhes de aplicativo, escolha Go to SQL editor (Ir para o editor de SQL). Para iniciar o aplicativo, escolha Yes, start application (Sim, iniciar o aplicativo) na caixa de diálogo exibida.
6. No editor SQL, escreva o código do aplicativo e verifique os resultados da seguinte forma:

   a. Copie o código de aplicativo a seguir e cole-o no editor:

```
CREATE OR REPLACE STREAM DESTINATION_SQL_STREAM (
  "TICKER" VARCHAR(4),
  "MOST_FREQUENT_VALUES" BIGINT
);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS
    INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT STREAM *
        FROM TABLE (TOP_K_ITEMS_TUMBLING(
            CURSOR(SELECT STREAM * FROM "SOURCE_SQL_STREAM_001"),
            'TICKER',         -- name of column in single quotes
```

```
            5,                          -- number of the most frequently occurring
values
            60                          -- tumbling window size in seconds
            )
        );
```

b. Escolha Save and run SQL.

Na guia Real-time analytics (Análise em tempo real), você pode ver todos os fluxos de aplicativo criados pelo aplicativo e verificar os dados.

Exemplo: agregação dos resultados parciais de uma consulta

Se um streaming de dados do Amazon Kinesis contiver registros que tenham um horário do evento que não corresponda exatamente ao horário da ingestão, uma seleção de resultados em uma janela em cascata contém os registros que chegaram, mas não necessariamente ocorreram, dentro da janela. Nesse caso, a janela em cascata conterá somente um conjunto parcial dos resultados que você deseja. Há várias abordagens que você pode usar para corrigir o problema:

- Use somente uma janela em cascata e agregue resultados parciais no pós-processamento por meio de um banco de dados ou data warehouse usando upserts. Essa abordagem é eficiente no processamento de um aplicativo. Ela manipula os dados atrasados indefinidamente para operadores agregados (`sum`, `min`, `max`, e assim por diante). A desvantagem dessa abordagem é que você deve desenvolver e manter lógica adicional do aplicativo na camada de banco de dados.
- Use uma janela deslizante e em cascata, que produz resultados parciais logo, mas também continua a produzir resultados completos ao longo do período da janela deslizante. Essa abordagem trata os dados tardios com uma substituição em vez de um upsert para que nenhuma lógica adicional do aplicativo precise ser adicionada à camada de banco de dados. A desvantagem dessa abordagem é que ela usa mais unidades de processamento do Kinesis (KPUs) e ainda produz dois resultados, o que pode não funcionar para alguns casos de uso.

Para obter mais informações sobre janelas deslizantes e em cascata, consulte Consultas em janelas.

No procedimento a seguir, a agregação de janela em cascata produz dois resultados parciais (enviados ao stream no aplicativo `CALC_COUNT_SQL_STREAM`) que devem ser combinados para produzir um resultado final. O aplicativo, então, produz uma segunda agregação (enviada ao stream no aplicativo `DESTINATION_SQL_STREAM`) que combina os dois resultados parciais.

Para criar um aplicativo que agrega resultados parciais usando um horário do evento

1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Selecione Data Analytics (Análise de dados) no painel de navegação. Crie um aplicativo do Kinesis Data Analytics conforme descrito no tutorial Conceitos básicos do Amazon Kinesis Data Analytics para aplicativos SQL.
3. No editor SQL, substitua o código de aplicativo pelo seguinte:

```
CREATE OR REPLACE STREAM "CALC_COUNT_SQL_STREAM"
    (TICKER      VARCHAR(4),
    TRADETIME   TIMESTAMP,
    TICKERCOUNT       DOUBLE);

CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM"
    (TICKER      VARCHAR(4),
    TRADETIME   TIMESTAMP,
    TICKERCOUNT       DOUBLE);

CREATE PUMP "CALC_COUNT_SQL_PUMP_001" AS
    INSERT INTO "CALC_COUNT_SQL_STREAM" ("TICKER","TRADETIME", "TICKERCOUNT")
    SELECT STREAM
        "TICKER_SYMBOL",
        STEP("SOURCE_SQL_STREAM_001"."ROWTIME" BY INTERVAL '1' MINUTE) as
 "TradeTime",
        COUNT(*) AS "TickerCount"
    FROM "SOURCE_SQL_STREAM_001"
    GROUP BY
        STEP("SOURCE_SQL_STREAM_001".ROWTIME BY INTERVAL '1' MINUTE),
        STEP("SOURCE_SQL_STREAM_001"."APPROXIMATE_ARRIVAL_TIME" BY INTERVAL '1'
 MINUTE),
        TICKER_SYMBOL;

CREATE PUMP "AGGREGATED_SQL_PUMP" AS
    INSERT INTO "DESTINATION_SQL_STREAM" ("TICKER","TRADETIME", "TICKERCOUNT")
    SELECT STREAM
        "TICKER",
        "TRADETIME",
        SUM("TICKERCOUNT") OVER W1 AS "TICKERCOUNT"
    FROM "CALC_COUNT_SQL_STREAM"
    WINDOW W1 AS (PARTITION BY "TRADETIME" RANGE INTERVAL '10' MINUTE PRECEDING);
```

A instrução `SELECT` no código de aplicativo filtra as linhas no `SOURCE_SQL_STREAM_001` por alterações de preços de ações maiores de 1% e insere essas linhas em outro stream no aplicativo `CHANGE_STREAM` usando uma bomba.

4. Escolha Save and run SQL.

A primeira bomba produz um stream para `CALC_COUNT_SQL_STREAM` que é semelhante ao seguinte (observe que o conjunto de resultados está incompleto):

Filter by column name

ROWTIME	TICKER	TRADETIME	TICKERCOUNT
2018-01-30 22:57:00.0	BAC	2018-01-30 22:56:00.0	2.0
2018-01-30 22:57:00.0	ALY	2018-01-30 22:56:00.0	2.0
2018-01-30 22:57:00.0	DFG	2018-01-30 22:56:00.0	5.0
2018-01-30 22:57:00.0	CVB	2018-01-30 22:56:00.0	6.0

A segunda bomba produz um stream para `DESTINATION_SQL_STREAM` que contém o conjunto de resultados completo:

Filter by column name

ROWTIME	TICKER	TRADETIME	TICKERCOUNT
2018-01-30 23:17:00.0	PPL	2018-01-30 23:16:00.0	4.0
2018-01-30 23:18:00.0	TGT	2018-01-30 23:17:00.0	8.0
2018-01-30 23:18:00.0	DFT	2018-01-30 23:17:00.0	6.0
2018-01-30 23:18:00.0	KFU	2018-01-30 23:17:00.0	5.0

Exemplos: junção

Esta seção fornece exemplos de aplicativos Kinesis Data Analytics que usam consultas de junção. Cada exemplo fornece instruções passo a passo e código para configurar e testar seu aplicativo Kinesis Data Analytics.

Tópicos

Exemplo: adição de dados de referência a um aplicativo do Kinesis Data Analytics

Neste exercício, você adicionará dados de referência a um aplicativo existente de análise se dados do Kinesis Data Analytics. Para obter informações sobre os dados de referência, consulte os seguintes tópicos:

- Amazon Kinesis Data Analytics para aplicativos SQL: como funciona
- Configuração de entrada do aplicativo

Neste exercício, adicione dados de referência ao aplicativo criado no exercício Conceitos básicos do Kinesis Data Analytics. Os dados de referência fornecem o nome da empresa para cada símbolo de índice. Por exemplo:

```
Ticker, Company
AMZN,Amazon
ASD, SomeCompanyA
MMB, SomeCompanyB
WAS,  SomeCompanyC
```

Primeiro, conclua as etapas do exercício Conceitos básicos para criar um aplicativo inicial. Em seguida, siga estas etapas para configurar e adicionar dados de referência ao aplicativo:

1. Preparar os dados
   - Armazene os dados de referência anteriores como um objeto no Amazon Simple Storage Service (Amazon S3).
   - Crie um perfil do IAM, que o Kinesis Data Analytics pode assumir para ler o objeto do Amazon S3 em seu nome.

2. Adicione a fonte de dados de referência ao aplicativo.

   O Kinesis Data Analytics lê o objeto do Amazon S3 e cria uma tabela de referência de aplicativo que você pode consultar no código do aplicativo.

3. Teste o código.

   No código do aplicativo, você gravará uma consulta de junção para fazer a junção do fluxo do aplicativo com a tabela de referência de aplicativo, a fim de obter o nome da empresa em cada símbolo de índice.

Tópicos

Etapa 1: preparar

Nesta seção, armazene os dados de referência de exemplo como um objeto no bucket do Amazon S3. Você também cria um perfil do IAM que o Kinesis Data Analytics pode assumir para ler o objeto em seu nome.

Armazene os dados de referência como objeto do Amazon S3

Nesta etapa, armazene os dados de referência como um objeto do Amazon S3.

1. Abra um editor de texto, adicione os seguintes dados e salve o arquivo como `TickerReference.csv`.

   ```
   Ticker, Company
   AMZN,Amazon
   ASD, SomeCompanyA
   MMB, SomeCompanyB
   WAS,  SomeCompanyC
   ```

2. Faça upload do arquivo `TickerReference.csv` no bucket do S3. Para obter mais informações, consulte Fazer upload de objetos no Amazon S3 no Manual do usuário do Amazon Simple Storage Service.

Criar uma função do IAM

Crie um perfil do IAM, que o Kinesis Data Analytics pode assumir para ler o objeto do Amazon S3.

1. Em AWS Identity and Access Management (IAM), crie um perfil do IAM chamado **`KinesisAnalytics-ReadS3Object`**. Para criar o perfil, siga as instruções em Criação de um perfil para um serviço da Amazon (AWS Management Console) no Guia do usuário do IAM.

   No console do IAM, especifique o seguinte:

   - Em Selecione o tipo de perfil, escolha AWS Lambda. Após criar a função, você alterará a política de confiança para permitir que o Kinesis Data Analytic (não AWS Lambda) assuma a função.
   - Não anexe nenhuma política na página Attach Policy.
2. Atualize as políticas de perfil do IAM:

   a. No console do IAM, selecione o perfil criado.

   b. Na guia Relações de confiança, atualize a política de confiança para conceder ao Kinesis Data Analytics permissões para assumir a função. A política de confiança é mostrada a seguir:

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

   c. Na guia Permissões, anexe uma política gerenciada pela Amazon chamada AmazonS3ReadOnlyAccess. Isso concede ao perfil permissões para ler um objeto do Amazon S3. Esta política é mostrada a seguir:

   ```
   {
   ```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Resource": "*"
    }
  ]
}
```

Etapa 2: Adicionar a fonte de dados de referência à configuração do aplicativo.

Nesta etapa, adicione a fonte de dados de referência à configuração do aplicativo. Para começar, as seguintes informações são necessárias:

- Nome do seu bucket do S3 e nome da chave do objeto
- O nome do recurso da Amazon (ARN) do perfil do IAM.

1. Na página principal do aplicativo, selecione Connect reference data (Conectar dados de referência).
2. Na página Conectar fonte de dados de referência, selecione o bucket do Amazon S3 que contém seu objeto de dados de referência e insira o nome da chave do objeto.
3. Insira **`CompanyName`** em In-application reference table name (Nome da tabela de referência no aplicativo).
4. Na seção Access to chosen resources (Acesso aos recursos escolhidos), selecione Choose from IAM roles that Kinesis Analytics can assume (Selecionar funções do IAM que o Kinesis Analytics possa assumir) e selecione a função do IAM KinesisAnalytics-ReadS3Object que você criou na seção anterior.
5. Escolha Discover schema (Descobrir esquema). O console detectará duas colunas nos dados de referência.
6. Escolha Save and close.

Etapa 3: Teste: consultar a tabela de referência de aplicativo

Agora você pode consultar a tabela de referência de aplicativo, `CompanyName`. Você pode usar as informações de referência para enriquecer o aplicativo, fazendo a junção dos dados de preço do índice com a tabela de referência de dados. O resultado mostrará o nome da empresa.

1. Substitua o código do aplicativo pelo seguinte. A consulta faz a junção do fluxo de entrada de aplicativo com a tabela de referência de aplicativo. O código do aplicativo grava os resultados em outro fluxo no aplicativo, `DESTINATION_SQL_STREAM`.

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (ticker_symbol VARCHAR(4),
 "Company" varchar(20), sector VARCHAR(12), change DOUBLE, price DOUBLE);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS INSERT INTO "DESTINATION_SQL_STREAM"
   SELECT STREAM ticker_symbol, "c"."Company", sector, change, price
   FROM "SOURCE_SQL_STREAM_001" LEFT JOIN "CompanyName" as "c"
   ON "SOURCE_SQL_STREAM_001".ticker_symbol = "c"."Ticker";
```

2. Verifique se a saída do aplicativo aparece na guia SQLResults. Verifique se algumas linhas mostram nomes de empresa (os dados de referência de exemplo não têm todos os nomes de empresa).

Exemplos: Machine Learning

Esta seção fornece exemplos de aplicativos do Amazon Kinesis Data Analytics que usam consultas de machine learning. As consultas de machine learning realizam análises complexas nos dados, contando com o histórico dos dados no fluxo para encontrar padrões incomuns. Os exemplos fornecem instruções passo a passo para configurar e testar o seu aplicativo Kinesis Data Analytics.

Tópicos

Exemplo: detecção de anomalias de dados em um fluxo (função RANDOM_CUT_FOREST)

O Amazon Kinesis Data Analytics fornece uma função (`RANDOM_CUT_FOREST`) que pode atribuir uma pontuação de anomalias a cada registro de acordo com os valores nas colunas numéricas. Para obter mais informações, consulte `RANDOM_CUT_FOREST` Funções em Amazon Managed Service for Apache Flink SQL Reference.

Neste exercício, você grava o código de aplicativo para atribuir uma pontuação de anomalias a registros na origem de streaming do aplicativo. Para configurar o aplicativo, faça o seguinte:

1. Configurar uma origem de streaming: configure um fluxo de dados do Kinesis e grave dados `heartRate` de amostra, como mostrado a seguir:

   ```
   {"heartRate": 60, "rateType":"NORMAL"}
   ...
   {"heartRate": 180, "rateType":"HIGH"}
   ```

   O procedimento fornece um script Python para preencher o stream. Os valores `heartRate` são gerados aleatoriamente, com 99% dos registros com valores `heartRate` entre 60 e 100, e apenas 1% de valores `heartRate` entre 150 e 200. Assim, os registros que têm valores `heartRate` entre 150 e 200 são anomalias.
2. Configurar entrada: usando o console, crie um aplicativo do Kinesis Data Analytics e configure a entrada de aplicativo mapeando a origem de streaming para um fluxo no aplicativo (`SOURCE_SQL_STREAM_001`). Quando o aplicativo é iniciado, o Kinesis Data Analytics lê continuamente a origem do streaming e insere registros no streaming do aplicativo de entrada.
3. Especificar o código de aplicativo – O exemplo usa o seguinte código de aplicativo:

   ```
   --Creates a temporary stream.
   CREATE OR REPLACE STREAM "TEMP_STREAM" (
            "heartRate"        INTEGER,
            "rateType"         varchar(20),
            "ANOMALY_SCORE"    DOUBLE);

   --Creates another stream for application output.
   CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
            "heartRate"        INTEGER,
            "rateType"         varchar(20),
            "ANOMALY_SCORE"    DOUBLE);
   ```

```
-- Compute an anomaly score for each record in the input stream
-- using Random Cut Forest
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "TEMP_STREAM"
      SELECT STREAM "heartRate", "rateType", ANOMALY_SCORE
      FROM TABLE(RANDOM_CUT_FOREST(
              CURSOR(SELECT STREAM * FROM "SOURCE_SQL_STREAM_001")));

-- Sort records by descending anomaly score, insert into output stream
CREATE OR REPLACE PUMP "OUTPUT_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM * FROM "TEMP_STREAM"
      ORDER BY FLOOR("TEMP_STREAM".ROWTIME TO SECOND), ANOMALY_SCORE DESC;
```

O código lê linhas no `SOURCE_SQL_STREAM_001`, atribui uma pontuação de anomalias e grava as linhas resultantes em outro stream no aplicativo (`TEMP_STREAM`). O código de aplicativo, em seguida, classifica os registros no `TEMP_STREAM` e salva os resultados em outro stream no aplicativo (`DESTINATION_SQL_STREAM`). Você usa bombas para inserir linhas em streamings do aplicativo de entrada. Para obter mais informações, consulte Bombas e fluxos no aplicativo.

4. Configurar a saída: configure a saída de aplicativo para manter os dados no `DESTINATION_SQL_STREAM` para um destino externo, que é outro fluxo de dados do Kinesis. Verificar as pontuações de anomalias atribuídas a cada registro e determinar qual pontuação indica que uma anomalia (é preciso estar alerta) é externa ao aplicativo. Você pode usar uma função AWS Lambda para processar essas pontuações de anomalias e configurar alertas.

O exercício usa a região do Leste dos EUA (Norte da Virgínia) (`us-east-1`) para criar esses streams e o aplicativo. Se você usar qualquer outra região, precisará atualizar o código de acordo.

Tópicos

- Etapa 1: preparar
- Etapa 2: criar um aplicativo
- Etapa 3: Configuração da saída de aplicativo
- Etapa 4: verificar a saída

Próxima etapa

Etapa 1: preparar

Etapa 1: preparar

Antes de criar um aplicativo Amazon Kinesis Data Analytics para este exercício, crie dois fluxos de dados do Kinesis. Configure um dos streamings como a origem de streaming do aplicativo e outro como o destino em que o Kinesis Data Analytics mantém a saída do aplicativo.

Tópicos

Etapa 1.1: criar os fluxos de dados de entrada e saída

Nesta seção, você cria dois streams do Kinesis: `ExampleInputStream` e `ExampleOutputStream`. Crie esses streams usando o AWS Management Console ou a AWS CLI.

- Para usar o console do

  1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
  2. Selecione Create data stream (Criar stream de dados). Crie um fluxo com um estilhaço chamado `ExampleInputStream`. Para obter mais informações, consulte Criar um fluxo no Guia do desenvolvedor do Amazon Kinesis Data Streams.
  3. Repita a etapa anterior, criando um streaming com um estilhaço denominado `ExampleOutputStream`.

- Para usar a AWS CLI

  1. Use o seguinte comando Kinesis `create-stream` AWS CLI para criar o primeiro stream (`ExampleInputStream`).

     ```
     $ aws kinesis create-stream \
     --stream-name ExampleInputStream \
     --shard-count 1 \
     --region us-east-1 \
     --profile adminuser
     ```

  2. Execute o mesmo comando, alterando o nome do fluxo para `ExampleOutputStream`. Esse comando cria o segundo fluxo usado pelo aplicativo para gravar a saída.

Etapa 1.2: Gravação de registros de amostra no stream de entrada

Nesta etapa, você executa o código Python para gerar continuamente registros de amostra e gravar esses registros no fluxo `ExampleInputStream`.

```
{"heartRate": 60, "rateType":"NORMAL"}
...
{"heartRate": 180, "rateType":"HIGH"}
```

1. Instale o Python e o `pip`.

   Para obter informações sobre como instalar o Python, consulte o site do Python.

   Você pode instalar dependências usando o pip. Para obter informações sobre como instalar o pip, consulte Installation no site do pip.

2. Execute o código do Python a seguir. O comando `put-record` no código grava os registros JSON no fluxo.

```
from enum import Enum
import json
import random
import boto3

STREAM_NAME = "ExampleInputStream"


class RateType(Enum):
    normal = "NORMAL"
    high = "HIGH"


def get_heart_rate(rate_type):
    if rate_type == RateType.normal:
        rate = random.randint(60, 100)
    elif rate_type == RateType.high:
        rate = random.randint(150, 200)
    else:
        raise TypeError
    return {"heartRate": rate, "rateType": rate_type.value}
```

```
def generate(stream_name, kinesis_client, output=True):
    while True:
        rnd = random.random()
        rate_type = RateType.high if rnd < 0.01 else RateType.normal
        heart_rate = get_heart_rate(rate_type)
        if output:
            print(heart_rate)
        kinesis_client.put_record(
            StreamName=stream_name,
            Data=json.dumps(heart_rate),
            PartitionKey="partitionkey",
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Próxima etapa

Etapa 2: criar um aplicativo

Etapa 2: criar um aplicativo

Nesta seção, você cria um aplicativo Amazon Kinesis Data Analytics, como segue:

- Configure a entrada do aplicativo para usar o fluxo de dados do Kinesis criado no the section called “Etapa 1: preparar” por você como a origem do streaming.
- Use o modelo Anomaly Detection (Detecção de anomalia) no console.

Para criar um aplicativo.

1. Siga as etapas 1, 2 e 3 no exercício do Kinesis Data Analytics Conceitos básicos (consulte Etapa 3.1: criar um aplicativo).

   - Na configuração de origem, faça o seguinte:
     - Especifique a origem do streaming que você criou na seção anterior.
     - Após o console inferir o esquema, edite-o e defina o tipo da coluna `heartRate` para `INTEGER`.

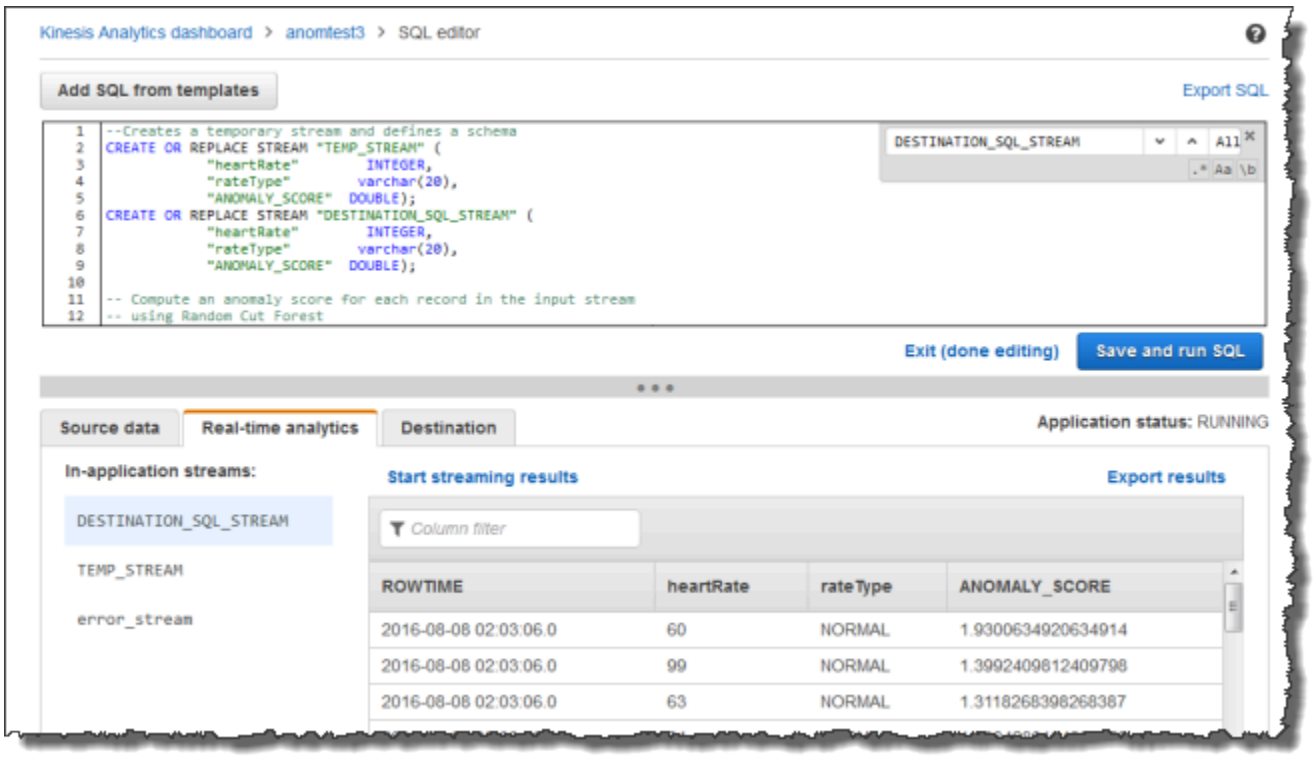
A maioria dos valores de frequência cardíaca são normais e o processo de descoberta provavelmente atribuirá o tipo `TINYINT` a esta coluna. Mas uma porcentagem pequena dos valores mostra uma frequência cardíaca alta. Se esses valores altos não forem adequados para o tipo `TINYINT`, o Kinesis Data Analytics enviará essas linhas para o fluxo de erro. Atualize o tipo de dados para `INTEGER` a fim de acomodar todos os dados de frequência cardíaca gerados.

- Use o modelo Anomaly Detection (Detecção de anomalia) no console. Em seguida, atualize o código de modelo para fornecer o nome apropriado à coluna.

2. Atualize o código de aplicativo fornecendo nomes de colunas. O código de aplicativo resultante é mostrado a seguir (cole esse código no editor SQL):

```
--Creates a temporary stream.
CREATE OR REPLACE STREAM "TEMP_STREAM" (
         "heartRate"        INTEGER,
         "rateType"         varchar(20),
         "ANOMALY_SCORE"    DOUBLE);

--Creates another stream for application output.
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
         "heartRate"        INTEGER,
         "rateType"         varchar(20),
         "ANOMALY_SCORE"    DOUBLE);

-- Compute an anomaly score for each record in the input stream
-- using Random Cut Forest
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "TEMP_STREAM"
      SELECT STREAM "heartRate", "rateType", ANOMALY_SCORE
      FROM TABLE(RANDOM_CUT_FOREST(
              CURSOR(SELECT STREAM * FROM "SOURCE_SQL_STREAM_001")));

-- Sort records by descending anomaly score, insert into output stream
CREATE OR REPLACE PUMP "OUTPUT_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM * FROM "TEMP_STREAM"
      ORDER BY FLOOR("TEMP_STREAM".ROWTIME TO SECOND), ANOMALY_SCORE DESC;
```

3. Execute o código SQL e revise os resultados no console do Kinesis Data Analytics:



Próxima etapa

[Etapa 3: Configuração da saída de aplicativo](#)

Etapa 3: Configuração da saída de aplicativo

Depois de preencher the section called “Etapa 2: criar um aplicativo”, você terá o código de aplicativo que lê os dados de frequência cardíaca em uma origem de streaming e atribui uma nova pontuação de anomalias a cada uma delas.

Você já pode enviar os resultados do aplicativo do streaming do aplicativo de entrada para um destino externo, outro fluxo de dados do Kinesis (`OutputStreamTestingAnomalyScores`). Você pode analisar as pontuações de anomalias e determinar qual frequência cardíaca é anômala. É possível estender mais este aplicativo para gerar alertas.

Siga estas etapas para configurar a saída de aplicativo:

1. Abra o console do Amazon Kinesis Data Analytics. No editor SQL, escolha Destination ou Add a destination no painel do aplicativo.

2. Na página Connect to destination (Conectar-se ao destino), escolha o fluxo `OutputStreamTestingAnomalyScores` criado na seção anterior.

   Agora, você tem um destino externo, em que o Amazon Kinesis Data Analytics mantém quaisquer registros que o aplicativo gravar no fluxo no aplicativo `DESTINATION_SQL_STREAM`.

3. Como opção, configure o AWS Lambda para monitorar o stream `OutputStreamTestingAnomalyScores` e enviar alertas. Para obter instruções, consulte Pré-processar dados usando uma função do Lambda. Se você não definir alertas, poderá rever os registros que o Kinesis Data Analytics grava no destino externo, que é o fluxo de dados do Kinesis `OutputStreamTestingAnomalyScores`, conforme descrito em Etapa 4: verificar a saída.

Próxima etapa

Etapa 4: verificar a saída

Etapa 4: verificar a saída

Depois de configurar a saída do aplicativo em the section called “Etapa 3: Configuração da saída de aplicativo”, use os seguintes comandos da AWS CLI para ler registros no fluxo de destino gravados pelo aplicativo:

1. Execute o comando `get-shard-iterator` para obter um ponteiro para os dados no stream de saída.

   ```
   aws kinesis get-shard-iterator \
   --shard-id shardId-000000000000 \
   --shard-iterator-type TRIM_HORIZON \
   --stream-name OutputStreamTestingAnomalyScores \
   --region us-east-1 \
   --profile adminuser
   ```

   Você obtém uma resposta com um valor de iterador de estilhaços, como mostrado no seguinte exemplo de resposta:

   ```
   {
       "ShardIterator":
       "shard-iterator-value"   }
   ```

Copie o valor de iterador de estilhaços.

2. Execute o comando AWS CLI `get-records`.

```
aws kinesis get-records \
--shard-iterator shared-iterator-value \
--region us-east-1 \
--profile adminuser
```

O comando retorna uma página de registros e outro iterador de estilhaços que pode ser usado no comando `get-records` subsequente para buscar o próximo conjunto de registros.

Exemplo: como detectar anomalias de dados e obter uma explicação (função RANDOM_CUT_FOREST_WITH_EXPLANATION)

O Amazon Kinesis Data Analytics fornece a função `RANDOM_CUT_FOREST_WITH_EXPLANATION` que atribui uma pontuação de anomalias a cada registro de acordo com os valores nas colunas numéricas. A função também fornece uma explicação da anomalia. Para obter mais informações, consulte RANDOM_CUT_FOREST_WITH_EXPLANATION em Amazon Managed Service for Apache Flink SQL Reference.

Neste exercício, você escreve o código de aplicativo para obter pontuações de anomalias para os registros na origem de streaming do aplicativo. Você também pode obter uma explicação para cada anomalia.

Tópicos

- Etapa 1: Preparar os dados
- Etapa 2: Criar um aplicativo de análise
- Etapa 3: Examinar os resultados

Primeira etapa

Etapa 1: Preparar os dados

Etapa 1: Preparar os dados

Antes de criar um Amazon Kinesis Data Analytics para este exemplo, crie um fluxo de dados do Kinesis para usar como origem de streaming para o seu aplicativo. Além disso, execute o código Python para gravar dados simulados de pressão arterial no fluxo.

Tópicos

- Etapa 1.1: Criar um fluxo de dados do Kinesis
- Etapa 1.2: Gravação de registros de amostra no stream de entrada

Etapa 1.1: Criar um fluxo de dados do Kinesis

Nesta seção, você cria um fluxo de dados do Kinesis chamado `ExampleInputStream`. Você pode criar esse fluxo de dados usando o AWS Management Console ou a AWS CLI.

- Para usar o console:
  1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
  2. Selecione Data Streams (Fluxos de dados) no painel de navegação. Em seguida, escolha Create Kinesis stream (Criar fluxo do Kinesis).
  3. Para o nome, digite **`ExampleInputStream`**. Para o número de estilhaços, digite **`1`**.
- Como alternativa, para usar a AWS CLI para criar o fluxo de dados , execute o seguinte comando:

```
$ aws kinesis create-stream --stream-name ExampleInputStream --shard-count 1
```

Etapa 1.2: Gravação de registros de amostra no stream de entrada

Nesta etapa, execute o código Python para gerar continuamente os registros de exemplo e gravá-los no fluxo de dados que você criou.

1. Instale o Python e o pip.

   Para obter informações sobre como instalar o Python, consulte Python.

   Você pode instalar dependências usando o pip. Para obter informações sobre como instalar o pip, consulte Instalação na documentação do pip.

2. Execute o código do Python a seguir. Você pode alterar a região a ser usada neste exemplo. O comando `put-record` no código grava os registros JSON no fluxo.

```
from enum import Enum
import json
import random
import boto3

STREAM_NAME = "ExampleInputStream"


class PressureType(Enum):
    low = "LOW"
    normal = "NORMAL"
    high = "HIGH"


def get_blood_pressure(pressure_type):
    pressure = {"BloodPressureLevel": pressure_type.value}
    if pressure_type == PressureType.low:
        pressure["Systolic"] = random.randint(50, 80)
        pressure["Diastolic"] = random.randint(30, 50)
    elif pressure_type == PressureType.normal:
        pressure["Systolic"] = random.randint(90, 120)
        pressure["Diastolic"] = random.randint(60, 80)
    elif pressure_type == PressureType.high:
        pressure["Systolic"] = random.randint(130, 200)
        pressure["Diastolic"] = random.randint(90, 150)
    else:
        raise TypeError
    return pressure


def generate(stream_name, kinesis_client):
    while True:
        rnd = random.random()
        pressure_type = (
            PressureType.low
            if rnd < 0.005
            else PressureType.high
            if rnd > 0.995
            else PressureType.normal
```

```
        )
        blood_pressure = get_blood_pressure(pressure_type)
        print(blood_pressure)
        kinesis_client.put_record(
            StreamName=stream_name,
            Data=json.dumps(blood_pressure),
            PartitionKey="partitionkey",
        )


if __name__ == "__main__":
    generate(STREAM_NAME, boto3.client("kinesis"))
```

Próxima etapa

Etapa 2: Criar um aplicativo de análise

Etapa 2: Criar um aplicativo de análise

Nesta seção, você cria um aplicativo do Amazon Kinesis Data Analytics e o configura para usar o fluxo de dados do Kinesis que você criou em the section called “Etapa 1: Preparar os dados”. Em seguida, você executa o código do aplicativo que usa a função `RANDOM_CUT_FOREST_WITH_EXPLANATION`.

Para criar um aplicativo.

1. Abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
2. Escolha Data Analytics (Análise de dados) no painel de navegação e, em seguida, Create application (Criar aplicativo).
3. Forneça o nome e a descrição do aplicativo (opcional) e escolha Create application.
4. Escolha Connect streaming data (Conectar fluxo de dados) e, em seguida, escolha ExampleInputStream na lista.
5. Escolha Discover schema e verifique se `Systolic` e `Diastolic` aparecem como colunas `INTEGER`. Se elas tiverem um outro tipo, escolha Edit schema e atribua o tipo `INTEGER` a ambas.
6. Em Real time analytics, escolha Go to SQL editor. Quando solicitado, confirme a execução do aplicativo.
7. Cole o seguinte código no editor SQL e, em seguida, escolha Save and run SQL.

