aws

Guía para desarrolladores

AWS Data Pipeline

Versión de API 2012-10-29

AWS Data Pipeline: Guía para desarrolladores

Table of Contents

AWS Data Pipeline ya no está disponible para nuevos clientes. Los clientes actuales de AWS Data Pipeline pueden seguir utilizando el servicio con normalidad. Más información

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.

¿Qué es () AWS Data Pipeline?

> **Note**
>
> El servicio de AWS Data Pipeline está en modo de mantenimiento y no se prevén nuevas funciones ni ampliaciones regionales. Para obtener más información y saber cómo migrar las cargas de trabajo existentes, consulte Migración de cargas de trabajo desde AWS Data Pipeline.

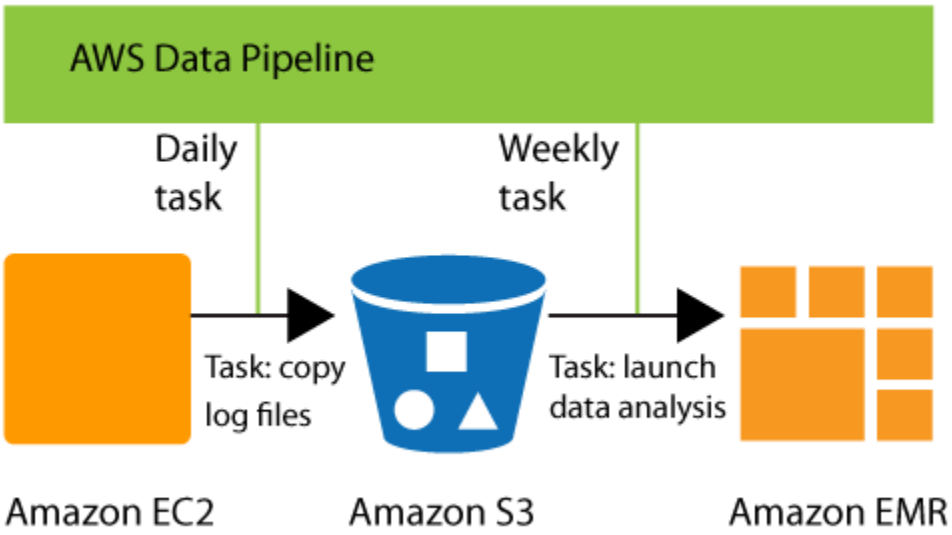
AWS Data Pipeline es un servicio web que puede utilizar para automatizar el movimiento y la transformación de los datos. Con AWS Data Pipeline, puede definir flujos de trabajo controlados por datos en los que las tareas dependan de que se hayan realizado correctamente las tareas anteriores. Debe definir los parámetros de las transformaciones de datos y AWS Data Pipeline aplicará la lógica que haya configurado.

Los siguientes componentes de AWS Data Pipeline se combinan para administrar los datos:

- Una definición de canalización especifica la lógica de negocio de la administración de datos. Para obtener más información, consulte Sintaxis de los archivos de definición de la canalización.
- Una canalización programa y ejecuta tareas mediante la creación de instancias Amazon EC2 que llevan a cabo las actividades de trabajo definidas. Solo tiene que cargar la definición de canalización en la canalización y, a continuación, activar la canalización. Puede editar la definición de la canalización de una canalización en ejecución y activar de nuevo la canalización para que surta efecto. Puede desactivar la canalización, modificar una fuente de datos y, a continuación, activar la canalización de nuevo. Cuando haya terminado con la canalización, puede eliminarla.
- Task Runner realiza un sondeo para comprobar si hay tareas y, a continuación, realiza esas tareas. Por ejemplo, Task Runner podría copiar archivos de registro en Amazon S3 y lanzar clústeres de Amazon EMR. Task Runner se instala y se ejecuta automáticamente en los recursos creados por las definiciones de la canalización. Puede escribir una aplicación de ejecución de tareas personalizada o puede usar la aplicación Task Runner que se incluye en AWS Data Pipeline. Para obtener más información, consulte Aplicaciones de ejecución de tareas.

Por ejemplo, puede utilizar AWS Data Pipeline para archivar los registros del servidor web en Amazon Simple Storage Service (Amazon S3) cada día y, a continuación, ejecutar un clúster de Amazon EMR (Amazon EMR) cada semana en esos registros para generar informes de tráfico. AWS

Data Pipeline programa las tareas diarias para copiar los datos y la tarea semanal para lanzar el clúster de Amazon EMR. AWS Data Pipeline también garantiza que Amazon EMR espera a que se carguen los datos al final del día en Amazon S3 antes de empezar su análisis, incluso si hay un retraso imprevisto en la carga de los registros.



Contenido

Migración de cargas de trabajo desde AWS Data Pipeline

AWS lanzó el servicio de AWS Data Pipeline en 2012. En ese momento, los clientes buscaban un servicio que les ayudara a transferir datos de forma fiable entre diferentes orígenes de datos mediante una variedad de opciones informáticas. Ahora hay otros servicios que ofrecen a los clientes una mejor experiencia de integración de datos. Por ejemplo, puede utilizar AWS Glue para ejecutar y orquestar las aplicaciones de Apache Spark, AWS Step Functions para ayudar a orquestar los componentes del servicio de AWS o Amazon Managed Workflows para Apache Airflow (Amazon MWAA) para ayudar a gestionar la orquestación del flujo de trabajo de Apache Airflow.

En este tema se explica cómo hacer la migración de AWS Data Pipeline a opciones alternativas. La opción que elija depende de su carga de trabajo actual en AWS Data Pipeline. Puede migrar los casos de uso típicos de AWS Data Pipeline a AWS Glue, AWS Step Functions o Amazon MWAA.

Migración de cargas de trabajo a AWS Glue

AWS Glue es un servicio de integración de datos sin servidor que facilita a los usuarios de análisis descubrir, preparar, migrar e integrar datos de varios orígenes. Incluye herramientas para la creación, la ejecución de trabajos y la orquestación de flujos de trabajo. Con AWS Glue, puede descubrir y conectarse a más de 70 orígenes de datos diversos y administrar sus datos en un catálogo de datos centralizado. Puede crear, ejecutar y supervisar visualmente canalizaciones de extracción, transformación y carga (ETL) para cargar datos en los lagos de datos. Además, puede buscar y consultar datos catalogados de forma inmediata mediante Amazon Athena, Amazon EMR y Amazon Redshift Spectrum.

Recomendamos migrar su carga de trabajo de AWS Data Pipeline a AWS Glue cuando:

- Está buscando un servicio de integración de datos sin servidor que admita diversos orígenes de datos, interfaces de creación que incluyan editores visuales y cuadernos, y funciones avanzadas de administración de datos, como la calidad de los datos y la detección de datos confidenciales.
- Su carga de trabajo se puede migrar a flujos de trabajo AWS Glue, trabajos (en Python o Apache Spark) y rastreadores (por ejemplo, la canalización actual se basa en Apache Spark).
- Necesita una plataforma única que pueda gestionar todos los aspectos de su canalización de datos, incluidos la ingesta, el procesamiento, la transferencia, las pruebas de integridad y los controles de calidad.
- Su canalización existente se creó a partir de una plantilla predefinida en la consola AWS Data Pipeline, como la exportación de una tabla de DynamoDB a Amazon S3, y busca la misma plantilla con el mismo propósito.
- Su carga de trabajo no depende de una aplicación específica del ecosistema de Hadoop, como Apache Hive.
- Su carga de trabajo no requiere la orquestación de servidores en las instalaciones.

Con AWS, paga una tarifa por hora que se factura por segundo, por los rastreadores (detección de datos) y los trabajos de ETL (procesamiento y carga de datos). AWS Glue Studio es un motor de orquestación integrado de recursos AWS Glue y se ofrece sin costo adicional. Para obtener más información sobre precios, consulte Precios de AWS Glue.

Migración de cargas de trabajo a AWS Step Functions

AWS Step Functions es un servicio de orquestación sin servidor que le permite crear flujos de trabajo para sus aplicaciones esenciales desde el punto de vista empresarial. Con Step Functions, utiliza un editor visual para crear flujos de trabajo e integrarlos directamente con más de 11 000 acciones para más de 250 servicios de AWS, como AWS Lambda, Amazon EMR, DynamoDB y más. Puede usar Step Functions para orquestar las canalizaciones de procesamiento de datos, gestionar los errores y trabajar con las limitaciones de regulación de los servicios de AWS subyacentes. Puede crear flujos de trabajo que procesen y publiquen modelos de machine learning, orquesten microservicios y controlen servicios de AWS, por ejemplo AWS Glue, para crear flujos de trabajo de extracción, transformación y carga (ETL). También puede crear flujos de trabajo automatizados y de larga duración para aplicaciones que requieren la interacción humana.

Al igual que AWS Data Pipeline, AWS Step Functions es un servicio totalmente gestionado por AWS. No se le pedirá que gestione la infraestructura, parchee a los trabajadores, gestione las actualizaciones de la versión del sistema operativo o similares.

Recomendamos migrar su carga de trabajo de AWS Data Pipeline a AWS Step Functions cuando:

- Está buscando un servicio de orquestación de flujos de trabajo sin servidor y de alta disponibilidad.
- Está buscando una solución rentable que cobre al mismo nivel que la ejecución de una sola tarea.
- Sus cargas de trabajo orquestan tareas para diversos servicios de AWS más, como Amazon EMR, Lambda, AWS Glue o DynamoDB.
- Está buscando una solución de bajo código que incluya un diseñador visual de arrastrar y soltar para la creación de flujos de trabajo y que no requiera aprender nuevos conceptos de programación.
- Está buscando un servicio que proporcione integraciones con más de 250 servicios de AWS adicionales que abarquen más de 11 000 acciones listas para usar, además de permitir integraciones con actividades y servicios personalizados y no de AWS.

Tanto AWS Data Pipeline como Step Functions utilizan el formato JSON para definir los flujos de trabajo. Esto permite almacenar sus flujos de trabajo en el control de código fuente, administrar las versiones, controlar el acceso y automatizarlos con CI/CD. Step Functions utiliza una sintaxis llamada Amazon State Language, que se basa completamente en JSON y permite una transición perfecta entre las representaciones textuales y visuales del flujo de trabajo.

Con Step Functions, puede elegir la misma versión de Amazon EMR que utiliza actualmente en AWS Data Pipeline.

Para migrar actividades en recursos gestionados AWS Data Pipeline, puede usar la AWS integración del servicio de SDK en Step Functions para automatizar el aprovisionamiento y la limpieza de los recursos.

Para migrar actividades en servidores en las instalaciones, instancias de EC2 administradas por el usuario o un clúster EMR administrado por el usuario, puede instalar un agente SSM en la instancia. Puede iniciar el comando mediante AWS Systems Manager Run Command de Step Functions. También puede iniciar la máquina de estados a partir de la programación definida en Amazon EventBridge.

AWS Step Functions tiene dos tipos de flujos de trabajo: flujos de trabajo estándar y flujos de trabajo exprés. En el caso de los flujos de trabajo estándar, se le cobrará en función del número de transiciones de estado necesarias para ejecutar la aplicación. En el caso de los flujos de trabajo exprés, se le cobrará en función del número de solicitudes del flujo de trabajo y de su duración. Obtenga más información sobre los precios en Precios de AWS Step Functions.

Migración de cargas de trabajo a Amazon MWAA

Flujos de trabajo administrados de Amazon para Apache Airflow (MWAA) es un servicio de orquestación administrada para Apache Airflow que facilita la configuración y el funcionamiento de canalizaciones de datos integrales en la nube a escala. Apache Airflow es una herramienta de código abierto que se utiliza para crear, programar y supervisar mediante programación secuencias de procesos y tareas denominadas “flujos de trabajo”. Con Amazon MWAA, puede usar el lenguaje de programación Airflow y Python para crear flujos de trabajo sin tener que administrar la infraestructura subyacente para garantizar la escalabilidad, la disponibilidad y la seguridad. Amazon MWAA escala automáticamente su capacidad de ejecución del flujo de trabajo para adaptarla a sus necesidades y está integrado con los servicios de seguridad de AWS para proporcionarle un acceso rápido y seguro a sus datos.

Al igual que AWS Data Pipeline, Amazon MWAA son servicios totalmente gestionados proporcionados por AWS. Si bien necesita aprender varios conceptos nuevos específicos de estos servicios, no es necesario que administre la infraestructura, aplique parches a los trabajadores, administre las actualizaciones de las versiones del sistema operativo o algo similar.

Le recomendamos que migre sus cargas de trabajo de AWS Data Pipeline a Amazon MWAA cuando:

- Está buscando un servicio gestionado y de alta disponibilidad para orquestar flujos de trabajo escritos en Python.
- Desea realizar la transición a una tecnología de código abierto totalmente gestionada y ampliamente adoptada, Apache Airflow, para lograr la máxima portabilidad.
- Necesita una plataforma única que pueda gestionar todos los aspectos de su canalización de datos, incluidos la ingesta, el procesamiento, la transferencia, las pruebas de integridad y los controles de calidad.
- Está buscando un servicio diseñado para orquestar la canalización de datos con funciones como una interfaz de usuario completa para facilitar la observabilidad, reinicios en caso de flujos de trabajo fallidos, recargas y reintentos de tareas.
- Está buscando un servicio que incluya más de 800 operadores y sensores prediseñados, que abarquen tanto servicios de AWS como los que no lo sean de AWS.

Los flujos de trabajo de Amazon MWAA se definen como gráficos acíclicos dirigidos (Directed Acyclic Graphs, DAG) que utilizan Python, por lo que también puede tratarlos como código fuente. El marco extensible de Python de Airflow le permite crear flujos de trabajo que se conecten con prácticamente cualquier tecnología. Viene con una interfaz de usuario completa para ver y monitorear los flujos de trabajo y se puede integrar fácilmente con los sistemas de control de versiones para automatizar el proceso de CI/CD.

Con Amazon MWAA, puede elegir la misma versión de Amazon EMR que utiliza actualmente en AWS Data Pipeline.

AWS cobra por el tiempo de funcionamiento de su entorno de Airflow más cualquier escalado automático adicional para proporcionar más capacidad de trabajadores o servidores web. Obtenga más información sobre los precios en Precios de Amazon Managed Workflows para Apache Airflow.

Mapeo de conceptos

La siguiente tabla contiene un mapeo de los principales conceptos utilizados por los servicios. Ayudará a las personas familiarizadas con Data Pipeline a entender la terminología de Step Functions y MWAA.

Data Pipeline	Adherencia	Step Functions	Amazon MWAA
Canalizaciones	Flujos de trabajo	Flujos de trabajo	Gráficos acrílicos directos

Data Pipeline	Adherencia	Step Functions	Amazon MWAA
Definición de la canalización JSON	Definición de flujos de trabajo o esquemas basados en Python	Amazon State Language JSON	Basado en Python
Actividades	Tareas	Estados y tareas	Tareas (operadores y sensores)
instancias	Ejecuciones de trabajo	Ejecuciones	DAG se ejecuta
Attempts	Intentos	Captadores y recolectores	Reintentos
Calendario de canalización	Activadores de programación	Tareas del programador de EventBridge	Cron, cronogramas, datos
Expresiones y funciones de canalizaciones	Biblioteca de esquemas	Step Functions, funciones intrínsecas y AWS Lambda	Framework extensible de Python

Muestras

En las siguientes secciones se enumeran ejemplos públicos a los que puede hacer referencia para migrar de AWS Data Pipeline a servicios individuales. Puede utilizarlos como ejemplos y crear su propia canalización a partir de los servicios individuales actualizándolos y probándolos en función de su caso de uso.

AWS GlueEjemplos de

La siguiente lista contiene ejemplos de implementaciones para los casos de uso de AWS Data Pipeline más comunes con AWS Glue.

- Ejecución de trabajos de Spark
- Copiar datos de JDBC a Amazon S3 (incluido Amazon Redshift)
- Copia de datos desde Amazon S3 a JDBC (incluido Amazon Redshift)
- Copia de datos de Amazon S3 a DynamoDB

- Movimiento de datos desde y hacia Amazon Redshift
- Acceso entre cuentas y entre regiones a tablas de DynamoDB

Ejemplos de AWS Step Functions

La siguiente lista contiene ejemplos de implementaciones para los casos de uso más comunes de AWS Data Pipeline con AWS Step Functions.

- Administración de un trabajo de Amazon EMR
- Ejecución de un trabajo de procesamiento de datos en Amazon EMR sin servidor
- Ejecución de trabajos de Hive/Pig/Hadoop
- Consulta de conjuntos de datos de gran tamaño (Amazon Athena, Amazon S3,AWS Glue)
- Ejecución de flujos de trabajo de ETL con Amazon Redshift
- Orquestar de AWS Gluerastreadores

Consulte tutoriales adicionales y ejemplos de proyectos para usar AWS Step Functions.

Muestras de Amazon MWAA

La siguiente lista contiene ejemplos de implementaciones para los casos de uso de AWS Data Pipeline más comunes con Amazon MWAA.

- Ejecución de un trabajo de Amazon EMR
- Creación de un complemento personalizado para Apache Hive y Hadoop
- Copia de datos desde Amazon S3 a Redshift
- Ejecutar un script de intérprete de comandos en una instancia EC2 remota
- Orquestación de flujos de trabajo híbridos (locales)

Consulte tutoriales adicionales y ejemplos de proyectos para usar Amazon MWAA.

Servicios relacionados

AWS Data Pipeline funciona con los siguientes servicios para almacenar datos.

- Amazon DynamoDB: ofrece una base de datos NoSQL totalmente administrada con un desempeño rápido a bajo costo. Para obtener más información, consulte la Guía para desarrolladores de Amazon DynamoDB.
- Amazon RDS: ofrece una base de datos relacional totalmente administrada que se puede escalar a conjuntos de datos de gran tamaño. Para obtener más información, consulte la Guía para desarrolladores del servicio de base de datos relacional de Amazon.
- Amazon Redshift: proporciona un almacenamiento de datos de varios petabytes rápido y totalmente administrado que permite analizar gran cantidad de datos de forma sencilla y económica. Para obtener más información, consulte la Guía de desarrollador de base de datos de Amazon Redshift.
- Amazon S3: proporciona un almacenamiento de objetos seguro, altamente escalable y duradero. Para obtener más información, consulte la Guía del usuario de Amazon Simple Storage Service.

AWS Data Pipeline funciona con los siguientes servicios de computación para transformar datos.

- Amazon EC2: proporciona capacidad de computación de tamaño variable (literalmente, servidores en los centros de datos de Amazon) que se utilizan para crear y alojar sistemas de software. Para obtener más información, consulte la Guía del usuario de Amazon EC2.
- Amazon EMR: hace que sea fácil, rápido y rentable distribuir y procesar grandes cantidades de datos entre servidores de Amazon EC2, utilizando un marco como Apache Hadoop o Apache Spark. Para obtener más información, consulte la Guía del desarrollador de Amazon EMR.

Acceso a AWS Data Pipeline

Puede crear, acceder y administrar las canalizaciones desde cualquiera de las siguientes interfaces:

- Consola de administración de AWS: proporciona una interfaz web que se puede utilizar para obtener acceso a AWS Data Pipeline.
- AWS Command Line Interface (AWS CLI): proporciona comandos para numerosos servicios de AWS, incluido AWS Data Pipeline, y es compatible con Windows, macOS y Linux. Para obtener más información sobre la instalación de la AWS CLI, consulte AWS Command Line Interface. Para obtener una lista de comandos para AWS Data Pipeline, consulte datapipeline.
- AWS SDKs (SDK de AWS): proporcionan API específicas de cada lenguaje y se encargan de muchos de los detalles de la conexión, tales como el cálculo de firmas, el control de reintentos de solicitud y el control de errores. Para obtener más información, consulte AWS SDKs.

- API de consulta: proporciona acciones de API de nivel bajo a las que se llama mediante solicitudes HTTPS. Utilizar la API de consulta es la forma más directa de obtener acceso a AWS Data Pipeline, pero requiere que la aplicación controle niveles de detalle de bajo nivel, tales como la generación del código hash para firmar la solicitud y el control de errores. Para obtener más información, consulte la Referencia de la API de AWS Data Pipeline.

Precios

Con los servicios de Amazon Web Services, solo se paga por lo que se usa. Para AWS Data Pipeline, se paga por la canalización según la frecuencia con la que estén programadas las ejecuciones de las actividades y las condiciones previas y el lugar en que se ejecuten. Para más información, consulte Precios de AWS Data Pipeline.

Si su cuenta de AWS tiene menos de 12 meses, puede utilizar la capa gratuita. La capa gratuita incluye tres condiciones previas de baja frecuencia y cinco actividades de baja frecuencia al mes sin ningún tipo de costo. Para obtener más información, consulte Capa gratuita de AWS.

Tipos de instancia compatibles con las actividades de trabajo de canalización

Cuando AWS Data Pipeline ejecuta una canalización, compila los componentes de la canalización para crear un conjunto de instancias Amazon EC2 procesables. Cada instancia contiene toda la información para realizar una tarea específica. El conjunto completo de instancias es la lista de tareas pendientes de la canalización. AWS Data Pipeline entrega las instancias a las aplicaciones de ejecución de tareas para que las procesen.

Las instancias EC2 aparecen en diferentes configuraciones, lo que se conoce como tipos de instancia. Cada tipo de instancia tiene una CPU, entrada/salida y capacidad de almacenamiento diferentes. Además de especificar el tipo de instancia para una actividad, puede elegir distintas opciones de compra. No todos los tipos de instancia están disponibles en todas las regiones de AWS. Si un tipo de instancia no está disponible, es posible que su canalización no se pueda aprovisionar o que se bloquee durante el aprovisionamiento. Para obtener información sobre la disponibilidad de las instancias, consulte la Página de precios de Amazon EC2. Abra el enlace de la opción de compra de instancias y filtre por Región para ver si un tipo de instancia está disponible en la región. Para obtener más información sobre estos tipos de instancias, familias y tipos de virtualización, consulte Instancias de Amazon EC2 y Matriz de tipos de instancias de las AMI de Amazon Linux.

En la siguiente tabla, se describen los tipos de instancias que admite AWS Data Pipeline. Puede utilizar AWS Data Pipeline para lanzar instancias de Amazon EC2 en cualquier región, incluidas las regiones en las que no se admite AWS Data Pipeline. Para obtener más información sobre las regiones en las que se admite AWS Data Pipeline, consulte Regiones y puntos de enlace de AWS.

Contenido

Instancias Amazon EC2 predeterminadas por región de AWS

Si no especifica un tipo de instancia en la definición de canalización, AWS Data Pipeline lanza una instancia de forma predeterminada.

En la tabla siguiente, se muestran las instancias Amazon EC2 que AWS Data Pipeline utiliza de forma predeterminada en las regiones en las que se admite AWS Data Pipeline.

Nombre de la región	Región	Tipo de instancia
Este de EE. UU. (Norte de Virginia)	us-east-1	m1.small
Oeste de EE. UU. (Oregón)	us-west-2	m1.small
Asia-Pacífico (Sídney)	ap-southeast-2	m1.small
Asia-Pacífico (Tokio)	ap-northeast-1	m1.small
UE (Irlanda)	eu-west-1	m1.small

En la tabla siguiente se muestran las instancias de Amazon EC2 que AWS Data Pipeline lanza de forma predeterminada en las regiones en las que no se admite AWS Data Pipeline.

Nombre de la región	Región	Tipo de instancia
Este de EE. UU. (Ohio)	us-east-2	t2.small

Nombre de la región	Región	Tipo de instancia
Oeste de EE. UU. (Norte de California)	us-west-1	m1.small
Asia-Pacífico (Bombay)	ap-south-1	t2.small
Asia-Pacífico (Singapur)	ap-southeast-1	m1.small
Asia-Pacífico (Seúl)	ap-northeast-2	t2.small
Canadá (centro)	ca-central-1	t2.small
UE (Fráncfort)	eu-central-1	t2.small
UE (Londres)	eu-west-2	t2.small
UE (París)	eu-west-3	t2.small
América del Sur (São Paulo)	sa-east-1	m1.small

Instancias Amazon EC2 compatibles adicionales

Además de las instancias predeterminadas que se crean en caso de que no especifique un tipo de instancia en la definición de canalización, son compatibles las siguientes instancias.

En la tabla siguiente, se muestran las instancias Amazon EC2 que AWS Data Pipelineadmite y que puede crear, si se especifica.

Clase de instancia	Tipos de instancias
Fin general	t2.nano | t2.micro | t2.small | t2.medium | t2.large
Optimizadas para computación	c3.large | c3.xlarge | c3.2xlarge | c3.4xlarge | c3.8xlarge | c4.large | c4.xlarge | c4.2xlarge | c4.4xlarge | c4.8xlarge | c5.xlarge | c5.9xlarge | c5.2xlarge | c5.4xlarge | c5.9xlarge | c5.18xlarge | c5d.xlarge | c5d.2xlarge | c5d.4xlarge | c5d.9xlarge | c5d.18xlarge

Clase de instancia	Tipos de instancias
Optimizadas para la memoria	m3.medium | m3.large | m3.xlarge | m3.2xlarge | m4.large | m4.xlarge | m4.2xlarge | m4.4xlarge | m4.10xlarge | m4.16xlarge | m5.xlarge | m5.2xlarge | m5.4xlarge | m5.12xlarge | m5.24xlarge | m5d.xlarge | m5d.2xlarge | m5d.4xlarge | m5d.12xlarge | m5d.24xlarge r3.large | r3.xlarge | r3.2xlarge | r3.4xlarge | r3.8xlarge | r4.large | r4.xlarge | r4.2xlarge | r4.4xlarge | r4.8xlarge | r4.16xlarge
Optimizadas para el almacenamiento	i2.xlarge | i2.2xlarge | i2.4xlarge | i2.8xlarge | hs1.8xlarge | g2.2xlarge | g2.8xlarge | d2.xlarge | d2.2xlarge | d2.4xlarge | d2.8xlarge

Instancias Amazon EC2 admitidas para clústeres de Amazon EMR

En esta tabla siguiente, se muestran las instancias Amazon EC2 que AWS Data Pipeline admite y que puede crear para clústeres de Amazon EMR, si se especifica. Para más información, consulte Tipos de instancias admitidas en la Guía de administración de Amazon EMR.

Clase de instancia	Tipos de instancias
Propósito general	m1.small | m1.medium | m1.large | m1.xlarge | m3.xlarge | m3.2xlarge
Optimizadas para computación	c1.medium | c1.xlarge | c3.xlarge | c3.2xlarge | c3.4xlarge | c3.8xlarge | cc1.4xlarge| cc2.8xlarge | c4.large | c4.xlarge | c4.2xlarge| c4.4xlarge | c4.8xlarge | c5.xlarge | c5.9xlarge | c5.2xlarge | c5.4xlarge | c5.9xlarge | c5.18xlarge | c5d.xlarge | c5d.2xlarge | c5d.4xlarge | c5d.9xlarge | c5d.18xlarge
Optimizadas para la memoria	m2.xlarge | m2.2xlarge | m2.4xlarge | r3.xlarge | r3.2xlarge | r3.4xlarge | r3.8xlarge | cr1.8xlarge | m4.large | m4.xlarge | m4.2xlarge | m4.4xlarge | m4.10xlarge | m4.16large | m5.xlarge | m5.2xlarge | m5.4xlarge | m5.12xlarge | m5.24xlarge | m5d.xlarge | m5d.2xlarge | m5d.4xlarge | m5d.12xlarge | m5d.24xlarge |

Clase de instancia	Tipos de instancias
	r4.large | r4.xlarge | r4.2xlarge | r4.4xlarge | r4.8xlarge | r4.16xlarge
Optimizadas para el almacenamiento	h1.4xlarge | hs1.2xlarge | hs1.4xlarge| hs1.8xlarge | i2.xlarge | i2.2xlarge | i2.4large | i2.8xlarge | d2.xlarge | d2.2xlarge| d2.4xlarge | d2.8xlarge
Computación acelerada	g2.2xlarge | cg1.4xlarge

AWS Data PipelineConceptos de

Antes de comenzar, lea acerca de los conceptos y los componentes clave de AWS Data Pipeline.

Contenido

Definición de la canalización

Una definición de canalización es la forma de comunicar la lógica de negocio a AWS Data Pipeline. Contiene la siguiente información:

- Nombres, ubicaciones y formatos de sus orígenes de datos
- Actividades que transforman los datos
- La programación de esas actividades
- Recursos que ejecutan sus actividades y condiciones previas
- Condiciones previas que deben cumplirse antes de que las actividades se puedan programar
- Modos de avisarle con actualizaciones de estado a medida que continúa la ejecución de la canalización

En la definición de la canalización, AWS Data Pipeline determina las tareas, las programa y las asigna a las aplicaciones de ejecución de tareas. Si una tarea no se completa correctamente, AWS Data Pipeline vuelve a intentar realizarla siguiendo las instrucciones y, en caso necesario, la reasigna a otra aplicación de ejecución de tareas. Si la tarea devuelve error repetidamente, puede configurar la canalización para que le notifique.

Por ejemplo, en la definición de la canalización, podría especificar que los archivos de registro generados por la aplicación deben archivarse cada mes de 2013 en un bucket de Amazon S3. AWS Data Pipeline crearía 12 tareas, cada una de ellas haciendo una copia de los datos correspondientes a un mes, independientemente de si el mes tenía 30, 31, 28 o 29 días.

Puede crear una definición de la canalización de cualquiera de estas formas:

- Gráficamente, mediante la consola de AWS Data Pipeline
- Textualmente, escribiendo un archivo JSON en el formato usado por la interfaz de línea de comandos
- Mediante programación, llamando al servicio web con uno de los SDK de AWS o la API de AWS Data Pipeline

Una definición de la canalización puede contener los siguientes tipos de componentes.

Componentes de canalización

Nodos de datos

La ubicación de los datos de entrada para una tarea o la ubicación donde se van a almacenar los datos de salida.

Actividades

Una definición del trabajo que se realizará de manera programada mediante un recurso informático y, habitualmente, nodos de datos de entrada y salida.

Condiciones previas

Una instrucción condicional que debe ser "true" antes de que una acción pueda ejecutarse.

Recursos

El recurso informático que realiza el trabajo que define una canalización.

Acciones

Una acción que se desencadena al cumplirse condiciones especificadas como, por ejemplo, el error de una actividad.

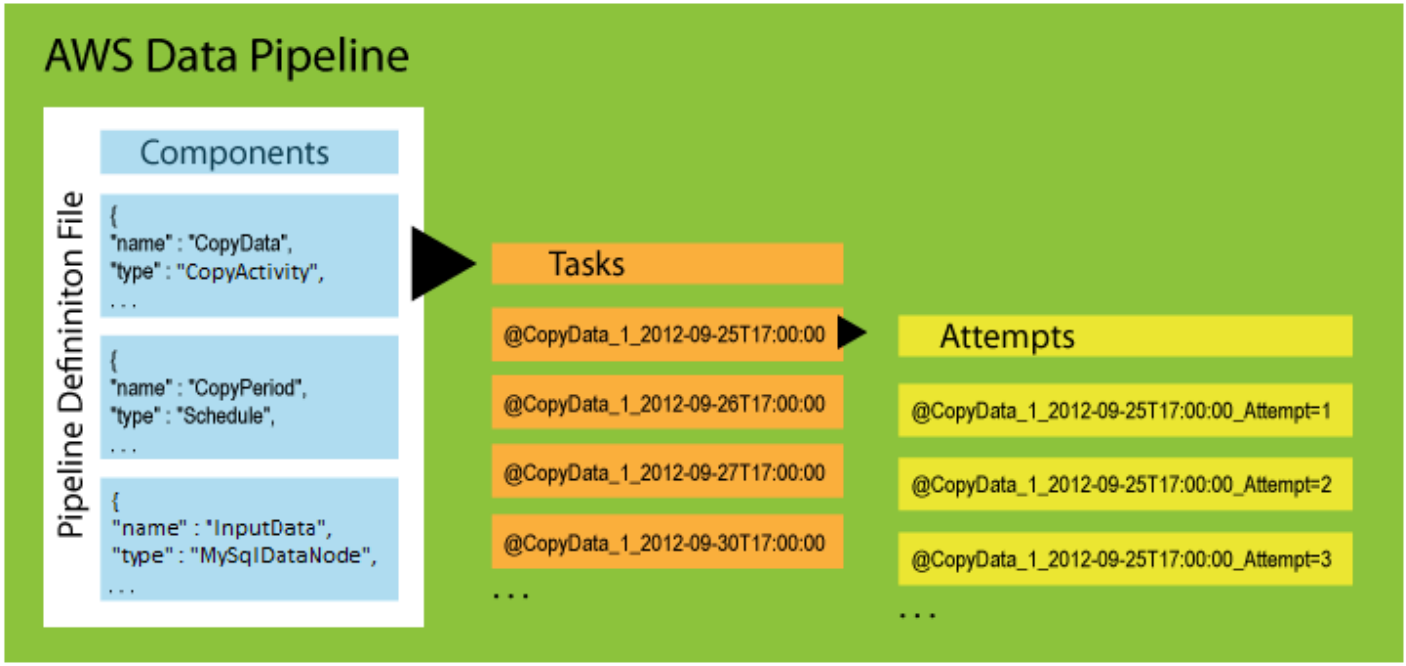
Para obtener más información, consulte Sintaxis de los archivos de definición de la canalización.

Componentes de canalización, instancias e intentos

Hay tres tipos de elementos asociados a una canalización programada:

- Componentes de canalización: los componentes de canalización representan la lógica empresarial de la canalización y están representados por las diferentes secciones de una definición de canalización. Los componentes de canalización especifican los orígenes de datos, las actividades, la programación y las condiciones previas del flujo de trabajo. Pueden heredar propiedades de los componentes principales. Las relaciones entre los componentes se definen por referencia. Los componentes de canalización definen las reglas de administración de datos.
- Instancias: cuando AWS Data Pipeline ejecuta una canalización, compila los componentes de canalización para crear un conjunto de instancias procesables. Cada instancia contiene toda la información para realizar una tarea específica. El conjunto completo de instancias es la lista de tareas pendientes de la canalización. AWS Data Pipeline entrega las instancias a las aplicaciones de ejecución de tareas para que las procesen.
- Intentos: para proporcionar una administración de datos sólida, AWS Data Pipeline vuelve a probar una operación fallida. Sigue haciéndolo hasta que la tarea alcanza el número máximo de reintentos permitidos. Los objetos de intento realizan un seguimiento de los diversos intentos, resultados y motivos de error si corresponde. Básicamente, es la instancia con un contador. AWS Data Pipeline realiza reintentos con los mismos recursos de los intentos anteriores, como los clústeres de Amazon EMR y las instancias EC2.

> **Note**
>
> El reintento de tareas fallidas constituye una parte importante de una estrategia de tolerancia a errores, mientras que las definiciones de AWS Data Pipeline proporcionan condiciones y umbrales para controlar los reintentos. Sin embargo, demasiados reintentos pueden retrasar la detección de un error no recuperable, ya que AWS Data Pipeline no notifica ningún error hasta que ha agotado todos los reintentos especificados. Los reintentos adicionales pueden acumular otros cargos si se ejecutan en recursos de AWS. Por ello, considere con cautela cuándo conviene utilizar valores más altos para la configuración predeterminada de AWS Data Pipeline que se usan para controlar los reintentos y los ajustes relacionados.

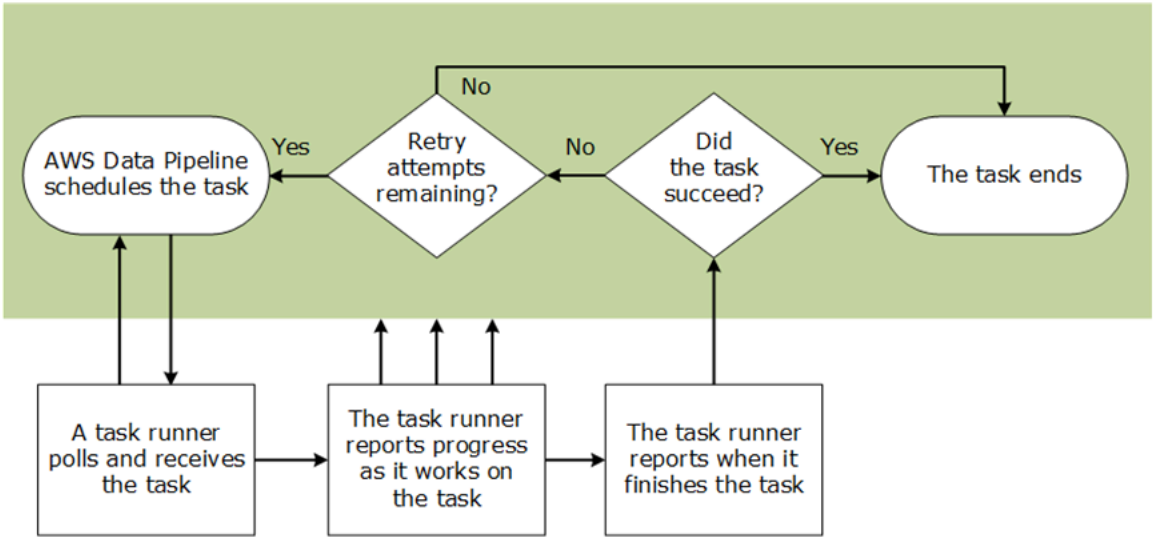


Aplicaciones de ejecución de tareas

Una aplicación de ejecución de tareas es una aplicación que sondea AWS Data Pipeline para comprobar si hay tareas y, a continuación, realiza dichas tareas.

Task Runner es una implementación predeterminada de una aplicación de ejecución de tareas proporcionada por AWS Data Pipeline. Cuando se instala y configura Task Runner, sondea AWS Data Pipeline para comprobar si hay tareas asociadas a las canalizaciones que están activadas. Cuando una tarea se asigna a Task Runner, realiza dicha tarea y notifica su estado a AWS Data Pipeline.

En el diagrama siguiente, se ilustra cómo interactúan AWS Data Pipeline y una aplicación de ejecución de tareas para procesar una tarea programada. Una tarea es una unidad discreta de trabajo que el servicio AWS Data Pipeline comparte con una aplicación de ejecución de tareas. Difiere de una canalización, que es una definición general de actividades y recursos que suele generar varias tareas.



Puede usar Task Runner de dos formas para procesar una canalización:

- AWS Data Pipeline instala Task Runner automáticamente en los recursos que el servicio web AWS Data Pipeline lanza y administra.
- Usted instala Task Runner en un recurso informático que administra como, por ejemplo, una instancia EC2 de ejecución prolongada, o un servidor en las instalaciones.

Para obtener más información sobre el trabajo con la aplicación de ejecución de tareas, consulte Operación de Task Runner.

Nodos de datos

En AWS Data Pipeline, un nodo de datos define la ubicación y el tipo de datos que una actividad de canalización usa como entrada o salida. AWS Data Pipeline admite los siguientes tipos de nodos de datos:

Nodo Dynamo DBData

Una tabla de DynamoDB que contiene datos para que HiveActivity o EmrActivity los use.

SqlDataNode

Una tabla SQL y una consulta de base de datos que representan los datos para que una actividad de canalización los use.

> **Note**
>
> Anteriormente, se usaba MySqlDataNode. Use SqlDataNode en su lugar.

RedshiftDataNode

Una tabla de Amazon Redshift que contiene datos para que RedshiftCopyActivity los use.

S3 DataNode

Una ubicación de Amazon S3 que contiene uno o varios archivos para que una actividad de canalización los use.

Bases de datos

AWS Data Pipeline admite los siguientes tipos de bases de datos:

JdbcDatabase

Una base de datos JDBC.

RdsDatabase

Bases de datos de Amazon RDS.

RedshiftDatabase

Base de datos de Amazon Redshift.

Actividades

En AWS Data Pipeline, una actividad es un componente de canalización que define el trabajo que se debe realizar. AWS Data Pipeline proporciona varias actividades previamente empaquetadas que se adaptan a escenarios comunes, como mover datos de una ubicación a otra, ejecutar consultas de Hive, etc. Las actividades son ampliables, por lo que puede ejecutar sus propios scripts personalizados para admitir infinitas combinaciones.

AWS Data Pipeline admite los siguientes tipos de actividades:

CopyActivity

Copia datos de una ubicación a otra.

EmrActivity

Ejecuta un clúster de Amazon EMR.

HiveActivity

Ejecuta una consulta de Hive en un clúster de Amazon EMR.

HiveCopyActivity

Ejecuta una consulta de Hive en un clúster de Amazon EMR con soporte para filtrado de datos avanzado y soporte para S3 DataNode y Nodo Dynamo DBData.

PigActivity

Ejecuta un script de Pig en un clúster de Amazon EMR.

RedshiftCopyActivity

Copia datos desde y hacia tablas Amazon Redshift.

ShellCommandActivity

Ejecuta un comando de shell de UNIX/Linux personalizado como actividad.

SqlActivity

Ejecuta una consulta SQL en una base de datos.

Algunas actividades poseen una compatibilidad especial para uso transitorio de datos y tablas de la base de datos. Para obtener más información, consulte Datos y tablas transitorios con actividades de canalización.

Condiciones previas

En AWS Data Pipeline, una condición previa es un componente de canalización que contiene instrucciones condicionales que deben cumplirse antes de que una actividad pueda ejecutarse. Por ejemplo, una condición previa puede comprobar si los datos de origen están presentes antes de que una actividad de canalización intente copiarlos. AWS Data Pipeline proporciona varias condiciones previas preempaquetadas que se adaptan a escenarios comunes, por ejemplo, si una tabla de la base de datos existe, si una clave de Amazon S3 está presente, etc. Sin embargo, las condiciones

previas son ampliables y le permiten ejecutar sus propios scripts personalizados para admitir infinitas combinaciones.

Hay dos tipos de condiciones previas; condiciones previas administradas por el sistema y condiciones previas administradas por el usuario. El servicio web AWS Data Pipeline ejecuta automáticamente las condiciones previas administradas por el sistema, que no requieren ningún recurso informático. Las condiciones previas administradas por el usuario solo se ejecutan en los recursos informáticos que especifique mediante los campos `runsOn` o `workerGroup`. El recurso `workerGroup` se deriva de la actividad que usa la condición previa.

Condiciones previas administradas por el sistema

DBDataDynamo existe

Comprueba si existen datos en una tabla de DynamoDB específica.

Dynamo existe DBTable

Comprueba si existe una tabla de DynamoDB.

S3 KeyExists

Comprueba si existe una clave de Amazon S3.

S3 PrefixNotEmpty

Comprueba si un prefijo de Amazon S3 está vacío.

Condiciones previas administradas por el usuario

Existe

Comprueba si existe un nodo de datos.

ShellCommandPrecondition

Ejecuta un comando de shell de Unix/Linux personalizado como condición previa.

Recursos

En AWS Data Pipeline, un recurso es el recurso informático que realiza el trabajo especificado por una actividad de canalización. AWS Data Pipeline admite los siguientes tipos de recursos:

Ec2Resource

Una instancia EC2 que realiza el trabajo definido por una actividad de canalización.

EmrCluster

Un clúster de Amazon EMR que realiza el trabajo definido por una actividad de canalización, como EmrActivity.

Los recursos pueden ejecutarse en la misma región con su conjunto de datos de trabajo, incluso una región distinta de la de AWS Data Pipeline. Para obtener más información, consulte Uso de una canalización con recursos en varias regiones.

Límites de recursos

AWS Data Pipeline se escala para adaptarse a un número elevado de tareas simultáneas, y es posible configurarlo para crear automáticamente los recursos necesarios para gestionar grandes cargas de trabajo. Estos recursos se crean automáticamente bajo su control y se tienen en cuenta para los límites de recursos de la cuenta de AWS. Por ejemplo, si configura AWS Data Pipeline para que cree automáticamente un clúster de 20 nodos de Amazon EMR para procesar datos y su cuenta de AWS tiene un límite de instancias EC2 establecido en 20, es posible que agote sin darse cuenta sus recursos de reposición disponibles. Como resultado, tenga en cuenta estas restricciones de recursos en el diseño o aumente los límites de su cuenta en consonancia. Para obtener más información sobre Service Limits, consulte Límites de los servicios de AWS en la Referencia general de AWS.

> **Note**
>
> El límite es una instancia por objeto de componente `Ec2Resource`.

Plataformas admitidas

Las canalizaciones pueden lanzar sus recursos en las siguientes plataformas:

EC2-Classic

Los recursos se ejecutan en una sola red plana que comparte con otros clientes.

EC2-VPC

Los recursos se ejecutan en una nube virtual privada (VPC), que está aislada lógicamente para su cuenta de AWS.

Su cuenta de AWS puede lanzar recursos en ambas plataformas o solo en EC2-VPC, según cada región. Para obtener más información, consulte Plataformas compatibles en la Guía del usuario de Amazon EC2.

Si su cuenta de AWS solo admite EC2-VPC, creamos una VPC predeterminada automáticamente en cada región de AWS. De forma predeterminada, lanzamos sus recursos en una subred predeterminada de la VPC predeterminada. De forma alternativa, puede crear una VPC no predeterminada y especificar una de sus subredes al configurar sus recursos. A continuación, lanzamos sus recursos en la subred especificada de la VPC no predeterminada.

Al lanzar una instancia en una VPC, debe especificar un grupo de seguridad creado específicamente para esa VPC. No puede especificar un grupo de seguridad que ha creado para EC2-Classic al lanzar una instancia en una VPC. Además, debe usar el ID de grupo de seguridad y no el nombre del grupo de seguridad para identificar un grupo de seguridad de una VPC.

Instancias de spot Amazon EC2 con clústers Amazon EMR y AWS Data Pipeline

Las canalizaciones pueden utilizar instancias de spot de Amazon EC2 para los nodos de tareas en sus recursos del clúster de Amazon EMR. De forma predeterminada, las canalizaciones usan instancias bajo demanda. Las instancias de spot le permiten usar instancias EC2 libres y ejecutarlas. El modelo de precios de instancias de spot complementa los modelos de precios de instancias reservadas y bajo demanda, proporcionando posiblemente la opción más rentable para obtener capacidad de cómputo, dependiendo de su aplicación. Para obtener más información, consulte la página de producto de Instancias de spot de Amazon EC2.

Si se usan instancias de spot, AWS Data Pipeline envía el precio máximo ofrecido por la instancia de spot a Amazon EMR cuando se lanza el clúster. También asigna de forma automática el trabajo del clúster al número de nodos de tareas de instancia de spot que se defina mediante el campo `taskInstanceCount`. AWS Data Pipeline limita las instancias de spot para nodos de tarea a fin de garantizar que haya nodos principales bajo demanda disponibles para ejecutar la canalización.

Puede editar una instancia de recurso de canalización fallida o completada para añadir instancias de spot; cuando la canalización vuelve a lanzar el clúster, utiliza instancias de spot para los nodos de tarea.

Consideraciones de instancias de spot

Al usar instancias de spot con AWS Data Pipeline, se aplican las siguientes consideraciones:

- Las instancias de spot pueden finalizarse cuando el precio de la instancia de spot supere el precio máximo ofrecido por la instancia o por razones de capacidad de Amazon EC2. Sin embargo, los datos no se pierden porque AWS Data Pipeline emplea clústeres con nodos principales que siempre son instancias bajo demanda y no están sujetos a la terminación.
- Las instancias de spot pueden tardar más tiempo en empezar, ya que cumple su capacidad de forma asíncrona. Por lo tanto, una canalización de una instancia de spot podría ejecutarse más lentamente que una canalización de instancia bajo demanda equivalente.
- Su clúster podría no ejecutarse si no recibe sus instancias de spot, como cuando su precio máximo es demasiado bajo.

Acciones

AWS Data PipelineLas acciones de son pasos que sigue un componente de canalización cuando se producen determinados eventos, como actividades realizadas correctamente, fallidas o que se realizan tarde. El campo de evento de una actividad hace referencia a una acción, como una referencia a `snsalarm` en el campo `onLateAction` de `EmrActivity`.

AWS Data Pipeline confía en las notificaciones de Amazon SNS como forma principal de indicar el estado de las canalizaciones y sus componentes de un modo desatendido. Para obtener más información, consulte Amazon SNS. Además de las notificaciones de SNS, se puede usar la CLI y la consola de AWS Data Pipeline para obtener información sobre el estado de la canalización.

AWS Data Pipeline admite las siguientes acciones:

SnsAlarm

> Una acción que envía una notificación de SNS a un tema basado en los eventos `onSuccess`, `OnFail` y `onLateAction`.

Finalizar

Una acción que desencadena la cancelación de una actividad, recurso o nodo de datos pendientes o inacabados. No puede finalizar acciones que incluyen `onSuccess`, `OnFail` u `onLateAction`.

Monitorización proactiva de canalizaciones

La mejor forma de detectar problemas es monitorizar sus canalizaciones de manera proactiva desde el inicio. Puede configurar los componentes de canalización para informarle de determinadas situaciones o eventos, como cuando un componente de canalización produce un error o no comienza a su hora de inicio programada. AWS Data Pipeline facilita la configuración de notificaciones proporcionando campos de evento en los componentes de canalización que se pueden asociar a notificaciones de Amazon SNS como `onSuccess`, `OnFail` y `onLateAction`.

Configurándose para AWS Data Pipeline

Antes de usarlo AWS Data Pipeline por primera vez, complete las siguientes tareas.

Tareas

- Inscríbase en AWS
- Cree funciones de IAM AWS Data Pipeline y canalice los recursos
- Permita que la entidad principal de IAM (usuarios y grupos) realicen las acciones necesarias
- Concesión de acceso mediante programación

Después de completar estas tareas, puede empezar a utilizarlas AWS Data Pipeline. Para ver un tutorial básico, consulte Introducción a AWS Data Pipeline.

Inscríbase en AWS

Cuando te registras en Amazon Web Services (AWS), tu cuenta de AWS se registra automáticamente en todos los servicios de AWS, incluidos AWS Data Pipeline. Solo se le cobrará por los servicios que utilice. Para obtener más información sobre las tasas AWS Data Pipeline de uso, consulte AWS Data Pipeline.

Inscríbase en una Cuenta de AWS

Si no tiene uno Cuenta de AWS, complete los siguientes pasos para crearlo.

Para suscribirse a una Cuenta de AWS

1. Abrir https://portal.aws.amazon.com/billing/registro.
2. Siga las instrucciones que se le indiquen.

   Parte del procedimiento de registro consiste en recibir una llamada telefónica o mensaje de texto e indicar un código de verificación en el teclado del teléfono.

   Cuando te registras en un Cuenta de AWS, Usuario raíz de la cuenta de AWSse crea un. El usuario raíz tendrá acceso a todos los Servicios de AWS y recursos de esa cuenta. Como práctica recomendada de seguridad, asigne acceso administrativo a un usuario y utilice únicamente el usuario raíz para realizar tareas que requieren acceso de usuario raíz.

AWS te envía un correo electrónico de confirmación una vez finalizado el proceso de registro. En cualquier momento, puede ver la actividad de su cuenta actual y administrarla accediendo a https://aws.amazon.com/y seleccionando Mi cuenta.

Creación de un usuario con acceso administrativo

Después de crear un usuario administrativo Cuenta de AWS, asegúrelo Usuario raíz de la cuenta de AWS AWS IAM Identity Center, habilite y cree un usuario administrativo para no usar el usuario root en las tareas diarias.

Proteja su Usuario raíz de la cuenta de AWS

1. Inicie sesión Consola de administración de AWScomo propietario de la cuenta seleccionando el usuario root e introduciendo su dirección de Cuenta de AWS correo electrónico. En la siguiente página, escriba su contraseña.

   Para obtener ayuda para iniciar sesión con el usuario raíz, consulte Iniciar sesión como usuario raíz en la Guía del usuario de AWS Sign-In .
2. Active la autenticación multifactor (MFA) para el usuario raíz.

   Para obtener instrucciones, consulte Habilitar un dispositivo MFA virtual para el usuario Cuenta de AWS raíz (consola) en la Guía del usuario de IAM.

Creación de un usuario con acceso administrativo

1. Activar IAM Identity Center.

   Consulte las instrucciones en Activar AWS IAM Identity Center en la Guía del usuario de AWS IAM Identity Center .
2. En IAM Identity Center, conceda acceso administrativo a un usuario.

   Para ver un tutorial sobre su uso Directorio de IAM Identity Center como fuente de identidad, consulte Configurar el acceso de los usuarios con la configuración predeterminada Directorio de IAM Identity Center en la Guía del AWS IAM Identity Center usuario.

Inicio de sesión como usuario con acceso de administrador

- Para iniciar sesión con el usuario de IAM Identity Center, use la URL de inicio de sesión que se envió a la dirección de correo electrónico cuando creó el usuario de IAM Identity Center.

Para obtener ayuda para iniciar sesión con un usuario del Centro de identidades de IAM, consulte Iniciar sesión en el portal de AWS acceso en la Guía del AWS Sign-In usuario.

Concesión de acceso a usuarios adicionales

1. En IAM Identity Center, cree un conjunto de permisos que siga la práctica recomendada de aplicar permisos de privilegios mínimos.

   Para conocer las instrucciones, consulte Create a permission set en la Guía del usuario de AWS IAM Identity Center .
2. Asigne usuarios a un grupo y, a continuación, asigne el acceso de inicio de sesión único al grupo.

   Para conocer las instrucciones, consulte Add groups en la Guía del usuario de AWS IAM Identity Center .

Cree funciones de IAM AWS Data Pipeline y canalice los recursos

AWS Data Pipeline requiere funciones de IAM que determinan los permisos para realizar acciones y acceder AWS a los recursos. La función de canalización determina los permisos que AWS Data Pipeline tienen y una función de recurso determina los permisos que tienen las aplicaciones que se ejecutan en los recursos de canalización, como EC2 las instancias. Podrá especificarlos al crear una canalización. Incluso si no especifica un rol personalizado y usa los roles `DataPipelineDefaultRole` predeterminados`DataPipelineDefaultResourceRole`, primero debe crear los roles y adjuntar políticas de permisos. Para obtener más información, consulte Funciones de IAM para AWS Data Pipeline.

Permita que la entidad principal de IAM (usuarios y grupos) realicen las acciones necesarias

Para trabajar con una canalización, una entidad principal de IAM (un usuario o un grupo) de su cuenta debe poder realizar las acciones de AWS Data Pipeline necesarias y las acciones para otros servicios, tal como se defina en su canalización.

Para simplificar los permisos, puedes adjuntar la política AWSDataPipeline_FullAccessgestionada a las entidades principales de IAM. Esta política gestionada permite al director realizar todas

las acciones que requiera un usuario y realizar las `iam:PassRole` acciones en las funciones predeterminadas que se utilizan AWS Data Pipeline cuando no se especifica una función personalizada.

Le recomendamos encarecidamente que evalúe detenidamente esta política administrada y restrinja los permisos únicamente a los que necesiten sus usuarios. Si es necesario, utilice esta política como punto de partida y, a continuación, elimine los permisos para crear una política de permisos en línea más restrictiva que pueda adjuntar a las entidades principales de IAM. Para obtener más información acerca de las políticas de permisos de ejemplo, consulte Ejemplos de políticas para AWS Data Pipeline.