```
--Creates a temporary stream.
CREATE OR REPLACE STREAM "TEMP_STREAM" (
         "Systolic"                 INTEGER,
         "Diastolic"                INTEGER,
         "BloodPressureLevel"       varchar(20),
         "ANOMALY_SCORE"            DOUBLE,
         "ANOMALY_EXPLANATION"      varchar(512));

--Creates another stream for application output.
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
         "Systolic"                 INTEGER,
         "Diastolic"                INTEGER,
         "BloodPressureLevel"       varchar(20),
         "ANOMALY_SCORE"            DOUBLE,
         "ANOMALY_EXPLANATION"      varchar(512));

-- Compute an anomaly score with explanation for each record in the input stream
-- using RANDOM_CUT_FOREST_WITH_EXPLANATION
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "TEMP_STREAM"
      SELECT STREAM "Systolic", "Diastolic", "BloodPressureLevel", ANOMALY_SCORE,
 ANOMALY_EXPLANATION
      FROM TABLE(RANDOM_CUT_FOREST_WITH_EXPLANATION(
              CURSOR(SELECT STREAM * FROM "SOURCE_SQL_STREAM_001"), 100, 256,
 100000, 1, true));

-- Sort records by descending anomaly score, insert into output stream
CREATE OR REPLACE PUMP "OUTPUT_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM * FROM "TEMP_STREAM"
      ORDER BY FLOOR("TEMP_STREAM".ROWTIME TO SECOND), ANOMALY_SCORE DESC;
```

Próxima etapa

Etapa 3: Examinar os resultados

Etapa 3: Examinar os resultados

Quando você executa o código SQL para este exemplo, primeiro consulte as linhas com uma pontuação de anomalias igual a zero. Isso acontece durante a fase de aprendizagem inicial. Em seguida, você obtém resultados semelhantes aos seguintes:

```
ROWTIME SYSTOLIC DIASTOLIC BLOODPRESSURELEVEL ANOMALY_SCORE ANOMALY_EXPLANATION
27:49.0 101      66        NORMAL              0.711460417   {"Systolic":
{"DIRECTION":"LOW","STRENGTH":"0.0922","ATTRIBUTION_SCORE":"0.3792"},"Diastolic":
{"DIRECTION":"HIGH","STRENGTH":"0.0210","ATTRIBUTION_SCORE":"0.3323"}}
27:50.0 144      123       HIGH                3.855851061   {"Systolic":
{"DIRECTION":"HIGH","STRENGTH":"0.8567","ATTRIBUTION_SCORE":"1.7447"},"Diastolic":
{"DIRECTION":"HIGH","STRENGTH":"7.0982","ATTRIBUTION_SCORE":"2.1111"}}
27:50.0 113      69        NORMAL              0.740069409   {"Systolic":
{"DIRECTION":"LOW","STRENGTH":"0.0549","ATTRIBUTION_SCORE":"0.3750"},"Diastolic":
{"DIRECTION":"LOW","STRENGTH":"0.0394","ATTRIBUTION_SCORE":"0.3650"}}
27:50.0 105      64        NORMAL              0.739644157   {"Systolic":
{"DIRECTION":"HIGH","STRENGTH":"0.0245","ATTRIBUTION_SCORE":"0.3667"},"Diastolic":
{"DIRECTION":"LOW","STRENGTH":"0.0524","ATTRIBUTION_SCORE":"0.3729"}}
27:50.0 100      65        NORMAL              0.736993425   {"Systolic":
{"DIRECTION":"HIGH","STRENGTH":"0.0203","ATTRIBUTION_SCORE":"0.3516"},"Diastolic":
{"DIRECTION":"LOW","STRENGTH":"0.0454","ATTRIBUTION_SCORE":"0.3854"}}
27:50.0 108      69        NORMAL              0.733767202   {"Systolic":
{"DIRECTION":"LOW","STRENGTH":"0.0974","ATTRIBUTION_SCORE":"0.3961"},"Diastolic":
{"DIRECTION":"LOW","STRENGTH":"0.0189","ATTRIBUTION_SCORE":"0.3377"}}
```

- O algoritmo da função `RANDOM_CUT_FOREST_WITH_EXPLANATION` vê que as colunas `Systolic` e `Diastolic` são numéricas e as utiliza como entrada.
- A coluna `BloodPressureLevel` tem dados de texto e, portanto, não é levada em conta pelo algoritmo. Essa coluna é simplesmente um auxiliar visual para ajudá-lo a reconhecer rapidamente os níveis de pressão arterial normal, alto e baixo neste exemplo.
- Na coluna `ANOMALY_SCORE`, os registros com pontuações mais altas são mais anormais. O segundo registro neste exemplo de conjunto de resultados é o mais anormal, com uma pontuação de anomalia de 3.855851061.
- Para entender o quanto cada uma das colunas numéricas levadas em conta pelo algoritmo contribui para a pontuação de anomalia, consulte o campo JSON chamado `ATTRIBUTION_SCORE` na coluna `ANOMALY_SCORE`. No caso de uma segunda linha nesse conjuntos de resultados de amostra, as colunas `Systolic` e `Diastolic` contribuem para a anomalia na proporção 1.7447:2.1111. Em outras palavras, 45 por cento da explicação da pontuação de anomalia é atribuível ao valor sistólico, e a atribuição restante é por conta do valor diastólico.
- Para determinar a direção na qual o ponto representado pela segunda linha neste exemplo é anormal, consulte o campo JSON chamado `DIRECTION`. Ambos os valores diastólico e sistólico são marcados como `HIGH` neste caso. Para determinar a confiança de que essas direções estão

corretas, consulte o campo JSON chamado `STRENGTH`. Neste exemplo, o algoritmo está mais seguro que o valor diastólico é alto. De fato, o nível normal para a leitura do valor diastólico é geralmente 60-80, e 123 é muito mais alto do que o esperado.

Exemplo: detectar hotspots em um streaming (função HOTSPOTS)

O Amazon Kinesis Data Analytics oferece uma função `HOTSPOTS`, que pode localizar e retornar informações sobre regiões relativamente densas nos dados. Para obter mais informações, consulte HOTSPOTS em Amazon Managed Service for Apache Flink SQL Reference.

Neste exercício, você escreve o código do aplicativo para localizar hotspots na origem de streaming do aplicativo. Para configurar o aplicativo, siga as seguintes etapas:

1. Configurar uma origem de streaming: configure um streaming do Kinesis e escreva dados de coordenadas de amostra conforme mostrado a seguir:

   ```
   {"x": 7.921782426109737, "y": 8.746265312709893, "is_hot": "N"}
   {"x": 0.722248626528026, "y": 4.648868803193405, "is_hot": "Y"}
   ```

   O exemplo fornece um script Python para preencher o streaming. Os valores `x` e `y` são gerados aleatoriamente, com alguns registros sendo agrupados em determinados locais.

   O campo `is_hot` será fornecido como um indicador se o script tiver gerado intencionalmente o valor como parte de um ponto de acesso. Isso pode ajudar a avaliar se a função de detecção do ponto de acesso está operando corretamente.
2. Crie o aplicativo: usando o AWS Management Console, você poderá criar um aplicativo Kinesis Data Analytics. Configure a entrada do aplicativo mapeando a origem do streaming para um streaming do aplicativo de entrada (`SOURCE_SQL_STREAM_001`). Quando o aplicativo é iniciado, o Kinesis Data Analytics lê continuamente a origem do streaming e insere registros no streaming do aplicativo de entrada.

   Neste exercício, você usa o seguinte código para o aplicativo:

   ```
   CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
       "x" DOUBLE,
       "y" DOUBLE,
       "is_hot" VARCHAR(4),
       HOTSPOTS_RESULT VARCHAR(10000)
   );
   ```

```
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
    INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT "x", "y", "is_hot", "HOTSPOTS_RESULT"
    FROM TABLE (
        HOTSPOTS(
            CURSOR(SELECT STREAM "x", "y", "is_hot" FROM "SOURCE_SQL_STREAM_001"),
            1000,
            0.2,
            17)
    );
```

O código lê linhas no `SOURCE_SQL_STREAM_001`, analisa-o em busca de hotspots significativos e grava os dados resultantes em outro streaming do aplicativo de entrada (`DESTINATION_SQL_STREAM`). Você usa bombas para inserir linhas em streamings do aplicativo de entrada. Para obter mais informações, consulte Bombas e fluxos no aplicativo.

3. Configurar a saída: configure a saída do aplicativo para enviar dados do aplicativo para um destino externo, outro fluxo de dados do Kinesis. Revise as pontuações do ponto de acesso e determine quais indicam se um ponto de acesso ocorreu (e que você precisa ser alertado). Você pode usar uma função do AWS Lambda para processar ainda mais informações do ponto de acesso e configurar alertas.
4. Verificar a saída – O exemplo inclui um aplicativo JavaScript que lê dados do streaming de saída e os exibe graficamente, de maneira que você possa ver os hotspots gerados pelo aplicativo em tempo real.

O exercício usa a região do Oeste dos EUA (Oregon) (`us-west-2`) para criar esses streams e o aplicativo. Se você usar qualquer outra região, atualize o código de acordo.

Tópicos

- Etapa 1: criar os streamings de entrada e saída
- Etapa 2: criar o aplicativo Kinesis Data Analytics
- Etapa 3: configurar a saída do aplicativo
- Etapa 4: verificar a saída do aplicativo

Etapa 1: criar os streamings de entrada e saída

Antes de criar um aplicativo Amazon Kinesis Data Analytics para o exemplo de Hotspots, crie dois fluxos de dados do Kinesis. Configure um dos streamings como a origem de streaming do aplicativo e outro como o destino em que o Kinesis Data Analytics mantém a saída do aplicativo.

Tópicos

- Etapa 1.1: Criar os fluxos de dados do Kinesis
- Etapa 1.2: Gravação de registros de amostra no stream de entrada

Etapa 1.1: Criar os fluxos de dados do Kinesis

Nesta seção, você cria dois fluxos de dados do Kinesis: `ExampleInputStream` e `ExampleOutputStream`.

Crie esses fluxos de dados usando o console ou a AWS CLI.

- Para criar os fluxos de dados usando o console:

  1. Faça login no AWS Management Console e abra o console do Kinesis em https://console.aws.amazon.com/kinesis.
  2. Selecione Data Streams (Fluxos de dados) no painel de navegação.
  3. Escolha Create Kinesis stream (Criar fluxo do Kinesis) e crie um fluxo com um estilhaço chamado `ExampleInputStream`.
  4. Repita a etapa anterior, criando um streaming com um estilhaço denominado `ExampleOutputStream`.

- Para criar fluxos de dados usando a AWS CLI:

  - Crie fluxos (`ExampleInputStream` e `ExampleOutputStream`) usando o seguinte comando Kinesis `create-stream` AWS CLI. Para criar o segundo streaming, que o aplicativo usará para gravar a saída, execute o mesmo comando, alterando o nome do streaming para `ExampleOutputStream`.

    ```
    $ aws kinesis create-stream \
    --stream-name ExampleInputStream \
    --shard-count 1 \
    --region us-west-2 \
    --profile adminuser
    ```

```
$ aws kinesis create-stream \
--stream-name ExampleOutputStream \
--shard-count 1 \
--region us-west-2 \
--profile adminuser
```

Etapa 1.2: Gravação de registros de amostra no stream de entrada

Nesta etapa, você executa o código Python para gerar continuamente registros de amostra e gravar no stream `ExampleInputStream`.

```
{"x": 7.921782426109737, "y": 8.746265312709893, "is_hot": "N"}
{"x": 0.722248626580026, "y": 4.648868803193405, "is_hot": "Y"}
```

1. Instale o Python e o `pip`.

   Para obter informações sobre como instalar o Python, consulte o site do Python.

   Você pode instalar dependências usando o pip. Para obter informações sobre como instalar o pip, consulte Installation no site do pip.
2. Execute o código do Python a seguir. Esse código faz o seguinte:

   - Gera um ponto de acesso em potencial em algum lugar do plano (X, Y).
   - Gera um conjunto de 1.000 pontos para cada ponto de acesso. Desses pontos, 20% são agrupados em torno do ponto de acesso. Os demais são gerados aleatoriamente em todo o espaço.
   - O comando `put-record` grava os registros JSON no streaming.

   > **Important**
   >
   > Não faça upload desse arquivo em um servidor web, porque ele contém suas credenciais da AWS.

```
import json
```

```
from pprint import pprint
import random
import time
import boto3

STREAM_NAME = "ExampleInputStream"


def get_hotspot(field, spot_size):
    hotspot = {
        "left": field["left"] + random.random() * (field["width"] - spot_size),
        "width": spot_size,
        "top": field["top"] + random.random() * (field["height"] - spot_size),
        "height": spot_size,
    }
    return hotspot


def get_record(field, hotspot, hotspot_weight):
    rectangle = hotspot if random.random() < hotspot_weight else field
    point = {
        "x": rectangle["left"] + random.random() * rectangle["width"],
        "y": rectangle["top"] + random.random() * rectangle["height"],
        "is_hot": "Y" if rectangle is hotspot else "N",
    }
    return {"Data": json.dumps(point), "PartitionKey": "partition_key"}


def generate(
    stream_name, field, hotspot_size, hotspot_weight, batch_size, kinesis_client
):
    """
    Generates points used as input to a hotspot detection algorithm.
    With probability hotspot_weight (20%), a point is drawn from the hotspot;
    otherwise, it is drawn from the base field. The location of the hotspot
    changes for every 1000 points generated.
    """
    points_generated = 0
    hotspot = None
    while True:
        if points_generated % 1000 == 0:
            hotspot = get_hotspot(field, hotspot_size)
        records = [
            get_record(field, hotspot, hotspot_weight) for _ in range(batch_size)
```

```
        ]
        points_generated += len(records)
        pprint(records)
        kinesis_client.put_records(StreamName=stream_name, Records=records)

        time.sleep(0.1)


if __name__ == "__main__":
    generate(
        stream_name=STREAM_NAME,
        field={"left": 0, "width": 10, "top": 0, "height": 10},
        hotspot_size=1,
        hotspot_weight=0.2,
        batch_size=10,
        kinesis_client=boto3.client("kinesis"),
    )
```

Próxima etapa

Etapa 2: criar o aplicativo Kinesis Data Analytics

Etapa 2: criar o aplicativo Kinesis Data Analytics

Nesta seção do Exemplo de hotspots, você cria um aplicativo Kinesis Data Analytics da seguinte forma:

- Configure a entrada do aplicativo para usar o fluxo de dados do Kinesis criado por você como a origem do streaming na Etapa 1.
- Use o código do aplicativo fornecido no AWS Management Console.

Para criar um aplicativo.

1. Crie um aplicativo Kinesis Data Analytics seguindo as etapas 1, 2 e 3 no exercício Conceitos básicos (consulte Etapa 3.1: criar um aplicativo).

   Na configuração de origem, faça o seguinte:

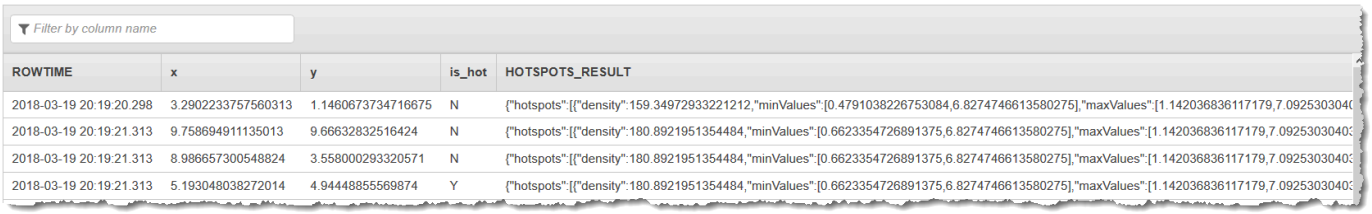
   - Especifique a origem do streaming criado por você em the section called “Etapa 1: criar streamings”.

- Depois que o console inferir o esquema, edite-o. Verifique se os tipos de coluna `x` e `yDOUBLE` estão definidos como e se o tipo de coluna `IS_HOT` está definido como `VARCHAR`.

2. Use o seguinte código de aplicativo (você pode colar esse código no editor SQL):

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM" (
    "x" DOUBLE,
    "y" DOUBLE,
    "is_hot" VARCHAR(4),
    HOTSPOTS_RESULT VARCHAR(10000)
);
CREATE OR REPLACE PUMP "STREAM_PUMP" AS
    INSERT INTO "DESTINATION_SQL_STREAM"
    SELECT "x", "y", "is_hot", "HOTSPOTS_RESULT"
    FROM TABLE (
        HOTSPOTS(
            CURSOR(SELECT STREAM "x", "y", "is_hot" FROM "SOURCE_SQL_STREAM_001"),
            1000,
            0.2,
            17)
    );
```

3. Execute o código SQL e analise os resultados.



Filter by column name

ROWTIME	x	y	is_hot	HOTSPOTS_RESULT
2018-03-19 20:19:20.298	3.2902233757560313	1.1460673734716675	N	{"hotspots":[{"density":159.34972933221212,"minValues":[0.4791038226753084,6.8274746613580275],"maxValues":[1.142036836117179,7.0925303040
2018-03-19 20:19:21.313	9.758694911135013	9.66632832516424	N	{"hotspots":[{"density":180.8921951354484,"minValues":[0.6623354726891375,6.8274746613580275],"maxValues":[1.142036836117179,7.0925303040
2018-03-19 20:19:21.313	8.986657300548824	3.558000293320571	N	{"hotspots":[{"density":180.8921951354484,"minValues":[0.6623354726891375,6.8274746613580275],"maxValues":[1.142036836117179,7.0925303040
2018-03-19 20:19:21.313	5.193048038272014	4.94448855569874	Y	{"hotspots":[{"density":180.8921951354484,"minValues":[0.6623354726891375,6.8274746613580275],"maxValues":[1.142036836117179,7.0925303040

Próxima etapa

[Etapa 3: configurar a saída do aplicativo]

Etapa 3: configurar a saída do aplicativo

A esta altura do [Exemplo de hotspots], você tem o código do aplicativo do Amazon Kinesis Data Analytics descobrindo hotspots significativos em uma origem de streaming e atribuindo uma pontuação de calor a cada um deles.

Você já pode enviar o resultado do aplicativo do stream do aplicativo de entrada para um destino externo, outro fluxo de dados do Kinesis (`ExampleOutputStream`). Em seguida, você pode analisar as pontuações do ponto de acesso e determinar qual é um limite apropriado para o calor do ponto de acesso. É possível estender mais este aplicativo para gerar alertas.

Para configurar a saída do aplicativo

1. Abra o console do Kinesis Data Analytics em https://console.aws.amazon.com/kinesisanalytics.
2. No editor SQL, escolha Destination ou Add a destination no painel do aplicativo.
3. Na página Add a destination (Adicionar um destino), escolha Select from your streams (Selecionar um dos fluxos). Em seguida, escolha o fluxo `ExampleOutputStream` criado na seção anterior.

   Agora, você tem um destino externo, em que o Amazon Kinesis Data Analytics mantém quaisquer registros que o aplicativo gravar no fluxo no aplicativo `DESTINATION_SQL_STREAM`.
4. Como opção, configure o AWS Lambda para monitorar o stream `ExampleOutputStream` e enviar alertas. Para obter mais informações, consulte Como usar a função do Lambda como saída. Você também pode analisar os registros gravados pelo Kinesis Data Analytics no destino externo, o stream `ExampleOutputStream` do Kinesis, como descrito em Etapa 4: verificar a saída do aplicativo.

Próxima etapa

Etapa 4: verificar a saída do aplicativo

Etapa 4: verificar a saída do aplicativo

Nesta seção do Exemplo dos hotspots, você configura um aplicativo web que exibe as informações do ponto de acesso em um controle Scalable Vector Graphics (SVG – Gráfico vetorial escalável).

1. Crie um arquivo denominado `index.html` com o conteúdo a seguir.

```
<!doctype html>
<html lang=en>
<head>
    <meta charset=utf-8>
    <title>hotspots viewer</title>

    <style>
```

```
    #visualization {
      display: block;
      margin: auto;
    }

    .point {
      opacity: 0.2;
    }

    .hot {
      fill: red;
    }

    .cold {
      fill: blue;
    }

    .hotspot {
      stroke: black;
      stroke-opacity: 0.8;
      stroke-width: 1;
      fill: none;
    }
    </style>
    <script src="https://sdk.amazonaws.com/js/aws-sdk-2.202.0.min.js"></script>
    <script src="https://d3js.org/d3.v4.min.js"></script>
</head>
<body>
<svg id="visualization" width="600" height="600"></svg>
<script src="hotspots_viewer.js"></script>
</body>
</html>
```

2. Crie um arquivo no mesmo diretório chamado `hotspots_viewer.js` com o conteúdo a seguir. Forneça as credenciais e nome do stream de saída nas variáveis fornecidas.

```
// Visualize example output from the Kinesis Analytics hotspot detection algorithm.
// This script assumes that the output stream has a single shard.

// Modify this section to reflect your AWS configuration
var awsRegion = "",        // The  where your Kinesis Analytics application is
 configured.
```

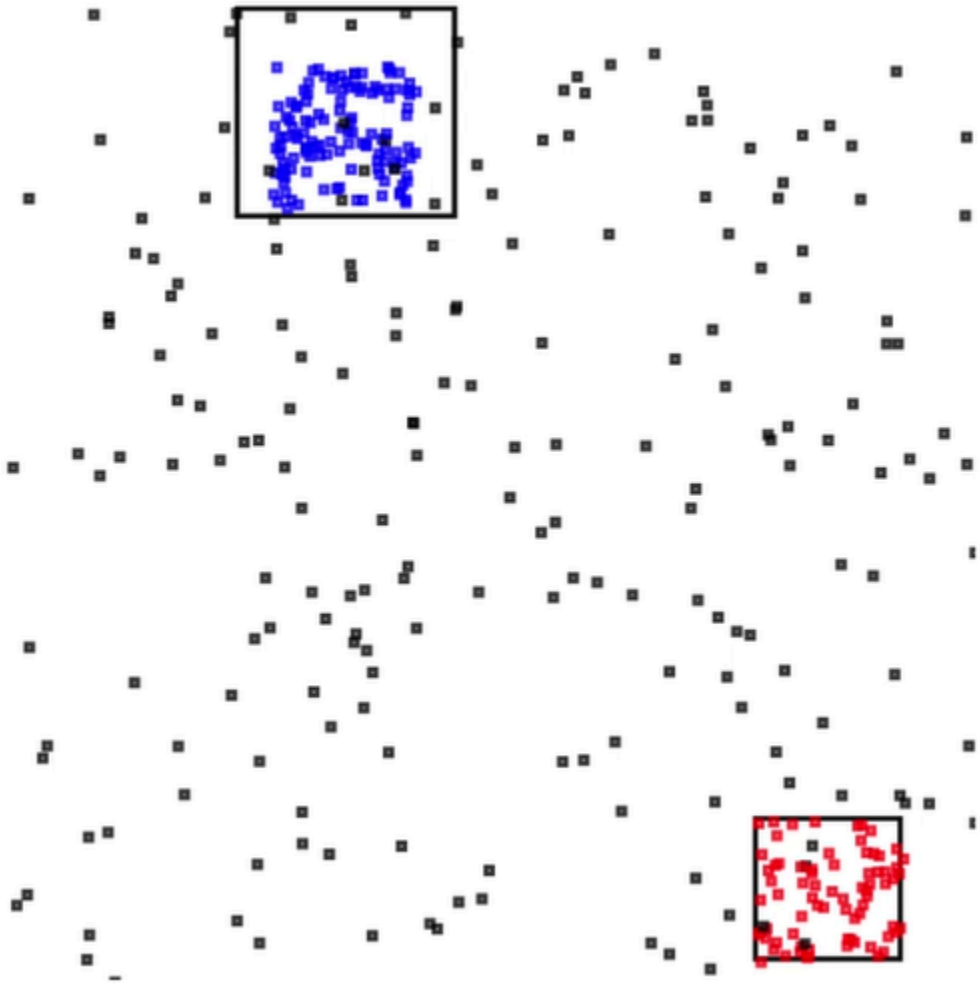
```
    accessKeyId = "",      // Your Access Key ID
    secretAccessKey = "",  // Your Secret Access Key
    outputStream = "";     // The name of the Kinesis Stream where the output from
 the HOTSPOTS function is being written

// The variables in this section should reflect way input data was generated and
 the parameters that the HOTSPOTS
// function was called with.
var windowSize = 1000, // The window size used for hotspot detection
    minimumDensity = 40,  // A filter applied to returned hotspots before
 visualization
    xRange = [0, 10],  // The range of values to display on the x-axis
    yRange = [0, 10];  // The range of values to display on the y-axis

////////////////////////////////////////////////////////////////////////////////
// D3 setup
////////////////////////////////////////////////////////////////////////////////

var svg = d3.select("svg"),
    margin = {"top": 20, "right": 20, "bottom": 20, "left": 20},
    graphWidth = +svg.attr("width") - margin.left - margin.right,
    graphHeight = +svg.attr("height") - margin.top - margin.bottom;

// Return the linear function that maps the segment [a, b] to the segment [c, d].
function linearScale(a, b, c, d) {
    var m = (d - c) / (b - a);
    return function(x) {
        return c + m * (x - a);
    };
}

// helper functions to extract the x-value from a stream record and scale it for
 output
var xValue = function(r) { return r.x; },
    xScale = linearScale(xRange[0], xRange[1], 0, graphWidth),
    xMap = function(r) { return xScale(xValue(r)); };

// helper functions to extract the y-value from a stream record and scale it for
 output
var yValue = function(r) { return r.y; },
    yScale = linearScale(yRange[0], yRange[1], 0, graphHeight),
    yMap = function(r) { return yScale(yValue(r)); };
```

```
// a helper function that assigns a CSS class to a point based on whether it was
 generated as part of a hotspot
var classMap = function(r) { return r.is_hot == "Y" ? "point hot" : "point
 cold"; };

var g = svg.append("g")
    .attr("transform", "translate(" + margin.left + "," + margin.top + ")");

function update(records, hotspots) {

    var points = g.selectAll("circle")
        .data(records, function(r) { return r.dataIndex; });

    points.enter().append("circle")
        .attr("class", classMap)
        .attr("r", 3)
        .attr("cx", xMap)
        .attr("cy", yMap);

    points.exit().remove();

    if (hotspots) {
        var boxes = g.selectAll("rect").data(hotspots);

        boxes.enter().append("rect")
            .merge(boxes)
            .attr("class", "hotspot")
            .attr("x", function(h) { return xScale(h.minValues[0]); })
            .attr("y", function(h) { return yScale(h.minValues[1]); })
            .attr("width", function(h) { return xScale(h.maxValues[0]) -
 xScale(h.minValues[0]); })
            .attr("height", function(h) { return yScale(h.maxValues[1]) -
 yScale(h.minValues[1]); });

        boxes.exit().remove();
    }
}

////////////////////////////////////////////////////////////////////////////////
// Use the AWS SDK to pull output records from Kinesis and update the visualization
////////////////////////////////////////////////////////////////////////////////

var kinesis = new AWS.Kinesis({
    "region": awsRegion,
```

```
    "accessKeyId": accessKeyId,
    "secretAccessKey": secretAccessKey
});

var textDecoder = new TextDecoder("utf-8");

// Decode an output record into an object and assign it an index value
function decodeRecord(record, recordIndex) {
    var record = JSON.parse(textDecoder.decode(record.Data));
    var hotspots_result = JSON.parse(record.HOTSPOTS_RESULT);
    record.hotspots = hotspots_result.hotspots
        .filter(function(hotspot) { return hotspot.density >= minimumDensity});
    record.index = recordIndex
    return record;
}

// Fetch a new records from the shard iterator, append them to records, and update
 the visualization
function getRecordsAndUpdateVisualization(shardIterator, records, lastRecordIndex)
 {
    kinesis.getRecords({
        "ShardIterator": shardIterator
    }, function(err, data) {
        if (err) {
            console.log(err, err.stack);
            return;
        }

        var newRecords = data.Records.map(function(raw) { return decodeRecord(raw,
 ++lastRecordIndex); });
        newRecords.forEach(function(record) { records.push(record); });

        var hotspots = null;
        if (newRecords.length > 0) {
            hotspots = newRecords[newRecords.length - 1].hotspots;
        }

        while (records.length > windowSize) {
            records.shift();
        }

        update(records, hotspots);
```

```
        getRecordsAndUpdateVisualization(data.NextShardIterator, records,
 lastRecordIndex);
    });
}

// Get a shard iterator for the output stream and begin updating the visualization.
 Note that this script will only
// read records from the first shard in the stream.
function init() {
    kinesis.describeStream({
        "StreamName": outputStream
    }, function(err, data) {
        if (err) {
            console.log(err, err.stack);
            return;
        }

        var shardId = data.StreamDescription.Shards[0].ShardId;

        kinesis.getShardIterator({
            "StreamName": outputStream,
            "ShardId": shardId,
            "ShardIteratorType": "LATEST"
        }, function(err, data) {
            if (err) {
                console.log(err, err.stack);
                return;
            }
            getRecordsAndUpdateVisualization(data.ShardIterator, [], 0);
        })
    });
}

// Start the visualization
init();
```

3. Com o código Python da primeira seção em execução, abra `index.html` em um navegador da web. As informações do ponto de conexão são exibidas na página, como mostrado a seguir.



Exemplos: alertas e erros

Esta seção fornece exemplos de aplicativos Kinesis Data Analytics que usam alertas e erros. Cada exemplo fornece instruções passo a passo e código para ajudar você a configurar e testar seu aplicativo Kinesis Data Analytics.

Tópicos

- Exemplo: criar alertas simples
- Exemplo: criar alertas controlados
- Exemplo: exploração do fluxo de erros no aplicativo

Exemplo: criar alertas simples

Nesse aplicativo Kinesis Data Analytics, a consulta é executada continuamente no fluxo no aplicativo criado pelo fluxo de demonstração. Para obter mais informações, consulte Consultas contínuas.

Se alguma linha mostrar uma alteração maior do que 1% no preço da ação, as linhas serão inseridas em outro fluxo de aplicativo. No exercício, você pode configurar a saída de aplicativos para que os resultados de um destino externo permaneçam. Em seguida, investigue mais os resultados. Por exemplo, use uma função AWS Lambda para processar registros e enviar alertas.

Para criar um aplicativo de alertas simples

1. Crie o aplicativo de análise como descrito no exercício Conceitos básicos do Kinesis Data Analytics.
2. No editor SQL no Kinesis Data Analytics, substitua o código de aplicativo pelo seguinte:

```
CREATE OR REPLACE STREAM "DESTINATION_SQL_STREAM"
           (ticker_symbol VARCHAR(4),
            sector        VARCHAR(12),
            change        DOUBLE,
            price         DOUBLE);

CREATE OR REPLACE PUMP "STREAM_PUMP" AS
   INSERT INTO "DESTINATION_SQL_STREAM"
      SELECT STREAM ticker_symbol, sector, change, price
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  (ABS(Change / (Price - Change)) * 100) > 1;
```

   A instrução `SELECT` no código do aplicativo filtra linhas em `SOURCE_SQL_STREAM_001` para mudanças nos preços das ações superiores a 1%. Em seguida, ela insere essas linhas em outro fluxo no aplicativo `DESTINATION_SQL_STREAM` usando uma bomba. Para obter mais informações sobre o padrão de codificação que explica o uso de bombas para inserir linhas em fluxos no aplicativo, consulte Código do aplicativo.
3. Escolha Save and run SQL.
4. Adicione um destino. Para fazer isso, escolha a guia Destination (Destino) no editor SQL ou Add a destination (Adicionar um destino) na página de detalhes do aplicativo.

   a. No editor SQL, selecione a guia Destination (Destino) e, em seguida, escolha Connect to a destination (Conectar-se a um destino).

Na página Connect to destination (Conectar-se ao destino) escolha Create New (Criar novo).

b. Escolha Go to Kinesis Streams.

c. No console do Amazon Kinesis Data Streams, crie um novo fluxo do Kinesis (por exemplo, `gs-destination`) com um fragmento. Aguarde até que o status do stream seja ACTIVE.

d. Volte para o console do Kinesis Data Analytics. Na página Connect to destination (Conectar-se ao destino), escolha o fluxo que você criou.

Se o fluxo não for exibido, atualize a página.

e. Escolha Save and continue.

Agora, você tem um destino externo, um fluxo de dados do Kinesis, em que o Kinesis Data Analytics mantém a saída de aplicativo no stream no aplicativo `DESTINATION_SQL_STREAM`.

5. Configure o AWS Lambda para monitorar o stream do Kinesis criado e invoque uma função do Lambda.

Para obter instruções, consulte Pré-processar dados usando uma função do Lambda.

Exemplo: criar alertas controlados

Nesse aplicativo Kinesis Data Analytics, a consulta é executada continuamente no fluxo no aplicativo que foi criado pelo fluxo de demonstração. Para obter mais informações, consulte Consultas contínuas. Se alguma linha mostrar uma alteração maior do que 1% no preço da ação, as linhas serão inseridas em outro fluxo de aplicativo. O aplicativo limita os alertas de forma que um alerta seja enviado imediatamente quando o preço da ação for alterado. No entanto, não será enviado mais de um alerta por minuto por símbolo de ação para o fluxo no aplicativo.

Para criar um aplicativo de alertas controlados

1. Crie o aplicativo Kinesis Data Analytics, como descrito no exercício Conceitos básicos do Kinesis Data Analytics.
2. No editor SQL no Kinesis Data Analytics, substitua o código de aplicativo pelo seguinte:

```
CREATE OR REPLACE STREAM "CHANGE_STREAM"
           (ticker_symbol VARCHAR(4),
            sector        VARCHAR(12),
```

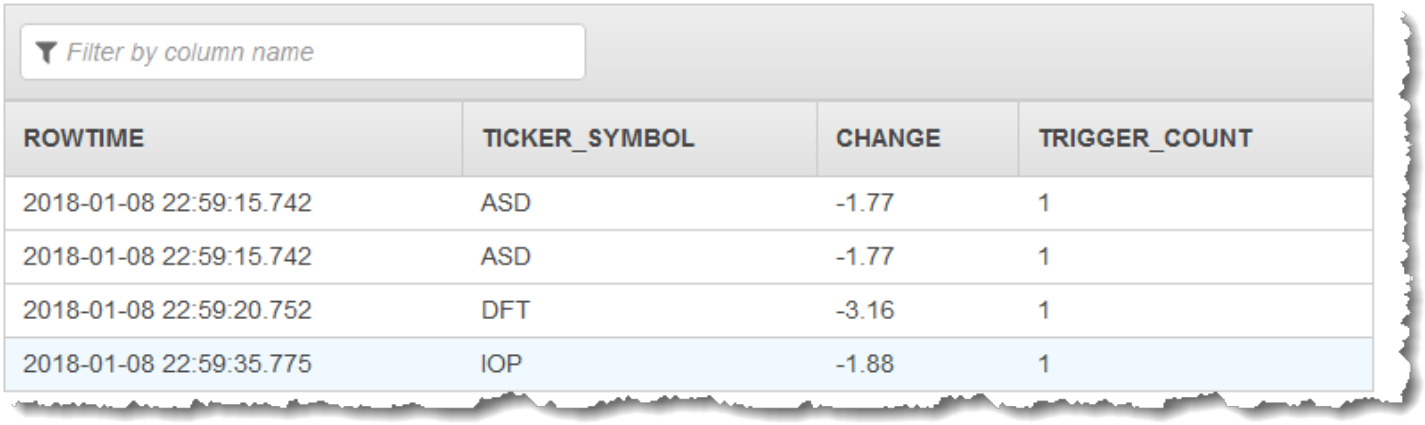
```
            change        DOUBLE,
            price         DOUBLE);

CREATE OR REPLACE PUMP "change_pump" AS
   INSERT INTO "CHANGE_STREAM"
      SELECT STREAM ticker_symbol, sector, change, price
      FROM   "SOURCE_SQL_STREAM_001"
      WHERE  (ABS(Change / (Price - Change)) * 100) > 1;

-- ** Trigger Count and Limit **
-- Counts "triggers" or those values that evaluated true against the previous where
 clause
-- Then provides its own limit on the number of triggers per hour per ticker symbol
 to what
-- is specified in the WHERE clause

CREATE OR REPLACE STREAM TRIGGER_COUNT_STREAM (
   ticker_symbol VARCHAR(4),
   change REAL,
   trigger_count INTEGER);

CREATE OR REPLACE PUMP trigger_count_pump AS INSERT INTO TRIGGER_COUNT_STREAM
SELECT STREAM ticker_symbol, change, trigger_count
FROM (
    SELECT STREAM ticker_symbol, change, COUNT(*) OVER W1 as trigger_count
    FROM "CHANGE_STREAM"
    --window to perform aggregations over last minute to keep track of triggers
    WINDOW W1 AS (PARTITION BY ticker_symbol RANGE INTERVAL '1' MINUTE PRECEDING)
)
WHERE trigger_count >= 1;
```

A instrução SELECT no código de aplicativo filtra as linhas no SOURCE_SQL_STREAM_001 por alterações de preços de ações maiores de 1% e insere essas linhas em outro fluxo no aplicativo CHANGE_STREAM usando uma bomba.

Em seguida, o aplicativo cria um segundo fluxo denominado TRIGGER_COUNT_STREAM para os alertas controlados. A segunda consulta seleciona registros a partir de uma janela que salta para a frente toda vez que um registro é admitido nela, de forma que apenas um registro por índice de ações por minuto seja gravado no fluxo.

3. Escolha Save and run SQL.

O exemplo apresenta a saída de um fluxo para `TRIGGER_COUNT_STREAM` que é semelhante ao seguinte:


Filter by column name

ROWTIME	TICKER_SYMBOL	CHANGE	TRIGGER_COUNT
2018-01-08 22:59:15.742	ASD	-1.77	1
2018-01-08 22:59:15.742	ASD	-1.77	1
2018-01-08 22:59:20.752	DFT	-3.16	1
2018-01-08 22:59:35.775	IOP	-1.88	1

Exemplo: exploração do fluxo de erros no aplicativo

O Amazon Kinesis Data Analytics fornece um fluxo de erros de aplicativo para cada aplicativo que você criou. Todas as linhas que o aplicativo não pode processar são enviadas a esse fluxo de erros. É recomendável manter os dados do fluxo de erros em um destino externo para que você possa investigar.

Execute os seguintes exercícios no console. Nestes exemplos, você insere erros na configuração de entrada editando o esquema inferido pelo processo de descoberta e verifica as linhas enviadas ao fluxo de erros.

Tópicos

- Inserção de erro de análise
- Inserção de erro de divisão por zero

Inserção de erro de análise

Neste exercício, você insere um erro de análise.

1. Crie o aplicativo Kinesis Data Analytics como descrito no exercício Conceitos básicos do Kinesis Data Analytics.
2. Na página de detalhes do aplicativo, escolha Connect streaming data (Conectar dados de streaming).

3. Se você fez o exercício em Conceitos básicos, há um fluxo de demonstração (`kinesis-analytics-demo-stream`) na sua conta. Na página Connect to source (Conectar-se à fonte), escolha este fluxo de demonstração.
4. O Kinesis Data Analytics faz uma amostra no fluxo de demonstração para inferir um esquema para o fluxo de entrada de aplicativo criado. O console mostra o esquema inferido e os dados de exemplo na guia Formatted stream sample.
5. Em seguida, edite o esquema e modifique o tipo de coluna para inserir o erro de análise. Escolha Edit schema (Editar esquema).
6. Altere o tipo de coluna `TICKER_SYMBOL` de `VARCHAR(4)` para `INTEGER`.

   Agora que o tipo de coluna no esquema da aplicação que é criado está inválido, o Kinesis Data Analytics não pode trazer dados para o stream do aplicativo. Em vez disso, ele envia linhas para o fluxo de erros.
7. Escolha Save schema.
8. Escolha Refresh schema samples.

   Observe que não há linhas no exemplo Formatted stream. No entanto, a guia Error stream mostra os dados com uma mensagem de erro. A guia Error stream mostra os dados enviados ao fluxo de erros de aplicativo.

   Como você mudou o tipo de dados da coluna, o Kinesis Data Analytics não conseguiu extrair os dados no fluxo de entrada no aplicativo. Ele enviou os dados para o fluxo de erros.

Inserção de erro de divisão por zero

Neste exercício, você atualiza o código do aplicativo para introduzir um erro de tempo de execução (divisão por zero). Observe que o Amazon Kinesis Data Analytics envia as linhas resultantes para o fluxo de erros no aplicativo, e não para o fluxo no aplicativo `DESTINATION_SQL_STREAM` no qual os resultados devem ser gravados.

1. Crie o aplicativo Kinesis Data Analytics como descrito no exercício Conceitos básicos do Kinesis Data Analytics.

   Verifique os resultados na guia Real-time analytics da seguinte forma:

   Sour
2. Atualize a instrução `SELECT` no código do aplicativo para inserir a divisão por zero. Por exemplo:

```
SELECT STREAM ticker_symbol, sector, change, (price / 0) as ProblemColumn
FROM "SOURCE_SQL_STREAM_001"
WHERE sector SIMILAR TO '%TECH%';
```

3. Execute o aplicativo.

   Como ocorre o erro de tempo de execução de divisão por zero, em vez de gravar resultados no `DESTINATION_SQL_STREAM`, o Kinesis Data Analytics envia as linhas ao fluxo de erros no aplicativo. Na guia Real-time analytics (Análise em tempo real), escolha o fluxo de erros. Em seguida, você poderá ver as linhas no fluxo de erros de aplicativo.

Exemplos: aceleradores de soluções

O site Soluções AWS tem modelos do AWS CloudFormation disponíveis que você pode usar para criar rapidamente soluções completas de dados de streaming.

Os seguintes modelos estão disponíveis:

Informações em tempo real sobre a atividade Conta da AWS

Essa solução registra e visualiza as métricas de acesso a recursos e uso para sua(s) Conta da AWS(s) em tempo real. Para obter mais informações, consulte Insights em tempo real sobre a atividade Conta da AWS.

Monitoramento de dispositivo AWS IoT em tempo real com o Kinesis Data Analytics

Esta solução coleta, processa, analisa e visualiza dados de conectividade e atividade de dispositivos da IoT em tempo real. Para obter mais informações, consulte Monitoramento de dispositivos AWS IoT em tempo real com o Kinesis Data Analytics.

Análise da web em tempo real com o Kinesis Data Analytics

Esta solução coleta, processa, analisa e visualiza os dados do fluxo de cliques do site em tempo real. Para obter mais informações, consulte Análises da web em tempo real com o Kinesis Data Analytics.

Solução Amazon Connected Vehicle

Esta solução coleta, processa, analisa e visualiza dados de IoT de veículos em tempo real. Para obter mais informações, consulte Solução Amazon Connected Vehicle.

Segurança em

A segurança na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficiará de uma arquitetura de data center e rede criada para atender aos requisitos das organizações mais sensíveis à segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O modelo de responsabilidade compartilhada descreve isto como segurança da nuvem e segurança na nuvem.

- Segurança da nuvem — AWS é responsável por proteger a infraestrutura que executa AWS os serviços na AWS nuvem. AWS também fornece serviços que você pode usar com segurança. A eficácia da nossa segurança é regularmente testada e verificada por auditores de terceiros como parte dos Programas de conformidade da AWS. Para saber mais sobre os programas de conformidade que se aplicam ao , consulte AWS Serviços no escopo pelo programa de conformidade.
- Segurança na nuvem — Sua responsabilidade é determinada pelo AWS serviço que você usa. Você também é responsável por outros fatores, como a confidencialidade de seus dados, os requisitos da sua organização, leis e regulamentos aplicáveis.

Esta documentação ajuda a entender como aplicar o modelo de responsabilidade compartilhada ao usar o . Os tópicos a seguir mostram como configurar o para atender aos seus objetivos de segurança e conformidade. Você também aprenderá como usar outros serviços da Amazon que podem ajudar você a monitorar e proteger seus recursos.

Tópicos

Amazon Kinesis Data Analytics para aplicativos SQL

Você pode proteger seus dados usando ferramentas fornecidas pela AWS. O Kinesis Data Analytics pode trabalhar com serviços que oferecem suporte à criptografia de dados, incluindo Kinesis Data Streams, Firehose e Amazon S3.

Criptografia de dados no Kinesis Data Analytics

Criptografia em repouso

Observe o seguinte sobre a criptografia de dados em repouso com o Kinesis Data Analytics:

- Você pode criptografar dados no stream de dados de entrada do Kinesis usando o. StartStreamEncryption Para obter mais informações, consulte O que é criptografia do lado do servidor para o Kinesis Data Streams?.
- Os dados de saída podem ser criptografados em repouso usando o Firehose para armazenar dados em um bucket criptografado do Amazon S3. É possível especificar a chave de criptografia que o bucket do Amazon S3 utiliza. Para obter mais informações, consulte Protecting Data Using Server-Side Encryption with KMS–Managed Keys (SSE-KMS).
- O código do aplicativo é criptografado em repouso.
- Os dados de referência do aplicativo são criptografados em repouso.

Criptografia em trânsito

O Kinesis Data Analytics criptografa todos os dados em trânsito. A criptografia em trânsito é habilitada para todos os aplicativos do Kinesis Data Analytics e não pode ser desabilitada.

O Kinesis Data Analytics criptografa dados em trânsito nos seguintes cenários:

- Dados em trânsito do Kinesis Data Streams para o Kinesis Data Analytics.
- Dados em trânsito entre componentes internos no Kinesis Data Analytics.
- Dados em trânsito entre o Kinesis Data Analytics e o Firehose.

Gerenciamento de chaves

A criptografia de dados no Kinesis Data Analytics usa chaves gerenciadas pelo serviço. Chaves gerenciadas pelo cliente não são compatíveis.

Gerenciamento de identidade e acesso no Kinesis Data Analytics

O Amazon Kinesis Data Analytics precisa de permissões para ler registros em uma origem de streaming especificada na configuração de entrada do aplicativo. O Amazon Kinesis Data Analytics também precisa de permissões para gravar a saída do aplicativo nos fluxos especificados na configuração de saída do aplicativo.

Você pode conceder essas permissões criando um perfil do IAM que o Amazon Kinesis Data Analytics possa assumir. As permissões que você concede a essa função determinam o que o Amazon Kinesis Data Analytics pode fazer quando o serviço assume a função.

> **Note**
>
> As informações desta seção serão úteis se você quiser criar um perfil do IAM por sua conta. Quando você cria um aplicativo no console do Amazon Kinesis Data Analytics, este pode criar um perfil do IAM para você nesse momento. O console usa a seguinte convenção de nomenclatura para os perfis do IAM criados:
>
> ```
> kinesis-analytics-ApplicationName
> ```
>
> Depois que a função for criada, você poderá analisar a função e as políticas anexadas no console do IAM;.

Cada perfil do IAM tem duas políticas anexadas. Na política de confiança, especifique quem pode assumir a função. Na política de permissões (pode haver uma ou mais), você especifica as permissões que deseja conceder a essa função. As seções a seguir descrevem essas políticas, que você pode usar ao criar um perfil do IAM.

Política de confiança

Para conceder permissões ao Amazon Kinesis Data Analytics para assumir uma função para acessar origens de streaming ou fontes de referência, anexe a seguinte política de confiança a um perfil do IAM:

```
{
```

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

Política de permissões

Se você estiver criando um perfil do IAM para permitir que o Amazon Kinesis Data Analytics leia uma origem de streaming do aplicativo, deverá conceder permissões para ações de leitura relevantes. Dependendo da sua fonte (por exemplo, um stream do Kinesis, um stream de entrega do Firehose ou uma fonte de referência em um bucket do Amazon S3), você pode anexar a seguinte política de permissões.

Política de permissões para leitura de um stream do Kinesis

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadInputKinesis",
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStream",
                "kinesis:GetShardIterator",
                "kinesis:GetRecords",
                "kinesis:ListShards"
            ],
            "Resource": [
                "arn:aws:kinesis:aws-region:aws-account-id:stream/inputStreamName"
            ]
        }
    ]
}
```

Política de permissões para leitura de um stream de entrega do Firehose

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ReadInputFirehose",
            "Effect": "Allow",
            "Action": [
                "firehose:DescribeDeliveryStream",
                "firehose:Get*"
            ],
            "Resource": [
                "arn:aws:firehose:aws-region:aws-account-
id:deliverystream/inputFirehoseName"
            ]
        }
    ]
}
```

> **Note**
>
> A permissão `firehose:Get*` se refere a um acessador interno usado pelo Kinesis Data Analytics para acessar o stream. Não há acessador público para um stream de entrega do Firehose.

Se você instruir o Amazon Kinesis Data Analytics a gravar a saída em destinos externos na configuração de saída do aplicativo, será necessário conceder a seguinte permissão ao perfil do IAM.

Política de permissões para gravação em um stream do Kinesis

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "WriteOutputKinesis",
            "Effect": "Allow",
            "Action": [
                "kinesis:DescribeStream",
                "kinesis:PutRecord",
```

```
                "kinesis:PutRecords"
            ],
            "Resource": [
                "arn:aws:kinesis:aws-region:aws-account-id:stream/output-stream-name"
            ]
        }
    ]
}
```

Política de permissões para gravação em um stream de entrega do Firehose

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "WriteOutputFirehose",
            "Effect": "Allow",
            "Action": [
                "firehose:DescribeDeliveryStream",
                "firehose:PutRecord",
                "firehose:PutRecordBatch"
            ],
            "Resource": [
                "arn:aws:firehose:aws-region:aws-account-id:deliverystream/output-
firehose-name"
            ]
        }
    ]
}
```

Política de permissões para leitura de uma fonte de dados de referência em um bucket do Amazon S3

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:Get*",
        "s3:List*"
```

```
      ],
      "Resource": "*"
    }
  ]
}
```

Prevenção do problema “confused deputy” entre serviços

Em AWS, a representação entre serviços pode ocorrer quando um serviço (o serviço de chamada) chama outro serviço (o serviço chamado). O serviço de chamadas pode ser manipulado para agir sobre os recursos de outro cliente, mesmo que não deva ter as permissões adequadas, resultando no confuso problema do delegado.

Para evitar que delegados confusos, AWS fornece ferramentas que ajudam você a proteger seus dados em todos os serviços usando diretores de serviços que receberam acesso aos recursos em sua conta. Esta seção se concentra na prevenção de “confused deputy” entre serviços específicos do Kinesis Data Analytics. Você pode aprender mais sobre esse tópico na seção The confused deputy problem do Guia do usuário do IAM.

No contexto do Kinesis Data Analytics for SQL, recomendamos usar as chaves de contexto de condição global aws SourceArnSourceAccount: e aws: na política de confiança de sua função para limitar o acesso à função somente às solicitações geradas pelos recursos esperados.

Use `aws:SourceArn` se quiser que apenas um recurso seja associado ao acesso entre serviços. Use `aws:SourceAccount` se quiser permitir que qualquer recurso nessa conta seja associado ao uso entre serviços.

O valor de `aws:SourceArn` deve ser o ARN do recurso usado pelo Kinesis Data Analytics, que é especificado com o seguinte formato: `arn:aws:kinesisanalytics:region:account:resource`.

A abordagem recomendada para o problema de “confused deputy” é usar a chave de contexto de condição global `aws:SourceArn` com o ARN completo do recurso.

Se você não souber o ARN completo do recurso ou estiver especificando vários recursos, use a chave `aws:SourceArn` com caracteres curingas (*) para as partes desconhecidas do ARN. Por exemplo: `arn:aws:kinesisanalytics::111122223333:*`.

Embora a maioria das ações na API do Kinesis Data Analytics for SQL AddApplicationInput, CreateApplicationcomo, DeleteApplicatione, sejam feitas no contexto de aplicativos específicos,

DiscoverInputSchemaa ação não é executada no contexto de nenhum aplicativo. Isso significa que a função usada nessa ação não deve especificar totalmente um recurso na chave de condição `SourceArn`. Veja a seguir um exemplo que usa um ARN curinga:

```
{
   ...
   "ArnLike":{
      "aws:SourceArn":"arn:aws:kinesisanalytics:us-east-1:123456789012:*"
   }
   ...
}
```

O perfil padrão gerado pelo Kinesis Data Analytics para SQL usa esse caractere curinga. Isso garante que a descoberta do esquema de entrada funcione perfeitamente na experiência do console. No entanto, recomendamos editar a Política de Confiança para usar um ARN completo depois de descobrir o esquema para implementar uma mitigação completa do "confused deputy".

As políticas de funções que você fornece ao Kinesis Data Analytics, bem como as políticas de confiança das funções geradas para você, podem usar as chaves de condição aws SourceArn: e aws SourceAccount:.

Para se proteger contra o problema do confused deputy, execute as seguintes tarefas:

Para se proteger contra o problema do confused deputy

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Selecione Perfis e, em seguida, selecione o perfil que você deseja modificar.
3. Selecione Edit trust policy (Editar política de confiança).
4. Na página Editar política de confiança, substitua a política JSON padrão por uma política que usa uma ou ambas as chaves de contexto de condição global `aws:SourceArn` e `aws:SourceAccount`. Veja os exemplos de políticas a seguir:
5. Selecione Update policy.

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

```
         "Principal":{
            "Service":"kinesisanalytics.amazonaws.com"
         },
         "Action":"sts:AssumeRole",
         "Condition":{
            "StringEquals":{
               "aws:SourceAccount":"Account ID"
            },
            "ArnEquals":{
               "aws:SourceArn":"arn:aws:kinesisanalytics:us-
east-1:123456789012:application/my-app"
            }
         }
      }
   ]
}
```

Autenticação e controle de acesso para

O acesso ao requer credenciais. Essas credenciais devem ter permissões para acessar AWS recursos, como um aplicativo ou uma instância do Amazon Elastic Compute Cloud (Amazon EC2). As seções a seguir fornecem detalhes sobre como você pode usar o AWS Identity and Access Management (IAM) e o para garantir o acesso aos recursos.

Controle de acesso

É possível ter credenciais válidas para autenticar suas solicitações. No entanto, a menos que tenha permissões, não é possível criar nem acessar os recursos do . Por exemplo, você deve ter permissões para criar um cofre do aplicativo do .

As seções a seguir descrevem como gerenciar permissões para o . Recomendamos que você leia a visão geral primeiro.

- Visão geral do gerenciamento de permissões de acesso aos seus recursos do
- Uso de políticas baseadas em identidade (políticas do IAM) para
- Permissões de API: referência a ações, permissões e recursos

Autenticando com identidades

A autenticação é a forma como você faz login AWS usando suas credenciais de identidade. Você deve estar autenticado (conectado AWS) como o Usuário raiz da conta da AWS, como usuário do IAM ou assumindo uma função do IAM.

Você pode entrar AWS como uma identidade federada usando credenciais fornecidas por meio de uma fonte de identidade. AWS IAM Identity Center Usuários (IAM Identity Center), a autenticação de login único da sua empresa e suas credenciais do Google ou do Facebook são exemplos de identidades federadas. Quando você faz login como uma identidade federada, o administrador já configurou anteriormente a federação de identidades usando perfis do IAM. Ao acessar AWS usando a federação, você está assumindo indiretamente uma função.

Dependendo do tipo de usuário que você é, você pode entrar no AWS Management Console ou no portal de AWS acesso. Para obter mais informações sobre como fazer login AWS, consulte Como fazer login Conta da AWS no Guia do Início de Sessão da AWS usuário.

Se você acessar AWS programaticamente, AWS fornece um kit de desenvolvimento de software (SDK) e uma interface de linha de comando (CLI) para assinar criptograficamente suas solicitações usando suas credenciais. Se você não usa AWS ferramentas, você mesmo deve assinar as solicitações. Para obter mais informações sobre como usar o método recomendado para assinar solicitações por conta própria, consulte Assinatura de solicitações de AWS API no Guia do usuário do IAM.

Independentemente do método de autenticação usado, também pode ser exigido que você forneça informações adicionais de segurança. Por exemplo, AWS recomenda que você use a autenticação multifator (MFA) para aumentar a segurança da sua conta. Para saber mais, consulte Autenticação multifator no Guia do usuário do AWS IAM Identity Center . Usar a autenticação multifator (MFA) na AWS no Guia do usuário do IAM.

Conta da AWS usuário root

Ao criar uma Conta da AWS, você começa com uma identidade de login que tem acesso completo a todos Serviços da AWS os recursos da conta. Essa identidade é chamada de usuário Conta da AWS raiz e é acessada fazendo login com o endereço de e-mail e a senha que você usou para criar a conta. É altamente recomendável não usar o usuário-raiz para tarefas diárias. Proteja as credenciais do usuário-raiz e use-as para executar as tarefas que somente ele pode executar. Para obter a lista completa das tarefas que exigem login como usuário-raiz, consulte Tarefas que exigem credenciais de usuário-raiz no Guia do usuário do IAM.

Identidade federada

Como prática recomendada, exija que usuários humanos, incluindo usuários que precisam de acesso de administrador, usem a federação com um provedor de identidade para acessar Serviços da AWS usando credenciais temporárias.

Uma identidade federada é um usuário do seu diretório de usuários corporativo, de um provedor de identidade da web AWS Directory Service, do diretório do Identity Center ou de qualquer usuário que acesse usando credenciais fornecidas Serviços da AWS por meio de uma fonte de identidade. Quando as identidades federadas são acessadas Contas da AWS, elas assumem funções, e as funções fornecem credenciais temporárias.

Para o gerenciamento de acesso centralizado, recomendamos usar o . AWS IAM Identity Center Você pode criar usuários e grupos no IAM Identity Center ou pode se conectar e sincronizar com um conjunto de usuários e grupos em sua própria fonte de identidade para uso em todos os seus Contas da AWS aplicativos. Para obter mais informações sobre o Centro de Identidade do IAM, consulte "What is IAM Identity Center?" (O que é o Centro de Identidade do IAM?) noAWS IAM Identity Center Guia do usuário do .

Grupos e usuários do IAM

Um usuário do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas para uma única pessoa ou aplicativo. Sempre que possível, recomendamos depender de credenciais temporárias em vez de criar usuários do IAM com credenciais de longo prazo, como senhas e chaves de acesso. No entanto, se você tiver casos de uso específicos que exijam credenciais de longo prazo com usuários do IAM, recomendamos alternar as chaves de acesso. Para obter mais informações, consulte Alterne as chaves de acesso regularmente para casos de uso que exijam credenciais de longo prazo no Guia do usuário do IAM.

Um grupo do IAM é uma identidade que especifica uma coleção de usuários do IAM. Não é possível fazer login como um grupo. É possível usar grupos para especificar permissões para vários usuários de uma vez. Os grupos facilitam o gerenciamento de permissões para grandes conjuntos de usuários. Por exemplo, você pode ter um grupo chamado IAMAdmins e atribuir a esse grupo permissões para administrar atributos do IAM.

Usuários são diferentes de perfis. Um usuário é exclusivamente associado a uma pessoa ou a uma aplicação, mas um perfil pode ser assumido por qualquer pessoa que precisar dele. Os usuários têm credenciais permanentes de longo prazo, mas os perfis fornecem credenciais temporárias. Para

saber mais, consulte Quando criar um usuário do IAM (em vez de um perfil) no Guia do usuário do IAM.

Perfis do IAM

Uma função do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas. Ele é semelhante a um usuário do IAM, mas não está associado a uma pessoa específica. Você pode assumir temporariamente uma função do IAM no AWS Management Console trocando de funções. Você pode assumir uma função chamando uma operação de AWS API AWS CLI ou usando uma URL personalizada. Para obter mais informações sobre métodos para o uso de perfis, consulte Usar perfis do IAM no Guia do usuário do IAM.

Perfis do IAM com credenciais temporárias são úteis nas seguintes situações:

- Acesso de usuário federado: para atribuir permissões a identidades federadas, você pode criar um perfil e definir permissões para ele. Quando uma identidade federada é autenticada, essa identidade é associada ao perfil e recebe as permissões definidas pelo mesmo. Para obter mais informações sobre perfis para federação, consulte Criar um perfil para um provedor de identidades de terceiros no Guia do usuário do IAM. Se você usar o IAM Identity Center, configure um conjunto de permissões. Para controlar o que suas identidades podem acessar após a autenticação, o IAM Identity Center correlaciona o conjunto de permissões a um perfil no IAM. Para obter informações sobre conjuntos de permissões, consulte Conjuntos de permissões noAWS IAM Identity Center Guia do usuário do .
- Permissões temporárias para usuários do IAM: um usuário ou um perfil do IAM pode assumir um perfil do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- Acesso entre contas: é possível usar um perfil do IAM para permitir que alguém (uma entidade principal confiável) em outra conta acesse recursos em sua conta. Os perfis são a principal forma de conceder acesso entre contas. No entanto, com alguns Serviços da AWS, você pode anexar uma política diretamente a um recurso (em vez de usar uma função como proxy). Para saber a diferença entre perfis e políticas baseadas em atributo para acesso entre contas, consulte Como os perfis do IAM diferem das políticas baseadas em atributo no Guia do usuário do IAM.
- Acesso entre serviços — Alguns Serviços da AWS usam recursos em outros Serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicações no Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso usando as permissões da entidade principal de chamada, usando um perfil de serviço ou uma função vinculada ao serviço.

- Sessões de acesso direto (FAS) — Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Encaminhar sessões de acesso.
- Perfil de serviço: um perfil de serviço é um perfil do IAMhttps://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do usuário do IAM.
- Função vinculada ao serviço — Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um. AWS service (Serviço da AWS) O serviço pode assumir o perfil de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em você Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para perfis vinculados ao serviço.

- Aplicativos em execução no Amazon EC2 — Você pode usar uma função do IAM para gerenciar credenciais temporárias para aplicativos que estão sendo executados em uma instância do EC2 e fazendo AWS CLI solicitações de API. AWS É preferível fazer isso a armazenar chaves de acesso na instância do EC2. Para atribuir uma AWS função a uma instância do EC2 e disponibilizá-la para todos os seus aplicativos, você cria um perfil de instância anexado à instância. Um perfil de instância contém a perfil e permite que os programas em execução na instância do EC2 obtenham credenciais temporárias. Para mais informações, consulte Usar um perfil do IAM para conceder permissões a aplicações em execução nas instâncias do Amazon EC2 no Guia do usuário do IAM.

Para saber se deseja usar as funções do IAM, consulte Quando criar um perfil do IAM (em vez de um usuário) no Guia do usuário do IAM.

Visão geral do gerenciamento de permissões de acesso aos seus recursos do

> **Warning**
>
> Para novos projetos, recomendamos que você use o novo Managed Service for Apache Flink Studio em vez de aplicativos SQL. O Managed Service for Apache Flink Studio combina facilidade de uso com recursos analíticos avançados, permitindo que você crie aplicativos sofisticados de processamento de stream em minutos.

Para conceder acesso, adicione as permissões aos seus usuários, grupos ou perfis:

- Usuários e grupos em AWS IAM Identity Center:

  Crie um conjunto de permissões. Siga as instruções em Criação de um conjunto de permissões no Guia do usuário do AWS IAM Identity Center .
- Usuários gerenciados no IAM com provedor de identidades:

  Crie um perfil para a federação de identidades. Siga as instruções em Criar um perfil para um provedor de identidades de terceiros (federação) no Guia do usuário do IAM.
- Usuários do IAM:
  - Crie um perfil que seu usuário possa assumir. Siga as instruções em Criação de um perfil para um usuário do IAM no Guia do usuário do IAM.
  - (Não recomendado) Vincule uma política diretamente a um usuário ou adicione um usuário a um grupo de usuários. Siga as instruções em Adição de permissões a um usuário (console) no Guia do usuário do IAM.

> **Note**
>
> Um administrador da conta (ou usuário administrador) é um usuário com privilégios de administrador. Para obter mais informações, consulte Práticas recomendadas de segurança no IAM no Guia do usuário do IAM.

Tópicos

- Recursos e operações
- Noções básicas sobre propriedade de recursos
- Gerenciamento do acesso aos recursos
- Especificar elementos da política: ações, efeitos e entidades principais
- Especificar condições em uma política

Recursos e operações

No , o primeiro recurso é um aplicativo. Em uma política, você usa um Nome de recurso da Amazon (ARN) para identificar o recurso a que a política se aplica.

Esses recursos têm nomes dos recursos da Amazon exclusivos associados, conforme mostrado na tabela a seguir.

Tipo de recurso	Formato do ARN
Aplicativo	`arn:aws:kinesisanalytics:` *`region`*`:`*`account-id`*`:application/` *`application-name`*

fornece um conjunto de operações para trabalhar com recursos. Para ver uma lista das operações disponíveis, consulte Ações.

Noções básicas sobre propriedade de recursos

Ele Conta da AWS possui os recursos que são criados na conta, independentemente de quem criou os recursos. Especificamente, o proprietário Conta da AWS do recurso é a entidade principal (ou seja, a conta raiz, um usuário ou uma função do IAM) que autentica a solicitação de criação do recurso. Os seguintes exemplos mostram como isso funciona:

- Se você usar as credenciais da sua conta raiz Conta da AWS para criar um aplicativo, você Conta da AWS é o proprietário do recurso. (No , o recurso é um aplicativo.)
- Se você criar um usuário no seu Conta da AWS e conceder permissões para criar um aplicativo para esse usuário, o usuário poderá criar um aplicativo. No entanto, seu Conta da AWS, ao qual o usuário pertence, é proprietário do recurso do aplicativo. É altamente recomendável que você conceda permissões aos perfis e não aos usuários.

- Se você criar uma função do IAM Conta da AWS com permissões para criar um aplicativo, qualquer pessoa que possa assumir a função poderá criar um aplicativo. Seu Conta da AWS, ao qual o usuário pertence, é proprietário do recurso do aplicativo.

Gerenciamento do acesso aos recursos

A política de permissões descreve quem tem acesso a quê. A seção a seguir explica as opções disponíveis para a criação de políticas de permissões.

> **Note**
>
> Esta seção discute o uso do IAM no contexto do . Não são fornecidas informações detalhadas sobre o serviço IAM. Para obter a documentação completa do IAM, consulte O que é o IAM? no Manual do usuário do IAM. Para obter informações sobre a sintaxe e as descrições da política do IAM, consulte a Referência da política JSON do IAM no Manual do usuário do IAM.

As políticas anexadas a uma identidade do IAM são chamadas de políticas baseadas em identidade (políticas do IAM). As políticas anexadas a um recurso são conhecidas como políticas baseadas em recurso. O suporta apenas políticas baseadas em identidade (políticas do IAM).

Tópicos

- Políticas baseadas em identidade (políticas do IAM)
- Políticas baseadas em recurso

Políticas baseadas em identidade (políticas do IAM)

Você pode anexar políticas a identidades do IAM. Por exemplo, você pode fazer o seguinte:

- Anexar uma política de permissões a um usuário ou grupo na conta: para conceder a um usuário permissões para criar um recurso, como um aplicativo, você pode anexar uma política de permissões a um usuário ou grupo ao qual o usuário pertença.
- Anexar uma política de permissões a uma função: você pode anexar uma política de permissões baseada em identidade a um perfil do IAM para conceder permissões entre contas. Por exemplo, o administrador da conta A pode criar uma função para conceder permissões entre contas a outra Conta da AWS (por exemplo, conta B) ou a um serviço da Amazon da seguinte forma:

1. Um administrador da Conta A cria um perfil do IAM e anexa uma política de permissões ao perfil que concede permissões em recursos da Conta A.
2. Um administrador da conta A anexa uma política de confiança ao perfil identificando a conta B como a entidade principal, que pode assumir a função.
3. O administrador da conta B pode delegar permissões para assumir o perfil a todos os usuários na conta B. Isso permite que os usuários na conta B criem ou acessem recursos na conta A. A entidade principal na política de confiança também pode ser um serviço principal, se você quiser conceder a um serviço da Amazon permissões para assumir o perfil.

Para obter mais informações sobre o uso do IAM para delegar permissões, consulte Gerenciamento de acesso no Guia do usuário do IAM.

Veja a seguir um exemplo de política que concede permissão para a ação `kinesisanalytics:CreateApplication` , o que é necessário para criar um aplicativo.

> **Note**
>
> Este é um exemplo de política introdutória. Quando você anexar a política ao usuário, o usuário poderá criar um aplicativo usando o AWS SDK AWS CLI ou. Mas o usuário precisará de mais permissões para configurar a entrada e a saída. Além disso, o usuário precisará de mais permissões ao usar o console. As seções mais recentes fornecem mais informações.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1473028104000",
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:CreateApplication"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Para mais informações sobre como usar políticas baseadas em identidade com o , consulte Uso de políticas baseadas em identidade (políticas do IAM) para . Para obter mais informações sobre usuários, grupos, funções e permissões, consulte Identidades (usuários, grupos e funções) no Guia do usuário do IAM.

Políticas baseadas em recurso

Outros serviços, como o Amazon S3, também aceitam políticas de permissões baseadas em recurso. Por exemplo, você pode anexar uma política a um bucket do S3 para gerenciar permissões de acesso a esse bucket. O não aceita políticas baseadas em recurso.

Especificar elementos da política: ações, efeitos e entidades principais

Para cada recurso do , o serviço define um conjunto de operações da API. Para conceder permissões a essas operações da API, o define um conjunto de ações que podem ser especificadas em uma política. Algumas operações da API podem exigir permissões para mais de uma ação a fim de realizar a operação da API. Para obter mais informações sobre os recursos e operações da API, consulte Recursos e operações e Ações.

Estes são os elementos de política mais básicos:

- Recurso: use um nome do recurso da Amazon (ARN) para identificar o recurso ao qual a política se aplica. Para obter mais informações, consulte Recursos e operações.
- Ação: você usa palavras-chave de ação para identificar operações de recursos que deseja permitir ou negar. Por exemplo, você pode usar `create` para permitir que os usuários criem um aplicativo.
- Efeito: você especifica o efeito, permitir ou negar, quando o usuário solicita a ação específica. Se você não conceder (permitir) explicitamente acesso a um recurso, o acesso estará implicitamente negado. Você também pode negar explicitamente o acesso a um recurso, para ter certeza de que um usuário não consiga acessá-lo, mesmo que uma política diferente conceda acesso.
- Principal: em políticas baseadas em identidade (políticas do IAM), o usuário ao qual a política é anexada é implicitamente a entidade principal. Para as políticas baseadas em recurso, você especifica quais usuários, contas, serviços ou outras entidades deseja que recebam permissões (aplica-se somente a políticas baseadas em recurso). O não aceita politicas baseadas em recurso.

Para saber mais sobre a sintaxe e as descrições de políticas do IAM, consulte a Referência da política JSON do IAM no Guia do usuário do IAM.

Para obter uma em tabela mostrando todas as operações da API e os recursos aos quais elas se aplicam, consulte Permissões de API: referência a ações, permissões e recursos.

Especificar condições em uma política

Ao conceder permissões, você pode usar a linguagem da política de acesso para especificar as condições quando uma política deve entrar em vigor. Por exemplo, é recomendável aplicar uma política somente após uma data específica. Para obter mais informações sobre como especificar condições em uma linguagem de política, consulte Condition no Guia do usuário do IAM.

Para expressar condições, você usa chaves de condição predefinidas. Não existem chaves de condição específicas do . No entanto, existem chaves AWS de condição abrangentes que você pode usar conforme apropriado. Para obter uma lista completa AWS de chaves abrangentes, consulte Chaves disponíveis para condições no Guia do usuário do IAM.

Uso de políticas baseadas em identidade (políticas do IAM) para

Veja a seguir exemplos de políticas baseadas em identidade que demonstram como um administrador de conta pode anexar políticas de permissões a identidades do IAM (isto é, usuários, grupos e perfis) e conceder permissões para realizar operações em recursos.

> **Important**
>
> Recomendamos que você analise primeiramente os tópicos introdutórios que explicam os conceitos básicos e as opções disponíveis para gerenciar o acesso aos recursos do . Para ter mais informações, consulte Visão geral do gerenciamento de permissões de acesso aos seus recursos do .

Tópicos

- Permissões necessárias para usar o console do
- Políticas gerenciadas pela Amazon (predefinidas) para
- Exemplos de política gerenciada pelo cliente

A seguir, um exemplo de uma política de permissões.

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