Se debe incluir una declaración de política similar a la del siguiente ejemplo en una política adjunta a cualquier entidad principal de IAM que utilice la canalización. Esta declaración permite a la entidad principal de IAM realizar la acción `PassRole` en los roles que utiliza una canalización. Si no usa los roles predeterminados, reemplace *`MyPipelineRole`* y *`MyResourceRole`* con los roles personalizados que cree.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource": [
                "arn:aws:iam::*:role/MyPipelineRole",
                "arn:aws:iam::*:role/MyResourceRole"
            ]
        }
    ]
}
```

El siguiente procedimiento muestra cómo crear un grupo de IAM, adjuntar la política AWSDataPipeline_FullAccessadministrada al grupo y, a continuación, añadir usuarios al grupo. Puede usar este procedimiento para cualquier política en línea

Para crear un grupo de usuarios **`DataPipelineDevelopers`** y adjuntar la política AWSDataPipeline_FullAccess

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Groups (Grupos), Create New Group (Crear grupo nuevo).
3. Ingrese un Nombre de grupo (por ejemplo, **`DataPipelineDevelopers`**) y, a continuación, elija Paso siguiente.
4. Introduzca **`AWSDataPipeline_FullAccess`** para Filtrar y, a continuación, selecciónelo de la lista.
5. Elija Next Step (Paso siguiente) y, a continuación, seleccione Create Group (Crear grupo).
6. Para añadir usuarios al grupo.
   a. Seleccione el grupo que creó de la lista de grupos.
   b. Elija Group Actions, Add Users to Group.
   c. Seleccione los usuarios que desea agregar y, a continuación, elija Agregar usuarios al grupo.

Concesión de acceso mediante programación

Los usuarios necesitan acceso programático si quieren interactuar con personas AWS ajenas a. Consola de administración de AWS La forma de conceder el acceso programático depende del tipo de usuario que acceda. AWS

Para conceder acceso programático a los usuarios, elija una de las siguientes opciones.

¿Qué usuario necesita acceso programático?	Para	Mediante
IAM	(Recomendado) Utilice las credenciales de la consola como credenciales temporales para firmar las solicitudes programáticas dirigidas al AWS CLI AWS SDKs, o. AWS APIs	Siga las instrucciones de la interfaz que desea utilizar: • Para ello AWS CLI, consulte Iniciar sesión para el desarrollo AWS local en la

¿Qué usuario necesita acceso programático?	Para	Mediante
		Guía del AWS Command Line Interface usuario. • Para ello AWS SDKs, consulte Iniciar sesión para el desarrollo AWS local en la Guía de referencia de AWS SDKs and Tools.
Identidad del personal (Usuarios administrados en el IAM Identity Center)	Utilice credenciales temporales para firmar las solicitudes programáticas dirigidas al AWS CLI, AWS SDKs, o AWS APIs.	Siga las instrucciones de la interfaz que desea utilizar: • Para ello AWS CLI, consulte Configuración del AWS CLI uso AWS IAM Identity Center en la Guía del AWS Command Line Interface usuario. • Para AWS SDKs ver las herramientas y AWS APIs, consulte la autenticación del Centro de Identidad de IAM en la Guía de referencia de herramientas AWS SDKs y herramientas.
IAM	Utilice credenciales temporales para firmar las solicitudes programáticas dirigidas al AWS CLI AWS SDKs, o. AWS APIs	Siga las instrucciones de Uso de credenciales temporales con AWS recursos de la Guía del usuario de IAM.

¿Qué usuario necesita acceso programático?	Para	Mediante
IAM	(No recomendado) Utilice credenciales de larga duración para firmar las solicitudes programáticas dirigidas al AWS CLI AWS SDKs, o. AWS APIs	Siga las instrucciones de la interfaz que desea utilizar: • Para ello AWS CLI, consulte Autenticación con credencia les de usuario de IAM en la Guía del AWS Command Line Interface usuario. • Para obtener AWS SDKs información sobre las herramientas, consulte Autenticarse con credencia les de larga duración en la Guía de referencia de herramientas AWS SDKs y herramientas. • Para ello AWS APIs, consulte Administrar las claves de acceso para los usuarios de IAM en la Guía del usuario de IAM.

Introducción a AWS Data Pipeline

AWS Data Pipeline le ayuda a secuenciar, programar, ejecutar y administrar cargas de trabajo de procesamiento de datos recurrentes de forma fiable y rentable. Este servicio simplifica el diseño de actividades de extracción, transformación y carga (ETL) mediante datos estructurados y sin estructurar, tanto local como en la nube, según su lógica empresarial.

Para usar AWS Data Pipeline, cree una definición de canalización que especifique la lógica de negocio para su procesamiento de datos. Una definición de la canalización típica consta de actividades que definen el trabajo que se realizará, y nodos de datos que definen la ubicación y el tipo de datos de entrada y salida y una programación que determina cuándo se realizan las actividades.

En este tutorial, ejecuta un script de comandos de shell que cuenta el número de solicitudes GET en registros del servidor web Apache. Esta canalización se ejecuta cada 15 minutos durante una hora y escribe la salida a Amazon S3 en cada iteración.

Requisitos previos

Antes de comenzar, complete las tareas de Configurándose para AWS Data Pipeline.

Objetos de canalización

La canalización usa los siguientes objetos:

ShellCommandActivity

Lee el archivo de registro de entrada y cuenta el número de errores.

S3 DataNode (input)

El bucket de S3 que contiene el archivo de registro de entrada.

S3 DataNode (salida)

El bucket de S3 para la salida.

Ec2Resource

El recurso informático que AWS Data Pipeline usa para realizar la actividad.

Tenga en cuenta que si tiene una gran cantidad de datos de los archivos de registro, puede configurar su canalización para usar un clúster de EMR a fin de procesar los archivos en lugar de una instancia EC2.

Schedule

Define que la actividad se realiza cada 15 minutos durante una hora.

Tareas

- Crear la canalización
- Monitorizar la canalización en ejecución
- Ver la salida
- Eliminar la canalización

Crear la canalización

La forma más rápida de comenzar a trabajar con AWS Data Pipeline es usar una definición de la canalización denominada plantilla.

Para crear la canalización

1. Abra la consola de AWS Data Pipeline en https://console.aws.amazon.com/datapipeline/.
2. En la barra de navegación, seleccione una región. Puede seleccionar cualquier región disponible, independientemente de su ubicación. Muchos recursos de AWS son específicos de cada región, pero AWS Data Pipeline le permite usar recursos que se encuentran en una región diferente a la de la canalización.
3. La primera pantalla que vea dependerá de si ha creado una canalización en la región actual.
   a. Si no ha creado una canalización en esta región, la consola muestra una pantalla introductoria. Elija Get started now.
   b. Si ya ha creado una canalización en esta región, la consola muestra una página que enumera sus canalizaciones para la región. Elija Create new pipeline (Crear nueva canalización).
4. En Nombre, escriba el nombre de la canalización.
5. (Opcional) En Descripción, escriba una descripción para su canalización.
6. Para Source, seleccione Build using a template y, a continuación, seleccione la siguiente plantilla: Getting Started using ShellCommandActivity.

7. En la sección Parameters, que se abrió al seleccionar la plantilla, deje S3 input folder y Shell command to run con sus valores predeterminados. Haga clic en el icono de la carpeta junto a S3 output folder, seleccione uno de los buckets o carpetas y, a continuación, haga clic en Select.
8. En Schedule, deje los valores predeterminados. Al activar la canalización, empieza la ejecución de la canalización y, después, continúa cada 15 minutos durante una hora.

   Si lo prefiere, puede seleccionar Run once on pipeline activation en su lugar.
9. En Configuración de canalización, deje el registro activado. Elija el icono de carpeta en la ubicación de S3 para los registros, seleccione uno de sus buckets o carpetas y, a continuación, elija Seleccionar.

   Si lo prefiere, puede desactivar el registro en su lugar.
10. En Seguridad/acceso, deje Roles de IAM en Predeterminado.
11. Haga clic en Activate (Activar).

    Si lo prefiere, puede elegir Editar en Architect para modificar esta canalización. Por ejemplo, puede añadir condiciones previas.

Monitorizar la canalización en ejecución

Después de activar la canalización, se abrirá la página Execution details, donde puede monitorizar el progreso de la canalización.

Para monitorizar el progreso de la canalización

1. Haga clic en Update o pulse F5 para actualizar el estado mostrado.

   > **Tip**
   >
   > Si no hay ninguna ejecución en la lista, asegúrese de que Start (in UTC) y End (in UTC) abarquen el principio y el final programados de la canalización y, a continuación, haga clic en Update.
2. Cuando el estado de todos los objetos en la canalización es `FINISHED`, la canalización ha completado correctamente las tareas programadas.

3. Si la canalización no se completa correctamente, compruebe su configuración para ver si existe algún problema. Para obtener más información sobre cómo solucionar problemas de ejecuciones de instancias de la canalización, consulte Resolución de problemas comunes.

Ver la salida

Abra la consola de Amazon S3 y vaya al bucket. Si ejecutó su canalización cada 15 minutos durante una hora, verá cuatro subcarpetas con marca de tiempo. Cada subcarpeta contiene la salida en un archivo denominado `output.txt`. Dado que ejecutamos el script en el mismo archivo de entrada cada vez, los archivos de salida son idénticos.

Eliminar la canalización

Para dejar de incurrir en cargos, elimine su canalización. Al eliminar su canalización, se borran la definición de la canalización y todos los objetos asociados.

Para eliminar su canalización

1. En la página Lista de canalizaciones, seleccione la canalización.
2. Haga clic en Acciones y, después, Eliminar.
3. Cuando se le pida confirmación, seleccione Eliminar.

Cuando ya no necesite la salida de este tutorial, elimine las carpetas de salida del bucket de Amazon S3.

Trabajar con canalizaciones

Puede administrar, crear y modificar canalizaciones con la interfaz de la línea de comandos (CLI) o SDK de AWS. En las secciones siguientes, se presentan conceptos fundamentales de AWS Data Pipeline y se muestra cómo trabajar con canalizaciones.

> **Important**
>
> Antes de comenzar, consulte Configurándose para AWS Data Pipeline.

Contenido

Creación de una canalización

AWS Data Pipeline ofrece varias maneras de crear canalizaciones:

- Utilice la AWS Command Line Interface (CLI) con una plantilla proporcionada para su comodidad. Para obtener más información, consulte Cree una canalización a partir de plantillas de Data Pipeline mediante la CLI.
- Utilice la AWS Command Line Interface (CLI) con un archivo de definición de canalización en formato JSON.

- Utilice un SDK de AWS con una API específica del idioma. Para obtener más información, consulte Uso de la API.

Cree una canalización a partir de plantillas de Data Pipeline mediante la CLI

Data Pipeline proporciona varias definiciones de canalización preconfiguradas, conocidas como plantillas. Puede utilizar plantillas para comenzar a utilizar AWS Data Pipeline con rapidez. Estas plantillas están disponibles en un bucket público en la ubicación de Amazon S3: `s3://datapipeline-us-east-1/templates/`. Estas plantillas predefinidas se crean para lograr casos de uso específicos y se pueden usar para crear canalizaciones. Puede utilizar `aws s3 ls --recursive "s3://datapipeline-us-east-1/templates/"` para enumerar todas las plantillas disponibles.

Crear una canalización a partir de una plantilla mediante CLI

Digamos que quiere crear una canalización que exporte una tabla de DynamoDB a Amazon S3. La plantilla que se utilizará en este caso se encuentra en: `s3://datapipeline-us-east-1/templates/DynamoDB Templates/Export DynamoDB table to S3.json`

Para descargar la plantilla JSON y crear una canalización mediante la CLI

1. Descargue la plantilla mediante la `aws s3 cp` CLI o curl. Por ejemplo:

   ```
   aws s3 cp "s3://datapipeline-us-east-1/templates/DynamoDB Templates/Export DynamoDB
    table to S3.json" <destination directory>
   ```

2. Realice cambios en la plantilla descargada según sea necesario. Por ejemplo, para usar la última versión de EMR, cambie el campo `releaseLabel` del objeto `EmrClusterForBackup`, cambie los tipos de instancia maestra y principal y cambie los valores predeterminados de los parámetros de la plantilla.
3. Creación de una canalización mediante la CLI de `create-pipeline`. Por ejemplo:

   ```
   aws datapipeline create-pipeline --name my-ddb-backup-pipeline --unique-id my-ddb-
   backup-pipeline --region ap-northeast-1
   ```

4. Anote el ID de canalización creado.
5. Use la `put-pipeline-definition` para cargar la definición. Proporcione los valores de los parámetros cuyos valores predeterminados desee anular mediante la opción `--parameter-values`.

Para obtener más información acerca de las plantillas, consulte Choose a template (Elegir una plantilla).

Choose a template (Elegir una plantilla)

Las siguientes plantillas se encuentran disponibles para su descarga desde el bucket de Amazon S3: `s3://datapipeline-us-east-1/templates/`.

Plantillas

- Introducción al uso de ShellCommandActivity
- Ejecutar comandos de la CLI de AWS.
- Exportación de tabla de DynamoDB a S3
- Importación de datos de copia de seguridad de DynamoDB desde S3
- Ejecución de trabajos en un clúster de Amazon EMR
- Full copy of Amazon RDS MySQL Table to Amazon S3
- Incremental copy of Amazon RDS MySQL table to Amazon S3
- Load S3 data into Amazon RDS MySQL table
- Full copy of Amazon RDS MySQL table to Amazon Redshift
- Incremental copy of an Amazon RDS MySQL table to Amazon Redshift
- Cargar datos desde Amazon S3 en Amazon Redshift

Introducción al uso de ShellCommandActivity

La plantilla Getting Started using ShellCommandActivity ejecuta un script de plantilla de comandos para contar el número de solicitudes GET en un archivo de registro. La salida se escribe en una ubicación de Amazon S3 con marca de tiempo en cada ejecución programada de la canalización.

La plantilla usa los siguientes objetos de canalización:

- ShellCommandActivity
- S3InputNode
- S3OutputNode
- Ec2Resource

Ejecutar comandos de la CLI de AWS.

Esta plantilla ejecuta un comando de la AWS CLI especificado por el usuario a intervalos programados.

Exportación de tabla de DynamoDB a S3

La plantilla Export DynamoDB table to S3 programa un clúster de Amazon EMR para exportar datos de una tabla de DynamoDB a un bucket de Amazon S3. Esta plantilla utiliza un clúster de Amazon EMR de tamaño proporcional al valor del rendimiento disponible para la tabla de DynamoDB. Aunque puede ampliar el número de IOP en una tabla, puede incurrir en costos adicionales durante la importación y la exportación. Anteriormente, la exportación utilizaba una HiveActivity, pero ahora utiliza MapReduce nativo.

La plantilla usa los siguientes objetos de canalización:

- EmrActivity
- EmrCluster
- Nodo Dynamo DBData
- S3 DataNode

Importación de datos de copia de seguridad de DynamoDB desde S3

La plantilla Import DynamoDB backup data from S3 programa un clúster de Amazon EMR para cargar una copia de seguridad de DynamoDB previamente creada en Amazon S3 en una tabla de DynamoDB. Los elementos existentes en la tabla de DynamoDB se actualizan con los datos de la copia de seguridad y se añaden elementos nuevos a la tabla. Esta plantilla utiliza un clúster de Amazon EMR de tamaño proporcional al valor del rendimiento disponible para la tabla de DynamoDB. Aunque puede ampliar el número de IOP en una tabla, puede incurrir en costos adicionales durante la importación y la exportación. Anteriormente, la importación utilizaba una HiveActivity, pero ahora utiliza MapReduce nativo.

La plantilla usa los siguientes objetos de canalización:

- EmrActivity
- EmrCluster
- Nodo Dynamo DBData
- S3 DataNode

- S3 PrefixNotEmpty

Ejecución de trabajos en un clúster de Amazon EMR

La plantilla Run Job on an Elastic MapReduce Cluster lanza un clúster de Amazon EMR en función de los parámetros proporcionados y comienza a ejecutar pasos en función de la programación especificada. Una vez que el trabajo se completa, el clúster de EMR termina. Se puede especificar acciones de arranque opcionales para instalar software adicional y para cambiar la configuración de la aplicación en el clúster.

La plantilla usa los siguientes objetos de canalización:

- EmrActivity
- EmrCluster

Full copy of Amazon RDS MySQL Table to Amazon S3

La plantilla Full Copy of RDS MySQL Table to S3 copia una tabla de Amazon RDS MySQL completa y almacena la salida en una ubicación de Amazon S3. La salida se almacena como un archivo CSV en una subcarpeta con marca de tiempo bajo la ubicación de Amazon S3 especificada.

La plantilla usa los siguientes objetos de canalización:

- CopyActivity
- Ec2Resource
- SqlDataNode
- S3 DataNode

Incremental copy of Amazon RDS MySQL table to Amazon S3

La plantilla Incremental Copy of RDS MySQL Table to S3 realiza una copia incremental de los datos desde una tabla de Amazon RDS MySQL y almacena la salida en una ubicación de Amazon S3. La tabla MySQL de Amazon RDS debe tener una columna Last Modified (Última modificación).

Esta plantilla copia los cambios que se realicen en la tabla entre intervalos programados a partir de la hora de comienzo programada. El tipo de programación es series temporales, de modo que si se ha programado una copia para una hora determinada, AWS Data Pipeline copia las filas que tengan una marca temporal “Last Modified” dentro de la misma hora. Las eliminaciones físicas realizadas en

la tabla no se copian. La salida se escribe en una subcarpeta con marca de tiempo bajo la ubicación de Amazon S3 en cada ejecución programada.

La plantilla usa los siguientes objetos de canalización:

- CopyActivity
- Ec2Resource
- SqlDataNode
- S3 DataNode

Load S3 data into Amazon RDS MySQL table

La plantilla Load S3 Data into RDS MySQL Table programa una instancia EC2 para copiar el archivo CSV desde la ruta de archivo de Amazon S3 que se especifica a continuación en una tabla MYSQL de Amazon RDS. El archivo CSV no debe tener un encabezado de fila. La plantilla actualiza las entradas existentes en la tabla de MySQL de Amazon RDS con las de los datos de Amazon S3 y agrega nuevas entradas a partir de los datos de Amazon S3 a la tabla de MySQL de Amazon RDS. Puede cargar los datos en una tabla existente o proporcionar una consulta SQL para crear una nueva tabla.

La plantilla usa los siguientes objetos de canalización:

- CopyActivity
- Ec2Resource
- SqlDataNode
- S3 DataNode

Plantilla de Amazon RDS a Amazon Redshift

Las dos plantillas siguientes copian tablas de MySQL de Amazon RDS a Amazon Redshift con un script de traducción, que crea una tabla de Amazon Redshift utilizando el esquema de tabla de origen, con las siguientes salvedades:

- Si no se especifica una clave de distribución clave, la primera clave principal de la tabla de Amazon RDS se establece como clave de distribución.
- No puede omitir una columna que esté presente en MySQL de Amazon RDS cuando esté haciendo una copia en Amazon Redshift.

- (Opcional) Puede proporcionar un mapeo de tipo de datos de columna MySQL de Amazon RDS a Amazon Redshift como uno de los parámetros de la plantilla. Si se especifica esto, el script lo utiliza para crear la tabla de Amazon Redshift.

Si se utiliza el modo de inserción de `Overwrite_Existing` de Amazon Redshift:

- Si no se proporciona una clave de distribución, se utilizará una clave principal de la tabla de MySQL de Amazon RDS.
- Si hay claves principales compuestas en la tabla, la primera se utiliza como clave de distribución si no se ha proporcionado la clave de distribución. Solo la primera clave compuesta se establece como clave principal en la tabla de Amazon Redshift.
- Si no se proporciona una clave de distribución y no hay ninguna clave principal en la tabla MySQL de Amazon RDS, la operación de copia producirá un error.

Para obtener más información sobre Amazon Redshift, consulte los siguientes temas:

- Clúster de Amazon Redshift
- COPIA de Amazon Redshift
- Estilos de distribución y ejemplos de DISTKEY
- Claves de ordenación

En la tabla siguiente se describe cómo el script traduce los tipos de datos:

Traducciones de tipos de datos entre MySQL y Amazon Redshift

Tipo de datos de MySQL	Tipos de datos de Amazon Redshift	Notas
TINYINT, TINYINT (tamaño)	SMALLINT	MySQL: -128 a 127. Se puede especificar el número máximo de dígitos entre paréntesis. Amazon Redshift: INT2. Entero firmado de dos bytes
TINYINT UNSIGNED,	SMALLINT	MySQL: 0 a 255 UNSIGNED. Se puede especificar el

Tipo de datos de MySQL	Tipos de datos de Amazon Redshift	Notas
TINYINT (tamaño) UNSIGNED		número máximo de dígitos entre paréntesis. Amazon Redshift: INT2. Entero firmado de dos bytes
SMALLINT, SMALLINT(tamaño)	SMALLINT	MySQL: -32768 a 32767 normal. Se puede especificar el número máximo de dígitos entre paréntesis. Amazon Redshift: INT2. Entero firmado de dos bytes
SMALLINT UNSIGNED, SMALLINT(tamaño) UNSIGNED,	INTEGER	MySQL: 0 a 65535 UNSIGNED*. Se puede especificar el número máximo de dígitos entre paréntesis Amazon Redshift: INT4. Entero firmado de cuatro bytes
MEDIUMINT, MEDIUMINT(tamaño)	INTEGER	MySQL: 388608 a 8388607. Se puede especificar el número máximo de dígitos entre paréntesis Amazon Redshift: INT4. Entero firmado de cuatro bytes

Tipo de datos de MySQL	Tipos de datos de Amazon Redshift	Notas
MEDIUMINT UNSIGNED, MEDIUMINT(tamaño) UNSIGNED	INTEGER	MySQL: 0 a 16777215. Se puede especificar el número máximo de dígitos entre paréntesis Amazon Redshift: INT4. Entero firmado de cuatro bytes
INT, INT(tamaño)	INTEGER	MySQL: 147483648 a 2147483647 Amazon Redshift: INT4. Entero firmado de cuatro bytes
INT UNSIGNED, INT(tamaño) UNSIGNED	BIGINT	MySQL: 0 a 4294967295 Amazon Redshift: INT8. Entero firmado de ocho bytes
BIGINT BIGINT(tamaño)	BIGINT	Amazon Redshift: INT8. Entero firmado de ocho bytes
BIGINT UNSIGNED BIGINT(tamaño) UNSIGNED	VARCHAR(20*4)	MySQL: 0 a 184467440 73709551615 Amazon Redshift: sin equivalente nativo, se utiliza una matriz de char.

Tipo de datos de MySQL	Tipos de datos de Amazon Redshift	Notas
FLOAT FLOAT(tamaño,d) FLOAT(tamaño,d) UNSIGNED	REAL	Se puede especificar el número máximo de dígitos en el parámetro de tamaño. El número máximo de dígitos a la derecha del punto decimal se especifica en el parámetro d. Amazon Redshift: FLOAT4.
DOUBLE(tamaño,d)	DOUBLE PRECISION	Se puede especificar el número máximo de dígitos en el parámetro de tamaño. El número máximo de dígitos a la derecha del punto decimal se especifica en el parámetro d. Amazon Redshift: FLOAT8.
DECIMAL(tamaño,d)	DECIMAL(tamaño,d)	Un valor DOUBLE almacenado como una cadena, que permite un separador decimal fijo. Se puede especificar el número máximo de dígitos en el parámetro de tamaño. El número máximo de dígitos a la derecha del punto decimal se especifica en el parámetro d. Amazon Redshift: sin equivalente nativo.

Tipo de datos de MySQL	Tipos de datos de Amazon Redshift	Notas
CHAR(tamaño)	VARCHAR(tamaño*4)	Contiene una cadena de longitud fija, que puede contener letras, números y caracteres especiales. El tamaño fijo se especifica como el parámetro entre paréntesis. Puede almacenar hasta 255 caracteres. Se rellena por la derecha con espacios. Amazon Redshift: el tipo de datos CHAR no admite caracteres multibyte, así que se utiliza VARCHAR. El número máximo de bytes por carácter es de 4, según RFC3629, lo que limita la tabla de caracteres para U+10FFFF.
VARCHAR(tamaño)	VARCHAR(tamaño*4)	Puede almacenar hasta 255 caracteres. VARCHAR no admite los siguientes puntos de código UTF-8 no válidos: 0xD800 - 0xDFFF, (secuencias de bytes: ED A0 80 - ED BF BF), 0xFDD0 - 0xFDEF, 0xFFFE y 0xFFFF, (secuencias de bytes: EF B7 90 - EF B7 AF, EF BF BE y EF BF BF)

Tipo de datos de MySQL	Tipos de datos de Amazon Redshift	Notas
TINYTEXT	VARCHAR(255*4)	Contiene una cadena con una longitud máxima de 255 caracteres
TEXT	VARCHAR (máx.)	Contiene una cadena con una longitud máxima de 65 535 caracteres.
MEDIUMTEXT	VARCHAR (máx.)	De 0 a 16 777 215 caracteres
LONGTEXT	VARCHAR (máx.)	De 0 a 4 294 967 295 caracteres
BOOLEAN BOOL TINYINT(1)	BOOLEAN	MySQL: estos tipos son sinónimos de TINYINT (1). Un valor de cero se considera falso. Los valores distintos de cero se consideran verdadero.
BINARY[(M)]	varchar (255)	M es de 0 a 255 bytes, FIXED
VARBINARY(M)	VARCHAR (máx.)	De 0 a 65 535 bytes
TINYBLOB	VARCHAR (255)	De 0 a 255 bytes
BLOB	VARCHAR (máx.)	De 0 a 65 535 bytes
MEDIUMBLOB	VARCHAR (máx.)	De 0 a 16 777 215 bytes
LONGBLOB	VARCHAR (máx.)	De 0 a 4 294 967 295 bytes
ENUM	VARCHAR(255*2)	El límite no se aplica a la longitud de la cadena enum literal, sino a definición de la tabla del número de valores enum.

Tipo de datos de MySQL	Tipos de datos de Amazon Redshift	Notas
SET	VARCHAR(255*2)	Igual que enum.
DATE	DATE	(AAAA-MM-DD) "1000-01-01" a "9999-12-31"
TIME	VARCHAR(10*4)	(hh:mm:ss) "-838:59:59" a "838:59:59"
DATETIME	TIMESTAMP	(AAAA-MM-DD hh:mm:ss) 1000-01-01 00:00:00" a "9999-12-31 23:59:59"
TIMESTAMP	TIMESTAMP	(AAAAMMDDhhmmss) 19700101000000 a 2037+
YEAR	VARCHAR(4*4)	(YYYY) De 1900 a 2155

Tipo de datos de MySQL	Tipos de datos de Amazon Redshift	Notas
column SERIAL	Generación de ID/Este atributo no es necesario para un data warehouse OLAP, puesto que esta columna se copia. La palabra clave SERIAL no se agrega al traducir.	SERIAL es en realidad una entidad denominada SEQUENCE. Existe de forma independiente del resto de la tabla. column GENERATED BY DEFAULT equivalente a: CREATE SEQUENCE name; CREATE TABLE table (column INTEGER NOT NULL DEFAULT nextval(n ame));
column BIGINT UNSIGNED NOT NULL AUTO_INCR EMENT UNIQUE	Generación de ID/Este atributo no es necesario para un data warehouse OLAP, puesto que esta columna se copia. Por lo tanto, la palabra clave SERIAL no se agrega al traducir.	SERIAL es en realidad una entidad denominada SEQUENCE. Existe de forma independiente del resto de la tabla. column GENERATED BY DEFAULT equivalente a: CREATE SEQUENCE name; CREATE TABLE table (column INTEGER NOT NULL DEFAULT nextval(n ame));

Tipo de datos de MySQL	Tipos de datos de Amazon Redshift	Notas
ZEROFILL	La palabra clave ZEROFILL no se agrega al traducir.	INT UNSIGNED ZEROFILL NOT NULL ZEROFILL rellena el valor que se muestra del campo con ceros hasta el ancho de visualización especificado en la definición de columna. Los valores cuya longitud supera la de la visualización no se truncan. Tenga en cuenta que el uso de ZEROFILL también implica UNSIGNED.

Full copy of Amazon RDS MySQL table to Amazon Redshift

La plantilla Full copy of Amazon RDS MySQL table to Amazon Redshift copia toda la tabla MySQL de Amazon RDS en una tabla de Amazon Redshift mediante datos transitorios en una carpeta de Amazon S3. La carpeta de preparación de Amazon S3 debe estar en la misma región que el clúster de Amazon Redshift. Se creará una tabla de Amazon Redshift con el mismo esquema que la tabla de MySQL de Amazon RDS de origen si todavía no existe. Proporcione cualquier anulación de tipo de datos de columna MySQL de Amazon RDS a Amazon Redshift que desee aplicar durante la creación de la tabla de Amazon Redshift.

La plantilla usa los siguientes objetos de canalización:

- CopyActivity
- RedshiftCopyActivity
- S3 DataNode
- SqlDataNode
- RedshiftDataNode
- RedshiftDatabase

Incremental copy of an Amazon RDS MySQL table to Amazon Redshift

La plantilla Incremental copy of Amazon RDS MySQL table to Amazon Redshift copia datos de una tabla MySQL de Amazon RDS en una tabla de Amazon Redshift mediante datos transitorios en una carpeta de Amazon S3.

La carpeta de preparación de Amazon S3 debe estar en la misma región que el clúster de Amazon Redshift.

AWS Data Pipeline utiliza un script de traducción para crear una tabla de Amazon Redshift con el mismo esquema que la tabla de MySQL de Amazon RDS de origen si todavía no existe. Debe proporcionar cualquier anulación de tipo de datos de columna MySQL de Amazon RDS a Amazon Redshift que desee aplicar durante la creación de la tabla de Amazon Redshift.

Esta plantilla copia los cambios que se realicen en la tabla de MySQL de Amazon RDS entre intervalos programados a partir de la hora de comienzo programada. Las eliminaciones físicas realizadas en la tabla de MySQL de Amazon RDS no se copian. Debe proporcionar el nombre de la columna que almacena el valor de hora de la última modificación.

Al utilizar la plantilla predeterminada para crear canalizaciones para copia de Amazon RDS incremental, se crea una actividad con el nombre predeterminado "`RDSToS3CopyActivity`". Puede cambiar el nombre.

La plantilla usa los siguientes objetos de canalización:

- CopyActivity
- RedshiftCopyActivity
- S3 DataNode
- SqlDataNode
- RedshiftDataNode
- RedshiftDatabase

Cargar datos desde Amazon S3 en Amazon Redshift

La plantilla Load data from S3 into Redshift copia de datos de una carpeta de Amazon S3 en una tabla de Amazon Redshift. Puede cargar los datos en una tabla existente o proporcionar una consulta SQL para crear la tabla.

Los datos se copian en función de las opciones de COPY de Amazon Redshift. La tabla de Amazon Redshift debe tener el mismo esquema que los datos de Amazon S3. Para conocer las opciones de COPY, consulte COPY en la Guía de desarrollador de base de datos de Amazon Redshift.

La plantilla usa los siguientes objetos de canalización:

- CopyActivity
- RedshiftCopyActivity
- S3 DataNode
- RedshiftDataNode
- RedshiftDatabase
- Ec2Resource

Creación de una canalización mediante plantillas parametrizadas

Puede utilizar una plantilla parametrizada para personalizar una definición de canalización. Esto permite crear una definición de canalización común pero proporcionar diferentes parámetros cuando se agregue la definición de canalización a una nueva canalización.

Contenido

Añadir myVariables a la definición de la canalización

Al crear el archivo de definición de canalización, especifique las variables utilizando la siguiente sintaxis: #{my*Variable*}. Es necesario que la variable lleve el prefijo `my`. Por ejemplo, el siguiente archivo de definición de canalización, `pipeline-definition.json`, incluye las siguientes variables: *myShellCmd*, *myS3InputLoc* y *myS3OutputLoc*.

> **Note**
>
> Una definición de canalización tiene un límite superior de 50 parámetros.

```
{
  "objects": [
    {
      "id": "ShellCommandActivityObj",
      "input": {
        "ref": "S3InputLocation"
      },
      "name": "ShellCommandActivityObj",
      "runsOn": {
        "ref": "EC2ResourceObj"
      },
      "command": "#{myShellCmd}",
      "output": {
        "ref": "S3OutputLocation"
      },
      "type": "ShellCommandActivity",
      "stage": "true"
    },
    {
      "id": "Default",
      "scheduleType": "CRON",
      "failureAndRerunMode": "CASCADE",
      "schedule": {
        "ref": "Schedule_15mins"
      },
      "name": "Default",
      "role": "DataPipelineDefaultRole",
      "resourceRole": "DataPipelineDefaultResourceRole"
    },
    {
      "id": "S3InputLocation",
      "name": "S3InputLocation",
      "directoryPath": "#{myS3InputLoc}",
      "type": "S3DataNode"
    },
    {
      "id": "S3OutputLocation",
      "name": "S3OutputLocation",
      "directoryPath": "#{myS3OutputLoc}/#{format(@scheduledStartTime, 'YYYY-MM-dd-HH-
mm-ss')}",
      "type": "S3DataNode"
    },
    {
```

```
      "id": "Schedule_15mins",
      "occurrences": "4",
      "name": "Every 15 minutes",
      "startAt": "FIRST_ACTIVATION_DATE_TIME",
      "type": "Schedule",
      "period": "15 Minutes"
    },
    {
      "terminateAfter": "20 Minutes",
      "id": "EC2ResourceObj",
      "name": "EC2ResourceObj",
   "instanceType":"t1.micro",
      "type": "Ec2Resource"
    }
  ]
}
```

Definir objetos de parámetro

Puede crear un archivo independiente con objetos de parámetro que definen las variables de la definición de canalización. Por ejemplo, el siguiente archivo JSON, `parameters.json`, contiene objetos de parámetro para las variables myShellCmd, *myS3InputLoc* y *myS3OutputLoc* del ejemplo anterior de definición de canalización.

```
{
  "parameters": [
    {
      "id": "myShellCmd",
      "description": "Shell command to run",
      "type": "String",
      "default": "grep -rc \"GET\" ${INPUT1_STAGING_DIR}/* > ${OUTPUT1_STAGING_DIR}/
output.txt"
    },
    {
      "id": "myS3InputLoc",
      "description": "S3 input location",
      "type": "AWS::S3::ObjectKey",
      "default": "s3://us-east-1.elasticmapreduce.samples/pig-apache-logs/data"
    },
    {
      "id": "myS3OutputLoc",
      "description": "S3 output location",
      "type": "AWS::S3::ObjectKey"
```

```
    }
  ]
}
```

> **Note**
>
> Podría agregar estos objetos directamente al archivo de definición de canalización en lugar de utilizar un archivo independiente.

En la siguiente tabla se describen los atributos para objetos de parámetro.

Atributos de parámetro

Atributo	Tipo	Descripción
`id`	Cadena	El identificador único del parámetro. Para enmascarar el valor mientras se escribe o se muestra, agregue un asterisco ("*") como prefijo. Por ejemplo, `*myVariable` . Tenga en cuenta que esto cifra el valor antes de que lo almacene AWS Data Pipeline.
Descripción	Cadena	Una descripción del parámetro .
tipo	String, Integer, Double o AWS::S3::ObjectKey	El tipo de parámetro que define la gama permitida de valores de entrada y reglas de validación. El tipo predeterminado es String.
opcional	Booleano	Indica si el parámetro es opcional u obligatorio. El valor predeterminado es `false`.

Atributo	Tipo	Descripción
allowedValues	Lista de cadenas	Enumera todos los valores permitidos para el parámetro.
predeterminado	Cadena	El valor predeterminado para el parámetro. Si especifica un valor para este parámetro utilizando valores de parámetro, anula el valor predeterminado.
isArray	Booleano	Indica si el parámetro es una matriz.

Definir valores de parámetros

Puede crear un archivo independiente para definir las variables a través de valores de parámetros. Por ejemplo, el siguiente archivo JSON, `file://values.json`, contiene el valor de la variable *`myS3OutputLoc`* del ejemplo de definición de canalización anterior.

```
{
  "values":
    {
      "myS3OutputLoc": "myOutputLocation"
    }
}
```

Envío de la definición de canalización

Al enviar la definición de la canalización, puede especificar parámetros, objetos de parámetro y valores de parámetro. Por ejemplo, puede utilizar el comando de put-pipeline-definition de la AWS CLI como se indica a continuación:

```
$ aws datapipeline put-pipeline-definition --pipeline-id id --pipeline-definition
 file://pipeline-definition.json \
--parameter-objects file://parameters.json --parameter-values-uri file://values.json
```

> **Note**
>
> Una definición de canalización tiene un límite superior de 50 parámetros. El tamaño del archivo para `parameter-values-uri` tiene un límite superior de 15 kB.

Visualización de las canalizaciones

Puede ver sus canalizaciones con la consola o la interfaz de la línea de comandos (CLI).

Para ver las canalizaciones mediante la AWS CLI

- Utilice el siguiente comando list-pipelines para mostrar las canalizaciones:

```
aws datapipeline list-pipelines
```

Interpretación de los códigos de estado de la canalización

Los niveles de estado que se muestran en la CLI y la consola de AWS Data Pipeline indican el estado de una canalización y de sus componentes. El estado de la canalización es simplemente una visión general de una canalización; para ver más información, consulte el estado de los componentes de canalización individuales.

Una canalización tiene un estado `SCHEDULED` si está lista (la definición de canalización ha superado la validación), en funcionamiento o ha terminado la realización de un trabajo. Una canalización tiene un estado `PENDING` si no está activada o no puede realizar trabajo (por ejemplo, la definición de canalización no ha superado la validación).

Una canalización se considera inactiva si su estado es `PENDING` `INACTIVE` o `FINISHED`. Las canalizaciones inactivas incurren en cargos (para obtener más información, consulte Precios).

Códigos de estado

`ACTIVATING`

Se está iniciando el componente o recurso, como una instancia EC2.

`CANCELED`

El componente fue cancelado por un usuario o AWS Data Pipeline antes de que pudiera ejecutarse. Esto puede ocurrir automáticamente cuando se produce un error en un componente o recurso diferente del que depende este componente.

`CASCADE_FAILED`

El componente o recurso se canceló como resultado de un error en cascada en una de sus dependencias, pero es probable que el componente no fuera la fuente original del error.

`DEACTIVATING`

Se está desactivando la canalización.

`FAILED`

El componente o recurso ha detectado un error y ha dejado de funcionar. Cuando se produce un error en un componente o recurso, las cancelaciones y los errores pueden repercutir en cascada en otros componentes que dependen de él.

`FINISHED`

El componente completó el trabajo que se le había asignado.

`INACTIVE`

Se desactivó la canalización.

`PAUSED`

El componente estaba en pausa y no está realizando su trabajo en este momento.

`PENDING`

La canalización está lista para activarse por primera vez.

`RUNNING`

El recurso está en ejecución y listo para recibir trabajo.

`SCHEDULED`

El recurso está programado para ejecutarse.

`SHUTTING_DOWN`

El recurso se cierra después de completar correctamente su trabajo.

`SKIPPED`

El componente omitió los intervalos de ejecución tras la activación de la canalización mediante una marca de tiempo posterior a la programación actual.

`TIMEDOUT`

El recurso superó el umbral de `terminateAfter` y fue detenido por AWS Data Pipeline. Cuando el recurso alcanza este estado, AWS Data Pipeline ignora los valores `actionOnResourceFailure`, `retryDelay` y`retryTimeout` de ese recurso. Este estado solo se aplica a los recursos.

`VALIDATING`

La definición de canalización está siendo validada por AWS Data Pipeline.

`WAITING_FOR_RUNNER`

El componente está esperando a que su cliente trabajador recupere un elemento de trabajo. La relación entre el componente y el cliente del trabajador se controla mediante los campos `runsOn` o `workerGroup` definidos por ese componente.

`WAITING_ON_DEPENDENCIES`

El componente comprueba que se cumplen sus condiciones previas predeterminadas y configuradas por el usuario antes de realizar su trabajo.

Interpretación del estado de salud de canalizaciones y componentes

Cada canalización y cada componente dentro de dicha canalización devuelve un estado de salud de `HEALTHY`, `ERROR`, `"-"`, `No Completed Executions` o `No Health Information Available`. Una canalización solo tiene un estado de salud después de que un componente de la canalización haya completado su primera ejecución o las condiciones previas del componente hayan producido un error. El estado de salud de los componentes se agrega en un estado salud de la canalización en el que los estados de error son visibles primero cuando se ven los detalles de la ejecución de la canalización.

Estados de salud de canalización

`HEALTHY`

El estado de salud agregado de todos los componentes es `HEALTHY`. Esto significa que al menos un componente debe haberse completado con éxito. Puede hacer clic en el estado `HEALTHY`

para ver la instancia de componente de canalización completada más recientemente en la página Execution Details.

`ERROR`

Al menos un componente de la canalización tiene un estado de salud de `ERROR`. Puede hacer clic en el estado `ERROR` para ver la instancia de componente de canalización que haya producido un error más recientemente en la página Execution Details.

`No Completed Executions` o bien `No Health Information Available`.

No se notificó ningún estado de salud para esta canalización.

> **Note**
>
> Aunque los componentes actualizan su estado de salud casi de inmediato, el estado de salud de una canalización puede tardar hasta cinco minutos en actualizarse.

Estados de salud de componentes

`HEALTHY`

Un componente (`Activity` o `DataNode`) tiene un estado de salud de `HEALTHY` si ha completado una ejecución que se haya marcado con el estado `FINISHED` o `MARK_FINISHED`. Puede hacer clic en el nombre del componente o en el estado `HEALTHY` para ver las instancias de componentes de canalización completadas más recientemente en la página Execution Details.

`ERROR`

Se ha producido un error en el nivel de componente o una de las condiciones previas ha producido un error. Los estados `FAILED`, `TIMEOUT` y `CANCELED` disparan este error. Puede hacer clic en el nombre del componente o en el estado `ERROR` para ver la instancia de componente de canalización que haya producido un error más recientemente en la página Execution Details.

`No Completed Executions` or `No Health Information Available`

No se notificó ningún estado de salud para este componente.

Visualización de las definiciones de canalización

Utilice la interfaz de la línea de comandos (CLI) para ver la definición de la canalización. La CLI imprime un archivo de definición de canalización en formato JSON. Para obtener información acerca de la sintaxis y el uso de los archivos de definición de canalización, consulte Sintaxis de los archivos de definición de la canalización.

Cuando se utiliza la CLI, es buena idea recuperar la definición de la canalización antes de enviar modificaciones, ya que es posible que otro usuario o proceso haya cambiado la definición de la canalización después de la última vez que usted trabajara en ella. Puede descargar una copia de la definición actual y utilizarla como base para sus modificaciones. Así, se asegurará de trabajar con la definición de canalización más reciente. También es buena idea volver a recuperar la definición de la canalización después de modificarla, para asegurarse de que la actualización se haya realizado correctamente.

Si utiliza la CLI, puede obtener dos versiones diferentes de la canalización. La versión `active` es la canalización que se está ejecutando en este momento. La versión `latest` es una copia que crea al editar una canalización en funcionamiento. Cuando se carga la canalización editada, pasa a ser la versión `active` y la versión `active` anterior deja de estar disponible.

Para obtener una definición de canalización mediante la AWS CLI

Para obtener la definición completa de la canalización, utilice el siguiente comando get-pipeline-definition. La definición de la canalización se imprime en la salida estándar (stdout).

En el ejemplo siguiente se obtiene la definición de canalización para la canalización especificada.

```
aws datapipeline get-pipeline-definition --pipeline-id df-00627471SOVYZEXAMPLE
```

Para recuperar una versión específica de una canalización, utilice la opción `--version`. En el siguiente ejemplo se recupera la versión `active` de la canalización especificada.

```
aws datapipeline get-pipeline-definition --version active --id df-00627471SOVYZEXAMPLE
```

Visualización de detalles de instancias de canalización

Puede monitorizar el progreso de la canalización. Para obtener más información acerca del estado de la instancia, consulte Interpretación de los detalles de estado de la canalización. Para obtener más información sobre cómo solucionar problemas de ejecuciones de instancias de la canalización, consulte Resolución de problemas comunes.

Para monitorizar el progreso de una canalización mediante la AWS CLI

Para recuperar detalles de la instancia de la canalización, tal como un historial de las veces que se ha ejecutado una canalización, utilice el comando list-runs. Este comando permite filtrar la lista de ejecuciones devueltas en función de su estado actual o del intervalo de fechas en que se lanzaron. Filtrar los resultados es útil porque, dependiendo de la edad y la programación de la canalización, el historial de ejecuciones puede ser grande.

El siguiente ejemplo recupera información de todas las ejecuciones.

```
aws datapipeline list-runs --pipeline-id df-00627471SOVYZEXAMPLE
```

El siguiente ejemplo recupera información de todas las ejecuciones completadas.

```
aws datapipeline list-runs --pipeline-id df-00627471SOVYZEXAMPLE --status finished
```

El siguiente ejemplo recupera información de todas las ejecuciones lanzadas en el período especificado.

```
aws datapipeline list-runs --pipeline-id df-00627471SOVYZEXAMPLE --start-interval
 "2013-09-02","2013-09-11"
```

Visualización de registros de canalización

El registro de nivel de canalización se admite en el momento de crear la canalización si se especifica una ubicación de Amazon S3 en la consola o mediante un valor de `pipelineLogUri` en el objeto predeterminado en el SDK o la CLI. La estructura de directorios para cada canalización dentro de dicho URI es como la siguiente:

```
pipelineId
    -componentName
        -instanceId
            -attemptId
```

Para la canalización, `df-00123456ABC7DEF8HIJK`, la estructura del directorio tiene este aspecto:

```
df-00123456ABC7DEF8HIJK
    -ActivityId_fXNzc
        -@ActivityId_fXNzc_2014-05-01T00:00:00
```

```
            -@ActivityId_fXNzc_2014-05-01T00:00:00_Attempt=1
```

En el caso de `ShellCommandActivity`, en cada intento se almacenan en el directorio los registros `stderr` y `stdout` asociados a estas actividades.

Para los recursos como `EmrCluster`, donde se establece un valor de `emrLogUri`, ese valor tiene precedencia. De lo contrario, los recursos (incluidos los registros de TaskRunner de esos recursos) siguen la estructura de registro de la canalización anterior.

Para ver los registros de una ejecución de canalización determinada:

1. Recupere el `ObjectId` llamando a `query-objects` para obtener el ID exacto del objeto. Por ejemplo:

   ```
   aws datapipeline query-objects --pipeline-id <pipeline-id> --sphere ATTEMPT --region
    ap-northeast-1
   ```

   `query-objects` es una CLI paginada y puede devolver un token de paginación si hay más ejecuciones para un `pipeline-id` determinado. Puede usar el token para realizar todos los intentos hasta encontrar el objeto esperado. Por ejemplo, un ObjectID devuelto tendría el siguiente aspecto: `@TableBackupActivity_2023-05-020T18:05:18_Attempt=1`
2. Con ObjectID, recupere la ubicación del registro mediante:

   ```
   aws datapipeline describe-objects –pipeline-id <pipeline-id> --object-ids <object-id>
    --query "pipelineObjects[].fields[?key=='@logLocation'].stringValue"
   ```

Mensaje de error de una actividad fallida

Para obtener el mensaje de error, primero obtenga el ObjectID utilizando `query-objects`.

Tras recuperar el ObjectID erróneo, utilice la CLI `describe-objects` para obtener el mensaje de error real.

```
aws datapipeline describe-objects --region ap-northeast-1 --pipeline-id
 <pipeline-id> --object-ids <object-id> --query "pipelineObjects[].fields[?
key=='errorMessage'].stringValue"
```

Cancelar, volver a ejecutar o marcar un objeto como terminado

Utilice la CLI `set-status` para cancelar un objeto en ejecución, volver a ejecutar un objeto fallido o marcar un objeto en ejecución como Finalizado.

Primero, obtenga el ID del objeto mediante la CLI `query-objects`. Por ejemplo:

```
aws datapipeline query-objects --pipeline-id <pipeline-id> --sphere INSTANCE --region
 ap-northeast-1
```

Utilice la CLI `set-status` para cambiar el estado del objeto deseado. Por ejemplo:

```
aws datapipeline set-status –pipeline-id <pipeline-id> --region ap-northeast-1 --status
 TRY_CANCEL --object-ids <object-id>
```

Edición de la canalización

Para cambiar algún aspecto de una de las canalizaciones, puede actualizar su definición de canalización. Después de modificar una canalización que se esté ejecutando, debe reactivar la canalización para que se apliquen los cambios. Además, puede volver a ejecutar uno o varios componentes de la canalización.

Contenido

Limitaciones

Mientras la canalización se encuentre en el estado `PENDING` y no esté activada, podrá hacer cambios en ella. Después de activar una canalización, puede editarla con las siguientes restricciones. Los cambios que realice se aplicarán a las nuevas ejecuciones de los objetos de la canalización después de guardarlos y, a continuación, volver a activar la canalización.

- No se puede eliminar un objeto
- No se puede cambiar el período de programación de un objeto existente
- No se pueden agregar, eliminar o modificar campos de referencia de un objeto existente
- No se puede hacer referencia a un objeto existente en un campo de salida de un objeto nuevo
- No se puede cambiar la fecha de inicio programada de un objeto (en su lugar, active la canalización con una fecha y hora determinadas)

Edición de una canalización mediante la AWS CLI

Puede editar una canalización mediante las herramientas de la línea de comandos.

En primer lugar, descargue una copia de la definición de la canalización actual mediante el comando get-pipeline-definition. Al hacerlo, puede asegurarse de estar modificando la definición de canalización más reciente. El siguiente ejemplo imprime la definición de la canalización en la salida estándar (stdout).

```
aws datapipeline get-pipeline-definition --pipeline-id df-00627471SOVYZEXAMPLE
```

Guarde la definición de la canalización en un archivo y edítela como sea necesario. Actualice la definición de la canalización con el comando put-pipeline-definition. En el siguiente ejemplo se carga el archivo de definición de canalización actualizado.

```
aws datapipeline put-pipeline-definition --pipeline-id df-00627471SOVYZEXAMPLE --
pipeline-definition file://MyEmrPipelineDefinition.json
```

Puede recuperar la definición de la canalización de nuevo con el comando `get-pipeline-definition` para asegurarse de que la actualización se ha realizado correctamente. Para activar la canalización, utilice el siguiente comando activate-pipeline:

```
aws datapipeline activate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE
```

Si lo prefiere, puede activar la canalización a partir de una fecha y hora determinadas, utilizando la opción `--start-timestamp` de la siguiente manera:

```
aws datapipeline activate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE --start-
timestamp YYYY-MM-DDTHH:MM:SSZ
```

Para volver a ejecutar uno o más componentes de la canalización, utilice el comando set-status.

Clonación de la canalización

La clonación crea una copia de una canalización y permite especificar un nombre para la nueva canalización. Puede clonar una canalización que se encuentre en cualquier estado, incluso si tiene errores; sin embargo, la nueva canalización permanecerá en el estado `PENDING` hasta que la active manualmente. Para la nueva canalización, la operación de clonación utiliza la versión más reciente

de la definición de canalización original, en lugar de la versión activa. En la operación de clonación, no se copia en la nueva canalización el programa completo de la canalización original, sino solo el ajuste de período.

Para clonar una canalización mediante la CLI AWS:

1. Cree una nueva canalización con un nombre nuevo y un identificador único. Anotar el ID de canalización devuelto.
2. Utilice la CLI `get-pipeline-definition` para obtener la definición de canalización de la canalización existente que se va a clonar y escríbala en un archivo temporal. Tenga en cuenta la ruta absoluta del archivo.
3. Utilice la CLI `put-pipeline-definition` para copiar la definición de canalización de la canalización existente a la nueva canalización.
4. Utilice la CLI `get-pipeline-definition` para obtener la definición de la nueva canalización y verificar la definición de la canalización.

```
# Create Pipeline (returns <new-pipeline-id>)
aws datapipeline create-pipeline --name my-cloned-pipeline --unique-id my-cloned-
pipeline --region ap-northeast-1

#Get pipeline definition of existing pipeline
aws datapipeline get-pipeline-definition --pipeline-id <existing-pipeline-id> --
region ap-northeast-1 > existing_pipeline_definition.json

# Put pipeline definition to new pipeline
aws datapipeline put-pipeline-definition --pipeline-id <new-
pipeline-id> --region ap-northeast-1 --pipeline-definition file://
<absolute_path_to_existing_pipeline_definition.json>

# get pipeline definition of new pipeline
aws datapipeline get-pipeline-definition --pipeline-id <new-pipeline-id> --region
 ap-northeast-1
```

Etiquetado de la canalización

Las etiquetas son pares de clave/valor que distinguen entre mayúsculas y minúsculas y constan de una clave y un valor opcional, ambos definidos por el usuario. Puede aplicar hasta diez etiquetas a cada canalización. Las claves de las etiquetas deben ser únicas para cada canalización. Si agrega

una etiqueta con una clave que ya está asociada a la canalización, se actualizará el valor de esa etiqueta.

La aplicación de una etiqueta a una canalización también propaga las etiquetas a sus recursos subyacentes (por ejemplo, clústeres de Amazon EMR e instancias Amazon EC2). Sin embargo, no aplica estas etiquetas a los recursos que se encuentren en estado `FINISHED` u otro estado terminado. Puede utilizar la CLI para aplicar etiquetas a estos recursos, si es necesario.

Cuando haya terminado de utilizar una etiqueta, puede eliminarla de la canalización.

Para etiquetar la canalización mediante la CLI de AWS

Para agregar etiquetas a una nueva canalización, agregue la opción `--tags` al comando create-pipeline. Por ejemplo, la siguiente opción crea una canalización con dos etiquetas, una etiqueta `environment` con un valor de `production` y una etiqueta `owner` con un valor de `sales`.

```
--tags key=environment,value=production key=owner,value=sales
```

Para agregar etiquetas a una canalización existente, utilice el comando add-tags como se indica a continuación:

```
aws datapipeline add-tags --pipeline-id df-00627471SOVYZEXAMPLE --tags
 key=environment,value=production key=owner,value=sales
```

Para quitar etiquetas de una canalización existente, utilice el comando remove-tags como se indica a continuación:

```
aws datapipeline remove-tags --pipeline-id df-00627471SOVYZEXAMPLE --tag-keys
 environment owner
```

Desactivación de la canalización

La desactivación de una canalización en ejecución detiene la ejecución de la canalización. Para reanudar la ejecución de la canalización, puede activar la canalización. Esto le permitirá hacer cambios. Por ejemplo, si está escribiendo datos en una base de datos programada para someterse a mantenimiento, puede desactivar la canalización, esperar a que se complete el mantenimiento y, a continuación, activar la canalización.

Cuando desactive una canalización, puede especificar lo que ocurre con las actividades en ejecución. De forma predeterminada, estas actividades se cancelarán inmediatamente. También puede hacer que AWS Data Pipeline espere hasta que las actividades finalicen antes de desactivar la canalización.

Cuando active una canalización desactivada, puede especificar cuándo se reanudará. Con la AWS CLI o la API, la canalización se reanuda de manera predeterminada a partir de la última ejecución completada. También es posible especificar la fecha y la hora en la que se debe reanudar.

Desactivación de la canalización mediante la AWS CLI

Utilice el siguiente comando deactivate-pipeline para desactivar una canalización:

```
aws datapipeline deactivate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE
```

Para desactivar la canalización después de que hayan finalizado todas las actividades de ejecución, agregue la opción `--no-cancel-active`, como se indica a continuación:

```
aws datapipeline deactivate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE --no-cancel-
active
```

Cuando esté listo, puede reanudar la ejecución de la canalización donde se haya quedado mediante el siguiente comando activate-pipeline:

```
aws datapipeline activate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE
```

Para iniciar la canalización a partir de una fecha y una hora determinadas, agregue la opción `--start-timestamp`, como se indica a continuación:

```
aws datapipeline activate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE --start-
timestamp YYYY-MM-DDTHH:MM:SSZ
```

Eliminación de la canalización

Si ya no necesita una canalización, tal como una canalización creada durante las pruebas de una aplicación, es recomendable que la elimine del uso activo. La eliminación de una canalización

la coloca en un estado de eliminación. Cuando la canalización está en el estado eliminado, su definición de canalización y su historial de ejecuciones desaparecen. Por lo tanto, ya no es posible realizar operaciones en la canalización, incluida su descripción.

> **Important**
>
> No se puede restaurar una canalización después de eliminarla, así que antes de eliminar una canalización debe asegurarse de que no la necesitará en el futuro.

Para eliminar una canalización mediante la AWS CLI

Para eliminar una canalización, utilice el comando delete-pipeline. El siguiente comando elimina la canalización especificada.

```
aws datapipeline delete-pipeline --pipeline-id df-00627471SOVYZEXAMPLE
```

Datos y tablas transitorios con actividades de canalización

AWS Data Pipeline puede utilizar datos transitorios de entrada y salida en las canalizaciones para facilitar el uso de determinadas actividades, tales como `ShellCommandActivity` y `HiveActivity`.

El uso transitorio de datos le permite copiar datos del nodo de datos de entrada en el recurso donde se ejecuta la actividad y, de manera similar, desde el recurso en el nodo de datos de salida.

Los datos transitorios del recurso de Amazon EMR o Amazon EC2 están disponibles mediante el uso de variables especiales en los comandos de intérprete de comandos o scripts de Hive de la actividad.

El uso de tablas transitorias es similar al de datos transitorios, excepto en que los datos almacenados adoptan la forma de tablas almacenadas, específicamente.

AWS Data Pipeline admite los escenarios siguientes:

- Uso transitorio de datos con `ShellCommandActivity`
- Tablas transitorias con nodos de datos compatibles con el uso de datos transitorios y Hive
- Tablas transitorias con nodos de datos no compatibles con el uso de datos transitorios y Hive

> **Note**
>
> Funciones solo transitorias cuando el campo `stage` esté establecido en `true` en una actividad, como `ShellCommandActivity`. Para obtener más información, consulte ShellCommandActivity.

Además, las actividades y los nodos de datos se pueden relacionar de cuatro maneras:

Almacenamiento local de los datos en un recurso

Los datos de entrada se copian automáticamente en el sistema de archivos local del recurso. Los datos de salida se copian automáticamente del sistema de archivos local del recurso en el nodo de datos de salida. Por ejemplo, cuando se configuran entradas y salidas `ShellCommandActivity` con staging = true, los datos de entrada están disponibles como INPUTx_STAGING_DIR y los datos de salida están disponibles como OUTPUTx_STAGING_DIR, donde x es el número de entrada o salida.

Almacenamiento transitorio de definiciones de entrada y salida para una actividad

El formato de datos de entrada (nombres de columna y nombres de tabla) se copia automáticamente en el recurso de la actividad. Por ejemplo, cuando se configura `HiveActivity` con staging = true. El formato de datos especificado en la entrada `S3DataNode` se utiliza para el uso transitorio de la definición de tabla de la tabla Hive.

Uso transitorio no habilitado

Los objetos de entrada y salida y sus campos están disponibles para la actividad, pero no los propios datos. Por ejemplo, `EmrActivity` de forma predeterminada o al configurar otras actividades con staging = false. En esta configuración, los campos de datos están disponibles para que la actividad pueda hacer referencia a ellos mediante la sintaxis de expresión de AWS Data Pipeline, pero esto solo ocurre cuando se satisface la dependencia. Esto sirve solamente como comprobación de dependencias. El código de la actividad es responsable de copiar los datos de la entrada en el recurso que ejecuta la actividad.

Relación de dependencia entre objetos

Existe una relación de dependencia entre dos objetos, que produce como resultado una situación similar a cuando no se ha habilitado el uso transitorio. Esto provoca que un nodo de datos o una actividad actúen como una condición previa para la ejecución de otra actividad.

Uso transitorio de datos con ShellCommandActivity

Considere un escenario en que se utilice `ShellCommandActivity` con objetos `S3DataNode` como entrada y salida de datos. AWS Data Pipeline almacena automáticamente los nodos de datos de manera transitoria para ponerlos a disposición del comando de shell como si fueran carpetas de archivos locales mediante las variables de entorno `${INPUT1_STAGING_DIR}` y `${OUTPUT1_STAGING_DIR}`, tal y como se muestra en el siguiente ejemplo. La parte numérica de las variables denominadas `INPUT1_STAGING_DIR` y `OUTPUT1_STAGING_DIR` se incrementa en función del número de nodos de datos a los que hace referencia la actividad.

> **Note**
>
> Esta situación solo funciona como se describe si las entradas y salidas de datos son objetos `S3DataNode`. Además, el uso transitorio de datos de salida solamente se permite cuando `directoryPath` se establece en el objeto `S3DataNode` de salida.

```
{
  "id": "AggregateFiles",
  "type": "ShellCommandActivity",
  "stage": "true",
  "command": "cat ${INPUT1_STAGING_DIR}/part* > ${OUTPUT1_STAGING_DIR}/aggregated.csv",
  "input": {
    "ref": "MyInputData"
  },
  "output": {
    "ref": "MyOutputData"
  }
},
{
  "id": "MyInputData",
  "type": "S3DataNode",
  "schedule": {
    "ref": "MySchedule"
  },
  "filePath": "s3://my_bucket/source/#{format(@scheduledStartTime,'YYYY-MM-
dd_HHmmss')}/items"
  }
},
{
  "id": "MyOutputData",
```

```
  "type": "S3DataNode",
  "schedule": {
    "ref": "MySchedule"
  },
  "directoryPath": "s3://my_bucket/destination/#{format(@scheduledStartTime,'YYYY-MM-
dd_HHmmss')}"
  }
},
...
```

Tablas transitorias con nodos de datos compatibles con el uso de datos transitorios y Hive

Considere un escenario en que se utilice `HiveActivity` con objetos `S3DataNode` como entrada y salida de datos. AWS Data Pipeline almacena automáticamente los nodos de datos de manera transitoria para ponerlos a disposición del script de Hive como si fueran tablas de Hive mediante las variables de `${input1}` y `${output1}`, tal y como se muestra en el siguiente ejemplo para `HiveActivity`. La parte numérica de las variables denominadas `input` y `output` se incrementa en función del número de nodos de datos a los que hace referencia la actividad.

> **Note**
>
> Esta situación solo funciona como se describe si las entradas y salidas de datos son objetos `S3DataNode` o `MySqlDataNode`. El uso transitorio de tablas no se admite para `DynamoDBDataNode`.

```
{
  "id": "MyHiveActivity",
  "type": "HiveActivity",
  "schedule": {
    "ref": "MySchedule"
  },
  "runsOn": {
    "ref": "MyEmrResource"
  },
  "input": {
    "ref": "MyInputData"
  },
  "output": {
```

```
    "ref": "MyOutputData"
  },
  "hiveScript": "INSERT OVERWRITE TABLE ${output1} select * from ${input1};"
},
{
  "id": "MyInputData",
  "type": "S3DataNode",
  "schedule": {
    "ref": "MySchedule"
  },
  "directoryPath": "s3://test-hive/input"
  }
},
{
  "id": "MyOutputData",
  "type": "S3DataNode",
  "schedule": {
    "ref": "MySchedule"
  },
  "directoryPath": "s3://test-hive/output"
  }
},
...
```

Tablas transitorias con nodos de datos no compatibles con el uso de datos transitorios y Hive

Considere un escenario en que se utilice `HiveActivity` con objetos `DynamoDBDataNode` como entrada y un objeto `S3DataNode` como salida de datos. No está disponible ningún uso transitorio de datos para `DynamoDBDataNode`, por lo que primero debe crear manualmente la tabla dentro del script de Hive utilizando el nombre de variable "`#{input.tableName}`" para hacer referencia a la tabla de DynamoDB. Se aplica una nomenclatura similar si la tabla de DynamoDB es la salida, excepto si se usa la variable `#{output.tableName}`. El uso transitorio está disponible para el objeto `S3DataNode` en este ejemplo, así que se puede hacer referencia al nodo de datos de salida como `${output1}`.

> **Note**
>
> En este ejemplo, la variable del nombre de la tabla tiene como prefijo el carácter # (hash) porque AWS Data Pipeline utiliza expresiones para obtener acceso a `tableName` o

`directoryPath`. Para obtener más información acerca de cómo funciona la evaluación de expresiones en AWS Data Pipeline, consulte Evaluación de expresiones.

```
{
  "id": "MyHiveActivity",
  "type": "HiveActivity",
  "schedule": {
    "ref": "MySchedule"
  },
  "runsOn": {
    "ref": "MyEmrResource"
  },
  "input": {
    "ref": "MyDynamoData"
  },
  "output": {
    "ref": "MyS3Data"
  },
  "hiveScript": "-- Map DynamoDB Table
SET dynamodb.endpoint=dynamodb.us-east-1.amazonaws.com;
SET dynamodb.throughput.read.percent = 0.5;
CREATE EXTERNAL TABLE dynamodb_table (item map<string,string>)
STORED BY 'org.apache.hadoop.hive.dynamodb.DynamoDBStorageHandler'
TBLPROPERTIES ("dynamodb.table.name" = "#{input.tableName}");
INSERT OVERWRITE TABLE ${output1} SELECT * FROM dynamodb_table;"
},
{
  "id": "MyDynamoData",
  "type": "DynamoDBDataNode",
  "schedule": {
    "ref": "MySchedule"
  },
  "tableName": "MyDDBTable"
},
{
  "id": "MyS3Data",
  "type": "S3DataNode",
  "schedule": {
    "ref": "MySchedule"
  },
  "directoryPath": "s3://test-hive/output"
  }
```

```
},
...
```

Uso de una canalización con recursos en varias regiones

De forma predeterminada, los recursos `Ec2Resource` y `EmrCluster` se ejecutan en la misma región que AWS Data Pipeline, pero AWS Data Pipeline dispone de la capacidad de organizar flujos de datos entre varias regiones, tales como la ejecución de recursos de una región que consolide datos de entrada de otra región. Al permitir que los recursos se ejecuten una región determinada, también dispone de flexibilidad para coubicar los recursos con sus conjuntos de datos dependientes y maximizar el rendimiento, reduciendo las latencias y evitando cargos por transferencia de datos entre regiones. Puede configurar recursos para ejecutarlos en una región diferente de AWS Data Pipeline mediante el campo `region` de `Ec2Resource` y `EmrCluster`.

El siguiente archivo JSON de canalización de ejemplo muestra cómo ejecutar un recurso `EmrCluster` en la región Europa (Irlanda), suponiendo que exista en la misma región una gran cantidad de datos para el clúster en el que se va a trabajar. En este ejemplo, la única diferencia con una canalización típica es que el valor del campo `EmrCluster` de `region` está establecido en `eu-west-1`.

```
{
  "objects": [
    {
      "id": "Hourly",
      "type": "Schedule",
      "startDateTime": "2014-11-19T07:48:00",
      "endDateTime": "2014-11-21T07:48:00",
      "period": "1 hours"
    },
    {
      "id": "MyCluster",
      "type": "EmrCluster",
      "masterInstanceType": "m3.medium",
      "region": "eu-west-1",
      "schedule": {
        "ref": "Hourly"
      }
    },
    {
      "id": "MyEmrActivity",
```

```
      "type": "EmrActivity",
      "schedule": {
        "ref": "Hourly"
      },
      "runsOn": {
        "ref": "MyCluster"
      },
      "step": "/home/hadoop/contrib/streaming/hadoop-streaming.jar,-input,s3n://
elasticmapreduce/samples/wordcount/input,-output,s3://eu-west-1-bucket/wordcount/
output/#{@scheduledStartTime},-mapper,s3n://elasticmapreduce/samples/wordcount/
wordSplitter.py,-reducer,aggregate"
    }
  ]
}
```

En la tabla siguiente se muestran las regiones que puede elegir y los códigos de región asociados que debe utilizar en el campo `region`.

> **Note**
>
> La siguiente lista incluye regiones en las que AWS Data Pipeline puede orquestar flujos de trabajo y lanzar recursos de Amazon EMR o Amazon EC2. Es posible que AWS Data Pipeline no sea compatible en estas regiones. Para obtener más información sobre las regiones en las que se admite AWS Data Pipeline, consulte Regiones y puntos de enlace de AWS.

Nombre de la región	Código de región
Este de EE. UU. (Norte de Virginia)	us-east-1
Este de EE. UU. (Ohio)	us-east-2
Oeste de EE. UU. (Norte de California)	us-west-1
Oeste de EE. UU. (Oregón)	us-west-2
Canadá (centro)	ca-central-1
Europa (Irlanda)	eu-west-1

Nombre de la región	Código de región
Europa (Londres)	eu-west-2
Europa (Fráncfort)	eu-central-1
Asia-Pacífico (Singapur)	ap-southeast-1
Asia-Pacífico (Sídney)	ap-southeast-2
Asia-Pacífico (Bombay)	ap-south-1
Asia-Pacífico (Tokio)	ap-northeast-1
Asia-Pacífico (Seúl)	ap-northeast-2
América del Sur (São Paulo)	sa-east-1

Errores en cascada y repeticiones de ejecuciones

AWS Data Pipeline permite configurar la manera en la que se comportan los objetos de canalización cuando una dependencia produce un error o un usuario la cancela. Puede asegurarse de que los errores se propaguen en cascada a otros objetos de canalización (consumidores), para evitar esperas indefinidas. Todas las actividades, nodos de datos y condiciones previas tiene un campo llamado `failureAndRerunMode` con un valor predeterminado de `none`. Para habilitar los errores en cascada, establezca el campo `failureAndRerunMode` en `cascade`.

Cuando este campo está habilitado, se producen errores en cascada si un objeto de canalización está bloqueado en el estado `WAITING_ON_DEPENDENCIES` y las dependencias han producido un error sin ningún comando pendiente. Durante un error en cascada, se producen los eventos siguientes:

- Cuando un objeto produce un error, sus consumidores se establecen en `CASCADE_FAILED` y las condiciones previas del objeto original y de sus consumidores se establecen en `CANCELED`.
- Los objetos cuyo estado sea ya `FINISHED`, `FAILED` o `CANCELED` no se tienen en cuenta.

El error en cascada no funciona en dependencias de objetos con errores (ascendentes) excepto para condiciones previas asociadas con el objeto con errores original. Los objetos de canalización

afectados por un error en cascada pueden disparar reintentos o acciones posteriores tales como `onFail`.

Los efectos detallados de un error en cascada dependen del tipo de objeto.

Actividades

Una actividad cambia a `CASCADE_FAILED` si cualquiera de sus dependencias produce un error y, posteriormente, desencadena un error en cascada en los consumidores de la actividad. Si se produce un error en un recurso del que depende una actividad, la actividad adopta el estado `CANCELED` y todos sus consumidores cambian a `CASCADE_FAILED`.

Nodos de datos y condiciones previas

Si se configura un nodo de datos como salida de una actividad que produce un error, el nodo de datos cambia al estado `CASCADE_FAILED`. El error de un nodo de datos se propaga a las condiciones previas asociadas, que cambian al estado `CANCELED`.

Recursos

Si los objetos que dependen de un recurso se encuentran en el estado `FAILED` y el propio recurso está en el estado `WAITING_ON_DEPENDENCIES`, el recurso cambia al estado `FINISHED`.

Volver a ejecutar objetos con errores en cascada

De forma predeterminada, volver a ejecutar cualquier actividad o nodo de datos solo vuelve a ejecutar el recurso asociado. Sin embargo, ajustar el campo `failureAndRerunMode` en `cascade` en un objeto de canalización permite volver a ejecutar un comando en un objeto de destino para propagarlo a todos los consumidores, en las siguientes condiciones:

- Los consumidores del objeto de destino están en el estado `CASCADE_FAILED`.
- Las dependencias del objeto de destino no tienen comandos pendientes de volver a ejecutarse.
- Las dependencias del objeto de destino no están en el estado `FAILED`, `CASCADE_FAILED` ni `CANCELED`.

Si trata de volver a ejecutar un objeto `CASCADE_FAILED` y cualquiera de sus dependencias está en estado `FAILED`, `CASCADE_FAILED` o `CANCELED`, la nueva ejecución producirá un error y devolverá el objeto al estado `CASCADE_FAILED`. Para volver a ejecutar correctamente el objeto que ha producido un error, debe seguir el error en sentido ascendente a lo largo de la cadena de

dependencia para encontrar el origen del error y volver a ejecutar ese objeto en su lugar. Cuando se emite un comando de nueva ejecución en un recurso, también se intenta volver a ejecutar los objetos que dependen de él.

Error en cascada y reposiciones

Si habilita el error en cascada y tiene una canalización que provoca muchas reposiciones, los errores de tiempo de ejecución de canalización pueden provocar la creación y eliminación de recursos en rápida sucesión sin realizar ningún trabajo útil. AWS Data Pipeline trata de alertarle sobre esta situación con el siguiente mensaje de advertencia cuando se guarda una canalización: *Pipeline_object_name* `has 'failureAndRerunMode' field set to 'cascade' and you are about to create a backfill with scheduleStartTime` *start_time*`. This can result in rapid creation of pipeline objects in case of failures.` Esto se debe a que un error en cascada puede establecer rápidamente las actividades en sentido descendente como `CASCADE_FAILED` y cerrar clústeres de EMR y recursos de EC2 que ya no se necesitan. Le recomendamos que pruebe las canalizaciones con intervalos de tiempo cortos para limitar los efectos de esta situación.

Sintaxis de los archivos de definición de la canalización

Las instrucciones de esta sección son para trabajar manualmente con archivos de definición de canalización mediante la interfaz de línea de comandos (CLI) de AWS Data Pipeline. Se trata de una alternativa al diseño interactivo de una canalización mediante la consola de AWS Data Pipeline.

Puede crear manualmente archivos de definición de canalización con cualquier editor de texto que permita guardar archivos con el formato de archivo UTF-8 y enviarlos a través de la interfaz de línea de comandos de AWS Data Pipeline.

AWS Data Pipeline también es compatible con diversas funciones y expresiones complejas de las definiciones de canalización. Para obtener más información, consulte Expresiones y funciones de canalizaciones.

Estructura de archivos

El primer paso en la creación de canalizaciones es componer objetos de definición de canalización en un archivo de definición de canalización. El siguiente ejemplo ilustra la estructura general de un archivo de definición de canalización. Este archivo define dos objetos, delimitados por "{" y "}" y separados por una coma.

En el siguiente ejemplo, el primer objeto define dos pares de nombre-valor, conocidos como campos. El segundo objeto define tres campos.

```
{
  "objects" : [
    {
       "name1" : "value1",
       "name2" : "value2"
    },
    {
       "name1" : "value3",
       "name3" : "value4",
       "name4" : "value5"
    }
  ]
}
```

Al crear un archivo de definición de canalización, debe seleccionar los tipos de objetos de canalización que necesite, agregarlos al archivo de definición de canalización y, a continuación, agregar los campos correspondientes. Para obtener más información acerca de los objetos de canalización, consulte Referencia de objeto de canalización.

Por ejemplo, podría crear un objeto de definición de canalización para un nodo de datos de entrada y otro para el nodo de datos de salida. A continuación, cree otro objeto de definición de canalización para una actividad, como procesar los datos de entrada con Amazon EMR.

Campos de canalización

Cuando sepa qué tipos de objetos incluir en el archivo de definición de canalización, agregue campos a la definición de cada objeto de canalización. Los nombres de campo se encierran entre comillas y están separados de los valores de campo por un espacio, un signo de dos puntos y un espacio, como se muestra en el siguiente ejemplo.

```
"name" : "value"
```

El valor del campo puede ser una cadena de texto, una referencia a otro objeto, una llamada de función, una expresión o una lista ordenada de cualquiera de los tipos anteriores. Para obtener más información sobre los tipos de datos que se pueden utilizar para los valores de campo, consulte Tipos de datos simples . Para obtener más información acerca de las funciones que puede utilizar para evaluar los valores de campo, consulte Evaluación de expresiones.

Los campos están limitados a 2048 caracteres. Los objetos pueden tener un tamaño de 20 KB, lo que significa que no se puede agregar muchos campos grandes a un objeto.

Cada objeto de canalización debe contener los siguientes campos: `id` y `type`, tal y como se muestra en el siguiente ejemplo. También es posible que se necesiten otros campos, en función del tipo de objeto. Seleccione un valor para `id` que tenga sentido para usted y que sea único dentro de la definición de la canalización. El valor de `type` especifica el tipo de objeto. Especifique uno de los tipos de objeto de definición de canalización compatibles, que aparecen en el tema Referencia de objeto de canalización.

```
{
  "id": "MyCopyToS3",
  "type": "CopyActivity"
}
```

Para obtener más información acerca de los campos obligatorios y opcionales para cada objeto, consulte la documentación del objeto.

Para incluir campos de un objeto en otro objeto, utilice el campo `parent` con una referencia al objeto. Por ejemplo, el objeto "B" incluye sus campos, "B1" y "B2", además de los campos de objeto "A", "A1" y "A2".

```
{
  "id" : "A",
  "A1" : "value",
  "A2" : "value"
},
{
  "id" : "B",
  "parent" : {"ref" : "A"},
  "B1" : "value",
  "B2" : "value"
}
```

Puede definir campos comunes en un objeto con el ID "Default". Estos campos se incluyen automáticamente en todos los objetos del archivo de definición de canalización que no establezcan de forma explícita su campo `parent` para hacer referencia a otro objeto.

```
{
  "id" : "Default",
```

```
  "onFail" : {"ref" : "FailureNotification"},
  "maximumRetries" : "3",
  "workerGroup" : "myWorkerGroup"
}
```

Campos definidos por el usuario

Puede crear campos personalizados o definidos por el usuario en los componentes de la canalización y hacer referencia a los mismos con expresiones. En el siguiente ejemplo se muestra un campo personalizado denominado `myCustomField` y `my_customFieldReference` agregado a un objeto S3DataNode:

```
{
  "id": "S3DataInput",
  "type": "S3DataNode",
  "schedule": {"ref": "TheSchedule"},
  "filePath": "s3://bucket_name",
  "myCustomField": "This is a custom value in a custom field.",
  "my_customFieldReference": {"ref":"AnotherPipelineComponent"}
  },
```

Un campo definido por el usuario debe tener un nombre precedido por la palabra "my" en minúsculas, seguida de una letra mayúscula o un guion bajo. Además, un campo definido por el usuario puede ser un valor de cadena como en el ejemplo `myCustomField` anterior o una referencia a otro componente como en el ejemplo `my_customFieldReference` anterior.

> **Note**
>
> En los campos definidos por el usuario, AWS Data Pipeline solo comprueba si hay referencias válidas a otros componentes de la canalización, pero no los valores de cadena de los campos personalizados que usted agregue.

Uso de la API

> **Note**
>
> Si no escribe programas que interactúan con AWS Data Pipeline, no es necesario que instale ninguno de los SDK de AWS. Puede crear y ejecutar canalizaciones con la interfaz de línea

de comandos o con la consola. Para obtener más información, consulte Configurándose para AWS Data Pipeline

La forma más sencilla de escribir aplicaciones que interactúen con AWS Data Pipeline o de implementar un Task Runner personalizado es usar uno de los SDK de AWS. Los SDK de AWS proporcionan funcionalidad que simplifica la llamada a las API del servicio web desde su entorno de programación preferido. Para obtener más información, consulte Instalar el SDK de AWS.

Instalar el SDK de AWS

Los SDK de AWS ofrecen funciones que encapsulan la API y se encargan de muchos de los detalles de la conexión, tales como el cálculo de firmas, el control de reintentos de solicitud y el control de errores. Los SDK también contienen código de ejemplo, tutoriales y otros recursos para ayudarle a empezar a escribir aplicaciones que llaman a AWS. Llamar a las funciones contenedoras de un SDK puede simplificar en gran medida el proceso de escribir una aplicación de AWS. Para obtener más información sobre cómo descargar y usar los SDK de AWS, vaya a Código de muestra y bibliotecas.

AWS Data PipelineLa compatibilidad con está disponible en los SDK para las siguientes plataformas:

- AWS SDK para Java
- AWS SDK para Node.js
- AWS SDK para PHP
- El AWS SDK para Python (Boto)
- AWS SDK para Ruby
- AWS SDK para .NET

Realización de una solicitud HTTP a AWS Data Pipeline

Para obtener una descripción completa de los objetos de programación de AWS Data Pipeline, consulte la Referencia de la API de AWS Data Pipeline.

Si no usa ninguno de los SDK de AWS, puede realizar operaciones de AWS Data Pipeline sobre HTTP mediante el método de solicitud POST. El método POST requiere que especifique la operación en el encabezado de la solicitud y proporcione los datos para la operación en formato JSON en el cuerpo de la solicitud.

Contenido de los encabezados HTTP

AWS Data Pipeline requiere que figure la siguiente información en el encabezado de una solicitud HTTP:

- `host` El punto de enlace de AWS Data Pipeline.

  Para obtener información sobre los puntos de enlace, consulte Regiones y puntos de conexión.
- `x-amz-date` Debe proporcionar la marca temporal que figura en el encabezado Date de HTTP o en el encabezado x-amz-date de AWS. (Algunas bibliotecas de cliente HTTP no permiten configurar el encabezado Date). Cuando hay un encabezado x-amz-date presente, el sistema hace caso omiso de cualquier encabezado Date durante la autenticación de la solicitud.

  La fecha debe especificarse en uno de los tres formatos siguientes, como se especifica en HTTP/1.1 RFC:
  - Sun, 06 Nov 1994 08:49:37 GMT (RFC 822, actualizado por RFC 1123)
  - Sunday, 06-Nov-94 08:49:37 GMT (RFC 850, obsoleto en RFC 1036)
  - Sun Nov 6 08:49:37 1994 (formato asctime() de ANSI C)
- `Authorization` El conjunto de parámetros de autorización que AWS usa para garantizar la validez y autenticidad de la solicitud. Para obtener más información acerca de la creación de este encabezado, vaya a Proceso de firma Signature Version 4.
- `x-amz-target` el servicio de destino de la solicitud y la operación de los datos, en el formato: `<<serviceName>>_<<API version>>.<<operationName>>`

  Por ejemplo: ., `DataPipeline_20121129.ActivatePipeline`
- `content-type` Especifica JSON y la versión. Por ejemplo: ., `Content-Type: application/x-amz-json-1.0`

A continuación se muestra un ejemplo de un encabezado en una solicitud HTTP para activar una canalización.

```
POST / HTTP/1.1
host: https://datapipeline.us-east-1.amazonaws.com
x-amz-date: Mon, 12 Nov 2012 17:49:52 GMT
x-amz-target: DataPipeline_20121129.ActivatePipeline
Authorization: AuthParams
```

```
Content-Type: application/x-amz-json-1.1
Content-Length: 39
Connection: Keep-Alive
```

Contenido de cuerpo HTTP

El cuerpo de una solicitud HTTP contiene los datos de la operación especificada en el encabezado de la solicitud HTTP. Los datos deben formatearse de acuerdo con el esquema de datos JSON para cada API de AWS Data Pipeline. El esquema de datos JSON de AWS Data Pipeline define los tipos de datos y parámetros (como operadores de comparación y constantes de enumeración) disponibles para cada operación.

Formatear el cuerpo de una solicitud HTTP

Use el formato de datos JSON para transmitir los valores de los datos y la estructura de datos, de forma simultánea. Los elementos se pueden anidar en otros elementos mediante la notación de corchete. En el siguiente ejemplo se muestra una solicitud para poner una definición de la canalización compuesta de tres objetos y sus slots correspondientes.

```
{
 "pipelineId": "df-00627471SOVYZEXAMPLE",
 "pipelineObjects":
  [
    {"id": "Default",
     "name": "Default",
     "slots":
      [
        {"key": "workerGroup",
         "stringValue": "MyWorkerGroup"}
      ]
    },
    {"id": "Schedule",
     "name": "Schedule",
     "slots":
      [
       {"key": "startDateTime",
         "stringValue": "2012-09-25T17:00:00"},
       {"key": "type",
        "stringValue": "Schedule"},
       {"key": "period",
        "stringValue": "1 hour"},
```

```
        {"key": "endDateTime",
         "stringValue": "2012-09-25T18:00:00"}
      ]
    },
    {"id": "SayHello",
     "name": "SayHello",
     "slots":
      [
        {"key": "type",
         "stringValue": "ShellCommandActivity"},
        {"key": "command",
         "stringValue": "echo hello"},
        {"key": "parent",
         "refValue": "Default"},
        {"key": "schedule",
         "refValue": "Schedule"}

      ]
    }
  ]
}
```

Gestionar la respuesta HTTP

Estos son algunos encabezados importantes en la respuesta HTTP y cómo debe gestionarlos en su aplicación:

- HTTP/1.1: este encabezado viene seguido de un código de estado. Un valor de código de 200 indica el éxito de la operación. Cualquier otro valor indica un error.
- x-amzn-RequestId: este encabezado contiene un ID de solicitud que puede usar si tiene que solucionar los problemas de una solicitud con AWS Data Pipeline. Un ejemplo de un ID de solicitud es K2QH8DNOU907N97FNA2GDLL8OBVV4KQNSO5AEMVJF66Q9ASUAAJG.
- x-amz-crc32: AWS Data Pipeline calcula una suma de comprobación CRC32 de la carga de HTTP y devuelve esta suma de comprobación en el encabezado x-amz-crc32. Recomendamos que calcule su propia suma de comprobación CRC32 en el lado del cliente y la compare con el encabezado x-amz-crc32; si las sumas de comprobación no coinciden, podría indicar que los datos sufrieron daños en tránsito. Si esto ocurre, debería volver a realizar la solicitud.

No es necesario que los usuarios de SDK de AWS realicen manualmente esta verificación, ya que los SDK calculan la suma de comprobación de cada respuesta desde Amazon DynamoDB y reintentan automáticamente si se detecta una discordancia.

Ejemplo de solicitud y respuesta JSON de AWS Data Pipeline

En los siguientes ejemplos se muestra una solicitud para crear una nueva canalización. A continuación, se muestra la respuesta de AWS Data Pipeline, incluido el identificador de canalización de la canalización recién creada.

Solicitud HTTP POST

```
POST / HTTP/1.1
host: https://datapipeline.us-east-1.amazonaws.com
x-amz-date: Mon, 12 Nov 2012 17:49:52 GMT
x-amz-target: DataPipeline_20121129.CreatePipeline
Authorization: AuthParams
Content-Type: application/x-amz-json-1.1
Content-Length: 50
Connection: Keep-Alive

{"name": "MyPipeline",
 "uniqueId": "12345ABCDEFG"}
```

AWS Data PipelineRespuesta de

```
HTTP/1.1 200
x-amzn-RequestId: b16911ce-0774-11e2-af6f-6bc7a6be60d9
x-amz-crc32: 2215946753
Content-Type: application/x-amz-json-1.0
Content-Length: 2
Date: Mon, 16 Jan 2012 17:50:53 GMT

{"pipelineId": "df-00627471SOVYZEXAMPLE"}
```

Seguridad en AWS Data Pipeline

La seguridad en la nube AWS es la máxima prioridad. Como AWS cliente, usted se beneficia de los centros de datos y las arquitecturas de red diseñados para cumplir con los requisitos de las organizaciones más sensibles a la seguridad.

La seguridad es una responsabilidad compartida entre AWS usted y usted. El modelo de responsabilidad compartida describe esto como seguridad de la nube y seguridad en la nube:

- Seguridad de la nube: AWS es responsable de proteger la infraestructura que ejecuta AWS los servicios en la AWS nube. AWS también le proporciona servicios que puede utilizar de forma segura. Los auditores externos prueban y verifican periódicamente la eficacia de nuestra seguridad como parte de los AWS programas de de . Para obtener más información sobre los programas de conformidad aplicables AWS Data Pipeline, consulte Servicios de AWS dentro del alcance por programa de conformidad .
- Seguridad en la nube: su responsabilidad viene determinada por el AWS servicio que utilice. También es responsable de otros factores, incluida la confidencialidad de los datos, los requisitos de la empresa y la legislación y la normativa aplicables.

Esta documentación le ayuda a comprender cómo aplicar el modelo de responsabilidad compartida cuando se utiliza AWS Data Pipeline. Los siguientes temas muestran cómo configurarlo AWS Data Pipeline para cumplir sus objetivos de seguridad y conformidad. También aprenderá a utilizar otros servicios de AWS que le ayudan a supervisar y proteger sus AWS Data Pipeline recursos.

Temas

Protección de datos en AWS Data Pipeline

El modelo de responsabilidad compartida de AWS se aplica a la protección de datos de AWS Data Pipeline. Como se describe en este modelo, AWS es responsable de proteger la infraestructura global que ejecuta toda la Nube de AWS. Es responsable de mantener el control sobre su contenido que se encuentra alojado en esta infraestructura. Este contenido incluye la configuración de seguridad y las tareas de administración para el Servicios de AWS que utiliza. Para obtener más información sobre la privacidad de datos, consulte Preguntas frecuentes sobre la privacidad de datos. Para obtener información sobre la protección de datos en Europa, consulta la publicación de blog sobre el Modelo de responsabilidad compartida de AWS y GDPR en el Blog de seguridad de AWS.

Con fines de protección de datos, recomendamos proteger las credenciales de la Cuenta de AWS y configurar cuentas de usuario individuales con AWS IAM Identity Center o AWS Identity and Access Management (IAM). De esta manera, solo se otorgan a cada usuario los permisos necesarios para cumplir sus obligaciones laborales. También recomendamos proteger sus datos de la siguiente manera:

- Utiliza la autenticación multifactor (MFA) en cada cuenta.
- Utilice SSL/TLS para comunicarse con los recursos de AWS. Recomendamos TLS 1.2 o una versión posterior.
- Configure los registros de API y de actividad de los usuarios con AWS CloudTrail.
- Utilice las soluciones de cifrado de AWS, junto con todos los controles de seguridad predeterminados dentro de los servicios de Servicios de AWS.
- Utilice servicios de seguridad administrados avanzados, como Amazon Macie, que lo ayuden a detectar y proteger los datos confidenciales almacenados en Amazon S3.
- Si necesita módulos criptográficos validados FIPS 140-2 al acceder a AWS a través de una interfaz de línea de comandos o una API, utilice un punto de conexión de FIPS. Para obtener más información sobre los puntos de conexión de FIPS disponibles, consulte Estándar de procesamiento de la información federal (FIPS) 140-2.
- AWS Data Pipeline es compatible con IMDSv2 para los recursos de Amazon EMR y Amazon EC2. Para usar IMDSv2 con Amazon EMR, utilice las versiones 5.23.1, 5.27.1 o 5.32 o posteriores o la versión 6.2 o posterior. Para obtener más información, consulte Configurar las solicitudes de servicio de metadatos de instancia de Amazon EC2 y utilizar IMDSv2.

Se recomienda encarecidamente no ingresar nunca información confidencial o sensible, como, por ejemplo, direcciones de correo electrónico de clientes, en etiquetas o campos de formato libre, tales como el campo Nombre. Esto incluye las situaciones en las que debe trabajar con la AWS Data Pipeline u otros Servicios de AWS a través de la consola, la API, la AWS CLI o los SDK de AWS. Cualquier dato que ingrese en etiquetas o campos de texto de formato libre utilizados para nombres se puede emplear para los registros de facturación o diagnóstico. Si proporciona una URL a un servidor externo, recomendamos encarecidamente que no incluya información de credenciales en la URL a fin de validar la solicitud para ese servidor.

Identity and Access Management para AWS Data Pipeline

Las credenciales de seguridad le identifican en los servicios de AWS y le conceden el uso ilimitado de sus recursos de AWS, como sus canalizaciones. Puede utilizar las funciones de AWS Data Pipeline y AWS Identity and Access Management (IAM) para permitir que AWS Data Pipeline otros usuarios accedan a sus AWS Data Pipeline recursos sin compartir sus credenciales de seguridad.

Las organizaciones pueden compartir el acceso a canalizaciones, de modo que las personas de esa organización puedan desarrollarlas y mantenerlas en equipo. Sin embargo, sería necesario hacer lo siguiente, por ejemplo:

- Controlar qué usuarios pueden acceder a canalizaciones específicas
- Proteger una canalización de producción para que no se modifique por error.
- Permitir que un auditor tenga acceso de solo lectura a las canalizaciones, pero impedirle realizar cambios.

AWS Data Pipeline está integrado con AWS Identity and Access Management (IAM), que ofrece una amplia gama de funciones:

- Cree usuarios y grupos en su. Cuenta de AWS
- Comparta fácilmente sus AWS recursos entre los usuarios de su Cuenta de AWS.
- Asignación de credenciales de seguridad exclusivas a los usuarios.
- Control del acceso de los usuarios a los servicios y recursos.
- Obtención de una sola factura para todos los usuarios de su Cuenta de AWS.

Al usar IAM with AWS Data Pipeline, puede controlar si los usuarios de su organización pueden realizar una tarea mediante acciones de API específicas y si pueden usar recursos de AWS

específicos. Puede utilizar políticas de IAM basadas en las etiquetas de las canalizaciones y los grupos de trabajadores para compartir las canalizaciones con otros usuarios y controlar su nivel de acceso.

Contenido

Políticas de IAM para AWS Data Pipeline

De forma predeterminada, las entidades de IAM no tienen permiso para crear ni modificar recursos de AWS. Para permitir a las entidades de IAM crear o modificar recursos y realizar tareas, debe crear políticas de IAM que concedan a esas entidades de IAM permisos para usar los recursos y las acciones de la API que necesitarán y, a continuación, asociar dichas políticas a las entidades de IAM que requieran dichos permisos.

Cuando se asocia una política a un usuario o grupo de usuarios, les otorga o deniega el permiso para realizar las tareas especificadas en los recursos indicados. Para obtener información general acerca de las políticas de IAM, consulte Permisos y políticas en la Guía de usuario de IAM. Para obtener más información sobre cómo crear y administrar políticas personalizadas de IAM, consulte Administración de políticas de IAM.

Contenido

Sintaxis de la política

Una política de IAM es un documento JSON que contiene una o varias instrucciones. Cada instrucción tiene la estructura siguiente:

```
{
  "Statement":[{
    "Effect":"effect",
```

```
    "Action":"action",
    "Resource":"*",
    "Condition":{
      "condition":{
        "key":"value"
        }
      }
    }
  ]
}
```

Una instrucción de política se compone de los siguientes elementos:

- Effect: el valor de effect puede ser `Allow` o `Deny`. De forma predeterminada, las entidades de IAM no tienen permiso para utilizar los recursos y las acciones de la API, por lo que se deniegan todas las solicitudes. Si se concede un permiso explícito se anula el valor predeterminado. Una denegación explícita invalida cualquier permiso concedido.
- Action: el valor de action es la acción de la API para la que concede o deniega permisos. Para ver una lista de las acciones AWS Data Pipeline, consulte la referencia sobre las acciones de la AWS Data Pipeline API.
- Resource: el recurso al que afecta la acción. El único valor válido aquí es `"*"`.
- Condition: las condiciones son opcionales. Se pueden usar para controlar cuándo entrará en vigor la política.

  AWS Data Pipeline implementa las claves de contexto de todo AWS (consulte Claves disponibles para las condiciones), además de las siguientes claves específicas del servicio.

  - `datapipeline:PipelineCreator`: para conceder el acceso al usuario que ha creado la canalización. Por ejemplo, consulte Grant the pipeline owner full access.
  - `datapipeline:Tag`: para conceder acceso basado en el etiquetado de la canalización. Para obtener más información, consulte Control del acceso a canalizaciones mediante etiquetas.
  - `datapipeline:workerGroup`: para conceder acceso basado en el nombre del grupo de trabajadores. Para obtener más información, consulte Control del acceso a canalizaciones mediante grupos de procesos de trabajo.

Control del acceso a canalizaciones mediante etiquetas

Puede crear políticas de IAM que hagan referencia a las etiquetas de la canalización. Esto le permite utilizar el etiquetado de canalización para hacer lo siguiente:

- Conceder acceso de solo lectura a una canalización.
- Conceda acceso a una canalización read/write
- Bloquear el acceso a una canalización.

Por ejemplo, suponga que un administrador tiene dos entornos de canalización, producción y desarrollo, además de un grupo de IAM para cada entorno. En el caso de las canalizaciones del entorno de producción, el administrador concede read/write acceso a los usuarios del grupo de IAM de producción, pero concede acceso de solo lectura a los usuarios del grupo de IAM de desarrolladores. En el caso de las canalizaciones del entorno de desarrollo, el administrador concede read/write acceso a los grupos de IAM de producción y de desarrolladores.

Para conseguir este escenario, el administrador etiqueta las canalizaciones de producción con la etiqueta “environment=production” y conecta la siguiente política al grupo de IAM de desarrolladores. La primera instrucción concede acceso de solo lectura a todas las canalizaciones. La segunda declaración permite el read/write acceso a las canalizaciones que no tienen la etiqueta «environment=production».

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "datapipeline:Describe*",
        "datapipeline:ListPipelines",
        "datapipeline:GetPipelineDefinition",
        "datapipeline:QueryObjects"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "datapipeline:*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {"datapipeline:Tag/environment": "production"}
      }
```

```
    }
  ]
}
```

Además, el administrador conecta la siguiente política al grupo de IAM de producción. Esta instrucción concede acceso completo a todas las canalizaciones.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "datapipeline:*",
      "Resource": "*"
    }
  ]
}
```

Para ver más ejemplos, consulte Grant users read-only access based on a tag y Grant users full access based on a tag.

Control del acceso a canalizaciones mediante grupos de procesos de trabajo

Puede crear políticas de IAM que hagan referencia a los nombres de los grupos de trabajadores.

Por ejemplo, suponga que un administrador tiene dos entornos de canalización, producción y desarrollo, además de un grupo de IAM para cada entorno. El administrador tiene tres servidores de bases de datos con aplicaciones de ejecución de tareas configuradas para los entornos de producción, preproducción y desarrollo, respectivamente. El administrador quiere asegurarse de que los usuarios en el grupo de IAM de producción pueden crear canalizaciones que envíen tareas a los recursos de producción y que los usuarios del grupo de IAM de desarrollo pueden crear canalizaciones que envíen tareas a los recursos de preproducción y desarrollo.

Para conseguir este escenario, el administrador instala la aplicación de ejecución de tareas en los recursos de producción con credenciales de producción y establece `workerGroup` en "prodresource". Además, el administrador instala la aplicación de ejecución de tareas en los recursos

de desarrollo con credenciales de desarrollo y establece `workerGroup` en "pre-production" y "development". El administrador conecta la siguiente política al grupo de IAM de desarrolladores para bloquear el acceso a los recursos “prodresource”. La primera instrucción concede acceso de solo lectura a todas las canalizaciones. La segunda sentencia permite el read/write acceso a las canalizaciones cuando el nombre del grupo de trabajo tiene el prefijo «dev» o «pre-prod».

Asimismo, el administrador conecta la siguiente política al grupo de IAM de producción para conceder acceso a los recursos “prodresource”. La primera instrucción concede acceso de solo lectura a todas las canalizaciones. La segunda sentencia concede el read/write acceso cuando el nombre del grupo de trabajo tiene el prefijo «prod».

Ejemplos de políticas para AWS Data Pipeline

Los siguientes ejemplos muestran cómo conceder a los usuarios acceso completo o restringido a canalizaciones.

Contenido

Ejemplo 1: Otorgar a los usuarios acceso de solo lectura basado en una etiqueta

La siguiente política permite a los usuarios usar las acciones de la AWS Data Pipeline API de solo lectura, pero solo con las canalizaciones que tengan la etiqueta «environment=production».

La acción de la ListPipelines API no admite la autorización basada en etiquetas.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "datapipeline:Describe*",
```

```
        "datapipeline:GetPipelineDefinition",
        "datapipeline:ValidatePipelineDefinition",
        "datapipeline:QueryObjects"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "datapipeline:Tag/environment": "production"
        }
      }
    }
  ]
}
```

Ejemplo 2: Otorgar a los usuarios acceso completo basado en una etiqueta

La siguiente política permite a los usuarios utilizar todas las acciones de la AWS Data Pipeline API, con la excepción de las canalizaciones que tengan la etiqueta «environment=test» ListPipelines, pero solo con ellas.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "datapipeline:*"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "datapipeline:Tag/environment": "test"
        }
      }
    }
```

```
  ]
}
```

Ejemplo 3: Otorgar acceso completo al propietario de la canalización

La siguiente política permite a los usuarios utilizar todas las acciones de la AWS Data Pipeline API, pero solo con sus propias canalizaciones.

Ejemplo 4: conceder a los usuarios acceso a la consola AWS Data Pipeline

La siguiente política permite a los usuarios crear y administrar una canalización mediante la consola de AWS Data Pipeline .

Esta política incluye la acción relativa a `PassRole` los permisos para recursos específicos vinculados a las `roleARN` AWS Data Pipeline necesidades de cada uno. Para obtener más información sobre el `PassRole` permiso basado en la identidad (IAM), consulte la entrada del blog Cómo conceder permisos para lanzar EC2 instancias con funciones de IAM (permiso). PassRole

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [{
   "Action": [
    "cloudwatch:*",
    "datapipeline:*",
    "dynamodb:DescribeTable",
    "elasticmapreduce:AddJobFlowSteps",
    "elasticmapreduce:ListInstance*",
    "iam:AddRoleToInstanceProfile",
    "iam:CreateInstanceProfile",
    "iam:GetInstanceProfile",
    "iam:GetRole",
    "iam:GetRolePolicy",
    "iam:ListInstanceProfiles",
    "iam:ListInstanceProfilesForRole",
    "iam:ListRoles",
    "rds:DescribeDBInstances",
    "rds:DescribeDBSecurityGroups",
    "redshift:DescribeClusters",
    "redshift:DescribeClusterSecurityGroups",
```

```
    "s3:List*",
    "sns:ListTopics"
   ],
   "Effect": "Allow",
   "Resource": [
    "*"
   ]
  },
  {
   "Action": "iam:PassRole",
   "Effect": "Allow",
   "Resource": [
    "arn:aws:iam::*:role/DataPipelineDefaultResourceRole",
    "arn:aws:iam::*:role/DataPipelineDefaultRole"
   ]
  }
 ]
}
```

Funciones de IAM para AWS Data Pipeline

AWS Data Pipeline usa AWS Identity and Access Management roles. Las políticas de permisos asociadas a las funciones de IAM determinan qué acciones AWS Data Pipeline y aplicaciones pueden realizar y a qué AWS recursos pueden acceder. Para obtener más información, consulte Roles de IAM en la Guía del usuario de IAM.

AWS Data Pipeline requiere dos funciones de IAM:

- La función Pipeline controla el AWS Data Pipeline acceso a sus recursos de AWS. En las definiciones de objetos de canalización, el campo de `role` especifica este rol.
- El rol de EC2 instancia controla el acceso que las aplicaciones que se ejecutan en las EC2 instancias, incluidas las EC2 instancias de los clústeres de Amazon EMR, tienen a AWS los recursos. En las definiciones de objetos de canalización, el campo de `resourceRole` especifica este rol.

> **Important**
>
> Si creó una canalización antes del 3 de octubre de 2022 con la AWS Data Pipeline consola con las funciones predeterminadas, AWS Data Pipeline creó la

`DataPipelineDefaultRole` suya y adjuntó la política `AWSDataPipelineRole` administrada a la función. A partir del 3 de octubre de 2022, la política de administrada de `AWSDataPipelineRole` quedará obsoleta y se debe especificar el rol de canalización para una canalización al usar la consola.
Le recomendamos que revise las canalizaciones existentes y determine si `DataPipelineDefaultRole` está asociada a la canalización y si `AWSDataPipelineRole` está asociada a ese rol. Si es así, revise el acceso que permite esta política para asegurarse de que es adecuado para sus requisitos de seguridad. Añada, actualice o sustituya las políticas y declaraciones de políticas adjuntas a esta función según sea necesario. Como alternativa, puede actualizar una canalización para usar un rol que cree con diferentes políticas de permisos.

Ejemplo de políticas de permisos para AWS Data Pipeline roles

Cada función tiene una o más políticas de permisos adjuntas que determinan los recursos de AWS a los que puede acceder la función y las acciones que puede realizar. En este tema se proporciona un ejemplo de política de permisos para el rol de canalización. También proporciona el contenido de`AmazonEC2RoleforDataPipelineRole`, que es la política administrada para el rol de EC2 instancia predeterminado,`DataPipelineDefaultResourceRole`.

Ejemplo de política de permisos para roles de canalización

La política de ejemplo que se muestra a continuación tiene como objetivo permitir las funciones esenciales que AWS Data Pipeline requieren ejecutar una canalización con los recursos de Amazon EC2 y Amazon EMR. También proporciona permisos para acceder a otros AWS recursos, como Amazon Simple Storage Service y Amazon Simple Notification Service, que requieren muchas canalizaciones. Si los objetos definidos en una canalización no requieren los recursos de un AWS servicio, te recomendamos encarecidamente que elimines los permisos de acceso a ese servicio. Por ejemplo, si su canalización no define un Nodo Dynamo DBData o usa la acción SnsAlarm, le recomendamos que elimine las instrucciones de permiso para esas acciones.

- *111122223333*Sustitúyalo por tu ID de AWS cuenta.
- Sustituya *NameOfDataPipelineRole* por el nombre del rol de canalización (la función a la que se asocia esta política).
- *NameOfDataPipelineResourceRole*Sustitúyalo por el nombre del rol de la EC2 instancia.
- Sustituya *us-west-1* por la región correspondiente a la aplicación.

Política administrada predeterminada para el rol de EC2 instancia

El contenido de `AmazonEC2RoleforDataPipelineRole` se muestra a continuación. Esta es la política administrada asociada a la función de recursos predeterminada para AWS Data Pipeline,`DataPipelineDefaultResourceRole`. Cuando definas un rol de recurso para tu canalización, te recomendamos empezar con esta política de permisos y, después, eliminar los permisos para las acciones de AWS servicio que no sean necesarias.

Se muestra la versión 3 de la política, que es la versión más reciente en el momento de escribir este artículo. Consulte la versión más reciente de la política en la consola de IAM.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
      "Effect": "Allow",
      "Action": [
        "cloudwatch:*",
        "datapipeline:*",
        "dynamodb:*",
        "ec2:Describe*",
        "elasticmapreduce:AddJobFlowSteps",
        "elasticmapreduce:Describe*",
        "elasticmapreduce:ListInstance*",
        "elasticmapreduce:ModifyInstanceGroups",
        "rds:Describe*",
        "redshift:DescribeClusters",
        "redshift:DescribeClusterSecurityGroups",
        "s3:*",
        "sdb:*",
        "sns:*",
        "sqs:*"
      ],
      "Resource": ["*"]
    }]
}
```

Creación de roles de IAM AWS Data Pipeline y edición de permisos de rol

Utilice los siguientes procedimientos para crear roles para AWS Data Pipeline usar la consola de IAM. El proceso consta de cuatro pasos. En primer lugar, cree una política de permisos para adjuntar al rol. A continuación, se crea el rol y se adjunta la política. Después de crear un rol, puede cambiar los permisos del rol adjuntando y separando las políticas de permisos.

> **Note**
>
> Al crear roles para AWS Data Pipeline usar la consola, tal como se describe a continuación, IAM crea y adjunta las políticas de confianza adecuadas que requiere el rol.

Para crear una política de permisos para utilizarla con un rol AWS Data Pipeline

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Políticas y, a continuación, Crear política.
3. Seleccione la pestaña JSON.
4. Si está creando un rol de canalización, copie y pegue el contenido del ejemplo de política en Ejemplo de política de permisos para roles de canalización, editándolo según convenga para sus requisitos de seguridad. Como alternativa, si vas a crear un rol de EC2 instancia personalizado, haz lo mismo con el ejemplo dePolítica administrada predeterminada para el rol de EC2 instancia.
5. Elija Revisar política.
6. Introduzca un nombre para la política, por ejemplo, `MyDataPipelineRolePolicy`, y una Descripción, y a continuación, elija Crear política.
7. Tome nota del nombre de la política. Lo necesita cuando crea su rol.

Para crear un rol de IAM para AWS Data Pipeline

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Roles y luego seleccione Crear rol.
3. En Elegir un caso de uso, seleccione Canalización de datos.
4. En Seleccione su caso de uso, realice una de las siguientes acciones:
   - Elija `Data Pipeline` para crear un rol de canalización.

- Elija `EC2 Role for Data Pipeline` para crear un rol de recurso.

5. Elija Siguiente: permisos.
6. Si AWS Data Pipeline aparece la política predeterminada para, siga los pasos siguientes para crear el rol y, a continuación, edítelo según las instrucciones del siguiente procedimiento. De lo contrario, introduzca el nombre de la política que creó en el procedimiento anterior y, a continuación, selecciónela en la lista.
7. Seleccione Siguiente: etiquetas, introduzca las etiquetas que desee añadir al rol y, a continuación, elija Siguiente: revisar.
8. Escriba el nombre del rol (por ejemplo, `MyDataPipelineRole`) y una descripción opcional y después elija Crear rol.

Para adjuntar o desvincular una política de permisos para un rol de IAM para AWS Data Pipeline

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. Seleccione Roles en el panel de navegación.
3. En el cuadro de búsqueda, comience a escribir el nombre del rol que desea editar (por ejemplo, DataPipelineDefaultRoleo) MyDataPipelineRoley, a continuación, elija el nombre del rol de la lista.
4. En la página Permisos, haga lo siguiente:

   - Para separar una política de permisos, en Políticas de permisos, pulse el botón de eliminación situado en el extremo derecho de la entrada de la política. Cuando se le pida confirmación, elija Desvincular.
   - Para adjuntar una política que haya creado anteriormente, elija Adjuntar políticas. En el cuadro de búsqueda, comience a escribir el nombre de la política que desee editar, selecciónela de la lista y, a continuación, elija Adjuntar política.

Cambiar las funciones de una canalización existente

Si quieres asignar un rol de canalización o un rol de recurso diferente a un canalización, puedes usar el editor de arquitectos de la consola. AWS Data Pipeline

Para editar las funciones asignadas a una canalización mediante la consola

1. Abre la AWS Data Pipeline consola en https://console.aws.amazon.com/datapipeline/.

2. Seleccione la canalización de la lista y, a continuación, elija Acciones, Editar.
3. En el panel derecho del editor de arquitectos, elija Otros.
4. En las listas Función de recurso y Función, elija las funciones AWS Data Pipeline que desee asignar y, a continuación, seleccione Guardar.

Registro y monitorización en AWS Data Pipeline

AWS Data Pipeline se integra con AWS CloudTrail, un servicio que proporciona un registro de las acciones hechas por un usuario, un rol o un servicio de AWS en AWS Data Pipeline. CloudTrail captura todas las llamadas a la API de AWS Data Pipeline como eventos. Las llamadas capturadas incluyen las llamadas desde la consola de AWS Data Pipeline y las llamadas desde el código a las operaciones de la API de AWS Data Pipeline. Si crea un registro de seguimiento, puede habilitar la entrega continua de eventos de CloudTrail a un bucket de Amazon S3, incluidos los eventos para AWS Data Pipeline. Si no configura un registro de seguimiento, puede ver los eventos más recientes en la consola de CloudTrail en el Historial de eventos. Mediante la información recopilada por CloudTrail, puede determinar la solicitud que se realizó a AWS Data Pipeline, la dirección IP desde la que se realizó, quién la realizó y cuándo, etc.

Para obtener más información sobre CloudTrail, consulte la Guía del usuario de AWS CloudTrail.

AWS Data PipelineInformación de en CloudTrail

CloudTrail se habilita en su cuenta de AWS cuando la crea. Cuando se produce una actividad en AWS Data Pipeline, esa actividad se registra en un evento de CloudTrail junto con otros eventos de servicio de AWS en Historial de eventos. Puede ver, buscar y descargar los últimos eventos de la cuenta de AWS. Para obtener más información, consulte Visualización de eventos con el historial de eventos de CloudTrail.

Para mantener un registro continuo de eventos en la cuenta de AWS, incluidos los eventos de AWS Data Pipeline, cree un registro de seguimiento. Un registro de seguimiento permite a CloudTrail enviar archivos de registro a un bucket de Amazon S3. De forma predeterminada, cuando se crea un registro de seguimiento en la consola, el registro de seguimiento se aplica a todas las regiones de AWS. El registro de seguimiento registra los eventos de todas las regiones de la partición de AWS y envía los archivos de registro al bucket de Amazon S3 especificado. También es posible configurar otros servicios de AWS para analizar en profundidad y actuar en función de los datos de eventos recopilados en los registros de CloudTrail. Para más información, consulte los siguientes temas:

- Introducción a la creación de registros de seguimiento

- Servicios e integraciones compatibles con CloudTrail
- Configuración de notificaciones de Amazon SNS para CloudTrail
- Recepción de archivos de registro de CloudTrail de varias regiones y Recepción de archivos de registro de CloudTrail de varias cuentas

Todas las acciones de AWS Data Pipeline las registra CloudTrail y se documentan en el capítulo sobre acciones de la Referencia de la API de AWS Data Pipeline. Por ejemplo, las llamadas a la acción CreatePipeline generan entradas en los archivos de registro de CloudTrail.

Cada entrada de registro o evento contiene información sobre quién generó la solicitud. La información de identidad del usuario le ayuda a determinar lo siguiente:

- Si la solicitud se realizó con las credenciales raíz o las credenciales de rol de IAM.
- Si la solicitud se realizó con credenciales de seguridad temporales de un rol o fue un usuario federado.
- Si la solicitud la realizó otro servicio de AWS.

Para obtener más información, consulte el Elemento userIdentity de CloudTrail.

Descripción de las entradas de archivos de registro de AWS Data Pipeline

Un registro de seguimiento es una configuración que permite la entrega de eventos como archivos de registro a un bucket de Amazon S3 que especifique. Los archivos de registro de CloudTrail pueden contener una o varias entradas de registro. Un evento representa una solicitud específica realizada desde un origen cualquiera y contiene información sobre la acción solicitada, la fecha y la hora de la acción, los parámetros de la solicitud, etc. Los archivos de registro de CloudTrail no rastrean el orden en la pila de las llamadas públicas a la API, por lo que estas no aparecen en ningún orden específico.

En el ejemplo que sigue se muestra una entrada de registro de CloudTrail que ilustra la operación `CreatePipeline`:

```
    {
"Records": [
  {
    "eventVersion": "1.02",
    "userIdentity": {
```

```
        "type": "Root",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::aws-account-id:role/role-name",
        "accountId": "role-account-id",
        "accessKeyId": "role-access-key"
      },
      "eventTime": "2014-11-13T19:15:15Z",
      "eventSource": "datapipeline.amazonaws.com",
      "eventName": "CreatePipeline",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "72.21.196.64",
      "userAgent": "aws-cli/1.5.2 Python/2.7.5 Darwin/13.4.0",
      "requestParameters": {
        "name": "testpipeline",
        "uniqueId": "sounique"
      },
      "responseElements": {
        "pipelineId": "df-06372391ZG65EXAMPLE"
      },
      "requestID": "65cbf1e8-6b69-11e4-8816-cfcbadd04c45",
      "eventID": "9f99dce0-0864-49a0-bffa-f72287197758",
      "eventType": "AwsApiCall",
      "recipientAccountId": "role-account-id"
    },
      ...additional entries
  ]
}
```

Respuesta a incidentes en AWS Data Pipeline

La respuesta a un incidente AWS Data Pipeline es una AWS responsabilidad. AWS tiene una política y un programa formales y documentados que rigen la respuesta a los incidentes.

Los problemas operativos de AWS con amplia repercusión se publican en AWS Service Health Dashboard. Los problemas operativos también se publican en las cuentas individuales a través del Personal Health Dashboard.

Validación del cumplimiento para AWS Data Pipeline

AWS Data Pipeline no está incluido en el ámbito de ningún programa de conformidad de AWS. Para obtener una lista de los servicios de AWS en el ámbito de programas de conformidad específicos,

consulte AWS Services in Scope by Compliance Program (Servicios de AWS en el ámbito del programa de conformidad. Para obtener información general, consulte AWS Compliance Programs (Programas de conformidad de AWS).

Resiliencia en AWS Data Pipeline

La infraestructura AWS global se basa en AWS regiones y zonas de disponibilidad. AWS Las regiones proporcionan varias zonas de disponibilidad aisladas y separadas físicamente, que están conectadas mediante redes de baja latencia, alto rendimiento y alta redundancia. Con las zonas de disponibilidad, puede diseñar y utilizar aplicaciones y bases de datos que realizan una conmutación por error automática entre las zonas sin interrupciones. Las zonas de disponibilidad tienen una mayor disponibilidad, tolerancia a errores y escalabilidad que las infraestructuras tradicionales de uno o varios centros de datos.

Para obtener más información sobre AWS las regiones y las zonas de disponibilidad, consulte Infraestructura global.AWS

Seguridad de la infraestructura en AWS Data Pipeline

Como servicio gestionado, AWS Data Pipeline está protegido por los procedimientos de seguridad de red AWS global que se describen en el documento técnico Amazon Web Services: Overview of Security Processes.

Utiliza las llamadas a la API AWS publicadas para acceder a AWS Data Pipeline través de la red. Los clientes deben ser compatibles con la seguridad de la capa de transporte (TLS) 1.0 o una versión posterior. Recomendamos TLS 1.2 o una versión posterior. Los clientes también deben ser compatibles con conjuntos de cifrado con confidencialidad directa total (PFS) tales como Ephemeral Diffie-Hellman (DHE) o Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

Además, las solicitudes deben estar firmadas mediante un ID de clave de acceso y una clave de acceso secreta que esté asociada a una entidad principal de IAM. También puedes utilizar AWS Security Token Service (AWS STS) para generar credenciales de seguridad temporales para firmar solicitudes.

Análisis de configuración y vulnerabilidad en AWS Data Pipeline

La configuración y los controles de TI son una responsabilidad compartida entre usted AWS y usted, nuestro cliente. Para obtener más información, consulte el modelo de responsabilidad AWS compartida.

Tutoriales

Los siguientes tutoriales le guiarán paso a paso por el proceso de creación y uso de canalizaciones con AWS Data Pipeline.

Tutoriales

- Procesar datos utilizando Amazon EMR con Hadoop Streaming
- Copiar datos CSV entre buckets de Amazon S3 mediante AWS Data Pipeline
- Exportar datos de MySQL a Amazon S3 con la AWS Data Pipeline
- Copiar datos a Amazon Redshift con AWS Data Pipeline

Procesar datos utilizando Amazon EMR con Hadoop Streaming

Puede utilizar AWS Data Pipeline para administrar clústeres de Amazon EMR. Con AWS Data Pipeline, puede especificar las condiciones previas que se deben cumplir antes de lanzar el clúster (por ejemplo, garantizar que los datos de hoy se hayan cargado en Amazon S3), una programación para ejecutar repetidamente el clúster y la configuración de clúster que se debe utilizar. En el siguiente tutorial se describen los pasos que ha de seguir para lanzar un clúster sencillo.

En este tutorial, creará una canalización para que un clúster sencillo de Amazon EMR ejecute un trabajo de Hadoop Streaming preexistente proporcionado por Amazon EMR y envíe una notificación de Amazon SNS una vez que la tarea se complete correctamente. Para esta tarea, puede utilizar el recurso del clúster de Amazon EMR proporcionado por AWS Data Pipeline. La aplicación de ejemplo se denomina WordCount y también se puede ejecutar manualmente desde la consola de Amazon EMR. Tenga en cuenta que los clústeres generados por AWS Data Pipeline en su nombre se muestran en la consola de Amazon EMR y se facturan a su cuenta de AWS.

Objetos de canalización

La canalización usa los siguientes objetos:

EmrActivity

Define el trabajo que se debe realizar en la canalización (ejecutar un trabajo de Hadoop Streaming preexistente proporcionado por Amazon EMR).

EmrCluster

Recurso que AWS Data Pipeline utiliza para llevar a cabo esta actividad.

Un clúster es un conjunto de instancias Amazon EC2. AWS Data Pipeline inicia el clúster y, a continuación, lo termina una vez finalizada la tarea.

Schedule

Fecha de inicio, hora y duración de esta actividad. De forma opcional, puede especificar la fecha y hora de finalización.

SnsAlarm

Envía una notificación de Amazon SNS al tema especificado una vez que la tarea finaliza correctamente.

Contenido

Antes de empezar

Asegúrese de haber completado los pasos siguientes.

- Completar las tareas de Configurándose para AWS Data Pipeline.
- (Opcional) Configurar una VPC para el clúster y un grupo de seguridad para la VPC.
- Crear un tema para enviar notificaciones por correo electrónico y anotar el nombre de recurso de Amazon (ARN) del tema. Para obtener más información, consulte Creación de un tema en Guía de introducción de Amazon Simple Notification Service.

Lanzar un clúster mediante la línea de comando

Si ejecuta periódicamente un clúster de Amazon EMR para analizar registros de web o realizar análisis de datos científicos, puede utilizar AWS Data Pipeline para administrar los clústeres de Amazon EMR. Con AWS Data Pipeline, puede especificar las condiciones previas que se deben cumplir antes de que se lance el clúster (por ejemplo, garantizar que los datos de hoy se hayan cargado en Amazon S3). Este tutorial le mostrará cómo lanzar un clúster, que puede ser un modelo

para una canalización sencilla basada en Amazon EMR o como parte de una canalización más compleja.

Requisitos previos

Antes de poder utilizar la CLI; debe llevar a cabo los pasos siguientes:

1. Instale y configure la interfaz de la línea de comandos (CLI). Para obtener más información, consulte Acceso a AWS Data Pipeline.
2. Asegúrese de que existan los roles de IAM denominados DataPipelineDefaultRole y DataPipelineDefaultResourceRole. La consola AWS Data Pipeline crea estos roles automáticamente. Si no ha utilizado la consola AWS Data Pipeline al menos una vez, debe crear estos roles manualmente. Para obtener más información, consulte Funciones de IAM para AWS Data Pipeline.

Tareas

- Creación del archivo de definición de canalización
- Actualización y activación de la definición de la canalización
- Supervisar las ejecuciones de la canalización

Creación del archivo de definición de canalización

El código siguiente es el archivo de definición de canalización para un clúster sencillo de Amazon EMR que ejecuta un trabajo de Hadoop Streaming existente proporcionado por Amazon EMR. Esta aplicación de ejemplo se denomina WordCount y también puede ejecutarla desde la consola de Amazon EMR.

Copie este código en un archivo de texto y guárdelo como `MyEmrPipelineDefinition.json`. Debe sustituir la ubicación del bucket de Amazon S3 por el nombre de un bucket de Amazon S3 de su propiedad. También debe sustituir las fechas de inicio y final. Para lanzar clústeres de forma inmediata, establezca `startDateTime` en la fecha de un día del pasado y `endDateTime` en un día del futuro. AWS Data Pipeline comenzará entonces a lanzar los clústeres "vencidos" inmediatamente, en un intento de solucionar lo que percibe como un atasco de trabajo. Esta reposición significa que no es necesario esperar una hora para ver cómo AWS Data Pipeline lanza su primer clúster.

```
{
```

```
  "objects": [
    {
      "id": "Hourly",
      "type": "Schedule",
      "startDateTime": "2012-11-19T07:48:00",
      "endDateTime": "2012-11-21T07:48:00",
      "period": "1 hours"
    },
    {
      "id": "MyCluster",
      "type": "EmrCluster",
      "masterInstanceType": "m1.small",
      "schedule": {
        "ref": "Hourly"
      }
    },
    {
      "id": "MyEmrActivity",
      "type": "EmrActivity",
      "schedule": {
        "ref": "Hourly"
      },
      "runsOn": {
        "ref": "MyCluster"
      },
      "step": "/home/hadoop/contrib/streaming/hadoop-streaming.jar,-input,s3n://
elasticmapreduce/samples/wordcount/input,-output,s3://myawsbucket/wordcount/
output/#{@scheduledStartTime},-mapper,s3n://elasticmapreduce/samples/wordcount/
wordSplitter.py,-reducer,aggregate"
    }
  ]
}
```

Esta canalización tiene tres objetos:

- `Hourly`, que representa el programa del trabajo. Puede establecer un programa como uno de los campos de una actividad. Cuando lo haga, la actividad se ejecutará de acuerdo con dicho programa o, en este caso, cada hora.
- `MyCluster`, que representa el conjunto de instancias Amazon EC2 utilizadas para ejecutar el clúster. Puede especificar el tamaño y el número de instancias EC2 que se ejecutarán como el clúster. Si no especifica el número de instancias, el clúster se lanzará con dos, un nodo principal y un nodo de tarea. Puede especificar una subred en la que lanzar el clúster. Puede añadir

configuraciones adicionales al clúster, tales como acciones de arranque para cargar software adicional en la AMI proporcionada por Amazon EMR.

- `MyEmrActivity`, que representa el cálculo que se procesará con el clúster. Amazon EMR admite varios tipos de clústeres, entre los que se incluyen streaming, Cascading y Scripted Hive. El campo `runsOn` vuelve a hacer referencia a MyCluster y lo utiliza como especificación para sustentar el clúster.

Actualización y activación de la definición de la canalización

Debe cargar la definición de su canalización y activarla. En los siguientes comandos de ejemplo, reemplace *`pipeline_name`* por una etiqueta para su canalización y *`pipeline_file`* por la ruta completa para el archivo de definición de canalización `.json`.

AWS CLI

Para crear su definición de canalización y activarla, use el siguiente comando create-pipeline. Anote el ID de su canalización, ya que utilizará este valor con la mayoría de los comandos de la CLI.

```
aws datapipeline create-pipeline --name pipeline_name --unique-id token
{
    "pipelineId": "df-00627471SOVYZEXAMPLE"
}
```

Para cargar su definición de la canalización, utilice el comando siguiente: put-pipeline-definition.

```
aws datapipeline put-pipeline-definition --pipeline-id df-00627471SOVYZEXAMPLE --
pipeline-definition file://MyEmrPipelineDefinition.json
```

Si la canalización se valida correctamente, el campo `validationErrors` estará vacío. Debe revisar todas las advertencias.

Para activar la canalización, utilice el siguiente comando activate-pipeline:

```
aws datapipeline activate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE
```

Puede comprobar que su canalización aparece en la lista de canalizaciones mediante el siguiente comando list-pipelines.

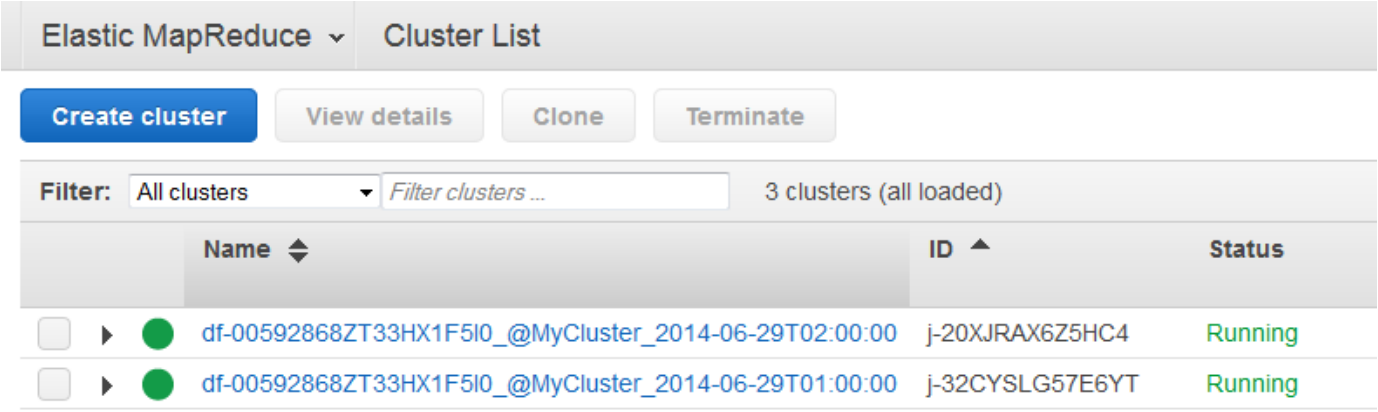
```
aws datapipeline list-pipelines
```

Supervisar las ejecuciones de la canalización

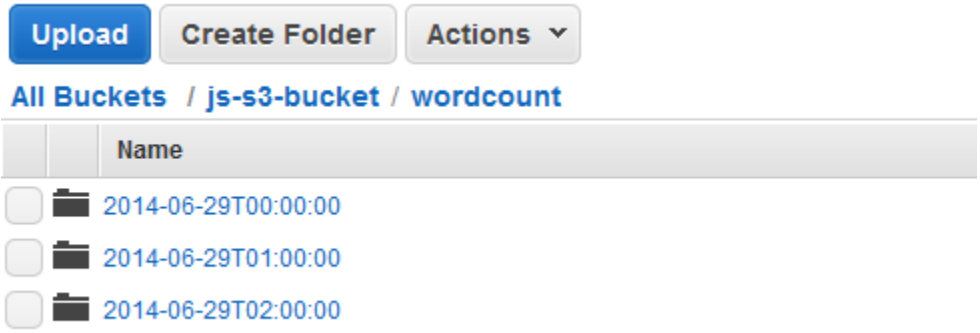
Puede ver los clústeres que lanza AWS Data Pipeline utilizando la consola de Amazon EMR y puede ver la carpeta de salida mediante la consola de Amazon S3.

Para comprobar el progreso de los clústeres que lanza AWS Data Pipeline

1. Abra la consola de Amazon EMR.
2. El nombre de los clústeres generados por AWS Data Pipeline tiene el formato siguiente: *<pipeline-identifier>*_@*<emr-cluster-name>*_*<launch-time>*.



3. Cuando finalice una de las ejecuciones, abra la consola de Amazon S3 y compruebe que la carpeta de salida con marca temporal existe y contiene los resultados esperados del clúster.



Copiar datos CSV entre buckets de Amazon S3 mediante AWS Data Pipeline

Después de leer ¿Qué es () AWS Data Pipeline? y decidir que desea usar AWS Data Pipeline para automatizar el movimiento y transformación de sus datos, es el momento de comenzar a trabajar en

la creación de canalizaciones de datos. Para ayudarle a entender el funcionamiento de AWS Data Pipeline, vamos a seguir una tarea sencilla paso a paso.

En este tutorial se le guía a través del proceso de creación de una canalización de datos para copiar datos de un bucket de Amazon S3 en otro y, a continuación, enviar una notificación de Amazon SNS tras completarse correctamente la actividad de copia. Use una instancia EC2 administrada por AWS Data Pipeline para esta actividad de copia.

Objetos de canalización

La canalización usa los siguientes objetos:

CopyActivity

La actividad que AWS Data Pipeline realiza para esta canalización (copiar datos CSV de un bucket de Amazon S3 en otro).

> **Important**
>
> Existen limitaciones al usar el formato de archivo CSV con `CopyActivity` y `S3DataNode`. Para obtener más información, consulte CopyActivity.

Schedule

La fecha de inicio, la hora y la periodicidad de esta actividad. De forma opcional, puede especificar la fecha y hora de finalización.

Ec2Resource

El recurso (una instancia EC2) que usa AWS Data Pipeline para realizar esta actividad.

S3 DataNode

Los nodos de entrada y salida (buckets de Amazon S3) para esta canalización.

SnsAlarm

La acción que AWS Data Pipeline debe ejecutar al cumplirse las condiciones especificadas (enviar notificaciones de Amazon SNS a un tema una vez finalizada correctamente la tarea).

Contenido

- Copiar datos CSV mediante la línea de comandos

Antes de empezar

Asegúrese de haber completado los pasos siguientes.

- Completar las tareas de Configurándose para AWS Data Pipeline.
- (Opcional) Configure una VPC para la instancia y un grupo de seguridad para la VPC.
- Crear un bucket de Amazon S3 como origen de datos.

  Para obtener más información, consulte Crear un bucket en la Guía del usuario de Amazon Simple Storage Service.
- Cargue sus datos en su bucket de Amazon S3.

  Para obtener más información, consulte Add an Object to a Bucket (Adición de un objeto a un bucket) en la Guía del usuario de Amazon Simple Storage Service.
- Crear otro bucket de Amazon S3 como destino de los datos.
- Crear un tema para enviar notificaciones por correo electrónico y anotar el nombre de recurso de Amazon (ARN) del tema. Para obtener más información, consulte Creación de un tema en Guía de introducción de Amazon Simple Notification Service.
- (Opcional) En este tutorial, se usan las políticas de roles de IAM predeterminadas creadas por AWS Data Pipeline. Si prefiere crear y configurar sus propias políticas de roles de IAM y relaciones de confianza, siga las instrucciones descritas en Funciones de IAM para AWS Data Pipeline.

Copiar datos CSV mediante la línea de comandos

Puede crear y usar canalizaciones para copiar datos de un bucket de Amazon S3 en otro.

Requisitos previos

Debe seguir estos pasos antes de comenzar:

1. Instale y configure la interfaz de la línea de comandos (CLI). Para obtener más información, consulte Acceso a AWS Data Pipeline.
2. Asegúrese de que existan los roles de IAM denominados DataPipelineDefaultRole y DataPipelineDefaultResourceRole. La consola AWS Data Pipeline crea estos roles automáticamente. Si no ha utilizado la consola AWS Data Pipeline al menos una vez, debe crear

estos roles manualmente. Para obtener más información, consulte Funciones de IAM para AWS Data Pipeline.

Tareas

- Definir una canalización en formato JSON
- Cargar y activar la definición de canalización

Definir una canalización en formato JSON

En este escenario de ejemplo, se muestra cómo usar definiciones de canalización JSON y la CLI de AWS Data Pipeline para programar la copia de datos entre dos buckets de Amazon S3 en un tiempo específico. Este es el archivo JSON de definición de la canalización completo seguido de una explicación de cada una de sus secciones.

> **Note**
>
> Le recomendamos que use un editor de texto que pueda ayudarle a comprobar la sintaxis de los archivos con formato JSON y que asigne un nombre al archivo con la extensión .json.

En este ejemplo, para mayor claridad, omitimos los campos opcionales y mostramos únicamente los campos obligatorios. El archivo JSON de la canalización completo en este ejemplo es:

```
{
  "objects": [
    {
      "id": "MySchedule",
      "type": "Schedule",
      "startDateTime": "2013-08-18T00:00:00",
      "endDateTime": "2013-08-19T00:00:00",
      "period": "1 day"
    },
    {
      "id": "S3Input",
      "type": "S3DataNode",
      "schedule": {
        "ref": "MySchedule"
      },
      "filePath": "s3://amzn-s3-demo-bucket/source/inputfile.csv"
```

```
    },
    {
      "id": "S3Output",
      "type": "S3DataNode",
      "schedule": {
        "ref": "MySchedule"
      },
      "filePath": "s3://amzn-s3-demo-bucket/destination/outputfile.csv"
    },
    {
      "id": "MyEC2Resource",
      "type": "Ec2Resource",
      "schedule": {
        "ref": "MySchedule"
      },
      "instanceType": "m1.medium",
      "role": "DataPipelineDefaultRole",
      "resourceRole": "DataPipelineDefaultResourceRole"
    },
    {
      "id": "MyCopyActivity",
      "type": "CopyActivity",
      "runsOn": {
        "ref": "MyEC2Resource"
      },
      "input": {
        "ref": "S3Input"
      },
      "output": {
        "ref": "S3Output"
      },
      "schedule": {
        "ref": "MySchedule"
      }
    }
  ]
}
```

Programación

La canalización define un programa con una fecha de inicio y finalización, junto con un período para determinar con qué frecuencia se ejecuta la actividad en esta canalización.

```
{
  "id": "MySchedule",
  "type": "Schedule",
  "startDateTime": "2013-08-18T00:00:00",
  "endDateTime": "2013-08-19T00:00:00",
  "period": "1 day"
},
```

Nodos de datos Amazon S3

A continuación, el componente de canalización S3DataNode de entrada define una ubicación para los archivos de entrada; en este caso, una ubicación de bucket de Amazon S3. El componente S3DataNode de entrada se define por los siguientes campos:

```
{
  "id": "S3Input",
  "type": "S3DataNode",
  "schedule": {
    "ref": "MySchedule"
  },
  "filePath": "s3://example-bucket/source/inputfile.csv"
},
```

Id

El nombre definido por el usuario para la ubicación de entrada (una etiqueta solo con fines de referencia).

Tipo

El tipo de componente de canalización, que es “S3DataNode” para que coincida con la ubicación donde residen los datos, en un bucket de Amazon S3.

Programación

Una referencia al componente de programación que creamos en las líneas anteriores del archivo JSON etiquetado "MySchedule".

Ruta

La ruta a los datos asociados al nodo de datos. La sintaxis de un nodo de datos viene determinada por su tipo. Por ejemplo, la sintaxis de una ruta de Amazon S3 sigue una sintaxis diferente que es adecuada para una tabla de la base de datos.

A continuación, el componente S3DataNode de salida define la ubicación de destino de salida para los datos. Sigue el mismo formato que el componente S3DataNode de entrada, excepto el nombre del componente y una ruta diferente para indicar el archivo de destino.