```
        {
            "Sid": "Stmt1473028104000",
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:CreateApplication"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

A política tem uma instrução:

- A primeira instrução concede permissões para uma ação do (`kinesisanalytics:CreateApplication`) em um recurso que está usando o nome de recurso da Amazon (ARN) para o aplicativo. O ARN, neste caso, especifica um caractere curinga (*) para indicar que a permissão é concedida para qualquer recurso.

Para obter uma tabela mostrando todas as operações da API e os recursos aos quais elas se aplicam, consulte Permissões de API: referência a ações, permissões e recursos.

Permissões necessárias para usar o console do

Para um usuário trabalhar com o console do , você deve conceder as permissões necessárias. Por exemplo, se você quiser que um usuário tenha permissões para criar aplicativo, conceda permissões que mostre a ele as origens de streaming na conta, para que o usuário possa configurar a entrada e a saída no console.

Recomendamos o seguinte:

- Use as políticas gerenciadas pela Amazon para conceder permissões de usuário. Para saber quais são as políticas disponíveis, consulte Políticas gerenciadas pela Amazon (predefinidas) para .
- Crie políticas personalizadas. Nesse caso, recomendamos que você examine o exemplo fornecido nesta seção. Para ter mais informações, consulte Exemplos de política gerenciada pelo cliente.

Políticas gerenciadas pela Amazon (predefinidas) para

AWS aborda muitos casos de uso comuns fornecendo políticas autônomas do IAM que são criadas e administradas pela AWS. Essas políticas gerenciadas pela Amazon concedem as permissões necessárias para casos de uso comuns para que você não precise investigar quais permissões são necessárias. Para obter mais informações, consulte Políticas gerenciadas pela Amazon no Guia do usuário do IAM.

As seguintes políticas gerenciadas pela Amazon, que você pode anexar aos usuários em sua conta, são específicas para:

- **`AmazonKinesisAnalyticsReadOnly`**: concede permissões para ações que permitem que um usuário liste aplicativos e analise a configuração de entrada/saída. Ele também concede permissões que permitem ao usuário visualizar uma lista de streams do Kinesis e streams de entrega do Firehose. Enquanto o aplicativo está em execução, o usuário pode visualizar dados de origem e resultados de análises em tempo real no console.

- **`AmazonKinesisAnalyticsFullAccess`**: concede permissões para todas as ações e todas as outras permissões para um usuário criar e gerenciar aplicativos. Entretanto, observe o seguinte:

  - Essas permissões não serão suficientes se o usuário quiser criar uma nova função do IAM no console (elas permitirão que o usuário selecione uma função existente). Se você quiser que o usuário possa criar um perfil do IAM no console, adicione a política gerenciada pela Amazon `IAMFullAccess`.

  - Um usuário deve ter permissão para a ação `iam:PassRole` para que possa especificar um perfil do IAM ao configurar o aplicativo. Esta política gerenciada pela Amazon concede ao usuário permissão para a ação `iam:PassRole` somente nos perfis do IAM que começam com o prefixo `service-role/kinesis-analytics`.

    Se o usuário quiser configurar o aplicativo com um perfil que não tem esse prefixo, primeiro é necessário conceder explicitamente permissões de usuário para a ação `iam:PassRole` no perfil específico.

Você também pode criar as próprias políticas do IAM personalizadas a fim de conceder permissões para ações e recursos. Você pode anexar essas políticas personalizadas a usuários ou grupos do que exijam essas permissões.

Exemplos de política gerenciada pelo cliente

Os exemplos desta seção apresentam um grupo de políticas de amostra que você pode anexar a um usuário. Se você for iniciante na criação de políticas, recomendamos primeiro criar um usuário em sua conta. Em seguida, anexe as políticas ao usuário sequencialmente, conforme descrito nas etapas desta seção. Em seguida, você pode usar o console para verificar os efeitos de cada política à medida que anexa a política ao usuário.

Inicialmente, o usuário não tem permissões e não pode fazer nada no console. À medida que você anexar políticas ao usuário, poderá verificar se o usuário pode executar várias ações no console.

Recomendamos o uso de duas janelas do navegador. Em uma janela, crie o usuário e conceda permissões. Na outra, faça login AWS Management Console usando as credenciais do usuário e verifique as permissões à medida que você as concede.

Para obter exemplos que mostrem como criar um perfil do IAM que possa ser usado como perfil de execução no seu aplicativo, consulte Criação de perfis do IAM no Guia do usuário do IAM.

Etapas de exemplo

- Etapa 1: Criar um usuário do IAM
- Etapa 2: Conceder ao usuário permissões para ações que não são específicas do
- Etapa 3: Permitir que o usuário visualize uma lista de aplicativos e os detalhes
- Etapa 4: Permitir que o usuário inicie um aplicativo específico
- Etapa 5: Permitir que o usuário crie um aplicativo do
- Etapa 6: Permitir que o aplicativo use o pré-processamento do Lambda

Etapa 1: Criar um usuário do IAM

Primeiro, é preciso criar um usuário do IAM, adicionar o usuário a um grupo do IAM com permissões administrativas e, em seguida, conceder permissões administrativas ao usuário criado. Em seguida, você pode acessar AWS usando um URL especial e as credenciais desse usuário.

Para obter instruções, consulte Criar o primeiro usuário do IAM e o grupo de administradores no Guia do usuário do IAM.

Etapa 2: Conceder ao usuário permissões para ações que não são específicas do

Vamos primeiro conceder a um usuário permissão para todas as ações que não são específicas do de que o usuário precisará ao trabalhar com aplicativos do . Isso inclui permissões para trabalhar com streams (ações do Amazon Kinesis Data Streams, ações do Amazon Data Firehose) e permissões para ações. CloudWatch Anexe a política a seguir ao usuário.

Você precisa atualizar a política fornecendo um nome de função do IAM para o qual deseja conceder a permissão `iam:PassRole` ou especificar um caractere curinga (*) indicando todas as funções do IAM. Essa não é uma prática segura; no entanto, você pode não ter uma função do IAM específica criada durante esse teste.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kinesis:CreateStream",
                "kinesis:DeleteStream",
                "kinesis:DescribeStream",
                "kinesis:ListStreams",
                "kinesis:PutRecord",
                "kinesis:PutRecords"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "firehose:DescribeDeliveryStream",
                "firehose:ListDeliveryStreams"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics"
            ],
            "Resource": "*"
        },
```

```
        {
            "Effect": "Allow",
            "Action": "logs:GetLogEvents",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListPolicyVersions",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/service-role/role-name"
        }
    ]
}
```

Etapa 3: Permitir que o usuário visualize uma lista de aplicativos e os detalhes

A política a seguir concede a um usuário as seguintes permissões:

- Permissão para a ação `kinesisanalytics:ListApplications` para que o usuário possa visualizar uma lista de aplicativos. Isso é uma chamada de API de nível de serviço e, portanto, especifique "*" como valor de `Resource`.
- Permissão para a ação `kinesisanalytics:DescribeApplication` para que você possa obter informações sobre qualquer um dos aplicativos.

Adicione essa política ao usuário.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:ListApplications"
            ],
            "Resource": "*"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:DescribeApplication"
            ],
            "Resource": "arn:aws:kinesisanalytics:aws-region:aws-account-
id:application/*"
        }
    ]
}
```

Verifique essas permissões fazendo login no console usando as credenciais de usuário.

Etapa 4: Permitir que o usuário inicie um aplicativo específico

Se quiser que o usuário inicie um dos aplicativos existentes do , anexe a política a seguir ao usuário. A política fornece a permissão para a ação `kinesisanalytics:StartApplication`. Você deve atualizar a política fornecendo o ID da conta, a AWS região e o nome do aplicativo.

```
{
    "Version": "2012-10-17",
    "Statement": [
               {
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:StartApplication"
            ],
            "Resource": "arn:aws:kinesisanalytics:aws-region:aws-account-
id:application/application-name"
        }
    ]
}
```

Etapa 5: Permitir que o usuário crie um aplicativo do

Se quiser que o usuário crie um aplicativo, anexe a política a seguir ao usuário. Você deve atualizar a política e fornecer uma AWS região, o ID da sua conta e um nome de aplicativo específico que você deseja que o usuário crie ou um “*” para que o usuário possa especificar qualquer nome de aplicativo (e, assim, criar vários aplicativos).

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1473028104000",
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:CreateApplication"
            ],
            "Resource": [
                "*"
            ]
        },

        {
            "Effect": "Allow",
            "Action": [
                "kinesisanalytics:StartApplication",
                "kinesisanalytics:UpdateApplication",
                "kinesisanalytics:AddApplicationInput",
                "kinesisanalytics:AddApplicationOutput"
            ],
            "Resource": "arn:aws:kinesisanalytics:aws-region:aws-account-
id:application/application-name"
        }
    ]
}
```

Etapa 6: Permitir que o aplicativo use o pré-processamento do Lambda

Se quiser que o aplicativo use o pré-processamento do Lambda, anexe a política a seguir ao perfil.

```
     {
       "Sid": "UseLambdaFunction",
       "Effect": "Allow",
       "Action": [
           "lambda:InvokeFunction",
           "lambda:GetFunctionConfiguration"
       ],
       "Resource": "<FunctionARN>"
   }
```

Permissões de API: referência a ações, permissões e recursos

Ao configurar Controle de acesso e escrever uma política de permissões que você pode anexar a uma identidade do IAM (políticas baseadas em identidade), é possível usar a lista de a seguir como referência. A inclui cada operação de API, as ações correspondentes para as quais você pode conceder permissões para realizar a ação e o AWS recurso para o qual você pode conceder as permissões. Você especifica as ações no campo `Action` da política e o valor do recurso no campo `Resource` da política.

Você pode usar chaves AWS de condição abrangentes em suas políticas para expressar condições. Para obter uma lista completa AWS de chaves gerais, consulte Chaves disponíveis no Guia do usuário do IAM.

> **Note**
>
> Para especificar uma ação, use o prefixo `kinesisanalytics` seguido do nome da operação da API (por exemplo, `kinesisanalytics:AddApplicationInput`).

API e permissões necessárias para ações

Operação de API:

Permissões obrigatórias (ação de API):

Recursos:

API e permissões necessárias para ações

API do Amazon RDS e permissões necessárias para ações

Operação de API:AddApplicationInput

Ação: `kinesisanalytics:AddApplicationInput`

Recursos:

`arn:aws:kinesisanalytics:` *`region`*`:`*`accountId`*`:application/`*`application-name`*

GetApplicationState

O console usa um método interno chamado `GetApplicationState` para criar uma amostra ou acessar dados do aplicativo. Seu aplicativo do serviço precisa ter permissões para que a API `kinesisanalytics:GetApplicationState` interna crie uma amostra ou acesse dados do aplicativo por meio do AWS Management Console.

Monitoramento

fornece a funcionalidade de monitoramento para os aplicativos. Para ter mais informações, consulte Monitoramento.

Validação de conformidade para o Amazon Kinesis Data Analytics para aplicativos SQL

Auditores terceirizados avaliam a segurança e a conformidade do Amazon Kinesis Data Analytics como parte de AWS vários programas de conformidade. Isso inclui SOC, PCI, HIPAA e outros.

Para obter uma lista de AWS serviços no escopo de programas de conformidade específicos, consulte Amazon Services in Scope by Compliance Program. Para obter informações gerais, consulte Programas de conformidade da AWS.

Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact. Para obter mais informações, consulte Baixar relatórios em AWS Artifact.

Sua responsabilidade de conformidade ao usar o Kinesis Data Analytics é determinada pela confidencialidade dos seus dados, pelos objetivos de conformidade da sua empresa e pelos regulamentos e leis aplicáveis. Se seu uso do Kinesis Data Analytics estiver sujeito à conformidade com normas como HIPAA ou PCI, a AWS fornecerá recursos para ajudar:

- Guias de início rápido sobre segurança e conformidade — Esses guias de implantação discutem considerações arquitetônicas e fornecem etapas para a implantação de ambientes básicos focados em segurança e conformidade em. AWS
- Documento técnico sobre arquitetura para segurança e conformidade com a HIPAA — Este whitepaper descreve como as empresas podem usar para criar aplicativos compatíveis com a HIPAA. AWS
- AWS Recursos de conformidade — Essa coleção de pastas de trabalho e guias pode ser aplicada ao seu setor e local.

- AWS Config— Esse AWS serviço avalia se suas configurações de recursos estão em conformidade com as práticas internas, as diretrizes e os regulamentos do setor.
- AWS Security Hub— Esse AWS serviço fornece uma visão abrangente do seu estado de segurança interno, AWS que ajuda você a verificar sua conformidade com os padrões e as melhores práticas do setor de segurança.

Resiliência no Amazon Kinesis Data Analytics

A infraestrutura AWS global é construída em torno de AWS regiões e zonas de disponibilidade. AWS As regiões fornecem várias zonas de disponibilidade fisicamente separadas e isoladas, conectadas a redes de baixa latência, alta taxa de transferência e alta redundância. Com as zonas de disponibilidade, é possível projetar e operar aplicações e bancos de dados que executam o failover automaticamente entre as zonas de disponibilidade sem interrupção. As Zonas de Disponibilidade são mais altamente disponíveis, tolerantes a falhas e escaláveis que uma ou várias infraestruturas de datacenter tradicionais.

Para obter mais informações sobre AWS regiões e zonas de disponibilidade, consulte Infraestrutura AWS global.

Além da infraestrutura AWS global, o Kinesis Data Analytics oferece vários recursos para ajudar a suportar suas necessidades de resiliência e backup de dados.

Recuperação de desastres

O Kinesis Data Analytics é executado em um modo sem servidor e cuida das degradações do host, da disponibilidade da zona de disponibilidade e outros problemas relacionados à infraestrutura, fazendo uma migração automática. Quando isso acontece, o Kinesis Data Analytics garante que o aplicativo seja processado sem nenhuma perda de dados. Para ter mais informações, consulte Modelo de entrega para manter a saída do aplicativo em um destino externo.

Segurança da infraestrutura no Kinesis Data Analytics para aplicativos SQL

Como um serviço gerenciado, o Amazon Kinesis Data Analytics é protegido AWS pelos procedimentos globais de segurança de rede descritos no whitepaper Amazon Web Services: Visão geral dos processos de segurança.

Você usa chamadas de API AWS publicadas para acessar o Kinesis Data Analytics pela rede. Os clientes devem oferecer suporte a Transport Layer Security (TLS) 1.2 ou posterior. Os clientes também devem ter suporte a conjuntos de criptografia com perfect forward secrecy (PFS) como Ephemeral Diffie-Hellman (DHE) ou Ephemeral Elliptic Curve Diffie-Hellman (ECDHE). A maioria dos sistemas modernos como Java 7 e versões posteriores é compatível com esses modos.

Além disso, as solicitações devem ser assinadas utilizando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou é possível usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

Práticas recomendadas de segurança para o Kinesis Data Analytics

O Amazon Kinesis Data Analytics fornece uma série de recursos de segurança a serem considerados no desenvolvimento e na implementação das suas próprias políticas de segurança. As melhores práticas a seguir são diretrizes gerais e não representam uma solução completa de segurança. Como essas melhores práticas podem não ser adequadas ou suficientes para o seu ambiente, trate-as como considerações úteis em vez de prescrições.

Use perfis do IAM para acessar outros serviços da Amazon

Seu aplicativo Kinesis Data Analytics deve ter credenciais válidas para acessar recursos em outros serviços, como fluxos de dados Kinesis, streams de entrega Firehose ou buckets Amazon S3. Você não deve armazenar AWS credenciais diretamente no aplicativo ou em um bucket do Amazon S3. Essas são credenciais de longo prazo que não são automaticamente alternadas e podem ter um impacto comercial significativo se forem comprometidas.

Em vez disso, você deve usar um perfil do IAM para gerenciar credenciais temporárias para que o aplicativo acesse outros recursos. Quando você usa um perfil, não precisa usar credenciais de longo prazo para acessar outros recursos.

Para obter mais informações, consulte os seguintes tópicos no Manual do usuário do IAM:

- Funções do IAM
- Cenários comuns para funções: usuários, aplicativos e serviços

Implemente a criptografia do lado do servidor em recursos dependentes

Dados em repouso e dados em trânsito são criptografados no Kinesis Data Analytics, e essa criptografia não pode ser desabilitada. Você deve implementar a criptografia do lado do servidor em seus recursos dependentes, como streams de dados Kinesis, streams de entrega Firehose e buckets Amazon S3. Para obter mais informações sobre como implementar a criptografia do lado do servidor em recursos dependentes, consulte Proteção de dados.

Use CloudTrail para monitorar chamadas de API

O Kinesis Data Analytics é integrado AWS CloudTrail com, um serviço que fornece um registro das ações realizadas por um usuário, função ou serviço da Amazon no Kinesis Data Analytics.

Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita à Kinesis Data Analytics, o endereço IP a partir do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para ter mais informações, consulte the section called “Usar o AWS CloudTrail”.

Monitoramento do para aplicativos SQL

Monitoramento é uma parte importante da manutenção da confiabilidade, da disponibilidade e da performance do e de seu aplicativo . Você deve coletar dados de monitoramento de todas as partes da sua AWS solução para poder depurar com mais facilidade uma falha multiponto, caso ocorra. Porém, para começar a monitorar o , é necessário criar um plano de monitoramento que inclua respostas às seguintes perguntas:

- Quais são seus objetivos de monitoramento?
- Quais recursos você vai monitorar?
- Com que frequência você vai monitorar esses recursos?
- Quais ferramentas de monitoramento você usará?
- Quem realizará o monitoramento das tarefas?
- Quem deve ser notificado quando algo der errado?

A próxima etapa é estabelecer uma linha de base de desempenho normal de em seu ambiente, medindo o desempenho em vários momentos e em diferentes condições de carga. Enquanto monitora o , você pode armazenar dados históricos de monitoramento. Se fizer isso, você poderá compará-los com os dados de desempenho atuais, identificar padrões de desempenho normais e anomalias de desempenho, e elaborar métodos para resolver problemas.

Com o , você monitora o aplicativo. O aplicativo processa fluxos de dados (entrada ou saída), ambos incluindo identificadores que você pode usar para restringir sua pesquisa em CloudWatch registros. Para obter informações sobre como processar fluxos de dados, consulte Amazon Kinesis Data Analytics para aplicativos SQL: como funciona.

A métrica mais importante é `millisBehindLatest`, que indica o tempo de atraso de um aplicativo na leitura da origem de streaming. Em um caso típico, o milissegundos de atraso deve ser zero ou estar próximo a zero. É comum aparecerem breves picos, que representam um aumento em `millisBehindLatest`.

Recomendamos que você configure um CloudWatch alarme que seja acionado quando o aplicativo estiver atrasado por mais de uma hora na leitura da fonte de streaming. Em alguns casos de uso que exigem um processamento quase em tempo real, como a emissão de dados processados para um aplicativo ao vivo, você pode optar por definir o alarme em um valor mais baixo, como cinco minutos, por exemplo.

Tópicos

Ferramentas de monitoramento

AWS fornece várias ferramentas que você pode usar para monitorar. É possível configurar algumas dessas ferramentas para fazer o monitoramento em seu lugar, e, ao mesmo tempo, algumas das ferramentas exigem intervenção manual. Recomendamos que as tarefas de monitoramento sejam automatizadas ao máximo possível.

Ferramentas de monitoramento automatizadas

Você pode usar as seguintes ferramentas de monitoramento automatizadas para observar o e gerar relatórios quando algo estiver errado:

- Amazon CloudWatch Alarms — Observe uma única métrica durante um período de tempo especificado por você e execute uma ou mais ações com base no valor da métrica em relação a um determinado limite em vários períodos. A ação é uma notificação enviada para um tópico do Amazon Simple Notification Service (Amazon SNS) ou para uma política do Amazon EC2 Auto Scaling. CloudWatch os alarmes não invocam ações simplesmente porque estão em um determinado estado; o estado deve ter sido alterado e mantido por um determinado número de períodos. Para ter mais informações, consulte Monitoramento com a Amazon CloudWatch.
- Amazon CloudWatch Logs — Monitore, armazene e acesse seus arquivos de log de AWS CloudTrail ou de outras fontes. Para obter mais informações, consulte Monitoramento de arquivos de log no Guia CloudWatch do usuário da Amazon.
- Amazon CloudWatch Events — Combine eventos e encaminhe-os para uma ou mais funções ou streams de destino para fazer alterações, capturar informações de estado e tomar medidas corretivas. Para obter mais informações, consulte O que é Amazon CloudWatch Events no Guia CloudWatch do usuário da Amazon.
- AWS CloudTrail Monitoramento de registros — compartilhe arquivos de log entre contas, monitore arquivos de CloudTrail log em tempo real enviando-os para o CloudWatch Logs, grave aplicativos de processamento de log em Java e valide se seus arquivos de log não foram alterados após a entrega. CloudTrail Para obter mais informações, consulte Trabalhando com arquivos de CloudTrail log no GuiaAWS CloudTrail do usuário.

Ferramentas de monitoramento manual

Outra parte importante do monitoramento envolve o monitoramento manual dos itens que os CloudWatch alarmes não cobrem. Os AWS Management Console painéis CloudWatch, Trusted Advisor,, e outros fornecem uma at-a-glance visão do estado do seu AWS ambiente.

- A página CloudWatch inicial mostra o seguinte:
  - Alertas e status atual
  - Gráficos de alertas e recursos
  - Estado de integridade do serviço

  Além disso, você pode usar CloudWatch para fazer o seguinte:
  - Crie painéis personalizados para monitorar os serviços com os quais você se preocupa.
  - Colocar em gráfico dados de métrica para solucionar problemas e descobrir tendências
  - Pesquisar e procurar todas as métricas
  - Criar e editar alertas para ser notificado sobre problemas
- AWS Trusted Advisor pode ajudá-lo a monitorar seu para melhorar o desempenho, a confiabilidade, a segurança e a economia. Quatro verificações Trusted Advisor estão disponíveis para todos os usuários. Mais de 50 verificações estão disponíveis para os usuários com um plano de suporte de Negócios ou Empresarial. Para ter mais informações, consulte AWS Trusted Advisor.

Monitoramento com a Amazon CloudWatch

Você pode monitorar aplicativos usando a Amazon CloudWatch. CloudWatch coleta e processa dados brutos em métricas legíveis e quase em tempo real. Essas estatísticas são mantidas durante um período de duas semanas. Você pode acessar as informações históricas e ter uma perspectiva melhor sobre o desempenho do aplicativo web ou o do serviço. Por padrão, os dados métricos são enviados automaticamente para CloudWatch. Para obter mais informações, consulte O que é a Amazon CloudWatch? no Guia do CloudWatch usuário da Amazon.

Tópicos

- Métricas e dimensões
- Visualizar métricas e dimensões do
- Criando CloudWatch alarmes para monitorar

- Trabalhando com a Amazon CloudWatch Logs

Métricas e dimensões

O namespace `AWS/KinesisAnalytics` inclui as métricas a seguir.

Métrica	Descrição
`Bytes`	O número de bytes lidos (por fluxo de entrada) ou gravados (por fluxo de saída). Níveis: por fluxo de entrada e por fluxo de saída
`KPUs`	O número de unidades de processamento do Kinesis que são usadas para executar o aplicativo de processamento de fluxos. O número médio de KPUs usadas a cada hora determina o faturamento do seu aplicativo. Níveis: aplicativo
`MillisBehindLatest`	Indica com que atraso um aplicativo está lendo a origem do streaming. Níveis: aplicativo
`Records`	O número de registros lidos (por fluxo de entrada) ou gravados (por fluxo de saída). Níveis: por fluxo de entrada e por fluxo de saída
`Success`	1 para cada tentativa de entrega bem-sucedida ao destino configurado para seu aplicativo; 0 para cada tentativa de entrega falha. O valor médio dessa métrica indica quantas entregas bem-sucedidas foram realizadas. Níveis: por destino.

Métrica	Descrição
`InputProcessing.Duration`	O tempo gasto para cada invocação de AWS Lambda função realizada por. Níveis: por fluxo de entrada
`InputProcessing.OkRecords`	O número de registros retornados por uma função do Lambda que foram marcados com o status `Ok`. Níveis: por fluxo de entrada
`InputProcessing.OkBytes`	A soma de bytes dos registros retornados por uma função do Lambda que foram marcados com o status `Ok`. Níveis: por fluxo de entrada
`InputProcessing.DroppedRecords`	O número de registros retornados por uma função do Lambda que foram marcados com o status `Dropped`. Níveis: por fluxo de entrada
`InputProcessing.ProcessingFailedRecords`	O número de registros retornados por uma função do Lambda que foram marcados com o status `ProcessingFailed` . Níveis: por fluxo de entrada
`InputProcessing.Success`	O número de invocações do Lambda pelo bem-sucedidas. Níveis: por fluxo de entrada
`LambdaDelivery.OkRecords`	O número de registros retornados por uma função do Lambda que foram marcados com o status `Ok`. Níveis: por destino Lambda

Métrica	Descrição
`LambdaDelivery.DeliveryFailedRecords`	O número de registros retornados por uma função do Lambda que foram marcados com o status `DeliveryFailed` . Níveis: por destino Lambda
`LambdaDelivery.Duration`	O tempo necessário para cada invocação de função do Lambda executada pelo . Níveis: por destino Lambda

fornece métricas para as dimensões a seguir.

Dimensão	Descrição
`Flow`	Por fluxo de entrada: entrada Por fluxo de saída: saída
`Id`	Por fluxo de entrada: ID de entrada Por fluxo de saída: ID de saída

Visualizar métricas e dimensões do

Quando seu aplicativo processa fluxos de dados, envia as seguintes métricas e dimensões para o. CloudWatch É possível usar os procedimentos a seguir para visualizar as métricas do .

No console, as métricas são agrupadas primeiro pelo namespace do serviço e depois por várias combinações de dimensão em cada namespace.

Para visualizar métricas usando o CloudWatch console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecione Métricas.
3. No painel CloudWatch Métricas por categoria, escolha uma categoria de métricas.

4. No painel superior, role para visualizar a lista completa de métricas.

Para visualizar métricas usando o AWS CLI

- Em um prompt de comando, use o seguinte comando.