```
{
  "id": "S3Output",
  "type": "S3DataNode",
  "schedule": {
    "ref": "MySchedule"
  },
  "filePath": "s3://example-bucket/destination/outputfile.csv"
},
```

Recurso

Esta es una definición del recurso informático que realiza la operación de copia. En este ejemplo, AWS Data Pipeline debe crear automáticamente una instancia EC2 para realizar la tarea de copia y terminar el recurso tras completarse la tarea. Los campos definidos aquí controlan la creación y función de la instancia EC2 que realiza el trabajo. El componente Resource se define por los siguientes campos:

```
{
  "id": "MyEC2Resource",
  "type": "Ec2Resource",
  "schedule": {
    "ref": "MySchedule"
  },
  "instanceType": "m1.medium",
  "role": "DataPipelineDefaultRole",
  "resourceRole": "DataPipelineDefaultResourceRole"
},
```

Id

El nombre definido por el usuario para el programa de canalización, que es una etiqueta solo con fines de referencia.

Tipo

El tipo de recurso informático para realizar el trabajo; en este caso, una instancia EC2. Hay otros tipos de recursos disponibles, como un tipo EmrCluster.

Programación

El programa en el que desea crear este recurso informático.

instanceType

El tamaño de la instancia EC2 que se creará. Asegúrese de que establece el tamaño adecuado de la instancia EC2 que mejor se adapte a la carga del trabajo que desea realizar con AWS Data Pipeline. En este caso, establecemos una instancia EC2 m1.medium. Para obtener más información acerca de los diferentes tipos de instancia y cuándo usar cada uno, consulte el tema Tipos de instancias de Amazon EC2 en http://aws.amazon.com/ec2/instance-types/.

Rol

El rol de IAM de la cuenta que tiene acceso a recursos, como el acceso a un bucket de Amazon S3 para recuperar datos.

resourceRole

El rol de IAM de la cuenta que crea recursos, como la creación y configuración de una instancia EC2 en su nombre. resourceRole

Actividad

La última sección del archivo JSON es la definición de la actividad que representa el trabajo que se realizará. En este ejemplo, se usa `CopyActivity` para copiar datos de un archivo CSV en un bucket http://aws.amazon.com/ec2/instance-types/ a otro. El componente `CopyActivity` se define por los siguientes campos:

```
{
  "id": "MyCopyActivity",
  "type": "CopyActivity",
  "runsOn": {
    "ref": "MyEC2Resource"
  },
  "input": {
    "ref": "S3Input"
  },
  "output": {
    "ref": "S3Output"
  },
  "schedule": {
```

```
    "ref": "MySchedule"
  }
}
```

Id

El nombre definido por el usuario para la actividad, que es una etiqueta solo con fines de referencia.

Tipo

El tipo de actividad que se realizará, como MyCopyActivity.

runsOn

El recurso informático que realiza el trabajo que define esta actividad. En este ejemplo, proporcionamos una referencia a la instancia de EC2 definidos anteriormente. El uso del campo `runsOn` conlleva que AWS Data Pipeline cree la instancia EC2 automáticamente. El campo `runsOn` indica que el recurso existe en la infraestructura de AWS, mientras que el valor de `workerGroup` indica que desea usar sus propios recursos locales para realizar el trabajo.

Input

La ubicación de los datos que copiar.

Output

Los datos de la ubicación de destino.

Programación

La programación en la que ejecutar esta actividad.

Cargar y activar la definición de canalización

Debe cargar la definición de su canalización y activarla. En los siguientes comandos de ejemplo, reemplace *`pipeline_name`* por una etiqueta para su canalización y *`pipeline_file`* por la ruta completa para el archivo de definición de canalización `.json`.

AWS CLI

Para crear su definición de canalización y activarla, use el siguiente comando create-pipeline. Anote el ID de su canalización, ya que utilizará este valor con la mayoría de los comandos de la CLI.

```
aws datapipeline create-pipeline --name pipeline_name --unique-id token
{
    "pipelineId": "df-00627471SOVYZEXAMPLE"
}
```

Para cargar su definición de la canalización, utilice el comando siguiente: put-pipeline-definition.

```
aws datapipeline put-pipeline-definition --pipeline-id df-00627471SOVYZEXAMPLE --
pipeline-definition file://MyEmrPipelineDefinition.json
```

Si la canalización se valida correctamente, el campo `validationErrors` estará vacío. Debe revisar todas las advertencias.

Para activar la canalización, utilice el siguiente comando activate-pipeline:

```
aws datapipeline activate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE
```

Puede comprobar que su canalización aparece en la lista de canalizaciones mediante el siguiente comando list-pipelines.

```
aws datapipeline list-pipelines
```

Exportar datos de MySQL a Amazon S3 con la AWS Data Pipeline

Este tutorial le guiará a lo largo del proceso de creación de una canalización de datos para copiar datos (filas) de una tabla de una base de datos de MySQL en un archivo CSV (valores separados por comas) en un bucket de Amazon S3 y, a continuación, enviar una notificación de Amazon SNS después de que la actividad de copia se realice correctamente. Usará una instancia EC2 proporcionada por AWS Data Pipeline para realizar esta actividad de copia.

Objetos de canalización

La canalización usa los siguientes objetos:

- CopyActivity
- Ec2Resource
- MySqlDataNode

- S3 DataNode
- SnsAlarm

Contenido

Antes de empezar

Asegúrese de haber completado los pasos siguientes.

- Completar las tareas de Configurándose para AWS Data Pipeline.
- (Opcional) Configure una VPC para la instancia y un grupo de seguridad para la VPC.
- Cree un bucket de Amazon S3 como salida de datos.

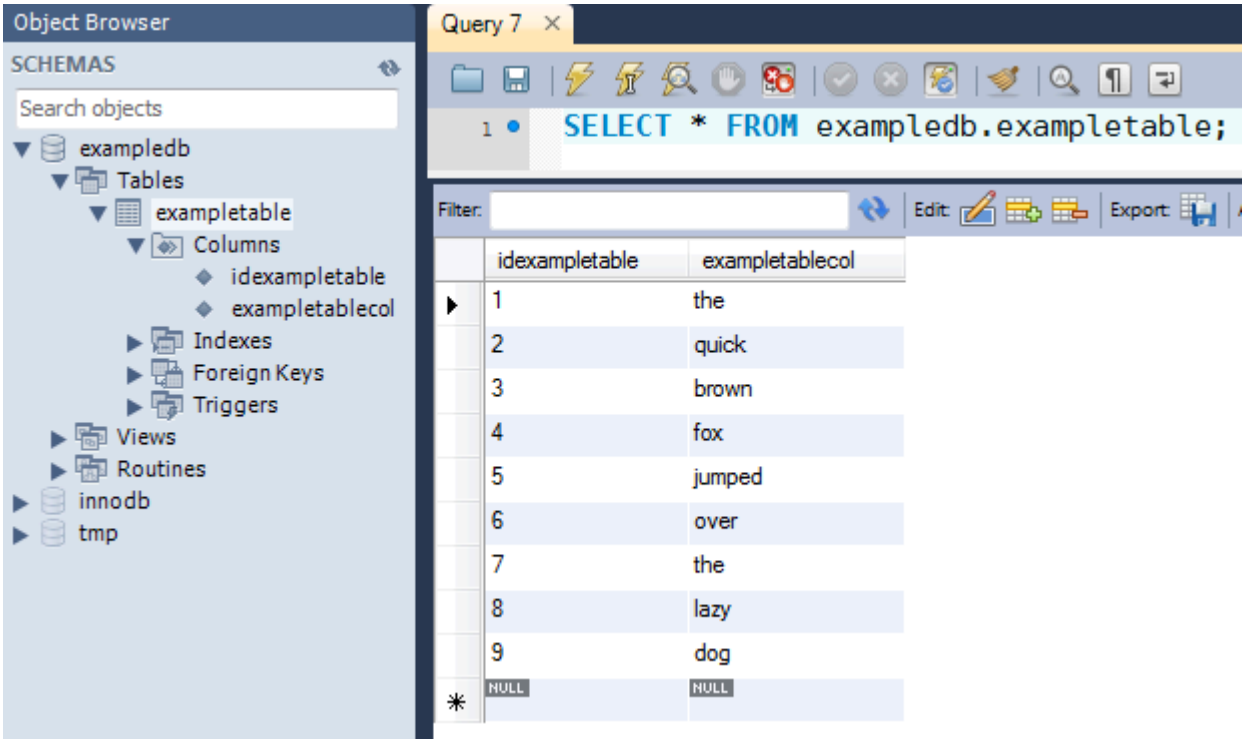
  Para obtener más información, consulte Creación de un bucket en la Guía del usuario de Amazon Simple Storage Service.
- Cree y lance una instancia de base de datos de MySQL como origen de datos.

  Para obtener más información, consulte la sección sobre Almacenamiento de instancias de base de datos en la Guía del usuario de Introducción a Amazon RDS. Cuando ya tenga una instancia de Amazon RDS, consulte Crear una tabla en la documentación de MySQL.

  > **Note**
  >
  > Anote el nombre de usuario y la contraseña que utilizó para crear la instancia de MySQL. Una vez lanzada la instancia de base de datos de MySQL, anote el punto de enlace de la instancia. Necesitará esta información más tarde.

- Conéctese a su instancia de base de datos de MySQL, cree una tabla y, a continuación, añada valores de datos de prueba a la tabla recién creada.

  Para fines de ilustración, hemos creado este tutorial utilizando una tabla de MySQL con la siguiente configuración y datos de ejemplo. La siguiente captura de pantalla pertenece a MySQL Workbench 5.2 CE:



Para obtener más información, consulte Crear una tabla en la documentación de MySQL y la página del producto de MySQL Workbench.

- Crear un tema para enviar notificaciones por correo electrónico y anotar el nombre de recurso de Amazon (ARN) del tema. Para obtener más información, consulte Creación de un tema en la Guía de introducción de Amazon Simple Notification Service.
- (Opcional) En este tutorial, se usan las políticas de roles de IAM predeterminadas creadas por AWS Data Pipeline. Si prefiere crear y configurar su propia política de roles de IAM y relaciones de confianza, siga las instrucciones descritas en Funciones de IAM para AWS Data Pipeline.

Copia de datos de MySQL mediante la línea de comandos

Cree una canalización para copiar datos de una tabla de MySQL en un archivo en un bucket de Amazon S3.

Requisitos previos

Debe seguir estos pasos antes de comenzar:

1. Instale y configure la interfaz de la línea de comandos (CLI). Para obtener más información, consulte Acceso a AWS Data Pipeline.

2. Asegúrese de que existan los roles de IAM denominados DataPipelineDefaultRole y DataPipelineDefaultResourceRole. La consola AWS Data Pipeline crea estos roles automáticamente. Si no ha utilizado la consola AWS Data Pipeline al menos una vez, debe crear estos roles manualmente. Para obtener más información, consulte Funciones de IAM para AWS Data Pipeline.
3. Configure un bucket de Amazon S3 y una instancia de Amazon RDS. Para obtener más información, consulte Antes de empezar.

Tareas

- Definir una canalización en formato JSON
- Cargar y activar la definición de canalización

Definir una canalización en formato JSON

Este escenario de ejemplo muestra cómo utilizar definiciones de la canalización de JSON y la CLI de AWS Data Pipeline para copiar datos (filas) de una tabla en una base de datos MySQL en un archivo CSV (valores separados por comas) en un bucket de Amazon S3 en un intervalo de tiempo especificado.

Este es el archivo JSON de definición de la canalización completo seguido de una explicación de cada una de sus secciones.

> **Note**
>
> Le recomendamos que use un editor de texto que pueda ayudarle a comprobar la sintaxis de los archivos con formato JSON y que asigne un nombre al archivo con la extensión .json.

```
{
  "objects": [
    {
      "id": "ScheduleId113",
      "startDateTime": "2013-08-26T00:00:00",
      "name": "My Copy Schedule",
      "type": "Schedule",
      "period": "1 Days"
    },
    {
```

```
      "id": "CopyActivityId112",
      "input": {
        "ref": "MySqlDataNodeId115"
      },
      "schedule": {
        "ref": "ScheduleId113"
      },
      "name": "My Copy",
      "runsOn": {
        "ref": "Ec2ResourceId116"
      },
      "onSuccess": {
        "ref": "ActionId1"
      },
      "onFail": {
        "ref": "SnsAlarmId117"
      },
      "output": {
        "ref": "S3DataNodeId114"
      },
      "type": "CopyActivity"
    },
    {
      "id": "S3DataNodeId114",
      "schedule": {
        "ref": "ScheduleId113"
      },
      "filePath": "s3://amzn-s3-demo-bucket/rds-output/output.csv",
      "name": "My S3 Data",
      "type": "S3DataNode"
    },
    {
      "id": "MySqlDataNodeId115",
      "username": "my-username",
      "schedule": {
        "ref": "ScheduleId113"
      },
      "name": "My RDS Data",
      "*password": "my-password",
      "table": "table-name",
      "connectionString": "jdbc:mysql://your-sql-instance-name.id.region-
name.rds.amazonaws.com:3306/database-name",
      "selectQuery": "select * from #{table}",
      "type": "SqlDataNode"
```

```
    },
    {
      "id": "Ec2ResourceId116",
      "schedule": {
        "ref": "ScheduleId113"
      },
      "name": "My EC2 Resource",
      "role": "DataPipelineDefaultRole",
      "type": "Ec2Resource",
      "resourceRole": "DataPipelineDefaultResourceRole"
    },
    {
      "message": "This is a success message.",
      "id": "ActionId1",
      "subject": "RDS to S3 copy succeeded!",
      "name": "My Success Alarm",
      "role": "DataPipelineDefaultRole",
      "topicArn": "arn:aws:sns:us-east-1:123456789012:example-topic",
      "type": "SnsAlarm"
    },
    {
      "id": "Default",
      "scheduleType": "timeseries",
      "failureAndRerunMode": "CASCADE",
      "name": "Default",
      "role": "DataPipelineDefaultRole",
      "resourceRole": "DataPipelineDefaultResourceRole"
    },
    {
      "message": "There was a problem executing #{node.name} at for period
 #{node.@scheduledStartTime} to #{node.@scheduledEndTime}",
      "id": "SnsAlarmId117",
      "subject": "RDS to S3 copy failed",
      "name": "My Failure Alarm",
      "role": "DataPipelineDefaultRole",
      "topicArn": "arn:aws:sns:us-east-1:123456789012:example-topic",
      "type": "SnsAlarm"
    }
  ]
}
```

Nodo de datos MySQL

El componente de entrada MySqlDataNode de la canalización define una ubicación para los datos de entrada; en este caso, una instancia de Amazon RDS. El componente MySqlDataNode de la entrada se define por medio de los siguientes campos:

```
{
  "id": "MySqlDataNodeId115",
  "username": "my-username",
  "schedule": {
    "ref": "ScheduleId113"
  },
  "name": "My RDS Data",
  "*password": "my-password",
  "table": "table-name",
  "connectionString": "jdbc:mysql://your-sql-instance-name.id.region-
name.rds.amazonaws.com:3306/database-name",
  "selectQuery": "select * from #{table}",
  "type": "SqlDataNode"
},
```

Id

El nombre definido por el usuario, que es una etiqueta solo con fines de referencia.

Nombre de usuario

El nombre de usuario de la cuenta de la base de datos que tiene permisos suficientes para recuperar los datos de la tabla de base de datos. Sustituya *my-username* por el nombre de su cuenta de usuario.

Programación

Una referencia al componente de programación que creamos en las líneas anteriores del archivo JSON.

Nombre

El nombre definido por el usuario, que es una etiqueta solo con fines de referencia.

*Password

La contraseña de la cuenta de la base de datos con el asterisco como prefijo para indicar que AWS Data Pipeline debe cifrar el valor de la contraseña. Sustituya *my-password* por la

contraseña correcta para su cuenta de usuario. El campo de la contraseña está precedido por el carácter especial del asterisco. Para obtener más información, consulte Caracteres especiales.

Tabla

El nombre de la tabla de base datos que contiene los datos que se van a copiar. Sustituya *table-name* por el nombre de la tabla de base de datos.

connectionString

La cadena de conexión JDBC para el objeto CopyActivity que se va a conectar a la base de datos.

selectQuery

Una consulta SQL SELECT válida que especifica qué datos se van a copiar de la tabla de base de datos. Tenga en cuenta que `#{table}` es una expresión que reutiliza el nombre de la tabla proporcionado por la variable "table" en las líneas anteriores del archivo JSON.

Tipo

El tipo SqlDataNode, que es una instancia de Amazon RDS que utiliza MySQL en este ejemplo.

> **Note**
>
> El tipo MySqlDataNode está obsoleto. Aunque puede seguir usando MySqlDataNode, le recomendamos que utilice SqlDataNode.

Nodo de datos Amazon S3

A continuación, el componente de canalización S3Output define una ubicación para el archivo de salida; en este caso, un archivo CSV en una ubicación de bucket de Amazon S3. El componente S3DataNode de salida se define por los siguientes campos:

```
{
  "id": "S3DataNodeId114",
  "schedule": {
    "ref": "ScheduleId113"
  },
  "filePath": "s3://amzn-s3-demo-bucket/rds-output/output.csv",
  "name": "My S3 Data",
  "type": "S3DataNode"
},
```

Id

El ID definido por el usuario, que es una etiqueta solo con fines de referencia.

Programación

Una referencia al componente de programación que creamos en las líneas anteriores del archivo JSON.

filePath

La ruta a los datos asociados al nodo de datos, que es un archivo de salida CSV en este ejemplo.

Nombre

El nombre definido por el usuario, que es una etiqueta solo con fines de referencia.

Tipo

El tipo de objeto de canalización, que es S3DataNode para que coincida con la ubicación donde residen los datos, en un bucket de Amazon S3.

Recurso

Esta es una definición del recurso informático que realiza la operación de copia. En este ejemplo, AWS Data Pipeline debe crear automáticamente una instancia EC2 para realizar la tarea de copia y terminar el recurso tras completarse la tarea. Los campos definidos aquí controlan la creación y función de la instancia EC2 que realiza el trabajo. El componente Resource se define por los siguientes campos:

```
{
  "id": "Ec2ResourceId116",
  "schedule": {
    "ref": "ScheduleId113"
  },
  "name": "My EC2 Resource",
  "role": "DataPipelineDefaultRole",
  "type": "Ec2Resource",
  "resourceRole": "DataPipelineDefaultResourceRole"
},
```

Id

El ID definido por el usuario, que es una etiqueta solo con fines de referencia.

Programación

El programa en el que desea crear este recurso informático.

Nombre

El nombre definido por el usuario, que es una etiqueta solo con fines de referencia.

Rol

El rol de IAM de la cuenta que tiene acceso a recursos, como el acceso a un bucket de Amazon S3 para recuperar datos.

Tipo

El tipo de recurso informático para realizar el trabajo; en este caso, una instancia EC2. Hay otros tipos de recursos disponibles, como un tipo EmrCluster.

resourceRole

El rol de IAM de la cuenta que crea recursos, como la creación y configuración de una instancia EC2 en su nombre. resourceRole

Actividad

La última sección del archivo JSON es la definición de la actividad que representa el trabajo que se realizará. En este caso, utilizamos un componente CopyActivity para copiar datos de un archivo en un bucket de Amazon S3 en otro archivo. El componente CopyActivity se define por los siguientes campos:

```
{
  "id": "CopyActivityId112",
  "input": {
    "ref": "MySqlDataNodeId115"
  },
  "schedule": {
    "ref": "ScheduleId113"
  },
  "name": "My Copy",
  "runsOn": {
    "ref": "Ec2ResourceId116"
  },
  "onSuccess": {
    "ref": "ActionId1"
  },
```

```
  "onFail": {
    "ref": "SnsAlarmId117"
  },
  "output": {
    "ref": "S3DataNodeId114"
  },
  "type": "CopyActivity"
},
```

Id

El ID definido por el usuario, que es una etiqueta solo con fines de referencia.

Input

La ubicación de los datos de MySQL que se van a copiar.

Programación

La programación en la que ejecutar esta actividad.

Nombre

El nombre definido por el usuario, que es una etiqueta solo con fines de referencia.

runsOn

El recurso informático que realiza el trabajo que define esta actividad. En este ejemplo, proporcionamos una referencia a la instancia de EC2 definidos anteriormente. El uso del campo `runs0n` conlleva que AWS Data Pipeline cree la instancia EC2 automáticamente. El campo `runs0n` indica que el recurso existe en la infraestructura de AWS, mientras que el valor de workerGroup indica que desea usar sus propios recursos locales para realizar el trabajo.

onSuccess

La SnsAlarm que se va a enviar si la actividad se realiza de forma correcta.

onFail

La SnsAlarm que se va a enviar si la actividad no se realiza de forma correcta.

Output

La ubicación en Amazon S3 del archivo de salida CSV.

Tipo

El tipo de actividad que se va a realizar.

Cargar y activar la definición de canalización

Debe cargar la definición de su canalización y activarla. En los siguientes comandos de ejemplo, reemplace *pipeline_name* por una etiqueta para su canalización y *pipeline_file* por la ruta completa para el archivo de definición de canalización `.json`.

AWS CLI

Para crear su definición de canalización y activarla, use el siguiente comando create-pipeline. Anote el ID de su canalización, ya que utilizará este valor con la mayoría de los comandos de la CLI.

```
aws datapipeline create-pipeline --name pipeline_name --unique-id token
{
    "pipelineId": "df-00627471SOVYZEXAMPLE"
}
```

Para cargar su definición de la canalización, utilice el comando siguiente: put-pipeline-definition.

```
aws datapipeline put-pipeline-definition --pipeline-id df-00627471SOVYZEXAMPLE --
pipeline-definition file://MyEmrPipelineDefinition.json
```

Si la canalización se valida correctamente, el campo `validationErrors` estará vacío. Debe revisar todas las advertencias.

Para activar la canalización, utilice el siguiente comando activate-pipeline:

```
aws datapipeline activate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE
```

Puede comprobar que su canalización aparece en la lista de canalizaciones mediante el siguiente comando list-pipelines.

```
aws datapipeline list-pipelines
```

Copiar datos a Amazon Redshift con AWS Data Pipeline

En este tutorial, se le enseña el proceso de creación de una canalización que transfiere datos de Amazon S3 a Amazon Redshift periódicamente mediante la plantilla Copy to Redshift de la consola de AWS Data Pipeline, o mediante un archivo de definición de canalización con la CLI de AWS Data Pipeline.

Amazon S3 es un servicio web que le permite almacenar datos en la nube. Para obtener más información, consulte la Guía del usuario de Amazon Simple Storage Service.

Amazon Redshift es un servicio de almacenamiento de datos en la nube. Para obtener más información, consulte la Guía de administración de Amazon Redshift.

Este tutorial tiene varios requisitos previos. Tras completar los siguientes pasos, puede continuar el tutorial mediante la consola o la CLI.

Contenido

Antes de comenzar: configurar las opciones de COPY y cargar datos

Antes de copiar datos a Amazon Redshift dentro de AWS Data Pipeline, asegúrese de:

- Cargar datos desde Amazon S3.
- Configure la actividad `COPY` en Amazon Redshift.

Una vez que tenga estas opciones en funcionamiento y que haya completado correctamente una carga de datos, transfiera estas opciones a AWS Data Pipeline para realizar la copia en este servicio.

Para conocer las opciones de `COPY`, consulte COPY en la Guía de desarrollador de base de datos de Amazon Redshift.

Para obtener información sobre los pasos para cargar los datos desde Amazon S3, consulte Carga de datos de Amazon S3 en la Guía de desarrollador de base de datos de Amazon Redshift.

Por ejemplo, el siguiente comando SQL en Amazon Redshift crea una nueva tabla denominada `LISTING` y copia datos de muestra de un bucket disponible públicamente en Amazon S3.

Reemplace el `<iam-role-arn>` y la región por la suya propia.

Para obtener más información acerca de este ejemplo, consulte Cargar datos de muestra de Amazon S3 en la Guía de introducción de Amazon Redshift.

```
create table listing(
```

```
 listid integer not null distkey,
 sellerid integer not null,
 eventid integer not null,
 dateid smallint not null  sortkey,
 numtickets smallint not null,
 priceperticket decimal(8,2),
 totalprice decimal(8,2),
 listtime timestamp);

copy listing from 's3://awssampledbuswest2/tickit/listings_pipe.txt'
credentials 'aws_iam_role=<iam-role-arn>'
delimiter '|' region 'us-west-2';
```

Configurar la canalización, crear un grupo de seguridad y crear un clúster de Amazon Redshift

Para configurar el tutorial

1. Completar las tareas de Configurándose para AWS Data Pipeline.
2. Crear un grupo de seguridad.
   a. Abra la consola de Amazon EC2.
   b. En el panel de navegación, haga clic en Grupos de seguridad.
   c. Haga clic en Crear grupos de seguridad.
   d. Especifique un nombre y una descripción para el grupo de seguridad.
   e. [EC2-Classic] Seleccione No VPC para VPC.
   f. [EC2-VPC] Seleccione el ID de su VPC para VPC.
   g. Haga clic en Crear.
3. [EC2-Classic] Cree un grupo de seguridad de clúster de Amazon Redshift y especifique el grupo de seguridad de Amazon EC2.
   a. Abra la consola de Amazon Redshift.
   b. En el panel de navegación, haga clic en Grupos de seguridad.
   c. Haga clic en Create Cluster Security Group.
   d. En el cuadro de diálogo Create Cluster Security Group, especifique un nombre y una descripción de grupo de seguridad de clúster.
   e. Haga clic en el nombre del nuevo grupo de seguridad de clúster.

f. Haga clic en Add Connection Type.

g. En el cuadro de diálogo Add Connection Type, seleccione EC2 Security Group en Connection Type, seleccione el grupo de seguridad que creó en EC2 Security Group Name y, a continuación, haga clic en Authorize.

4. [EC2-VPC] Cree un grupo de seguridad de clúster de Amazon Redshift y especifique el grupo de seguridad de VPC.

   a. Abra la consola de Amazon EC2.

   b. En el panel de navegación, haga clic en Grupos de seguridad.

   c. Haga clic en Crear grupos de seguridad.

   d. En el cuadro de diálogo Create Security Group, especifique un nombre y una descripción del grupo de seguridad y seleccione el ID de su VPC para VPC.

   e. Haga clic en Add Rule. Especifique el tipo, el protocolo y el rango de puertos, y comience a escribir el ID del grupo de seguridad en Source. Seleccione el grupo de seguridad que creó en el segundo paso.

   f. Haga clic en Crear.

5. El siguiente es un resumen de los pasos.

   Si dispone de un clúster de Amazon Redshift existente, anote el ID del clúster.

   Para crear un nuevo clúster y cargar datos de muestra, siga los pasos en Introducción a Amazon Redshift. Para obtener más información sobre la creación de clústeres, consulte Creación de un clúster en la Guía de administración de clústeres de Amazon Redshift.

   a. Abra la consola de Amazon Redshift.

   b. Haga clic en Launch Cluster.

   c. Proporcione los detalles obligatorios para su clúster y, a continuación, haga clic en Continue.

   d. Proporcione la configuración de nodos y, a continuación, haga clic en Continue.

   e. En la página de información de configuración adicional, seleccione el grupo de seguridad de clúster que creó y, a continuación, haga clic en Continue.

   f. Revise las especificaciones para su clúster y, a continuación, haga clic en Launch Cluster.

Copiar datos en Amazon Redshift mediante la línea de comandos

En este tutorial, se muestra cómo copiar datos de Amazon S3 en Amazon Redshift. Creará una nueva tabla en Amazon Redshift y, a continuación, usará AWS Data Pipeline para transferir datos a esta tabla desde un bucket de Amazon S3 público, el cual contiene datos de entrada de ejemplo en formato CSV. Los registros se guardan en un bucket de Amazon S3 de su propiedad.

Amazon S3 es un servicio web que le permite almacenar datos en la nube. Para obtener más información, consulte la Guía del usuario de Amazon Simple Storage Service. Amazon Redshift es un servicio de almacenamiento de datos en la nube. Para obtener más información, consulte la Guía de administración de Amazon Redshift.

Requisitos previos

Debe seguir estos pasos antes de comenzar:

1. Instale y configure la interfaz de la línea de comandos (CLI). Para obtener más información, consulte Acceso a AWS Data Pipeline.
2. Asegúrese de que existan los roles de IAM denominados DataPipelineDefaultRole y DataPipelineDefaultResourceRole. La consola AWS Data Pipeline crea estos roles automáticamente. Si no ha utilizado la consola AWS Data Pipeline al menos una vez, debe crear estos roles manualmente. Para obtener más información, consulte Funciones de IAM para AWS Data Pipeline.
3. Configure el comando `COPY` en Amazon Redshift, ya que tendrá que disponer de estas mismas opciones en funcionamiento al realizar la copia en AWS Data Pipeline. Para obtener más información, consulte Antes de comenzar: configurar las opciones de COPY y cargar datos.
4. Configuración de una base de datos de Amazon Redshift Para obtener más información, consulte Configurar la canalización, crear un grupo de seguridad y crear un clúster de Amazon Redshift.

Tareas

- Definir una canalización en formato JSON
- Cargar y activar la definición de canalización

Definir una canalización en formato JSON

En este escenario de ejemplo, se muestra cómo copiar datos de un bucket de Amazon S3 en Amazon Redshift.

Este es el archivo JSON de definición de la canalización completo seguido de una explicación de cada una de sus secciones. Le recomendamos que use un editor de texto que pueda ayudarle a comprobar la sintaxis de los archivos con formato JSON y asigne un nombre al archivo mediante la extensión de archivo `.json`.

```
{
  "objects": [
    {
      "id": "CSVId1",
      "name": "DefaultCSV1",
      "type": "CSV"
    },
    {
      "id": "RedshiftDatabaseId1",
      "databaseName": "dbname",
      "username": "user",
      "name": "DefaultRedshiftDatabase1",
      "*password": "password",
      "type": "RedshiftDatabase",
      "clusterId": "redshiftclusterId"
    },
    {
      "id": "Default",
      "scheduleType": "timeseries",
      "failureAndRerunMode": "CASCADE",
      "name": "Default",
      "role": "DataPipelineDefaultRole",
      "resourceRole": "DataPipelineDefaultResourceRole"
    },
    {
      "id": "RedshiftDataNodeId1",
      "schedule": {
        "ref": "ScheduleId1"
      },
      "tableName": "orders",
      "name": "DefaultRedshiftDataNode1",
```

```
      "createTableSql": "create table StructuredLogs (requestBeginTime CHAR(30)
PRIMARY KEY DISTKEY SORTKEY, requestEndTime CHAR(30), hostname CHAR(100), requestDate
varchar(20));",
      "type": "RedshiftDataNode",
      "database": {
        "ref": "RedshiftDatabaseId1"
      }
    },
    {
      "id": "Ec2ResourceId1",
      "schedule": {
        "ref": "ScheduleId1"
      },
      "securityGroups": "MySecurityGroup",
      "name": "DefaultEc2Resource1",
      "role": "DataPipelineDefaultRole",
      "logUri": "s3://myLogs",
      "resourceRole": "DataPipelineDefaultResourceRole",
      "type": "Ec2Resource"
    },
    {
      "id": "ScheduleId1",
      "startDateTime": "yyyy-mm-ddT00:00:00",
      "name": "DefaultSchedule1",
      "type": "Schedule",
      "period": "period",
      "endDateTime": "yyyy-mm-ddT00:00:00"
    },
    {
      "id": "S3DataNodeId1",
      "schedule": {
        "ref": "ScheduleId1"
      },
      "filePath": "s3://datapipeline-us-east-1/samples/hive-ads-samples.csv",
      "name": "DefaultS3DataNode1",
      "dataFormat": {
        "ref": "CSVId1"
      },
      "type": "S3DataNode"
    },
    {
      "id": "RedshiftCopyActivityId1",
      "input": {
        "ref": "S3DataNodeId1"
```

```
      },
      "schedule": {
        "ref": "ScheduleId1"
      },
      "insertMode": "KEEP_EXISTING",
      "name": "DefaultRedshiftCopyActivity1",
      "runsOn": {
        "ref": "Ec2ResourceId1"
      },
      "type": "RedshiftCopyActivity",
      "output": {
        "ref": "RedshiftDataNodeId1"
      }
    }
  ]
}
```

Para obtener más información acerca de estos objetos, consulte la siguiente documentación.

Objetos

- Nodos de datos
- Recurso
- Actividad

Nodos de datos

En este ejemplo se usan un nodo de datos de entrada, un nodo de datos de salida y una base de datos.

Nodo de datos de entrada

El componente de canalización `S3DataNode` de entrada define la ubicación de los datos de entrada en Amazon S3 y el formato de datos de los datos de entrada. Para obtener más información, consulte S3 DataNode.

Este componente de entrada se define por los siguientes campos:

```
{
  "id": "S3DataNodeId1",
  "schedule": {
```

```
    "ref": "ScheduleId1"
  },
  "filePath": "s3://datapipeline-us-east-1/samples/hive-ads-samples.csv",
  "name": "DefaultS3DataNode1",
  "dataFormat": {
    "ref": "CSVId1"
  },
  "type": "S3DataNode"
},
```

`id`

El ID definido por el usuario, que es una etiqueta solo con fines de referencia.

`schedule`

Una referencia al componente de programación.

`filePath`

La ruta a los datos asociados al nodo de datos, que es un archivo de entrada CSV en este ejemplo.

`name`

El nombre definido por el usuario, que es una etiqueta solo con fines de referencia.

`dataFormat`

Una referencia al formato de los datos para la actividad que se procesará.

Nodo de datos de salida

El componente de canalización `RedshiftDataNode` de salida define una ubicación para los datos de salida; en este caso, una tabla de una base de datos de Amazon Redshift. Para obtener más información, consulte RedshiftDataNode. Este componente de salida se define por los siguientes campos:

```
{
  "id": "RedshiftDataNodeId1",
  "schedule": {
    "ref": "ScheduleId1"
  },
  "tableName": "orders",
```

```
  "name": "DefaultRedshiftDataNode1",
  "createTableSql": "create table StructuredLogs (requestBeginTime CHAR(30) PRIMARY
 KEY DISTKEY SORTKEY, requestEndTime CHAR(30), hostname CHAR(100), requestDate
 varchar(20));",
  "type": "RedshiftDataNode",
  "database": {
    "ref": "RedshiftDatabaseId1"
  }
},
```

`id`

El ID definido por el usuario, que es una etiqueta solo con fines de referencia.

`schedule`

Una referencia al componente de programación.

`tableName`

Nombre de la tabla de Amazon Redshift.

`name`

El nombre definido por el usuario, que es una etiqueta solo con fines de referencia.

`createTableSql`

Una expresión SQL para crear la tabla en la base de datos.

`database`

Una referencia a la base de datos de Amazon Redshift.

Database

El componente `RedshiftDatabase` se define por los siguientes campos. Para obtener más información, consulte RedshiftDatabase.

```
{
  "id": "RedshiftDatabaseId1",
  "databaseName": "dbname",
  "username": "user",
  "name": "DefaultRedshiftDatabase1",
  "*password": "password",
```

```
  "type": "RedshiftDatabase",
  "clusterId": "redshiftclusterId"
},
```

`id`

El ID definido por el usuario, que es una etiqueta solo con fines de referencia.

`databaseName`

El nombre de la base de datos lógica.

`username`

El nombre de usuario para conectarse a la base de datos.

`name`

El nombre definido por el usuario, que es una etiqueta solo con fines de referencia.

`password`

La contraseña para conectarse a la base de datos.

`clusterId`

El ID del clúster de Redshift.

Recurso

Esta es una definición del recurso informático que realiza la operación de copia. En este ejemplo, AWS Data Pipeline debe crear automáticamente una instancia EC2 para realizar la tarea de copia y terminar la instancia tras completarse la tarea. Los campos definidos aquí controlan la creación y función de la instancia que realiza el trabajo. Para obtener más información, consulte Ec2Resource.

`Ec2Resource` se define por los siguientes campos:

```
{
  "id": "Ec2ResourceId1",
  "schedule": {
    "ref": "ScheduleId1"
  },
  "securityGroups": "MySecurityGroup",
  "name": "DefaultEc2Resource1",
```

```
  "role": "DataPipelineDefaultRole",
  "logUri": "s3://myLogs",
  "resourceRole": "DataPipelineDefaultResourceRole",
  "type": "Ec2Resource"
},
```

`id`

El ID definido por el usuario, que es una etiqueta solo con fines de referencia.

`schedule`

El programa en el que desea crear este recurso informático.

`securityGroups`

El grupo de seguridad que se va a utilizar para las instancias del grupo de recursos.

`name`

El nombre definido por el usuario, que es una etiqueta solo con fines de referencia.

`role`

El rol de IAM de la cuenta que tiene acceso a recursos, como el acceso a un bucket de Amazon S3 para recuperar datos.

`logUri`

La ruta de destino de Amazon S3 para realizar copias de seguridad de registros de Task Runner desde `Ec2Resource`.

`resourceRole`

El rol de IAM de la cuenta que crea recursos, como la creación y configuración de una instancia EC2 en su nombre. resourceRole

Actividad

La última sección del archivo JSON es la definición de la actividad que representa el trabajo que se realizará. En este caso, usamos un componente `RedshiftCopyActivity` para copiar los datos de Amazon S3 en Amazon Redshift. Para obtener más información, consulte RedshiftCopyActivity.

El componente `RedshiftCopyActivity` se define por los siguientes campos:

```
{
  "id": "RedshiftCopyActivityId1",
  "input": {
    "ref": "S3DataNodeId1"
  },
  "schedule": {
    "ref": "ScheduleId1"
  },
  "insertMode": "KEEP_EXISTING",
  "name": "DefaultRedshiftCopyActivity1",
  "runsOn": {
    "ref": "Ec2ResourceId1"
  },
  "type": "RedshiftCopyActivity",
  "output": {
    "ref": "RedshiftDataNodeId1"
  }
},
```

`id`

El ID definido por el usuario, que es una etiqueta solo con fines de referencia.

`input`

Una referencia al archivo de origen de Amazon S3.

`schedule`

La programación en la que ejecutar esta actividad.

`insertMode`

El tipo de inserción (`KEEP_EXISTING`, `OVERWRITE_EXISTING` o `TRUNCATE`).

`name`

El nombre definido por el usuario, que es una etiqueta solo con fines de referencia.

`runsOn`

El recurso informático que realiza el trabajo que define esta actividad.

`output`

Una referencia a la tabla de destino de Amazon Redshift.

Cargar y activar la definición de canalización

Debe cargar la definición de su canalización y activarla. En los siguientes comandos de ejemplo, reemplace *pipeline_name* por una etiqueta para su canalización y *pipeline_file* por la ruta completa para el archivo de definición de canalización `.json`.

AWS CLI

Para crear su definición de canalización y activarla, use el siguiente comando create-pipeline. Anote el ID de su canalización, ya que utilizará este valor con la mayoría de los comandos de la CLI.

```
aws datapipeline create-pipeline --name pipeline_name --unique-id token
{
    "pipelineId": "df-00627471SOVYZEXAMPLE"
}
```

Para cargar su definición de la canalización, utilice el comando siguiente: put-pipeline-definition.

```
aws datapipeline put-pipeline-definition --pipeline-id df-00627471SOVYZEXAMPLE --
pipeline-definition file://MyEmrPipelineDefinition.json
```

Si la canalización se valida correctamente, el campo `validationErrors` estará vacío. Debe revisar todas las advertencias.

Para activar la canalización, utilice el siguiente comando activate-pipeline:

```
aws datapipeline activate-pipeline --pipeline-id df-00627471SOVYZEXAMPLE
```

Puede comprobar que su canalización aparece en la lista de canalizaciones mediante el siguiente comando list-pipelines.

```
aws datapipeline list-pipelines
```

Expresiones y funciones de canalizaciones

En esta sección, se explica la sintaxis para utilizar expresiones y funciones en canalizaciones, incluidos los tipos de datos asociados.

Tipos de datos simples

Los siguientes tipos de datos se pueden establecer como valores de campo.

Tipos

- DateTime
- Numérico
- Referencias de objetos
- Periodo
- Cadena

DateTime

AWS Data Pipeline admite la fecha y la hora expresadas en el formato "AAAA-MM-DDTHH:MM:SS" en UTC/GMT únicamente. En el siguiente ejemplo, se establece el campo `startDateTime` de un objeto `Schedule` en `1/15/2012, 11:59 p.m.`, en la zona horaria UTC/GMT.

```
"startDateTime" : "2012-01-15T23:59:00"
```

Numérico

AWS Data Pipeline admite valores enteros y valores de coma flotante.

Referencias de objetos

Un objeto en la definición de la canalización. Puede ser el objeto actual, el nombre de un objeto definido en cualquier parte de la canalización o un objeto que muestra el objeto actual en un campo, al que se hace referencia con la palabra clave `node`. Para obtener más información acerca de `node`, consulte Objetos y campos de referencia. Para obtener más información acerca de los tipos de objetos de canalización, consulte Referencia de objeto de canalización.

Periodo

Indica la frecuencia con la que debe ejecutarse un evento programado. Se expresa en el formato "N [`years`|`months`|`weeks`|`days`|`hours`|`minutes`]", donde N es un valor positivo entero.

El período mínimo es de 15 minutos y el período máximo es de 3 años.

En el siguiente ejemplo, se establece el campo `period` del objeto `Schedule` en 3 horas. Esto crea una programación que se ejecuta cada tres horas.

```
"period" : "3 hours"
```

Cadena

Valores de cadena estándar. Las cadenas deben ir entre comillas dobles ("). Puede utilizar la barra oblicua inversa (\) para escapar caracteres en una cadena. No se admiten cadenas de varias líneas.

A continuación, se muestran ejemplos de valores de cadenas válidos para el campo `id`.

```
"id" : "My Data Object"

"id" : "My \"Data\" Object"
```

Las cadenas también pueden contener expresiones que se evalúan en valores de cadena. Se insertan en la cadena y están delimitados con: "#{" y "}". En el siguiente ejemplo, se utiliza una expresión para insertar el nombre del objeto actual en una ruta.

```
"filePath" : "s3://amzn-s3-demo-bucket/#{name}.csv"
```

Para obtener más información acerca del uso de expresiones, consulte Objetos y campos de referencia y Evaluación de expresiones.

Expresiones

Las expresiones le permiten compartir un valor entre objetos relacionados. El servicio web AWS Data Pipeline procesa las expresiones en tiempo de ejecución, lo que garantiza que todas las expresiones se sustituyan por el valor de la expresión.

Las expresiones están delimitadas por: "#{" y "}". Puede utilizar una expresión en cualquier objeto de definición de canalización donde una cadena sea legal. Si un slot es una referencia o de tipo ID, NAME, TYPE o SPHERE, su valor no se evalúa y se utiliza literalmente.

La siguiente expresión llama a una de las funciones de AWS Data Pipeline. Para obtener más información, consulte Evaluación de expresiones.

```
#{format(myDateTime,'YYYY-MM-dd hh:mm:ss')}
```

Objetos y campos de referencia

Las expresiones pueden utilizar campos del objeto actual en el que existe la expresión o los campos de otro objeto que está vinculado mediante una referencia.

Un formato de slot consta de una hora de creación, seguida de la hora de creación del objeto, como `@S3BackupLocation_2018-01-31T11:05:33`.

También puede hacer referencia al ID de slot exacto especificado en la definición de la canalización; por ejemplo, el ID de slot de la ubicación de copia de seguridad de Amazon S3. Para hacer referencia al ID de slot, utilice `#{parent.@id}`.

En el siguiente ejemplo, el campo `filePath` hace referencia al campo `id` en el mismo objeto para formar un nombre de archivo. El valor de `filePath` se evalúa como "`s3://amzn-s3-demo-bucket/ExampleDataNode.csv`".

```
{
  "id" : "ExampleDataNode",
  "type" : "S3DataNode",
  "schedule" : {"ref" : "ExampleSchedule"},
  "filePath" : "s3://amzn-s3-demo-bucket/#{parent.@id}.csv",
  "precondition" : {"ref" : "ExampleCondition"},
  "onFail" : {"ref" : "FailureNotify"}
}
```

Para utilizar un campo que existe en otro objeto vinculado mediante una referencia, utilice la palabra clave `node`. Esta palabra clave solo está disponible con objetos de alarma y condición previa.

Continuando con el ejemplo anterior, una expresión de una `SnsAlarm` puede hacer referencia al rango de fecha y hora de una `Schedule`, porque `S3DataNode` hace referencia a ambas.

En concreto, el campo `FailureNotify` de `message` puede utilizar los campos en tiempo de ejecución `@scheduledStartTime` y `@scheduledEndTime` desde `ExampleSchedule`, porque el campo `ExampleDataNode` de `onFail` hace referencia a `FailureNotify` y su campo `schedule` hace referencia a `ExampleSchedule`.

```
{
    "id" : "FailureNotify",
    "type" : "SnsAlarm",
    "subject" : "Failed to run pipeline component",
    "message": "Error for interval
 #{node.@scheduledStartTime}..#{node.@scheduledEndTime}.",
    "topicArn":"arn:aws:sns:us-east-1:28619EXAMPLE:ExampleTopic"
},
```

> **Note**
>
> Puede crear canalizaciones que tengan dependencias, como las tareas de la canalización que dependen del trabajo de otros sistemas o tareas. Si la canalización necesita determinados recursos, añada esas dependencias a la canalización con condiciones previas que se asociarán a los nodos de datos y las tareas. Esto hace que las canalizaciones sean más fáciles de depurar y más resistentes. Además, mantenga las dependencias dentro de una única canalización cuando sea posible, ya que es difícil solucionar problemas entre canalizaciones.

Expresiones anidadas

AWS Data Pipeline le permite anidar valores para crear expresiones más complejas. Por ejemplo, para realizar un cálculo de tiempo (restar 30 minutos de `scheduledStartTime`) y formatear el resultado para utilizarlo en una definición de canalización, puede utilizar la siguiente expresión en una actividad:

```
#{format(minusMinutes(@scheduledStartTime,30),'YYYY-MM-dd hh:mm:ss')}
```

y utilizando el prefijo node si la expresión forma parte de una SnsAlarm o Precondition:

```
#{format(minusMinutes(node.@scheduledStartTime,30),'YYYY-MM-dd hh:mm:ss')}
```

Listas

Las expresiones pueden evaluarse en listas y funciones en listas. Por ejemplo, supongamos que una lista se define de la siguiente forma: `"myList":["one","two"]`. Si esta lista se utiliza en la expresión `#{'this is ' + myList}`, se evaluará en `["this is one", "this is two"]`. Si tiene dos listas, Data Pipeline al final las nivela en su evaluación. Por ejemplo, si `myList1` se define como `[1,2]` y `myList2` se define como `[3,4]`, la expresión `[#{myList1}, #{myList2}]` se evalúa en `[1,2,3,4]`.

Expresión de nodo

AWS Data Pipeline utiliza la expresión `#{node.*}` en `SnsAlarm` o `PreCondition` como referencia al objeto principal de un componente de la canalización. Dado que se hace referencia a `SnsAlarm` y `PreCondition` desde una actividad o recurso sin ninguna referencia de vuelta desde ellos, `node` proporciona la manera de hacer referencia al remitente. Por ejemplo, la siguiente definición de canalización demuestra cómo una notificación de error puede utilizar `node` para hacer referencia a su principal, en este caso `ShellCommandActivity`, e incluye las horas de inicio y finalización programadas del principal en el mensaje `SnsAlarm`. La referencia scheduledStartTime de ShellCommandActivity no necesita el prefijo `node` porque scheduledStartTime hace referencia a sí mismo.

> **Note**
>
> Los campos precedidos por el signo AT (@) indican que esos campos son campos de tiempo de ejecución.

```
{
  "id" : "ShellOut",
  "type" : "ShellCommandActivity",
  "input" : {"ref" : "HourlyData"},
  "command" : "/home/userName/xxx.sh #{@scheduledStartTime} #{@scheduledEndTime}",
  "schedule" : {"ref" : "HourlyPeriod"},
  "stderr" : "/tmp/stderr:#{@scheduledStartTime}",
  "stdout" : "/tmp/stdout:#{@scheduledStartTime}",
  "onFail" : {"ref" : "FailureNotify"},
},
{
  "id" : "FailureNotify",
```

```
  "type" : "SnsAlarm",
  "subject" : "Failed to run pipeline component",
  "message": "Error for interval
 #{node.@scheduledStartTime}..#{node.@scheduledEndTime}.",
  "topicArn":"arn:aws:sns:us-east-1:28619EXAMPLE:ExampleTopic"
},
```

AWS Data Pipeline admite referencias transitivas para los campos definidos por el usuario, pero no para los campos de tiempo de ejecución. Una referencia transitiva es una referencia entre dos componentes de la canalización que dependen de otro componente de la canalización como intermediario. El siguiente ejemplo muestra una referencia a un campo definido por el usuario transitivo y una referencia a un campo en tiempo de ejecución no transitivo, ambos válidos. Para obtener más información, consulte Campos definidos por el usuario.

```
{
  "name": "DefaultActivity1",
  "type": "CopyActivity",
  "schedule": {"ref": "Once"},
  "input": {"ref": "s3nodeOne"},
  "onSuccess": {"ref": "action"},
  "workerGroup": "test",
  "output": {"ref": "s3nodeTwo"}
},
{
  "name": "action",
  "type": "SnsAlarm",
  "message": "S3 bucket '#{node.output.directoryPath}' succeeded at
 #{node.@actualEndTime}.",
  "subject": "Testing",
  "topicArn": "arn:aws:sns:us-east-1:28619EXAMPLE:ExampleTopic",
  "role": "DataPipelineDefaultRole"
}
```

Evaluación de expresiones

AWS Data Pipeline proporciona un conjunto de funciones que se pueden utilizar para calcular el valor de un campo. En el siguiente ejemplo, se utiliza la función `makeDate` para establecer el campo `startDateTime` de un objeto `Schedule` en `"2011-05-24T0:00:00"` GMT/UTC.

```
"startDateTime" : "makeDate(2011,5,24)"
```

Funciones matemáticas

Las funciones siguientes están disponibles para trabajar con valores numéricos.

Función	Descripción
+	Suma. Ejemplo:: `#{1 + 2}` Resultado: `3`
-	Resta. Ejemplo:: `#{1 - 2}` Resultado: `-1`
*	Multiplicación. Ejemplo:: `#{1 * 2}` Resultado: `2`
/	División. Si dividimos dos valores enteros, el resultado se trunca. Ejemplo: `#{1 / 2}`, Resultado: `0` Ejemplo: `#{1.0 / 2}`, Resultado: `.5`
^	Exponente. Ejemplo:: `#{2 ^ 2}` Resultado: `4.0`

Funciones de cadena

Las funciones siguientes sirven para trabajar con valores de cadena.

Función	Descripción
+	Concatenación. Los valores que no son de cadena se convierten primero en cadenas. Ejemplo:: `#{"hel" + "lo"}` Resultado: `"hello"`

Funciones de fecha y hora

Las funciones siguientes sirven para trabajar con valores DateTime. Para los ejemplos, el valor de `myDateTime` es `May 24, 2011 @ 5:10 pm GMT`.

> **Note**
>
> El formato de fecha y hora para AWS Data Pipeline es Joda Time, que sustituye a las clases de fecha y hora de Java. Para obtener más información, consulte Joda Time - Class DateTimeFormat.

Función	Descripción
`int day(DateTime myDateTime)`	Obtiene el día del valor de DateTime como un entero. Ejemplo:: `#{day(myDateTime)}` Resultado: `24`
`int dayOfYear(DateTime myDateTime)`	Obtiene el día del año del valor de DateTime como un entero.

Función	Descripción
	Ejemplo:: `#{dayOfYear(myDateTime)}` Resultado: `144`
`DateTime firstOfMonth(DateTime myDateTime)`	Crea un objeto DateTime para el inicio del mes en el DateTime especificado. Ejemplo:: `#{firstOfMonth(myDateTime)}` Resultado: `"2011-05-01T17:10:00z"`
`String format(DateTime myDateTime,String format)`	Crea un objeto String que es el resultado de convertir el valor de DateTime especificado utilizando la cadena de formato especificada. Ejemplo:: `#{format(myDateTime,'YYYY-MM-dd HH:mm:ss z')}` Resultado: `"2011-05-24T17:10:00 UTC"`
`int hour(DateTime myDateTime)`	Obtiene la hora del valor de DateTime como un entero. Ejemplo:: `#{hour(myDateTime)}` Resultado: 17

Función	Descripción
`DateTime makeDate(int year,int month,int day)`	Crea un objeto DateTime, en UTC, con el año, mes y día especificados, a medianoche. Ejemplo:: `#{makeDate(2011,5,24)}` Resultado: `"2011-05-24T0:00:00z"`
`DateTime makeDateTime(int year,int month,int day,int hour,int minute)`	Crea un objeto DateTime, en UTC, con el año, mes, día, hora y minuto especificados. Ejemplo:: `#{makeDateTime(2011,5,24,14,21)}` Resultado: `"2011-05-24T14:21:00z"`
`DateTime midnight(DateTime myDateTime)`	Crea un objeto DateTime para la medianoche actual, en relación con el valor de DateTime especificado. Por ejemplo, si `MyDateTime` es `2011-05-25T17:10:00z`, el resultado es el siguiente. Ejemplo:: `#{midnight(myDateTime)}` Resultado: `"2011-05-25T0:00:00z"`

Función	Descripción
`DateTime minusDays(DateTime myDateTime,int daysToSub)`	Crea un objeto DateTime que es el resultado de la resta del número especificado de días del valor de DateTime especificado. Ejemplo:: `#{minusDays(myDateTime,1)}` Resultado: `"2011-05-23T17:10:00z"`
`DateTime minusHours(DateTime myDateTime,int hoursToSub)`	Crea un objeto DateTime que es el resultado de la resta del número de horas especificado del valor de DateTime especificado. Ejemplo:: `#{minusHours(myDateTime,1)}` Resultado: `"2011-05-24T16:10:00z"`
`DateTime minusMinutes(DateTime myDateTime,int minutesToSub)`	Crea un objeto DateTime que es el resultado de la resta del número de minutos especificado del valor de DateTime especificado. Ejemplo:: `#{minusMinutes(myDateTime,1)}` Resultado: `"2011-05-24T17:09:00z"`

Función	Descripción
`DateTime minusMonths(DateTime myDateTime,int monthsToSub)`	Crea un objeto DateTime que es el resultado de la resta del número de meses especificado del valor de DateTime especificado. Ejemplo:: `#{minusMonths(myDateTime,1)}` Resultado: `"2011-04-24T17:10:00z"`
`DateTime minusWeeks(DateTime myDateTime,int weeksToSub)`	Crea un objeto DateTime que es el resultado de la resta del número de semanas especificado del valor de DateTime especificado. Ejemplo:: `#{minusWeeks(myDateTime,1)}` Resultado: `"2011-05-17T17:10:00z"`
`DateTime minusYears(DateTime myDateTime,int yearsToSub)`	Crea un objeto DateTime que es el resultado de la resta del número de años especificado del valor de DateTime especificado. Ejemplo:: `#{minusYears(myDateTime,1)}` Resultado: `"2010-05-24T17:10:00z"`

Función	Descripción
`int minute(DateTime myDateTime)`	Obtiene el minuto del valor de DateTime como un entero. Ejemplo:: `#{minute(myDateTime)}` Resultado: `10`
`int month(DateTime myDateTime)`	Obtiene el mes del valor de DateTime como un entero. Ejemplo:: `#{month(myDateTime)}` Resultado: 5
`DateTime plusDays(DateTime myDateTime,int daysToAdd)`	Crea un objeto DateTime que es el resultado de la suma del número de días especificado al valor de DateTime especificado. Ejemplo:: `#{plusDays(myDateTime,1)}` Resultado: `"2011-05-25T17:10:00z"`

Función	Descripción
`DateTime plusHours(DateTime myDateTime,int hoursToAdd)`	Crea un objeto DateTime que es el resultado de la suma del número de horas especificado al valor de DateTime especificado. Ejemplo:: `#{plusHours(myDateTime,1)}` Resultado: `"2011-05-24T18:10:00z"`
`DateTime plusMinutes(DateTime myDateTime,int minutesToAdd)`	Crea un objeto DateTime que es el resultado de la suma del número de minutos especificado al valor de DateTime especificado. Ejemplo:: `#{plusMinutes(myDateTime,1)}` Resultado: `"2011-05-24 17:11:00z"`
`DateTime plusMonths(DateTime myDateTime,int monthsToAdd)`	Crea un objeto DateTime que es el resultado de la suma del número de meses especificado al valor de DateTime especificado. Ejemplo:: `#{plusMonths(myDateTime,1)}` Resultado: `"2011-06-24T17:10:00z"`

Función	Descripción
`DateTime plusWeeks(DateTime myDateTime,int weeksToAdd)`	Crea un objeto DateTime que es el resultado de la suma del número de semanas especificado al valor de DateTime especificado. Ejemplo:: `#{plusWeeks(myDateTime,1)}` Resultado: `"2011-05-31T17:10:00z"`
`DateTime plusYears(DateTime myDateTime,int yearsToAdd)`	Crea un objeto DateTime que es el resultado de la suma del número de años especificado al valor de DateTime especificado. Ejemplo:: `#{plusYears(myDateTime,1)}` Resultado: `"2012-05-24T17:10:00z"`

Función	Descripción
`DateTime sunday(DateTime myDateTime)`	Crea un objeto DateTime para el domingo anterior, en relación con el valor de DateTime especificado. Si el valor de DateTime especificado es un domingo, el resultado es el valor de DateTime especificado. Ejemplo:: `#{sunday(myDateTime)}` Resultado: `"2011-05-22 17:10:00 UTC"`
`int year(DateTime myDateTime)`	Obtiene el año del valor de DateTime como un entero. Ejemplo:: `#{year(myDateTime)}` Resultado: `2011`
`DateTime yesterday(DateTime myDateTime)`	Crea un objeto DateTime para el día anterior, en relación con el valor de DateTime especificado. El resultado es el mismo que minusDays(1). Ejemplo:: `#{yesterday(myDateTime)}` Resultado: `"2011-05-23T17:10:00z"`

Caracteres especiales

AWS Data Pipeline utiliza determinados caracteres que tienen un significado especial en las definiciones de canalizaciones, tal y como se muestra en la siguiente tabla.

Carácter especial	Descripción	Ejemplos
@	Campo de tiempo de ejecución. Este carácter es un prefijo del nombre de campo que solo está disponibl e cuando se ejecuta una canalización.	@actualStartTime @failureReason @resourceStatus
#	Expresión. Las expresiones están delimitadas por: "#{" y "}", y el contenido de las llaves lo evalúa AWS Data Pipeline. Para obtener más información, consulte Expresiones.	#{format(myDateTime,'YYYY-MM-dd hh:mm:ss')} s3://amzn-s3-demo-bucket/#{id}.csv
*	Campo cifrado. Este carácter es un prefijo del nombre de campo que indica que AWS Data Pipeline debería cifrar el contenido de este campo en tránsito entre la consola o la CLI y el servicio AWS Data Pipeline.	*password

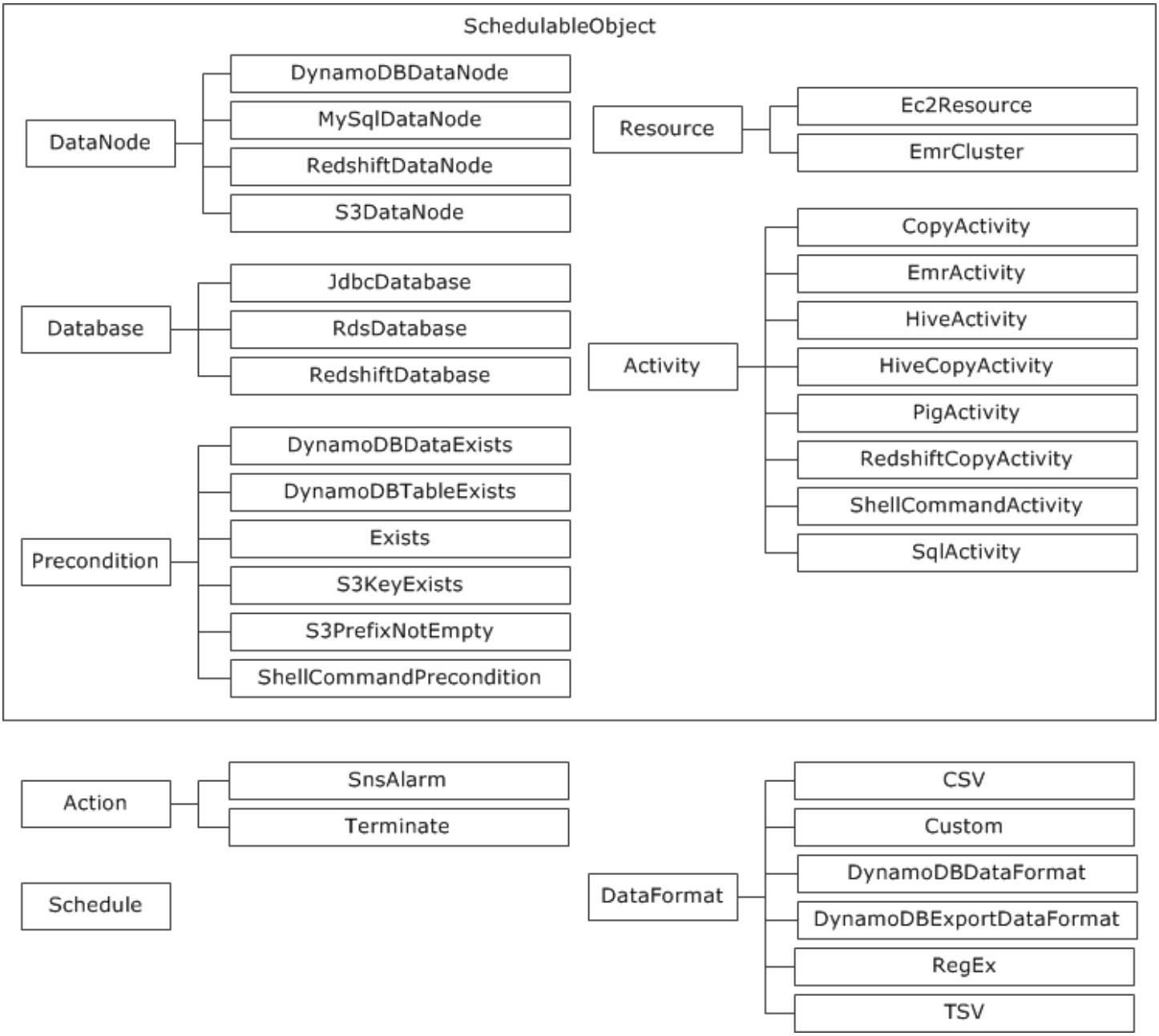
Referencia de objeto de canalización

Puede usar los siguientes componentes y objetos de canalización en su definición de la canalización.

Contenido

> **Note**
>
> Para ver un ejemplo de aplicación que usa el SDK de AWS Data Pipeline Java, consulte Data Pipeline DynamoDB Export Java Sample on. GitHub

A continuación se muestra la jerarquía de objetos de. AWS Data Pipeline



Nodos de datos

Los siguientes son los objetos del nodo de AWS Data Pipeline datos:

Objects

- Nodo Dynamo DBData
- MySqlDataNode
- RedshiftDataNode

- S3 DataNode
- SqlDataNode

Nodo Dynamo DBData

Define un nodo de datos utilizando DynamoDB, que se especifica como una entrada a un objeto `HiveActivity` o `EMRActivity`.

> **Note**
>
> El objeto `DynamoDBDataNode` no admite la condición previa `Exists`.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto. Este objeto hace referencia a otros dos objetos que se definirían en el mismo archivo de definición de canalización. `CopyPeriod` es un objeto `Schedule` y `Ready` es un objeto de condición previa.

```
{
  "id" : "MyDynamoDBTable",
  "type" : "DynamoDBDataNode",
  "schedule" : { "ref" : "CopyPeriod" },
  "tableName" : "adEvents",
  "precondition" : { "ref" : "Ready" }
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
tableName	Tabla de DynamoDB.	Cadena

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Los usuarios deben especificar una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Los usuarios pueden cumplir este requisito estableciendo explícitamente una programación en el objeto, por ejemplo, especificando «schedule»: {"ref»: "DefaultSchedule«}. En la mayoría de los casos, es mejor poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. O bien, si la canalización tiene un árbol de programas (programas dentro del programa maestro), los usuarios pueden crear un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte Programación.	Objeto de referencia, por ejemplo, «schedule»: {"ref»:» mScheduleId «}

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece este campo, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio establecido.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
dataFormat	DataFormat para los datos descritos por este nodo de datos. Actualmente se admite para HiveActivity y HiveCopyActivity.	Objeto de referencia, «DataFormat»: {"ref» DBData FormatId :"MyDynamo «}
dependsOn	Especificar la dependencia de otro objeto ejecutable.	Objeto de referencia, por ejemplo, «DependSon»: {"ref»:» «} myActivityId
failureAndRerunModo	Describe el comportamiento del nodo del consumidor cuando las dependencias producen un error o se vuelven a ejecutar.	Enumeración
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}

Campos opcionales	Description (Descripción)	Tipo de slot
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referencia, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referencia, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
pipelineLogUri	El URI de S3 (como 's3://BucketName/Key/ ') para cargar los registros de la canalización.	Cadena
precondition	Opcionalmente, defina una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condiciones previas.	Objeto de referencia, por ejemplo, «condición previa»: {"ref»:» «} myPreconditionId
readThroughputPercent	Define la velocidad de las operaciones de lectura para mantener la tasa de rendimiento aprovisionada de DynamoDB en el rango asignado para la tabla. El valor es un doble entre 0,1 y 1,0, inclusive.	Double
region	El código para la región en la que se encuentra la tabla de DynamoDB. Por ejemplo, us-east-1. Lo utiliza HiveActivity cuando realiza la puesta en escena de tablas de DynamoDB en Hive.	Enumeración

Campos opcionales	Description (Descripción)	Tipo de slot
reportProgressTimeout	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo
runsOn	El recurso informático para ejecutar la actividad o comando. Por ejemplo, una EC2 instancia de Amazon o un clúster de Amazon EMR.	Objeto de referencia, por ejemplo, «RunSon»: {"ref»:» myResourceId «}
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio del intervalo o al final de este. La programación de estilo de serie temporal significa que las instancias se programan al final de cada intervalo y la programación de estilo cron significa que las instancias se programan al principio de cada intervalo. Un programa bajo demanda le permite ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa bajo demanda, debe especificarse en el objeto predeterminado y debe ser el único scheduleType especificado para los objetos de la canalización. Para usar canalizaciones bajo demanda, basta con llamar a la ActivatePipeline operación para cada ejecución posterior. Los valores son: cron, ondemand y timeseries.	Enumeración

Campos opcionales	Description (Descripción)	Tipo de slot
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor runsOn y workerGroup existe, se hace caso omiso de workerGroup.	Cadena
writeThroughputPercent	Establece la velocidad de las operaciones de escritura para mantener la tasa de rendimiento aprovisionada de DynamoDB en el rango asignado para la tabla. El valor es un doble entre 0,1 y 1,0, inclusive.	Double

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
	dan lugar a objetos de instancia que ejecutan objetos de intento.	

MySqlDataNode

Define un nodo de datos utilizando MySQL.

> **Note**
>
> El tipo `MySqlDataNode` está obsoleto. Le recomendamos que utilice SqlDataNode en su lugar.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto. Este objeto hace referencia a otros dos objetos que se definirían en el mismo archivo de definición de canalización. `CopyPeriod` es un objeto `Schedule` y `Ready` es un objeto de condición previa.

```
{
  "id" : "Sql Table",
  "type" : "MySqlDataNode",
  "schedule" : { "ref" : "CopyPeriod" },
  "table" : "adEvents",
  "username": "user_name",
  "*password": "my_password",
  "connectionString": "jdbc:mysql://mysqlinstance-rds.example.us-
east-1.rds.amazonaws.com:3306/database_name",
  "selectQuery" : "select * from #{table} where eventTime >=
 '#{@scheduledStartTime.format('YYYY-MM-dd HH:mm:ss')}' and eventTime <
 '#{@scheduledEndTime.format('YYYY-MM-dd HH:mm:ss')}'",
  "precondition" : { "ref" : "Ready" }
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
tabla	Nombre de la tabla donde está la base de datos MySQL.	Cadena

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Los usuarios deben especificar una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Los usuarios pueden cumplir este requisito estableciendo explícitamente una programación en el objeto, por ejemplo, especificando «schedule»: {"ref»: "DefaultSchedule«}. En la mayoría de los casos, es mejor poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. O bien, si la canalización tiene un árbol de programas (programas dentro del programa maestro), los usuarios pueden crear un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	Objeto de referencia, por ejemplo, «schedule»: {"ref»:» myScheduleId «}

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio estableci do.	Periodo
createTableSql	Una expresión SQL CREATE TABLE que crea la tabla.	Cadena
database	El nombre de la base de datos.	Objeto de referencia, por ejemplo, «base de datos»: {"ref»:» myDatabaseId «}
dependsOn	Especifica la dependencia de otro objeto ejecutable.	Objeto de referenci a, por ejemplo, «dependSon»: {"ref»:» myActivityId «}
failureAndRerunModo	failureAndRerunMode.	Enumeración
insertQuery	Una instrucción SQL para insertar datos en la tabla.	Cadena
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas	Entero

Campos opcionales	Description (Descripción)	Tipo de slot
	ejecuciones no cuentan para el número de instancias activas.	
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
pipelineLogUri	El URI de S3 (como 's3://BucketName/Key/ ') para cargar los registros de la canalización.	Cadena
precondition	Opcionalmente, defina una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condicion es previas.	Objeto de referenci a, por ejemplo, «condición previa»: {"ref»:» «} myPrecond itionId

Campos opcionales	Description (Descripción)	Tipo de slot
reportProgressTimeout	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo
runsOn	El recurso informático para ejecutar la actividad o comando. Por ejemplo, una EC2 instancia de Amazon o un clúster de Amazon EMR.	Objeto de referencia, por ejemplo, «RunSon»: {"ref»:» myResourceId «}
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio del intervalo o al final de este. La programación de estilo de serie temporal significa que las instancias se programan al final de cada intervalo y la programación de estilo cron significa que las instancias se programan al principio de cada intervalo. Un programa bajo demanda le permite ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa bajo demanda, debe especificarse en el objeto predeterminado y debe ser el único scheduleType especificado para los objetos de la canalización. Para usar canalizaciones bajo demanda, basta con llamar a la ActivatePipeline operación para cada ejecución posterior. Los valores son: cron, ondemand y timeseries.	Enumeración

Campos opcionales	Description (Descripción)	Tipo de slot
schemaName	El nombre del esquema que tiene la tabla.	Cadena
selectQuery	Una instrucción SQL para recuperar datos de la tabla.	Cadena
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor runsOn y workerGroup existe, se hace caso omiso de workerGroup.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
	dan lugar a objetos de instancia que ejecutan objetos de intento.	

Véase también

- S3 DataNode

RedshiftDataNode

Define un nodo de datos utilizando Amazon Redshift. `RedshiftDataNode` representa las propiedades de los datos de una base de datos como, por ejemplo, una tabla de datos, que utiliza la canalización.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto.

```
{
  "id" : "MyRedshiftDataNode",
  "type" : "RedshiftDataNode",
  "database": { "ref": "MyRedshiftDatabase" },
  "tableName": "adEvents",
  "schedule": { "ref": "Hour" }
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
database	La base de datos en la que se encuentra la tabla.	Objeto de referenci a, por ejemplo, «database»: {"ref»:» myRedshiftDatabase Id "}

Campos obligatorios	Description (Descripción)	Tipo de slot
tableName	Nombre de la tabla de Amazon Redshift. La tabla se crea si aún no existe y tú la has proporcionado createTableSql.	Cadena

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Los usuarios deben especificar una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Los usuarios pueden cumplir este requisito estableciendo explícitamente una programación en el objeto, por ejemplo, especificando «schedule»: {"ref»: "DefaultSchedule«}. En la mayoría de los casos, es mejor poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. O bien, si la canalización tiene un árbol de programas (programas dentro del programa maestro), los usuarios pueden crear un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	Objeto de referencia, por ejemplo, «schedule»: {"ref»:» myScheduleId «}

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio estableci do.	Periodo
createTableSql	Una expresión SQL para crear la tabla en la base de datos. Se recomienda especificar el esquema en el que se debe crear la tabla, por ejemplo: CREATE TABLE mySchema. myTable (bestColumn varchar (25) primary key distkey, integer sortKey). numberOfW ins AWS Data Pipeline ejecuta el script en el createTableSql campo si la tabla, especific ada por TableName, no existe en el esquema especificado por el campo SchemaName. Por ejemplo, si especifica SchemaName como mySchema pero no incluye mySchema en el createTableSql campo, la tabla se crea en el esquema incorrecto (de forma predeterminada, se crearía en PUBLIC). Esto ocurre porque AWS Data Pipeline no analiza sus instrucci ones CREATE TABLE.	Cadena
dependsOn	Especificar la dependencia de otro objeto ejecutable.	Objeto de referenci a, por ejemplo, «dependSon»: {"ref»:» «} myActivit yId

Campos opcionales	Description (Descripción)	Tipo de slot
failureAndRerunModo	Describe el comportamiento del nodo del consumidor cuando las dependencias producen un error o se vuelven a ejecutar.	Enumeración
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	El número máximo de intentos en caso de error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}

Campos opcionales	Description (Descripción)	Tipo de slot
pipelineLogUri	El URI de S3 (como 's3://BucketName/Key/ ') para cargar los registros de la canalización.	Cadena
precondition	Opcionalmente, defina una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condicion es previas.	Objeto de referenci a, por ejemplo, «condición previa»: {"ref»:» «} myPrecond itionId
primaryKeys	Si no especifica primaryKeys para una tabla de destino en `RedShiftCopyActivity` , puede especificar una lista de columnas utilizando primaryKeys, que actuará como mergeKey. Sin embargo, si dispone de un valor de primaryKey definido en una tabla de Amazon Redshift, este ajuste anulará la clave existente.	Cadena
reportProgressTime out	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo
runsOn	El recurso informático para ejecutar la actividad o comando. Por ejemplo, una EC2 instancia de Amazon o un clúster de Amazon EMR.	Objeto de referenci a, por ejemplo, «RunSon»: {"ref»:» myResourceId «}

Campos opcionales	Description (Descripción)	Tipo de slot
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio del intervalo o al final de este. La programación de estilo de serie temporal significa que las instancia s se programan al final de cada intervalo y la programación de estilo cron significa que las instancias se programan al principio de cada intervalo. Un programa bajo demanda le permite ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa bajo demanda, debe especificarse en el objeto predeterminado y debe ser el único scheduleType especific ado para los objetos de la canalización. Para usar canalizaciones bajo demanda, basta con llamar a la ActivatePipeline operación para cada ejecución posterior. Los valores son: cron, ondemand y timeseries.	Enumeración
schemaName	Este campo opcional especifica el nombre del esquema para la tabla de Amazon Redshift. Si no se especifica, el nombre del esquema es PUBLIC, que es el esquema predeterm inado en Amazon Redshift. Para obtener más información, consulte la Guía de desarrollador de base de datos de Amazon Redshift.	Cadena
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor runsOn y workerGroup existe, se hace caso omiso de workerGroup.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

S3 DataNode

Define un nodo de datos utilizando Amazon S3. De forma predeterminada, el S3 DataNode utiliza el cifrado del lado del servidor. Si quieres inhabilitarlo, establece s3 EncryptionType en NONE.

> **Note**
>
> Al usar `S3DataNode` como entrada a `CopyActivity`, solo se admiten los formatos de datos CSV y TSV.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto. Este objeto hace referencia a otro objeto que se definiría en el mismo archivo de definición de canalización. `CopyPeriod` es un objeto `Schedule`.

```
{
  "id" : "OutputData",
  "type" : "S3DataNode",
  "schedule" : { "ref" : "CopyPeriod" },
  "filePath" : "s3://amzn-s3-demo-bucket/#{@scheduledStartTime}.csv"
}
```

Sintaxis

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Los usuarios deben especificar una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Los usuarios pueden cumplir este requisito estableciendo explícitamente una programación en el objeto, por ejemplo, especificando «schedule»: {"ref»: "DefaultSchedule«}. En la mayoría de los casos, es mejor poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. O bien, si la canalización tiene un árbol de programas (programas dentro del programa maestro), los usuarios pueden crear un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://	Objeto de referencia, por ejemplo, «schedule»: {"ref»:» mScheduleId «}

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
	docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio establecido.	Periodo
compression	El tipo de compresión de los datos descrito por el S3DataNode. «none» no significa compresión y «gzip» se comprime con el algoritmo gzip. Este campo solo se admite para su uso con Amazon Redshift y cuando se usa S3 DataNode con. CopyActivity	Enumeración
dataFormat	DataFormat para los datos descritos en este S3DataNode.	Objeto de referencia, por ejemplo, «dataFormat»: {"ref»:» myDataFormat Id "}
dependsOn	Especificar la dependencia de otro objeto ejecutable.	Objeto de referencia, por ejemplo, «DependSon»: {"ref»:» myActivityId «}

Campos opcionales	Description (Descripción)	Tipo de slot
directoryPath	Ruta del directorio Amazon S3 como URI: s3://my-bucket/my-key-for-directory. Debe proporcionar un valor filePath o directoryPath.	Cadena
failureAndRerunModo	Describe el comportamiento del nodo del consumidor cuando las dependencias producen un error o se vuelven a ejecutar.	Enumeración
filePath	La ruta al objeto en Amazon S3 como URI, por ejemplo: s3://my-bucket/my-key-for-file. Debe proporcionar un valor filePath o directoryPath. Estos valores representan una carpeta y un nombre de archivo. Use el valor directoryPath para acomodar varios archivos en un directorio.	Cadena
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo
manifestFilePath	La ruta de Amazon S3 a un archivo de manifiesto en el formato compatible con Amazon Redshift. AWS Data Pipeline utiliza el archivo de manifiesto para copiar los archivos de Amazon S3 especificados en la tabla. Este campo solo es válido cuando a RedShiftCopyActivity hace referencia al S3DataNode.	Cadena
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero

Campos opcionales	Description (Descripción)	Tipo de slot
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
pipelineLogUri	El URI de S3 (como 's3://BucketName/Key/ ') para cargar los registros de la canalización.	Cadena
precondition	Opcionalmente, defina una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condicion es previas.	Objeto de referenci a, por ejemplo, «condición previa»: {"ref»:» «} myPrecond itionId
reportProgressTime out	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo
runsOn	El recurso informático para ejecutar la actividad o comando. Por ejemplo, una EC2 instancia de Amazon o un clúster de Amazon EMR.	Objeto de referencia, por ejemplo, «RunSon»: {"ref»:» myResourceId «}
s3 EncryptionType	Sobrescribe el tipo de cifrado de Amazon S3. Los valores son SERVER_SIDE_ENCRYPTION o NONE. El cifrado en el servidor está habilitado de forma predeterminada.	Enumeración
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio del intervalo o al final de este. La programación de estilo de serie temporal significa que las instancias se programan al final de cada intervalo y la programación de estilo cron significa que las instancias se programan al principio de cada intervalo. Un programa bajo demanda le permite ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa bajo demanda, debe especificarse en el objeto predeterminado y debe ser el único scheduleType especificado para los objetos de la canalización. Para usar canalizaciones bajo demanda, basta con llamar a la ActivatePipeline operación para cada ejecución posterior. Los valores son: cron, ondemand y timeseries.	Enumeración

Campos opcionales	Description (Descripción)	Tipo de slot
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor runsOn y workerGroup existe, se hace caso omiso de workerGroup.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Véase también

- MySqlDataNode

SqlDataNode

Define un nodo de datos utilizando SQL.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto. Este objeto hace referencia a otros dos objetos que se definirían en el mismo archivo de definición de canalización. `CopyPeriod` es un objeto `Schedule` y `Ready` es un objeto de condición previa.

```
{
  "id" : "Sql Table",
  "type" : "SqlDataNode",
  "schedule" : { "ref" : "CopyPeriod" },
  "table" : "adEvents",
  "database":"myDataBaseName",
  "selectQuery" : "select * from #{table} where eventTime >=
 '#{@scheduledStartTime.format('YYYY-MM-dd HH:mm:ss')}' and eventTime <
 '#{@scheduledEndTime.format('YYYY-MM-dd HH:mm:ss')}'",
  "precondition" : { "ref" : "Ready" }
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
tabla	Nombre de la tabla donde está la base de datos SQL.	Cadena

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Los usuarios deben especificar una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Los usuarios pueden cumplir este requisito	Objeto de referencia, por ejemplo, «schedule»: {"ref»:» mScheduleId «}

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
	estableciendo explícitamente una programación en el objeto, por ejemplo, especificando «schedule»: {"ref»: "DefaultSchedule«}. En la mayoría de los casos, es mejor poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. O bien, si la canalización tiene un árbol de programas (programas dentro del programa maestro), los usuarios pueden crear un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio establecido.	Periodo
createTableSql	Una expresión SQL CREATE TABLE que crea la tabla.	Cadena
database	El nombre de la base de datos.	Objeto de referencia, por ejemplo, «base

Campos opcionales	Description (Descripción)	Tipo de slot
		de datos»: {"ref»:» myDatabaseId «}
dependsOn	Especifica la dependencia de otro objeto ejecutable.	Objeto de referencia, por ejemplo, «dependSon»: {"ref»:» myActivityId «}
failureAndRerunModo	failureAndRerunMode.	Enumeración
insertQuery	Una instrucción SQL para insertar datos en la tabla.	Cadena
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en ondemand.	Periodo
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referencia, por ejemplo, "onLateAction«: {" ref»:» myActionId «}

Campos opcionales	Description (Descripción)	Tipo de slot
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referencia, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
pipelineLogUri	El URI de S3 (como 's3://BucketName/Key/ ') para cargar los registros de la canalización.	Cadena
precondition	Opcionalmente, defina una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condiciones previas.	Objeto de referencia, por ejemplo, «condición previa»: {"ref»:» «} myPreconditionId
reportProgressTimeout	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo
runsOn	El recurso informático para ejecutar la actividad o comando. Por ejemplo, una EC2 instancia de Amazon o un clúster de Amazon EMR.	Objeto de referencia, por ejemplo, «RunSon»: {"ref»:» myResourceId «}

Campos opcionales	Description (Descripción)	Tipo de slot
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio del intervalo o al final de este. La programación de estilo de serie temporal significa que las instancia s se programan al final de cada intervalo y la programación de estilo cron significa que las instancias se programan al principio de cada intervalo. Un programa bajo demanda le permite ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa bajo demanda, debe especificarse en el objeto predeterminado y debe ser el único scheduleType especific ado para los objetos de la canalización. Para usar canalizaciones bajo demanda, basta con llamar a la ActivatePipeline operación para cada ejecución posterior. Los valores son: cron, ondemand y timeseries.	Enumeración
schemaName	El nombre del esquema que tiene la tabla.	Cadena
selectQuery	Una instrucción SQL para recuperar datos de la tabla.	Cadena
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor runsOn y workerGroup existe, se hace caso omiso de workerGroup.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Véase también

- S3 DataNode

Actividades

Los objetos de la AWS Data Pipeline actividad son los siguientes:

Objects

- CopyActivity
- EmrActivity

- HadoopActivity
- HiveActivity
- HiveCopyActivity
- PigActivity
- RedshiftCopyActivity
- ShellCommandActivity
- SqlActivity

CopyActivity

Copia los datos de una ubicación a otra. `CopyActivity`admite S3 DataNode y SqlDataNode como entrada y salida y la operación de copia se realiza normalmente record-by-record. Sin embargo, `CopyActivity` proporciona copia de Amazon S3 a Amazon S3 de alto desempeño cuando se cumplen todas las condiciones siguientes:

- La entrada y la salida son S3 DataNodes
- El campo `dataFormat` es el mismo para la entrada y la salida

Si proporciona archivos de datos comprimidos como entrada y no lo indica mediante el campo `compression` en los nodos de datos de S3, es posible que `CopyActivity` produzca un error. En este caso, `CopyActivity` no detecta correctamente el carácter de fin de registro y la operación produce un error. Además, `CopyActivity` admite la copia de un directorio a otro y la copia de un archivo a un directorio, pero la record-by-record copia se produce al copiar un directorio a un archivo. Por último, `CopyActivity` no admite la copia de archivos de Amazon S3 multiparte.

`CopyActivity` tiene limitaciones específicas en cuanto a su compatibilidad con CSV. Cuando usa un S3 DataNode como entrada`CopyActivity`, solo puede usar una Unix/Linux variante del formato de archivo de datos CSV para los campos de entrada y salida de Amazon S3. La Unix/Linux variante requiere lo siguiente:

- El separador debe ser el carácter "," (coma).
- Los registros no se indican entre comillas.
- El carácter de escape predeterminado es el valor ASCII 92 (barra diagonal invertida).
- El identificador de fin de registro es el valor ASCII 10 (o "\n").

Los sistemas basados en Windows suelen utilizar una secuencia de end-of-record caracteres diferente: un tren de ida y vuelta y una alimentación de línea al mismo tiempo (valor ASCII 13 y valor ASCII 10). Debe adaptarse a esta diferencia con un mecanismo adicional como, por ejemplo, un script previo a la copia para modificar los datos de entrada, a fin de garantizar que `CopyActivity` pueda detectar correctamente el fin de un registro; de lo contrario, `CopyActivity` devuelve error repetidamente.

Al usar `CopyActivity` para exportar desde un objeto RDS de PostgreSQL a un formato de datos TSV, el carácter NULL predeterminado es \n.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto. Este objeto hace referencia a otros tres objetos que se definirían en el mismo archivo de definición de canalización. `CopyPeriod` es un objeto `Schedule`, y `InputData` y `OutputData` son objetos del nodo de datos.

```
{
  "id" : "S3ToS3Copy",
  "type" : "CopyActivity",
  "schedule" : { "ref" : "CopyPeriod" },
  "input" : { "ref" : "InputData" },
  "output" : { "ref" : "OutputData" },
  "runsOn" : { "ref" : "MyEc2Resource" }
}
```

Sintaxis

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Los usuarios deben especificar una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Los usuarios pueden cumplir este requisito estableciendo explícitamente un horario en el objeto, por ejemplo, especificando «schedule»: {"ref»: "DefaultSchedule«}. En la mayoría	Objeto de referencia, por ejemplo, «schedule»: {"ref»:» myScheduleId «}

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
	de los casos, es mejor poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. O bien, si la canalización tiene un árbol de programas (programas dentro del programa maestro), los usuarios pueden crear un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
runsOn	El recurso informático para ejecutar la actividad o comando. Por ejemplo, una EC2 instancia de Amazon o un clúster de Amazon EMR.	Objeto de referencia, por ejemplo, «RunSon»: {"ref»:» myResourceId «}
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor runsOn y workerGroup existe, se hace caso omiso de workerGroup.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio estableci do.	Periodo
dependsOn	Especificar la dependencia de otro objeto ejecutable.	Objeto de referenci a, por ejemplo, «DependSon»: {"ref»:» myActivityId «}
failureAndRerunModo	Describe el comportamiento del nodo del consumidor cuando las dependencias producen un error o se vuelven a ejecutar.	Enumeración
input	El origen de datos de entrada.	Objeto de referencia, por ejemplo, «input»: {"ref»:» myDataNode Id "}
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en ondemand.	Periodo
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero

Campos opcionales	Description (Descripción)	Tipo de slot
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
salida	El origen de datos de salida.	Objeto de referencia, por ejemplo, «output»: {"ref»:» myDataNode Id "}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
pipelineLogUri	El URI de S3 (como 's3://BucketName/Key/ ') para cargar los registros de la canalización.	Cadena
precondition	Opcionalmente, defina una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condicion es previas.	Objeto de referenci a, por ejemplo, «condición previa»: {"ref»:» «} myPrecond itionId

Campos opcionales	Description (Descripción)	Tipo de slot
reportProgressTimeout	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio del intervalo o al final de este. La programación de estilo de serie temporal significa que las instancias se programan al final de cada intervalo y la programación de estilo cron significa que las instancias se programan al principio de cada intervalo. Un programa bajo demanda le permite ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa bajo demanda, debe especificarse en el objeto predeterminado y debe ser el único scheduleType especificado para los objetos de la canalización. Para usar canalizaciones bajo demanda, basta con llamar a la ActivatePipeline operación para cada ejecución posterior. Los valores son: cron, ondemand y timeseries.	Enumeración

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Véase también

- ShellCommandActivity
- EmrActivity
- Exportar datos de MySQL a Amazon S3 con la AWS Data Pipeline

EmrActivity

Ejecuta un clúster de EMR.

AWS Data Pipeline utiliza un formato para los pasos diferente al de Amazon EMR; por ejemplo, AWS Data Pipeline utiliza argumentos separados por comas después del nombre JAR en el campo step. `EmrActivity` En el siguiente ejemplo, se muestra un paso formateado para Amazon EMR, seguido de su equivalente para AWS Data Pipeline :

```
s3://amzn-s3-demo-bucket/MyWork.jar arg1 arg2 arg3
```

```
"s3://amzn-s3-demo-bucket/MyWork.jar,arg1,arg2,arg3"
```

Ejemplos

A continuación se muestra un ejemplo de este tipo de objeto. En este ejemplo se utilizan las versiones anteriores de Amazon EMR. Verifique que este ejemplo es adecuado para la versión del clúster de Amazon EMR que está utilizando.

Este objeto hace referencia a otros tres objetos que se definirían en el mismo archivo de definición de canalización. `MyEmrCluster` es un objeto `EmrCluster`, y `MyS3Input` y `MyS3Output` son objetos `S3DataNode`.

> **Note**
>
> En este ejemplo, puede reemplazar el campo `step` por su cadena de clúster deseada, que podría ser un script de Pig, un clúster de Hadoop Streaming, su propio JAR personalizado (incluidos sus parámetros), etc.

Hadoop 2.x (AMI 3.x)

```
{
  "id" : "MyEmrActivity",
  "type" : "EmrActivity",
  "runsOn" : { "ref" : "MyEmrCluster" },
  "preStepCommand" : "scp remoteFiles localFiles",
  "step" : ["s3://amzn-s3-demo-bucket/myPath/myStep.jar,firstArg,secondArg,-files,s3://
amzn-s3-demo-bucket/myPath/myFile.py,-input,s3://myinputbucket/path,-output,s3://
myoutputbucket/path,-mapper,myFile.py,-reducer,reducerName","s3://amzn-s3-demo-bucket/
myPath/myotherStep.jar,..."],
  "postStepCommand" : "scp localFiles remoteFiles",
  "input" : { "ref" : "MyS3Input" },
```

```
  "output" : { "ref" : "MyS3Output" }
}
```

> **Note**
>
> Para pasar argumentos a una aplicación en un paso, es necesario especificar la región en la ruta del script, como en el siguiente ejemplo. Además, es posible que necesite aplicar escape a los argumentos que transfiere. Por ejemplo, si usa `script-runner.jar` para ejecutar un script de shell y desea transferir argumentos al script, debe aplicar escape a las comas que los separan. En el siguiente slot del paso se ilustra cómo hacerlo:
>
> ```
> "step" : "s3://eu-west-1.elasticmapreduce/libs/script-runner/script-
> runner.jar,s3://datapipeline/echo.sh,a\\\\,b\\\\,c"
> ```
>
> Este paso usa `script-runner.jar` para ejecutar el script de shell `echo.sh` y transfiere `a`, `b` y `c` como argumento único al script. El primer carácter de escape se quita del argumento obtenido, por lo que es posible que sea necesario aplicar escape de nuevo. Por ejemplo, si tuviera `File\.gz` como argumento en JSON, podría aplicarle escape mediante `File\\\ \.gz`. Sin embargo, debido que el primer escape se ha descartado, debe usar `File\\\\\\ \\.gz` .

Sintaxis

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Especifiq ue una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Puede cumplir este requisito estableciendo de forma explícita un programa en el objeto, por ejemplo, especificando `"schedule": {"ref": "DefaultSchedule"}` . En la mayoría de los casos, es mejor poner la referencia de	Objeto de referenci a, por ejemplo, «schedule»: {"ref»:» «} mScheduleId

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
	programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. O bien, si la canalización tiene un árbol de programas (programas dentro del programa maestro), puede crear un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
runsOn	El clúster de Amazon EMR en el que se ejecutará este trabajo.	Objeto de referencia, por ejemplo, «RunSon»: {"ref»:» myEmrCluster Id "}
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor `runs0n` y existe `workerGroup` , `workerGroup` se ignora.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio estableci do.	Periodo
dependsOn	Especificar la dependencia de otro objeto ejecutable.	Objeto de referenci a, por ejemplo, «dependSon»: {"ref»:» myActivityId «}
failureAndRerunModo	failureAndRerunMode.	Enumeración
input	La ubicación de los datos de entrada.	Objeto de referencia, por ejemplo, «input»: {"ref»:» myDataNode Id "}
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en ondemand.	Periodo
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	El número máximo de intentos en caso de error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}

Campos opcionales	Description (Descripción)	Tipo de slot
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referencia, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referencia, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
salida	La ubicación de los datos de salida.	Objeto de referencia, por ejemplo, «output»: {"ref»:» myDataNode Id "}
parent	El elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
pipelineLogUri	El URI de Amazon S3, como 's3://BucketName/ Prefix/ 'para cargar los registros de la canalización.	Cadena
postStepCommand	Scripts de shell que se van a ejecutar después de terminar todos los pasos. Para especificar varios scripts, hasta 255, añada varios campos `postStepCommand` .	Cadena
precondition	Opcionalmente, defina una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condiciones previas.	Objeto de referencia, por ejemplo, «precondition»: {"ref»:» «} myPrecondition Id

Campos opcionales	Description (Descripción)	Tipo de slot
preStepCommand	Scripts de shell que se van a ejecutar antes de que se ejecute algún paso. Para especificar varios scripts, hasta 255, añada varios campos `preStepCommand` .	Cadena
reportProgressTimeout	El tiempo de espera para llamadas sucesivas del trabajo remoto a `reportProgress` . Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
resizeClusterBeforeEn ejecución	Cambiar el tamaño del clúster antes de realizar esta actividad para adaptarse a las tablas de DynamoDB especificadas como entradas o salidas. **Note** Si `EmrActivity` usa un `DynamoDBDataNode` nodo de datos de entrada o salida, y si lo establece en`TRUE`, AWS Data Pipeline comienza `resizeClusterBeforeRunning` a usar tipos de `m3.xlarge` instancia. Se sobrescriben las opciones de tipo de instancia con `m3.xlarge` , lo que podría aumentar los costos mensuales.	Booleano
resizeClusterMaxInstancias	Un límite del número máximo de instancias que el algoritmo de cambio de tamaño puede solicitar.	Entero
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio o al final del intervalo. Los valores son: `cron`, `ondemand` y `timeseries` . La programación `timeseries` significa que las instancias se programan al final de cada intervalo. La programación `cron` significa que las instancias se programan al principio de cada intervalo. Un programa `ondemand` le permite ejecutar una canalización una vez por activación. No tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa `ondemand`, debe especificarse en el objeto predeterminado y debe ser el único `scheduleType` especificado para los objetos de la canalización. Para usar canalizaciones `ondemand`, solo tiene que llamar a la operación `ActivatePipeline` para cada ejecución posterior.	Enumeración
paso	Uno o varios pasos para que se ejecute el clúster. Para especificar varios pasos, hasta 255, añada varios campos step. Utilice argumentos separados por comas después del nombre de JAR; por ejemplo, "`s3://amzn-s3-demo-bucket/MyWork.jar,arg1,arg2,arg3`".	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveIn

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
		stances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referenci a, por ejemplo, cascadeFailedOn «: {" ref»:» myRunnabl eObject Id "}
emrStepLog	Registros de pasos de Amazon EMR disponibl es únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El `errorId` si este objeto ha fallado.	Cadena
errorMessage	El `errorMessage` si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Véase también

- ShellCommandActivity
- CopyActivity
- EmrCluster

HadoopActivity

Ejecuta un MapReduce trabajo en un clúster. El clúster puede ser un clúster de EMR administrado por AWS Data Pipeline u otro recurso si lo usa. TaskRunner HadoopActivity Úselo cuando desee ejecutar el trabajo en paralelo. Esto le permite utilizar los recursos de programación del marco YARN o el negociador de MapReduce recursos de Hadoop 1. Si desea ejecutar el trabajo de forma secuencial mediante la acción de paso de Amazon EMR, puede usar EmrActivity.

Ejemplos

HadoopActivity mediante un clúster de EMR gestionado por AWS Data Pipeline

El siguiente HadoopActivity objeto utiliza un EmrCluster recurso para ejecutar un programa:

```
 {
   "name": "MyHadoopActivity",
   "schedule": {"ref": "ResourcePeriod"},
   "runsOn": {"ref": “MyEmrCluster”},
   "type": "HadoopActivity",
   "preActivityTaskConfig":{"ref":"preTaskScriptConfig”},
   "jarUri": "/home/hadoop/contrib/streaming/hadoop-streaming.jar",
   "argument": [
     "-files",
     “s3://elasticmapreduce/samples/wordcount/wordSplitter.py“,
     "-mapper",
     "wordSplitter.py",
     "-reducer",
     "aggregate",
     "-input",
     "s3://elasticmapreduce/samples/wordcount/input/",
     "-output",
     “s3://amzn-s3-demo-bucket/MyHadoopActivity/#{@pipelineId}/
#{format(@scheduledStartTime,'YYYY-MM-dd')}"
   ],
   "maximumRetries": "0",
   "postActivityTaskConfig":{"ref":"postTaskScriptConfig”},
   "hadoopQueue" : “high”
 }
```

Esta es la correspondiente*MyEmrCluster*, que configura las colas FairScheduler y en YARN para Hadoop 2: AMIs

```
{
  "id" : "MyEmrCluster",
  "type" : "EmrCluster",
   "hadoopSchedulerType" : "PARALLEL_FAIR_SCHEDULING",
  “amiVersion” : “3.7.0”,
  "bootstrapAction" : ["s3://Region.elasticmapreduce/bootstrap-
actions/configure-hadoop,-z,yarn.scheduler.capacity.root.queues=low
\,high\,default,-z,yarn.scheduler.capacity.root.high.capacity=50,-
```

```
z,yarn.scheduler.capacity.root.low.capacity=10,-
z,yarn.scheduler.capacity.root.default.capacity=30”]
}
```

Esto es lo que se usa para configurar en EmrCluster Hadoop 1: FairScheduler

```
{
      "id": "MyEmrCluster",
      "type": "EmrCluster",
      "hadoopSchedulerType": "PARALLEL_FAIR_SCHEDULING",
      "amiVersion": "2.4.8",
      "bootstrapAction": "s3://Region.elasticmapreduce/bootstrap-
actions/configure-hadoop,-m,mapred.queue.names=low\\\\,high\\\\,default,-
m,mapred.fairscheduler.poolnameproperty=mapred.job.queue.name"
          }
```

Las siguientes EmrCluster configuraciones para Hadoop 2 están basadas en Hadoop CapacityScheduler 2: AMIs

```
{
      "id": "MyEmrCluster",
      "type": "EmrCluster",
      "hadoopSchedulerType": "PARALLEL_CAPACITY_SCHEDULING",
      "amiVersion": "3.7.0",
      "bootstrapAction": "s3://Region.elasticmapreduce/bootstrap-
actions/configure-hadoop,-z,yarn.scheduler.capacity.root.queues=low
\\\\,high,-z,yarn.scheduler.capacity.root.high.capacity=40,-
z,yarn.scheduler.capacity.root.low.capacity=60"
    }
```

HadoopActivity usar un clúster de EMR existente

En este ejemplo, utiliza grupos de trabajo y TaskRunner a para ejecutar un programa en un clúster de EMR existente. La siguiente definición de canalización se utiliza para: HadoopActivity

- Ejecuta un MapReduce programa solo con *myWorkerGroup* recursos. Para obtener más información acerca de los grupos de procesos de trabajo, consulte Ejecución de trabajo en recursos existentes mediante Task Runner.
- Ejecute un preActivityTask Config and postActivityTask Config

```
{
  "objects": [
    {
      "argument": [
        "-files",
        "s3://elasticmapreduce/samples/wordcount/wordSplitter.py",
        "-mapper",
        "wordSplitter.py",
        "-reducer",
        "aggregate",
        "-input",
        "s3://elasticmapreduce/samples/wordcount/input/",
        "-output",
        "s3://amzn-s3-demo-bucket/MyHadoopActivity/#{@pipelineId}/
#{format(@scheduledStartTime,'YYYY-MM-dd')}"
      ],
      "id": "MyHadoopActivity",
      "jarUri": "/home/hadoop/contrib/streaming/hadoop-streaming.jar",
      "name": "MyHadoopActivity",
      "type": "HadoopActivity"
    },
    {
      "id": "SchedulePeriod",
      "startDateTime": "start_datetime",
      "name": "SchedulePeriod",
      "period": "1 day",
      "type": "Schedule",
      "endDateTime": "end_datetime"
    },
    {
      "id": "ShellScriptConfig",
      "scriptUri": "s3://amzn-s3-demo-bucket/scripts/preTaskScript.sh",
      "name": "preTaskScriptConfig",
      "scriptArgument": [
        "test",
        "argument"
      ],
      "type": "ShellScriptConfig"
    },
    {
      "id": "ShellScriptConfig",
      "scriptUri": "s3://amzn-s3-demo-bucket/scripts/postTaskScript.sh",
      "name": "postTaskScriptConfig",
```

```
      "scriptArgument": [
        "test",
        "argument"
      ],
      "type": "ShellScriptConfig"
    },
    {
      "id": "Default",
      "scheduleType": "cron",
      "schedule": {
        "ref": "SchedulePeriod"
      },
      "name": "Default",
      "pipelineLogUri": "s3://amzn-s3-demo-bucket/
logs/2015-05-22T18:02:00.343Z642f3fe415",
      "maximumRetries": "0",
      "workerGroup": "myWorkerGroup",
      "preActivityTaskConfig": {
        "ref": "preTaskScriptConfig"
      },
      "postActivityTaskConfig": {
        "ref": "postTaskScriptConfig"
      }
    }
  ]
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
jarUri	Ubicación de un JAR en Amazon S3 o en el sistema de archivos local del clúster con el que se va a ejecutar HadoopActivity.	Cadena

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Los usuarios deben especificar una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Los usuarios pueden cumplir este requisito estableciendo explícitamente una programación en el objeto, por ejemplo, especificando «schedule»: {"ref»: "DefaultSchedule«}. En la mayoría de los casos, es mejor poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. O bien, si la canalización tiene un árbol de programas (programas dentro del programa maestro), los usuarios pueden crear un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	Objeto de referencia, por ejemplo, «schedule»: {"ref»:» myScheduleId «}

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
runsOn	Clúster de EMR en el que se ejecutará este trabajo.	Objeto de referencia, por ejemplo, «RunSon»: {"ref»:» myEmrCluster Id "}

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor runsOn y workerGroup existe, se hace caso omiso de workerGroup.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
argumento	Argumentos que se pasan al archivo JAR.	Cadena
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio estableci do.	Periodo
dependsOn	Especificar la dependencia de otro objeto ejecutable.	Objeto de referenci a, por ejemplo, «DependSon»: {"ref»:» myActivityId «}
failureAndRerunModo	Describe el comportamiento del nodo del consumidor cuando las dependencias producen un error o se vuelven a ejecutar.	Enumeración
hadoopQueue	El nombre de cola de programador Hadoop en el que se enviará la actividad.	Cadena
input	Ubicación de los datos de entrada.	Objeto de referencia, por ejemplo, «input»:

Campos opcionales	Description (Descripción)	Tipo de slot
		{"ref»:» myDataNode Id "}
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en ondemand.	Periodo
mainClass	La clase principal del JAR con el que estás ejecutando HadoopActivity.	Cadena
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}

Campos opcionales	Description (Descripción)	Tipo de slot
salida	Ubicación de los datos de salida.	Objeto de referencia, por ejemplo, «output»: {"ref»:» myDataNode Id "}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
pipelineLogUri	El URI de S3 (como 's3://BucketName/Key/ ') para cargar los registros de la canalización.	Cadena
postActivityTaskCo nfig	Script de configuración después de la actividad que se va a ejecutar. Este consta de un URI del script de shell en Amazon S3 y una lista de argumentos.	Objeto de referencia, por ejemplo, "postActi vityTaskConfig»: {"ref»:» myShellScript ConfigId «}
preActivityTaskConfig	Script de configuración antes de la actividad que se va a ejecutar. Este consta de un URI del script de shell en Amazon S3 y una lista de argumentos.	Objeto de referencia, por ejemplo, "preActiv ityTaskConfig»: {"ref»:» myShellScript ConfigId «}
precondition	Opcionalmente, defina una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condicion es previas.	Objeto de referenci a, por ejemplo, «condición previa»: {"ref»:» myPrecond itionId «}

Campos opcionales	Description (Descripción)	Tipo de slot
reportProgressTimeout	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio del intervalo o al final de este. La programación de estilo de serie temporal significa que las instancias se programan al final de cada intervalo y la programación de estilo cron significa que las instancias se programan al principio de cada intervalo. Un programa bajo demanda le permite ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa bajo demanda, debe especificarse en el objeto predeterminado y debe ser el único scheduleType especificado para los objetos de la canalización. Para usar canalizaciones bajo demanda, basta con llamar a la ActivatePipeline operación para cada ejecución posterior. Los valores son: cron, ondemand y timeseries.	Enumeración

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Véase también

- ShellCommandActivity
- CopyActivity
- EmrCluster

HiveActivity

Ejecuta una consulta de Hive en un clúster de EMR. `HiveActivity` facilita la configuración de una actividad de Amazon EMR y crea automáticamente tablas de Hive basadas en datos de entrada procedentes de Amazon S3 o Amazon RDS. Todo lo que necesita especificar es el HiveQL para

que se ejecute en los datos de origen. AWS Data Pipeline crea automáticamente tablas de Hive con `${input1}${input2}`, etc., en función de los campos de entrada del objeto. `HiveActivity`

Para las entradas de Amazon S3 el campo `dataFormat` se usa para crear los nombres de las columnas de Hive.

En las entradas de MySQL (Amazon RDS), los nombres de las columnas para la consulta SQL se utilizan para crear los nombres de las columnas de Hive.

> **Note**
>
> Esta actividad usa CSV Serde de Hive.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto. Este objeto hace referencia a otros tres objetos que se definen en el mismo archivo de definición de canalización. `MySchedule` es un objeto `Schedule`, y `MyS3Input` y `MyS3Output` son objetos del nodo de datos.

```
{
  "name" : "ProcessLogData",
  "id" : "MyHiveActivity",
  "type" : "HiveActivity",
  "schedule" : { "ref": "MySchedule" },
  "hiveScript" : "INSERT OVERWRITE TABLE ${output1} select
 host,user,time,request,status,size from ${input1};",
  "input" : { "ref": "MyS3Input" },
  "output" : { "ref": "MyS3Output" },
  "runsOn" : { "ref": "MyEmrCluster" }
}
```

Sintaxis

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Especifiq	Objeto de referencia, por ejemplo,

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
	ue una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Puede cumplir este requisito estableciendo explícitamente una programación en el objeto, por ejemplo, especificando «schedule»: {"ref": "DefaultSchedule«}. En la mayoría de los casos, es mejor poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. O bien, si la canalización tiene un árbol de programas (programas dentro del programa maestro), puede crear un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html.	«schedule»: {"ref»:» mySchedulelId «}

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
hiveScript	El script de Hive que se ejecutará.	Cadena
scriptUri	La ubicación del script de Hive que se ejecutará (por ejemplo, s3://scriptLocation).	Cadena

Grupo obligatorio	Description (Descripción)	Tipo de slot
runsOn	El clúster de EMR en el que se ejecuta `HiveActivity` .	Objeto de referencia, por ejemplo, «RunSon»: {"ref»:» myEmrCluster Id "}
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor `runsOn` y existe `workerGroup` , `workerGroup` se ignora.	Cadena
input	El origen de datos de entrada.	Objeto de referencia, como «input»: {"ref»:» myDataNode Id "}
salida	El origen de datos de salida.	Objeto de referencia, como «output»: {"ref»:» myDataNode Id "}

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio establecido.	Periodo
dependsOn	Especificar la dependencia de otro objeto ejecutable.	Objeto de referencia, como «DependSon»: {"ref»:» myActivityId «}

Campos opcionales	Description (Descripción)	Tipo de slot
failureAndRerunModo	failureAndRerunMode.	Enumeración
hadoopQueue	El nombre de cola de programador Hadoop en el que se enviará el trabajo.	Cadena
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	El número máximo de intentos en caso de error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referenci a, como «OnFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referencia, como "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referencia, como «onSuccess»: {"ref»:» myActionId «}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, como «parent»: {"ref»:» myBaseObj ect Id "}

Campos opcionales	Description (Descripción)	Tipo de slot
pipelineLogUri	El URI de S3 (como 's3://BucketName/Key/ ') para cargar los registros de la canalización.	Cadena
postActivityTaskConfig	Script de configuración después de la actividad que se va a ejecutar. Este consta de un URI del script de shell en Amazon S3 y una lista de argumentos.	Objeto de referencia, como "postActivityTaskConfig»: {"ref»:» myShellScriptConfigId «}
preActivityTaskConfig	Script de configuración antes de la actividad que se va a ejecutar. Este consta de un URI del script de shell en Amazon S3 y una lista de argumentos.	Objeto de referencia, como "preActivityTaskConfig»: {"ref»:» myShellScriptConfigId «}
precondition	Opcionalmente, defina una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condiciones previas.	Objeto de referencia, como «condición previa»: {"ref»:» myPreconditionId «}
reportProgressTimeout	Tiempo de espera para llamadas sucesivas del trabajo remoto a `reportProgress` . Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
resizeClusterBeforeEn ejecución	Cambiar el tamaño del clúster antes de realizar esta actividad para adaptarse a los nodos de datos de DynamoDB especificados como entradas o salidas. **Note** Si tu actividad usa un `DynamoDBDataNode` nodo de datos de entrada o de salida, y si lo configuras en `TRUE`, AWS Data Pipeline comienza `resizeClusterBeforeRunning` a usar tipos de `m3.xlarge` instancias. Se sobrescriben las opciones de tipo de instancia con `m3.xlarge`, lo que podría aumentar los costos mensuales.	Booleano
resizeClusterMaxInstancias	Un límite del número máximo de instancias que el algoritmo de cambio de tamaño puede solicitar.	Entero
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio del intervalo o al final de este. La programación de estilo de serie temporal significa que las instancia s se programan al final de cada intervalo y la programación de estilo cron significa que las instancias se programan al principio de cada intervalo. Un programa bajo demanda le permite ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa bajo demanda, debe especificarse en el objeto predeterminado y debe ser el único scheduleType especific ado para los objetos de la canalización. Para usar canalizaciones bajo demanda, basta con llamar a la ActivatePipeline operación para cada ejecución posterior. Los valores son: cron, ondemand y timeseries.	Enumeración

Campos opcionales	Description (Descripción)	Tipo de slot
scriptVariable	Especifica variables de script para que Amazon EMR se pase a Hive al ejecutar un script. Por ejemplo, las siguientes variables de script de ejemplo pasarían una variable SAMPLE y FILTER_DATE a Hive: `SAMPLE=s3://elasticmapreduce/samples/hive-ads y FILTER_DATE=#{format(@scheduledStartTime,'YYYY-MM-dd')}%` . Este campo acepta varios valores y funciona con los campos `script` y `scriptUri` . Además, `scriptVariable` funciona independientemente de si stage se establece en `true` o `false`. Este campo es especialmente útil para enviar valores dinámicos a Hive mediante expresiones y funciones de AWS Data Pipeline .	Cadena
etapa	Determina si el uso transitorio se habilita antes o después de la ejecución del script. No se permite con Hive 11, de modo que use una versión 3.2.0 o superior de AMI de Amazon EMR.	Booleano

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, como «ActiveInstances»: {"ref»:» myRunnableObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, como cascadeFailedOn «: {" ref»:» myRunnableObject Id "}
emrStepLog	Registros de pasos de Amazon EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada de un objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada de un objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, como «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Véase también

- ShellCommandActivity
- EmrActivity

HiveCopyActivity

Ejecuta una consulta de Hive en un clúster de EMR. `HiveCopyActivity` facilita la copia de datos entre las tablas de DynamoDB. `HiveCopyActivity` acepta una instrucción de HiveQL para filtrar datos de entrada desde DynamoDB en el nivel de columna y de fila.

Ejemplo

En el siguiente ejemplo se muestra cómo usar `HiveCopyActivity` y `DynamoDBExportDataFormat` para copiar datos de un nodo `DynamoDBDataNode` a otro, mientras se filtran datos, en función de una marca temporal.

```
{
  "objects": [
    {
      "id" : "DataFormat.1",
      "name" : "DataFormat.1",
      "type" : "DynamoDBExportDataFormat",
      "column" : "timeStamp BIGINT"
    },
    {
```

```
      "id" : "DataFormat.2",
      "name" : "DataFormat.2",
      "type" : "DynamoDBExportDataFormat"
    },
    {
      "id" : "DynamoDBDataNode.1",
      "name" : "DynamoDBDataNode.1",
      "type" : "DynamoDBDataNode",
      "tableName" : "item_mapped_table_restore_temp",
      "schedule" : { "ref" : "ResourcePeriod" },
      "dataFormat" : { "ref" : "DataFormat.1" }
    },
    {
      "id" : "DynamoDBDataNode.2",
      "name" : "DynamoDBDataNode.2",
      "type" : "DynamoDBDataNode",
      "tableName" : "restore_table",
      "region" : "us_west_1",
      "schedule" : { "ref" : "ResourcePeriod" },
      "dataFormat" : { "ref" : "DataFormat.2" }
    },
    {
      "id" : "EmrCluster.1",
      "name" : "EmrCluster.1",
      "type" : "EmrCluster",
      "schedule" : { "ref" : "ResourcePeriod" },
      "masterInstanceType" : "m1.xlarge",
      "coreInstanceCount" : "4"
    },
    {
      "id" : "HiveTransform.1",
      "name" : "Hive Copy Transform.1",
      "type" : "HiveCopyActivity",
      "input" : { "ref" : "DynamoDBDataNode.1" },
      "output" : { "ref" : "DynamoDBDataNode.2" },
      "schedule" :{ "ref" : "ResourcePeriod" },
      "runsOn" : { "ref" : "EmrCluster.1" },
      "filterSql" : "`timeStamp` > unix_timestamp(\"#{@scheduledStartTime}\", \"yyyy-
MM-dd'T'HH:mm:ss\")"
    },
    {
      "id" : "ResourcePeriod",
      "name" : "ResourcePeriod",
      "type" : "Schedule",
```

```
      "period" : "1 Hour",
      "startDateTime" : "2013-06-04T00:00:00",
      "endDateTime" : "2013-06-04T01:00:00"
    }
  ]
}
```

Sintaxis

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Los usuarios deben especificar una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Los usuarios pueden cumplir este requisito estableciendo explícitamente una programación en el objeto, por ejemplo, especificando «schedule»: {"ref»: "DefaultSchedule«}. En la mayoría de los casos, es mejor poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. O bien, si la canalización tiene un árbol de programas (programas dentro del programa maestro), los usuarios pueden crear un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	Objeto de referencia, por ejemplo, «schedule»: {"ref»:» myScheduleId «}

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
runsOn	Especifique el clúster en el que ejecutar.	Objeto de referencia, por ejemplo, «RunSon»: {"ref»:» myResourceId «}
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor `runs0n` y existe `workerGroup` , `workerGroup` se ignora.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	El estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	El tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio establecido.	Periodo
dependsOn	Especifica la dependencia de otro objeto ejecutable.	Objeto de referencia, por ejemplo, «DependSon»: {"ref»:» myActivityId «}
failureAndRerunModo	failureAndRerunMode.	Enumeración
filterSql	Un fragmento de instrucción SQL de Hive que filtra una subred de datos de DynamoDB o Amazon S3 que copiar. El filtro solo debe	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
	contener predicados y no empezar por una `WHERE` cláusula, ya que la AWS Data Pipeline agrega automáticamente.	
input	El origen de datos de entrada. Debe ser `S3DataNode` o `DynamoDBDataNode` . Si usa `DynamoDBNode` , especifique `DynamoDBExportDataFormat` .	Objeto de referencia, por ejemplo, «input»: {"ref»:» myDataNode Id "}
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	El número máximo de intentos en caso de error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}

Campos opcionales	Description (Descripción)	Tipo de slot
salida	El origen de datos de salida. Si la entrada es `S3DataNode` , este debe ser `DynamoDBDataNode` . De lo contrario, este puede ser `S3DataNode` o `DynamoDBDataNode` . Si usa `DynamoDBNode` , especifique `DynamoDBExportDataFormat` .	Objeto de referencia, por ejemplo, «output»: {"ref»:» myDataNode Id "}
parent	El elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
pipelineLogUri	El URI de Amazon S3 como `'s3://BucketName/Key/'` , para cargar logs para la canalización.	Cadena
postActivityTaskConfig	El script de configuración después de la actividad que se va a ejecutar. Este consta de un URI del script de shell en Amazon S3 y una lista de argumentos.	Objeto de referencia, por ejemplo, "postActivityTaskConfig»: {"ref»:» myShellScriptConfigId «}
preActivityTaskConfig	El script de configuración antes de la actividad que se va a ejecutar. Este consta de un URI del script de shell en Amazon S3 y una lista de argumentos.	Objeto de referencia, por ejemplo, "preActivityTaskConfig»: {"ref»:» myShellScriptConfigId «}
precondition	Opcionalmente define una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condiciones previas.	Objeto de referencia, por ejemplo, «condición previa»: {"ref»:» myPreconditionId «}

Campos opcionales	Description (Descripción)	Tipo de slot
reportProgressTimeout	El tiempo de espera para llamadas sucesivas del trabajo remoto a `reportProgress` . Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
resizeClusterBeforeCorriendo	Cambiar el tamaño del clúster antes de realizar esta actividad para adaptarse a los nodos de datos de DynamoDB especificados como entradas o salidas. **Note** Si tu actividad usa un `DynamoDBDataNode` nodo de datos de entrada o de salida, y si lo configuras en `TRUE`, AWS Data Pipeline comienza `resizeClusterBeforeRunning` a usar tipos de `m3.xlarge` instancias. Se sobrescriben las opciones de tipo de instancia con `m3.xlarge` , lo que podría aumentar los costos mensuales.	Booleano
resizeClusterMaxInstancias	Un límite del número máximo de instancias que el algoritmo de cambio de tamaño puede solicitar.	Entero
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio del intervalo o al final de este. La programación de estilo de serie temporal significa que las instancia s se programan al final de cada intervalo y la programación de estilo cron significa que las instancias se programan al principio de cada intervalo. Un programa bajo demanda le permite ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa bajo demanda, debe especificarse en el objeto predeterminado y debe ser el único scheduleType especific ado para los objetos de la canalización. Para usar canalizaciones bajo demanda, basta con llamar a la ActivatePipeline operación para cada ejecución posterior. Los valores son: cron, ondemand y timeseries.	Enumeración

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveIn stances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referenci a, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnabl eObject Id "}
emrStepLog	Registros de pasos de Amazon EMR disponibl es únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromI nstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@ healthSta tusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTi me	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCom pletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referenci a, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan un objeto de intento.	Cadena

Véase también

- ShellCommandActivity
- EmrActivity

PigActivity

PigActivity proporciona soporte nativo para los scripts de Pig AWS Data Pipeline sin la necesidad de usar `ShellCommandActivity` o`EmrActivity`. Además, PigActivity admite la puesta en escena de datos. Cuando el campo de uso transitorio se establece en true, AWS Data Pipeline almacena de modo transitorio los datos de entrada como un esquema en Pig sin código adicional del usuario.

Ejemplo

En la siguiente canalización de ejemplo se muestra cómo utilizar `PigActivity`. En la canalización de ejemplo se ejecutan los siguientes pasos:

- MyPigActivity1 carga datos de Amazon S3 y ejecuta un script Pig que selecciona algunas columnas de datos y las carga en Amazon S3.
- MyPigActivity2 carga la primera salida, selecciona algunas columnas y tres filas de datos y la carga en Amazon S3 como segunda salida.
- MyPigActivity3 carga los segundos datos de salida, inserta dos filas de datos y solo la columna denominada «quinta» en Amazon RDS.

- MyPigActivity4 carga los datos de Amazon RDS, selecciona la primera fila de datos y los carga en Amazon S3.

```
{
  "objects": [
    {
      "id": "MyInputData1",
      "schedule": {
        "ref": "MyEmrResourcePeriod"
      },
      "directoryPath": "s3://amzn-s3-demo-bucket/pigTestInput",
      "name": "MyInputData1",
      "dataFormat": {
        "ref": "MyInputDataType1"
      },
      "type": "S3DataNode"
    },
    {
      "id": "MyPigActivity4",
      "scheduleType": "CRON",
      "schedule": {
        "ref": "MyEmrResourcePeriod"
      },
      "input": {
        "ref": "MyOutputData3"
      },
      "pipelineLogUri": "s3://amzn-s3-demo-bucket/path/",
      "name": "MyPigActivity4",
      "runsOn": {
        "ref": "MyEmrResource"
      },
      "type": "PigActivity",
      "dependsOn": {
        "ref": "MyPigActivity3"
      },
      "output": {
        "ref": "MyOutputData4"
      },
      "script": "B = LIMIT ${input1} 1; ${output1} = FOREACH B GENERATE one;",
      "stage": "true"
    },
    {
```

```
      "id": "MyPigActivity3",
      "scheduleType": "CRON",
      "schedule": {
        "ref": "MyEmrResourcePeriod"
      },
      "input": {
        "ref": "MyOutputData2"
      },
      "pipelineLogUri": "s3://amzn-s3-demo-bucket/path",
      "name": "MyPigActivity3",
      "runsOn": {
        "ref": "MyEmrResource"
      },
      "script": "B = LIMIT ${input1} 2; ${output1} = FOREACH B GENERATE Fifth;",
      "type": "PigActivity",
      "dependsOn": {
        "ref": "MyPigActivity2"
      },
      "output": {
        "ref": "MyOutputData3"
      },
      "stage": "true"
    },
    {
      "id": "MyOutputData2",
      "schedule": {
        "ref": "MyEmrResourcePeriod"
      },
      "name": "MyOutputData2",
      "directoryPath": "s3://amzn-s3-demo-bucket/PigActivityOutput2",
      "dataFormat": {
        "ref": "MyOutputDataType2"
      },
      "type": "S3DataNode"
    },
    {
      "id": "MyOutputData1",
      "schedule": {
        "ref": "MyEmrResourcePeriod"
      },
      "name": "MyOutputData1",
      "directoryPath": "s3://amzn-s3-demo-bucket/PigActivityOutput1",
      "dataFormat": {
        "ref": "MyOutputDataType1"
```

```
      },
      "type": "S3DataNode"
    },
    {
      "id": "MyInputDataType1",
      "name": "MyInputDataType1",
      "column": [
        "First STRING",
        "Second STRING",
        "Third STRING",
        "Fourth STRING",
        "Fifth STRING",
        "Sixth STRING",
        "Seventh STRING",
        "Eighth STRING",
        "Ninth STRING",
        "Tenth STRING"
      ],
      "inputRegEx": "^(\\S+) (\\S+) (\\S+) (\\S+) (\\S+) (\\S+) (\\S+) (\
\S+) (\\S+) (\\S+)",
      "type": "RegEx"
    },
    {
      "id": "MyEmrResource",
      "region": "us-east-1",
      "schedule": {
        "ref": "MyEmrResourcePeriod"
      },
      "keyPair": "example-keypair",
      "masterInstanceType": "m1.small",
      "enableDebugging": "true",
      "name": "MyEmrResource",
      "actionOnTaskFailure": "continue",
      "type": "EmrCluster"
    },
    {
      "id": "MyOutputDataType4",
      "name": "MyOutputDataType4",
      "column": "one STRING",
      "type": "CSV"
    },
    {
      "id": "MyOutputData4",
      "schedule": {
```

```
        "ref": "MyEmrResourcePeriod"
      },
      "directoryPath": "s3://amzn-s3-demo-bucket/PigActivityOutput3",
      "name": "MyOutputData4",
      "dataFormat": {
        "ref": "MyOutputDataType4"
      },
      "type": "S3DataNode"
    },
    {
      "id": "MyOutputDataType1",
      "name": "MyOutputDataType1",
      "column": [
        "First STRING",
        "Second STRING",
        "Third STRING",
        "Fourth STRING",
        "Fifth STRING",
        "Sixth STRING",
        "Seventh STRING",
        "Eighth STRING"
      ],
      "columnSeparator": "*",
      "type": "Custom"
    },
    {
      "id": "MyOutputData3",
      "username": "___",
      "schedule": {
        "ref": "MyEmrResourcePeriod"
      },
      "insertQuery": "insert into #{table} (one) values (?)",
      "name": "MyOutputData3",
      "*password": "___",
      "runsOn": {
        "ref": "MyEmrResource"
      },
      "connectionString": "jdbc:mysql://example-database-instance:3306/example-
database",
      "selectQuery": "select * from #{table}",
      "table": "example-table-name",
      "type": "MySqlDataNode"
    },
    {
```

```
      "id": "MyOutputDataType2",
      "name": "MyOutputDataType2",
      "column": [
        "Third STRING",
        "Fourth STRING",
        "Fifth STRING",
        "Sixth STRING",
        "Seventh STRING",
        "Eighth STRING"
      ],
      "type": "TSV"
    },
    {
      "id": "MyPigActivity2",
      "scheduleType": "CRON",
      "schedule": {
        "ref": "MyEmrResourcePeriod"
      },
      "input": {
        "ref": "MyOutputData1"
      },
      "pipelineLogUri": "s3://amzn-s3-demo-bucket/path",
      "name": "MyPigActivity2",
      "runsOn": {
        "ref": "MyEmrResource"
      },
      "dependsOn": {
        "ref": "MyPigActivity1"
      },
      "type": "PigActivity",
      "script": "B = LIMIT ${input1} 3; ${output1} = FOREACH B GENERATE Third, Fourth,
Fifth, Sixth, Seventh, Eighth;",
      "output": {
        "ref": "MyOutputData2"
      },
      "stage": "true"
    },
    {
      "id": "MyEmrResourcePeriod",
      "startDateTime": "2013-05-20T00:00:00",
      "name": "MyEmrResourcePeriod",
      "period": "1 day",
      "type": "Schedule",
      "endDateTime": "2013-05-21T00:00:00"
```

```
    },
    {
      "id": "MyPigActivity1",
      "scheduleType": "CRON",
      "schedule": {
        "ref": "MyEmrResourcePeriod"
      },
      "input": {
        "ref": "MyInputData1"
      },
      "pipelineLogUri": "s3://amzn-s3-demo-bucket/path",
      "scriptUri": "s3://amzn-s3-demo-bucket/script/pigTestScipt.q",
      "name": "MyPigActivity1",
      "runsOn": {
        "ref": "MyEmrResource"
      },
      "scriptVariable": [
        "column1=First",
        "column2=Second",
        "three=3"
      ],
      "type": "PigActivity",
      "output": {
        "ref": "MyOutputData1"
      },
      "stage": "true"
    }
  ]
}
```

El contenido de `pigTestScript.q` es el siguiente.

```
B = LIMIT ${input1} $three; ${output1} = FOREACH B GENERATE $column1, $column2, Third,
 Fourth, Fifth, Sixth, Seventh, Eighth;
```

Sintaxis

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Los usuarios	Objeto de referencia, por ejemplo,

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
	deben especificar una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Los usuarios pueden cumplir este requisito estableciendo explícitamente una programación en el objeto, por ejemplo, especificando «schedule»: {"ref»: "DefaultSchedule«}. En la mayoría de los casos, es mejor poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. O bien, si la canalización tiene un árbol de programas (programas dentro del programa maestro), los usuarios pueden crear un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	«schedule»: {"ref»:» mScheduleId «}

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
script	El script de Pig que se ejecutará.	Cadena
scriptUri	La ubicación del script de Pig que se ejecutará (por ejemplo, s3://scriptLocation).	Cadena

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
runsOn	Clúster EMR en el que se PigActivity ejecuta.	Objeto de referencia, por ejemplo, «RunSon»: {"ref»:» myEmrCluster Id "}
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor `runsOn` y existe `workerGroup` , `workerGroup` se ignora.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	El estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	El tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio establecido.	Periodo
dependsOn	Especifica la dependencia de otro objeto ejecutable.	Objeto de referencia, por ejemplo, «dependSon»: {"ref»:» myActivityId «}
failureAndRerunModo	failureAndRerunMode.	Enumeración
input	El origen de datos de entrada.	Objeto de referencia, por ejemplo, «input»:

Campos opcionales	Description (Descripción)	Tipo de slot
		{"ref»:» myDataNode Id "}
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en ondemand.	Periodo
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	El número máximo de intentos en caso de error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
salida	El origen de datos de salida.	Objeto de referencia, por ejemplo, «output»: {"ref»:» myDataNode Id "}

Campos opcionales	Description (Descripción)	Tipo de slot
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
pipelineLogUri	El URI de Amazon S3 (como 's3://BucketName/Key/ ') para cargar los registros de la canalización.	Cadena
postActivityTaskConfig	Script de configuración después de la actividad que se va a ejecutar. Este consta de un URI del script del intérprete de comandos en Amazon S3 y una lista de argumentos.	Objeto de referencia, por ejemplo, "postActivityTaskConfig»: {"ref»:» myShellScriptConfigId «}
preActivityTaskConfig	Script de configuración antes de la actividad que se va a ejecutar. Este consta de un URI del script de shell en Amazon S3 y una lista de argumentos.	Objeto de referencia, por ejemplo, "preActivityTaskConfig»: {"ref»:» myShellScriptConfigId «}
precondition	Opcionalmente, defina una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condiciones previas.	Objeto de referencia, por ejemplo, «condición previa»: {"ref»:» myPreconditionId «}
reportProgressTimeout	El tiempo de espera para llamadas sucesivas del trabajo remoto a `reportProgress` . Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
resizeClusterBeforeEn ejecución	Cambiar el tamaño del clúster antes de realizar esta actividad para adaptarse a los nodos de datos de DynamoDB especificados como entradas o salidas. **Note** Si tu actividad usa un `DynamoDBDataNode` nodo de datos de entrada o de salida, y si lo configuras en `TRUE`, AWS Data Pipeline comienza `resizeClusterBeforeRunning` a usar tipos de `m3.xlarge` instancias. Se sobrescriben las opciones de tipo de instancia con `m3.xlarge`, lo que podría aumentar los costos mensuales.	Booleano
resizeClusterMaxInstancias	Un límite del número máximo de instancias que el algoritmo de cambio de tamaño puede solicitar.	Entero
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio del intervalo o al final de este. La programación de estilo de serie temporal significa que las instancia s se programan al final de cada intervalo y la programación de estilo cron significa que las instancias se programan al principio de cada intervalo. Un programa bajo demanda le permite ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa bajo demanda, debe especificarse en el objeto predeterminado y debe ser el único scheduleType especific ado para los objetos de la canalización. Para usar canalizaciones bajo demanda, basta con llamar a la ActivatePipeline operación para cada ejecución posterior. Los valores son: cron, ondemand y timeseries.	Enumeración
scriptVariable	Los argumentos que se pasan al script de Pig. Puede usar scriptVariable con script o scriptUri.	Cadena
etapa	Determina si la puesta en escena está habilitad a y permite que tu script de Pig tenga acceso a las tablas de datos escalonados, como $ {INPUT1} y $ {}. OUTPUT1	Booleano

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveIn

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
		stances»: {"ref»:» Id "} myRunnableObject
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referenci a, por ejemplo, cascadeFailedOn «: {" ref»:» myRunnabl eObject Id "}
emrStepLog	Registros de pasos de Amazon EMR disponibl es únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Véase también

- ShellCommandActivity
- EmrActivity

RedshiftCopyActivity

Copia datos de DynamoDB o Amazon S3 en Amazon Redshift. Puede cargar datos en una nueva tabla o combinar datos fácilmente en una tabla existente.