```
aws cloudwatch list-metrics --namespace "AWS/KinesisAnalytics" --region region
```

métricas são coletadas nos seguintes níveis:

- Aplicativo
- Fluxo de entrada
- Fluxo de saída

Criando CloudWatch alarmes para monitorar

Você pode criar um CloudWatch alarme da Amazon que envia uma mensagem do Amazon SNS quando o alarme muda de estado. O alarme observa uma única métrica em um período especificado. Ele executa uma ou mais ações com base no valor da métrica em relação a um limite especificado ao longo de vários períodos. A ação é uma notificação enviada para um tópico do Amazon SNS ou uma política de Auto Scaling.

Os alertas invocam ações apenas para alterações de estado mantidas. Para que um CloudWatch alarme invoque uma ação, o estado deve ter sido alterado e mantido por um período de tempo especificado.

Você pode definir alarmes usando a CloudWatch API AWS Management Console, CloudWatch AWS CLI, ou, conforme descrito a seguir.

Para definir um alarme usando o CloudWatch console

1. Faça login no AWS Management Console e abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Escolha Create Alarm. O Create Alarm Wizard (Assistente de criação de alarmes) é iniciado.
3. Escolha Kinesis Analytics Metrics (Métricas do Kinesis Analytics). Role pelas métricas do para localizar a métrica em que deseja colocar um alarme.

Para exibir apenas as métricas do , procure o ID do sistema de arquivos. Escolha a métrica para a qual será criada um alarme e escolha Next (Próximo).

4. Insira valores em Name (Nome), Description (Descrição) e Whenever (Sempre que) para a métrica.
5. Se você quiser CloudWatch enviar um e-mail quando o estado do alarme for atingido, no campo Sempre que este alarme:, escolha Estado é ALARME. No campo Send notification to:, selecione um tópico do SNS existente. Se você selecionar Create topic, poderá definir o nome e o endereço de e-mail para uma nova lista de assinatura de e-mail. Essa lista é salva e aparece no campo para alarmes futuros.

   > **Note**
   >
   > Se você usar Create topic (Criar tópico) para criar um novo tópico do Amazon SNS, os endereços de e-mail deverão ser verificados antes que eles recebam notificações. Os e-mails são enviados apenas quando o alarme entra em um status de alarme. Se essa alteração no status de alarme ocorrer antes dos endereços de e-mail serem verificados, eles não receberão notificação.

6. Na seção Alarm Preview, visualize o alarme que você está prestes a criar.
7. Escolha Create Alarm para criar o alarme.

Para definir um alarme usando a CloudWatch CLI

- Chame `mon-put-metric-alarm`. Para obter mais informações, consulte a Amazon CloudWatch CLI Reference.

Para definir um alarme usando a CloudWatch API

- Chame `PutMetricAlarm`. Para obter mais informações, consulte a Amazon CloudWatch API Reference.

Trabalhando com a Amazon CloudWatch Logs

Se um aplicativo do estiver configurado incorretamente, ele poderá fazer a transição para um estado de execução durante a inicialização do aplicativo. Ou poderá atualizar, mas não processará nenhum

dado no fluxo de entrada no aplicativo. Ao adicionar uma opção de CloudWatch log ao aplicativo, você pode monitorar os problemas de configuração do aplicativo.

pode gerar erros de configuração nas seguintes condições:

- O fluxo de dados do Kinesis usado para entrada não existe.
- O stream de entrega do Amazon Data Firehose usado para entrada não existe.
- O bucket do Amazon S3 usado como fonte de dados de referência não existe.
- O arquivo especificado na fonte de dados de referência no bucket do S3 não existe.
- O recurso correto não está definido na função AWS Identity and Access Management (IAM) que gerencia as permissões relacionadas.
- A permissão correta não é definido na função do IAM que gerencia permissões relacionadas.
- não tem permissão para assumir a função do IAM que gerencia permissões relacionadas.

Para obter mais informações sobre a Amazon CloudWatch, consulte o Guia CloudWatch do usuário da Amazon.

Adicionando a ação PutLogEvents política

precisa de permissões para gravar erros de configuração incorreta. CloudWatch Você pode adicionar essas permissões à função do IAM assumida pelo , como descrito a seguir. Para obter mais informações sobre o uso de um perfil do IAM para , consulte Gerenciamento de identidade e acesso no Kinesis Data Analytics.

Política de confiança

Para conceder permissões ao para assumir uma função do IAM, anexe a política de confiança a seguir à função.

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

```
  ]
}
```

Política de permissões

Para conceder a um aplicativo permissões para gravar eventos de log a CloudWatch partir de um recurso, você pode usar a seguinte política de permissões do IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt0123456789000",
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:123456789012:log-group:my-log-group:log-
stream:my-log-stream*"
            ]
        }
    ]
}
```

Adição de monitoramento de erros de configuração

Use as ações de API a seguir para adicionar uma opção de CloudWatch log a um aplicativo novo ou existente ou alterar uma opção de log de um aplicativo existente.

> **Note**
>
> Atualmente, você só pode adicionar uma opção de CloudWatch log a um aplicativo usando ações de API. Você não pode adicionar opções de CloudWatch registro usando o console.

Adicionando uma opção de CloudWatch registro ao criar um aplicativo

O exemplo de código a seguir demonstra como usar a `CreateApplication` ação para adicionar uma opção de CloudWatch log ao criar um aplicativo. Para obter mais informações sobre `Create_Application`, consulte CreateApplication.

```
{
   "ApplicationCode": "<The SQL code the new application will run on the input
 stream>",
   "ApplicationDescription": "<A friendly description for the new application>",
   "ApplicationName": "<The name for the new application>",
   "Inputs": [ ... ],
   "Outputs": [ ... ],
   "CloudWatchLoggingOptions": [{
      "LogStreamARN": "<Amazon Resource Name (ARN) of the CloudWatch log stream to add
 to the new application>",
      "RoleARN": "<ARN of the role to use to access the log>"
 }]
}
```

Adicionando uma opção de CloudWatch log a um aplicativo existente

O exemplo de código a seguir demonstra como usar a ação `AddApplicationCloudWatchLoggingOption` para adicionar uma opção de log do CloudWatch a um aplicativo existente. Para obter mais informações sobre o `AddApplicationCloudWatchLoggingOption`, consulte AddApplicationCloudWatchLoggingOption.

```
{
   "ApplicationName": "<Name of the application to add the log option to>",
   "CloudWatchLoggingOption": {
      "LogStreamARN": "<ARN of the log stream to add to the application>",
      "RoleARN": "<ARN of the role to use to access the log>"
   },
   "CurrentApplicationVersionId": <Version of the application to add the log to>
}
```

Atualizando uma opção de CloudWatch registro existente

O exemplo de código a seguir demonstra como usar a `UpdateApplication` ação para modificar uma opção de CloudWatch log existente. Para obter mais informações sobre o `UpdateApplication`, consulte UpdateApplication.

```
{
   "ApplicationName": "<Name of the application to update the log option for>",
   "ApplicationUpdate": {
```

```
      "CloudWatchLoggingOptionUpdates": [
         {
            "CloudWatchLoggingOptionId": "<ID of the logging option to modify>",
            "LogStreamARNUpdate": "<ARN of the new log stream to use>",
            "RoleARNUpdate": "<ARN of the new role to use to access the log stream>"
         }
      ],
   },
   "CurrentApplicationVersionId": <ID of the application version to modify>
}
```

Excluindo uma opção de CloudWatch log de um aplicativo

O exemplo de código a seguir demonstra como usar a `DeleteApplicationCloudWatchLoggingOption` ação para excluir uma opção de CloudWatch log existente. Para obter mais informações sobre o `DeleteApplicationCloudWatchLoggingOption`, consulte DeleteApplicationCloudWatchLoggingOption.

```
{
   "ApplicationName": "<Name of application to delete log option from>",
   "CloudWatchLoggingOptionId": "<ID of the application log option to delete>",
   "CurrentApplicationVersionId": <Version of the application to delete the log option
 from>
}
```

Erros de configuração

As seções a seguir contêm detalhes sobre erros que você pode ver no Amazon CloudWatch Logs causados por um aplicativo mal configurado.

Formato da mensagem de erro

As mensagens de erro geradas pelo erro de configuração do aplicativo aparecem no formato a seguir.

```
{
```

```
"applicationARN": "string",
"applicationVersionId": integer,
"messageType": "ERROR",
"message": "string",
"inputId": "string",
"referenceId": "string",
"errorCode": "string"
"messageSchemaVersion": "integer",
}
```

Os campos em uma mensagem de erro contêm as seguintes informações:

- `applicationARN`: o nome de recurso da Amazon (ARN) do aplicativo que gerou o erro, por exemplo: `arn:aws:kinesisanalytics:us-east-1:112233445566:application/sampleApp`
- `applicationVersionId`: a versão do aplicativo no momento em que o erro foi detectado. Para ter mais informações, consulte ApplicationDetail.
- `messageType`: o tipo de mensagem. No momento, esse tipo pode ser apenas `ERROR`.
- `message`: os detalhes do erro, por exemplo:

  ```
  There is a problem related to the configuration of your input. Please check that the
   resource exists, the role has the correct permissions to access the resource and
   that Kinesis Analytics can assume the role provided.
  ```

- `inputId`: o ID associado à entrada do aplicativo. Esse valor só estará presente se essa entrada for a causa do erro. Esse valor não estará presente se `referenceId` estiver presente. Para ter mais informações, consulte DescribeApplication.
- `referenceId`: o ID associado à fonte de dados de referência do aplicativo. Esse valor só estará presente se essa fonte de dados for a causa do erro. Esse valor não estará presente se `inputId` estiver presente. Para ter mais informações, consulte DescribeApplication.
- `errorCode`: o identificador do erro. Esse ID é `InputError` ou `ReferenceDataError`.
- `messageSchemaVersion`: um valor que especifica a versão de esquema da mensagem atual, no momento 1. Você pode verificar esse valor para ver se o esquema da mensagem de erro foi atualizado.

Erros

Os erros que podem aparecer no CloudWatch Logs for incluem o seguinte.

O recurso não existe

Se um ARN for especificado para um fluxo de entrada do Kinesis que não existe, mas o ARN estiver sintaticamente correto, um erro semelhante ao seguinte será gerado.

```
{
"applicationARN": "arn:aws:kinesisanalytics:us-east-1:112233445566:application/
sampleApp",
"applicationVersionId": "5",
 "messageType": "ERROR",
 "message": "There is a problem related to the configuration of your input. Please
 check that the resource exists, the role has the correct permissions to access the
 resource and that Kinesis Analytics can assume the role provided.",
 "inputId":"1.1",
 "errorCode": "InputError",
 "messageSchemaVersion": "1"
}
```

Se uma chave de arquivo incorreta do Amazon S3 for usada como dados de referência, um erro semelhante ao seguinte será gerado.

```
{
 "applicationARN": "arn:aws:kinesisanalytics:us-east-1:112233445566:application/
sampleApp",
 "applicationVersionId": "5",
 "messageType": "ERROR",
 "message": "There is a problem related to the configuration of your reference data.
 Please check that the bucket and the file exist, the role has the correct permissions
 to access these resources and that Kinesis Analytics can assume the role provided.",
 "referenceId":"1.1",
 "errorCode": "ReferenceDataError",
 "messageSchemaVersion": "1"
}
```

A função não existe

Se um nome de região da Amazon (ARN) for especificado para uma função de entrada do IAM que não existe, mas o ARN estiver sintaticamente correto, um erro semelhante ao seguinte será gerado.

```
{
 "applicationARN": "arn:aws:kinesisanalytics:us-east-1:112233445566:application/
sampleApp",
```

```
 "applicationVersionId": "5",
 "messageType": "ERROR",
 "message": "There is a problem related to the configuration of your input. Please
 check that the resource exists, the role has the correct permissions to access the
 resource and that Kinesis Analytics can assume the role provided.",
 "inputId":null,
 "errorCode": "InputError",
 "messageSchemaVersion": "1"
}
```

A função não tem permissões para acessar o recurso

Se uma função de entrada for usada e não tiver permissão para acessar os recursos de entrada, como um fluxo de origem do Kinesis, um erro semelhante ao seguinte será gerado.

```
{
 "applicationARN": "arn:aws:kinesisanalytics:us-east-1:112233445566:application/
sampleApp",
 "applicationVersionId": "5",
 "messageType": "ERROR",
 "message": "There is a problem related to the configuration of your input. Please
 check that the resource exists, the role has the correct permissions to access the
 resource and that Kinesis Analytics can assume the role provided.",
 "inputId":null,
 "errorCode": "InputError",
 "messageSchemaVersion": "1"
}
```

Registro em log de chamadas de API com o AWS CloudTrail

é integrado ao AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, um perfil ou um serviço AWS em . O CloudTrail captura as chamadas de API do como eventos. As chamadas capturadas incluem as chamadas do console do e as chamadas de código para as operações da API do . Se você criar uma trilha, poderá habilitar a entrega contínua de eventos do CloudTrail para um bucket do Amazon S3, incluindo eventos para o . Se você não configurar uma trilha, ainda poderá visualizar os eventos mais recentes no console do CloudTrail em Event history (Histórico de eventos). Usando as informações coletadas pelo CloudTrail, é possível determinar a solicitação feita para o , o endereço IP no qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita, além de detalhes adicionais.

Para saber mais sobre o CloudTrail, consulte o Guia do usuário do AWS CloudTrail.

Informações no CloudTrail

O CloudTrail é habilitado em sua conta da AWS quando ela é criada. Quando ocorre uma atividade no , ela é registrada em um evento do CloudTrail junto com outros eventos de serviços da AWS em Histórico de eventos. Você pode visualizar, pesquisar e baixar os eventos recentes em sua conta da AWS. Para obter mais informações, consulte Como visualizar eventos com o histórico de eventos do CloudTrail.

Para obter um registro contínuo de eventos na conta da AWS, incluindo eventos do , crie uma trilha. Uma trilha permite que o CloudTrail entregue arquivos de log a um bucket do Amazon S3. Por padrão, quando você cria uma trilha no console, ela é aplicada a todas as . A trilha registra em log eventos de todas as regiões na partição da AWS e entrega os arquivos de log para o bucket do Amazon S3 especificado por você. Além disso, é possível configurar outros serviços da AWS para analisar mais ainda mais e agir com base nos dados de eventos coletados nos logs do CloudTrail. Para obter mais informações, consulte:

- Visão geral da criação de uma trilha
- Serviços e integrações compatíveis com o CloudTrail
- Configurar notificações do Amazon SNS para o CloudTrail
- Receber arquivos de log do CloudTrail de várias regiões e receber arquivos de log do CloudTrail de várias contas

Todas as ações são registradas pelo CloudTrail e documentadas na Referência de API. Por exemplo, as chamadas para as ações `CreateApplication` e `UpdateApplication` geram entradas nos arquivos de log do CloudTrail.

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Seja a solicitação feita com Usuário raiz da conta da AWS ou credenciais de usuário.
- Se a solicitação foi feita com credenciais de segurança temporárias de uma função ou de um usuário federado.
- Se a solicitação foi feita por outro serviço da AWS.

Para obter mais informações, consulte o Elemento userIdentity do CloudTrail.

Noções básicas das entradas dos arquivos de log do

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log a um bucket do Amazon S3 especificado. Os arquivos de log do CloudTrail contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer fonte e inclui informações sobre a ação solicitada, a data e a hora da ação, os parâmetros de solicitação e assim por diante. Os arquivos de log do CloudTrail não são um rastreamento de pilha ordenada de chamadas de API pública. Dessa forma, eles não são exibidos em uma ordem específica.

O exemplo a seguir mostra uma entrada de log do CloudTrail que demonstra as ações AddApplicationCloudWatchLoggingOption e DescribeApplication.

```
{
    "Records": [
        {
            "eventVersion": "1.05",
            "userIdentity": {
                "type": "IAMUser",
                "principalId": "EX_PRINCIPAL_ID",
                "arn": "arn:aws:iam::012345678910:user/Alice",
                "accountId": "012345678910",
                "accessKeyId": "EXAMPLE_KEY_ID",
                "userName": "Alice"
            },
            "eventTime": "2019-03-14T01:03:00Z",
            "eventSource": "kinesisanalytics.amazonaws.com",
            "eventName": "AddApplicationCloudWatchLoggingOption",
            "awsRegion": "us-east-1",
            "sourceIPAddress": "127.0.0.1",
            "userAgent": "aws-sdk-java/unknown-version Linux/x.xx",
            "requestParameters": {
                "currentApplicationVersionId": 1,
                "cloudWatchLoggingOption": {
                    "roleARN": "arn:aws:iam::012345678910:role/cloudtrail_test",
                    "logStreamARN": "arn:aws:logs:us-east-1:012345678910:log-
group:cloudtrail-test:log-stream:sql-cloudwatch"
                },
                "applicationName": "cloudtrail-test"
            },
            "responseElements": null,
            "requestID": "e897cd34-45f4-11e9-8912-e52573a36cd9",
            "eventID": "57fe50e9-c764-47c3-a0aa-d0c271fa1cbb",
```

```
            "eventType": "AwsApiCall",
            "apiVersion": "2015-08-14",
            "recipientAccountId": "303967445486"
        },
        {
            "eventVersion": "1.05",
            "userIdentity": {
                "type": "IAMUser",
                "principalId": "EX_PRINCIPAL_ID",
                "arn": "arn:aws:iam::012345678910:user/Alice",
                "accountId": "012345678910",
                "accessKeyId": "EXAMPLE_KEY_ID",
                "userName": "Alice"
            },
            "eventTime": "2019-03-14T05:37:20Z",
            "eventSource": "kinesisanalytics.amazonaws.com",
            "eventName": "DescribeApplication",
            "awsRegion": "us-east-1",
            "sourceIPAddress": "127.0.0.1",
            "userAgent": "aws-sdk-java/unknown-version Linux/x.xx",
            "requestParameters": {
                "applicationName": "cloudtrail-test"
            },
            "responseElements": null,
            "requestID": "3b74eb29-461b-11e9-a645-fb677e53d147",
            "eventID": "750d0def-17b6-4c20-ba45-06d9d45e87ee",
            "eventType": "AwsApiCall",
            "apiVersion": "2015-08-14",
            "recipientAccountId": "012345678910"
        }
    ]
}
```

Limites

Ao trabalhar com o Amazon Kinesis Data Analytics para aplicativos SQL, observe os seguintes limites:

- O Kinesis Data Analytics para SQL está disponível nas seguintes Regiões AWS: Leste dos EUA (Ohio), Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Canadá (Central), Europa (Paris), Europa (Irlanda), Europa (Frankfurt), Europa (Londres), Europa (Paris), Ásia-Pacífico (Hong Kong), Ásia-Pacífico (Seul), Ásia-Pacífico (Sydney), Ásia-Pacífico (Singapura), Ásia-Pacífico (Tóquio), América do Sul (São Paulo), AWS GovCloud (Leste dos EUA) e AWS GovCloud (Oeste dos EUA). Não temos planos de lançar o Kinesis Data Analytics para SQL em regiões AWS adicionais.
- Depois de 28 de junho de 2023, você não poderá criar novos aplicativos Kinesis Data Analytics para SQL usando o console de gestão AWS se ainda não estiver usando o Kinesis Data Analytics para SQL. Se você criou um aplicativo Kinesis Data Analytics para SQL antes de 28 de junho de 2023, não haverá alterações na forma como você cria e executa aplicativos hoje em uma região AWS em que você já usa o Kinesis Data Analytics para SQL. No entanto, você não poderá mais criar novos aplicativos usando o console AWS em uma região onde não usa o Kinesis Data Analytics para SQL.
- Depois de 12 de setembro de 2023, você não poderá criar novos aplicativos usando o Kinesis Data Firehose como fonte se ainda não estiver usando o Kinesis Data Analytics for SQL. Os clientes atuais que usam o Kinesis Data Analytics para aplicativos SQL com `KinesisFirehoseInput` podem continuar adicionando aplicativos com `KinesisFirehoseInput` em uma conta existente usando o Kinesis Data Analytics. Se você já é cliente e deseja criar uma nova conta com os aplicativos Kinesis Data Analytics para SQL com `KinesisFirehoseInput`, você pode criar um caso de suporte. Para obter mais informações, consulte a Central AWS Support.
- O tamanho de uma linha em um stream no aplicativo é limitado a 512 KB. O Kinesis Data Analytics usa até 1 KB para armazenar metadados. Esses metadados são contados em relação ao limite de linhas.
- O código SQL em um aplicativo limita-se a 100 KB.

- A janela mais longa que recomendamos para uma consulta em janela é de uma hora. Os fluxos no aplicativo são armazenados em armazenamento volátil, e interrupções inesperadas do aplicativo farão com que ele recrie o fluxo a partir dos dados de origem no armazenamento volátil.

- A maior taxa de transferência que recomendamos para um único fluxo no aplicativo é entre 2 e 20 MB/segundo, dependendo da complexidade da consulta do aplicativo.

- Você pode criar até 50 aplicativos do Kinesis Data Analytics por região AWS na sua conta. Você pode criar um caso para solicitar aplicativos adicionais por meio do formulário de aumento de limite de serviço. Para obter mais informações, consulte a Central AWS Support.

- O throughput máximo de streaming que um único aplicativo Kinesis Data Analytics para SQL pode processar é de aproximadamente 100 MB/seg. Isso pressupõe que você tenha aumentado o número de fluxos no aplicativo para o valor máximo de 64 e que tenha aumentado o limite de KPU para acima de 8 (veja o limite a seguir para obter detalhes). Se o aplicativo precisar processar mais de 100 MB/s de entrada, realize uma das seguintes ações:
  - Use vários aplicativos Kinesis Data Analytics para SQL para processar a entrada
  - Use Aplicativos Managed Service for Apache Flink para Java se quiser continuar a usar um único fluxo e aplicativo.

  > **Note**
  >
  > Recomendamos revisar periodicamente a métrica `InputProcessing.0kBytes` do seu aplicativo para que você possa planejar com antecedência o uso de vários aplicativos SQL ou migrar para Aplicativos Managed Service for for Apache Flink para Java se o throughput de entrada projetado do seu aplicativo exceder 100 MB/seg. Também recomendamos a criação de um alarme do CloudWatch em `InputProcessing.0kBytes` para que você seja notificado quando seu aplicativo estiver próximo do limite do throughput de entrada. Isso pode ser útil, pois você pode atualizar a consulta do aplicativo para compensar por um maior throughput, evitando assim a contrapressão e o atraso na análise. Para obter mais informações, consulte Solução de problemas. O alarme também pode ser útil se você tiver um mecanismo para reduzir o throughput upstream.

- A quantidade máxima de unidades de processamento do Kinesis (KPU) é oito. Para obter instruções sobre como solicitar um aumento desse limite, consulte Para solicitar um aumento de limite em Limites de serviço da Amazon.

  Com o Kinesis Data Analytics você paga apenas pelo que usa. É cobrada uma taxa por hora com base no número médio de KPUs usadas para executar o aplicativo de processamento de streams. Uma única KPU oferece 1 vCPU e 4 GB de memória.

- Cada aplicativo pode ter uma origem de streaming e até uma fonte de dados de referência.

- Você pode configurar até três destinos para o seu aplicativo Kinesis Data Analytics. É recomendável que você use um dos seguintes destinos para manter os dados do fluxo de erros de aplicativo.

- O objeto do Amazon S3 que armazena dados de referência pode chegar a um tamanho de até 1 GB.

- Se você alterar os dados de referência armazenados no bucket do S3 após fazer upload desses dados para uma tabela do aplicativo, será necessário usar a operação UpdateApplication (usando a API ou a AWS CLI) para atualizar os dados nessa tabela. No momento, o AWS Management Console não oferece suporte à atualização dos dados de referência no aplicativo.

- No momento, o Kinesis Data Analytics não é compatível com os dados gerados pelo Amazon Kinesis Producer Library (KPL).

- Você pode atribuir até 50 tags por aplicativo.

Práticas recomendadas

Esta seção descreve as práticas recomendadas ao trabalhar com aplicativos do Amazon Kinesis Data Analytics.

Tópicos

Como gerenciar aplicativos

Ao gerenciar aplicativos do Amazon Kinesis Data Analytics, siga estas práticas recomendadas:

- Configurar CloudWatch alarmes da Amazon — Você pode usar as CloudWatch métricas que o Kinesis Data Analytics fornece para monitorar o seguinte:
  - Bytes de entrada e registros de entrada (número de bytes e registros que entram no aplicativo)
  - Bytes de saída e registros de saída
  - `MillisBehindLatest` (tempo de atraso do aplicativo na leitura da origem de streaming)

  Recomendamos que você configure pelo menos dois CloudWatch alarmes nas seguintes métricas para seus aplicativos em produção:

  - `MillisBehindLatest`: na maioria dos casos, é recomendável definir esse alarme para que seja acionado quando o aplicativo estiver 1 hora atrás dos dados mais recentes, por 1 minuto, em média. Para aplicativos com necessidades end-to-end de processamento mais baixas, você pode ajustar isso para uma tolerância menor. Esse alarme pode ajudá-lo a garantir que o aplicativo está lendo os dados mais recentes.

- Para evitar a exceção `ReadProvisionedThroughputException`, restrinja o número de aplicativos de produção que leem o mesmo streaming de dados do Kinesis a dois aplicativos.

  > **Note**
  >
  > Nesse caso, aplicativo refere-se a qualquer aplicativo que possa ler a origem de streaming. Somente um aplicativo Kinesis Data Analytics pode ler um stream de entrega do Firehose. No entanto, muitos aplicativos podem ler de um stream de dados do Kinesis, como um aplicativo Kinesis Data Analytics ou. AWS Lambda O limite de aplicativo recomendado representa todos os aplicativos configurados para ler uma origem de streaming.

  O Amazon Kinesis Data Analytics lê uma origem de streaming aproximadamente uma vez a cada segundo por aplicativo. No entanto, um aplicativo atrasado pode ler dados a uma taxa mais rápida para sair do atraso. Para permitir que o throughput adequado dos aplicativos seja alcançado, limite o número de aplicativos que leem a mesma fonte de dados.

- Limite o número de aplicativos de produção que leem do mesmo stream de distribuição do Firehose para um aplicativo.

  Um stream de entrega do Firehose pode gravar em destinos como Amazon S3 e Amazon Redshift. Pode ser também uma origem de streaming para o aplicativo Kinesis Data Analytics. Portanto, recomendamos que você não configure mais de um aplicativo do Kinesis Data Analytics por stream de entrega do Firehose. Isso ajuda a garantir que o fluxo de entrega também possa realizar a entrega para outros destinos.

Dimensionar aplicativos

Configure o aplicativo para suas futuras necessidades de dimensionamento aumentando de maneira proativa o número de fluxos de entrada no aplicativo a partir do padrão (um). Recomendamos as seguintes opções de linguagem com base no throughput do aplicativo:

- Use vários fluxos e o Kinesis Data Analytics para aplicativos SQL se o aplicativo tiver necessidades de dimensionamento além de 100 MB/segundo.

- Use o Managed Service for Apache Flink Applications se quiser usar um único stream e aplicativo.

> **Note**
>
> Recomendamos revisar periodicamente a métrica `InputProcessing.0kBytes` do seu aplicativo para que você possa planejar com antecedência o uso de vários aplicativos SQL ou migrar para managed-flink/latest/java/ se o throughput de entrada projetado do seu aplicativo exceder 100 MB/seg.

Monitorar aplicativos

Recomendamos criar um CloudWatch alarme `InputProcessing.0kBytes` para que você seja notificado quando seu aplicativo estiver próximo do limite de taxa de transferência de entrada. Isso pode ser útil, pois você pode atualizar a consulta do aplicativo para compensar por um maior throughput, evitando assim a contrapressão e o atraso na análise. Para obter mais informações, consulte Solução de problemas. Isso também pode ser útil se você tiver um mecanismo para reduzir o throughput no upstream.

- O maior throughput que recomendamos para um único fluxo no aplicativo é entre 2 e 20 MB/segundo, dependendo da complexidade da consulta do aplicativo.
- O throughput máximo de streaming que um único aplicativo Kinesis Data Analytics para SQL pode processar é de aproximadamente 100 MB/seg. Isso pressupõe que você aumentou o número de streams no aplicativo para o valor máximo de 64 e aumentou seu limite de KPU para além de 8. Para obter mais informações, consulte Limites.

> **Note**
>
> Recomendamos revisar periodicamente a métrica `InputProcessing.0kBytes` do seu aplicativo para que você possa planejar com antecedência o uso de vários aplicativos SQL ou migrar para managed-flink/latest/java/ se o throughput de entrada projetado do seu aplicativo exceder 100 MB/seg.

Definição do esquema de entrada

Ao configurar a entrada do aplicativo no console, primeiro especifique uma origem de streaming. Em seguida, o console usa a API de descoberta (consulte DiscoverInputSchema) para inferir um schema por amostragem de registros na origem de streaming. O esquema, entre outras coisas, define nomes e tipos de dados das colunas no stream no aplicativo resultante. O console exibe o esquema. É recomendável fazer o seguinte com esse esquema inferido:

- Teste adequadamente o esquema inferido. O processo de descoberta usa somente uma amostra dos registros na origem de streaming para inferir um esquema. Se a origem de streaming tiver vários tipos de registro, pode ser que a API de descoberta não tenha feito amostragem de um ou mais tipos de registro. Essa situação pode resultar em um esquema que não reflete precisamente os dados na origem de streaming.

  Quando o aplicativo for iniciado, esses tipos de registro perdidos poderão resultar em erros de análise. O Amazon Kinesis Data Analytics envia esses registros ao stream de erros de aplicativo. Para reduzir esses erros de análise, é recomendável testar o esquema inferido de modo interativo no console e monitorar o fluxo do aplicativo em busca de registros perdidos.

- A API do Kinesis Data Analytics não permite a especificação da restrição `NOT NULL` em colunas na configuração de entrada. Se desejar restrições `NOT NULL` em colunas no fluxo do aplicativo, crie esses fluxos de aplicativo usando o código do aplicativo. Em seguida, você pode copiar dados de um fluxo de aplicativo para outro que e a restrição será aplicada.

  Qualquer tentativa de inserir linhas com valores `NULL` quando um valor é necessário resultará em um erro. O Kinesis Data Analytics envia esses erros ao fluxo de erros de aplicativo.

- Reduz os tipos de dados inferidos pelo processo de descoberta. O processo de descoberta recomenda colunas e tipos de dados com base em uma amostragem aleatória de registros na origem de streaming. É recomendável que você os analise cuidadosamente e considere a redução desses tipos de dados para cobrir todos os casos possíveis de registros na entrada. Isso garante menos erros de análise em todo o aplicativo enquanto ela está sendo executada. Por exemplo, se um esquema inferido tiver `SMALLINT` como tipo de coluna, talvez seja recomendável alterá-lo para `INTEGER`.

- Use funções SQL no código do aplicativo para tratar quaisquer dados ou colunas não estruturados. Talvez você tenha ter dados ou colunas não estruturados na entrada, como dados de log. Para ver exemplos, consulte Exemplo: Transformação de valores DateTime . Uma forma de tratar esse tipo de dados é definir o esquema com apenas uma coluna do tipo `VARCHAR(N)`, em que `N` é o maior linha possível esperada no fluxo. No código do aplicativo, você pode ler os registros de entrada e usar as funções `String` e `Date Time` para analisar e esquematizar os dados brutos.

- Trate totalmente os dados da origem de streaming que contêm aninhamento com mais de dois níveis de profundidade. Quando os dados de origem forem JSON, você poderá ter aninhamento. A API de descoberta infere um esquema que nivela um nível de aninhamento. No caso de dois níveis de aninhamento, a API de descoberta tenta nivelá-los. Para mais de dois níveis de aninhamento, o suporte a nivelamento é limitado. Para tratar completamente o aninhamento, é necessário modificar manualmente o esquema inferido para atender às suas necessidades. Use uma das seguintes estratégias para fazer isso:

  - Use o caminho de linha JSON para extrair seletivamente apenas os pares de valores de chave necessários ao aplicativo. Um caminho de linha JSON fornece um ponteiro para o par de valores de chave específico que você deseja inserir no aplicativo. Você pode fazer osso para qualquer nível de aninhamento.
  - Use o caminho de linha JSON para retirar seletivamente objetos JSON complexos e, em seguida, use as funções de manipulação de string no código do aplicativo para extrair os dados específicos de que você precisa.

Conexão às saídas

É recomendável que todos os aplicativos tenham pelo menos duas saídas:

- Use o primeiro destino para inserir os resultados das consultas SQL.
- Use o segundo destino para inserir todo o fluxo de erros e enviá-lo para um bucket do S3 por meio de um fluxo de entrega do Firehose.

Criação do código do aplicativo

Recomendamos o seguinte:

- Na instrução SQL, não especifique uma janela de tempo mais extensa do que uma hora, pelos seguintes motivos:
  - Às vezes, um aplicativo precisa ser reiniciado porque você o atualizou ou por motivos internos do Kinesis Data Analytics. Quando ele é reiniciado, todos os dados incluídos na janela devem ser lidos novamente na fonte de dados de streaming. Leva algum tempo para o Kinesis Data Analytics emitir a saída dessa janela.
  - O Kinesis Data Analytics precisa manter tudo o que está relacionado ao estado do aplicativo, bem como dados relevantes, durante esse tempo. Isso consome significativas unidades de processamento do Kinesis Data Analytics.
- Durante o desenvolvimento, use um tamanho de janela pequeno nas instruções SQL para que possa ver os resultados com maior rapidez. Quando você implantar o aplicativo no ambiente de produção, poderá definir o tamanho da janela conforme apropriado.
- Em vez de uma única instrução SQL complexa, é recomendável dividi-la em várias instruções, em cada etapa, e salvar os resultados em fluxos de aplicativo intermediários. Isso pode ajudar a agilizar a depuração.
- Ao usar janelas em cascata, é recomendável usar duas janelas, uma para tempo de processamento e outra para tempo lógico (horário da conclusão ou horário do evento). Para ter mais informações, consulte Time stamps e a coluna ROWTIME.

Teste de aplicativos

Ao mudar o esquema ou o código do aplicativo do Kinesis Data Analytics para seu aplicativo, é recomendável usar um aplicativo de teste para verificar as alterações antes de implantá-las em produção.

Configuração de um aplicativo de teste

Você pode configurar um aplicativo de teste por meio do console ou usando um modelo do AWS CloudFormation . O uso AWS CloudFormation de um modelo ajuda a garantir que as alterações de código feitas no aplicativo de teste e no aplicativo ativo sejam consistentes.

Ao configurar um aplicativo de teste, você pode conectar o aplicativo aos seus dados ativos ou preencher um fluxo com os dados de teste simulados. Recomendamos dois métodos para preencher um fluxo com dados simulados:

- Use o Kinesis Data Generator (KDG). O KDG usa um modelo de dados para enviar dados aleatórios para um fluxo do Kinesis. O KDG é simples de usar, mas não é apropriado para testar relações complexas entre itens de dados, como para aplicativos que detectam pontos de acesso a dados ou anomalias.
- Use um aplicativo Python personalizado para enviar dados mais complexos para um fluxo de dados do Kinesis. Um aplicativo Python pode gerar relações complexas entre itens de dados, como pontos de acesso ou anomalias. Para obter um exemplo de aplicativo Python que envia dados clusterizados em um hotspot de dados, consulte Exemplo: detectar hotspots em um streaming (função HOTSPOTS).

Ao executar seu aplicativo de teste, visualize seus resultados usando um destino (como um stream de entrega do Firehose para um banco de dados do Amazon Redshift) em vez de visualizar seu stream no aplicativo no console. Os dados que são exibidos no console são uma amostra do fluxo e não contêm todos os registros.

Teste de alterações no esquema

Ao alterar um esquema de fluxo de entrada do aplicativo, use o aplicativo de teste para verificar se as seguintes condições são verdadeiras:

- Os dados do fluxo estão sendo compelidos para o tipo de dado correto. Por exemplo, verifique se os dados de data e hora não estão sendo ingeridos no aplicativo como uma string.
- Os dados estão sendo analisados e impelidos para o tipo de dado que você deseja. Se ocorrerem erros de análise ou coerção, você pode visualizá-los no console ou atribuir um destino no fluxo de erros e visualizar os erros no armazenamento do destino.
- Os campos de dados são para dados de caractere têm comprimento suficiente e o aplicativo não está truncando os dados de caractere. Você pode verificar os registros de dados no armazenar de destino para confirmar se os dados do aplicativo não está sendo truncados.

Teste de alterações no código

Para testar alterações no código SQL, é necessário conhecer seu aplicativo nesse domínio. Você deve determinar que saída precisa ser testada e qual a saída correta. Quanto a possíveis áreas problemáticas que deve ser verificadas ao modificar o código SQL do aplicativo, consulte Solução de problemas do Amazon Kinesis Data Analytics para aplicativos SQL.

Solução de problemas do Amazon Kinesis Data Analytics para aplicativos SQL

As considerações a seguir podem ajudar a solucionar problemas que possam surgir com o Amazon Kinesis Data Analytics para aplicativos SQL.

Tópicos

Aplicativos interrompidos

- O que é um aplicativo Kinesis Data Analytics para SQL interrompido?

  Um aplicativo interrompido é um aplicativo que observamos que não processa nenhum registro por um período mínimo de três meses. Isso significa que os clientes estão pagando pelos recursos do Kinesis Data Analytics para SQL que não estão usando.

- Quando AWS começará a interromper os aplicativos ociosos?

  AWS começará a interromper os aplicativos inativos em 14 de novembro de 2023 e concluirá até 21 de novembro de 2023. Interromperemos os aplicativos ociosos no fuso horário de expediente da Região.

- Os aplicativos do Kinesis Data Analytics para SQL interrompidos podem ser reiniciados?

  Sim. Se você precisar reiniciar o aplicativo, poderá fazê-lo normalmente. Não há necessidade de um ticket de suporte.

- Quando um aplicativo ocioso AWS for interrompido, algum resultado da minha consulta também será excluído?

  Não. Primeiro, como seu aplicativo está ocioso, ele não está processando consultas. Segundo, os resultados da consulta não são armazenados no Kinesis Data Analytics para SQL. Você configura seu aplicativo Kinesis Data Analytics para SQL com um destino de coletor para onde os resultados de seus cálculos são enviados (por exemplo, no Amazon S3 ou em outro fluxo de dados). Dessa forma, você mantém a propriedade total dos seus dados e eles permanecerão recuperáveis de acordo com os termos desse serviço de armazenamento.
- O que faço se eu não quiser que minha inscrição seja interrompida?

  Você pode enviar um e-mail para a equipe de atendimento (kda-sql-questions@amazon.com) solicitando que as inscrições não sejam interrompidas em nenhum momento antes de 10 de novembro de 2023. O e-mail deve incluir a ID da sua conta e o ARN do aplicativo.

Não é possível executar o código SQL

Se você não souber o que fazer para que uma instrução SQL funcione corretamente, o Kinesis Data Analytics tem vários recursos para ajudá-lo nesse sentido:

- Para obter mais informações sobre instruções do &SQL;, consulte Exemplos de Kinesis Data Analytics para SQL. Esta seção fornece uma série de exemplos de SQL que você pode usar.
- A Amazon Kinesis Data Analytics SQL Reference fornece um guia detalhado para a criação de instruções SQL de streaming.
- Se o problema persistir, recomendamos que você faça uma pergunta nos Fóruns do Kinesis Data Analytics.

Não foi possível detectar ou descobrir meu esquema

Em alguns casos, o Kinesis Data Analytics não consegue detectar ou descobrir um esquema. Em muitos desses casos, você ainda poderá usar o Kinesis Data Analytics.

Suponhamos que você possua dados codificados por UTF-8 que não usam delimitador, dados que usam um formato diferente do formato de valores separados por vírgulas (CSV) ou uma API de descoberta que não tenha detectado o esquema. Nesses casos, você pode definir um esquema manualmente ou pode usar funções de manipulação de string para estruturar os dados.

Para descobrir o esquema do seu fluxo, o Kinesis Data Analytics faz amostras aleatórias dos dados mais recentes do fluxo. Se você não estiver enviando dados ao stream de modo consistente, o Kinesis Data Analytics poderá não ser capaz de recuperar uma amostra e detectar um esquema. Para obter mais informações, consulte Usar o recurso de descoberta de esquema em dados em streaming.

Dados de referência desatualizados

Os dados de referência são carregados do objeto do Amazon Simple Storage Service (Amazon S3) no aplicativo quando ele é iniciado/atualizado ou durante interrupções de aplicativo causadas por problemas de serviço.

Os dados de referência não são carregados no aplicativo quando há atualizações no objeto do Amazon S3.

Se os dados de referência no aplicativo não estiverem atualizados, você poderá recarregar os dados seguindo estas etapas:

1. No console do Kinesis Data Analytics, escolha o nome do aplicativo na lista e selecione Application details.
2. Escolha Go to SQL editor (Ir para o editor SQL) para abrir a página Real-time analytics (Análise em tempo real) do aplicativo.
3. Na visualização Source Data (Dados de origem), escolha o nome da sua tabela de dados de referência.
4. Escolha Actions (Ações), Synchronize reference data table (Sincronizar tabela de dados de referência).

Aplicativo não gravando no destino

Se os dados não estiverem sendo gravados no destino, confira o seguinte:

- Verifique se a função do aplicativo tem permissão suficiente para acessar o destino. Para ter mais informações, consulte Política de permissões para gravação em um stream do Kinesis ou Política de permissões para gravação em um stream de entrega do Firehose.
- Verifique se o destino do aplicativo está configurado corretamente e se o aplicativo está usando o nome correto para o stream de saída.

- Verifique as métricas do Amazon CloudWatch para o fluxo de saída para ver se os dados estão sendo gravados. Para ter informações sobre o uso de métricas do CloudWatch, consulte Monitoramento com a Amazon CloudWatch.
- Adicione um fluxo de logs do CloudWatch usando o the section called "AddApplicationCloudWatchLoggingOption". O aplicativo gravará erros de configuração no streaming de log.

Se a função e configuração de destino parecerem corretas, tente reiniciar o aplicativo, especificando `LAST_STOPPED_POINT` para a InputStartingPositionConfiguration.

Importantes parâmetros de integridade de aplicativo a serem monitorados

Para assegura que seu aplicativo está sendo executado corretamente, é recomendável que você monitore determinados parâmetros importantes.

O parâmetro mais importante a ser monitorado é a métrica `MillisBehindLatest` do Amazon CloudWatch. Essa métrica representa seu tempo de atraso na leitura do fluxo. Ela ajuda você a determinar se está processando registros no fluxo de origem com a rapidez suficiente.

Como regra geral, você deve configurar um alarme do CloudWatch a ser acionado se o atraso na leitura for superior a uma hora. No entanto, a quantidade de tempo depende do seu caso de uso. Você pode ajustá-lo conforme necessário.

Para obter mais informações, consulte Práticas recomendadas.

Erros de código inválidos ao executar um aplicativo

Quando você não conseguir salvar e executar o código SQL do aplicativo Amazon Kinesis Data Analytics, estas serão as causas mais comuns:

- O fluxo foi redefinido no código SQL: após criar um fluxo e a pump associada ao fluxo, você não poderá redefinir o mesmo fluxo no código. Para obter mais informações sobre como criar um fluxo, consulte CREATE STREAM na Amazon Kinesis Data Analytics SQL Reference. Para obter mais informações sobre como criar uma bomba, consulte CREATE PUMP.

- Uma cláusula GROUP BY usa várias colunas ROWTIME : você pode especificar apenas uma coluna ROWTIME na cláusula GROUP BY. Para obter mais informações, consulte GROUP BY e ROWTIME na Amazon Kinesis Data Analytics SQL Reference.
- Um ou mais tipos de dados têm um casting inválido: neste caso, o código tem um cast implícito inválido. Por exemplo, você pode converter um `timestamp` em um `bigint` no código.
- O nome de um fluxo é igual ao nome de um fluxo reservado por serviço – Um fluxo não pode ter o mesmo nome do fluxo reservado por serviço `error_stream`.

O aplicativo está gravando erros no stream de erros

Se o seu aplicativo estiver gravando erros no stream de erros no aplicativo, você pode decodificar o valor no campo `DATA_ROW` usando as bibliotecas padrão. Para obter mais informações sobre o stream de erros, consulte Como tratar erros.

Throughput insuficiente ou MillisBehindLatest alta

Se a métrica MillisBehindLatest do aplicativo estiver aumentando de forma constante ou estiver consistentemente acima de 1.000 (um segundo), pode ser devido aos seguintes motivos:

- Verifique a métrica InputBytes do CloudWatch de seu aplicativo. Se você está recebendo mais de 4 MB/s, isso pode causar um aumento na `MillisBehindLatest`. Para melhorar o throughput do aplicativo, aumente o valor do parâmetro `InputParallelism`. Para obter mais informações, consulte Paralelização dos streams de entrada para aumentar a taxa de transferência.
- Verifique a métrica Success de entrega de saída do aplicativo para falhas na entrega ao seu destino. Verifique se você configurou corretamente a saída e se seu fluxo de saída tem capacidade suficiente.
- Se o seu aplicativo usa uma função do AWS Lambda para pré-processamento ou como uma saída, verifique a métrica InputProcessing.Duration ou LambdaDelivery.Duration do CloudWatch. Se a duração da invocação da função do Lambda for maior do que 5 segundos, considere fazer o seguinte:
  - Aumente a alocação de Memória da função do Lambda. Você pode fazer isso no console do AWS Lambda, na página Configuration (Configurações) em Basic settings (Configurações básicas). Para obter mais informações, consulte Configuração de funções do Lambda no Guia do desenvolvedor do AWS Lambda.

  - Aumente o número de estilhaços no seu fluxo de entrada do aplicativo. Isso aumenta o número de funções paralelas que o aplicativo invocará, o que pode aumentar o throughput.
  - Verifique se a função não está fazendo chamadas de bloqueio que afetam o desempenho, como solicitações síncronas para recursos externos.
  - Examine sua função AWS Lambda para ver se há outras áreas onde você pode melhorar a performance. Verifique os logs do CloudWatch da função do Lambda do aplicativo. Para obter mais informações, consulte Acessar as métricas do Amazon CloudWatch para no Guia do desenvolvedor do AWS Lambda.
- Verifique se o aplicativo não está atingindo o limite padrão para unidades de processamento do Kinesis (KPU). Se o seu aplicativo estiver atingindo esse limite, você pode solicitar um aumento de limite. Para obter mais informações, consulte Escalabilidade automática de aplicativos para aumentar a taxa de transferência.
- Se o problema do seu aplicativo persistir após o aumento do limite de KPU, verifique se o throughput de entrada do aplicativo não excede 100 MB/seg. Se exceder 100 MB/seg, recomendamos implementar alterações para reduzir o throughput geral para estabilizar o aplicativo, por exemplo, reduzindo a quantidade de dados enviados para a fonte de dados de onde o aplicativo Kinesis Data Analytics Sql lê. Também recomendamos outras abordagens, incluindo aumentar o paralelismo do aplicativo, reduzir o período de tempo dos cálculos, alterar os tipos de dados colunares do VARCHAR para tipos de dados com tamanhos menores (por exemplo, INTEGER, LONG etc.) e reduzir os dados processados por amostragem ou filtragem.

> **Note**
>
> Recomendamos revisar periodicamente a métrica `InputProcessing.0kBytes` do seu aplicativo para que você possa planejar com antecedência o uso de vários aplicativos SQL ou migrar para managed-flink/latest/java/ se o throughput de entrada projetada do seu aplicativo exceder 100 MB/seg.

Referência SQL do Kinesis Data Analytics

Para obter mais informações sobre os elementos da linguagem SQL compatíveis com o Kinesis Data Analytics, consulte a Referência SQL do Kinesis Data Analytics.

Referência da API

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Amazon Managed Service for Apache Flink API V2 Documentation.

Você pode usar o AWS CLI para explorar a API Amazon Kinesis Data Analytics. Este guia fornece exercícios Conceitos básicos do Amazon Kinesis Data Analytics para aplicativos SQL que usam a AWS CLI.

Tópicos

Ações

As ações a seguir são compatíveis:

- AddApplicationCloudWatchLoggingOption
- AddApplicationInput
- AddApplicationInputProcessingConfiguration
- AddApplicationOutput
- AddApplicationReferenceDataSource
- CreateApplication
- DeleteApplication
- DeleteApplicationCloudWatchLoggingOption
- DeleteApplicationInputProcessingConfiguration
- DeleteApplicationOutput
- DeleteApplicationReferenceDataSource

- DescribeApplication
- DiscoverInputSchema
- ListApplications
- ListTagsForResource
- StartApplication
- StopApplication
- TagResource
- UntagResource
- UpdateApplication

AddApplicationCloudWatchLoggingOption

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Adiciona um fluxo de CloudWatch log para monitorar erros de configuração do aplicativo. Para obter mais informações sobre o uso de streams de CloudWatch log com aplicativos Amazon Kinesis Analytics, consulte Como trabalhar com o Amazon Logs. CloudWatch

Sintaxe da Solicitação

```
{
   "ApplicationName": "string",
   "CloudWatchLoggingOption": {
      "LogStreamARN": "string",
      "RoleARN": "string"
   },
   "CurrentApplicationVersionId": number
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

O nome do aplicativo Kinesis Analytics.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

CloudWatchLoggingOption

Fornece o Amazon Resource Name (ARN) do stream de CloudWatch logs e o ARN da função do IAM. Observação: para gravar mensagens do aplicativo CloudWatch, a função do IAM usada deve ter a ação `PutLogEvents` de política ativada.

Tipo: objeto CloudWatchLoggingOption

Obrigatório: Sim

CurrentApplicationVersionId

O ID da versão do aplicativo Kinesis Analytics.

Tipo: longo

Faixa válida: valor mínimo de 1. Valor máximo de 999999999.

Obrigatório: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

AddApplicationInput

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Adiciona uma origem de streaming ao seu aplicativo Amazon Kinesis. Para obter informações conceituais, consulte Configuração da entrada do aplicativo.

Você pode adicionar uma origem de streaming ao criar um aplicativo ou usar essa operação para adicionar uma origem de streaming depois de criar um aplicativo. Para obter mais informações, consulte CreateApplication.

Qualquer atualização da configuração, incluindo a adição de uma origem de streaming usando essa operação, resulta em uma nova versão do aplicativo. Você pode usar a DescribeApplicationoperação para encontrar a versão atual do aplicativo.

Essa operação exige permissões para executar a ação `kinesisanalytics:AddApplicationInput`.

Sintaxe da Solicitação

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "Input": {
      "InputParallelism": {
         "Count": number
      },
      "InputProcessingConfiguration": {
         "InputLambdaProcessor": {
            "ResourceARN": "string",
            "RoleARN": "string"
         }
      },
      "InputSchema": {
```

```
         "RecordColumns": [
            {
               "Mapping": "string",
               "Name": "string",
               "SqlType": "string"
            }
         ],
         "RecordEncoding": "string",
         "RecordFormat": {
            "MappingParameters": {
               "CSVMappingParameters": {
                  "RecordColumnDelimiter": "string",
                  "RecordRowDelimiter": "string"
               },
               "JSONMappingParameters": {
                  "RecordRowPath": "string"
               }
            },
            "RecordFormatType": "string"
         }
      },
      "KinesisFirehoseInput": {
         "ResourceARN": "string",
         "RoleARN": "string"
      },
      "KinesisStreamsInput": {
         "ResourceARN": "string",
         "RoleARN": "string"
      },
      "NamePrefix": "string"
   }
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

Nome do seu aplicativo Amazon Kinesis Analytics existente ao qual você deseja adicionar a origem de streaming.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

CurrentApplicationVersionId

Versão atual do seu aplicativo Amazon Kinesis Analytics. Você pode usar a DescribeApplicationoperação para encontrar a versão atual do aplicativo.

Tipo: longo

Faixa válida: valor mínimo de 1. Valor máximo de 999999999.

Obrigatório: Sim

Input

A entrada a ser adicionada.

Tipo: objeto Input

Obrigatório: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

CodeValidationException

O código do aplicativo fornecido pelo usuário (consulta) é inválido. Pode ser um simples erro de sintaxe.

Código de Status HTTP: 400

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

AddApplicationInputProcessingConfiguration

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Adiciona um InputProcessingConfigurationa um aplicativo. Um processador de entrada pré-processa registros no stream de entrada antes do código SQL do aplicativo ser executado. Atualmente, o único processador de entrada disponível é o AWS Lambda.

Sintaxe da Solicitação

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "InputId": "string",
   "InputProcessingConfiguration": {
      "InputLambdaProcessor": {
         "ResourceARN": "string",
         "RoleARN": "string"
      }
   }
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

O nome do aplicativo ao qual você deseja adicionar a configuração de processamento de entrada.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

CurrentApplicationVersionId

Versão do aplicativo ao qual você deseja adicionar a configuração de processamento de entrada. Você pode usar a DescribeApplicationoperação para obter a versão atual do aplicativo. Se a versão especificada não for a versão atual, `ConcurrentModificationException` será retornado.

Tipo: longo

Faixa válida: valor mínimo de 1. Valor máximo de 999999999.

Obrigatório: Sim

InputId

O ID da configuração de entrada a qual adicionar a configuração de processamento de entrada. Você pode obter uma lista dos IDs de entrada de um aplicativo usando a DescribeApplicationoperação.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

InputProcessingConfiguration

O InputProcessingConfigurationpara adicionar ao aplicativo.

Tipo: objeto InputProcessingConfiguration

Obrigatório: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2

- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

AddApplicationOutput

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Adiciona um destino externo ao aplicativo do Amazon Kinesis Analytics.

Se você quiser que o Amazon Kinesis Analytics entregue dados de um stream no aplicativo para um destino externo (como um stream do Amazon Kinesis, um stream do Amazon Kinesis Firehose ou uma função AWS Lambda), adicione a configuração relevante ao seu aplicativo usando essa operação. Você pode configurar uma ou mais saídas para seu aplicativo. Cada configuração de saída mapeia um stream no aplicativo e um destino externo.

É possível usar uma das configurações de saída para entregar dados de um stream de erros no aplicativo para um destino externo para que você possa analisar os erros. Para obter mais informações, consulte Compreensão da saída do aplicativo (destino).

Qualquer atualização da configuração, incluindo a adição de uma origem de streaming usando essa operação, resulta em uma nova versão do aplicativo. Você pode usar a DescribeApplicationoperação para encontrar a versão atual do aplicativo.

Para obter os limites do número de entradas e saídas do aplicativo que você pode configurar, consulte Limites.

Essa operação exige permissões para executar a ação `kinesisanalytics:AddApplicationOutput`.

Sintaxe da Solicitação

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "Output": {
      "DestinationSchema": {
```

```
            "RecordFormatType": "string"
         },
         "KinesisFirehoseOutput": {
            "ResourceARN": "string",
            "RoleARN": "string"
         },
         "KinesisStreamsOutput": {
            "ResourceARN": "string",
            "RoleARN": "string"
         },
         "LambdaOutput": {
            "ResourceARN": "string",
            "RoleARN": "string"
         },
         "Name": "string"
      }
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

O nome do aplicativo ao qual você deseja adicionar a configuração de saída.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

CurrentApplicationVersionId

Versão do aplicativo ao qual você deseja adicionar a configuração de saída. Você pode usar a DescribeApplicationoperação para obter a versão atual do aplicativo. Se a versão especificada não for a versão atual, `ConcurrentModificationException` será retornado.

Tipo: longo

Faixa válida: valor mínimo de 1. Valor máximo de 999999999.

Obrigatório: Sim

Output

Uma matriz de objetos, cada qual descrevendo uma configuração de saída. Na configuração de saída, você especifica o nome de um stream no aplicativo, um destino (ou seja, um stream do Amazon Kinesis, um stream de entrega do Amazon Kinesis Firehose ou AWS uma função Lambda) e registra a formação a ser usada ao gravar no destino.

Tipo: objeto Output

Obrigatório: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

AddApplicationReferenceDataSource

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Adiciona uma fonte de dados de referência a um aplicativo existente.

O Amazon Kinesis Analytics lê os dados de referência (ou seja, um objeto do Amazon S3) e cria uma tabela no aplicativo em seu aplicativo. Na solicitação, você fornece a origem (nome do bucket do S3 e nome da chave do objeto), o nome da tabela no aplicativo a ser criada e as informações do mapeamento necessárias que descrevem como os dados no objeto do Amazon S3 são mapeados para colunas na tabela no aplicativo resultante.

Para obter informações conceituais, consulte Configuração da entrada do aplicativo. Para obter os limites nas fontes de dados que você pode adicionar ao aplicativo, consulte Limites.

Essa operação exige permissões para executar a ação `kinesisanalytics:AddApplicationOutput`.

Sintaxe da Solicitação

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "ReferenceDataSource": {
      "ReferenceSchema": {
         "RecordColumns": [
            {
               "Mapping": "string",
               "Name": "string",
               "SqlType": "string"
            }
         ],
         "RecordEncoding": "string",
         "RecordFormat": {
```

```
            "MappingParameters": {
               "CSVMappingParameters": {
                  "RecordColumnDelimiter": "string",
                  "RecordRowDelimiter": "string"
               },
               "JSONMappingParameters": {
                  "RecordRowPath": "string"
               }
            },
            "RecordFormatType": "string"
         }
      },
      "S3ReferenceDataSource": {
         "BucketARN": "string",
         "FileKey": "string",
         "ReferenceRoleARN": "string"
      },
      "TableName": "string"
   }
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

O nome de um aplicativo existente.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

CurrentApplicationVersionId

Versão do aplicativo ao qual você está adicionando a fonte de dados de referência. Você pode usar a DescribeApplicationoperação para obter a versão atual do aplicativo. Se a versão especificada não for a versão atual, `ConcurrentModificationException` será retornado.

Tipo: longo

Faixa válida: valor mínimo de 1. Valor máximo de 999999999.

Obrigatório: Sim

ReferenceDataSource

A fonte de dados de referência pode ser um objeto em seu bucket do Amazon S3. O Amazon Kinesis Analytics lê o objeto e copia os dados na tabela no aplicativo que é criada. Você fornece um bucket do S3, o nome da chave do objeto e a tabela no aplicativo resultante que é criada. Forneça também uma função do IAM com as permissões necessárias que o Amazon Kinesis Analytics pode assumir para ler o objeto no bucket do S3 em seu nome.

Tipo: objeto ReferenceDataSource

Obrigatório: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

CreateApplication

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Cria um aplicativo do Amazon Kinesis Analytics. Você pode configurar cada aplicativo com uma origem de streaming como entrada, código do aplicativo para processar a entrada e até três destinos nos quais você deseja que o Amazon Kinesis Analytics grave os dados de saída do seu aplicativo. Para obter uma visão geral, consulte Como funciona.

Na configuração de entrada, você mapeia a fonte de streaming para um stream no aplicativo, que pode ser considerado uma tabela em constante atualização. No mapeamento, você deve fornecer um esquema para o stream no aplicativo e mapear cada coluna no stream no aplicativo para um elemento na origem do streaming.

O código do seu aplicativo é uma ou mais instruções do SQL que leem dados de entrada, transformam esses dados e geram uma saída. O código do seu aplicativo pode criar um ou mais artefatos SQL, como streams ou pumps SQL.

Na configuração de saída, você pode configurar o aplicativo para gravar dados de streams no aplicativo criados em seus aplicativos em até três destinos.

Para ler dados do seu stream de origem ou gravar dados nos streams de destino, o Amazon Kinesis Analytics precisa das suas permissões. Você concede essas permissões criando um perfil do IAM. Essa operação exige permissões para executar a ação `kinesisanalytics:CreateApplication`.

Para exercícios introdutórios para criar um aplicativo Amazon Kinesis Analytics, consulte Conceitos básicos.

Sintaxe da Solicitação

```
{
   "ApplicationCode": "string",
   "ApplicationDescription": "string",
```

```
   "ApplicationName": "string",
   "CloudWatchLoggingOptions": [
      {
         "LogStreamARN": "string",
         "RoleARN": "string"
      }
   ],
   "Inputs": [
      {
         "InputParallelism": {
            "Count": number
         },
         "InputProcessingConfiguration": {
            "InputLambdaProcessor": {
               "ResourceARN": "string",
               "RoleARN": "string"
            }
         },
         "InputSchema": {
            "RecordColumns": [
               {
                  "Mapping": "string",
                  "Name": "string",
                  "SqlType": "string"
               }
            ],
            "RecordEncoding": "string",
            "RecordFormat": {
               "MappingParameters": {
                  "CSVMappingParameters": {
                     "RecordColumnDelimiter": "string",
                     "RecordRowDelimiter": "string"
                  },
                  "JSONMappingParameters": {
                     "RecordRowPath": "string"
                  }
               },
               "RecordFormatType": "string"
            }
         },
         "KinesisFirehoseInput": {
            "ResourceARN": "string",
            "RoleARN": "string"
         },
```

```
            "KinesisStreamsInput": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "NamePrefix": "string"
         }
      ],
      "Outputs": [
         {
            "DestinationSchema": {
               "RecordFormatType": "string"
            },
            "KinesisFirehoseOutput": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "KinesisStreamsOutput": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "LambdaOutput": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "Name": "string"
         }
      ],
      "Tags": [
         {
            "Key": "string",
            "Value": "string"
         }
      ]
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationCode

Uma ou mais instruções do SQL que leem dados de entrada, transformam esses dados e geram uma saída. Por exemplo, você pode gravar uma instrução SQL que lê dados de um stream no

aplicativo, gerar uma média de execução do número de cliques em anúncios por fornecedor e inserir as linhas resultantes em outro stream no aplicativo usando bombas. Para obter mais informações sobre o padrão típico, consulte Código do aplicativo.

Você pode fornecer essa série de instruções SQL em que a saída de uma instrução pode ser usada como entrada para a próxima instrução. Você armazena os resultados intermediários criando bombas e streams no aplicativo.

Observe que o código do aplicativo deve criar streams com nomes especificados em `Outputs`. Por exemplo, se seu `Outputs` definir streams de saída chamados `ExampleOutputStream1` e `ExampleOutputStream2`, o código do aplicativo deverá criar esses streams.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 0. Tamanho máximo de 102.400.

Obrigatório: não

ApplicationDescription

Descrição resumida do aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 0. Tamanho máximo de 1.024.

Obrigatório: não

ApplicationName

Nome do seu aplicativo Amazon Kinesis Analytics (por exemplo, `sample-app`).

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

CloudWatchLoggingOptions

Use esse parâmetro para configurar um fluxo de CloudWatch log para monitorar erros de configuração do aplicativo. Para obter mais informações, consulte Como trabalhar com Amazon CloudWatch Logs.

Tipo: matriz de objetos CloudWatchLoggingOption

Obrigatório: não

Inputs

Use este parâmetro para configurar a entrada do aplicativo.

É possível configurar o aplicativo para receber entrada de uma única origem de streaming. Nessa configuração, você mapeia essa origem de streaming para um stream no aplicativo que é criado. O código do aplicativo pode consultar o stream no aplicativo como uma tabela (você pode considerar isso como uma tabela que é constantemente atualizada).

Para a origem de streaming, você fornece o nome de recurso da Amazon (ARN) e o formato de dados no stream (por exemplo, JSON, CSV etc.). Você também deve fornecer uma função do IAM que o Amazon Kinesis Analytics pode assumir para ler esse stream em seu nome.

Para criar o stream no aplicativo, especifique um esquema para transformar seus dados em uma versão esquematizada usada no SQL. No esquema, você fornece o mapeamento necessário dos elementos de dados na origem do streaming para registrar colunas no stream no aplicativo.

Tipo: matriz de objetos Input

Obrigatório: não

Outputs

Você pode configurar a saída do aplicativo para gravar dados de qualquer um dos streams no aplicativo em até três destinos.

Esses destinos podem ser streams do Amazon Kinesis, streams de entrega do Amazon Kinesis Firehose, destinos Lambda AWS ou qualquer combinação dos três.

Na configuração, você especifica o nome do stream no aplicativo, o stream de destino ou o nome do recurso da Amazon (ARN) da função do Lambda e o formato a ser usado ao gravar dados. Você também deve fornecer um perfil do IAM que o Amazon Kinesis Analytics possa assumir para gravar no stream de destino ou na função do Lambda em seu nome.

Na configuração de saída, você também fornece o stream de saída ou o ARN da função do Lambda. Para destinos de stream, você fornece o formato dos dados no stream (por exemplo, JSON, CSV). Forneça também um perfil do IAM que o Amazon Kinesis possa assumir para gravar no stream ou na função do Lambda em seu nome.

Tipo: matriz de objetos Output

Obrigatório: não

Tags

Uma lista de uma ou mais tags a serem atribuídas ao aplicativo. Uma tag é um par chave-valor que identifica um aplicativo. Observe que o número máximo de tags do aplicativo inclui as tags do sistema. O número máximo de tags do aplicativo definidas pelo usuário é de 50. Para obter mais informações, consulte Uso de tags.

Tipo: Matriz de objetos Tag

Membros da Matriz: Número mínimo de 1 item. Número máximo de 200 itens.

Obrigatório: Não

Sintaxe da Resposta

```
{
   "ApplicationSummary": {
      "ApplicationARN": "string",
      "ApplicationName": "string",
      "ApplicationStatus": "string"
   }
}
```

Elementos de Resposta

Se a ação for bem-sucedida, o serviço retornará uma resposta HTTP 200.

Os dados a seguir são retornados no formato JSON pelo serviço.

ApplicationSummary

Em resposta à sua solicitação `CreateApplication`, o Amazon Kinesis Analytics retorna uma resposta com um resumo do aplicativo criado, incluindo o nome do recurso da Amazon (ARN), nome e status do aplicativo.

Tipo: objeto ApplicationSummary

Erros

CodeValidationException

O código do aplicativo fornecido pelo usuário (consulta) é inválido. Pode ser um simples erro de sintaxe.

Código de Status HTTP: 400

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

LimitExceededException

Excedeu o número de inscrições permitidas.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

TooManyTagsException

Aplicação criada com muitas tags ou muitas tags adicionadas a uma aplicação. Observe que o número máximo de tags do aplicativo inclui as tags do sistema. O número máximo de tags do aplicativo definidas pelo usuário é de 50.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DeleteApplication

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Exclui o aplicativo especificado. O Amazon Kinesis Analytics interrompe a execução do aplicativo e exclui o aplicativo, incluindo quaisquer artefatos do aplicativo (como streams no aplicativo, tabela de referência e código do aplicativo).

Essa operação exige permissões para executar a ação `kinesisanalytics:DeleteApplication`.

Sintaxe da Solicitação

```
{
   "ApplicationName": "string",
   "CreateTimestamp": number
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

Nome do aplicativo Amazon Kinesis Analytics a ser excluído.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

CreateTimestamp

Para obter esse valor, você pode usar a operação `DescribeApplication`.

Tipo: carimbo de data/hora

Obrigatório: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DeleteApplicationCloudWatchLoggingOption

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Exclui um fluxo de CloudWatch log de um aplicativo. Para obter mais informações sobre o uso de streams de CloudWatch log com aplicativos Amazon Kinesis Analytics, consulte Como trabalhar com o Amazon Logs. CloudWatch

Sintaxe da Solicitação

```
{
   "ApplicationName": "string",
   "CloudWatchLoggingOptionId": "string",
   "CurrentApplicationVersionId": number
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

O nome do aplicativo Kinesis Analytics.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

CloudWatchLoggingOptionId

A opção `CloudWatchLoggingOptionId` de CloudWatch registro a ser excluída. Você pode obtê-lo `CloudWatchLoggingOptionId` usando a DescribeApplicationoperação.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

CurrentApplicationVersionId

O ID da versão do aplicativo Kinesis Analytics.

Tipo: longo

Faixa válida: valor mínimo de 1. Valor máximo de 999999999.

Obrigatório: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DeleteApplicationInputProcessingConfiguration

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Exclui um InputProcessingConfiguration de uma entrada.

Sintaxe da Solicitação

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "InputId": "string"
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

O nome do aplicativo Kinesis Analytics.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

CurrentApplicationVersionId

O ID da versão do aplicativo Kinesis Analytics.

Tipo: longo

Faixa válida: valor mínimo de 1. Valor máximo de 999999999.

Obrigatório: Sim

InputId

O ID da configuração de entrada da qual excluir a configuração de processamento de entrada. Você pode obter uma lista dos IDs de entrada de um aplicativo usando a DescribeApplicationoperação.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DeleteApplicationOutput

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Exclui a configuração de destino de saída da configuração do seu aplicativo. O Amazon Kinesis Analytics não gravará mais no destino de saída externo os dados do stream correspondente no aplicativo.