A continuación, se muestra información general de un caso de uso en el que utilizar `RedshiftCopyActivity`:

1. Comience por usar AWS Data Pipeline para organizar sus datos en Amazon S3.

2. Utilice `RedshiftCopyActivity` para mover los datos de Amazon RDS y Amazon EMR a Amazon Redshift.

   Esto le permite cargar sus datos en Amazon Redshift para poder analizarlos.
3. Utilice SqlActivity para realizar consultas SQL en los datos que ha cargado en Amazon Redshift.

Además, `RedshiftCopyActivity` le permite trabajar con un `S3DataNode`, dado que admite un archivo de manifiesto. Para obtener más información, consulte S3 DataNode.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto.

Para garantizar la conversión de formatos, este ejemplo utiliza los parámetros de conversión especiales EMPTYASNULL e IGNOREBLANKLINES en `commandOptions`. Para obtener más información, consulte Parámetros de conversión de datos en la Guía de desarrollador de base de datos de Amazon Redshift.

```
{
  "id" : "S3ToRedshiftCopyActivity",
  "type" : "RedshiftCopyActivity",
  "input" : { "ref": "MyS3DataNode" },
  "output" : { "ref": "MyRedshiftDataNode" },
  "insertMode" : "KEEP_EXISTING",
  "schedule" : { "ref": "Hour" },
  "runsOn" : { "ref": "MyEc2Resource" },
  "commandOptions": ["EMPTYASNULL", "IGNOREBLANKLINES"]
}
```

En la siguiente definición de canalización de ejemplo se muestra una actividad que usa el modo de inserción APPEND:

```
{
  "objects": [
    {
      "id": "CSVId1",
      "name": "DefaultCSV1",
      "type": "CSV"
    },
    {
      "id": "RedshiftDatabaseId1",
```

```
      "databaseName": "dbname",
      "username": "user",
      "name": "DefaultRedshiftDatabase1",
      "*password": "password",
      "type": "RedshiftDatabase",
      "clusterId": "redshiftclusterId"
    },
    {
      "id": "Default",
      "scheduleType": "timeseries",
      "failureAndRerunMode": "CASCADE",
      "name": "Default",
      "role": "DataPipelineDefaultRole",
      "resourceRole": "DataPipelineDefaultResourceRole"
    },
    {
      "id": "RedshiftDataNodeId1",
      "schedule": {
        "ref": "ScheduleId1"
      },
      "tableName": "orders",
      "name": "DefaultRedshiftDataNode1",
      "createTableSql": "create table StructuredLogs (requestBeginTime CHAR(30)
PRIMARY KEY DISTKEY SORTKEY, requestEndTime CHAR(30), hostname CHAR(100), requestDate
varchar(20));",
      "type": "RedshiftDataNode",
      "database": {
        "ref": "RedshiftDatabaseId1"
      }
    },
    {
      "id": "Ec2ResourceId1",
      "schedule": {
        "ref": "ScheduleId1"
      },
      "securityGroups": "MySecurityGroup",
      "name": "DefaultEc2Resource1",
      "role": "DataPipelineDefaultRole",
      "logUri": "s3://myLogs",
      "resourceRole": "DataPipelineDefaultResourceRole",
      "type": "Ec2Resource"
    },
    {
      "id": "ScheduleId1",
```

```
      "startDateTime": "yyyy-mm-ddT00:00:00",
      "name": "DefaultSchedule1",
      "type": "Schedule",
      "period": "period",
      "endDateTime": "yyyy-mm-ddT00:00:00"
    },
    {
      "id": "S3DataNodeId1",
      "schedule": {
        "ref": "ScheduleId1"
      },
      "filePath": "s3://datapipeline-us-east-1/samples/hive-ads-samples.csv",
      "name": "DefaultS3DataNode1",
      "dataFormat": {
        "ref": "CSVId1"
      },
      "type": "S3DataNode"
    },
    {
      "id": "RedshiftCopyActivityId1",
      "input": {
        "ref": "S3DataNodeId1"
      },
      "schedule": {
        "ref": "ScheduleId1"
      },
      "insertMode": "APPEND",
      "name": "DefaultRedshiftCopyActivity1",
      "runsOn": {
        "ref": "Ec2ResourceId1"
      },
      "type": "RedshiftCopyActivity",
      "output": {
        "ref": "RedshiftDataNodeId1"
      }
    }
  ]
}
```

La operación APPEND añade elementos a una tabla independientemente de las claves principales o de ordenación. Por ejemplo, si tiene la tabla siguiente, puede incluir un registro con el mismo valor de usuario e ID.

```
ID(PK)     USER
1          aaa
2          bbb
```

Puede incluir un registro con el mismo valor de usuario e ID:

```
ID(PK)     USER
1          aaa
2          bbb
1          aaa
```

> **Note**
>
> Si una operación APPEND se interrumpe y reintenta, la nueva ejecución de la canalización resultante podría iniciar la operación desde el principio. Esto puede ocasionar una duplicación adicional, por lo que debe ser consciente de este comportamiento, especialmente si tiene cualquier lógica que cuente el número de filas.

Para ver un tutorial, consulte Copiar datos a Amazon Redshift con AWS Data Pipeline.

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
insertMode	Determina qué AWS Data Pipeline ocurre con los datos preexistentes de la tabla de destino que se superponen con las filas de los datos que se van a cargar. Los valores válidos son: KEEP_EXISTING , OVERWRITE_EXISTING , TRUNCATE y APPEND. KEEP_EXISTING añade nuevas filas a la tabla y deja sin modificar las filas existentes. KEEP_EXISTING y OVERWRITE_EXISTING utilizan la clave principal, de	Enumeración

Campos obligatorios	Description (Descripción)	Tipo de slot
	ordenación y las claves de distribución para identificar qué filas entrantes se correspon den con filas existentes. Consulte Actualiza ción e inserción de datos nuevos en la Guía de desarrollador de base de datos de Amazon Redshift. TRUNCATE elimina todos los datos de la tabla de destino antes de escribir los nuevos datos. APPEND añade todos los registros al final de la tabla de Redshift. APPEND no requiere una clave principal, de distribución o de ordenació n, por lo que se podrían agregar elementos que pueden ser duplicados.	

Campos de invocació n de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Especifique una referencia de programac ión a otro objeto para establecer el orden de ejecución de dependencia para este objeto. En la mayoría de los casos, recomendamos poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. Por ejemplo, puede establecer un programa en el objeto de forma explícita especificando `"schedule": {"ref": "DefaultSchedule"}` .	Objeto de referenci a, como por ejemplo: `"schedule ":{"ref": "mySchedu leId"}`

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
	Si el programa maestro de la canalización contiene programas anidados, cree un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte Programación.	

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
runsOn	El recurso informático para ejecutar la actividad o comando. Por ejemplo, una EC2 instancia de Amazon o un clúster de Amazon EMR.	Objeto de referencia, por ejemplo, «RunSon»: {"ref»:» myResourceId «}
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor `runs0n` y existe `workerGroup` , workerGroup se ignora.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
	complete dentro del tiempo de inicio establecido.	

Campos opcionales	Description (Descripción)	Tipo de slot
commandOptions	Toma parámetros para pasar al nodo de datos de Amazon Redshift durante la operación `COPY`. Para más información sobre los parámetros, consulte COPIAR en la Guía para desarrolladores de bases de datos de Amazon Redshift. A medida que carga la tabla, `COPY` intenta convertir de forma implícita las cadenas al tipo de datos de la columna de destino. Además de las conversiones de datos predeterminadas que ocurren automáticamente, si recibe errores o tiene otras necesidades de conversión, puede especificar parámetros de conversión adicionales. Para obtener más información, consulte Parámetros de conversión de datos en la Guía de desarrollador de base de datos de Amazon Redshift. Si un formato de datos está asociado al nodo de datos de entrada o salida, los parámetros proporcionados se omiten. Dado que la operación de copia utiliza primero `COPY` para insertar los datos en una tabla provisional y, a continuación, utiliza un comando `INSERT` para copiar los datos desde la tabla provisional a la tabla de destino, algunos parámetros `COPY` no se aplican, como la capacidad del comando `COPY` para permitir la compresión automática de la tabla. Si la compresión es necesaria, añada los detalles de codificación de columna a la instrucción `CREATE TABLE`.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
	Además, en algunos casos en que es necesario descargar datos del clúster de Amazon Redshift y crear archivos en Amazon S3, `RedshiftCopyActivity` se basa en la operación `UNLOAD` de Amazon Redshift. Para mejorar el rendimiento durante la copia y la descarga, especifique el parámetro `PARALLEL OFF` del comando `UNLOAD`. Para obtener más información sobre los parámetros, consulte DESCARGAR en la Guía de desarrollador de base de datos de Amazon Redshift.	
dependsOn	Especificar la dependencia de otro objeto ejecutable.	Objeto de referencia: `"dependsOn": {"ref":"myActivityId"}`
failureAndRerunModo	Describe el comportamiento del nodo del consumidor cuando las dependencias producen un error o se vuelven a ejecutar.	Enumeración
input	El nodo de datos de entrada. El origen de datos puede ser Amazon S3, DynamoDB o Amazon Redshift.	Objeto de referencia: `"input": {"ref":"myDataNodeId"}`
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia: `"onFail": {"ref":"myActionId"}`
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referencia: `"onLateAction": {"ref":"myActionId"}`
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referencia: `"onSuccess": {"ref":"myActionId"}`
salida	El nodo de datos de salida. La ubicación de salida puede ser Amazon S3 o Amazon Redshift.	Objeto de referencia: `"output": {"ref":"myDataNodeId"}`
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia: `"parent": {"ref":"myBaseObjectId"}`
pipelineLogUri	El URI de S3 (como 's3://BucketName/Key/ ') para cargar los registros de la canalización.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
precondition	Opcionalmente, defina una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condiciones previas.	Objeto de referencia: `"precondition": {"ref":"myPreconditionId"}`
cola	Se corresponde a la configuración de `query_group` en Amazon Redshift que le permite asignar y priorizar actividades simultáneas en función de su ubicación en las colas. Amazon Redshift limita el número de conexiones simultáneas a 15. Para obtener más información, consulte Asignación de consultas a las colas en la Guía de desarrollador de base de datos de Amazon RDS.	Cadena
reportProgressTimeout	Tiempo de espera para llamadas sucesivas del trabajo remoto a `reportProgress` . Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
scheduleType	Le permite especificar la programación de objetos en su canalización. Los valores son: `cron`, `ondemand` y `timeseries` . La programación `timeseries` significa que las instancias se programan al final de cada intervalo. La programación `Cron` significa que las instancias se programan al principio de cada intervalo. Un programa `ondemand` le permite ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Para usar canalizaciones `ondemand`, solo tiene que llamar a la operación `ActivateP ipeline` para cada ejecución posterior. Si usa un programa `ondemand`, debe especific arlo en el objeto predeterminado y debe ser el único `scheduleType` especificado para los objetos de la canalización.	Enumeración

Campos opcionales	Description (Descripción)	Tipo de slot
transformSql	La expresión `SQL SELECT` que se utiliza para transformar los datos de entrada. Ejecute la expresión `transformSql` en la tabla denominada `staging`. Cuando se copian datos desde DynamoDB o Amazon S3, AWS Data Pipeline crea una tabla denominada “staging” y carga los datos en ella inicialmente. Los datos de esta tabla se utilizan para actualizar la tabla de destino. El esquema de salida de `transformSql` debe coincidir con el esquema de la tabla de destino final. Si especifica la opción `transformSql` , se crea una segunda tabla provisional a partir de la instrucción SQL especificada. Los datos de esta segunda tabla staging se actualizan en la tabla de destino final.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia: `"activeInstances": {"ref":"myRunnableObjectId"}`
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia: `"cascadeFailedOn": {"ref":"myRunnableObjectId"}`
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ Hora healthStatusUpdated	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia: `"waitingOn": {"ref":"myRunnableObjectId"}`

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto. Denota su lugar en el ciclo de vida. Por ejemplo, los objetos de componente dan lugar a objetos de instancia, que ejecutan objetos de intento.	Cadena

ShellCommandActivity

Ejecuta un comando o script. Puede usar `ShellCommandActivity` para ejecutar tareas programadas de serie temporal o similar a Cron.

Cuando el `stage` campo se establece en verdadero y se usa con un`S3DataNode`, `ShellCommandActivity` admite el concepto de datos de almacenamiento provisional, lo que significa que puede mover los datos de Amazon S3 a una ubicación de escenario, como Amazon EC2 o su entorno local, trabajar con los datos mediante scripts y el `ShellCommandActivity` y volver a moverlos a Amazon S3.

En este caso, cuando su comando de shell está conectado a un nodo `S3DataNode` de entrada, sus scripts de shell operan directamente en los datos mediante `${INPUT1_STAGING_DIR}`, `${INPUT2_STAGING_DIR}` y otros campos, que hacen referencia a los campos de entrada `ShellCommandActivity`.

De forma similar, la salida del comando del intérprete de comandos se puede almacenar de modo transitorio en un directorio de salida que se va a insertar automáticamente en Amazon S3, al que hacen referencia `${OUTPUT1_STAGING_DIR}`, `${OUTPUT2_STAGING_DIR}`, etc.

Estas expresiones pueden pasar como argumentos de línea de comandos al comando de shell para su uso en la lógica de transformación de datos.

`ShellCommandActivity` devuelve cadenas y códigos de error estilo Linux. Si `ShellCommandActivity` genera un error, el `error` devuelto es un valor distinto de cero.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto.

```
{
  "id" : "CreateDirectory",
  "type" : "ShellCommandActivity",
  "command" : "mkdir new-directory"
}
```

Sintaxis

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de `schedule`. Para establecer el orden de ejecución de dependencia de este objeto, especifique una referencia `schedule` a otro objeto. Para cumplir este requisito, establezca de forma explícita un `schedule` en el objeto, por ejemplo, especificando `"schedule": {"ref": "DefaultSchedule"}` . En la mayoría de los casos, es mejor poner la referencia de `schedule` en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. Si la canalización consta un árbol de programas (programas dentro del programa maestro), cree un objeto principal que tenga una referencia de programa. Para distribuir la carga, AWS Data Pipeline crea objetos físicos un poco antes de lo previsto, pero los ejecuta según lo programado.	Objeto de referencia, por ejemplo, «schedule»: {"ref»:» myScheduleId «}

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
	Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
comando	El comando que se va a ejecutar. Utilice $ para hacer referencia a parámetros posicionales y `scriptArgument` para especificar los parámetros para el comando. Este valor y cualquier parámetro asociado debe funcionar en el entorno desde el que se está ejecutando Task Runner.	Cadena
scriptUri	Una ruta del URI de Amazon S3 para que se descargue un archivo y se ejecute como comando de shell. Especifique solo un campo `scriptUri` o `command`. `scriptUri` no puede utilizar parámetros; utilice `command` en su lugar.	Cadena

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
runsOn	El recurso computacional para ejecutar la actividad o el comando, por ejemplo, una EC2	Objeto de referencia, por ejemplo,

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
	instancia de Amazon o un clúster de Amazon EMR.	«RunSon»: {"ref»:» myResourceId «}
workerGroup	Utilizado para dirigir tareas. Si proporciona un valor `runs0n` y existe `workerGroup` , `workerGroup` se ignora.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	El estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	El tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio especific ado.	Periodo
dependsOn	Especifica una dependencia de otro objeto ejecutable.	Objeto de referenci a, por ejemplo, «DependSon»: {"ref»:» myActivityId «}
failureAndRerunModo	failureAndRerunMode.	Enumeración
input	La ubicación de los datos de entrada.	Objeto de referencia, por ejemplo, «input»: {"ref»:» myDataNode Id "}
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
	completarse. Solo se activa cuando el tipo de programación no está establecido en ondemand.	
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	El número máximo de intentos en caso de error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
salida	La ubicación de los datos de salida.	Objeto de referencia, por ejemplo, «output»: {"ref»:» myDataNode Id "}
parent	El elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}

Campos opcionales	Description (Descripción)	Tipo de slot
pipelineLogUri	El URI de Amazon S3, como `'s3://BucketName/Key/'` para cargar registros para la canalización.	Cadena
precondition	Opcionalmente define una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condiciones previas.	Objeto de referencia, por ejemplo, «condición previa»: {"ref»:» myPreconditionId «}
reportProgressTimeout	El tiempo de espera para llamadas sucesivas a `reportProgress` por parte de actividades remotas. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
scheduleType	Le permite especificar si los objetos de la definición de la canalización deben programar se al principio del intervalo o al final de este. Los valores posibles son: `cron`, `ondemand` y `timeseries` . Si se establece en `timeseries` , las instancia s se programan al final de cada intervalo. Si se establece en `Cron`, las instancias se programan al inicio de cada intervalo. Si se establece en `ondemand`, puede ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa `ondemand`, especifíquelo en el objeto predeterminado como `scheduleType` único para los objetos de la canalización. Para usar canalizaciones `ondemand`, solo tiene que llamar a la operación `ActivatePipeline` para cada ejecución posterior.	Enumeración

Campos opcionales	Description (Descripción)	Tipo de slot
scriptArgument	Una serie de cadenas con formato JSON que se pasan al comando especificado por el comando. Por ejemplo, si el comando es `echo $1 $2`, especifique `scriptArgument` como `"param1", "param2"`. Para varios argumentos y parámetros, pase el `scriptArgument` del siguiente modo: `"scriptArgument":"arg1","scriptArgument":"param1","scriptArgument":"arg2","scriptArgument":"param2"`. El `scriptArgument` solo se puede utilizar con `command`; si se utiliza con `scriptUri` produce un error.	Cadena
etapa	Determina si está habilitado el espacio transitorio y permite que los comandos de shell tengan acceso a las variables de datos en el espacio transitorio, como `${INPUT1_STAGING_DIR}` y `${OUTPUT1_STAGING_DIR}`.	Booleano
stderr	La ruta que recibe los mensajes de error del sistema redirigidos desde el comando. Si utiliza el campo `runsOn`, esta debe ser una ruta de Amazon S3 debido a la naturaleza transitoria del recurso que ejecuta su actividad. No obstante, si especifica el campo `workerGroup`, se permite una ruta de archivo local.	Cadena
stdout	La ruta de Amazon S3 que recibe la salida redirigida del comando. Si utiliza el campo `runsOn`, esta debe ser una ruta de Amazon S3 debido a la naturaleza transitoria del recurso que ejecuta su actividad. No obstante, si especifica el campo `workerGroup`, se permite una ruta de archivo local.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	La lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El `cancellationReason` de este objeto se ha cancelado.	Cadena
@cascadeFailedOn	La descripción de la cadena de dependencias que provocó el error del objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}
emrStepLog	Registros de pasos de Amazon EMR disponibles únicamente sobre intentos de actividad de Amazon EMR.	Cadena
errorId	El `errorId` si este objeto ha fallado.	Cadena
errorMessage	El `errorMessage` si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que el objeto finalizó su ejecución.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en Amazon EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	El ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	La hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	La hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	La hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	La hora de la ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	La hora de finalización programada para el objeto.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@scheduledStartTime	La hora de comienzo programada para el objeto.	DateTime
@status	El estado del objeto.	Cadena
@version	La AWS Data Pipeline versión utilizada para crear el objeto.	Cadena
@waitingOn	La descripción de la lista de dependencias para la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	El error al describir el objeto mal estructurado.	Cadena
@pipelineId	El ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	El lugar de un objeto en el ciclo de vida. Los objetos de componente dan lugar a objetos de instancia, que ejecutan objetos de intento.	Cadena

Véase también

- CopyActivity
- EmrActivity

SqlActivity

Ejecuta una consulta SQL (script) en una base de datos.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto.

```
{
  "id" : "MySqlActivity",
  "type" : "SqlActivity",
  "database" : { "ref": "MyDatabaseID" },
  "script" : "SQLQuery" | "scriptUri" : s3://scriptBucket/query.sql,
  "schedule" : { "ref": "MyScheduleID" },
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
database	La base de datos en la que se ejecuta el script SQL suministrado.	Objeto de referencia, por ejemplo, «database»: {"ref»:» myDatabaseId «}

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Debe especificar una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Puede establecer un programa en el objeto de forma explícita, por ejemplo, especificando `"schedule": {"ref": "DefaultSchedule"}` . En la mayoría de los casos, es mejor poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa.	Objeto de referencia, por ejemplo, «schedule»: {"ref»:» mScheduleId «}

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
	Si la canalización tiene un árbol de programas anidados dentro del programa maestro, cree un objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
script	El script SQL que se va a ejecutar. Debe especificar script o scriptUri. Cuando el script se almacena en Amazon S3, script no se evalúa como una expresión. Especificar múltiples valores para scriptArgument es útil cuando el script se almacena en Amazon S3.	Cadena
scriptUri	Un URI que especifica la ubicación de un script de SQL para ejecutar en esta actividad.	Cadena

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
runsOn	El recurso informático para ejecutar la actividad o comando. Por ejemplo, una EC2 instancia de Amazon o un clúster de Amazon EMR.	Objeto de referencia, por ejemplo, «RunSon»: {"ref»:» myResourceId «}

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
workerGroup	El grupo de procesos de trabajo. Este se usa para dirigir tareas. Si proporciona un valor `runsOn` y existe `workerGroup` , `workerGroup` se ignora.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio establecido.	Periodo
dependsOn	Especificar la dependencia de otro objeto ejecutable.	Objeto de referencia, por ejemplo, «DependSon»: {"ref»:» myActivityId «}
failureAndRerunModo	Describe el comportamiento del nodo del consumidor cuando las dependencias producen un error o se vuelven a ejecutar.	Enumeración
input	Ubicación de los datos de entrada.	Objeto de referencia, por ejemplo, «input»: {"ref»:» myDataNode Id "}

Campos opcionales	Description (Descripción)	Tipo de slot
lateAfterTimeout	El período de tiempo desde el principio del programa de la canalización dentro del cual debe comenzar la ejecución del objeto.	Periodo
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deberían activarse si un objeto aún no se ha programado o aún no se ha completado en el período transcurrido desde el inicio programado de la canalización, tal como se especifica en 'lateAfterTimeout'.	Objeto de referencia, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referencia, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
salida	Ubicación de los datos de salida. Esto solo es útil para hacer referencia desde un script (por ejemplo`#{output.tablename}`) y para crear la tabla de salida configurando 'createTableSql' en el nodo de datos de salida. La salida de la consulta SQL no se escribe en el nodo de datos de salida.	Objeto de referencia, por ejemplo, «output»: {"ref»:» myDataNodeId "}

Campos opcionales	Description (Descripción)	Tipo de slot
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
pipelineLogUri	El URI de S3 (como 's3://BucketName/Key/ ') para cargar los registros de la canalización.	Cadena
precondition	Opcionalmente, defina una condición previa. Un nodo de datos no se marca como "READY" hasta que se han cumplido todas las condiciones previas.	Objeto de referencia, por ejemplo, «condición previa»: {"ref»:» «} myPreconditionId
cola	[Amazon Redshift solamente] Corresponde a la configuración de query_group en Amazon Redshift, que le permite asignar y priorizar actividades simultáneas en función de su ubicación en las colas. Amazon Redshift limita el número de conexiones simultáneas a 15. Para obtener más información, consulte Asignación de consultas a las colas en la Guía de desarrollador de base de datos de Amazon Redshift.	Cadena
reportProgressTimeout	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio del intervalo o al final de este. Los valores son: `cron`, `ondemand` y `timeseries` . La programación `timeseries` significa que las instancias se programan al final de cada intervalo. La programación `cron` significa que las instancias se programan al principio de cada intervalo. Un programa `ondemand` le permite ejecutar una canalización una vez por activación. Esto significa que no tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa `ondemand`, debe especificarse en el objeto predeterminado y debe ser el único `scheduleType` especificado para los objetos de la canalización. Para usar canalizaciones `ondemand`, solo tiene que llamar a la operación `ActivatePipeline` para cada ejecución posterior.	Enumeración
scriptArgument	Una lista de variables para el script. También puede colocar expresiones directamente en el campo del script. Especificar múltiples valores para scriptArgument es útil cuando el script se almacena en Amazon S3. Ejemplo: #{format (@scheduledStartTime, "YY-MM-DD HH:MM:SS"}\n# {format (PlusPeriod (@scheduledStartTime, «1 día»), "HH:MM:SS"} YY-MM-DD	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» Id "} myRunnableObject
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Recursos

Los siguientes son los objetos de AWS Data Pipeline recursos:

Objects

- Ec2Resource
- EmrCluster
- HttpProxy

Ec2Resource

EC2 Instancia de Amazon que realiza el trabajo definido por una actividad de canalización.

AWS Data Pipeline ahora es compatible con IMDSv2 la EC2 instancia de Amazon, que utiliza un método orientado a la sesión para gestionar mejor la autenticación al recuperar la información de metadatos de las instancias. Una sesión inicia y finaliza una serie de solicitudes que el software que se ejecuta en una EC2 instancia de Amazon utiliza para acceder a los metadatos y credenciales de la EC2 instancia de Amazon almacenados localmente. El software inicia una sesión con una simple solicitud HTTP PUT a. IMDSv2 IMDSv2 devuelve un token secreto al software que se ejecuta en la EC2 instancia de Amazon, que utilizará el token como contraseña IMDSv2 para realizar solicitudes de metadatos y credenciales.

> **Note**
>
> IMDSv2 Para usarlo en tu EC2 instancia de Amazon, tendrás que modificar la configuración, ya que la AMI predeterminada no es compatible con ella IMDSv2. Puede especificar una nueva versión de AMI que puede recuperar mediante el siguiente parámetro SSM: `/aws/service/ami-amazon-linux-latest/amzn-ami-hvm-x86_64-ebs`.

Para obtener información sobre EC2 las instancias de Amazon predeterminadas que AWS Data Pipeline crea si no especificas una instancia, consulta Instancias Amazon EC2 predeterminadas por región de AWS.

Ejemplos

EC2-Clásico

> **Important**
>
> Solo AWS las cuentas creadas antes del 4 de diciembre de 2013 son compatibles con la plataforma EC2 -Classic. Si tiene una de estas cuentas, puede que tenga la opción de crear objetos de EC2 recursos para una canalización en una red EC2 clásica en lugar de en una VPC. Te recomendamos encarecidamente que crees recursos para todas tus canalizaciones. VPCs Además, si tiene recursos existentes en EC2 -Classic, le recomendamos que los migre a una VPC.

El siguiente objeto de ejemplo lanza una EC2 instancia en EC2 -Classic, con algunos campos opcionales configurados.

```
{
  "id" : "MyEC2Resource",
  "type" : "Ec2Resource",
  "actionOnTaskFailure" : "terminate",
  "actionOnResourceFailure" : "retryAll",
  "maximumRetries" : "1",
  "instanceType" : "m5.large",
  "securityGroups" : [
    "test-group",
    "default"
  ],
  "keyPair" : "my-key-pair"
}
```

EC2-PVC

El siguiente objeto de ejemplo lanza una EC2 instancia en una VPC no predeterminada, con algunos campos opcionales configurados.

```
{
  "id" : "MyEC2Resource",
  "type" : "Ec2Resource",
  "actionOnTaskFailure" : "terminate",
  "actionOnResourceFailure" : "retryAll",
  "maximumRetries" : "1",
  "instanceType" : "m5.large",
  "securityGroupIds" : [
    "sg-12345678",
    "sg-12345678"
  ],
  "subnetId": "subnet-12345678",
  "associatePublicIpAddress": "true",
  "keyPair" : "my-key-pair"
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
resourceRole	La función de IAM que controla los recursos a los que puede acceder la EC2 instancia de Amazon.	Cadena
rol	El rol de IAM que se AWS Data Pipeline utiliza para crear la EC2 instancia.	Cadena

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Para establecer el orden de ejecución de dependencia para este objeto, especifique una referencia de programación a otro objeto. Puedes hacerlo de una de las siguientes formas: • Para garantizar que todos los objetos de la canalización heredan la programación, establezca una programación en el objeto explícitamente: `"schedule": {"ref": "DefaultSchedule"}` . En la mayoría de los casos, resulta útil poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden esa programación. • Si la canalización tiene programaciones anidadas en la programación maestra, puede crear un objeto principal que tenga una referencia de programación. Para	Objeto de referencia, por ejemplo, `"schedule":{"ref": "mySchedu leId"}`

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
	obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

Campos opcionales	Description (Descripción)	Tipo de slot
actionOnResourceFallo	La acción realizada después de un error de este recurso. Los valores válidos son `"retryall"` y `"retrynone"` .	Cadena
actionOnTaskFallo	La acción realizada después de un error de tarea de este recurso. Los valores válidos son `"continue"` o `"terminate"` .	Cadena
associatePublicIpDirección	Indica si se va a asignar una dirección IP pública a la instancia. Si la instancia está en Amazon EC2 o Amazon VPC, el valor predeterminado es. `true` De lo contrario, el valor predeterminado es `false`.	Booleano
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio especificado.	Periodo
availabilityZone	La zona de disponibilidad en la que se lanzará la EC2 instancia de Amazon.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
inhabilitar IMDSv1	El valor predeterminado es false y habilita tanto IMDSv1 y IMDSv2. Si lo establece en verdadero, se deshabilita IMDSv1 y solo proporciona IMDSv2s	Booleano
failureAndRerunModo	failureAndRerunMode.	Enumeración
httpProxy	El host proxy que utilizan los clientes para conectarse a AWS los servicios.	Objeto de referencia, por ejemplo, `"httpProxy": {"ref":"myHttpProxyId"}`
imageId	El ID de la AMI que se va a utilizar para la instancia. De forma predeterminada, AWS Data Pipeline utiliza el tipo de virtualización AMI de HVM. El AMI específico IDs utilizado se basa en una región. Puede sobrescribir la AMI predeterminada especificando la AMI HVM que desee. Para obtener más información sobre los tipos de AMI, consulte Tipos de virtualización de AMI de Linux y Búsqueda de una AMI de Linux en la Guía del EC2 usuario de Amazon.	Cadena
initTimeout	El tiempo que se debe esperar a que se inicie el recurso.	Periodo
instanceCount	Obsoleto.	Entero
instanceType	El tipo de EC2 instancia de Amazon que se va a iniciar.	Cadena
keyPair	El nombre del par de claves. Si lanzas una EC2 instancia de Amazon sin especificar un key pair, no podrás iniciar sesión en ella.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	El número máximo de intentos en caso de error.	Entero
minInstanceCount	Obsoleto.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, `"onFail": {"ref":"myActionId"}`
onLateAction	Acciones que deben iniciarse si un objeto no se ha programado o sigue ejecutándose.	Objeto de referencia, por ejemplo, `"onLateAction": {"ref":"myActionId"}`
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referencia, por ejemplo, `"onSuccess": {"ref":"myActionId"}`

Campos opcionales	Description (Descripción)	Tipo de slot
parent	El elemento principal del objeto actual del que se heredan las ranuras.	Objeto de referencia, por ejemplo, `"parent": {"ref":"myBaseObjectId"}`
pipelineLogUri	El URI de Amazon S3 (como `'s3://BucketName/Key/'`) para cargar registros para la canalización.	Cadena
region	El código de la región en la que debe ejecutarse la EC2 instancia de Amazon. De forma predeterminada, la instancia se ejecuta en la misma región que la canalización. Puede ejecutar la instancia en la misma región que un conjunto de datos dependiente.	Enumeración
reportProgressTimeout	El tiempo de espera para llamadas sucesivas del trabajo remoto a `reportProgress` . Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y se reintentarán.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo
runAsUser	El usuario que ejecutará el TaskRunner.	Cadena
runsOn	Este campo no está permitido en este objeto.	Objeto de referencia, por ejemplo, `"runsOn": {"ref":"myResourceId"}`

Campos opcionales	Description (Descripción)	Tipo de slot
scheduleType	El tipo de programación le permite especificar si los objetos de la definición de la canalizac ión deben programarse al principio o al final del intervalo, o bajo demanda. Valores son los siguientes: • `timeseries` . Las instancias se programan al final de cada intervalo. • `cron`. Las instancias se programan al comienzo de cada intervalo. • `ondemand`. Le permite ejecutar una canalización una vez por activación. No tiene que clonar o recrear la canalizac ión para ejecutarla de nuevo. Si usa un programa bajo demanda, debe especificarse en el objeto predeterminado y debe ser el único `scheduleType` especificado para los objetos de la canalización. Para usar canalizaciones bajo demanda, solo tiene que llamar a la operación `ActivatePipeline` para cada ejecución posterior.	Enumeración
securityGroupIds	El IDs de uno o más grupos de EC2 seguridad de Amazon que se van a utilizar para las instancias del grupo de recursos.	Cadena
securityGroups	Uno o más grupos EC2 de seguridad de Amazon para usarlos en las instancias del grupo de recursos.	Cadena
spotBidPrice	La cantidad máxima por hora para su instancia de spot en dólares, que es un valor decimal entre 0 y 20,00 (no incluidos).	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
subnetId	El ID de la EC2 subred de Amazon en la que se va a iniciar la instancia.	Cadena
terminateAfter	El número de horas después de las cuales se ha de finalizar el recurso.	Periodo
useOnDemandOnLastAttempt	En el último intento de solicitar una instancia de spot, realice una solicitud de instancias bajo demanda en lugar de instancias de spot. De este modo, se garantiza que si todos los intentos anteriores han fallado, el último intento no se verá interrumpido.	Booleano
workerGroup	Este campo no está permitido en este objeto.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, `"activeInstances": {"ref":"myRunnableObjectId"}`
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El `cancellationReason` de este objeto se ha cancelado.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@cascadeFailedOn	Descripción de la cadena de dependencias en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, `"cascadeFailedOn": {"ref":"myRunnableObjectId"}`
emrStepLog	Los registros de pasos solo están disponibles en los intentos de actividad de Amazon EMR.	Cadena
errorId	El ID de error si este objeto ha fallado.	Cadena
errorMessage	El mensaje de error si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@failureReason	El motivo del error del recurso.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades de Amazon EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ Hora healthStatusUpdated	Hora a la que el estado de salud se actualizó la última vez.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	La hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	La hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	La versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias para la que este objeto está a la espera.	Objeto de referencia, por ejemplo, `"waitingOn": {"ref":"myRunnableObjectId"}`

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	El lugar de un objeto en el ciclo de vida. Los objetos de componente dan lugar a objetos de instancia, que ejecutan objetos de intento.	Cadena

EmrCluster

Representa la configuración de un clúster de Amazon EMR. EmrActivity y HadoopActivity usan este objeto para lanzar un clúster.

Contenido

Programadores

Los programadores ofrecen un modo de especificar la asignación de recursos y la priorización de trabajo en un clúster de Hadoop. Los administradores o usuarios pueden elegir un programador para diversas clases de usuarios y aplicaciones. Un programador podría usar colas para asignar recursos a usuarios y aplicaciones. Usted configura esas colas al crear el clúster. A continuación, puede configurar la prioridad de determinados tipos de trabajo y usuario sobre otros. Esto proporciona un uso eficaz de los recursos del clúster, a la vez que se permite a más de un usuario enviar trabajo al clúster. Existen tres tipos de programador disponibles:

- FairScheduler— Intenta programar los recursos de manera uniforme durante un período de tiempo significativo.

- CapacityScheduler— Utiliza colas para permitir a los administradores de clústeres asignar usuarios a colas de diferente prioridad y asignación de recursos.
- Predeterminado: usado por el clúster, de cuya configuración podría encargarse su sitio.

Versiones de lanzamiento de Amazon EMR

Una versión de Amazon EMR es un conjunto de aplicaciones de código abierto del ecosistema de macrodatos. Cada versión incluye diferentes aplicaciones, componentes y características de macrodatos que selecciona para que Amazon EMR los instale y configure al crear un clúster. La versión se especifica mediante la etiqueta de versión. Las etiquetas de versión tienen el formato `emr-x.x.x` Por ejemplo, `emr-5.30.0`. Clústeres de Amazon EMR basados en la etiqueta de versión`emr-4.0.0` y posteriormente utilizan la propiedad `releaseLabel` para especificar la etiqueta de lanzamiento de un objeto `EmrCluster`. Las versiones anteriores utilizan la propiedad `amiVersion`.

> **Important**
>
> Todos los clústeres de All Amazon EMR creados con la versión 5.22.0 o posterior usan la firma de Signature Version 4 para autenticar las solicitudes en Amazon S3. Algunas versiones anteriores usan Signature Version 2. Se está interrumpiendo la compatibilidad con Signature Version 2. Para obtener más información, consulte Amazon S3 Update — Sigv2 Deprecation Period Extended and Modified (Actualización de Amazon S3: período de desaprobación de Sigv2 extendido y modificado). Recomendamos encarecidamente que utilice una versión de Amazon EMR compatible con Signature Version 4. Para versiones anteriores, comenzando con EMR 4.7.x, la versión más reciente de la serie se ha actualizado para admitir Signature Version 4. Cuando utilice una versión anterior de EMR, le recomendamos que utilice la versión más reciente de la serie. Además, evite las versiones anteriores a EMR 4.7.0.

Condiciones y limitaciones

Utilice la última versión de Task Runner

Si usa un objeto `EmrCluster` autoadministrado con una etiqueta de versión, utilice la Task Runner más actual. Para obtener más información acerca de Task Runner, consulte Operación de Task Runner. Puede configurar valores de propiedad para todas las clasificaciones de configuración de Amazon EMR. Para obtener más información, consulte Configuring Applications en la Guía de

lanzamiento de Amazon EMR, las the section called “EmrConfiguration” y las referencias de objeto the section called “Propiedad”.

Support para IMDSv2

Anteriormente, solo AWS Data Pipeline compatible IMDSv1. Ahora, AWS Data Pipeline es compatible con IMDSv2 Amazon EMR 5.23.1, 5.27.1 y 5.32 o versiones posteriores, y Amazon EMR 6.2 o versiones posteriores. IMDSv2 utiliza un método orientado a la sesión para gestionar mejor la autenticación al recuperar la información de metadatos de las instancias. Debes configurar tus instancias para realizar IMDSv2 llamadas mediante la creación de recursos administrados por los usuarios mediante -2.0. TaskRunner

Amazon EMR 5.32 o posterior y Amazon EMR 6.x

La serie de versiones 5.32 o posteriores y 6.x de Amazon EMR utiliza la versión 3.x de Hadoop, que introdujo cambios importantes en la forma en que se evalúa la ruta de clases de Hadoop en comparación con la versión 2.x de Hadoop. Las bibliotecas más comunes, como Joda-Time, se eliminaron de la ruta de clases.

Si EmrActivity o HadoopActivity ejecuta un archivo Jar que depende de una biblioteca que se eliminó en Hadoop 3.x, el paso no se realizará correctamente y mostrará el error `java.lang.NoClassDefFoundError` o `java.lang.ClassNotFoundException`. Esto puede ocurrir con los archivos Jar que se ejecutaron sin problemas con las versiones de lanzamiento 5.x de Amazon EMR.

Para solucionar el problema, debe copiar las dependencias del archivo Jar a la ruta de clases de Hadoop de un objeto `EmrCluster` antes de iniciar la actividad `EmrActivity` o `HadoopActivity`. Proporcionamos un script bash para hacerlo. El script bash está disponible en la siguiente ubicación, donde *`MyRegion`* se encuentra la AWS región en la que se ejecuta el `EmrCluster` objeto, por ejemplo. `us-west-2`

```
s3://datapipeline-MyRegion/MyRegion/bootstrap-actions/latest/TaskRunner/copy-jars-to-hadoop-classpath.sh
```

La forma de ejecutar el script depende de si `EmrActivity` `HadoopActivity` se ejecuta en un recurso administrado por AWS Data Pipeline o se ejecuta en un recurso autogestionado.

Si utiliza un recurso administrado por AWS Data Pipeline, añada un `bootstrapAction` al `EmrCluster` objeto. `bootstrapAction` especifica el script y los archivos Jar que se van a copiar

como argumentos. Puede añadir hasta 255 campos `bootstrapAction` por objeto `EmrCluster` y puede añadir un campo `bootstrapAction` a un objeto `EmrCluster` que ya tenga acciones de arranque.

Para especificar este script como una acción de arranque, utilice la siguiente sintaxis, donde `JarFileRegion` es la región en la que se guarda el archivo Jar y cada una *`MyJarFilen`* es la ruta absoluta en Amazon S3 de un archivo Jar que se va a copiar en la ruta de clases de Hadoop. No especifique los archivos Jar que estén en la ruta de clases de Hadoop de forma predeterminada.

```
s3://datapipeline-MyRegion/MyRegion/bootstrap-actions/latest/TaskRunner/copy-jars-to-
hadoop-classpath.sh,JarFileRegion,MyJarFile1,MyJarFile2[, ...]
```

El siguiente ejemplo especifica una acción de arranque que copia dos archivos Jar en Amazon S3: `my-jar-file.jar` y `emr-dynamodb-tool-4.14.0-jar-with-dependencies.jar`. La región utilizada en el ejemplo es us-west-2.

```
{
  "id" : "MyEmrCluster",
  "type" : "EmrCluster",
  "keyPair" : "my-key-pair",
  "masterInstanceType" : "m5.xlarge",
  "coreInstanceType" : "m5.xlarge",
  "coreInstanceCount" : "2",
  "taskInstanceType" : "m5.xlarge",
  "taskInstanceCount": "2",
  "bootstrapAction" : ["s3://datapipeline-us-west-2/us-west-2/bootstrap-actions/
latest/TaskRunner/copy-jars-to-hadoop-classpath.sh,us-west-2,s3://path/to/my-jar-
file.jar,s3://dynamodb-dpl-us-west-2/emr-ddb-storage-handler/4.14.0/emr-dynamodb-
tools-4.14.0-jar-with-dependencies.jar"]
}
```

Debe guardar y activar la canalización para que se aplique el cambio a la nueva `bootstrapAction`.

Si utiliza un recurso autogestionado, puede descargar el script en la instancia del clúster y ejecutarlo desde la línea de comandos mediante SSH. El script crea un directorio llamado "`/etc/hadoop/conf/shellprofile.d`" y un archivo llamado "`datapipeline-jars.sh`" en dicho directorio. Los archivos jar proporcionados como argumentos de la línea de comandos se copian en un directorio que el script crea llamado "`/home/hadoop/datapipeline_jars`". Si el clúster está configurado de forma diferente, modifique el script adecuadamente después de descargarlo.

La sintaxis para ejecutar el script en la línea de comandos es ligeramente diferente a la que se muestra la `bootstrapAction` en el ejemplo anterior. Utilice espacios en lugar de comas entre argumentos, como se muestra en el siguiente ejemplo.

```
./copy-jars-to-hadoop-classpath.sh us-west-2 s3://path/to/my-jar-file.jar s3://
dynamodb-dpl-us-west-2/emr-ddb-storage-handler/4.14.0/emr-dynamodb-tools-4.14.0-jar-
with-dependencies.jar
```

Permisos de Amazon EMR

Al crear un rol de IAM personalizado, piense detenidamente en los permisos mínimos necesarios para que su clúster realice su trabajo. Asegúrese de conceder acceso a los recursos necesarios, como archivos de Amazon S3 o datos de Amazon RDS, Amazon Redshift o DynamoDB. Si desea establecer `visibleToAllUsers` en False, su rol debe tener los permisos adecuados para hacerlo. Tenga en cuenta que `DataPipelineDefaultRole` no tiene estos permisos. Debe proporcionar una unión de los roles `DefaultDataPipelineResourceRole` y `DataPipelineDefaultRole` como el rol de objeto `EmrCluster` o crear su propio rol con este fin.

Sintaxis

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
horario	Este objeto se invoca dentro de la ejecución de un intervalo de programación. Especifique una referencia de programación a otro objeto para establecer el orden de ejecución de dependencia para este objeto. Puede cumplir este requisito estableciendo de forma explícita un programa en el objeto, por ejemplo, especificando `"schedule": {"ref": "DefaultSchedule"}` . En la mayoría de los casos, es mejor poner la referencia de programación en el objeto de la canalización predeterminado de modo que todos los objetos hereden ese programa. O bien, si la canalización tiene un árbol de programas (programas dentro del programa maestro), puede crear un	Objeto de referencia, por ejemplo, `"schedule":{"ref": "myScheduleId"}`

Campos de invocación de objetos	Description (Descripción)	Tipo de slot
	objeto principal que tenga una referencia de programación. Para obtener más información acerca de las configuraciones de programación opcionales de ejemplo, consulte https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-object-schedule.html	

Campos opcionales	Description (Descripción)	Tipo de slot
actionOnResourceError	La acción realizada después de un error de este recurso. Los valores válidos son "`retryall`", que reintenta todas las tareas en el clúster durante el tiempo especificado y "`retrynone `".	Cadena
actionOnTaskFallo	La acción realizada después de un error de tarea de este recurso. Los valores válidos son "continue", que significa que no debe terminar se el clúster, y "terminate".	Cadena
additionalMasterSecurityGroupIds	El identificador de los grupos de seguridad maestros adicionales del clúster de EMR, que sigue el formulario sg-01. XXXX6a Para obtener más información, consulte Grupos de seguridad adicionales de Amazon EMR en la Guía de administración de Amazon EMR.	Cadena
additionalSlaveSecurityGroupIds	El identificador de los grupos de seguridad secundarios adicionales del clúster de EMR, que sigue el formato `sg-01XXXX6a` .	Cadena
amiVersion	La versión de Imagen de máquina de Amazon (AMI) que Amazon EMR utiliza para instalar los	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
	nodos del clúster. Para obtener más informaci ón, consulte la Guía de administración de Amazon EMR.	
aplicaciones	Aplicaciones para instalar en el clúster con argumentos separados por comas. De forma predeterminada, están instalados Hive y Pig. Este parámetro se aplica solamente a la versión 4.0 y posteriores de Amazon EMR.	Cadena
attemptStatus	El estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio estableci do.	Periodo
availabilityZone	La zona de disponibilidad en la que ejecutar el clúster.	Cadena
bootstrapAction	Una acción que se ejecuta cuando comienza el clúster. Puede especificar argumentos separados por comas. Para especificar varias acciones, hasta 255, añada varios campos `bootstrapAction` . El comportamiento predeterminado consiste en comenzar el clúster sin ninguna acción de arranque.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
configuración	Configuración para el clúster de Amazon EMR. Este parámetro se aplica solamente a la versión 4.0 y posteriores de Amazon EMR.	Objeto de referencia, por ejemplo, `"configuration":{"ref":"myEmrConfigurationId"}`
coreInstanceBidPrecio	El precio spot máximo que está dispuesto a pagar por las EC2 instancias de Amazon. Si se especifica un precio de puja, Amazon EMR utiliza instancias de spot para el grupo de instancias. Se especifica en USD.	Cadena
coreInstanceCount	El número de nodos principales que se van a utilizar para el clúster.	Entero
coreInstanceType	El tipo de EC2 instancia de Amazon que se utilizará para los nodos principales. Consulte Instancias Amazon EC2 admitidas para clústeres de Amazon EMR .	Cadena
coreGroupConfiguration	La configuración del grupo de instancias principales del clúster de Amazon EMR. Este parámetro se aplica solamente a la versión 4.0 y posteriores de Amazon EMR.	Objeto de referencia, por ejemplo, `“configuration”: {“ref”: “myEmrConfigurationId”}`

Campos opcionales	Description (Descripción)	Tipo de slot
coreEbsConfiguration	La configuración de los volúmenes de Amazon EBS que se asociarán a cada uno de los nodos principales del grupo principal en el clúster de Amazon EMR. Para obtener más información, consulte Tipos de instancias que soportan la optimización de EBS en la Guía del EC2 usuario de Amazon.	Objeto de referencia, por ejemplo, `"coreEbsConfiguration": {"ref": "myEbsConfiguration"}`
customAmiId	Solo se aplica a las versiones 5.7.0 y posteriores de Amazon EMR. Especifica el ID de AMI de una AMI personalizada que se utilizará cuando Amazon EMR aprovisione instancias de Amazon EC2 . También se puede usar en lugar de acciones de arranque para personalizar las configuraciones de los nodos del clúster. Para obtener más información, consulte el siguiente tema en la Guía de administración de Amazon EMR. Uso de una AMI personalizada	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
EbsBlockDeviceConfig	La configuración de un dispositivo de bloques de Amazon EBS solicitado asociado al grupo de instancias. Incluye un determinado número de volúmenes que se asociará a cada instancia del grupo de instancias. Incluye `volumesPerInstance` y `volumeSpecification`, donde: • `volumesPerInstance` es el número de volúmenes de EBS con una configuración de volumen específica que se asociarán a cada instancia del grupo de instancias. • `volumeSpecification` son las especificaciones de volumen de Amazon EBS, como el tipo de volumen, las IOPS y el tamaño en Gigibytes (GiB) que se solicitarán para el volumen de EBS adjunto a una instancia EC2 del clúster de Amazon EMR.	Objeto de referencia, por ejemplo, `“EbsBlockDeviceConfig”: {“ref”: “myEbsBlockDeviceConfig”}`
emrManagedMasterSecurityGroupId	El identificador del grupo de seguridad principal del clúster de Amazon EMR, que sigue el formato de `sg-01XXXX6a`. Para obtener más información, consulte Configurar grupos de seguridad en la Guía de administración de Amazon EMR.	Cadena
emrManagedSlaveSecurityGroupId	El identificador del grupo de seguridad secundario del clúster de Amazon EMR, que sigue el formato de `sg-01XXXX6a`.	Cadena
enableDebugging	Habilita la depuración en el clúster de Amazon EMR.	Cadena
failureAndRerunModo	failureAndRerunMode.	Enumeración

Campos opcionales	Description (Descripción)	Tipo de slot
hadoopSchedulerType	El tipo de programador del clúster. Los tipos válidos son: `PARALLEL_FAIR_SCHEDULING` , `PARALLEL_CAPACITY_SCHEDULING` y `DEFAULT_SCHEDULER` .	Enumeración
httpProxy	El host proxy que usan los clientes para conectarse a los servicios de AWS.	Objeto de referencia, por ejemplo, «HttpProxy»: {"ref»:» myHttpProxy Id "}
initTimeout	El tiempo que se debe esperar a que se inicie el recurso.	Periodo
keyPair	El par de EC2 claves de Amazon que se utilizará para iniciar sesión en el nodo principal del clúster de Amazon EMR.	Cadena
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo
masterInstanceBidPrecio	El precio spot máximo que está dispuesto a pagar por las EC2 instancias de Amazon. Un valor decimal entre 0 y 20,00, exclusivo. Se especifica en USD. Al establecer este valor se habilitan las instancias de subasta para el nodo principal del clúster de Amazon EMR. Si se especifica un precio de puja, Amazon EMR utiliza instancias de spot para el grupo de instancias.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
masterInstanceType	El tipo de EC2 instancia de Amazon que se utilizará para el nodo principal. Consulte Instancias Amazon EC2 admitidas para clústeres de Amazon EMR .	Cadena
masterGroupConfiguration	La configuración del grupo de instancias maestro del clúster de Amazon EMR. Este parámetro se aplica solamente a la versión 4.0 y posteriores de Amazon EMR.	Objeto de referencia, por ejemplo, `“configuration”: {“ref”: “myEmrConfigurationId”}`
masterEbsConfiguration	La configuración de volúmenes de Amazon EBS que se asociará a cada uno de los nodos principales del grupo maestro en el clúster de Amazon EMR. Para obtener más información, consulte Tipos de instancias que soportan la optimización de EBS en la Guía del EC2 usuario de Amazon.	Objeto de referencia, por ejemplo, `“masterEbsConfiguration”: {“ref”: “myEbsConfiguration”}`
maxActiveInstances	El número máximo de instancias activas simultáneas de un componente. Las nuevas ejecuciones no cuentan para el número de instancias activas.	Entero
maximumRetries	maximumRetries.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, `"onFail": {"ref":"myActionId"}`

Campos opcionales	Description (Descripción)	Tipo de slot
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referencia, por ejemplo, `"onLateAction": {"ref":"myActionId"}`
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referencia, por ejemplo, `"onSuccess": {"ref":"myActionId"}`
parent	Elemento principal del objeto actual del que se heredan los slots.	Objeto de referencia, por ejemplo, `"parent": {"ref":"myBaseObjectId"}`
pipelineLogUri	El URI de Amazon S3 (como 's3://BucketName/Key/ ') para cargar los registros de la canalización.	Cadena
region	El código de la región en la que debe ejecutarse el clúster de Amazon EMR. De forma predeterminada, el clúster se ejecuta en la misma región que la canalización. Puede ejecutar el clúster en la misma región que un conjunto de datos dependiente.	Enumeración
releaseLabel	Etiqueta de la versión del clúster de EMR.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
reportProgressTimeout	Tiempo de espera para llamadas sucesivas del trabajo remoto a `reportProgress` . Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
resourceRole	La función de IAM que se AWS Data Pipeline utiliza para crear el clúster de Amazon EMR. El rol predeterminado es `DataPipelineDefaultRole` .	Cadena
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo
rol	La función de IAM se pasó a Amazon EMR para EC2 crear nodos.	Cadena
runsOn	Este campo no está permitido en este objeto.	Objeto de referencia, por ejemplo, `"runsOn": {"ref":"myResourceId"}`
SecurityConfiguration	El identificador de la configuración de seguridad de EMR que se aplicará al clúster. Este parámetro se aplica solamente a la versión 4.8.0 y posteriores de Amazon EMR.	Cadena
serviceAccessSecurityGroupId	El identificador del grupo de seguridad de acceso a los servicios del clúster de Amazon EMR.	Cadena. Sigue el formato `sg-01XXXX6a` , por ejemplo, `sg-1234abcd` .

Campos opcionales	Description (Descripción)	Tipo de slot
scheduleType	El tipo de programa le permite especificar si los objetos de la definición de la canalización deben programarse al principio o al final del intervalo. Los valores son: `cron`, `ondemand` y `timeseries` . La programación `timeseries` significa que las instancias se programan al final de cada intervalo. La programación `cron` significa que las instancias se programan al principio de cada intervalo. Un programa `ondemand` le permite ejecutar una canalización una vez por activación. No tiene que clonar o recrear la canalización para ejecutarla de nuevo. Si usa un programa `ondemand`, debe especificarse en el objeto predeterminado y debe ser el único `scheduleType` especificado para los objetos de la canalización. Para usar canalizaciones `ondemand`, solo tiene que llamar a la operación `ActivatePipeline` para cada ejecución posterior.	Enumeración
subnetId	El identificador de la subred en la que se lanza el clúster de Amazon EMR.	Cadena
supportedProducts	Un parámetro que instala software de terceros en un clúster de Amazon EMR, por ejemplo, una distribución de terceros de Hadoop.	Cadena
taskInstanceBidPrecio	El precio spot máximo que está dispuesto a pagar por EC2 las instancias. Un valor decimal entre 0 y 20,00, exclusivo. Se especifica en USD. Si se especifica un precio de puja, Amazon EMR utiliza instancias de spot para el grupo de instancias.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
taskInstanceCount	El número de nodos de tarea que se van a utilizar para el clúster de Amazon EMR.	Entero
taskInstanceType	El tipo de EC2 instancia de Amazon que se utilizará para los nodos de tareas.	Cadena
taskGroupConfiguration	La configuración del grupo de instancias de tareas del clúster de Amazon EMR. Este parámetro se aplica solamente a la versión 4.0 y posteriores de Amazon EMR.	Objeto de referencia, por ejemplo, `“configuration”: {“ref”: “myEmrConfigurationId”}`
taskEbsConfiguration	La configuración de los volúmenes de Amazon EBS que se asociarán a cada uno de los nodos de tarea del grupo de tareas en el clúster de Amazon EMR. Para obtener más información, consulte Tipos de instancias que soportan la optimización de EBS en la Guía del EC2 usuario de Amazon.	Objeto de referencia, por ejemplo, `“taskEbsConfiguration”: {“ref”: “myEbsConfiguration”}`
terminateAfter	Termina el recurso una vez transcurridas estas horas.	Entero

Campos opcionales	Description (Descripción)	Tipo de slot
VolumeSpecification	Las especificaciones de volumen de Amazon EBS, como el tipo de volumen, las IOPS y el tamaño en Gigibytes (GiB) que se solicitarán para el volumen de Amazon EBS adjunto a una EC2 instancia de Amazon en el clúster de Amazon EMR. El nodo puede ser un nodo principal, maestro o de tarea. El `VolumeSpecification` incluye: • `iops()` Entero. El número de I/O operaciones por segundo (IOPS) que admite el volumen de Amazon EBS, por ejemplo, 1000. Para obtener más información, consulte Características de E/S de EBS en la Guía EC2 del usuario de Amazon. • `sizeinGB()` . Entero. El tamaño del volumen de Amazon EBS, en gibibytes (GiB), por ejemplo 500. Para obtener información sobre las combinaciones válidas de tipos de volumen y tamaños de disco duro, consulte Tipos de volumen de EBS en la Guía del EC2 usuario de Amazon. • `volumetType` . Cadena. El tipo de volumen de Amazon EBS, por ejemplo, gp2. Entre los tipos de volumen admitidos se incluyen el estándar, gp2, io1, st1, sc1 y otros. Para obtener más información, consulte Tipos de volumen de EBS en la Guía del EC2 usuario de Amazon.	Objeto de referencia, por ejemplo, `“VolumeSpecification”: {“ref”: “myVolumeSpecification”}`

Campos opcionales	Description (Descripción)	Tipo de slot
useOnDemandOnLastAttempt	En el último intento de solicitar un recurso, haga una solicitud de instancias bajo demanda en lugar de instancias de spot. De este modo, se garantiza que si todos los intentos anteriores han fallado, el último intento no se verá interrumpido.	Booleano
workerGroup	Campo no permitido en este objeto.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencias en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, cascadeFailedOn «: {" ref»:» myRunnableObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
emrStepLog	Los registros de pasos de Amazon EMR están disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El ID de error si este objeto ha fallado.	Cadena
errorMessage	El mensaje de error si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
@failureReason	El motivo del error del recurso.	Cadena
@finishedTime	La hora a la que este objeto finalizó su ejecución.	DateTime
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades de Amazon EMR.	Cadena
@healthStatus	El estado de salud del objeto que refleja el éxito o el fracaso de la última instancia de objeto que alcanzó un estado terminado.	Cadena
@healthStatusFromInstanceId	ID del último objeto de instancia que alcanzó un estado terminado.	Cadena
@ healthStatusUpdated Hora	Hora a la que el estado de salud se actualizó la última vez.	DateTime
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
@lastDeactivatedTime	La hora a la que este objeto se desactivó la última vez.	DateTime
@ latestCompletedRun Hora	Hora de la última ejecución para la que se completó la ejecución.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@latestRunTime	Hora de la última ejecución para la que se programó la ejecución.	DateTime
@nextRunTime	Hora de ejecución que se va a programar a continuación.	DateTime
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias para la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	El lugar de un objeto en el ciclo de vida. Los objetos de componente dan lugar a objetos de instancia, que ejecutan objetos de intento.	Cadena

Ejemplos

A continuación se muestran ejemplos de este tipo de objeto.