Essa operação exige permissões para executar a ação `kinesisanalytics:DeleteApplicationOutput`.

Sintaxe da Solicitação

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "OutputId": "string"
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

Nome do aplicativo Amazon Kinesis Analytics.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

CurrentApplicationVersionId

Versão do aplicativo Amazon Kinesis Analytics. Você pode usar a DescribeApplicationoperação para obter a versão atual do aplicativo. Se a versão especificada não for a versão atual, `ConcurrentModificationException` será retornado.

Tipo: longo

Faixa válida: valor mínimo de 1. Valor máximo de 999999999.

Obrigatório: Sim

OutputId

ID da configuração a excluir. Cada configuração de saída adicionada ao aplicativo, quando o aplicativo é criado ou posteriormente usando a AddApplicationOutputoperação, tem uma ID exclusiva. É preciso fornecer o ID para identificar com exclusividade a configuração de saída que você deseja excluir da configuração da aplicação. Você pode usar a DescribeApplicationoperação para obter o específico`OutputId`.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DeleteApplicationReferenceDataSource

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Exclui uma configuração de fonte de dados de referência da configuração do aplicativo especificado.

Se o aplicativo estiver em execução, o Amazon Kinesis Analytics removerá imediatamente a tabela no aplicativo que você criou usando AddApplicationReferenceDataSourcea operação.

Essa operação exige permissões para executar a ação `kinesisanalytics.DeleteApplicationReferenceDataSource`.

Sintaxe da Solicitação

```
{
   "ApplicationName": "string",
   "CurrentApplicationVersionId": number,
   "ReferenceId": "string"
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

O nome de um aplicativo existente.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

CurrentApplicationVersionId

Versão do aplicativo. Você pode usar a DescribeApplicationoperação para obter a versão atual do aplicativo. Se a versão especificada não for a versão atual, `ConcurrentModificationException` será retornado.

Tipo: longo

Faixa válida: valor mínimo de 1. Valor máximo de 999999999.

Obrigatório: Sim

ReferenceId

ID da fonte de dados de referência. Quando você adiciona uma fonte de dados de referência ao seu aplicativo usando o AddApplicationReferenceDataSource, o Amazon Kinesis Analytics atribui uma ID. Você pode usar a DescribeApplicationoperação para obter o ID de referência.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DescribeApplication

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Retorna informações sobre um aplicativo específico do Amazon Kinesis Analytics.

Se você quiser recuperar uma lista de todos os aplicativos em sua conta, use a ListApplicationsoperação.

Essa operação exige permissões para executar a ação `kinesisanalytics:DescribeApplication`. Você pode usar `DescribeApplication` para obter o versionId atual do aplicativo, que você precisa para chamar outras operações, como `Update`.

Sintaxe da Solicitação

```
{
   "ApplicationName": "string"
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

Nome do aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

Sintaxe da Resposta

```
{
   "ApplicationDetail": {
      "ApplicationARN": "string",
      "ApplicationCode": "string",
      "ApplicationDescription": "string",
      "ApplicationName": "string",
      "ApplicationStatus": "string",
      "ApplicationVersionId": number,
      "CloudWatchLoggingOptionDescriptions": [
         {
            "CloudWatchLoggingOptionId": "string",
            "LogStreamARN": "string",
            "RoleARN": "string"
         }
      ],
      "CreateTimestamp": number,
      "InputDescriptions": [
         {
            "InAppStreamNames": [ "string" ],
            "InputId": "string",
            "InputParallelism": {
               "Count": number
            },
            "InputProcessingConfigurationDescription": {
               "InputLambdaProcessorDescription": {
                  "ResourceARN": "string",
                  "RoleARN": "string"
               }
            },
            "InputSchema": {
               "RecordColumns": [
                  {
                     "Mapping": "string",
                     "Name": "string",
                     "SqlType": "string"
                  }
               ],
               "RecordEncoding": "string",
               "RecordFormat": {
                  "MappingParameters": {
                     "CSVMappingParameters": {
```

```
                        "RecordColumnDelimiter": "string",
                        "RecordRowDelimiter": "string"
                     },
                     "JSONMappingParameters": {
                        "RecordRowPath": "string"
                     }
                  },
                  "RecordFormatType": "string"
               }
            },
            "InputStartingPositionConfiguration": {
               "InputStartingPosition": "string"
            },
            "KinesisFirehoseInputDescription": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "KinesisStreamsInputDescription": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "NamePrefix": "string"
         }
      ],
      "LastUpdateTimestamp": number,
      "OutputDescriptions": [
         {
            "DestinationSchema": {
               "RecordFormatType": "string"
            },
            "KinesisFirehoseOutputDescription": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "KinesisStreamsOutputDescription": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "LambdaOutputDescription": {
               "ResourceARN": "string",
               "RoleARN": "string"
            },
            "Name": "string",
            "OutputId": "string"
```

```
            }
         ],
         "ReferenceDataSourceDescriptions": [
            {
               "ReferenceId": "string",
               "ReferenceSchema": {
                  "RecordColumns": [
                     {
                        "Mapping": "string",
                        "Name": "string",
                        "SqlType": "string"
                     }
                  ],
                  "RecordEncoding": "string",
                  "RecordFormat": {
                     "MappingParameters": {
                        "CSVMappingParameters": {
                           "RecordColumnDelimiter": "string",
                           "RecordRowDelimiter": "string"
                        },
                        "JSONMappingParameters": {
                           "RecordRowPath": "string"
                        }
                     },
                     "RecordFormatType": "string"
                  }
               },
               "S3ReferenceDataSourceDescription": {
                  "BucketARN": "string",
                  "FileKey": "string",
                  "ReferenceRoleARN": "string"
               },
               "TableName": "string"
            }
         ]
      }
}
```

Elementos de Resposta

Se a ação for bem-sucedida, o serviço retornará uma resposta HTTP 200.

Os dados a seguir são retornados no formato JSON pelo serviço.

ApplicationDetail

Fornece uma descrição do aplicativo, como o nome do recurso da Amazon (ARN) do aplicativo, status, versão mais recente e detalhes da configuração de entrada e saída.

Tipo: objeto ApplicationDetail

Erros

ResourceNotFoundException

O aplicativo especificado não pode ser encontrado.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

DiscoverInputSchema

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Infere um esquema avaliando registros de amostra na origem de streaming especificada (stream do Amazon Kinesis ou stream de entrega do Amazon Kinesis Firehose) ou objeto do S3. Na resposta, a operação retorna o esquema inferido e também os registros de amostra que a operação usou para inferir o esquema.

Você pode usar o esquema inferido ao configurar uma origem de streaming para seu aplicativo. Para obter informações conceituais, consulte Configuração da entrada do aplicativo. Observe que quando você cria um aplicativo usando o console do Amazon Kinesis Analytics, o console usa essa operação para inferir um esquema e mostrá-lo na interface de usuário do console.

Essa operação exige permissões para executar a ação `kinesisanalytics:DiscoverInputSchema`.

Sintaxe da Solicitação

```
{
   "InputProcessingConfiguration": {
      "InputLambdaProcessor": {
         "ResourceARN": "string",
         "RoleARN": "string"
      }
   },
   "InputStartingPositionConfiguration": {
      "InputStartingPosition": "string"
   },
   "ResourceARN": "string",
   "RoleARN": "string",
   "S3Configuration": {
      "BucketARN": "string",
      "FileKey": "string",
```

```
        "RoleARN": "string"
    }
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

InputProcessingConfiguration

O InputProcessingConfigurationa ser usado para pré-processar os registros antes de descobrir o esquema dos registros.

Tipo: objeto InputProcessingConfiguration

Obrigatório: Não

InputStartingPositionConfiguration

Ponto em que você quer que o Amazon Kinesis Analytics comece a ler registros a partir das finalidades especificadas de descoberta da origem de streaming.

Tipo: objeto InputStartingPositionConfiguration

Obrigatório: Não

ResourceARN

O nome do recurso da Amazon (ARN) da origem do streaming.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARN

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para acessar o stream em seu nome.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

S3Configuration

Especifique esse parâmetro para descobrir um esquema de dados em um objeto do Amazon S3.

Tipo: objeto S3Configuration

Obrigatório: Não

Sintaxe da Resposta

```
{
   "InputSchema": {
      "RecordColumns": [
         {
            "Mapping": "string",
            "Name": "string",
            "SqlType": "string"
         }
      ],
      "RecordEncoding": "string",
      "RecordFormat": {
         "MappingParameters": {
            "CSVMappingParameters": {
               "RecordColumnDelimiter": "string",
               "RecordRowDelimiter": "string"
            },
            "JSONMappingParameters": {
               "RecordRowPath": "string"
            }
         },
         "RecordFormatType": "string"
      }
   },
   "ParsedInputRecords": [
      [ "string" ]
   ],
   "ProcessedInputRecords": [ "string" ],
```

```
    "RawInputRecords": [ "string" ]
}
```

Elementos de Resposta

Se a ação for bem-sucedida, o serviço retornará uma resposta HTTP 200.

Os dados a seguir são retornados no formato JSON pelo serviço.

InputSchema

Esquema inferido da origem do streaming. Identifica o formato dos dados na origem do streaming e como cada elemento de dados é mapeado para as colunas correspondentes no stream do aplicativo que você cria.

Tipo: objeto SourceSchema

ParsedInputRecords

Uma matriz de elementos, em que cada elemento corresponde a uma linha em um registro de fluxo (um registro de fluxo pode ter mais de uma linha).

Tipo: matriz de matrizes de strings

ProcessedInputRecords

Transmita dados que foram modificados pelo processador especificado no parâmetro `InputProcessingConfiguration`.

Tipo: matriz de strings

RawInputRecords

Dados brutos do stream que foram amostrados para inferir o esquema.

Tipo: matriz de strings

Erros

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceProvisionedThroughputExceededException

O Discovery não conseguiu obter um registro da fonte de streaming por causa do Amazon ProvisionedThroughputExceededException Kinesis Streams. Para obter mais informações, consulte a GetRecordsReferência da API do Amazon Kinesis Streams.

Código de Status HTTP: 400

ServiceUnavailableException

O serviço não está disponível. Volte e repita a operação.

Código de Status HTTP: 500

UnableToDetectSchemaException

O formato de dados é inválido. O Amazon Kinesis Analytics não consegue detectar o esquema para a origem do streaming específica.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python

- AWS SDK para Ruby V3

ListApplications

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Retorna uma lista dos aplicativos do Amazon Kinesis Analytics em sua conta. Para cada aplicativo, a resposta inclui o nome do aplicativo, nome do recurso da Amazon (ARN) e status. Se a resposta retornar o valor `HasMoreApplications` como verdadeiro, você poderá enviar outra solicitação adicionando o `ExclusiveStartApplicationName` ao corpo da solicitação e definindo o valor como o último nome do aplicativo da resposta anterior.

Se você quiser informações detalhadas sobre um aplicativo específico, use DescribeApplication.

Essa operação exige permissões para executar a ação `kinesisanalytics:ListApplications`.

Sintaxe da Solicitação

```
{
   "ExclusiveStartApplicationName": "string",
   "Limit": number
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ExclusiveStartApplicationName

Nome do aplicativo com o qual iniciar a lista. Ao usar a paginação para recuperar a lista, você não precisa especificar esse parâmetro na primeira solicitação. No entanto, nas solicitações subsequentes você adiciona o último nome do aplicativo da resposta anterior para obter a próxima página de aplicativos.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Obrigatório: não

Limit

Número máximo de aplicativos a serem listados.

Tipo: inteiro

Faixa válida: valor mínimo de 1. Valor máximo de 50.

Obrigatório: Não

Sintaxe da Resposta

```
{
   "ApplicationSummaries": [
      {
         "ApplicationARN": "string",
         "ApplicationName": "string",
         "ApplicationStatus": "string"
      }
   ],
   "HasMoreApplications": boolean
}
```

Elementos de Resposta

Se a ação for bem-sucedida, o serviço retornará uma resposta HTTP 200.

Os dados a seguir são retornados no formato JSON pelo serviço.

ApplicationSummaries

Lista de objetos `ApplicationSummary`.

Tipo: matriz de objetos ApplicationSummary

HasMoreApplications

Retorna verdadeiro se houver mais aplicativos para recuperar.

Tipo: booliano

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

ListTagsForResource

Recupera a lista de tags de chave-valor atribuídas à aplicação. Para obter mais informações, consulte Uso de tags.

Sintaxe da Solicitação

```
{
   "ResourceARN": "string"
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ResourceARN

O ARN da aplicação para a qual se deseja recuperar tags.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

Sintaxe da Resposta

```
{
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Elementos de Resposta

Se a ação for bem-sucedida, o serviço retornará uma resposta HTTP 200.

Os dados a seguir são retornados no formato JSON pelo serviço.

Tags

As tags de chave-valor atribuídas à aplicação.

Tipo: Matriz de objetos Tag

Membros da Matriz: Número mínimo de 1 item. Número máximo de 200 itens.

Erros

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2

- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

StartApplication

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Inicia o aplicativo Amazon Kinesis Analytics especificado. Depois de criar um aplicativo, você deve designar exclusivamente essa operação para iniciar seu aplicativo.

Depois que o aplicativo é iniciado, ele começa a consumir os dados de entrada, os processa e grava a saída no destino configurado.

O status da inscrição deve ser `READY` para você iniciar uma inscrição. Você pode obter o status do aplicativo no console ou usando a DescribeApplicationoperação.

Depois de iniciar o aplicativo, você pode impedir que o aplicativo processe a entrada chamando a StopApplicationoperação.

Essa operação exige permissões para executar a ação `kinesisanalytics:StartApplication`.

Sintaxe da Solicitação

```
{
   "ApplicationName": "string",
   "InputConfigurations": [
      {
         "Id": "string",
         "InputStartingPositionConfiguration": {
            "InputStartingPosition": "string"
         }
      }
   ]
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

Nome do aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

InputConfigurations

Identifica a entrada específica, por ID, que o aplicativo começa a consumir. O Amazon Kinesis Analytics começa a ler a origem de streaming associada à entrada. Também é possível especificar onde na origem de streaming você deseja que o Amazon Kinesis Analytics comece a ler.

Tipo: matriz de objetos InputConfiguration

Obrigatório: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

InvalidApplicationConfigurationException

A configuração do aplicativo fornecida pelo usuário não é válida.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

StopApplication

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Impede que o aplicativo processe os dados de entrada. Você pode interromper um aplicativo somente se ele estiver no estado de execução. Você pode usar a DescribeApplicationoperação para encontrar o estado do aplicativo. Depois que o aplicativo é interrompido, o Amazon Kinesis Analytics para de ler dados da entrada, o aplicativo para de processar dados e não há saída gravada no destino.

Essa operação exige permissões para executar a ação `kinesisanalytics:StopApplication`.

Sintaxe da Solicitação

```
{
   "ApplicationName": "string"
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

Nome do aplicativo em execução a ser interrompido.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

TagResource

Adiciona uma ou mais tags de valor-chave a um aplicativo Kinesis Analytics. Observe que o número máximo de tags do aplicativo inclui as tags do sistema. O número máximo de tags do aplicativo definidas pelo usuário é de 50. Para obter mais informações, consulte Uso de tags.

Sintaxe da Solicitação

```
{
   "ResourceARN": "string",
   "Tags": [
      {
         "Key": "string",
         "Value": "string"
      }
   ]
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ResourceARN

O ARN da aplicação ao qual atribuir as tags.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

Tags

As tags de chave-valor para atribuir à aplicação.

Tipo: Matriz de objetos Tag

Membros da Matriz: Número mínimo de 1 item. Número máximo de 200 itens.

Obrigatório: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

TooManyTagsException

Aplicação criada com muitas tags ou muitas tags adicionadas a uma aplicação. Observe que o número máximo de tags do aplicativo inclui as tags do sistema. O número máximo de tags do aplicativo definidas pelo usuário é de 50.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface

- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

UntagResource

Remove uma ou mais tags de um aplicativo Kinesis Analytics. Para obter mais informações, consulte Uso de tags.

Sintaxe da Solicitação

```
{
   "ResourceARN": "string",
   "TagKeys": [ "string" ]
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ResourceARN

O ARN do aplicativo Kinesis Analytics do qual remover as tags.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

TagKeys

Uma lista de chaves de tags a serem removidas da aplicação especificada.

Tipo: Matriz de strings

Membros da Matriz: Número mínimo de 1 item. Número máximo de 200 itens.

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Obrigatório: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

TooManyTagsException

Aplicação criada com muitas tags ou muitas tags adicionadas a uma aplicação. Observe que o número máximo de tags do aplicativo inclui as tags do sistema. O número máximo de tags do aplicativo definidas pelo usuário é de 50.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++

- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

UpdateApplication

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Atualiza um aplicativo Amazon Kinesis Analytics existente. Usando essa API, você pode atualizar o código, a configuração de entrada e a configuração de saída do aplicativo.

Observe que o Amazon Kinesis Analytics atualiza o `CurrentApplicationVersionId` cada vez que você atualiza seu aplicativo.

Essa operação exige permissão para a ação `kinesisanalytics:UpdateApplication`.

Sintaxe da Solicitação

```
{
   "ApplicationName": "string",
   "ApplicationUpdate": {
      "ApplicationCodeUpdate": "string",
      "CloudWatchLoggingOptionUpdates": [
         {
            "CloudWatchLoggingOptionId": "string",
            "LogStreamARNUpdate": "string",
            "RoleARNUpdate": "string"
         }
      ],
      "InputUpdates": [
         {
            "InputId": "string",
            "InputParallelismUpdate": {
               "CountUpdate": number
            },
            "InputProcessingConfigurationUpdate": {
               "InputLambdaProcessorUpdate": {
                  "ResourceARNUpdate": "string",
                  "RoleARNUpdate": "string"
               }
```

```
            },
            "InputSchemaUpdate": {
               "RecordColumnUpdates": [
                  {
                     "Mapping": "string",
                     "Name": "string",
                     "SqlType": "string"
                  }
               ],
               "RecordEncodingUpdate": "string",
               "RecordFormatUpdate": {
                  "MappingParameters": {
                     "CSVMappingParameters": {
                        "RecordColumnDelimiter": "string",
                        "RecordRowDelimiter": "string"
                     },
                     "JSONMappingParameters": {
                        "RecordRowPath": "string"
                     }
                  },
                  "RecordFormatType": "string"
               }
            },
            "KinesisFirehoseInputUpdate": {
               "ResourceARNUpdate": "string",
               "RoleARNUpdate": "string"
            },
            "KinesisStreamsInputUpdate": {
               "ResourceARNUpdate": "string",
               "RoleARNUpdate": "string"
            },
            "NamePrefixUpdate": "string"
         }
      ],
      "OutputUpdates": [
         {
            "DestinationSchemaUpdate": {
               "RecordFormatType": "string"
            },
            "KinesisFirehoseOutputUpdate": {
               "ResourceARNUpdate": "string",
               "RoleARNUpdate": "string"
            },
            "KinesisStreamsOutputUpdate": {
```

```
            "ResourceARNUpdate": "string",
            "RoleARNUpdate": "string"
         },
         "LambdaOutputUpdate": {
            "ResourceARNUpdate": "string",
            "RoleARNUpdate": "string"
         },
         "NameUpdate": "string",
         "OutputId": "string"
      }
   ],
   "ReferenceDataSourceUpdates": [
      {
         "ReferenceId": "string",
         "ReferenceSchemaUpdate": {
            "RecordColumns": [
               {
                  "Mapping": "string",
                  "Name": "string",
                  "SqlType": "string"
               }
            ],
            "RecordEncoding": "string",
            "RecordFormat": {
               "MappingParameters": {
                  "CSVMappingParameters": {
                     "RecordColumnDelimiter": "string",
                     "RecordRowDelimiter": "string"
                  },
                  "JSONMappingParameters": {
                     "RecordRowPath": "string"
                  }
               },
               "RecordFormatType": "string"
            }
         },
         "S3ReferenceDataSourceUpdate": {
            "BucketARNUpdate": "string",
            "FileKeyUpdate": "string",
            "ReferenceRoleARNUpdate": "string"
         },
         "TableNameUpdate": "string"
      }
   ]
```

```
    },
    "CurrentApplicationVersionId": number
}
```

Parâmetros da solicitação

A solicitação aceita os dados a seguir no formato JSON.

ApplicationName

Nome do aplicativo Amazon Kinesis Analytics a ser atualizado.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

ApplicationUpdate

Descreve as atualizações do aplicativo.

Tipo: objeto ApplicationUpdate

Obrigatório: Sim

CurrentApplicationVersionId

ID da versão de aplicativo atual. Você pode usar a DescribeApplicationoperação para obter esse valor.

Tipo: longo

Faixa válida: valor mínimo de 1. Valor máximo de 999999999.

Obrigatório: Sim

Elementos de Resposta

Se a ação tiver êxito, o serviço enviará de volta uma resposta HTTP 200 com um corpo HTTP vazio.

Erros

CodeValidationException

O código do aplicativo fornecido pelo usuário (consulta) é inválido. Pode ser um simples erro de sintaxe.

Código de Status HTTP: 400

ConcurrentModificationException

Exceção lançada como resultado da modificação simultânea em um aplicativo. Por exemplo, duas pessoas tentando editar o mesmo aplicativo ao mesmo tempo.

Código de Status HTTP: 400

InvalidArgumentException

O valor do parâmetro de entrada especificado é inválido.

Código de Status HTTP: 400

ResourceInUseException

O aplicativo não está disponível para esta operação.

Código de Status HTTP: 400

ResourceNotFoundException

A aplicação especificada não pode ser encontrada.

Código de Status HTTP: 400

UnsupportedOperationException

A solicitação foi rejeitada porque um parâmetro especificado não é compatível ou um recurso especificado não é válido para esta operação.

Código de Status HTTP: 400

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK para Go v2
- AWS SDK para Java V2
- AWS SDK para JavaScript V3
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

Tipos de dados

Os seguintes tipos de dados são compatíveis:

- ApplicationDetail
- ApplicationSummary
- ApplicationUpdate
- CloudWatchLoggingOption
- CloudWatchLoggingOptionDescription
- CloudWatchLoggingOptionUpdate
- CSVMappingParameters
- DestinationSchema
- Input
- InputConfiguration
- InputDescription
- InputLambdaProcessor
- InputLambdaProcessorDescription
- InputLambdaProcessorUpdate
- InputParallelism
- InputParallelismUpdate
- InputProcessingConfiguration

- InputProcessingConfigurationDescription
- InputProcessingConfigurationUpdate
- InputSchemaUpdate
- InputStartingPositionConfiguration
- InputUpdate
- JSONMappingParameters
- KinesisFirehoseInput
- KinesisFirehoseInputDescription
- KinesisFirehoseInputUpdate
- KinesisFirehoseOutput
- KinesisFirehoseOutputDescription
- KinesisFirehoseOutputUpdate
- KinesisStreamsInput
- KinesisStreamsInputDescription
- KinesisStreamsInputUpdate
- KinesisStreamsOutput
- KinesisStreamsOutputDescription
- KinesisStreamsOutputUpdate
- LambdaOutput
- LambdaOutputDescription
- LambdaOutputUpdate
- MappingParameters
- Output
- OutputDescription
- OutputUpdate
- RecordColumn
- RecordFormat
- ReferenceDataSource
- ReferenceDataSourceDescription
- ReferenceDataSourceUpdate

- S3Configuration
- S3ReferenceDataSource
- S3ReferenceDataSourceDescription
- S3ReferenceDataSourceUpdate
- SourceSchema
- Tag

ApplicationDetail

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Fornece uma descrição do aplicativo, incluindo o nome do recurso da Amazon (ARN) do aplicativo, o status, a versão mais recente e a configuração de entrada e saída.

Conteúdo

ApplicationARN

ARN do aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

ApplicationName

Nome do aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

ApplicationStatus

Status do aplicativo.

Tipo: sequências

Valores Válidos: `DELETING | STARTING | STOPPING | READY | RUNNING | UPDATING | AUTOSCALING`

Obrigatório: Sim

ApplicationVersionId

Fornece a versão da aplicação atual.

Tipo: longo

Faixa válida: valor mínimo de 1. Valor máximo de 999999999.

Obrigatório: Sim

ApplicationCode

Retorna o código do aplicativo que você forneceu para realizar a análise de dados em qualquer um dos fluxos no aplicativo no seu aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 0. Tamanho máximo de 102.400.

Obrigatório: não

ApplicationDescription

Descrição do aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 0. Tamanho máximo de 1.024.

Obrigatório: não

CloudWatchLoggingOptionDescriptions

Descreve os fluxos de CloudWatch log configurados para receber mensagens do aplicativo. Para obter mais informações sobre o uso de streams de CloudWatch log com aplicativos Amazon Kinesis Analytics, consulte Como trabalhar com o Amazon Logs. CloudWatch

Tipo: matriz de objetos CloudWatchLoggingOptionDescription

Obrigatório: não

CreateTimestamp

O timestamp em que a versão do aplicativo foi criada.

Tipo: carimbo de data/hora

Obrigatório: não

InputDescriptions

Descreve a configuração de entrada do aplicativo. Para obter mais informações, consulte Configuração de entrada do aplicativo.

Tipo: matriz de objetos InputDescription

Obrigatório: não

LastUpdateTimestamp

Time stamp da última atualização do aplicativo.

Tipo: carimbo de data/hora

Obrigatório: não

OutputDescriptions

Descreve a configuração de saída do aplicativo. Para obter mais informações, consulte Configuração da saída do aplicativo.

Tipo: matriz de objetos OutputDescription

Obrigatório: não

ReferenceDataSourceDescriptions

Descreve as fontes de dados de referência configuradas para o aplicativo. Para obter mais informações, consulte Configuração de entrada do aplicativo.

Tipo: matriz de objetos ReferenceDataSourceDescription

Obrigatório: não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

ApplicationSummary

> **Note**
>
> Esta documentação é para a versão 1 da API do Amazon Kinesis Data Analytics, que oferece suporte somente a aplicativos SQL. A versão 2 da API oferece suporte a aplicativos Java e SQL. Para obter mais informações sobre a versão 2, consulte Documentação da API V2 do Amazon Kinesis Data Analytics.

Fornece informações resumidas do aplicativo, incluindo nome do recurso da Amazon (ARN), nome e status do aplicativo.

Conteúdo

ApplicationARN

ARN do aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

ApplicationName

Nome do aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

ApplicationStatus

Status do aplicativo.

Tipo: sequências

Valores Válidos: `DELETING | STARTING | STOPPING | READY | RUNNING | UPDATING | AUTOSCALING`

Exigido: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

ApplicationUpdate

Descreve as atualizações a serem aplicadas a um aplicativo Amazon Kinesis Analytics existente.

Conteúdo

ApplicationCodeUpdate

Descreve as atualizações do código do aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 0. Tamanho máximo de 102.400.

Obrigatório: não

CloudWatchLoggingOptionUpdates

Descreve as atualizações das opções de CloudWatch registro do aplicativo.

Tipo: matriz de objetos CloudWatchLoggingOptionUpdate

Obrigatório: não

InputUpdates

Descreve as atualizações de configuração de entrada do aplicativo.

Tipo: matriz de objetos InputUpdate

Obrigatório: não

OutputUpdates

Descreve as atualizações de configuração de saída do aplicativo.

Tipo: matriz de objetos OutputUpdate

Obrigatório: não

ReferenceDataSourceUpdates

Descreve as atualizações da fonte de dados de referência do aplicativo.

Tipo: matriz de objetos ReferenceDataSourceUpdate

Obrigatório: não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

CloudWatchLoggingOption

Fornece uma descrição das opções de CloudWatch registro, incluindo o Amazon Resource Name (ARN) do stream de logs e o ARN da função.

Conteúdo

LogStreamARN

ARN do CloudWatch log para receber mensagens do aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

RoleARN

ARN do IAM do perfil a ser usado para enviar mensagens do aplicativo. Observação: para gravar mensagens do aplicativo CloudWatch, a função do IAM usada deve ter a ação `PutLogEvents` de política ativada.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

CloudWatchLoggingOptionDescription

Descrição da opção de CloudWatch registro.

Conteúdo

LogStreamARN

ARN do CloudWatch log para receber mensagens do aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

RoleARN

ARN do IAM do perfil a ser usado para enviar mensagens do aplicativo. Observação: para gravar mensagens do aplicativo CloudWatch, a função do IAM usada deve ter a ação `PutLogEvents` de política ativada.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

CloudWatchLoggingOptionId

ID da descrição da opção de CloudWatch registro.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

CloudWatchLoggingOptionUpdate

Descreve as atualizações das opções de CloudWatch registro.

Conteúdo

CloudWatchLoggingOptionId

ID da opção de CloudWatch registro a ser atualizada

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

LogStreamARNUpdate

ARN do CloudWatch log para receber mensagens do aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARNUpdate

ARN do IAM do perfil a ser usado para enviar mensagens do aplicativo. Observação: para gravar mensagens do aplicativo CloudWatch, a função do IAM usada deve ter a ação `PutLogEvents` de política ativada.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

CSVMappingParameters

Fornece informações adicionais de mapeamento quando o formato do registro usa delimitadores, como CSV. Por exemplo, os seguintes registros de amostra usam o formato CSV, em que os registros usam o "\n" como o delimitador de linha e uma vírgula (",") como o delimitador de coluna:

```
"name1", "address1"

"name2", "address2"
```

Conteúdo

RecordColumnDelimiter

Delimitador de coluna. Por exemplo, em um formato CSV, uma vírgula (",") é o delimitador típico de coluna.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1.

Obrigatório: Sim

RecordRowDelimiter

Row delimitador. Por exemplo, em um formato CSV, "\n" é o delimitador típico de linha.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1.

Obrigatório: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

DestinationSchema

Descreve o formato de dados quando os registros são gravados no destino. Para obter mais informações, consulte Configuração da saída do aplicativo.

Conteúdo

RecordFormatType

Especifica o formato dos registros no fluxo de saída.

Tipo: sequências

Valores Válidos: `JSON | CSV`

Exigido: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

Input

Ao configurar a entrada do aplicativo, você especifica a fonte de transmissão, o nome do fluxo no aplicativo que foi criado e o mapeamento entre os dois. Para obter mais informações, consulte Configuração de entrada do aplicativo.

Conteúdo

InputSchema

Descreve o formato dos dados na origem do streaming e como cada elemento de dados é mapeado para as colunas correspondentes no stream do aplicativo que está sendo criado.

Também é usado para descrever o formato da fonte de dados de referência.

Tipo: objeto SourceSchema

Obrigatório: Sim

NamePrefix

Prefixo de nome a ser usado na criação de um fluxo no aplicativo. Suponha que você especifique um prefixo ""MyInApplicationStream. Em seguida, o Amazon Kinesis Analytics cria um ou mais (de acordo com `InputParallelism` a contagem especificada) streams no aplicativo com os nomes MyInApplicationStream "_001," "MyInApplicationStream_002" e assim por diante.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Comprimento máximo de 32.

Obrigatório: Sim

InputParallelism

Descreve o número de fluxos do aplicativo a serem criados.

Os dados de sua origem são roteados para esses fluxos de entrada no aplicativo.

(consulte Configuração de entrada do aplicativo.

Tipo: objeto InputParallelism

Obrigatório: Não

InputProcessingConfiguration

O InputProcessingConfiguration para a entrada. Um processador de entrada transforma registros à medida que são recebidos do fluxo, antes do código SQL do aplicativo ser executado. Atualmente, a única configuração de processamento de entrada disponível é InputLambdaProcessor.

Tipo: objeto InputProcessingConfiguration

Obrigatório: Não

KinesisFirehoseInput

Se a origem do streaming for um fluxo de entrega do Amazon Kinesis Firehose, identifica o ARN do fluxo de entrega e uma função do IAM que permite que o Amazon Kinesis Analytics acesse o fluxo em seu nome.

Nota: `KinesisStreamsInput` ou `KinesisFirehoseInput` é necessário.

Tipo: objeto KinesisFirehoseInput

Obrigatório: Não

KinesisStreamsInput

Se a origem do streaming for um fluxo do Amazon Kinesis, identifica o nome de recurso da Amazon (ARN) do fluxo e uma função do IAM que permite que o Amazon Kinesis Analytics acesse o fluxo em seu nome.

Nota: `KinesisStreamsInput` ou `KinesisFirehoseInput` é necessário.

Tipo: objeto KinesisStreamsInput

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2

- AWS SDK para Ruby V3

InputConfiguration

Ao iniciar seu aplicativo, você fornece essa configuração que identifica a fonte de entrada e o ponto na fonte de entrada no qual você deseja que o aplicativo comece a processar registros.

Conteúdo

Id

ID da fonte de entrada Você pode obter esse ID chamando a DescribeApplicationoperação.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

InputStartingPositionConfiguration

Ponto em que você deseja que o aplicativo comece a processar registros da origem de streaming.

Tipo: objeto InputStartingPositionConfiguration

Obrigatório: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputDescription

Descreve a configuração de entrada do aplicativo. Para obter mais informações, consulte Configuração de entrada do aplicativo.

Conteúdo

InAppStreamNames

Retorna os nomes dos fluxos na aplicação que são mapeados para a origem do fluxo.

Tipo: matriz de strings

Restrições de tamanho: tamanho mínimo 1. Comprimento máximo de 32.

Obrigatório: não

InputId

ID de entrada associada à entrada do aplicativo. Essa é a ID que o Amazon Kinesis Analytics atribui a cada configuração de entrada que você adiciona ao aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Obrigatório: não

InputParallelism

Descreve o paralelismo configurado (número de fluxos da aplicação mapeados para a origem do streaming).