Contenido

Lanzar un clúster de Amazon EMR con hadoopVersion

Example

En el siguiente ejemplo se lanza un clúster de Amazon EMR mediante la versión de AMI 1.0 y Hadoop 0.20.

```
{
  "id" : "MyEmrCluster",
  "type" : "EmrCluster",
  "hadoopVersion" : "0.20",
  "keyPair" : "my-key-pair",
  "masterInstanceType" : "m3.xlarge",
  "coreInstanceType" : "m3.xlarge",
  "coreInstanceCount" : "10",
  "taskInstanceType" : "m3.xlarge",
  "taskInstanceCount": "10",
  "bootstrapAction" : ["s3://Region.elasticmapreduce/bootstrap-actions/configure-
hadoop,arg1,arg2,arg3","s3://Region.elasticmapreduce/bootstrap-actions/configure-
hadoop/configure-other-stuff,arg1,arg2"]
}
```

Lanzar un clúster de Amazon EMR con la etiqueta de versión emr-4.x o posterior

Example

En el siguiente ejemplo se lanza un clúster de Amazon EMR mediante el campo `releaseLabel` más reciente:

```
{
  "id" : "MyEmrCluster",
  "type" : "EmrCluster",
  "keyPair" : "my-key-pair",
  "masterInstanceType" : "m3.xlarge",
  "coreInstanceType" : "m3.xlarge",
  "coreInstanceCount" : "10",
  "taskInstanceType" : "m3.xlarge",
  "taskInstanceCount": "10",
  "releaseLabel": "emr-4.1.0",
  "applications": ["spark", "hive", "pig"],
  "configuration": {"ref":"myConfiguration"}
}
```

Instalar software adicional en el clúster de Amazon EMR

Example

`EmrCluster` proporciona el campo `supportedProducts` que instala software de terceros en un clúster de Amazon EMR; por ejemplo, permite instalar una distribución personalizada de Hadoop, como MapR. Acepta una lista de argumentos separada por comas para que el software de terceros la lea y actúe. En el siguiente ejemplo se muestra cómo usar el campo `supportedProducts` de `EmrCluster` para crear un clúster de edición MapR M3 personalizada con Karmasphere Analytics instalado y ejecutar un objeto `EmrActivity` en él.

```
{
    "id": "MyEmrActivity",
    "type": "EmrActivity",
    "schedule": {"ref": "ResourcePeriod"},
    "runsOn": {"ref": "MyEmrCluster"},
    "postStepCommand": "echo Ending job >> /mnt/var/log/stepCommand.txt",
    "preStepCommand": "echo Starting job > /mnt/var/log/stepCommand.txt",
    "step": "/home/hadoop/contrib/streaming/hadoop-streaming.jar,-input,s3n://
elasticmapreduce/samples/wordcount/input,-output, \
```

```
     hdfs:///output32113/,-mapper,s3n://elasticmapreduce/samples/wordcount/
wordSplitter.py,-reducer,aggregate"
  },
  {
    "id": "MyEmrCluster",
    "type": "EmrCluster",
    "schedule": {"ref": "ResourcePeriod"},
    "supportedProducts": ["mapr,--edition,m3,--version,1.2,--key1,value1","karmasphere-
enterprise-utility"],
    "masterInstanceType": "m3.xlarge",
    "taskInstanceType": "m3.xlarge"
}
```

deshabilitar el cifrado en el servidor en las versiones 3.x

Example

Una `EmrCluster` actividad creada con una versión 2.x de Hadoop AWS Data Pipeline habilita el cifrado del lado del servidor de forma predeterminada. Si desea deshabilitar el cifrado en el servidor, debe especificar una acción de arranque en la definición de objeto de clúster.

En el siguiente ejemplo se crea una actividad `EmrCluster` con el cifrado en el servidor deshabilitado:

```
{
   "id":"NoSSEEmrCluster",
   "type":"EmrCluster",
   "hadoopVersion":"2.x",
   "keyPair":"my-key-pair",
   "masterInstanceType":"m3.xlarge",
   "coreInstanceType":"m3.large",
   "coreInstanceCount":"10",
   "taskInstanceType":"m3.large",
   "taskInstanceCount":"10",
   "bootstrapAction":["s3://Region.elasticmapreduce/bootstrap-actions/configure-
hadoop,-e, fs.s3.enableServerSideEncryption=false"]
}
```

deshabilitar el cifrado en el servidor en las versiones 4.x

Example

Debe deshabilitar el cifrado en el servidor mediante un objeto `EmrConfiguration`.

En el siguiente ejemplo se crea una actividad `EmrCluster` con el cifrado en el servidor deshabilitado:

```
  {
     "name": "ReleaseLabelCluster",
     "releaseLabel": "emr-4.1.0",
     "applications": ["spark", "hive", "pig"],
     "id": "myResourceId",
     "type": "EmrCluster",
     "configuration": {
       "ref": "disableSSE"
     }
   },
   {
     "name": "disableSSE",
     "id": "disableSSE",
     "type": "EmrConfiguration",
     "classification": "emrfs-site",
     "property": [{
       "ref": "enableServerSideEncryption"
     }
     ]
   },
   {
     "name": "enableServerSideEncryption",
     "id": "enableServerSideEncryption",
     "type": "Property",
     "key": "fs.s3.enableServerSideEncryption",
     "value": "false"
   }
```

Configure Hadoop KMS y cree zonas de cifrado en HDFS ACLs

Example

Los siguientes objetos se crean ACLs para Hadoop KMS y crean zonas de cifrado y las claves de cifrado correspondientes en HDFS:

```
{
      "name": "kmsAcls",
      "id": "kmsAcls",
      "type": "EmrConfiguration",
      "classification": "hadoop-kms-acls",
```

```
      "property": [
        {"ref":"kmsBlacklist"},
        {"ref":"kmsAcl"}
      ]
    },
    {
      "name": "hdfsEncryptionZone",
      "id": "hdfsEncryptionZone",
      "type": "EmrConfiguration",
      "classification": "hdfs-encryption-zones",
      "property": [
        {"ref":"hdfsPath1"},
        {"ref":"hdfsPath2"}
      ]
    },
    {
      "name": "kmsBlacklist",
      "id": "kmsBlacklist",
      "type": "Property",
      "key": "hadoop.kms.blacklist.CREATE",
      "value": "foo,myBannedUser"
    },
    {
      "name": "kmsAcl",
      "id": "kmsAcl",
      "type": "Property",
      "key": "hadoop.kms.acl.ROLLOVER",
      "value": "myAllowedUser"
    },
    {
      "name": "hdfsPath1",
      "id": "hdfsPath1",
      "type": "Property",
      "key": "/myHDFSPath1",
      "value": "path1_key"
    },
    {
      "name": "hdfsPath2",
      "id": "hdfsPath2",
      "type": "Property",
      "key": "/myHDFSPath2",
      "value": "path2_key"
    }
```

especificar roles de IAM personalizados

Example

De forma predeterminada, AWS Data Pipeline pasa a `DataPipelineDefaultRole` ser la función de servicio Amazon EMR y `DataPipelineDefaultResourceRole` el perfil de EC2 instancia de Amazon para crear recursos en su nombre. Sin embargo, puede crear un rol de servicio Amazon EMR personalizado y un perfil de instancia personalizado y usarlos en su lugar. AWS Data Pipeline debe tener permisos suficientes para crear clústeres mediante el rol personalizado y debe añadirlo AWS Data Pipeline como entidad de confianza.

En el siguiente objeto de ejemplo se especifican los roles personalizados para el clúster de Amazon EMR:

```
{
   "id":"MyEmrCluster",
   "type":"EmrCluster",
   "hadoopVersion":"2.x",
   "keyPair":"my-key-pair",
   "masterInstanceType":"m3.xlarge",
   "coreInstanceType":"m3.large",
   "coreInstanceCount":"10",
   "taskInstanceType":"m3.large",
   "taskInstanceCount":"10",
   "role":"emrServiceRole",
   "resourceRole":"emrInstanceProfile"
}
```

Utilice el EmrCluster recurso en AWS SDK for Java

Example

En el siguiente ejemplo se muestra cómo usar `EmrCluster` y `EmrActivity` para crear un clúster de Amazon EMR 4.x a fin de ejecutar un paso de Spark mediante el SDK de Java:

```
public class dataPipelineEmr4 {

  public static void main(String[] args) {

 AWSCredentials credentials = null;
 credentials = new ProfileCredentialsProvider("/path/to/
AwsCredentials.properties","default").getCredentials();
```

```
 DataPipelineClient dp = new DataPipelineClient(credentials);
 CreatePipelineRequest createPipeline = new
 CreatePipelineRequest().withName("EMR4SDK").withUniqueId("unique");
 CreatePipelineResult createPipelineResult = dp.createPipeline(createPipeline);
 String pipelineId = createPipelineResult.getPipelineId();

 PipelineObject emrCluster = new PipelineObject()
     .withName("EmrClusterObj")
     .withId("EmrClusterObj")
     .withFields(
   new Field().withKey("releaseLabel").withStringValue("emr-4.1.0"),
   new Field().withKey("coreInstanceCount").withStringValue("3"),
   new Field().withKey("applications").withStringValue("spark"),
   new Field().withKey("applications").withStringValue("Presto-Sandbox"),
   new Field().withKey("type").withStringValue("EmrCluster"),
   new Field().withKey("keyPair").withStringValue("myKeyName"),
   new Field().withKey("masterInstanceType").withStringValue("m3.xlarge"),
   new Field().withKey("coreInstanceType").withStringValue("m3.xlarge")
   );

 PipelineObject emrActivity = new PipelineObject()
     .withName("EmrActivityObj")
     .withId("EmrActivityObj")
     .withFields(
   new Field().withKey("step").withStringValue("command-runner.jar,spark-submit,--
executor-memory,1g,--class,org.apache.spark.examples.SparkPi,/usr/lib/spark/lib/spark-
examples.jar,10"),
   new Field().withKey("runsOn").withRefValue("EmrClusterObj"),
   new Field().withKey("type").withStringValue("EmrActivity")
   );

 PipelineObject schedule = new PipelineObject()
     .withName("Every 15 Minutes")
     .withId("DefaultSchedule")
     .withFields(
   new Field().withKey("type").withStringValue("Schedule"),
   new Field().withKey("period").withStringValue("15 Minutes"),
   new Field().withKey("startAt").withStringValue("FIRST_ACTIVATION_DATE_TIME")
   );

 PipelineObject defaultObject = new PipelineObject()
     .withName("Default")
     .withId("Default")
     .withFields(
```

```
    new Field().withKey("failureAndRerunMode").withStringValue("CASCADE"),
    new Field().withKey("schedule").withRefValue("DefaultSchedule"),
    new
  Field().withKey("resourceRole").withStringValue("DataPipelineDefaultResourceRole"),
    new Field().withKey("role").withStringValue("DataPipelineDefaultRole"),
    new Field().withKey("pipelineLogUri").withStringValue("s3://myLogUri"),
    new Field().withKey("scheduleType").withStringValue("cron")
    );

  List<PipelineObject> pipelineObjects = new ArrayList<PipelineObject>();

  pipelineObjects.add(emrActivity);
  pipelineObjects.add(emrCluster);
  pipelineObjects.add(defaultObject);
  pipelineObjects.add(schedule);

  PutPipelineDefinitionRequest putPipelineDefintion = new PutPipelineDefinitionRequest()
      .withPipelineId(pipelineId)
      .withPipelineObjects(pipelineObjects);

  PutPipelineDefinitionResult putPipelineResult =
  dp.putPipelineDefinition(putPipelineDefintion);
  System.out.println(putPipelineResult);

  ActivatePipelineRequest activatePipelineReq = new ActivatePipelineRequest()
      .withPipelineId(pipelineId);
  ActivatePipelineResult activatePipelineRes = dp.activatePipeline(activatePipelineReq);

       System.out.println(activatePipelineRes);
       System.out.println(pipelineId);

     }

}
```

Configurar un clúster de Amazon EMR en una subred privada

Example

Este ejemplo incluye una configuración que lanza el clúster en una subred privada en una VPC. Para obtener más información, consulte Lanzar clústeres de Amazon EMR en una VPC en la Guía de administración de Amazon EMR. Esta configuración es opcional. Puede utilizarla en cualquier canalización que use un objeto `EmrCluster`.

Para lanzar un clúster de Amazon EMR en una subred privada, especifique `SubnetId`, `emrManagedMasterSecurityGroupId`, `emrManagedSlaveSecurityGroupId` y `serviceAccessSecurityGroupId` en su configuración `EmrCluster`.

```
{
  "objects": [
    {
      "output": {
        "ref": "S3BackupLocation"
      },
      "input": {
        "ref": "DDBSourceTable"
      },
      "maximumRetries": "2",
      "name": "TableBackupActivity",
      "step": "s3://dynamodb-emr-#{myDDBRegion}/emr-ddb-storage-handler/2.1.0/emr-
ddb-2.1.0.jar,org.apache.hadoop.dynamodb.tools.DynamoDbExport,#{output.directoryPath},#{input.t
      "id": "TableBackupActivity",
      "runsOn": {
        "ref": "EmrClusterForBackup"
      },
      "type": "EmrActivity",
      "resizeClusterBeforeRunning": "false"
    },
    {
      "readThroughputPercent": "#{myDDBReadThroughputRatio}",
      "name": "DDBSourceTable",
      "id": "DDBSourceTable",
      "type": "DynamoDBDataNode",
      "tableName": "#{myDDBTableName}"
    },
    {
      "directoryPath": "#{myOutputS3Loc}/#{format(@scheduledStartTime, 'YYYY-MM-dd-HH-
mm-ss')}",
      "name": "S3BackupLocation",
      "id": "S3BackupLocation",
      "type": "S3DataNode"
    },
    {
      "name": "EmrClusterForBackup",
      "coreInstanceCount": "1",
      "taskInstanceCount": "1",
      "taskInstanceType": "m4.xlarge",
```

```
      "coreInstanceType": "m4.xlarge",
      "releaseLabel": "emr-4.7.0",
      "masterInstanceType": "m4.xlarge",
      "id": "EmrClusterForBackup",
      "subnetId": "#{mySubnetId}",
      "emrManagedMasterSecurityGroupId": "#{myMasterSecurityGroup}",
      "emrManagedSlaveSecurityGroupId": "#{mySlaveSecurityGroup}",
      "serviceAccessSecurityGroupId": "#{myServiceAccessSecurityGroup}",
      "region": "#{myDDBRegion}",
      "type": "EmrCluster",
      "keyPair": "user-key-pair"
    },
    {
      "failureAndRerunMode": "CASCADE",
      "resourceRole": "DataPipelineDefaultResourceRole",
      "role": "DataPipelineDefaultRole",
      "pipelineLogUri": "#{myPipelineLogUri}",
      "scheduleType": "ONDEMAND",
      "name": "Default",
      "id": "Default"
    }
  ],
  "parameters": [
    {
      "description": "Output S3 folder",
      "id": "myOutputS3Loc",
      "type": "AWS::S3::ObjectKey"
    },
    {
      "description": "Source DynamoDB table name",
      "id": "myDDBTableName",
      "type": "String"
    },
    {
      "default": "0.25",
      "watermark": "Enter value between 0.1-1.0",
      "description": "DynamoDB read throughput ratio",
      "id": "myDDBReadThroughputRatio",
      "type": "Double"
    },
    {
      "default": "us-east-1",
      "watermark": "us-east-1",
      "description": "Region of the DynamoDB table",
```

```
      "id": "myDDBRegion",
      "type": "String"
    }
  ],
  "values": {
     "myDDBRegion": "us-east-1",
      "myDDBTableName": "ddb_table",
      "myDDBReadThroughputRatio": "0.25",
      "myOutputS3Loc": "s3://s3_path",
      "mySubnetId": "subnet_id",
      "myServiceAccessSecurityGroup":  "service access security group",
      "mySlaveSecurityGroup": "slave security group",
      "myMasterSecurityGroup": "master security group",
      "myPipelineLogUri": "s3://s3_path"
  }
}
```

Asociar volúmenes de EBS a los nodos del clúster

Example

Puede asociar volúmenes de EBS a cualquier tipo de nodo en el clúster de EMR dentro de la canalización. Para asociar volúmenes de EBS a los nodos, utilice `coreEbsConfiguration`, `masterEbsConfiguration` y `TaskEbsConfiguration` en su configuración `EmrCluster`.

Este ejemplo del clúster Amazon EMR utiliza volúmenes de Amazon EBS para sus nodos de tarea, maestro y principal. Para más información, consulte Volúmenes de Amazon EBS en Amazon EMR en la Guía de administración de Amazon EMR.

Estas configuraciones son opcionales. Puede utilizarlas en cualquier canalización que use un objeto `EmrCluster`.

En la canalización, haga clic en la configuración del objeto `EmrCluster`, seleccione Master EBS Configuration, (Configuración de EBS maestra) Core EBS Configuration, (Configuración de EBS principal) Task EBS Configuration (Configuración de EBS de tareas) y especifique los detalles de configuración de modo similar a como se muestra en el siguiente ejemplo.

```
{
  "objects": [
    {
      "output": {
        "ref": "S3BackupLocation"
```

```
      },
      "input": {
        "ref": "DDBSourceTable"
      },
      "maximumRetries": "2",
      "name": "TableBackupActivity",
      "step": "s3://dynamodb-emr-#{myDDBRegion}/emr-ddb-storage-handler/2.1.0/emr-
ddb-2.1.0.jar,org.apache.hadoop.dynamodb.tools.DynamoDbExport,#{output.directoryPath},#{input.t
      "id": "TableBackupActivity",
      "runsOn": {
        "ref": "EmrClusterForBackup"
      },
      "type": "EmrActivity",
      "resizeClusterBeforeRunning": "false"
    },
    {
      "readThroughputPercent": "#{myDDBReadThroughputRatio}",
      "name": "DDBSourceTable",
      "id": "DDBSourceTable",
      "type": "DynamoDBDataNode",
      "tableName": "#{myDDBTableName}"
    },
    {
      "directoryPath": "#{myOutputS3Loc}/#{format(@scheduledStartTime, 'YYYY-MM-dd-HH-
mm-ss')}",
      "name": "S3BackupLocation",
      "id": "S3BackupLocation",
      "type": "S3DataNode"
    },
    {
      "name": "EmrClusterForBackup",
      "coreInstanceCount": "1",
      "taskInstanceCount": "1",
      "taskInstanceType": "m4.xlarge",
      "coreInstanceType": "m4.xlarge",
      "releaseLabel": "emr-4.7.0",
      "masterInstanceType": "m4.xlarge",
      "id": "EmrClusterForBackup",
      "subnetId": "#{mySubnetId}",
      "emrManagedMasterSecurityGroupId": "#{myMasterSecurityGroup}",
      "emrManagedSlaveSecurityGroupId": "#{mySlaveSecurityGroup}",
      "region": "#{myDDBRegion}",
      "type": "EmrCluster",
      "coreEbsConfiguration": {
```

```
        "ref": "EBSConfiguration"
      },
      "masterEbsConfiguration": {
        "ref": "EBSConfiguration"
      },
      "taskEbsConfiguration": {
        "ref": "EBSConfiguration"
      },
      "keyPair": "user-key-pair"
    },
    {
       "name": "EBSConfiguration",
        "id": "EBSConfiguration",
        "ebsOptimized": "true",
        "ebsBlockDeviceConfig" : [
            { "ref": "EbsBlockDeviceConfig" }
        ],
        "type": "EbsConfiguration"
    },
    {
        "name": "EbsBlockDeviceConfig",
        "id": "EbsBlockDeviceConfig",
        "type": "EbsBlockDeviceConfig",
        "volumesPerInstance" : "2",
        "volumeSpecification" : {
            "ref": "VolumeSpecification"
        }
    },
    {
      "name": "VolumeSpecification",
      "id": "VolumeSpecification",
      "type": "VolumeSpecification",
      "sizeInGB": "500",
      "volumeType": "io1",
      "iops": "1000"
    },
    {
      "failureAndRerunMode": "CASCADE",
      "resourceRole": "DataPipelineDefaultResourceRole",
      "role": "DataPipelineDefaultRole",
      "pipelineLogUri": "#{myPipelineLogUri}",
      "scheduleType": "ONDEMAND",
      "name": "Default",
      "id": "Default"
```

```
    }
  ],
  "parameters": [
    {
      "description": "Output S3 folder",
      "id": "myOutputS3Loc",
      "type": "AWS::S3::ObjectKey"
    },
    {
      "description": "Source DynamoDB table name",
      "id": "myDDBTableName",
      "type": "String"
    },
    {
      "default": "0.25",
      "watermark": "Enter value between 0.1-1.0",
      "description": "DynamoDB read throughput ratio",
      "id": "myDDBReadThroughputRatio",
      "type": "Double"
    },
    {
      "default": "us-east-1",
      "watermark": "us-east-1",
      "description": "Region of the DynamoDB table",
      "id": "myDDBRegion",
      "type": "String"
    }
  ],
  "values": {
     "myDDBRegion": "us-east-1",
      "myDDBTableName": "ddb_table",
      "myDDBReadThroughputRatio": "0.25",
      "myOutputS3Loc": "s3://s3_path",
      "mySubnetId": "subnet_id",
      "mySlaveSecurityGroup": "slave security group",
      "myMasterSecurityGroup": "master security group",
      "myPipelineLogUri": "s3://s3_path"
  }
}
```

Véase también

- EmrActivity

HttpProxy

HttpProxy le permite configurar su propio proxy y hacer que Task Runner acceda al AWS Data Pipeline servicio a través de él. No es necesario configurar una Task Runner en ejecución con esta información.

Ejemplo de HttpProxy entrada TaskRunner

En la siguiente definición de canalización se muestra un objeto `HttpProxy`:

```
{
  "objects": [
    {
      "schedule": {
        "ref": "Once"
      },
      "pipelineLogUri": "s3://myDPLogUri/path",
      "name": "Default",
      "id": "Default"
    },
    {
      "name": "test_proxy",
      "hostname": "hostname",
      "port": "port",
      "username": "username",
      "*password": "password",
      "windowsDomain": "windowsDomain",
      "type": "HttpProxy",
      "id": "test_proxy",
    },
    {
      "name": "ShellCommand",
      "id": "ShellCommand",
      "runsOn": {
        "ref": "Resource"
      },
      "type": "ShellCommandActivity",
      "command": "echo 'hello world' "
    },
    {
      "period": "1 day",
      "startDateTime": "2013-03-09T00:00:00",
      "name": "Once",
```

```
      "id": "Once",
      "endDateTime": "2013-03-10T00:00:00",
      "type": "Schedule"
    },
    {
      "role": "dataPipelineRole",
      "httpProxy": {
        "ref": "test_proxy"
      },
      "actionOnResourceFailure": "retrynone",
      "maximumRetries": "0",
      "type": "Ec2Resource",
      "terminateAfter": "10 minutes",
      "resourceRole": "resourceRole",
      "name": "Resource",
      "actionOnTaskFailure": "terminate",
      "securityGroups": "securityGroups",
      "keyPair": "keyPair",
      "id": "Resource",
      "region": "us-east-1"
    }
  ],
  "parameters": []
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
hostname	El host proxy que usarán los clientes para conectarse a los servicios de AWS.	Cadena
puerto	El puerto del host proxy que usarán los clientes para conectarse a los servicios de AWS.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo,

Campos opcionales	Description (Descripción)	Tipo de slot
		«parent»: {"ref»:» myBaseObject Id "}
*password	Contraseña de proxy.	Cadena
s3 NoProxy	Deshabilite el proxy HTTP cuando se conecte a Amazon S3	Booleano
nombre de usuario	Nombre de usuario de proxy.	Cadena
windowsDomain	windowsDomain	Cadena
windowsWorkgroup	El nombre de grupo de trabajo de Windows para el proxy NTLM.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Condiciones previas

Los siguientes son los objetos de AWS Data Pipeline condición previa:

Objects

- DBDataDynamo existe
- Dynamo existe DBTable
- Existe
- S3 KeyExists
- S3 PrefixNotEmpty
- ShellCommandPrecondition

DBDataDynamo existe

Una condición previa para comprobar que los datos existen en una tabla de DynamoDB.

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
rol	Especifica el rol que se va a usar para ejecutar la condición previa.	Cadena
tableName	Tabla de DynamoDB que se comprobará.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio estableci do.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
failureAndRerunModo	Describe el comportamiento del nodo del consumidor cuando las dependencias producen un error o se vuelven a ejecutar.	Enumeración
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
preconditionTimeout	Período desde el comienzo después del cual la condición previa se marca como fallida si aún no se ha satisfecho.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
reportProgressTimeout	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
currentRetryCount	Número de veces que se probó la condición previa en este intento.	Cadena
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
lastRetryTime	Última vez que se probó la condición previa en este intento.	Cadena
nodo	El nodo para el que se está realizando esta condición previa.	Objeto de referencia, por ejemplo, «node»: {"ref»:» myRunnabl eObject Id "}
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Dynamo existe DBTable

Una condición previa para comprobar que la tabla de DynamoDB existe.

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
rol	Especifica el rol que se va a usar para ejecutar la condición previa.	Cadena
tableName	Tabla de DynamoDB que se comprobará.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio estableci do.	Periodo
failureAndRerunModo	Describe el comportamiento del nodo del consumidor cuando las dependencias producen un error o se vuelven a ejecutar.	Enumeración
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}

Campos opcionales	Description (Descripción)	Tipo de slot
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
preconditionTimeout	Período desde el comienzo después del cual la condición previa se marca como fallida si aún no se ha satisfecho.	Periodo
reportProgressTimeout	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referenci a, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnabl eObject Id "}
currentRetryCount	Número de veces que se probó la condición previa en este intento.	Cadena
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
lastRetryTime	Última vez que se probó la condición previa en este intento.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
nodo	El nodo para el que se está realizando esta condición previa.	Objeto de referencia, por ejemplo, «node»: {"ref»:» myRunnabl eObject Id "}
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referenci a, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Existe

Comprueba si existe un objeto del nodo de datos.

> **Note**
>
> Recomendamos que use condiciones previas administradas por el sistema en su lugar. Para obtener más información, consulte Condiciones previas.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto. El objeto `InputData` hace referencia a este objeto, `Ready`, además de a otro objeto que se definiría en el mismo archivo de definición de canalización. `CopyPeriod` es un objeto `Schedule`.

```
{
  "id" : "InputData",
  "type" : "S3DataNode",
  "schedule" : { "ref" : "CopyPeriod" },
  "filePath" : "s3://amzn-s3-demo-bucket/InputData/#{@scheduledStartTime.format('YYYY-
MM-dd-hh:mm')}.csv",
  "precondition" : { "ref" : "Ready" }
},
{
  "id" : "Ready",
  "type" : "Exists"
}
```

Sintaxis

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
	complete dentro del tiempo de inicio estableci do.	
failureAndRerunModo	failureAndRerunMode.	Enumeración
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en ondemand.	Periodo
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
preconditionTimeout	Período desde el comienzo después del cual la condición previa se marca como fallida si aún no se ha satisfecho.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
reportProgressTimeout	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
nodo	nodo.	Objeto de referencia, por ejemplo, «node»: {"ref»:» myRunnabl eObject Id "}
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Véase también

- ShellCommandPrecondition

S3 KeyExists

Comprueba si existe una clave en un nodo de datos de Amazon S3.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto. La condición previa se activará cuando la clave, `s3://amzn-s3-demo-bucket/mykey`, a la que hace referencia el parámetro `s3Key`, existe.

```
{
```

```
"id" : "InputReady",
"type" : "S3KeyExists",
"role" : "test-role",
"s3Key" : "s3://amzn-s3-demo-bucket/mykey"
}
```

También puede utilizar `S3KeyExists` como una condición previa en la segunda canalización que espera a que finalice la primera canalización. Para ello:

1. Escriba un archivo en Amazon S3 tras la finalización de la primera canalización.
2. Cree una condición previa `S3KeyExists` en la segunda canalización.

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
rol	Especifica el rol que se va a usar para ejecutar la condición previa.	Cadena
s3Key	La clave de Amazon S3.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera antes de intentar completar el trabajo remoto una vez más. Si se establece , se intenta de nuevo una actividad remota que no se complete dentro del tiempo de inicio establecido.	Periodo
failureAndRerunModo	failureAndRerunMode.	Enumeración
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
	de programación no está establecido en `ondemand`.	
maximumRetries	Número máximo de intentos que se iniciarán en caso de error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
preconditionTimeout	preconditionTimeout.	Periodo
reportProgressTime out	Tiempo de espera para llamadas sucesivas del trabajo remoto a `reportProgress` . Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos consecutivos.	Periodo

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}
currentRetryCount	Número de veces que se probó la condición previa en este intento.	Cadena
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
lastRetryTime	Última vez que se probó la condición previa en este intento.	Cadena
nodo	El nodo para el que se está realizando esta condición previa.	Objeto de referencia, por ejemplo, «node»: {"ref»:» myRunnabl eObject Id "}
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referenci a, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Véase también

- ShellCommandPrecondition

S3 PrefixNotEmpty

Una condición previa para comprobar que los objetos de Amazon S3 con el prefijo especificado (representado como un URI) están presentes.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto mediante campos obligatorios, opcionales y de expresión.

```
{
  "id" : "InputReady",
  "type" : "S3PrefixNotEmpty",
  "role" : "test-role",
  "s3Prefix" : "#{node.filePath}"
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
rol	Especifica el rol que se va a usar para ejecutar la condición previa.	Cadena
s3Prefix	Prefijo de Amazon S3 para comprobar la existencia de objetos.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio estableci do.	Periodo
failureAndRerunModo	Describe el comportamiento del nodo del consumidor cuando las dependencias producen un error o se vuelven a ejecutar.	Enumeración
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero

Campos opcionales	Description (Descripción)	Tipo de slot
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
preconditionTimeout	Período desde el comienzo después del cual la condición previa se marca como fallida si aún no se ha satisfecho.	Periodo
reportProgressTime out	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referencia, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnableObject Id "}
currentRetryCount	Número de veces que se probó la condición previa en este intento.	Cadena
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
lastRetryTime	Última vez que se probó la condición previa en este intento.	Cadena
nodo	nodo.	Objeto de referencia, por ejemplo, «node»: {"ref»:» myRunnabl eObject Id "}
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referenci a, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Véase también

- ShellCommandPrecondition

ShellCommandPrecondition

Un comando de Unix/Linux shell que se puede ejecutar como condición previa.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto.

```
{
  "id" : "VerifyDataReadiness",
  "type" : "ShellCommandPrecondition",
  "command" : "perl check-data-ready.pl"
}
```

Sintaxis

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
comando	El comando que se va a ejecutar. Este valor y cualquier parámetro asociado debe funcionar	Cadena

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
	en el entorno desde el que se está ejecutando Task Runner.	
scriptUri	Una ruta del URI de Amazon S3 para que se descargue un archivo y se ejecute como comando de shell. Solo debe estar presente un campo de comando o un scriptUri. scriptUri no puede utilizar parámetros; utilice un comando en su lugar.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
attemptStatus	Estado más reciente notificado por la actividad remota.	Cadena
attemptTimeout	Tiempo de espera para que se complete el trabajo remoto. Si se establece, se puede reintentar una actividad remota que no se complete dentro del tiempo de inicio estableci do.	Periodo
failureAndRerunModo	Describe el comportamiento del nodo del consumidor cuando las dependencias producen un error o se vuelven a ejecutar.	Enumeración
lateAfterTimeout	El tiempo transcurrido desde el inicio de la canalización dentro del cual el objeto debe completarse. Solo se activa cuando el tipo de programación no está establecido en `ondemand`.	Periodo

Campos opcionales	Description (Descripción)	Tipo de slot
maximumRetries	Número máximo de reintentos cuando se produce un error.	Entero
onFail	Acción que se debe ejecutar cuando el objeto actual produzca un error.	Objeto de referencia, por ejemplo, «onFail»: {"ref»:» myActionId «}
onLateAction	Acciones que deben iniciarse si un objeto todavía no se ha programado o no se ha completado.	Objeto de referenci a, por ejemplo, "onLateAction«: {" ref»:» myActionId «}
onSuccess	Acción que se debe ejecutar cuando el objeto actual se complete correctamente.	Objeto de referenci a, por ejemplo, «onSuccess»: {"ref»:» myActionId «}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
preconditionTimeout	Período desde el comienzo después del cual la condición previa se marca como fallida si aún no se ha satisfecho.	Periodo
reportProgressTime out	Tiempo de espera para llamadas sucesivas del trabajo remoto a reportProgress. Si se establece, las actividades remotas que no informen de su progreso durante el período especificado pueden considerarse estancadas y, en consecuencia, reintentarse.	Periodo
retryDelay	Duración del tiempo de espera entre dos reintentos.	Periodo
scriptArgument	Argumento que se transfiere al script de shell.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
stderr	La ruta de Amazon S3 que recibe los mensajes de error del sistema redirigidos desde el comando. Si utiliza el campo `runsOn`, esta debe ser una ruta de Amazon S3 debido a la naturaleza transitoria del recurso que ejecuta su actividad. No obstante, si especifica el campo `workerGroup` , se permite una ruta de archivo local.	Cadena
stdout	La ruta de Amazon S3 que recibe la salida redirigida del comando. Si utiliza el campo `runsOn`, esta debe ser una ruta de Amazon S3 debido a la naturaleza transitoria del recurso que ejecuta su actividad. No obstante, si especifica el campo `workerGroup` , se permite una ruta de archivo local.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@activeInstances	Lista de los objetos de instancias activas programados actualmente.	Objeto de referencia, por ejemplo, «ActiveInstances»: {"ref»:» myRunnableObject Id "}
@actualEndTime	La hora a la que finalizó la ejecución de este objeto.	DateTime
@actualStartTime	La hora a la que comenzó la ejecución de este objeto.	DateTime
cancellationReason	El valor de cancellationReason si este objeto se ha cancelado.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@cascadeFailedOn	Descripción de la cadena de dependencia en la que ha fallado el objeto.	Objeto de referenci a, por ejemplo, "cascadeFailedOn«: {" ref»:» myRunnabl eObject Id "}
emrStepLog	Registros de pasos de EMR disponibles únicamente sobre intentos de actividad de EMR.	Cadena
errorId	El valor de errorId si este objeto ha fallado.	Cadena
errorMessage	El valor de errorMessage si este objeto ha fallado.	Cadena
errorStackTrace	El seguimiento de la pila de error si este objeto ha fallado.	Cadena
hadoopJobLog	Los registros de trabajo de Hadoop disponibles sobre intentos de actividades basadas en EMR.	Cadena
hostname	El nombre de host del cliente que recogió el intento de tarea.	Cadena
nodo	El nodo para el que se está realizando esta condición previa.	Objeto de referencia, por ejemplo, «node»: {"ref»:» myRunnabl eObject Id "}
reportProgressTime	La hora más reciente a la que la actividad remota notificó algún progreso.	DateTime
@scheduledEndTime	Hora de finalización programada para el objeto.	DateTime
@scheduledStartTime	Hora de comienzo programada para el objeto.	DateTime

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@status	El estado de este objeto.	Cadena
@version	Versión de la canalización con la que se creó el objeto.	Cadena
@waitingOn	Descripción de la lista de dependencias de la que este objeto está a la espera.	Objeto de referencia, por ejemplo, «WaitingOn»: {"ref»:» myRunnableObject Id "}

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Véase también

- ShellCommandActivity
- Existe

Bases de datos

Los siguientes son los objetos de la AWS Data Pipeline base de datos:

Objects

- JdbcDatabase
- RdsDatabase
- RedshiftDatabase

JdbcDatabase

Define una base de datos JDBC.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto.

```
{
  "id" : "MyJdbcDatabase",
  "type" : "JdbcDatabase",
  "connectionString" : "jdbc:redshift://hostname:portnumber/dbname",
  "jdbcDriverClass" : "com.amazon.redshift.jdbc41.Driver",
  "jdbcDriverJarUri" : "s3://redshift-downloads/drivers/RedshiftJDBC41-1.1.6.1006.jar",
  "username" : "user_name",
  "*password" : "my_password"
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
connectionString	La cadena de conexión JDBC para acceder a la base de datos.	Cadena
jdbcDriverClass	La clase de controlador que se va a cargar antes de establecer la conexión JDBC.	Cadena
*password	La contraseña que se debe suministrar.	Cadena
nombre de usuario	nombre de usuario.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
databaseName	El nombre de la base de datos lógica a la que conectarse.	Cadena
jdbcDriverJarUri	La ubicación en Amazon S3 del archivo JAR del controlador JDBC que se utiliza para conectarse a la base de datos. AWS Data Pipeline debe tener permiso para leer este archivo JAR.	Cadena
jdbcProperties	Pares de la forma A=B que se configurarán como propiedades en conexiones JDBC para esta base de datos.	Cadena
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
	dan lugar a objetos de instancia que ejecutan objetos de intento.	

RdsDatabase

Define una base de datos Amazon RDS.

> **Note**
>
> RdsDatabase no es compatible con Aurora. Use the section called “JdbcDatabase” para Aurora, en su lugar.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto.

```
{
  "id" : "MyRdsDatabase",
  "type" : "RdsDatabase",
  "region" : "us-east-1",
  "username" : "user_name",
  "*password" : "my_password",
  "rdsInstanceId" : "my_db_instance_identifier"
}
```

Para el motor de Oracle, se requiere el campo `jdbcDriverJarUri` y puede especificar el siguiente controlador: `http://www.oracle.com/technetwork/database/features/jdbc/jdbc-drivers-12c-download-1958347.html`. Para el motor de SQL Server, se requiere el campo `jdbcDriverJarUri` y puede especificar el siguiente controlador: `https://www.microsoft.com/en-us/download/details.aspx?displaylang=en&id=11774`. Para los motores de MySQL y PostgreSQL, el campo `jdbcDriverJarUri` es opcional.

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
*password	La contraseña que se debe suministrar.	Cadena
rdsInstanceId	La propiedad `DBInstanceIdentifier` de la instancia de base de datos.	Cadena
nombre de usuario	nombre de usuario.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
databaseName	El nombre de la base de datos lógica a la que conectarse.	Cadena
jdbcDriverJarUri	La ubicación en Amazon S3 del archivo JAR del controlador JDBC que se utiliza para conectarse a la base de datos. AWS Data Pipeline debe tener permiso para leer este archivo JAR. En el caso de los motores MySQL y PostgreSQL, se utiliza el controlador predeterminado si no se especifica este campo, pero puede anular el valor predeterminado utilizando este campo. Para los motores de Oracle y SQL Server, este campo es obligator io.	Cadena
jdbcProperties	Pares de la forma A=B que se configurarán como propiedades en conexiones JDBC para esta base de datos.	Cadena
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}

Campos opcionales	Description (Descripción)	Tipo de slot
region	El código de la región en la que se encuentra la base de datos. Por ejemplo, us-east-1.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

RedshiftDatabase

Define una base de datos Amazon Redshift. `RedshiftDatabase` representa las propiedades de la base de datos que utiliza la canalización.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto.

```
{
  "id" : "MyRedshiftDatabase",
```

```
  "type" : "RedshiftDatabase",
  "clusterId" : "myRedshiftClusterId",
  "username" : "user_name",
  "*password" : "my_password",
  "databaseName" : "database_name"
}
```

De forma predeterminada, el objeto usa el controlador Postgres, que requiere el campo `clusterId`. Para usar el controlador Amazon Redshift, especifique la cadena de conexión de la base de datos Amazon Redshift de la consola de (comienza por “jdbc:redshift:”) en el campo `connectionString` en su lugar.

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
*password	La contraseña que se debe suministrar.	Cadena
nombre de usuario	nombre de usuario.	Cadena

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
clusterId	El identificador que proporcionó el usuario cuando se creó el clúster de Amazon Redshift. Por ejemplo, si el punto de conexión de su clúster de Amazon Redshift es mydb.example.us-east-1.redshift.amazonaws.com, el identificador correcto es `mydb`. En la consola de Amazon Redshift, puede obtener este valor del nombre o identificador del clúster.	Cadena
connectionString	El punto de conexión de JDBC para conectarse a una instancia de Amazon Redshift que es propiedad de una cuenta diferente a la de la	Cadena

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
	canalización. No puede especificar `connectionString` ni `clusterId` .	

Campos opcionales	Description (Descripción)	Tipo de slot
databaseName	El nombre de la base de datos lógica a la que conectarse.	Cadena
jdbcProperties	Pares con el formato A=B que se configuran como propiedades en conexiones JDBC para esta base de datos.	Cadena
parent	Elemento principal del objeto actual del que se heredan los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
region	El código de la región en la que se encuentra la base de datos. Por ejemplo, us-east-1.	Enumeración

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Formatos de los datos

Los siguientes son los objetos AWS Data Pipeline de formato de datos:

Objects

- Formato de los datos CSV
- Formato de los datos personalizado
- Formato Dynamo DBData
- Dinamo DBExport DataFormat
- RegEx Formato de datos
- Formato de datos TSV

Formato de los datos CSV

Un formato de datos delimitado por comas donde el separador de columnas es una coma y el separador de registros es un carácter de nueva línea.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto.

```
{
  "id" : "MyOutputDataType",
  "type" : "CSV",
  "column" : [
```

```
    "Name STRING",
    "Score INT",
    "DateOfBirth TIMESTAMP"
  ]
}
```

Sintaxis

Campos opcionales	Description (Descripción)	Tipo de slot
columna	Nombre de la columna con el tipo de datos especificado por cada campo para los datos descritos por este nodo de datos. Ejemplo: nombre de host STRING. Para varios valores, use nombres de columna y tipos de datos separados por un espacio.	Cadena
escapeChar	Un carácter, por ejemplo "\", que indica al analizador que omita el carácter siguiente.	Cadena
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Formato de los datos personalizado

Un formato de datos personalizado definido por una combinación de un determinado separador de columnas, separador de registros y carácter de escape.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto.

```
{
  "id" : "MyOutputDataType",
  "type" : "Custom",
  "columnSeparator" : ",",
  "recordSeparator" : "\n",
  "column" : [
    "Name STRING",
    "Score INT",
    "DateOfBirth TIMESTAMP"
  ]
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
columnSeparator	Un carácter que indica el final de una columna en un archivo de datos.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
columna	Nombre de la columna con el tipo de datos especificado por cada campo para los datos descritos por este nodo de datos. Ejemplo: nombre de host STRING. Para varios valores, use nombres de columna y tipos de datos separados por un espacio.	Cadena
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
recordSeparator	Un carácter que indica el final de una fila en un archivo de datos; por ejemplo, "\n". Solo se admiten caracteres únicos.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
	dan lugar a objetos de instancia que ejecutan objetos de intento.	

Formato Dynamo DBData

Aplica un esquema a una tabla de DynamoDB para hacerla accesible mediante una consulta de Hive. `DynamoDBDataFormat` se utiliza con un objeto `HiveActivity` y una entrada y salida `DynamoDBDataNode`. `DynamoDBDataFormat` requiere que se especifiquen todas las columnas en la consulta de Hive. A fin de obtener una mayor flexibilidad para especificar determinadas columnas en una consulta de Hive o soporte de Amazon S3, consulte Dinamo DBExport DataFormat.

> **Note**
>
> Los tipos booleanos de DynamoDB no están asignados a los tipos booleanos de Hive. Sin embargo, es posible asignar valores enteros de DynamoDB de 0 o 1 a tipos booleanos de Hive.

Ejemplo

En el siguiente ejemplo se muestra cómo usar `DynamoDBDataFormat` para asignar un esquema a una entrada `DynamoDBDataNode`, que permite a un objeto `HiveActivity` obtener acceso a los datos por columnas con nombres y copiar los datos a una salida `DynamoDBDataNode`.

```
{
  "objects": [
    {
      "id" : "Exists.1",
      "name" : "Exists.1",
      "type" : "Exists"
    },
    {
      "id" : "DataFormat.1",
      "name" : "DataFormat.1",
      "type" : "DynamoDBDataFormat",
      "column" : [
         "hash STRING",
        "range STRING"
```

```
      ]
    },
    {
      "id" : "DynamoDBDataNode.1",
      "name" : "DynamoDBDataNode.1",
      "type" : "DynamoDBDataNode",
      "tableName" : "$INPUT_TABLE_NAME",
      "schedule" : { "ref" : "ResourcePeriod" },
      "dataFormat" : { "ref" : "DataFormat.1" }
    },
    {
      "id" : "DynamoDBDataNode.2",
      "name" : "DynamoDBDataNode.2",
      "type" : "DynamoDBDataNode",
      "tableName" : "$OUTPUT_TABLE_NAME",
      "schedule" : { "ref" : "ResourcePeriod" },
      "dataFormat" : { "ref" : "DataFormat.1" }
    },
    {
      "id" : "EmrCluster.1",
      "name" : "EmrCluster.1",
      "type" : "EmrCluster",
      "schedule" : { "ref" : "ResourcePeriod" },
      "masterInstanceType" : "m1.small",
      "keyPair" : "$KEYPAIR"
    },
    {
      "id" : "HiveActivity.1",
      "name" : "HiveActivity.1",
      "type" : "HiveActivity",
      "input" : { "ref" : "DynamoDBDataNode.1" },
      "output" : { "ref" : "DynamoDBDataNode.2" },
      "schedule" : { "ref" : "ResourcePeriod" },
      "runsOn" : { "ref" : "EmrCluster.1" },
      "hiveScript" : "insert overwrite table ${output1} select * from ${input1} ;"
    },
    {
      "id" : "ResourcePeriod",
      "name" : "ResourcePeriod",
      "type" : "Schedule",
      "period" : "1 day",
      "startDateTime" : "2012-05-04T00:00:00",
      "endDateTime" : "2012-05-05T00:00:00"
    }
```

```
  ]
}
```

Sintaxis

Campos opcionales	Description (Descripción)	Tipo de slot
columna	El nombre de la columna con el tipo de datos especificado por cada campo para los datos descritos por este nodo de datos. Por ejemplo, `hostname STRING`. Para varios valores, use nombres de columna y tipos de datos separados por un espacio.	Cadena
parent	El elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, como «parent»: {"ref»:» myBaseObj ect Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	La versión de canalización utilizada para crear el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	El error al describir el objeto mal estructurado.	Cadena
@pipelineId	El ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
	dan lugar a objetos de instancia que ejecutan objetos de intento.	

Dinamo DBExport DataFormat

Aplica un esquema a una tabla de DynamoDB para hacerla accesible mediante una consulta de Hive. Utilice `DynamoDBExportDataFormat` con un objeto `HiveCopyActivity` y una entrada y salida `DynamoDBDataNode` o `S3DataNode`. `DynamoDBExportDataFormat` tiene los beneficios siguientes:

- Da soporte tanto a DynamoDB como a Amazon S3
- Permite filtrar datos por determinadas columnas en su consulta de Hive
- Exporta todos los atributos desde DynamoDB, incluso si se tiene un esquema disperso

> **Note**
>
> Los tipos booleanos de DynamoDB no están asignados a los tipos booleanos de Hive. Sin embargo, es posible asignar valores enteros de DynamoDB de 0 o 1 a tipos booleanos de Hive.

Ejemplo

En el siguiente ejemplo se muestra cómo usar `HiveCopyActivity` y `DynamoDBExportDataFormat` para copiar datos de un nodo `DynamoDBDataNode` a otro, mientras se filtra en función de una marca temporal.

```
{
  "objects": [
    {
      "id" : "DataFormat.1",
      "name" : "DataFormat.1",
      "type" : "DynamoDBExportDataFormat",
      "column" : "timeStamp BIGINT"
    },
    {
```

```
      "id" : "DataFormat.2",
      "name" : "DataFormat.2",
      "type" : "DynamoDBExportDataFormat"
    },
    {
      "id" : "DynamoDBDataNode.1",
      "name" : "DynamoDBDataNode.1",
      "type" : "DynamoDBDataNode",
      "tableName" : "item_mapped_table_restore_temp",
      "schedule" : { "ref" : "ResourcePeriod" },
      "dataFormat" : { "ref" : "DataFormat.1" }
    },
    {
      "id" : "DynamoDBDataNode.2",
      "name" : "DynamoDBDataNode.2",
      "type" : "DynamoDBDataNode",
      "tableName" : "restore_table",
      "region" : "us_west_1",
      "schedule" : { "ref" : "ResourcePeriod" },
      "dataFormat" : { "ref" : "DataFormat.2" }
    },
    {
      "id" : "EmrCluster.1",
      "name" : "EmrCluster.1",
      "type" : "EmrCluster",
      "schedule" : { "ref" : "ResourcePeriod" },
      "masterInstanceType" : "m1.xlarge",
      "coreInstanceCount" : "4"
    },
    {
      "id" : "HiveTransform.1",
      "name" : "Hive Copy Transform.1",
      "type" : "HiveCopyActivity",
      "input" : { "ref" : "DynamoDBDataNode.1" },
      "output" : { "ref" : "DynamoDBDataNode.2" },
      "schedule" : { "ref" : "ResourcePeriod" },
      "runsOn" : { "ref" : "EmrCluster.1" },
      "filterSql" : "`timeStamp` > unix_timestamp(\"#{@scheduledStartTime}\", \"yyyy-
MM-dd'T'HH:mm:ss\")"
    },
    {
      "id" : "ResourcePeriod",
      "name" : "ResourcePeriod",
      "type" : "Schedule",
```

```
      "period" : "1 Hour",
      "startDateTime" : "2013-06-04T00:00:00",
      "endDateTime" : "2013-06-04T01:00:00"
    }
  ]
}
```

Sintaxis

Campos opcionales	Description (Descripción)	Tipo de slot
columna	Nombre de la columna con el tipo de datos especificado por cada campo para los datos descritos por este nodo de datos. Ej.: hostname STRING	Cadena
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

RegEx Formato de datos

Un formato de datos personalizado definido por una expresión regular.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto.

```
{
  "id" : "MyInputDataType",
  "type" : "RegEx",
  "inputRegEx" : "([^ ]*) ([^ ]*) ([^ ]*) (-|\\[[^\\]]*\\]) ([^ \"]*|\"[^\"]*\") (-|
[0-9]*) (-|[0-9]*)(?: ([^ \"]*|\"[^\"]*\") ([^ \"]*|\"[^\"]*\"))?",
  "outputFormat" : "%1$s %2$s %3$s %4$s %5$s %6$s %7$s %8$s %9$s",
  "column" : [
    "host STRING",
    "identity STRING",
    "user STRING",
    "time STRING",
    "request STRING",
    "status STRING",
    "size STRING",
    "referer STRING",
    "agent STRING"
  ]
}
```

Sintaxis

Campos opcionales	Description (Descripción)	Tipo de slot
columna	Nombre de la columna con el tipo de datos especificado por cada campo para los datos	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
	descritos por este nodo de datos. Ejemplo: nombre de host STRING. Para varios valores, use nombres de columna y tipos de datos separados por un espacio.	
inputRegEx	La expresión regular para analizar un archivo de entrada de S3. inputRegEx proporciona una forma de recuperar columnas de datos relativamente desestructurados de un archivo.	Cadena
outputFormat	Los campos de columnas recuperados por inputRegEx, pero referenciados como %1$s %2$s, con la sintaxis del formateador Java.	Cadena
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» Id "} myBaseObject

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Formato de datos TSV

Un formato de datos delimitado por comas donde el separador de columnas es un tabulador y el separador de registros es un carácter de nueva línea.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto.

```
{
  "id" : "MyOutputDataType",
  "type" : "TSV",
  "column" : [
    "Name STRING",
    "Score INT",
    "DateOfBirth TIMESTAMP"
  ]
}
```

Sintaxis

Campos opcionales	Description (Descripción)	Tipo de slot
columna	Nombre de columna y tipo de dato de los datos que se describen en este nodo de datos. Por ejemplo, "`Name STRING`" indica una columna denominada `Name` con los campos del tipo de dato `STRING`. Separe varios pares de nombre de columna y tipo de dato con comas (tal como se muestra en el ejemplo).	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
columnSeparator	El carácter que separa los campos de una columna de los campos de la siguiente columna. El valor predeterminado es '\t'.	Cadena
escapeChar	Un carácter, por ejemplo "\", que indica al analizador que omita el carácter siguiente.	Cadena
parent	Elemento principal del objeto actual del que se heredan los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
recordSeparator	El carácter que separa registros. El valor predeterminado es '\n'.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia, que ejecutan objetos de intento.	Cadena

Acciones

Los objetos de AWS Data Pipeline acción son los siguientes:

Objects

- SnsAlarm
- Finalizar

SnsAlarm

Envía un mensaje de notificación de Amazon SNS cuando una actividad falla o finaliza correctamente.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto. Los valores de `node.input` y `node.output` proceden de la actividad o el nodo de datos que hace referencia a este objeto en su campo `onSuccess`.

```
{
  "id" : "SuccessNotify",
  "name" : "SuccessNotify",
  "type" : "SnsAlarm",
  "topicArn" : "arn:aws:sns:us-east-1:28619EXAMPLE:ExampleTopic",
  "subject" : "COPY SUCCESS: #{node.@scheduledStartTime}",
  "message" : "Files were copied from #{node.input} to #{node.output}."
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
message	El texto de la notificación de Amazon SNS.	Cadena
rol	El rol de IAM que se debe utilizar para crear la alarma de Amazon SNS.	Cadena
subject	El asunto del mensaje de notificación de Amazon SNS.	Cadena

Campos obligatorios	Description (Descripción)	Tipo de slot
topicArn	El ARN de tema de Amazon SNS de destino para el mensaje.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
nodo	nodo.	Objeto de referencia, por ejemplo, «node»: {"ref»:» myRunnabl eObject Id "}
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
	dan lugar a objetos de instancia que ejecutan objetos de intento.	

Finalizar

Acción que desencadena la cancelación de una actividad, un recurso o un nodo de datos pendientes o inacabados. AWS Data Pipeline intenta poner la actividad, el recurso o el nodo de datos en el estado CANCELADO si no comienza por el `lateAfterTimeout` valor.

No puede finalizar acciones que incluyan recursos `onSuccess`, `OnFail` u `onLateAction`.

Ejemplo

A continuación se muestra un ejemplo de este tipo de objeto. En este ejemplo, el campo `onLateAction` de `MyActivity` contiene una referencia a la acción `DefaultAction1`. Al proporcionar una acción para `onLateAction`, también debe facilitar un valor `lateAfterTimeout` para indicar el período de tiempo desde el inicio programado de la canalización tras el cual la actividad se considera tardía.

```
{
  "name" : "MyActivity",
  "id" : "DefaultActivity1",
  "schedule" : {
    "ref" : "MySchedule"
  },
  "runsOn" : {
    "ref" : "MyEmrCluster"
  },
  "lateAfterTimeout" : "1 Hours",
  "type" : "EmrActivity",
  "onLateAction" : {
    "ref" : "DefaultAction1"
  },
  "step" : [
    "s3://amzn-s3-demo-bucket/myPath/myStep.jar,firstArg,secondArg",
    "s3://amzn-s3-demo-bucket/myPath/myOtherStep.jar,anotherArg"
  ]
},
```

```
{
  "name" : "TerminateTasks",
  "id" : "DefaultAction1",
  "type" : "Terminate"
}
```

Sintaxis

Campos opcionales	Description (Descripción)	Tipo de slot
parent	Elemento principal del objeto actual del que se heredan los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
nodo	nodo.	Objeto de referencia, por ejemplo, «node»: {"ref»:» myRunnableObject Id "}
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
	lugar a objetos de instancia, que ejecutan objetos de intento.	

Schedule

Define los tiempos de un evento programado, como cuando se ejecuta una actividad.

> **Note**
>
> Cuando la hora de inicio de un programa es pasada, se AWS Data Pipeline rellena el proceso y se empieza a programar las ejecuciones inmediatamente a partir de la hora de inicio especificada. Para pruebas o desarrollo, use un intervalo relativamente corto. De lo contrario, AWS Data Pipeline intenta poner en cola y programar todas las ejecuciones de la canalización para ese intervalo. AWS Data Pipeline intenta evitar rellenamientos accidentales si el componente de la canalización `scheduledStartTime` se produjo antes de hace 1 día bloqueando la activación de la canalización.

Ejemplos

A continuación se muestra un ejemplo de este tipo de objeto. Define un programa de cada hora comenzando a las 00:00:00 horas el 01-09-2012 y finalizando a las 00:00:00 horas el 01-10-2012. El primer período finaliza a las 01:00:00 el 01-09-2012.

```
{
  "id" : "Hourly",
  "type" : "Schedule",
  "period" : "1 hours",
  "startDateTime" : "2012-09-01T00:00:00",
  "endDateTime" : "2012-10-01T00:00:00"
}
```

La siguiente canalización comenzará en `FIRST_ACTIVATION_DATE_TIME` y se ejecutará cada hora hasta las 22:00:00 horas el 25-04-2014.

```
{
```

```
    "id": "SchedulePeriod",
    "name": "SchedulePeriod",
    "startAt": "FIRST_ACTIVATION_DATE_TIME",
    "period": "1 hours",
    "type": "Schedule",
    "endDateTime": "2014-04-25T22:00:00"
  }
```

La siguiente canalización comenzará en `FIRST_ACTIVATION_DATE_TIME`, se ejecutará cada hora y se completará tras tres coincidencias.

```
{
    "id": "SchedulePeriod",
    "name": "SchedulePeriod",
    "startAt": "FIRST_ACTIVATION_DATE_TIME",
    "period": "1 hours",
    "type": "Schedule",
    "occurrences": "3"
  }
```

La siguiente canalización comenzará a las 22:00:00 el 25-04-2014, se ejecutará cada hora y finalizará tras tres coincidencias.

```
{
    "id": "SchedulePeriod",
    "name": "SchedulePeriod",
    "startDateTime": "2014-04-25T22:00:00",
    "period": "1 hours",
    "type": "Schedule",
    "occurrences": "3"
  }
```

Bajo demanda mediante el objeto Default

```
{
  "name": "Default",
  "resourceRole": "DataPipelineDefaultResourceRole",
  "role": "DataPipelineDefaultRole",
  "scheduleType": "ondemand"
}
```

Bajo demanda con el objeto Schedule explícito

```
{
  "name": "Default",
  "resourceRole": "DataPipelineDefaultResourceRole",
  "role": "DataPipelineDefaultRole",
  "scheduleType": "ondemand"
},
{
  "name": "DefaultSchedule",
  "type": "Schedule",
  "id": "DefaultSchedule",
  "period": "ONDEMAND_PERIOD",
  "startAt": "ONDEMAND_ACTIVATION_TIME"
},
```

En los siguientes ejemplos se muestra cómo un Schedule se puede heredar del objeto Default, establecer de forma explícita para ese objeto o proporcionar mediante una referencia principal:

Schedule heredado del objeto Default

```
{
  "objects": [
  {
      "id": "Default",
      "failureAndRerunMode":"cascade",
      "resourceRole": "DataPipelineDefaultResourceRole",
      "role": "DataPipelineDefaultRole",
      "pipelineLogUri": "s3://myLogsbucket",
      "scheduleType": "cron",
      "schedule": {
        "ref": "DefaultSchedule"
      }
   },
   {
      "type": "Schedule",
      "id": "DefaultSchedule",
      "occurrences": "1",
      "period": "1 Day",
      "startAt": "FIRST_ACTIVATION_DATE_TIME"
    },
    {
      "id": "A_Fresh_NewEC2Instance",
      "type": "Ec2Resource",
      "terminateAfter": "1 Hour"
```

```
    },
    {
      "id": "ShellCommandActivity_HelloWorld",
      "runsOn": {
        "ref": "A_Fresh_NewEC2Instance"
      },
      "type": "ShellCommandActivity",
      "command": "echo 'Hello World!'"
    }
  ]
}
```

Schedule explícito en el objeto

```
{
  "objects": [
  {
      "id": "Default",
      "failureAndRerunMode":"cascade",
      "resourceRole": "DataPipelineDefaultResourceRole",
      "role": "DataPipelineDefaultRole",
      "pipelineLogUri": "s3://myLogsbucket",
      "scheduleType": "cron"

   },
   {
      "type": "Schedule",
      "id": "DefaultSchedule",
      "occurrences": "1",
      "period": "1 Day",
      "startAt": "FIRST_ACTIVATION_DATE_TIME"
    },
    {
      "id": "A_Fresh_NewEC2Instance",
      "type": "Ec2Resource",
      "terminateAfter": "1 Hour"
    },
    {
      "id": "ShellCommandActivity_HelloWorld",
      "runsOn": {
        "ref": "A_Fresh_NewEC2Instance"
      },
      "schedule": {
```

```
        "ref": "DefaultSchedule"
      },
      "type": "ShellCommandActivity",
      "command": "echo 'Hello World!'"
    }
  ]
}
```

Schedule de la referencia principal

```
{
  "objects": [
  {
      "id": "Default",
      "failureAndRerunMode":"cascade",
      "resourceRole": "DataPipelineDefaultResourceRole",
      "role": "DataPipelineDefaultRole",
      "pipelineLogUri": "s3://myLogsbucket",
      "scheduleType": "cron"

   },
   {
      "id": "parent1",
      "schedule": {
        "ref": "DefaultSchedule"
      }

   },
   {
      "type": "Schedule",
      "id": "DefaultSchedule",
      "occurrences": "1",
      "period": "1 Day",
      "startAt": "FIRST_ACTIVATION_DATE_TIME"
    },
    {
      "id": "A_Fresh_NewEC2Instance",
      "type": "Ec2Resource",
      "terminateAfter": "1 Hour"
    },
    {
      "id": "ShellCommandActivity_HelloWorld",
      "runsOn": {
```

```
        "ref": "A_Fresh_NewEC2Instance"
      },
      "parent": {
        "ref": "parent1"
      },
      "type": "ShellCommandActivity",
      "command": "echo 'Hello World!'"
    }
  ]
}
```

Sintaxis

Campos obligatorios	Description (Descripción)	Tipo de slot
periodo	Con qué frecuencia se debe ejecutar la canalización. El formato es "N [minutos|horas|días|semanas|meses]", donde N es un número seguido por uno de los especificadores de tiempo. Por ejemplo, "15 minutos", ejecuta la canalización cada 15 minutos. El período mínimo es de 15 minutos y el período máximo es de 3 años.	Periodo

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
startAt	La fecha y hora en la que se inician las ejecuciones programadas de la canalización. Un valor válido es FIRST_ACTIVATION_DATE_TIME, que queda obsoleto en favor de la creación de una canalización bajo demanda.	Enumeración

Grupo obligatorio (se requiere uno de los siguientes)	Description (Descripción)	Tipo de slot
startDateTime	La fecha y hora en la que se inician las ejecuciones programadas. Debe usar uno de ellos startDateTime o StartAt, pero no ambos.	DateTime

Campos opcionales	Description (Descripción)	Tipo de slot
endDateTime	La fecha y la hora para finalizar las ejecuciones programadas. Debe ser una fecha y una hora posteriores al valor de startDateTime StartAt. El comportamiento predeterminado es programar ejecuciones hasta que la canalización se cierre.	DateTime
occurrences	El número de veces que se ejecutará la canalización una vez activada. No puedes usar ocurrencias con endDateTime.	Entero
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@firstActivationTime	La hora de creación del objeto.	DateTime
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Utilidades

Los siguientes objetos de utilidad configuran otros objetos de canalización:

Temas

- ShellScriptConfig
- EmrConfiguration
- Propiedad

ShellScriptConfig

Utilízalo con una actividad para ejecutar un script de shell para preActivityTask Config y postActivityTask Config. Este objeto está disponible para HadoopActivityHiveActivity, HiveCopyActivity, y PigActivity. Especifica un URI de S3 y una lista de argumentos para el script.

Ejemplo

A ShellScriptConfig con argumentos:

```
{
   "id" : "ShellScriptConfig_1”,
   "name" : “prescript”,
   "type" : "ShellScriptConfig",
   "scriptUri": “s3://my-bucket/shell-cleanup.sh”,
```

```
  "scriptArgument" : ["arg1","arg2"]
}
```

Sintaxis

Este objeto incluye los siguientes campos.

Campos opcionales	Description (Descripción)	Tipo de slot
parent	Elemento principal del objeto actual del que se heredan los slots.	Objeto de referenci a, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
scriptArgument	Una lista de argumentos que se van a usar con el script de shell.	Cadena
scriptUri	El URI de script en Amazon S3 que se debe descargar y ejecutar.	Cadena

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
	lugar a objetos de instancia, que ejecutan objetos de intento.	

EmrConfiguration

El EmrConfiguration objeto es la configuración utilizada para los clústeres de EMR con la versión 4.0.0 o superior. Las configuraciones (en forma de lista) son un parámetro de la llamada a la RunJobFlow API. La API de configuración de Amazon EMR toma una clasificación y propiedades. AWS Data Pipeline utiliza EmrConfiguration los objetos Property correspondientes para configurar una EmrCluster aplicación como Hadoop, Hive, Spark o Pig en clústeres de EMR lanzados en una ejecución en canalización. Como la configuración solo se puede cambiar para los clústeres nuevos, no puedes proporcionar un EmrConfiguration objeto para los recursos existentes. Para obtener más información, consulte https://docs.aws.amazon.com/ElasticMapReduce/latest/ReleaseGuide/.

Ejemplo

El siguiente objeto de configuración establece las propiedades `io.file.buffer.size` y `fs.s3.block.size` en `core-site.xml`:

```
[
   {
      "classification":"core-site",
      "properties":
      {
         "io.file.buffer.size": "4096",
         "fs.s3.block.size": "67108864"
      }
   }
]
```

La definición de objeto de canalización correspondiente utiliza un EmrConfiguration objeto y una lista de objetos Property en el `property` campo:

```
{
  "objects": [
    {
      "name": "ReleaseLabelCluster",
```

```
      "releaseLabel": "emr-4.1.0",
      "applications": ["spark", "hive", "pig"],
      "id": "ResourceId_I1mCc",
      "type": "EmrCluster",
      "configuration": {
        "ref": "coresite"
      }
    },
    {
      "name": "coresite",
      "id": "coresite",
      "type": "EmrConfiguration",
      "classification": "core-site",
      "property": [{
        "ref": "io-file-buffer-size"
      },
      {
        "ref": "fs-s3-block-size"
      }
      ]
    },
    {
      "name": "io-file-buffer-size",
      "id": "io-file-buffer-size",
      "type": "Property",
      "key": "io.file.buffer.size",
      "value": "4096"
    },
    {
      "name": "fs-s3-block-size",
      "id": "fs-s3-block-size",
      "type": "Property",
      "key": "fs.s3.block.size",
      "value": "67108864"
    }
  ]
}
```

El siguiente ejemplo es una configuración anidada usada para establecer el entorno de Hadoop con la clasificación `hadoop-env`:

```
[
  {
```

```
    "classification": "hadoop-env",
    "properties": {},
    "configurations": [
      {
        "classification": "export",
        "properties": {
          "YARN_PROXYSERVER_HEAPSIZE": "2396"
        }
      }
    ]
  }
]
```

El objeto de definición de la canalización correspondiente que usa esta configuración se muestra a continuación:

```
{
  "objects": [
    {
      "name": "ReleaseLabelCluster",
      "releaseLabel": "emr-4.0.0",
      "applications": ["spark", "hive", "pig"],
      "id": "ResourceId_I1mCc",
      "type": "EmrCluster",
      "configuration": {
        "ref": "hadoop-env"
      }
    },
    {
      "name": "hadoop-env",
      "id": "hadoop-env",
      "type": "EmrConfiguration",
      "classification": "hadoop-env",
      "configuration": {
        "ref": "export"
      }
    },
    {
      "name": "export",
      "id": "export",
      "type": "EmrConfiguration",
      "classification": "export",
      "property": {
```

```
        "ref": "yarn-proxyserver-heapsize"
      }
    },
    {
      "name": "yarn-proxyserver-heapsize",
      "id": "yarn-proxyserver-heapsize",
      "type": "Property",
      "key": "YARN_PROXYSERVER_HEAPSIZE",
      "value": "2396"
    },
  ]
}
```

El siguiente ejemplo modifica una propiedad específica de Hive para un clúster de EMR:

```
{
    "objects": [
        {
            "name": "hivesite",
            "id": "hivesite",
            "type": "EmrConfiguration",
            "classification": "hive-site",
            "property": [
                {
                    "ref": "hive-client-timeout"
                }
            ]
        },
        {
            "name": "hive-client-timeout",
            "id": "hive-client-timeout",
            "type": "Property",
            "key": "hive.metastore.client.socket.timeout",
            "value": "2400s"
        }
    ]
}
```

Sintaxis

Este objeto incluye los siguientes campos.

Campos obligatorios	Description (Descripción)	Tipo de slot
clasificación	Clasificación de la configuración.	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
configuración	Subconfiguración de esta configuración.	Objeto de referencia, por ejemplo, «configuración»: {"ref»:» myEmrConfiguration Id "}
parent	Elemento principal del objeto actual del que se heredarán los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}
propiedad	Propiedad de configuración.	Objeto de referencia, por ejemplo, «propiedad»: {"ref»:» myPropertyId «}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia que ejecutan objetos de intento.	Cadena

Véase también

- EmrCluster
- Propiedad
- Guía de publicación de Amazon EMR

Propiedad

Una propiedad clave-valor única para usar con un EmrConfiguration objeto.

Ejemplo

La siguiente definición de canalización muestra un EmrConfiguration objeto y los objetos Property correspondientes para lanzar un objeto: EmrCluster

```
{
  "objects": [
    {
      "name": "ReleaseLabelCluster",
      "releaseLabel": "emr-4.1.0",
      "applications": ["spark", "hive", "pig"],
      "id": "ResourceId_I1mCc",
      "type": "EmrCluster",
      "configuration": {
        "ref": "coresite"
      }
    },
    {
```

```
      "name": "coresite",
      "id": "coresite",
      "type": "EmrConfiguration",
      "classification": "core-site",
      "property": [{
        "ref": "io-file-buffer-size"
      },
      {
        "ref": "fs-s3-block-size"
      }
      ]
    },
    {
      "name": "io-file-buffer-size",
      "id": "io-file-buffer-size",
      "type": "Property",
      "key": "io.file.buffer.size",
      "value": "4096"
    },
    {
      "name": "fs-s3-block-size",
      "id": "fs-s3-block-size",
      "type": "Property",
      "key": "fs.s3.block.size",
      "value": "67108864"
    }
  ]
}
```

Sintaxis

Este objeto incluye los siguientes campos.

Campos obligatorios	Description (Descripción)	Tipo de slot
clave	key	Cadena
valor	valor	Cadena

Campos opcionales	Description (Descripción)	Tipo de slot
parent	Elemento principal del objeto actual del que se heredan los slots.	Objeto de referencia, por ejemplo, «parent»: {"ref»:» myBaseObject Id "}

Campos de tiempo de ejecución	Description (Descripción)	Tipo de slot
@version	Versión de la canalización con la que se creó el objeto.	Cadena

Campos del sistema	Description (Descripción)	Tipo de slot
@error	Error al describir el objeto mal estructurado.	Cadena
@pipelineId	ID de la canalización a la que pertenece este objeto.	Cadena
@sphere	La esfera de un objeto denota su lugar en el ciclo de vida: los objetos de componente dan lugar a objetos de instancia, que ejecutan objetos de intento.	Cadena

Véase también

- EmrCluster
- EmrConfiguration
- Guía de publicación de Amazon EMR

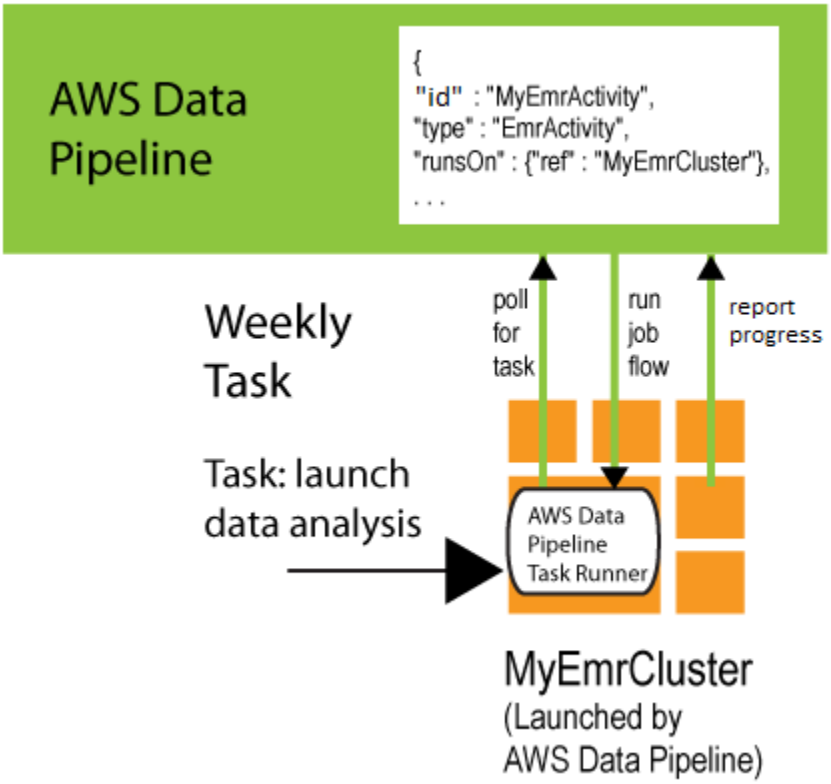
Operación de Task Runner

Task Runner es una aplicación agente de tareas que sondea a AWS Data Pipeline para detectar tareas programadas y las ejecuta en instancias de Amazon EC2, clústeres de Amazon EMR u otros recursos informáticos e informa de su estado mientras lo hace. En función de la aplicación, puede elegir:

- Permitir que AWS Data Pipeline instale y administre una o más aplicaciones de Task Runner automáticamente. Cuando se activa una canalización, se crea automáticamente el objeto `Ec2Instance` o `EmrCluster` predeterminado al que hace referencia el campo runsOn de una actividad. AWS Data Pipeline se encarga de instalar Task Runner en una instancia EC2 o en el nodo maestro de un clúster de EMR. En este caso, AWS Data Pipeline puede hacer automáticamente la mayor parte de la administración de la instancia o el clúster.
- Ejecutar la totalidad o partes de una canalización en recursos que usted administra. Los recursos potenciales incluyen una instancia Amazon EC2 de ejecución prolongada, un clúster de Amazon EMR o un servidor físico. Puede instalar una aplicación de ejecución de tareas (que puede ser Task Runner o un agente de tareas personalizado que usted haya ideado) casi en cualquier lugar, siempre que pueda comunicarse con el servicio web de AWS Data Pipeline. En este caso, usted asume un control casi completo sobre los recursos que se utilizan y cómo se administran, y debe instalar y configurar manualmente Task Runner. Para ello, utilice los procedimientos de esta sección, tal y como se describe en Ejecución de trabajo en recursos existentes mediante Task Runner.

Task Runner sobre recursos gestionados de AWS Data Pipeline

Cuando AWS Data Pipeline lanza y administra un recurso, el servicio web instala automáticamente Task Runner en dicho recurso para procesar tareas en la canalización. Puede especificar un recurso informático (ya sea una instancia Amazon EC2 o un clúster de Amazon EMR) para el campo `runs0n` de un objeto de actividad. Cuando AWS Data Pipeline lanza este recurso, instala Task Runner en dicho recurso y lo configura para procesar todos los objetos de actividad cuyo campo `runs0n` esté establecido en ese recurso. Cuando AWS Data Pipeline termina el recurso, los registros de Task Runner se publican en una ubicación de Amazon S3 antes de que esta se cierre.



Por ejemplo, si utiliza `EmrActivity` en una canalización y especifica un recurso `EmrCluster` en el campo `runsOn`. Cuando AWS Data Pipeline procesa dicha actividad, lanza un clúster de Amazon EMR e instala Task Runner en el nodo maestro. A continuación, este Task Runner procesa las tareas de las actividades cuyo campo `runsOn` esté establecido en ese objeto `EmrCluster`. El siguiente fragmento de una definición de canalización muestra esta relación entre los dos objetos.

```
{
  "id" : "MyEmrActivity",
  "name" : "Work to perform on my data",
  "type" : "EmrActivity",
  "runsOn" : {"ref" : "MyEmrCluster"},
  "preStepCommand" : "scp remoteFiles localFiles",
  "step" : "s3://amzn-s3-demo-bucket/myPath/myStep.jar,firstArg,secondArg",
  "step" : "s3://amzn-s3-demo-bucket/myPath/myOtherStep.jar,anotherArg",
  "postStepCommand" : "scp localFiles remoteFiles",
  "input" : {"ref" : "MyS3Input"},
  "output" : {"ref" : "MyS3Output"}
},
{
  "id" : "MyEmrCluster",
  "name" : "EMR cluster to perform the work",
```

```
  "type" : "EmrCluster",
  "hadoopVersion" : "0.20",
  "keypair" : "myKeyPair",
  "masterInstanceType" : "m1.xlarge",
  "coreInstanceType" : "m1.small",
  "coreInstanceCount" : "10",
  "taskInstanceType" : "m1.small",
  "taskInstanceCount": "10",
  "bootstrapAction" : "s3://elasticmapreduce/libs/ba/configure-hadoop,arg1,arg2,arg3",
  "bootstrapAction" : "s3://elasticmapreduce/libs/ba/configure-other-stuff,arg1,arg2"
}
```

Para obtener información y ejemplos de la ejecución de esta actividad, consulte EmrActivity.

Si tiene varios recursos administrados por AWS Data Pipeline en una canalización, Task Runner se instala en cada uno de ellos y todos ellos sondean a AWS Data Pipeline para detectar las tareas que hay que procesar.

Ejecución de trabajo en recursos existentes mediante Task Runner

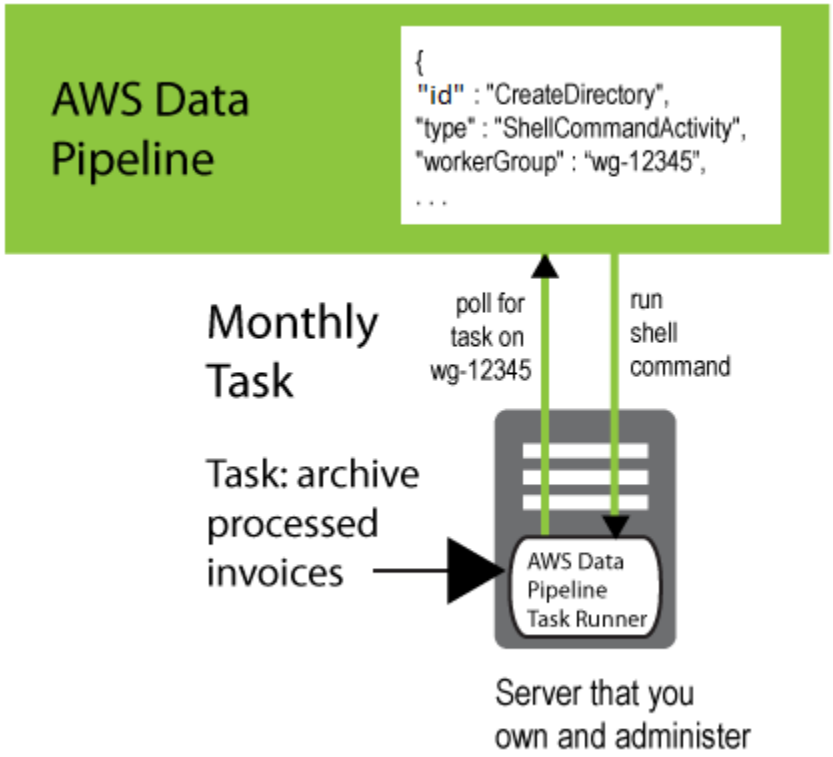
Puede instalar Task Runner en recursos informáticos que administre como, por ejemplo, una instancia Amazon EC2, o una estación de trabajo o un servidor físicos. Task Runner se puede instalar en cualquier lugar, en cualquier sistema operativo o hardware compatible, siempre que pueda comunicarse con el servicio web de AWS Data Pipeline.

Este enfoque puede resultar útil cuando, por ejemplo, desee utilizar AWS Data Pipeline para procesar datos almacenados dentro del firewall de la organización. Si instala Task Runner en un servidor de la red local, puede obtener acceso a la base de datos local de forma segura y, a continuación, sondear a AWS Data Pipeline para detectar la siguiente tarea que se deba ejecutar. Cuando AWS Data Pipeline termina el procesamiento o elimina la canalización, la instancia de Task Runner permanece en ejecución en el recurso informático hasta que se cierra de manera manual. Los registros de Task Runner persisten después de que se haya completado la ejecución de la canalización.

Para utilizar Task Runner en un recurso que administre, primero debe descargar Task Runner y, a continuación, instalarlo en el recurso informático mediante los procedimientos de esta sección.

> **Note**
>
> Solo puede instalar Task Runner en Linux, UNIX o macOS. Task Runner no es compatible con el sistema operativo Windows.
> Para usar Task Runner 2.0, la versión mínima de Java necesaria es 1.7.

Para conectar un Task Runner que haya instalado con las actividades de canalización que debe procesar, añada un campo `workerGroup` al objeto y configure Task Runner para sondear ese valor de grupo de procesos de trabajo. Para ello, transfiera la cadena del grupo de procesos de trabajo en forma de parámetro (por ejemplo, `--workerGroup=wg-12345`) cuando ejecute el archivo JAR de Task Runner.




```
{
  "id" : "CreateDirectory",
  "type" : "ShellCommandActivity",
  "workerGroup" : "wg-12345",
  "command" : "mkdir new-directory"
}
```

Instalación de Task Runner

En esta sección, se explica cómo instalar y configurar Task Runner y sus requisitos previos. La instalación es un proceso manual sencillo.

Para instalar Task Runner

1. Task Runner requiere las versiones de Java 1.6 o 1.8. Para determinar si se encuentra instalado Java y la versión que se está ejecutando, utilice el siguiente comando:

   ```
   java -version
   ```

   Si no tiene Java 1.6 o 1.8 instalado en su equipo, descargue una de estas versiones desde http://www.oracle.com/technetwork/java/index.html. Descargue e instale Java y, a continuación, continúe con el paso siguiente.
2. Descargue `TaskRunner-1.0.jar` desde https://s3.amazonaws.com/datapipeline-us-east-1/us-east-1/software/latest/TaskRunner/TaskRunner-1.0.jar y, a continuación, cópielo en una carpeta del recurso informático de destino. En los clústeres de Amazon EMR que ejecuten tareas `EmrActivity`, instale Task Runner en el nodo maestro del clúster.
3. Al usar Task Runner para conectarse al servicio web AWS Data Pipeline y procesar los comandos, los usuarios necesitan acceder mediante programación a un rol que tenga permisos para crear o administrar canalizaciones de datos. Para obtener más información, consulte Concesión de acceso mediante programación.
4. Task Runner se conecta al servicio web AWS Data Pipeline mediante HTTPS. Si utiliza un recurso de AWS, asegúrese de que HTTPS esté habilitado en la tabla de enrutamiento y la ACL de subred adecuadas. Si utiliza un firewall o un proxy, asegúrese de que el puerto 443 esté abierto.

(Opcional) Otorgar a Task Runner acceso a Amazon RDS

Amazon RDS permite controlar el acceso a las instancias de bases de datos mediante grupos de seguridad de base de datos (grupos de seguridad de base de datos). Un grupo de seguridad de base de datos realiza las mismas funciones que un firewall que controla el acceso de red a su instancia de base de datos. De forma predeterminada, el acceso de red está deshabilitado para sus instancias de base de datos. Debe modificar sus grupos de seguridad de base de datos para permitir que Task

Runner obtenga acceso a sus instancias de Amazon RDS. Task Runner obtiene acceso a Amazon RDS desde la instancia en la que se ejecuta, por lo que las cuentas y los grupos de seguridad que añada a la instancia de Amazon RDS dependen de dónde instale Task Runner.

Para conceder acceso a Task Runner en EC2-Classic

1. Abra la consola de Amazon RDS.
2. En el panel de navegación, elija Instances y seleccione la instancia de base de datos.
3. En Security and Network (Seguridad y redes), seleccione el grupo de seguridad, lo cual abre la página Grupos de seguridad con este grupo de seguridad de base de datos seleccionado. Seleccione el icono de detalles del grupo de seguridad de base de datos.
4. Bajo Security Group Details, cree una regla con los valores adecuados de Connection Type y Details. Estos campos dependen de dónde se esté ejecutando Task Runner, tal y como se describe aquí:

   - `Ec2Resource`
     - Connection Type: `EC2 Security Group`

       Details: *`my-security-group-name`* (el nombre del grupo de seguridad que ha creado para la instancia EC2)
   - `EmrResource`
     - Connection Type: `EC2 Security Group`

       Detalles: `ElasticMapReduce-master`
     - Connection Type: `EC2 Security Group`

       Detalles: `ElasticMapReduce-slave`
   - Su entorno local
     - Connection Type: `CIDR/IP`:

       Details: *`my-ip-address`* (la dirección IP de su equipo o el intervalo de direcciones IP de la red, si el equipo está tras un firewall)
5. Haga clic en Add.

Para conceder acceso a Task Runner en EC2-VPC

1. Abra la consola de Amazon RDS.

2. En el panel de navegación, seleccione Instances (Instancias).
3. Seleccione el icono de detalles de la instancia de base de datos. En Seguridad y redes, abra el enlace al grupo de seguridad, lo que le lleva a la consola de Amazon EC2. Si está utilizando el diseño de consola anterior para los grupos de seguridad, seleccionando el icono que se muestra en la parte superior de la página de la consola para cambiar al nuevo diseño de consola.
4. En la pestaña Inbound (Entrada), elija Edit (Editar), Add Rule (Agregar regla). Especifique el puerto de la base de datos que utilizó al lanzar la instancia de base de datos. El origen depende de dónde se esté ejecutando Task Runner, tal y como se describe aquí:

   - `Ec2Resource`
     - *`my-security-group-id`* (el ID del grupo de seguridad que ha creado para la instancia EC2)
   - `EmrResource`
     - *`master-security-group-id`* (el ID del grupo de seguridad `ElasticMapReduce-master`)
     - *`slave-security-group-id`* (el ID del grupo de seguridad `ElasticMapReduce-slave`)
   - Su entorno local
     - *`ip-address`* (la dirección IP del equipo o el intervalo de direcciones IP de la red, si el equipo está detrás de un firewall)
5. Haga clic en Guardar.

Iniciar Task Runner

En una ventana de comandos nueva establecida en el directorio en el que haya instalado Task Runner, inicie Task Runner, con el siguiente comando.

```
java -jar TaskRunner-1.0.jar --config ~/credentials.json --workerGroup=myWorkerGroup --
region=MyRegion --logUri=s3://amzn-s3-demo-bucket/foldername
```

La opción `--config` apunta a su archivo de credenciales.

La opción `--workerGroup` especifica el nombre del grupo de procesos de trabajo, que debe ser el mismo valor especificado en la canalización para las tareas que va a procesar.

La opción `--region` especifica la región de servicio desde la que extraer las tareas a ejecutar.

La opción `--logUri` se utiliza para enviar los registros comprimidos a una ubicación en Amazon S3.

Cuando Task Runner está activo, imprime la ruta donde se escriben los archivos de registro en la ventana de terminal. A continuación se muestra un ejemplo.

```
Logging to /Computer_Name/.../output/logs
```

Task Runner se debe ejecutar desconectada del shell de inicio de sesión. Si utiliza un terminal de aplicación para conectarse al equipo, puede que tenga que utilizar una utilidad como nohup o screen para evitar que la aplicación Task Runner se cierre al cerrar la sesión. Para obtener más información acerca de las opciones de línea de comandos, consulte Opciones de configuración de Task Runner.

Verificación del registro de Task Runner

La forma más sencilla de verificar si Task Runner está en funcionamiento es comprobar si está escribiendo archivos de registro. Task Runner escribe archivos de registro cada hora en el directorio, `output/logs`, bajo el directorio donde está instalado Task Runner. El nombre del archivo es `Task Runner.log.YYYY-MM-DD-HH`, donde HH va de 00 a 23, en UDT. Para ahorrar espacio de almacenamiento, los archivos de registro de más de ocho horas de antigüedad se comprimen con GZip.

Subprocesos y condiciones previas de Task Runner

Task Runner utiliza un grupo de subprocesos para cada una de las tareas, actividades y condiciones previas. El valor predeterminado de `--tasks` es 2, lo que significa que hay dos subprocesos asignados del grupo de tareas y que cada subproceso sondea el servicio AWS Data Pipeline para detectar tareas nuevas. Por lo tanto, `--tasks` es un atributo de ajuste de rendimiento que se puede utilizar para ayudar a optimizar el rendimiento de la canalización.

En Task Runner se ejecuta la lógica de reintentos de la canalización para condiciones previas. Se asignan dos subprocesos de condición previa para sondear AWS Data Pipeline para detectar objetos de condición previa. Task Runner respeta los campos retryDelay y preconditionTimeout de los objetos de condición previa que usted defina.

En muchos casos, reducir el tiempo de espera de sondeo de condición previa y el número de reintentos sirve para mejorar significativamente el desempeño de la aplicación. Del mismo modo, es posible que las aplicaciones con condiciones previas de ejecución prolongada necesiten que se aumenten los valores de tiempo de espera y reintentos. Para obtener más información acerca de los objetos de condición previa, consulte Condiciones previas.

Opciones de configuración de Task Runner

Estas son las opciones de configuración disponibles en la línea de comandos cuando se lanza Task Runner.

Parámetro de línea de comando	Descripción
`--help`	Ayuda de la línea de comando. Ejemplo:: `Java -jar TaskRunner-1.0.jar --help`
`--config`	Ruta y nombre de archivo del archivo `credentials.json` .
`--accessId`	Su ID de clave de acceso de AWS para que Task Runner lo utilice a la hora de realizar solicitudes. Las opciones `--accessID` y `--secretKey` proporcionan una alternativa al uso de un archivo credentials.json file. Si también se proporciona un archivo `credentials.json` , las opciones `--accessID` y `--secretKey` tienen prioridad.
`--secretKey`	Su clave secreta de AWS para que Task Runner la utilice a la hora de realizar solicitudes. Para obtener más información, consulte `--accessID` .
`--endpoint`	Un punto de enlace es una URL que es el punto de entrada de un servicio web. El punto de enlace de servicio de AWS Data Pipeline en la región donde se realizan las solicitudes. Opcional. En general, es suficiente especificar una región y no es necesario establecer el punto de enlace. Para obtener una lista de las regiones y los puntos de enlace de AWS Data Pipeline, consulte Regiones y puntos de enlace

Parámetro de línea de comando	Descripción
	de AWS Data Pipeline en Referencia general de AWS.
`--workerGroup`	El nombre de un grupo de empleados para los que Task Runner recupera trabajo. Obligatorio. Cuando Task Runner sondea el servicio web, utiliza las credenciales suministradas y el valor de `workerGroup` para seleccionar qué tareas debe recuperar (si procede). Puede utilizar cualquier nombre que sea significativo para usted; el único requisito es que la cadena debe situarse entre Task Runner y las correspon dientes actividades de la canalización. El nombre del grupo de procesos de trabajo está vinculado a una región. Aunque haya nombres de grupo de procesos de trabajo idénticos en otras regiones, Task Runner siempre obtiene tareas de la región especificada en `--region`.
`--taskrunnerId`	El ID que utilizará la aplicación de ejecución de tareas para informar del progreso. Opcional.
`--output`	El directorio de Task Runner para el registro de archivos de salida. Opcional. Los archivos de registro se almacenan en un directorio local hasta que se envían a Amazon S3. Esta opción tiene prioridad sobre el directorio predeterm inado.

Parámetro de línea de comando	Descripción
`--region`	La región que se va a utilizar. Es opcional, pero se recomienda establecer siempre la región. Si no se especifica la región, Task Runner recupera tareas de la región de servicio predeterminada, `us-east-1` . Otras regiones que se admiten son: `eu-west-1` , `ap-northeast-1` , `ap-southeast-2` , `us-west-2` .
`--logUri`	La ruta de destino de Amazon S3 en la que Task Runner hará copias de seguridad de los archivos de registro cada hora. Cuando Task Runner termina, los registros activos del directorio local se envían a la carpeta de destino de Amazon S3.
`--proxyHost`	El host del proxy que utilizan los clientes de Task Runner para conectarse a los servicios de AWS.
`--proxyPort`	El puerto del host proxy que utilizan los clientes de Task Runner para conectarse a los servicios de AWS.
`--proxyUsername`	El nombre de usuario del proxy.
`--proxyPassword`	Contraseña para el proxy.
`--proxyDomain`	windowsDomain
`--proxyWorkstation`	El nombre de estación de trabajo de Windows para el proxy NTLM.

Uso de Task Runner con un Proxy

Si está utilizando un host proxy, puede especificar su configuración cuando invoque a Task Runner o establecer la variable de entorno, HTTPS_PROXY. La variable de entorno utilizada con Task Runner acepta la misma configuración que se utiliza para la interfaz de línea de comando de AWS.

Task Runner y AMI personalizadas

Cuando se especifica un objeto `Ec2Resource` para la canalización, AWS Data Pipeline crea automáticamente una instancia EC2 utilizando una AMI que instala y configura Task Runner. En este caso, se requiere un tipo de instancia compatible con PV. También puede crear una AMI personalizada con Task Runner y, a continuación, especificar el ID de esta AMI mediante el campo `imageId` del objeto `Ec2Resource`. Para obtener más información, consulte Ec2Resource.

Una AMI personalizada debe cumplir los siguientes requisitos para que AWS Data Pipeline pueda utilizarla de forma satisfactoria para Task Runner:

- Crear la AMI en la misma región en la que se ejecutarán las instancias. Para obtener más información, consulte Creación de su propia AMI en la Guía del usuario de Amazon EC2.
- Asegurarse de que el tipo de virtualización de la AMI sea compatible con el tipo de instancia que planee utilizar. Por ejemplo, los tipos de instancia I2 y G2 requieren una AMI HVM y los tipos de instancia T1, C1, M1 y M2 requieren una AMI PV. Para obtener más información, consulte Tipos de virtualización de la AMI de Linux en la Guía del usuario de Amazon EC2.
- Instalar el siguiente software:
  - Linux
  - Bash
  - wget
  - unzip
  - Java 1.6 o 1.8
  - cloud-init
- Crear y configurar una cuenta de usuario denominada `ec2-user`.

Solución de problemas

Cuando hay un problema con AWS Data Pipeline, el síntoma más frecuente es que una canalización no se ejecuta. Puede utilizar los datos que proporcionan la consola y la CLI para identificar el problema y encontrar una solución.

Contenido

Localización de errores en canalizaciones

La consola de AWS Data Pipeline es una herramienta muy práctica para supervisar visualmente el estado de las canalizaciones y localizar fácilmente los errores relacionados con ejecuciones que han fallado o no se han completado.

Para localizar los errores debidos a que las ejecuciones de las canalizaciones han fallado o no se han completado con la consola

1. En la página List Pipelines, si en la columna Status de cualquiera de las instancias de canalización se indica un estado que no es FINISHED, eso significa que la canalización está esperando a que se cumpla alguna condición previa o ha fallado y es necesario solucionar los problemas.
2. En la página List Pipelines (Enumerar canalizaciones), localice la canalización de la instancia y seleccione el triángulo situado a la izquierda de la misma para ampliar los detalles.
3. En la parte inferior de este panel, elija View execution details (Ver detalles de la ejecución); se abrirá el panel Instance summary (Resumen de la instancia) para mostrar los detalles de la instancia seleccionada.
4. En el panel Instance summary (Resumen de la instancia), seleccione el triángulo situado junto a la instancia para ver detalles adicionales sobre ella y, a continuación, elija Details (Detalles), More... (Más...). Si el estado de la instancia seleccionada es FAILED (FALLIDO), el cuadro

de detalles tendrá entradas para el mensaje de error, `errorStackTrace` y otra información. Puede guardar esta información en un archivo. Seleccione Aceptar.

5. En el panel Instance summary (Resumen de la instancia), elija Intentos para ver los detalles de cada fila de intento.
6. Para realizar una acción en una instancia incompleta o con errores, seleccione la casilla de verificación situada junto a la instancia. Esto activa las acciones. A continuación, seleccione una acción (`Rerun|Cancel|Mark Finished`).

Identificación del clúster de Amazon EMR que da servicio a su canalización

Si falla un `EMRCluster` o una `EMRActivity` y la información de error proporcionada por la consola de AWS Data Pipeline, puede identificar el clúster de Amazon EMR que da servicio a la canalización utilizando la consola de Amazon EMR. Esto le ayuda a localizar los registros que Amazon EMR proporciona para obtener más información sobre los errores que se producen.

Para ver información más detallada sobre los errores de Amazon EMR

1. En la consola de AWS Data Pipeline, seleccione el triángulo situado junto a la instancia de canalización para ampliar los detalles de la instancia.
2. Elija View execution details (Ver detalles de la ejecución) y seleccione el triángulo situado junto al componente.
3. En la columna Detalles, elija More... (Más...). Se abrirá la pantalla de información con los detalles del componente. Localice y copie el valor instanceParent en la pantalla, por ejemplo: `@EmrActivityId_xiFDD_2017-09-30T21:40:13`
4. Vaya a la consola de Amazon EMR, busque un clúster cuyo nombre coincida con el valor instanceParent y elija Depurar.

> **Note**
>
> Para que el botón Debug funcione, la definición de la canalización debe tener establecida la opción `enableDebugging` de EmrActivity en `true` y la opción `EmrLogUri` en una ruta válida.

5. Ahora que sabe que el clúster de Amazon EMR contiene el error que provoca el error de la canalización, siga los Consejos para la solución de problemas en la Guía para desarrolladores de Amazon EMR.

Interpretación de los detalles de estado de la canalización

Los distintos niveles de estado que se muestran en la CLI y la consola de AWS Data Pipeline indican el estado de una canalización y sus componentes. El estado de la canalización es simplemente una visión general de una canalización; para ver más información, consulte el estado de los componentes de canalización individuales. Para ello, haga clic en una canalización en la consola o recupere los detalles del componente de la canalización utilizando la CLI.

Códigos de estado

`ACTIVATING`

Se está iniciando el componente o recurso, como una instancia EC2.

`CANCELED`

El componente fue cancelado por un usuario o AWS Data Pipeline antes de que pudiera ejecutarse. Esto puede ocurrir automáticamente cuando se produce un error en un componente o recurso diferente del que depende este componente.

`CASCADE_FAILED`

El componente o recurso se canceló como resultado de un error en cascada en una de sus dependencias, pero es probable que el componente no fuera la fuente original del error.

`DEACTIVATING`

Se está desactivando la canalización.

`FAILED`

El componente o recurso ha detectado un error y ha dejado de funcionar. Cuando se produce un error en un componente o recurso, las cancelaciones y los errores pueden repercutir en cascada en otros componentes que dependen de él.

`FINISHED`

El componente completó el trabajo que se le había asignado.

`INACTIVE`

Se desactivó la canalización.

`PAUSED`

El componente estaba en pausa y no está realizando su trabajo en este momento.

`PENDING`

La canalización está lista para activarse por primera vez.

`RUNNING`

El recurso está en ejecución y listo para recibir trabajo.

`SCHEDULED`

El recurso está programado para ejecutarse.

`SHUTTING_DOWN`

El recurso se cierra después de completar correctamente su trabajo.

`SKIPPED`

El componente omitió los intervalos de ejecución tras la activación de la canalización mediante una marca de tiempo posterior a la programación actual.

`TIMEDOUT`

El recurso superó el umbral de `terminateAfter` y fue detenido por AWS Data Pipeline. Cuando el recurso alcanza este estado, AWS Data Pipeline ignora los valores `actionOnResourceFailure`, `retryDelay` y`retryTimeout` de ese recurso. Este estado solo se aplica a los recursos.

`VALIDATING`

La definición de canalización está siendo validada por AWS Data Pipeline.

`WAITING_FOR_RUNNER`

El componente está esperando a que su cliente trabajador recupere un elemento de trabajo. La relación entre el componente y el cliente del trabajador se controla mediante los campos `runsOn` o `workerGroup` definidos por ese componente.

`WAITING_ON_DEPENDENCIES`

El componente comprueba que se cumplen sus condiciones previas predeterminadas y configuradas por el usuario antes de realizar su trabajo.

Localización de los registros de error

En esta sección, se explica cómo encontrar los diferentes registros que escribe AWS Data Pipeline, que se pueden utilizar para determinar el origen de determinados errores.

Registros de canalización

Le recomendamos que configure las canalizaciones para crear archivos de registro en una ubicación persistente, como en el siguiente ejemplo, en el que se utiliza el campo `pipelineLogUri` en un objeto de una canalización por `Default` para hacer que todos los componentes de la canalización utilicen de manera predeterminada la ubicación del registro de Amazon S3 (puede cambiarlo configurando una ubicación de registro en un componente específico de la canalización).

> **Note**
>
> Task Runner almacena sus registros en otra ubicación de forma predeterminada, que podría no estar disponible cuando termina la canalización y la instancia que ejecuta Task Runner. Para obtener más información, consulte Verificación del registro de Task Runner.

Para configurar la ubicación de registro utilizando la CLI de AWS Data Pipeline en un archivo JSON de canalización, comience el archivo de canalización con el siguiente texto:

```
{ "objects": [
{
  "id":"Default",
  "pipelineLogUri":"s3://amzn-s3-demo-bucket/error_logs"
},
...
```

Después de configurar un directorio de registro de canalización, Task Runner crea una copia de los registros en el directorio, con el mismo formato y los mismos nombres de archivo que se describen en la sección anterior sobre los registros de Task Runner.

Registros de trabajos de Hadoop y de pasos de Amazon EMR

Con cualquier actividad basada en Hadoop como, por ejemplo HadoopActivity, HiveActivity o PigActivity, puede ver los registros de trabajos de Hadoop en la ubicación que se devuelve en el slot de tiempo de ejecución, hadoopJobLog. EmrActivity tiene sus propias características de registro, y esos registros se almacenan utilizando la ubicación seleccionada por Amazon EMR y que devuelve el slot de tiempo de ejecución, emrStepLog. Para obtener más información, consulte Ver archivos de registro en la Guía para desarrolladores de Amazon EMR.

Resolución de problemas comunes

En este tema, se ofrecen diversos síntomas de los problemas de AWS Data Pipeline y los pasos recomendados para solucionarlos.

Contenido

Canalización bloqueada en estado pendiente

Si una canalización aparece en el estado PENDING, indica que una canalización aún no se ha activado o la activación ha fallado debido a un error en su definición. Asegúrese de que no ha recibido ningún error al enviar la canalización mediante la CLI de AWS Data Pipeline o al intentar guardar o activar la canalización mediante la consola de AWS Data Pipeline. Además, compruebe que la canalización tiene una definición válida.

Para ver la definición de la canalización en la pantalla mediante la CLI:

```
aws datapipeline --get-pipeline-definition --pipeline-id df-EXAMPLE_PIPELINE_ID
```

Asegúrese de que la definición de la canalización está completa, compruebe las llaves de cierre, verifique las comas necesarias, compruebe si faltan referencias y si hay otros errores de sintaxis. Lo mejor es utilizar un editor de texto que permita validar visualmente la sintaxis de los archivos JSON.

Componente de la canalización bloqueado en el estado Waiting for Runner

Si la canalización se encuentra en el estado SCHEDULED y una o varias tareas aparecen atascadas en el estado WAITING_FOR_RUNNER, asegúrese de que ha establecido un valor válido en los campos runsOn o workerGroup para esas tareas. Si ambos valores están vacíos o no existen, la tarea no se puede iniciar, ya que no existe una asociación entre la tarea y un trabajador para llevar a cabo las tareas. En esta situación, se ha definido el trabajo, pero no se ha definido qué equipo realiza dicho trabajo. Si procede, compruebe que el valor de workerGroup asignado al componente de la canalización sea exactamente el mismo nombre, con las mismas mayúsculas y minúsculas, que el valor de workerGroup que ha configurado para Task Runner.

> **Note**
>
> Si proporciona un valor runsOn y workerGroup existe, se hace caso omiso de workerGroup.

Otra posible causa de este problema es que el punto de conexión y la clave de acceso proporcionados a Task Runner no sean los mismos que los de la consola de AWS Data Pipeline o el equipo en el que están instaladas las herramientas de la CLI de AWS Data Pipeline. Es posible que haya creado nuevas canalizaciones sin errores visibles, pero Task Runner sondea la posición errónea debido a la diferencia de credenciales o sondea la ubicación correcta con permisos insuficientes para identificar y ejecutar el trabajo especificado en la definición de la canalización.

Componente de la canalización bloqueado en el estado WAITING_ON_DEPENDENCIES

Si la canalización se encuentra en el estado `SCHEDULED` y una o varias tareas aparecen atascadas en el estado `WAITING_ON_DEPENDENCIES`, asegúrese de que se han cumplido las condiciones previas iniciales de la canalización. Si las condiciones previas del primer objeto de la cadena de lógica no se cumplen, ninguno de los objetos que dependen de ese primer objeto podrá salir del estado `WAITING_ON_DEPENDENCIES`.

Por ejemplo, tenga en cuenta el siguiente fragmento de una definición de canalización. En este caso, el objeto `InputData` tiene la condición previa 'Ready', que especifica que los datos deben existir antes de que el objeto InputData se haya completado. Si los datos no existen, el objeto InputData permanece en el estado `WAITING_ON_DEPENDENCIES`, esperando a que estén disponibles los datos especificados por el campo de la ruta. Los objetos que dependen de InputData también permanecen en un estado `WAITING_ON_DEPENDENCIES` esperando a que el objeto InputData alcance el estado `FINISHED`.

```
{
    "id": "InputData",
    "type": "S3DataNode",
    "filePath": "s3://elasticmapreduce/samples/wordcount/wordSplitter.py",
    "schedule":{"ref":"MySchedule"},
    "precondition": "Ready"
},
{
    "id": "Ready",
    "type": "Exists"
...
```

Además, compruebe que los objetos tengan los permisos adecuados para acceder a los datos. En el ejemplo anterior, si la información del campo de las credenciales no tuviera permisos para acceder a los datos especificados en el campo de la ruta, el objeto InputData se bloquearía en el estado `WAITING_ON_DEPENDENCIES`, ya que no puede obtener acceso a los datos especificados por el campo de la ruta, aunque esos datos existan.

También es posible que un recurso que se comunica con Amazon S3 no tenga asociada una dirección IP pública. Por ejemplo, un `Ec2Resource` en una subred pública debe tener asociada una dirección IP pública.

Por último, en determinadas condiciones, las instancias de recursos pueden alcanzar el estado `WAITING_ON_DEPENDENCIES` mucho antes del inicio programado de sus actividades asociadas, lo que puede dar la impresión de que el recurso o la actividad fallan.

No se ejecuta cuando está programada

Compruebe que ha elegido el tipo de programación correcta que determina si su tarea comienza al principio del intervalo de programación (tipo de programación de estilo cron) o al final del intervalo de programación (tipo de programación de serie temporal).

Además, compruebe que ha especificado correctamente las fechas en los objetos de la programación y que los valores de endDateTime y startDateTime están en formato UTC, como en el siguiente ejemplo:

```
{
    "id": "MySchedule",
    "startDateTime": "2012-11-12T19:30:00",
    "endDateTime":"2012-11-12T20:30:00",
    "period": "1 Hour",
    "type": "Schedule"
},
```

Los componentes de la canalización se ejecutan en el orden incorrecto

Es posible que se haya dado cuenta de que las horas de inicio y finalización de la ejecución de los componentes de la canalización están en el orden incorrecto o en una secuencia que no es la esperada. Es importante entender que los componentes de la canalización pueden comenzar a ejecutarse simultáneamente si se cumplen las condiciones previas en el momento del arranque. En otras palabras, los componentes de canalización no se ejecutan de forma secuencial de forma predeterminada; si necesita una orden de ejecución específica, debe controlar el orden de ejecución con condiciones previas y campos `dependsOn`.

Compruebe que está utilizando el campo `dependsOn` que se ha rellenado con una referencia a los componentes de la canalización con los requisitos previos correctos y que todos los indicadores necesarios entre los componentes están presentes para lograr el orden que necesita.

El clúster de EMR falla con un error: el token de seguridad que se incluye en la solicitud no es válido

Verifique sus roles de IAM, políticas y relaciones de confianza, tal y como se describe en Funciones de IAM para AWS Data Pipeline.

Permisos insuficientes para obtener acceso a los recursos

Los permisos que ha establecido en los roles de IAM determinan si AWS Data Pipeline puede tener acceso a los clústeres de EMR y las instancias EC2 para ejecutar sus canalizaciones. Además, IAM proporciona el concepto de las relaciones de confianza que van más allá para permitir la creación de recursos en su nombre. Por ejemplo, al crear una canalización que utiliza una instancia EC2 para ejecutar un comando para trasladar datos, AWS Data Pipeline puede aprovisionar esta instancia EC2 automáticamente. Si tiene problemas, en particular con recursos a los que usted puede tener acceso manualmente, pero no AWS Data Pipeline, verifique sus roles de IAM, políticas y relaciones de confianza, tal y como se describe en Funciones de IAM para AWS Data Pipeline.

Status Code: 400 Error Code: PipelineNotFoundException

Este error significa que los roles predeterminados de IAM podrían no tener los permisos necesarios para que AWS Data Pipeline funcione correctamente. Para obtener más información, consulte Funciones de IAM para AWS Data Pipeline.

La creación de una canalización produce un error del token de seguridad

Recibe el siguiente mensaje de error cuando intenta crear una canalización:

Failed to create pipeline with 'pipeline_name'. Error: UnrecognizedClientException - The security token included in the request is invalid.

No se pueden ver los detalles de la canalización en la consola

El filtro de la canalización de la consola de AWS Data Pipeline se aplica a la fecha de inicio programada de una canalización, sin tener en cuenta la hora a la que se envió la canalización. Es posible enviar una nueva canalización utilizando una fecha de inicio programada que está en el pasado, que es posible que no muestre el filtro de fecha predeterminado. Para ver los detalles de la canalización, cambie el filtro de fecha para asegurarse de que la fecha de inicio de la canalización programada se ajusta al filtro de intervalos de fechas.

Error in remote runner Status Code: 404, AWS Service: Amazon S3

Este error significa que Task Runner no ha podido tener acceso a los archivos en Amazon S3. Verifique lo siguiente:

- Si se han establecido correctamente las credenciales.
- Si el bucket de Amazon S3 al que está intentando acceder existe
- Si está autorizado a acceder al bucket de Amazon S3

Acceso denegado: no está autorizado a realizar la función datapipeline:

En los registros de Task Runner, es posible que vea un error que es similar a lo siguiente:

- ERROR Status Code: 403
- AWS Service: DataPipeline
- AWS Error Code: AccessDenied
- Mensaje de error de AWS: User: arn:aws:sts::XXXXXXXXXXXX:federated-user/i-XXXXXXXX is not authorized to perform: datapipeline:PollForTask.

> **Note**
>
> En este mensaje de error, PollForTask podría sustituirse por nombres de otros permisos de AWS Data Pipeline.

Este mensaje de error indica que el rol de IAM especificado necesita permisos adicionales necesarios para interactuar con AWS Data Pipeline. Asegúrese de que la política de roles de IAM contiene las siguientes líneas, donde PollForTask se sustituye por el nombre del permiso que desea añadir (use * para conceder todos los permisos). Para obtener más información acerca de cómo crear un nuevo rol de IAM; y aplicarle una política, consulte la Administración de políticas de IAM en la guía de Uso de IAM.

```
{
"Action": [ "datapipeline:PollForTask" ],
"Effect": "Allow",
"Resource": ["*"]
```

```
}
```

Las AMI de Amazon EMR más antiguas pueden crear datos falsos para archivos CSV de gran tamaño

En las AMI de Amazon EMR anteriores a la versión a 3.9 (3.8 y anteriores), AWS Data Pipeline utiliza un InputFormat personalizado para leer y escribir archivos CSV para su uso con trabajos de MapReduce. Esto se utiliza cuando el servicio utiliza tablas provisionales para enviar y recibir datos de Amazon S3. Se descubrió un problema con este InputFormat. Este problema consistía en que, al leer registros de archivos CSV de gran tamaño, se podían producir tablas que no se copiaban correctamente. Este problema se solucionó en las versiones de Amazon EMR posteriores. Utilice la AMI 3.9 de Amazon EMR o la versión 4.0.0 o superior de Amazon EMR.

Aumento de los límites de AWS Data Pipeline

Ocasionalmente, es posible que exceda los límites del sistema específicos de AWS Data Pipeline. Por ejemplo, el límite de canalización predeterminado es de 20 canalizaciones con 50 objetos en cada una de ellas. Si descubre que necesita más canalizaciones de las que marca el límite, considere la posibilidad de combinar varias canalizaciones para crear menos con más objetos en cada una de ellas. Para obtener más información sobre los límites de AWS Data Pipeline, consulte AWS Data PipelineLímites de . Sin embargo, si no puede solucionar el problema de los límites utilizando la técnica de la combinación de canalizaciones, aumente su capacidad utilizando el siguiente formulario: Data Pipeline Limit Increase.

AWS Data PipelineLímites de

Para asegurarse de que haya capacidad para todos los usuarios, AWS Data Pipeline impone límites sobre los recursos que se pueden asignar y la frecuencia con la que es posible asignarlos.

Contenido

Límites de la cuenta

Los siguientes límites se aplican a una única cuenta de AWS. Si necesita más capacidad, puede utilizar el formulario de solicitud del Centro de Amazon Web Services Support para aumentarla.

Atributo	Límite	Ajustable
Número de canalizaciones	100	Sí
Número de objetos por canalización	100	Sí
Número de instancias activas por objeto	5	Sí
Número de campos por objeto	50	No
Número de bytes UTF8 por nombre o identificador de campo	256	No
Número de butes UTF8 por campo	10 240	No

Atributo	Límite	Ajustable
Número de bytes UTF8 por objeto	15 360 (incluidos los nombres de campos)	No
Tasa de creación de una instancia a partir de un objeto	1 por cada 5 minutos	No
Reintentos actividad de una canalización	5 por tarea	No
Retraso mínimo entre reintentos	2 minutos	No
Intervalo de programación mínimo	15 minutos	No
Número máximo de acumulaciones en un solo objeto	32	No
Número máximo de instancias EC2 por objeto Ec2Resource	1	No

Límites de llamadas a servicios web

AWS Data Pipeline limita la frecuencia con la que se puede llamar a la API de servicios web. Estos límites también se aplican a los agentes de AWS Data Pipeline que llaman a la API de servicios web en su nombre, como la consola, la CLI y Task Runner.

Los siguientes límites se aplican a una única cuenta de AWS. Esto significa que el uso total de la cuenta, incluido el de los usuarios de , no puede superar estos límites.

La frecuencia de ráfaga permite ahorrar llamadas a servicios web durante los períodos de inactividad y gastarlas en un breve período de tiempo. Por ejemplo, CreatePipeline tiene una frecuencia normal de una llamada cada cinco segundos. Si no llama al servicio durante 30 segundos, habrá ahorrado

seis llamadas. A continuación, podría llamar al servicio web seis veces en un segundo. Dado que este valor está por debajo del límite de ráfaga y mantiene la media de llamadas dentro del límite de frecuencia normal, las llamadas no se ven limitadas.

Si supera el límite de frecuencia y el límite de ráfaga, la llamada al servicio web produce un error y devuelve una excepción de limitación controlada. La implementación predeterminada de un proceso de trabajo, Task Runner, reintenta automáticamente las llamadas al API que fallan con una excepción de limitación controlada. Task Runner tiene un retardo para que los intentos posteriores de llamar al API se produzcan a intervalos cada vez mayores. Si escribe un proceso de trabajo, le recomendamos que implemente un lógica de reintentos similar.

Estos límites se aplican a una única cuenta de AWS.

API	Límite de frecuencia normal	Límite de ráfaga
ActivatePipeline	1 llamada por segundo	100 llamadas
CreatePipeline	1 llamada por segundo	100 llamadas
DeletePipeline	1 llamada por segundo	100 llamadas
DescribeObjects	2 llamadas por segundo	100 llamadas
DescribePipelines	1 llamada por segundo	100 llamadas
GetPipelineDefinition	1 llamada por segundo	100 llamadas
PollForTask	2 llamadas por segundo	100 llamadas
ListPipelines	1 llamada por segundo	100 llamadas
PutPipelineDefinition	1 llamada por segundo	100 llamadas
QueryObjects	2 llamadas por segundo	100 llamadas
ReportTaskProgress	10 llamadas por segundo	100 llamadas
SetTaskStatus	10 llamadas por segundo	100 llamadas
SetStatus	1 llamada por segundo	100 llamadas

API	Límite de frecuencia normal	Límite de ráfaga
ReportTaskRunnerHeartbeat	1 llamada por segundo	100 llamadas
ValidatePipelineDefinition	1 llamada por segundo	100 llamadas

Consideraciones de escalado

AWS Data Pipeline se escala para adaptarse a un número elevado de tareas simultáneas, y es posible configurarlo para crear automáticamente los recursos necesarios para gestionar grandes cargas de trabajo. Usted mantiene el control de estos recursos que se crean automáticamente, y se tienen en cuenta para los límites de recursos de la cuenta de AWS. Por ejemplo, si configura AWS Data Pipeline para que cree automáticamente un clúster de Amazon EMR de 20 nodos para procesar datos y su cuenta de AWS tiene un límite de instancias EC2 establecido en 20, es posible que agote sin darse cuenta de sus recursos de reposición disponibles. Como resultado, tenga en cuenta estas restricciones de recursos en el diseño o aumente los límites de su cuenta en consonancia.

Si necesita más capacidad, puede utilizar el formulario de solicitud del Centro de Amazon Web Services Support para aumentarla.

AWS Data PipelineRecursos

A continuación, se muestran recursos para ayudarle a usar AWS Data Pipeline.

- AWS Data PipelineInformación de producto de : página web principal con información acerca de AWS Data Pipeline.
- AWS Data Pipeline Preguntas frecuentes: trata las 20 preguntas principales formuladas por los desarrolladores acerca de este producto.
- Notas de la versión: proporcionan información general de alto nivel de la versión actual. Destacan de forma específica las características nuevas, las correcciones y los problemas que se conocen.
- Foros de debate de AWS Data Pipeline: un foro de la comunidad para que los desarrolladores traten aspectos técnicos relacionados con Amazon Web Services.

- Clases y talleres: enlaces a cursos basados en roles y especializados, además de laboratorios autoguiados para ayudarlo a desarrollar sus conocimientos sobre AWS y obtener experiencia práctica.
- Centro para desarrolladores de AWS: explore los tutoriales, descargue herramientas y obtenga información sobre los eventos para desarrolladores de AWS.
- Herramientas para desarrolladores de AWS: enlaces a herramientas para desarrolladores, SDK, conjuntos de herramientas de IDE y herramientas de línea de comandos para desarrollar y administrar aplicaciones de AWS.
- Centro de recursos de introducción: aprenda a configurar su Cuenta de AWS, únase a la comunidad de AWS y lance su primera aplicación.
- Tutoriales prácticos: comience con tutoriales paso a paso antes de lanzar su primera aplicación en AWS.
- Documentos técnicos de AWS: enlaces a una lista completa de documentos técnicos de AWS que tratan una gran variedad de temas técnicos, como arquitecturas, seguridad y economía de la nube, escritos por arquitectos de soluciones de AWS o expertos técnicos.
- Centro de AWS Support: punto para crear y administrar los casos de AWS Support. También incluye enlaces a otros recursos útiles como foros, preguntas técnicas frecuentes, estado de los servicios y de AWS Trusted Advisor.
- Soporte: la página web principal para obtener información acerca de Soporte, un canal de soporte individualizado y de respuesta rápida que le ayudará a crear y ejecutar aplicaciones en la nube.

- Contacta con nosotros – Un punto central de contacto para las consultas relacionadas con la facturación AWS, cuentas, eventos, abuso y demás problemas.
- AWSTérminos del sitio de : información detallada sobre nuestros derechos de autor y marca comercial, su cuenta, licencia y acceso al sitio, entre otros temas.

Historial de documentos

Esta documentación hace referencia a la versión del 29 de octubre de 2012 de AWS Data Pipeline.

Cambio	Descripción	Fecha de lanzamiento
AWS Data Pipeline ya no está disponible para los nuevos clientes	AWS Data Pipeline ya no está disponible para los nuevos clientes. Los clientes existentes de AWS Data Pipeline pueden seguir utilizando el servicio con normalidad. Más información	25 de julio de 2025
Se agregó documentación para realizar ciertos procedimientos mediante la CLI de AWS. Se eliminaron los procedimientos relacionados con la consola AWS Data Pipeline.	Para obtener más información, consulte Clonación de la canalización, Visualización de registros de canalización y Cree una canalización a partir de plantillas de Data Pipeline mediante la CLI.	26 de mayo de 2023
Se han añadido más contenido y ejemplos para hacer la migración de AWS Data Pipeline a otros servicios alternativos.	Se actualizó el tema para hacer la migración de AWS Data Pipeline a AWS Glue, AWS Step Functions o Amazon MWAA con más información sobre cada alternativa, mapeos conceptuales entre los servicios y ejemplos. Para obtener más información, consulte Migración de cargas de trabajo desde AWS Data Pipeline.	31 de marzo de 2023
Se agregó información sobre la compatibilidad de AWS Data Pipeline con IMDSv2.	AWS Data Pipeline es compatible con IMDSv2 para los recursos de Amazon EMR y Amazon EC2. Para obtener más información, consulte Protección de datos en AWS Data Pipeline, EmrCluster y Ec2Resource.	16 de diciembre de 2022

Cambio	Descripción	Fecha de lanzamiento
Se agregó un tema para hacer la migración de AWS Data Pipeline a otros servicios alternativos.	Ahora hay otros servicios de AWS que ofrecen a los clientes una mejor experiencia de integración de datos. Puede migrar los casos de uso típicos de AWS Data Pipeline a AWS Glue, AWS Step Functions o Amazon MWAA. Para obtener más información, consulte Migración de cargas de trabajo desde AWS Data Pipeline.	16 de diciembre de 2022
Se actualizaron las listas de instancias de Amazon EC2 y Amazon EMR compatibles. Se ha actualizado la lista de ID de las AMI de HVM (máquina virtual de hardware) que se usan para las instancias.	Se actualizaron las listas de instancias de Amazon EC2 y Amazon EMR compatibles. Para obtener más información, consulte Tipos de instancia compatibles con las actividades de trabajo de canalización. Se ha actualizado la lista de ID de las AMI de HVM (máquina virtual de hardware) que se usan para las instancias. Para obtener más información, Sintaxis y busque `imageId`.	9 de noviembre de 2018

Cambio	Descripción	Fecha de lanzamiento
Se ha añadido la configuración para asociar volúmenes de Amazon EBS a nodos del clúster y para lanzar un clúster de Amazon EMR en una subred privada.	Se han añadido opciones de configuración a un objeto `EMRcluster` . Puede utilizar estas opciones en canalizaciones que utilizan clústeres de Amazon EMR. Utilice los campos `coreEbsConfiguration` , `masterEbsConfiguration` , y `TaskEbsConfiguration` para configurar la asociación de volúmenes de Amazon EBS a los nodos principal, maestro y de tarea del clúster de Amazon EMR. Para obtener más información, consulte Asociar volúmenes de EBS a los nodos del clúster. Utilice los campos `emrManagedMasterSecurityGroupId` , `emrManagedSlaveSecurityGroupId` y `ServiceAccessSecurityGroupId` para configurar un clúster de Amazon EMR en una subred privada. Para obtener más información, consulte Configurar un clúster de Amazon EMR en una subred privada. Para obtener más información acerca de la sintaxis `EMRcluster` , consulte EmrCluster.	19 de abril de 2018
Se ha añadido una lista de instancias de Amazon EC2 y Amazon EMR compatibles.	Se ha añadido la lista de las instancias que AWS Data Pipeline crea de forma predeterminada si no se especifica un tipo de instancia en la definición de la canalización. Se ha añadido una lista de instancias de Amazon EC2 y Amazon EMR compatibles. Para obtener más información, consulte Tipos de instancia compatibles con las actividades de trabajo de canalización.	22 de marzo de 2018

Cambio	Descripción	Fecha de lanzamiento
Se ha añadido compatibilidad con canalizaciones bajo demanda.	• Se ha añadido compatibilidad con canalizaciones bajo demanda, lo que le permite volver a ejecutar una canalización activándola de nuevo.	22 de febrero de 2016
Compatibilidad adicional para bases de datos RDS	• Se han añadido `rdsInstanceId` , `region` y `jdbcDriverJarUri` a RdsDatabase. • Se ha actualizado `database` en SqlActivity para permitir también `RdsDatabase` .	17 de agosto de 2015
Soporte para JDBC adicional	• Se ha actualizado `database` en SqlActivity para permitir también `JdbcDatabase` . • Se ha agregado `jdbcDriverJarUri` a JdbcDatabase • Se ha añadido `initTimeout` a Ec2Resource y EmrCluster. • Se agregó `runAsUser` a Ec2Resource.	7 de julio de 2015
HadoopActivity, zona de disponibilidad y soporte de spot	• Se ha añadido compatibilidad para enviar trabajos paralelos a clústeres de Hadoop. Para obtener más información, consulte HadoopActivity. • Se ha añadido la posibilidad de solicitar instancias de spot con Ec2Resource y EmrCluster. • Se ha añadido la posibilidad de lanzar recursos de `EmrCluster` en una zona de disponibilidad especificada.	1 de junio de 2015
Desactivación de canalizaciones	Se ha añadido compatibilidad para desactivar canalizaciones activas. Para obtener más información, consulte Desactivación de la canalización.	7 de abril de 2015

Cambio	Descripción	Fecha de lanzamiento
Plantillas y consola actualizados	Se agregó contenido nuevo Se ha actualizado el capítulo de introducción para utilizar la plantilla Getting Started with ShellCommandActivity. Para obtener más información, consulte Cree una canalización a partir de plantillas de Data Pipeline mediante la CLI.	25 de noviembre de 2014
Compatibilidad con VPC	Se ha añadido compatibilidad para lanzar recursos en una nube virtual privada (VPC).	12 de marzo de 2014
Compatibilidad de regiones	Se ha añadido compatibilidad con varias regiones de servicio. Además de `us-east-1` , AWS Data Pipeline se admite en las regiones `eu-west-1` , `ap-northeast-1` , `ap-southeast-2` y `us-west-2` .	20 de febrero de 2014
Compatibilidad con Amazon Redshift	Se ha añadido compatibilidad con Amazon Redshift en AWS Data Pipeline, incluida una nueva plantilla de consola (Copy to Redshift) y un tutorial para demostrar el uso de la plantilla. Para obtener más información, consulte Copiar datos a Amazon Redshift con AWS Data Pipeline, RedshiftDataNode, RedshiftDatabase y RedshiftCopyActivity.	6 de noviembre de 2013
PigActivity	Se ha añadido PigActivity, que dispone de compatibilidad nativa con Pig. Para obtener más información, consulte PigActivity.	15 de octubre de 2013
Nueva plantilla de consola, actividad y formato de datos	Se ha añadido la plantilla de consola CrossRegion DynamoDB Copy, que incluye los nuevos HiveCopyActivity y DynamoDBExportDataFormat.	21 de agosto de 2013
Errores en cascada y repeticiones de ejecuciones	Se ha añadido información sobre el error en cascada de AWS Data Pipeline y el comportamiento de repetición de ejecuciones. Para obtener más información, consulte Errores en cascada y repeticiones de ejecuciones.	8 de agosto de 2013

Cambio	Descripción	Fecha de lanzamiento
Vídeo de solución de problemas	Se ha añadido el vídeo de solución de problemas básicos de AWS Data Pipeline. Para obtener más información, consulte Solución de problemas.	17 de julio de 2013
Edición de canalizaciones activas	Se ha añadido más información sobre la edición de canalizaciones activas y la repetición de la ejecución de componentes de canalización. Para obtener más información, consulte Edición de la canalización.	17 de julio de 2013
Uso de recursos en diferentes regiones	Se ha añadido más información sobre el uso de los recursos en distintas regiones. Para obtener más información, consulte Uso de una canalización con recursos en varias regiones.	17 de junio de 2013
Estado WAITING_ON_DEPENDENCIES	El estado CHECKING_PRECONDITIONS ha cambiado a WAITING_ON_DEPENDENCIES y se ha añadido el campo en tiempo de ejecución @waitingOn para objetos de canalización.	20 de mayo de 2013
DynamoDBDataFormat	Se ha añadido la plantilla DynamoDBDataFormat.	23 de abril de 2013
Vídeo de registros web de procesos y compatibilidad con instancias de spot	Se ha introducido el vídeo “Procesar registros web con AWS Data Pipeline, Amazon EMR y Hive” y la compatibilidad con instancias de spot de Amazon EC2.	21 de febrero de 2013
	Versión inicial de la Guía del desarrollador de AWS Data Pipeline.	20 de diciembre de 2012