Tipo: objeto InputParallelism

Obrigatório: Não

InputProcessingConfigurationDescription

A descrição do pré-processador que é executado nos registros dessa entrada antes que o código a aplicação seja executado.

Tipo: objeto InputProcessingConfigurationDescription

Obrigatório: Não

InputSchema

Descreve o formato dos dados na origem do streaming e como cada elemento de dados é mapeado para as colunas correspondentes no stream do aplicativo que está sendo criado.

Tipo: objeto SourceSchema

Obrigatório: Não

InputStartingPositionConfiguration

Ponto em que o aplicativo está configurado para ler a partir do stream de entrada.

Tipo: objeto InputStartingPositionConfiguration

Obrigatório: Não

KinesisFirehoseInputDescription

Se um stream de entrega do Amazon Kinesis Firehose for configurado como fonte de streaming, fornece o ARN do stream de entrega e um perfil do IAM que permite que o Amazon Kinesis Analytics acesse o stream em seu nome.

Tipo: objeto KinesisFirehoseInputDescription

Obrigatório: Não

KinesisStreamsInputDescription

Se um stream do Amazon Kinesis for configurado como fonte do streaming, fornece o nome de recurso da Amazon (ARN) do stream e um perfil do IAM que permite que o Amazon Kinesis Analytics acesse o stream em seu nome.

Tipo: objeto KinesisStreamsInputDescription

Obrigatório: Não

NamePrefix

Prefixo do nome no aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Comprimento máximo de 32.

Obrigatório: não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputLambdaProcessor

Um objeto que contém o Amazon Resource Name (ARN) da função AWS Lambda que é usada para pré-processar registros no stream e o ARN da função do IAM que é usada para acessar a função Lambda. AWS

Conteúdo

ResourceARN

O ARN da função do AWS Lambda que opera em registros no fluxo.

> **Note**
>
> Para especificar uma versão anterior da função Lambda do que a mais recente, inclua a versão da função Lambda no ARN da função Lambda. Para obter mais informações sobre ARNs do Lambda, consulte Exemplos de ARNs: Lambda AWS

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

RoleARN

O ARN da função do IAM que é usada para acessar a função Lambda AWS .

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputLambdaProcessorDescription

Um objeto que contém o Amazon Resource Name (ARN) da função AWS Lambda que é usada para pré-processar registros no stream e o ARN da função do IAM que é usada para acessar a expressão Lambda. AWS

Conteúdo

ResourceARN

O ARN da função do Lambda AWS que é usada para pré-processar os registros no fluxo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARN

O ARN da função do IAM que é usada para acessar a função Lambda AWS .

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputLambdaProcessorUpdate

Representa uma atualização do InputLambdaProcessorque é usada para pré-processar os registros no fluxo.

Conteúdo

ResourceARNUpdate

O nome do recurso da Amazon (ARN) da nova função do Lambda AWS usada para pré-processar os registros no stream.

> **Note**
>
> Para especificar uma versão anterior da função Lambda do que a mais recente, inclua a versão da função Lambda no ARN da função Lambda. Para obter mais informações sobre ARNs do Lambda, consulte Exemplos de ARNs: Lambda AWS

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARNUpdate

O ARN da nova função do IAM que é usada para acessar a função Lambda AWS .

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputParallelism

Descreve o número de fluxos no aplicativo a serem criados para uma determinada origem de streaming. Para obter informações sobre paralelismo, consulte Configuração de entrada do aplicativo.

Conteúdo

Count

Número de fluxos do aplicativo a serem criados. Para obter mais informações, consulte Limites.

Tipo: inteiro

Intervalo válido: valor mínimo de 1. Valor máximo de 64.

Obrigatório: não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputParallelismUpdate

Fornece atualizações na contagem de paralelismo.

Conteúdo

CountUpdate

Número de streams no aplicativo a serem criados para a origem de streaming especificada.

Tipo: inteiro

Intervalo válido: valor mínimo de 1. Valor máximo de 64.

Obrigatório: não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputProcessingConfiguration

Fornece uma descrição de um processador que é usado para pré-processar os registros no fluxo antes de ser processado pelo código do aplicativo. Atualmente, o único processador de entrada disponível é o AWS Lambda.

Conteúdo

InputLambdaProcessor

O InputLambdaProcessorthat é usado para pré-processar os registros no stream antes de serem processados pelo código do aplicativo.

Tipo: objeto InputLambdaProcessor

Obrigatório: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputProcessingConfigurationDescription

Fornece informações de configuração sobre um processador de entrada. Atualmente, o único processador de entrada disponível é o AWS Lambda.

Conteúdo

InputLambdaProcessorDescription

Fornece informações de configuração sobre o associado InputLambdaProcessorDescription.

Tipo: objeto InputLambdaProcessorDescription

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputProcessingConfigurationUpdate

Descreve as atualizações de um InputProcessingConfiguration.

Conteúdo

InputLambdaProcessorUpdate

Fornece informações de atualização para um InputLambdaProcessor.

Tipo: objeto InputLambdaProcessorUpdate

Obrigatório: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputSchemaUpdate

Descreve as atualizações do esquema de entrada do aplicativo.

Conteúdo

RecordColumnUpdates

Uma lista dos objetos `RecordColumn`. Cada objeto descreve o mapeamento do elemento na origem do streaming para a coluna correspondente criada no fluxo da aplicação.

Tipo: Matriz de objetos RecordColumn

Membros da Matriz: Número mínimo de 1 item. Número máximo de 1000 itens.

Obrigatório: não

RecordEncodingUpdate

Especifica a codificação dos registros na origem do streaming. Por exemplo: UTF-8.

Tipo: string

Padrão: `UTF-8`

Obrigatório: não

RecordFormatUpdate

Especifica o formato dos registros na origem do streaming.

Tipo: objeto RecordFormat

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputStartingPositionConfiguration

Descreve o ponto em que a aplicação lê a partir da origem do streaming.

Conteúdo

InputStartingPosition

A posição inicial no stream.

- `NOW` - Comece a ler logo após o registro mais recente no stream, comece com a data e hora da solicitação emitida pelo cliente.
- `TRIM_HORIZON` - Comece a ler o último registro não cortado no stream, que é o registro mais antigo disponível no stream. Essa opção não está disponível para um stream de entrega do Amazon Kinesis Firehose.
- `LAST_STOPPED_POINT` - Continue a leitura de onde o aplicativo parou de ler pela última vez.

Tipo: sequências

Valores Válidos: `NOW | TRIM_HORIZON | LAST_STOPPED_POINT`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

InputUpdate

Descreve as atualizações em uma configuração de entrada específica (identificada pelo `InputId` de um aplicativo).

Conteúdo

InputId

ID de entrada da entrada do aplicativo a ser atualizado.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

InputParallelismUpdate

Descreve as atualizações de paralelismo (o número de streams no aplicativo que o Amazon Kinesis Analytics cria para a origem de streaming específica).

Tipo: objeto InputParallelismUpdate

Obrigatório: Não

InputProcessingConfigurationUpdate

Descreve as atualizações para uma configuração de processamento de entrada.

Tipo: objeto InputProcessingConfigurationUpdate

Obrigatório: Não

InputSchemaUpdate

Descreve o formato dos dados na origem do streaming e como cada elemento de dados é mapeado para as colunas correspondentes no fluxo da aplicação que está sendo criado.

Tipo: objeto InputSchemaUpdate

Obrigatório: Não

KinesisFirehoseInputUpdate

Se um stream de entrega do Amazon Kinesis Firehose for a origem de streaming a ser atualizada, fornece um ARN de stream atualizado e um ARN de perfil do IAM atualizados.

Tipo: objeto KinesisFirehoseInputUpdate

Obrigatório: Não

KinesisStreamsInputUpdate

Se um stream do Amazon Kinesis for a origem de streaming a ser atualizada, fornece um nome do recurso da Amazon (ARN) do stream atualizado e um ARN do perfil do IAM atualizados.

Tipo: objeto KinesisStreamsInputUpdate

Obrigatório: Não

NamePrefixUpdate

Prefixo de nome para streams no aplicativo que o Amazon Kinesis Analytics cria para a origem de streaming específica.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Comprimento máximo de 32.

Obrigatório: não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

JSONMappingParameters

Fornece informações adicionais de mapeamento quando JSON é o formato do registro na origem do streaming.

Conteúdo

RecordRowPath

Caminho para o pai de nível superior que contém os registros.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1.

Obrigatório: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisFirehoseInput

Identifica um fluxo de entrega do Amazon Kinesis Firehose como a origem de streaming. Você fornece o nome de recurso da Amazon (ARN) do fluxo de entrega e um ARN da função do IAM que permite que o Amazon Kinesis Analytics acesse o fluxo em seu nome.

Conteúdo

ResourceARN

ARN do fluxo de entrega de entrada.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

RoleARN

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para acessar o stream em seu nome. Você precisa se certificar de que a função tenha as permissões necessárias para acessar o fluxo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2

- AWS SDK para Ruby V3

KinesisFirehoseInputDescription

Descreve o stream de entrega do Amazon Kinesis Firehose que está configurado como a origem do streaming na configuração de entrada do aplicativo.

Conteúdo

ResourceARN

O nome do recurso da Amazon (ARN) do fluxo de entrega do Amazon Kinesis Firehose.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARN

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para acessar o stream.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisFirehoseInputUpdate

Ao atualizar a configuração de entrada do aplicativo, fornece informações sobre um stream de entrega do Amazon Kinesis Firehose como a origem de streaming.

Conteúdo

ResourceARNUpdate

O nome do recurso da Amazon (ARN) do stream de entrega de entrada do Amazon Kinesis Firehose.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARNUpdate

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para acessar o stream em seu nome. Você precisa conceder as permissões necessárias para essa função.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisFirehoseOutput

Ao configurar a saída do aplicativo, identifica um fluxo de entrega do Amazon Kinesis Firehose como o destino. Você fornece o nome de recurso da Amazon (ARN) do fluxo, bem como uma função do IAM que habilita o Amazon Kinesis Analytics para gravar no fluxo em seu nome.

Conteúdo

ResourceARN

O ARN do fluxo de entrega do Amazon Kinesis Firehose de destino onde a gravação será feita.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

RoleARN

O ARN da função do IAM que o Amazon Kinesis Analytics pode presumir para gravar o fluxo de destino em seu nome. Você precisa conceder as permissões necessárias para essa função.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisFirehoseOutputDescription

Para uma saída do aplicativo, descreve um fluxo de entrega do Amazon Kinesis Firehose configurado como o destino.

Conteúdo

ResourceARN

O nome do recurso da Amazon (ARN) do fluxo de entrega do Amazon Kinesis Firehose.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARN

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para acessar o stream.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisFirehoseOutputUpdate

Ao atualizar uma configuração de saída usando a UpdateApplicationoperação, fornece informações sobre um stream de entrega do Amazon Kinesis Firehose configurado como destino.

Conteúdo

ResourceARNUpdate

O nome do recurso da Amazon (ARN) do stream de entrega do Amazon Kinesis Firehose para gravar.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARNUpdate

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para acessar o stream em seu nome. Você precisa conceder as permissões necessárias para essa função.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisStreamsInput

Identifica um fluxo do Amazon Kinesis como a fonte de streaming. Você fornece o nome de recurso da Amazon (ARN) do fluxo, bem como um ARN de função do IAM que habilita o Amazon Kinesis Analytics para acessar o fluxo em seu nome.

Conteúdo

ResourceARN

ARN do fluxo do Amazon Kinesis de entrada para leitura.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

RoleARN

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para acessar o stream em seu nome. Você precisa conceder as permissões necessárias para essa função.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisStreamsInputDescription

Descreve o stream do Amazon Kinesis que está configurado como a origem do streaming na configuração de entrada do aplicativo.

Conteúdo

ResourceARN

O nome do recurso da Amazon (ARN) do stream de dados do Amazon Kinesis.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARN

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para acessar o stream.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisStreamsInputUpdate

Ao atualizar a configuração de entrada do aplicativo, fornece informações sobre um stream do Amazon Kinesis como a origem de streaming.

Conteúdo

ResourceARNUpdate

O nome do recurso da Amazon (ARN) do stream do Amazon Kinesis de entrada para leitura.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARNUpdate

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para acessar o stream em seu nome. Você precisa conceder as permissões necessárias para essa função.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisStreamsOutput

Ao configurar a saída do aplicativo, identifica um fluxo do Amazon Kinesis como o destino. Você fornece o nome de recurso da Amazon (ARN) do fluxo e um ARN de função do IAM que o Amazon Kinesis Analytics pode usar para gravar no fluxo em seu nome.

Conteúdo

ResourceARN

O ARN do stream do Amazon Kinesis de destino onde a gravação será feita.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

RoleARN

O ARN da função do IAM que o Amazon Kinesis Analytics pode presumir para gravar o fluxo de destino em seu nome. Você precisa conceder as permissões necessárias para essa função.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisStreamsOutputDescription

Para uma saída do aplicativo, descreve um stream de entrega do Amazon Kinesis configurado como o destino.

Conteúdo

ResourceARN

O nome do recurso da Amazon (ARN) do stream de dados do Amazon Kinesis.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARN

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para acessar o stream.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

KinesisStreamsOutputUpdate

Ao atualizar uma configuração de saída usando a UpdateApplicationoperação, fornece informações sobre um stream do Amazon Kinesis configurado como destino.

Conteúdo

ResourceARNUpdate

O nome do recurso da Amazon (ARN) do stream do Amazon Kinesis onde o resultado será gravado.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARNUpdate

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para acessar o stream em seu nome. Você precisa conceder as permissões necessárias para essa função.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

LambdaOutput

Ao configurar a saída do aplicativo, identifica uma função AWS Lambda como destino. Você fornece o nome de recurso da Amazon (ARN) da função e um ARN de função do IAM que o Amazon Kinesis Analytics pode usar para gravar na função em seu nome.

Conteúdo

ResourceARN

Nome de recurso da Amazon (ARN) da função Lambda de destino onde a gravação será feita.

> **Note**
>
> Para especificar uma versão anterior da função Lambda do que a mais recente, inclua a versão da função Lambda no ARN da função Lambda. Para obter mais informações sobre ARNs do Lambda, consulte Exemplos de ARNs: Lambda AWS

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

RoleARN

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para gravar na função de destino em seu nome. Você precisa conceder as permissões necessárias para essa função.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

LambdaOutputDescription

Para a saída de um aplicativo, descreve a função AWS Lambda configurada como seu destino.

Conteúdo

ResourceARN

Nome de recurso da Amazon (ARN) da função Lambda de destino.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARN

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para gravar no perfil de destino.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

LambdaOutputUpdate

Ao atualizar uma configuração de saída usando a UpdateApplicationoperação, fornece informações sobre uma função AWS Lambda configurada como destino.

Conteúdo

ResourceARNUpdate

Nome de recurso da Amazon (ARN) da função Lambda de destino.

> **Note**
>
> Para especificar uma versão anterior da função Lambda do que a mais recente, inclua a versão da função Lambda no ARN da função Lambda. Para obter mais informações sobre ARNs do Lambda, consulte Exemplos de ARNs: Lambda AWS

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

RoleARNUpdate

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para gravar na função de destino em seu nome. Você precisa conceder as permissões necessárias para essa função.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

MappingParameters

Na configuração da entrada do aplicativo, no momento da criação ou atualização de um aplicativo, fornece as informações de mapeamento adicionais específicas sobre o formato do registro (como JSON, CSV ou campos de registro delimitados por algum delimitador) na origem do streaming.

Conteúdo

CSVMappingParameters

Fornece informações adicionais de mapeamento quando o formato do registro usa delimitadores (por exemplo, CSV).

Tipo: objeto CSVMappingParameters

Obrigatório: Não

JSONMappingParameters

Fornece informações adicionais de mapeamento quando JSON é o formato do registro na origem do streaming.

Tipo: objeto JSONMappingParameters

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

Output

Descreve a configuração de saída do aplicativo na qual você identifica um fluxo no aplicativo e um destino no qual você deseja que os dados do fluxo no aplicativo sejam gravados. O destino pode ser um fluxo do Amazon Kinesis ou um stream de entrega do Amazon Kinesis Firehose.

Para limites sobre quantos destinos um aplicativo pode gravar e outras limitações, consulte Limites.

Conteúdo

DestinationSchema

Descreve o formato de dados quando os registros são gravados no destino. Para obter mais informações, consulte Configuração da saída do aplicativo.

Tipo: objeto DestinationSchema

Obrigatório: Sim

Name

Nome do fluxo no aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Comprimento máximo de 32.

Obrigatório: Sim

KinesisFirehoseOutput

Identifica um fluxo de entrega do Amazon Kinesis Firehose como o destino.

Tipo: objeto KinesisFirehoseOutput

Obrigatório: Não

KinesisStreamsOutput

Identifica um fluxo do Amazon Kinesis como o destino.

Tipo: objeto KinesisStreamsOutput

Obrigatório: Não

LambdaOutput

Identifica uma função AWS Lambda como destino.

Tipo: objeto LambdaOutput

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

OutputDescription

Descreve a configuração de saída do aplicativo, que inclui o nome do stream no aplicativo e o destino em que os dados do stream são gravados. O destino pode ser um fluxo do Amazon Kinesis ou um stream de entrega do Amazon Kinesis Firehose.

Conteúdo

DestinationSchema

Formato de dados usado para gravar dados no destino.

Tipo: objeto DestinationSchema

Obrigatório: Não

KinesisFirehoseOutputDescription

Descreve o stream de entrega do Amazon Kinesis Firehose configurado como o destino no qual a saída é gravada.

Tipo: objeto KinesisFirehoseOutputDescription

Obrigatório: Não

KinesisStreamsOutputDescription

Descreve o stream do Amazon Kinesis configurado como o destino em que a saída é gravada.

Tipo: objeto KinesisStreamsOutputDescription

Obrigatório: Não

LambdaOutputDescription

Descreve a função AWS Lambda configurada como o destino em que a saída é gravada.

Tipo: objeto LambdaOutputDescription

Obrigatório: Não

Name

O nome do stream no aplicativo configurado como saída.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Comprimento máximo de 32.

Obrigatório: não

OutputId

Um identificador exclusivo para a configuração de saída.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

OutputUpdate

Descreve as atualizações na configuração de saída identificada pelo `OutputId`.

Conteúdo

OutputId

Identifica a configuração de saída específica que você deseja atualizar.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

DestinationSchemaUpdate

Descreve o formato de dados quando os registros são gravados no destino. Para obter mais informações, consulte Configuração da saída do aplicativo.

Tipo: objeto DestinationSchema

Obrigatório: Não

KinesisFirehoseOutputUpdate

Descreve um stream de entrega do Amazon Kinesis Firehose como o destino da saída.

Tipo: objeto KinesisFirehoseOutputUpdate

Obrigatório: Não

KinesisStreamsOutputUpdate

Descreve um stream do Amazon Kinesis como o destino da saída.

Tipo: objeto KinesisStreamsOutputUpdate

Obrigatório: Não

LambdaOutputUpdate

Descreve uma função AWS Lambda como o destino da saída.

Tipo: objeto LambdaOutputUpdate

Obrigatório: Não

NameUpdate

Se você quiser especificar um fluxo na aplicação diferente para essa configuração de saída, use esse campo para especificar o novo nome do fluxo na aplicação.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Comprimento máximo de 32.

Obrigatório: não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

RecordColumn

Descreve o mapeamento de cada elemento na origem do streaming para a coluna correspondente criada no fluxo do aplicativo.

Também é usado para descrever o formato da fonte de dados de referência.

Conteúdo

Name

Nome da coluna criada no fluxo de entrada no aplicativo ou na tabela de referência.

Tipo: string

Obrigatório: Sim

SqlType

Tipo da coluna criada no fluxo de entrada no aplicativo ou na tabela de referência.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1.

Obrigatório: Sim

Mapping

Referência ao elemento de dados na entrada de streaming ou na fonte de dados de referência. Esse elemento é necessário se RecordFormatTypeforJSON.

Tipo: sequência

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2

- AWS SDK para Ruby V3

RecordFormat

Descreve o formato do registro e as informações de mapeamento relevantes que devem ser aplicadas para esquematizar os registros no fluxo.

Conteúdo

RecordFormatType

O tipo de formato do registro.

Tipo: sequências

Valores Válidos: `JSON | CSV`

Obrigatório: Sim

MappingParameters

Na configuração da entrada do aplicativo, no momento da criação ou atualização de um aplicativo, fornece as informações de mapeamento adicionais específicas sobre o formato do registro (como JSON, CSV ou campos de registro delimitados por algum delimitador) na origem do streaming.

Tipo: objeto MappingParameters

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

ReferenceDataSource

Descreve a fonte de dados de referência, fornecendo as informações de origem (nome do bucket do S3 e nome da chave do objeto), o nome resultante da tabela no aplicativo que foi criado e o esquema necessário para mapear os elementos de dados no objeto do Amazon S3 para a tabela no aplicativo.

Conteúdo

ReferenceSchema

Descreve o formato dos dados na origem do streaming e como cada elemento de dados é mapeado para as colunas correspondentes que são criadas no stream do aplicativo.

Tipo: objeto SourceSchema

Obrigatório: Sim

TableName

Nome da tabela no aplicativo a ser criada.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Comprimento máximo de 32.

Obrigatório: Sim

S3ReferenceDataSource

Identifica o bucket do S3 e o objeto que contém os dados de referência. Também identifica a função do IAM que o Amazon Kinesis Analytics pode assumir para ler esse objeto em seu nome. Um aplicativo do Amazon Kinesis Analytics carrega dados de referência apenas uma vez. Se os dados forem alterados, você chamará a operação `UpdateApplication` para acionar o recarregamento de dados em seu aplicativo.

Tipo: objeto S3ReferenceDataSource

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

ReferenceDataSourceDescription

Descreve a fonte de dados de referência configurada para um aplicativo.

Conteúdo

ReferenceId

ID da fonte de dados de referência. Essa é a ID que o Amazon Kinesis Analytics atribui quando você adiciona a fonte de dados de referência ao seu aplicativo usando AddApplicationReferenceDataSourcea operação.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

S3ReferenceDataSourceDescription

Fornece o nome do bucket do S3, o nome da chave do objeto que contém os dados de referência. Também fornece o nome do recurso da Amazon (ARN) do perfil do IAM que o Amazon Kinesis Analytics pode assumir para ler o objeto Amazon S3 e preencher a tabela de referência no aplicativo.

Tipo: objeto S3ReferenceDataSourceDescription

Obrigatório: Sim

TableName

O nome da tabela na aplicação criado pela configuração específica da fonte de dados de referência.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Comprimento máximo de 32.

Obrigatório: Sim

ReferenceSchema

Descreve o formato dos dados na origem do streaming e como cada elemento de dados é mapeado para as colunas correspondentes que são criadas no stream do aplicativo.

Tipo: objeto SourceSchema

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

ReferenceDataSourceUpdate

Quando você atualiza uma configuração de fonte de dados de referência para um aplicativo, esse objeto fornece todos os valores atualizados (como o nome do bucket de origem e nome da chave do objeto), o nome da tabela que é criada no aplicativo e as informações de mapeamento atualizadas que mapeiam os dados no objeto do Amazon S3 para a tabela de referência que é criada no aplicativo.

Conteúdo

ReferenceId

ID da fonte de dados de referência que está sendo atualizada. Você pode usar a DescribeApplicationoperação para obter esse valor.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 50.

Padrão: `[a-zA-Z0-9_.-]+`

Exigido: Sim

ReferenceSchemaUpdate

Descreve o formato dos dados na origem do streaming e como cada elemento de dados é mapeado para as colunas correspondentes que são criadas no stream do aplicativo.

Tipo: objeto SourceSchema

Obrigatório: Não

S3ReferenceDataSourceUpdate

Descreve o nome do bucket do S3, o nome da chave do objeto e o perfil do IAM que o Amazon Kinesis Analytics pode assumir para ler o objeto do Amazon S3 em seu nome e preencher a tabela de referência no aplicativo.

Tipo: objeto S3ReferenceDataSourceUpdate

Obrigatório: Não

TableNameUpdate

Nome da tabela que é criada no aplicativo por essa atualização.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Comprimento máximo de 32.

Obrigatório: não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

S3Configuration

Fornece uma descrição de uma fonte de dados do Amazon S3, incluindo o nome do recurso da Amazon (ARN) do bucket do S3, o ARN do perfil do IAM que é usado para acessar o bucket e o nome do objeto do Amazon S3 que contém os dados.

Conteúdo

BucketARN

O ARN do bucket do S3 que contém os dados.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

FileKey

O nome do objeto que contém os dados.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 1.024.

Obrigatório: Sim

RoleARN

ARN do IAM do perfil usado para acessar os dados.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

S3ReferenceDataSource

Identifica o bucket do S3 e o objeto que contém os dados de referência. Também identifica a função do IAM que o Amazon Kinesis Analytics pode assumir para ler esse objeto em seu nome.

Um aplicativo do Amazon Kinesis Analytics carrega dados de referência apenas uma vez. Se os dados mudarem, você chama a UpdateApplicationoperação para acionar o recarregamento de dados em seu aplicativo.

Conteúdo

BucketARN

Nome de recurso da Amazon (ARN) do bucket do S3.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

FileKey

Nome da chave do objeto que contém dados de referência.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 1.024.

Obrigatório: Sim

ReferenceRoleARN

O ARN da função do IAM que o serviço pode assumir para ler dados em seu nome. Essa função deve ter permissão para a ação `s3:GetObject` na política de objeto e confiança que permite que o serviço principal do Amazon Kinesis Analytics assuma essa função.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

S3ReferenceDataSourceDescription

Fornece o nome do bucket e o nome da chave de objeto que armazena os dados de referência.

Conteúdo

BucketARN

Nome de recurso da Amazon (ARN) do bucket do S3.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

FileKey

Nome da chave do objeto do Amazon S3.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 1.024.

Obrigatório: Sim

ReferenceRoleARN

ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para ler o objeto do Amazon S3 em seu nome para preencher a tabela de referência no aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Exigido: Sim

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

S3ReferenceDataSourceUpdate

Descreve o nome do bucket do S3, o nome da chave do objeto e o perfil do IAM que o Amazon Kinesis Analytics pode assumir para ler o objeto do Amazon S3 em seu nome e preencher a tabela de referência no aplicativo.

Conteúdo

BucketARNUpdate

Nome de recurso da Amazon (ARN) do bucket do S3.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: não

FileKeyUpdate

Nome de chave de objeto.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 1.024.

Obrigatório: não

ReferenceRoleARNUpdate

O ARN do perfil do IAM que o Amazon Kinesis Analytics pode assumir para ler o objeto do Amazon S3 e preencher o conteúdo no aplicativo.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 2.048.

Padrão: `arn:.*`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

SourceSchema

Descreve o formato dos dados na origem do streaming e como cada elemento de dados é mapeado para as colunas correspondentes que são criadas no stream do aplicativo.

Conteúdo

RecordColumns

Uma lista dos objetos `RecordColumn`.

Tipo: Matriz de objetos RecordColumn

Membros da Matriz: Número mínimo de 1 item. Número máximo de 1.000 itens.

Obrigatório: Sim

RecordFormat

Especifica o formato dos registros na origem do streaming.

Tipo: objeto RecordFormat

Obrigatório: Sim

RecordEncoding

Especifica a codificação dos registros na origem do streaming. Por exemplo: UTF-8.

Tipo: string

Padrão: `UTF-8`

Obrigatório: Não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

Tag

Um par de valores-chave (o valor é opcional) que você pode definir e atribuir aos AWS recursos. Se você especificar uma tag que já existe, o valor da tag será substituído pelo valor especificado na solicitação. Observe que o número máximo de tags do aplicativo inclui as tags do sistema. O número máximo de tags do aplicativo definidas pelo usuário é de 50. Para obter mais informações, consulte Uso de tags.

Conteúdo

Key

A chave da tag da chave-valor.

Tipo: sequência

Restrições de tamanho: tamanho mínimo 1. O tamanho máximo é 128.

Obrigatório: Sim

Value

O valor da tag de chave-valor. O valor é opcional.

Tipo: sequência

Restrições de tamanho: o tamanho mínimo é 0. O tamanho máximo é 256.

Obrigatório: não

Consulte também

Para obter mais informações sobre como usar essa API em um dos AWS SDKs específicos da linguagem, consulte o seguinte:

- AWS SDK for C++
- AWS SDK para Java V2
- AWS SDK para Ruby V3

Histórico do documento para Amazon Kinesis Data Analytics

A tabela a seguir descreve as alterações importantes na documentação desde a última versão do Kinesis Data Analytics.

- Versão da API: 14/08/2015
- Última atualização da documentação: 8 de maio de 2019

Alteração	Descrição	Data
Colocando tag nos aplicativos Kinesis Data Analytics	Use a marcação de aplicativos para determinar os custos por aplicativo, controlar o acesso ou para finalidades definidas pelo usuário. Para obter mais informações, consulte Uso de tags.	8 de maio de 2019
Registrar no log chamadas de API do Kinesis Data Analytics com AWS CloudTrail	O Amazon Kinesis Data Analytics se integra ao AWS CloudTrail, serviço que fornece um registro das ações realizadas por um usuário, um perfil ou um serviço da AWS no Kinesis Data Analytics. Para obter mais informaçõ es, consulte Usar o AWS CloudTrail.	22 de março de 2019
Kinesis Data Analytics disponível na região de Frankfurt	O Kinesis Analytics agora está disponível na região da Europa (Frankfurt). Para obter mais informações, consulte e endpoints: Kinesis Data Analytics.	18 de julho de 2018

Alteração	Descrição	Data
Usar dados de referência no console	Agora você pode trabalhar com dados de referência de aplicativo no console. Para obter mais informaçõ es, consulteExemplo: adição de dados de referência a um aplicativo do Kinesis Data Analytics	13 de julho de 2018
Exemplos de consultas em janela	Exemplos de uso para janelas e agregação. Para obter mais informações, consulteExemplos: janelas e agregação	9 de julho de 2018
Teste de aplicativos	Orientação sobre como testar alterações no esquema e código do aplicativo. Para obter mais informações, consulteTeste de aplicativos	3 de julho de 2018
Exemplo de aplicativos para pré-processamento de dados	Amostras de código adicionais para os operadores REGEX_LOG _PARSE, REGEX_REP LACE e DateTime. Para obter mais informações, consulteExemplos: transform ação de dados	18 de maio de 2018

Alteração	Descrição	Data
Aumento do tamanho de linhas retornadas e código SQL	O limite do tamanho de uma linha retornada aumenta para 512 KB, e o limite do tamanho do código SQL em um aplicativo aumenta para 100 KB. Para obter mais informações, consulte Limites.	2 de maio de 2018
Exemplos de função AWS Lambda em Java e .NET	Exemplos de código para criar funções do Lambda para pré-processamento de registros e destinos de aplicativo. Para obter mais informações, consulte Criar funções do Lambda para pré-processamento e Como criar funções do Lambda para destinos de aplicativos.	22 de março de 2018
Nova função HOTSPOTS	Localize e retorne informaçõ es sobre regiões relativamente densas nos dados. Para obter mais informações, consulte Exemplo: detectar hotspots em um streaming (função HOTSPOTS).	19 de março de 2018
Função do Lambda como destino	Envie os resultados da análise para uma função do Lambda como destino. Para obter mais informações, consulte Como usar a função do Lambda como saída.	20 de dezembro de 2017

Alteração	Descrição	Data
Nova função `RANDOM_CUT_FOREST_WITH_EXPLANATION`	Obtenha uma explicação sobre que campos contribuem para uma pontuação de anomalia em um fluxo de dados . Para obter mais informações, consulte Exemplo: como detectar anomalias de dados e obter uma explicação (função RANDOM_CUT_FOREST_WITH_EXPLANATION).	2 de novembro de 2017
Descoberta de esquema em dados estáticos	Execute a descoberta de esquema em dados estáticos armazenados em um bucket do Amazon S3. Para obter mais informações, consulte Usar o recurso de descoberta de esquema em dados estáticos.	6 de outubro de 2017
Recurso de pré-processamento do Lambda	Faça o pré-processamento de registros em um fluxo de entrada com o AWS Lambda antes da análise. Para obter mais informações, consulte Pré-processar dados usando uma função do Lambda.	6 de outubro de 2017

Alteração	Descrição	Data
Escalabilidade automática de aplicativos	Aumente automaticamente o throughput do seu aplicativo com a escalabilidade automática. Para obter mais informações, consulte Escalabilidade automática de aplicativos para aumentar a taxa de transferência.	13 de setembro de 2017
Vários fluxos de entrada no aplicativo	Aumente a taxa de transferência do aplicativo com vários streams no aplicativo. Para obter mais informações, consulte Paralelização dos streams de entrada para aumentar a taxa de transferência.	29 de junho de 2017
Guia de uso do AWS Management Console for Kinesis Data Analytics	Edite um esquema inferido e o código SQL usando o editor de esquema e o editor de SQL no console do Kinesis Data Analytics. Para obter mais informações, consulte Etapa 4 (opcional) Editar o esquema e código SQL usando o console.	7 de abril de 2017
Versão pública	Lançamento público do Guia do desenvolvedor do Amazon Kinesis Data Analytics.	11 de agosto de 2016
Versão de visualização	Pré-visualização do lançamento público do Guia do desenvolvedor do Amazon Kinesis Data Analytics.	29 de janeiro de 2016

Glossário do AWS

Para obter a terminologia mais recente da AWS, consulte o glossário da AWS na Referência do Glossário da AWS.